

# Documentación SIIGS/TES



# Contenido

<u>Paquete default Elementos procedurales</u>	2
<u>index.php</u>	2
<u>index.php</u>	3
<u>Paquete default Clases</u>	4
<u>Clase Index</u>	4
<u>Constructor construct</u>	4
<u>Método index</u>	4
<u>Paquete CodeIgniter Elementos procedurales</u>	6
<u>Form_validation.php</u>	6
<u>Pagination.php</u>	7
<u>template.php</u>	8
<u>Paquete CodeIgniter Clases</u>	9
<u>Clase CI_Form_validation</u>	9
<u>Var \$CI</u>	9
<u>Var \$error_string</u>	9
<u>Var \$config_rules</u>	10
<u>Var \$error_array</u>	10
<u>Var \$error_messages</u>	10
<u>Var \$error_prefix</u>	10
<u>Var \$error_suffix</u>	10
<u>Var \$field_data</u>	11
<u>Var \$safe_form_data</u>	11
<u>Constructor construct</u>	11
<u>Método alpha</u>	11
<u>Método alpha_dash</u>	12
<u>Método alpha_numeric</u>	12
<u>Método decimal</u>	12
<u>Método encode_php_tags</u>	13
<u>Método error</u>	13
<u>Método error_string</u>	13
<u>Método exact_length</u>	14
<u>Método greater_than</u>	14
<u>Método integer</u>	15
<u>Método is_natural</u>	15
<u>Método is_natural_no_zero</u>	15
<u>Método is_numeric</u>	16
<u>Método is_unique</u>	16
<u>Método less_than</u>	16
<u>Método matches</u>	17
<u>Método max_length</u>	17
<u>Método min_length</u>	17
<u>Método numeric</u>	18

Método prep_for_form	18
Método prep_url	18
Método regex_match	19
Método required	19
Método run	20
Método set_checkbox	20
Método set_error_delimiters	20
Método set_message	21
Método set_radio	21
Método set_rules	22
Método set_select	22
Método set_value	22
Método strip_image_tags	23
Método valid_base64	23
Método valid_email	24
Método valid_emails	24
Método valid_ip	24
Método xss_clean	25
Método execute	25
Método reduce_array	25
Método reset_post_array	26
Método translate_fieldname	26
Clase CI_Pagination	26
Var \$anchor_class	27
Var \$base_url	27
Var \$cur_page	27
Var \$cur_tag_close	27
Var \$cur_tag_open	27
Var \$display_pages	27
Var \$first_link	27
Var \$first_tag_close	27
Var \$first_tag_open	27
Var \$first_url	27
Var \$full_tag_close	28
Var \$full_tag_open	28
Var \$last_link	28
Var \$last_tag_close	28
Var \$last_tag_open	28
Var \$next_link	28
Var \$next_tag_close	28
Var \$next_tag_open	28
Var \$num_links	28
Var \$num_tag_close	28
Var \$num_tag_open	28
Var \$page_query_string	28
Var \$per_page	28
Var \$prefix	28
Var \$prev_link	29
Var \$prev_tag_close	29

<u>Var \$prev tag open</u>	29
<u>Var \$query string segment</u>	29
<u>Var \$suffix</u>	29
<u>Var \$total rows</u>	29
<u>Var \$uri segment</u>	29
<u>Var \$use page numbers</u>	29
<u>Constructor construct</u>	29
<u>Método create links</u>	29
<u>Método initialize</u>	30
<u>Clase CI Template</u>	30
<u>Var \$CI</u>	31
<u>Var \$config</u>	31
<u>Var \$css</u>	31
<u>Var \$js</u>	31
<u>Var \$master</u>	31
<u>Var \$output</u>	31
<u>Var \$parser</u>	31
<u>Var \$parser method</u>	31
<u>Var \$parse template</u>	31
<u>Var \$regions</u>	31
<u>Var \$template</u>	31
<u>Constructor CI Template</u>	31
<u>Método add css</u>	32
<u>Método add js</u>	32
<u>Método add region</u>	32
<u>Método add template</u>	33
<u>Método empty region</u>	33
<u>Método initialize</u>	33
<u>Método load</u>	34
<u>Método parse view</u>	34
<u>Método render</u>	35
<u>Método set master template</u>	35
<u>Método set parser</u>	35
<u>Método set parser method</u>	36
<u>Método set regions</u>	36
<u>Método set template</u>	36
<u>Método write</u>	37
<u>Método write view</u>	37
<u>Paquete Libreria Elementos procedurales</u>	39
<u>menubuilder.php</u>	39
<u>Paquete Libreria Clases</u>	40
<u>Clase Menubuilder</u>	40
<u>Constructor construct</u>	40
<u>Método build</u>	40
<u>Método crearMenu</u>	41
<u>Método isGranted</u>	41
<u>graph.php</u>	42
<u>obtenercurp.php</u>	43

<a href="#">tree.php</a>	44
<a href="#">Clase Graph</a>	44
<a href="#">Constructor construct</a>	44
<a href="#">Método graph_init</a>	44
<a href="#">Método map</a>	45
<a href="#">Clase Obtenercurp</a>	45
<a href="#">Var \$estados</a>	46
<a href="#">Constructor construct</a>	46
<a href="#">Método calculacurp</a>	47
<a href="#">Método calcular_crp</a>	47
<a href="#">Método curp</a>	48
<a href="#">Clase Tree</a>	48
<a href="#">Constructor construct</a>	48
<a href="#">Método create</a>	49
<a href="#">Paquete Session Elementos procedurales</a>	51
<a href="#">Session.php</a>	51
<a href="#">Paquete Session Clases</a>	52
<a href="#">Clase Session</a>	52
<a href="#">Var \$ci</a>	52
<a href="#">Var \$flashdata_key</a>	52
<a href="#">Var \$sess_expiration</a>	53
<a href="#">Var \$sess_namespace</a>	53
<a href="#">Var \$store</a>	53
<a href="#">Constructor construct</a>	53
<a href="#">Método all_userdata</a>	53
<a href="#">Método flashdata</a>	54
<a href="#">Método is_expired</a>	54
<a href="#">Método keep_flashdata</a>	54
<a href="#">Método sess_create</a>	55
<a href="#">Método sess_destroy</a>	55
<a href="#">Método set_flashdata</a>	55
<a href="#">Método set_userdata</a>	55
<a href="#">Método unset_userdata</a>	56
<a href="#">Método userdata</a>	56
<a href="#">Paquete SIIGS Elementos procedurales</a>	58
<a href="#">ayuda.php</a>	58
<a href="#">accion.php</a>	59
<a href="#">bitacora.php</a>	60
<a href="#">catalogo.php</a>	61
<a href="#">catalogocsv.php</a>	62
<a href="#">catalogo_x_raiz.php</a>	63
<a href="#">cie10.php</a>	64
<a href="#">controlador.php</a>	65
<a href="#">entorno.php</a>	66
<a href="#">errorlog.php</a>	67
<a href="#">grupo.php</a>	68
<a href="#">menu.php</a>	69
<a href="#">permiso.php</a>	70

<a href="#">raiz.php</a>	71
<a href="#">reglavaduna.php</a>	72
<a href="#">usuario.php</a>	73
<b>Paquete SIIGS Clases</b>	
<a href="#">Clase Accion</a>	74
<a href="#">Constructor construct</a>	74
<a href="#">Método delete</a>	74
<a href="#">Método index</a>	75
<a href="#">Método insert</a>	75
<a href="#">Método update</a>	75
<a href="#">Método view</a>	76
<a href="#">Clase Ayuda</a>	76
<a href="#">Constructor construct</a>	76
<a href="#">Método index</a>	76
<a href="#">Clase Bitacora</a>	77
<a href="#">Constructor construct</a>	77
<a href="#">Método index</a>	77
<a href="#">Método validateExistUsuario</a>	78
<a href="#">Método view</a>	78
<a href="#">Clase Catalogo</a>	79
<a href="#">Constructor construct</a>	79
<a href="#">Método checkpk</a>	79
<a href="#">Método checkTypeData</a>	79
<a href="#">Método delete</a>	80
<a href="#">Método index</a>	80
<a href="#">Método insert</a>	80
<a href="#">Método load</a>	81
<a href="#">Método loadupdate</a>	81
<a href="#">Método update</a>	81
<a href="#">Método view</a>	82
<a href="#">Método array unique recursive</a>	82
<a href="#">Clase CatalogoCsv</a>	83
<a href="#">Constructor construct</a>	83
<a href="#">Método ActivaEnCatalogo</a>	83
<a href="#">Método checkpk</a>	84
<a href="#">Método createTableAgeb</a>	84
<a href="#">Método createTableGeo</a>	84
<a href="#">Método createTableHemoGlobina</a>	84
<a href="#">Método createTablePob</a>	85
<a href="#">Método index</a>	85
<a href="#">Método loadupdate</a>	85
<a href="#">Método update</a>	85
<a href="#">Método view</a>	86
<a href="#">Método array unique recursive</a>	86
<a href="#">Clase Catalogo x raiz</a>	87
<a href="#">Constructor construct</a>	87
<a href="#">Método check</a>	87
<a href="#">Método delete</a>	88
<a href="#">Método insert</a>	88

<u>Método view</u>	88
<u>Clase Cie10</u>	89
<u>Constructor construct</u>	89
<u>Método ActivaEnCatalogo</u>	89
<u>Método AgregaEnCatalogo</u>	90
<u>Método index</u>	90
<u>Método insert</u>	90
<u>Método load</u>	91
<u>Método update</u>	91
<u>Método view</u>	92
<u>Método array unique recursive</u>	92
<u>Clase Controlador</u>	93
<u>Constructor construct</u>	93
<u>Método accion</u>	93
<u>Método delete</u>	93
<u>Método getGroupPermissions</u>	94
<u>Método help</u>	94
<u>Método index</u>	94
<u>Método insert</u>	95
<u>Método update</u>	95
<u>Método view</u>	96
<u>Clase Entorno</u>	96
<u>Constructor construct</u>	96
<u>Método delete</u>	96
<u>Método index</u>	97
<u>Método insert</u>	97
<u>Método update</u>	97
<u>Método view</u>	98
<u>Método ExistEntorno</u>	98
<u>Método ExistEntornoUpdate</u>	98
<u>Clase Errorlog</u>	99
<u>Constructor construct</u>	99
<u>Método index</u>	99
<u>Método view</u>	100
<u>Clase Grupo</u>	100
<u>Constructor construct</u>	100
<u>Método delete</u>	101
<u>Método index</u>	101
<u>Método insert</u>	101
<u>Método update</u>	102
<u>Método view</u>	102
<u>Método ifGroupExists</u>	102
<u>Clase Menu</u>	103
<u>Constructor construct</u>	103
<u>Método delete</u>	103
<u>Método index</u>	103
<u>Método insert</u>	104
<u>Método update</u>	104
<u>Método view</u>	105

<u>Clase Permisos</u>	105
<u>Constructor construct</u>	105
<u>Método index</u>	106
<u>Clase Raiz</u>	106
<u>Constructor construct</u>	106
<u>Método createasus</u>	106
<u>Método delete</u>	107
<u>Método getChildrenFromLevel</u>	107
<u>Método getDataKeyValue</u>	108
<u>Método getDataTreeFromId</u>	108
<u>Método getTreeBlock</u>	108
<u>Método index</u>	109
<u>Método iniciarusuario</u>	109
<u>Método insert</u>	109
<u>Método update</u>	110
<u>Método updateasus</u>	110
<u>Método view</u>	110
<u>Clase ReglaVacuna</u>	111
<u>Constructor construct</u>	111
<u>Método delete</u>	111
<u>Método index</u>	112
<u>Método insert</u>	112
<u>Método update</u>	112
<u>Método view</u>	113
<u>Clase Usuario</u>	113
<u>Var \$mas</u>	113
<u>Constructor construct</u>	114
<u>Método automatic_access</u>	114
<u>Método cerrar_etab</u>	114
<u>Método delete</u>	114
<u>Método form_init</u>	114
<u>Método getActivesByGroup</u>	114
<u>Método get_galleta</u>	115
<u>Método get_token</u>	115
<u>Método index</u>	115
<u>Método insert</u>	115
<u>Método load_update</u>	116
<u>Método login</u>	116
<u>Método logout</u>	116
<u>Método remember</u>	116
<u>Método reset</u>	116
<u>Método send_mail</u>	117
<u>Método token</u>	117
<u>Método update</u>	117
<u>Método update_info</u>	118
<u>Método view</u>	118
<u>Método ifUserExists</u>	118
<u>accion_model.php</u>	119
<u>ageb_model.php</u>	120

<a href="#">arbolsegmentacion_model.php</a>	121
<a href="#">bitacora_model.php</a>	122
<a href="#">catalogocsv_model.php</a>	123
<a href="#">catalogo_model.php</a>	124
<a href="#">catalogo_x_raiz_model.php</a>	125
<a href="#">cie10_model.php</a>	126
<a href="#">controladoraccion_model.php</a>	127
<a href="#">controlador_model.php</a>	128
<a href="#">entorno_model.php</a>	129
<a href="#">errorlog_model.php</a>	130
<a href="#">georeferencia_model.php</a>	131
<a href="#">grupo_model.php</a>	132
<a href="#">hemoglobina_model.php</a>	133
<a href="#">menu_model.php</a>	134
<a href="#">permiso_model.php</a>	135
<a href="#">poblacion_model.php</a>	136
<a href="#">raiz_model.php</a>	137
<a href="#">reglavaduna_model.php</a>	138
<a href="#">usuario_model.php</a>	139
<a href="#"><u>Clase Accion_model</u></a>	139
<a href="#">Constructor_construct</a>	139
<a href="#">Método_delete</a>	139
<a href="#">Método_getAll</a>	139
<a href="#">Método_getById</a>	140
<a href="#">Método_getDescripcion</a>	140
<a href="#">Método_getId</a>	140
<a href="#">Método_getMetodo</a>	140
<a href="#">Método_getMsgError</a>	141
<a href="#">Método_getNombre</a>	141
<a href="#">Método_getNumRows</a>	141
<a href="#">Método_insert</a>	141
<a href="#">Método_setDescripcion</a>	142
<a href="#">Método_setId</a>	142
<a href="#">Método_setMetodo</a>	142
<a href="#">Método_setNombre</a>	142
<a href="#">Método_setOffset</a>	143
<a href="#">Método_setRows</a>	143
<a href="#">Método_update</a>	143
<a href="#"><u>Clase Ageb_model</u></a>	144
<a href="#">Constructor_construct</a>	144
<a href="#">Método_getMsgError</a>	144
<a href="#">Método_process</a>	144
<a href="#">Método_searchageb</a>	145
<a href="#">Método_searchUM</a>	145
<a href="#"><u>Clase ArbolSegmentacion_model</u></a>	146
<a href="#">Constructor_construct</a>	146
<a href="#">Método_convertType</a>	146
<a href="#">Método_getById</a>	146
<a href="#">Método_getChildrenFromId</a>	147

<u>Método getChildrenFromLevel</u>	147
<u>Método getCluesFromId</u>	148
<u>Método getDataKeyValue</u>	148
<u>Método getDescripcionByld</u>	148
<u>Método getListChildrenLevel</u>	149
<u>Método getMsgError</u>	149
<u>Método getTree</u>	150
<u>Método getTreeBlock</u>	150
<u>Método getTreeBlockData</u>	150
<u>Método getUMParentsByld</u>	151
<u>Método addSelectedItems</u>	151
<u>Método addSelectedItems</u>	152
<b>Clase Bitacora model</b>	152
<b>Constructor construct</b>	152
<u>Método addFilter</u>	152
<u>Método delete</u>	153
<u>Método deleteByFilter</u>	153
<u>Método getAll</u>	153
<u>Método getByld</u>	154
<u>Método getFecha hora</u>	154
<u>Método getId</u>	154
<u>Método getId controlador accion</u>	155
<u>Método getId usuario</u>	155
<u>Método getMsgError</u>	155
<u>Método getNumRows</u>	155
<u>Método getParametros</u>	155
<u>Método insert</u>	156
<u>Método resetFilter</u>	156
<u>Método setFecha hora</u>	156
<u>Método setId</u>	157
<u>Método setId accion</u>	157
<u>Método setId controlador</u>	157
<u>Método setId controlador accion</u>	157
<u>Método setId usuario</u>	158
<u>Método setParametros</u>	158
<u>Método update</u>	158
<b>Clase CatalogoCsv model</b>	159
<b>Constructor construct</b>	159
<u>Método activaEnCatalogo</u>	159
<u>Método checkPk</u>	159
<u>Método getAll</u>	160
<u>Método getAllData</u>	160
<u>Método getByName</u>	160
<u>Método getCampos</u>	161
<u>Método getId</u>	161
<u>Método getLLave</u>	161
<u>Método getMsgError</u>	161
<u>Método getNombre</u>	162
<u>Método getNumRows</u>	162

<u>Método setCampos</u>	162
<u>Método setId</u>	163
<u>Método setLlave</u>	163
<u>Método setNombre</u>	163
<u>Método setOffset</u>	163
<u>Método setRows</u>	164
<u>Clase Catalogo_model</u>	164
Constructor construct	164
<u>Método checkPk</u>	164
<u>Método checkTypeData</u>	165
<u>Método delete</u>	165
<u>Método getAll</u>	165
<u>Método getAllData</u>	166
<u>Método getByName</u>	166
<u>Método getCampos</u>	166
<u>Método getId</u>	167
<u>Método getLLlave</u>	167
<u>Método getMsgError</u>	167
<u>Método getNombre</u>	167
<u>Método getNumRows</u>	168
<u>Método insert</u>	168
<u>Método setCampos</u>	168
<u>Método setId</u>	169
<u>Método setLlave</u>	169
<u>Método setNombre</u>	169
<u>Método setOffset</u>	169
<u>Método setRows</u>	170
<u>Método updateComentario</u>	170
<u>Clase Catalogo_x_raiz_model</u>	170
Constructor construct	171
<u>Método check</u>	171
<u>Método delete</u>	171
<u>Método getByArbol</u>	171
<u>Método getById</u>	172
<u>Método getByNivel</u>	172
<u>Método getColumnaDescripcion</u>	173
<u>Método getColumnaLlave</u>	173
<u>Método getGrado</u>	173
<u>Método getId</u>	173
<u>Método getIdRaiz</u>	173
<u>Método getMsgError</u>	173
<u>Método getNivel</u>	174
<u>Método getRelacionHijo</u>	174
<u>Método getRelacionPadre</u>	174
<u>Método getRelations</u>	174
<u>Método getTablaCatalogo</u>	175
<u>Método insert</u>	175
<u>Método getColumnaDescripcion</u>	175
<u>Método getColumnaLlave</u>	175

<u>Método setGrado</u>	176
<u>Método setId</u>	176
<u>Método setIdRaiz</u>	176
<u>Método setRelacionHijo</u>	177
<u>Método setRelacionPadre</u>	177
<u>Método setTablaCatalogo</u>	177
<u>Clase Cie10_model</u>	178
Constructor <u>construct</u>	178
<u>Método activaEnCatalogo</u>	178
<u>Método agregaEnCatalogo</u>	178
<u>Método checkPk</u>	179
<u>Método getAll</u>	179
<u>Método getAllData</u>	179
<u>Método getById</u>	180
<u>Método getCatalogoByName</u>	180
<u>Método getData</u>	180
<u>Método getDescripcion</u>	181
<u>Método getId</u>	181
<u>Método getMsgError</u>	181
<u>Método getNumRows</u>	181
<u>Método setDescripcion</u>	182
<u>Método setId</u>	182
<u>Método setOffset</u>	182
<u>Método setRows</u>	183
<u>Método update</u>	183
<u>Clase ControladorAccion_model</u>	183
Constructor <u>construct</u>	183
<u>Método getById</u>	184
<u>Método getId</u>	184
<u>Método getIdByPath</u>	184
<u>Método getMsgError</u>	185
<u>Método setHelp</u>	185
<u>Clase Controlador_model</u>	186
Constructor <u>construct</u>	186
<u>Método accionesUpdate</u>	186
<u>Método delete</u>	186
<u>Método getAccion</u>	187
<u>Método getAcciones</u>	187
<u>Método getAll</u>	187
<u>Método getByEntorno</u>	187
<u>Método getById</u>	188
<u>Método getClase</u>	188
<u>Método getDescripcion</u>	188
<u>Método getId</u>	188
<u>Método getIdEntorno</u>	188
<u>Método getMsgError</u>	189
<u>Método getNombre</u>	189
<u>Método getNumRows</u>	189
<u>Método getPermisos</u>	190

<u>Método insert</u>	190
<u>Método setAccion</u>	190
<u>Método setClase</u>	190
<u>Método setDescripcion</u>	191
<u>Método setId</u>	191
<u>Método setIdEntorno</u>	191
<u>Método setNombre</u>	192
<u>Método setOffset</u>	192
<u>Método setRows</u>	192
<u>Método update</u>	192
<b>Clase Entorno_model</b>	193
<u>Constructor construct</u>	193
<u>Método delete</u>	193
<u>Método getAll</u>	193
<u>Método getById</u>	194
<u>Método getByName</u>	194
<u>Método getDescripcion</u>	194
<u>Método getDirectorio</u>	195
<u>Método getHostname</u>	195
<u>Método getId</u>	195
<u>Método getInfo</u>	195
<u>Método getIp</u>	195
<u>Método getMsgError</u>	195
<u>Método getNombre</u>	196
<u>Método getPermissionsByGroup</u>	196
<u>Método insert</u>	196
<u>Método setDescripcion</u>	197
<u>Método setDirectorio</u>	197
<u>Método setHostname</u>	197
<u>Método setId</u>	197
<u>Método setIp</u>	198
<u>Método setNombre</u>	198
<u>Método update</u>	198
<b>Clase Errorlog_model</b>	199
<u>Constructor construct</u>	199
<u>Método addFilter</u>	199
<u>Método getAll</u>	199
<u>Método getById</u>	200
<u>Método getDescripcion</u>	200
<u>Método getFecha hora</u>	200
<u>Método getId</u>	201
<u>Método getId_controlador accion</u>	201
<u>Método getId_usuario</u>	201
<u>Método getNumRows</u>	201
<u>Método insert</u>	201
<u>Método resetFilter</u>	202
<u>Método save</u>	202
<u>Método setDescripcion</u>	202
<u>Método setId</u>	203

<u>Método setId_accion</u>	203
<u>Método setId_controlador</u>	203
<u>Método setId_controlador_accion</u>	203
<u>Método setId_usuario</u>	204
<u>Clase Georeferencia_model</u>	204
<u>Constructor construct</u>	204
<u>Método getMsgError</u>	205
<u>Método process</u>	205
<u>Clase Grupo_model</u>	205
<u>Constructor construct</u>	206
<u>Método delete</u>	206
<u>Método getAll</u>	206
<u>Método getById</u>	206
<u>Método getByNombre</u>	207
<u>Método getDescripcion</u>	207
<u>Método getEntornosById</u>	207
<u>Método getId</u>	208
<u>Método getMsgError</u>	208
<u>Método getNombre</u>	208
<u>Método getNumRows</u>	208
<u>Método insert</u>	209
<u>Método setDescripcion</u>	209
<u>Método setId</u>	209
<u>Método setNombre</u>	209
<u>Método update</u>	210
<u>Clase Hemoglobina_model</u>	210
<u>Constructor construct</u>	210
<u>Método getMsgError</u>	210
<u>Método process</u>	211
<u>Clase Menu_model</u>	211
<u>Constructor construct</u>	211
<u>Método addFilter</u>	212
<u>Método delete</u>	212
<u>Método deleteByFilter</u>	212
<u>Método getAll</u>	212
<u>Método getAtributo</u>	213
<u>Método getById</u>	213
<u>Método getByPadre</u>	213
<u>Método getId</u>	214
<u>Método getId_controlador</u>	214
<u>Método getId_padre</u>	214
<u>Método getId_raiz</u>	214
<u>Método getMsgError</u>	214
<u>Método getNombre</u>	215
<u>Método getNumRows</u>	215
<u>Método getRuta</u>	215
<u>Método hasChild</u>	215
<u>Método insert</u>	216
<u>Método resetFilter</u>	216

<u>Método setAtributo</u>	216
<u>Método setId</u>	216
<u>Método setId_controlador</u>	217
<u>Método setId_padre</u>	217
<u>Método setId_raiz</u>	217
<u>Método setNombre</u>	217
<u>Método setRuta</u>	218
<u>Método update</u>	218
<b>Clase Permiso model</b>	219
Constructor construct	219
<u>Método deletePermissions</u>	219
<u>Método getFecha</u>	219
<u>Método getId</u>	219
<u>Método getIdControladorAccion</u>	220
<u>Método getIdGrupo</u>	220
<u>Método getMsgError</u>	220
<u>Método getPermission</u>	220
<u>Método insertBatch</u>	221
<u>Método setFecha</u>	221
<u>Método setId</u>	221
<u>Método setIdControladorAccion</u>	221
<u>Método setIdGrupo</u>	222
<b>Clase Poblacion model</b>	222
Constructor construct	222
<u>Método getMsgError</u>	223
<u>Método process</u>	223
<b>Clase Raiz model</b>	223
Constructor construct	224
<u>Método delete</u>	224
<u>Método ExistInArbol</u>	224
<u>Método getAll</u>	224
<u>Método getById</u>	225
<u>Método getDescripcion</u>	225
<u>Método getId</u>	225
<u>Método getMsgError</u>	225
<u>Método insert</u>	226
<u>Método setDescripcion</u>	226
<u>Método setId</u>	226
<u>Método update</u>	226
<b>Clase ReglaVacuna model</b>	227
Constructor construct	227
<u>Método delete</u>	227
<u>Método getAlergias</u>	227
<u>Método getAll</u>	228
<u>Método getById</u>	228
<u>Método getDiaFinNacido</u>	228
<u>Método getDiaFinPrevia</u>	228
<u>Método getDialnicioNacido</u>	229
<u>Método getDialnicioPrevia</u>	229

<u>Método getDosis</u>	229
<u>Método getEsqComp</u>	229
<u>Método getForzarAplicacion</u>	229
<u>Método getId</u>	229
<u>Método getIdVacuna</u>	229
<u>Método getIdVacunaPrevia</u>	230
<u>Método getIdViaVacuna</u>	230
<u>Método getMsgError</u>	230
<u>Método getObservacionRegion</u>	230
<u>Método getOrdenEsqComp</u>	231
<u>Método getRegion</u>	231
<u>Método insert</u>	231
<u>Método setAlergias</u>	231
<u>Método setDiaFinNacido</u>	231
<u>Método setDiaFinPrevia</u>	232
<u>Método setDialInicioNacido</u>	232
<u>Método setDialInicioPrevia</u>	232
<u>Método setDosis</u>	232
<u>Método setEsqComp</u>	233
<u>Método setForzarAplicacion</u>	233
<u>Método setId</u>	233
<u>Método setIdVacuna</u>	234
<u>Método setIdVacunaPrevia</u>	234
<u>Método setIdViaVacuna</u>	234
<u>Método setObservacionRegion</u>	234
<u>Método setOrdenEsqComp</u>	235
<u>Método setRegion</u>	235
<u>Método update</u>	235
<u>Clase Usuario_model</u>	236
<u>Constructor construct</u>	236
<u>Método authenticate</u>	236
<u>Método checkCredentials</u>	236
<u>Método check_data</u>	237
<u>Método check_token</u>	237
<u>Método delete</u>	237
<u>Método getActivesByGroup</u>	238
<u>Método getActivo</u>	238
<u>Método getApellidoMaterno</u>	238
<u>Método getApellidoPaterno</u>	238
<u>Método getById</u>	238
<u>Método getByUsername</u>	239
<u>Método getClave</u>	239
<u>Método getCorreo</u>	239
<u>Método getgrupo</u>	239
<u>Método getId</u>	240
<u>Método getIdGrupo</u>	240
<u>Método getMsgError</u>	240
<u>Método getNombre</u>	240
<u>Método getNombreUsuario</u>	241

<u>Método getNumRows</u>	241
<u>Método getOnlyActives</u>	241
<u>Método getuser</u>	242
<u>Método get_grupo_entorno</u>	242
<u>Método get_permiso_entorno</u>	242
<u>Método get_usuario_entorno</u>	242
<u>Método insert</u>	243
<u>Método setActivo</u>	243
<u>Método setApellidoMaterno</u>	243
<u>Método setApellidoPaterno</u>	243
<u>Método setClave</u>	244
<u>Método setCorreo</u>	244
<u>Método setId</u>	244
<u>Método setIdGrupo</u>	245
<u>Método setNombre</u>	245
<u>Método setNombreUsuario</u>	245
<u>Método update</u>	245
<u>Método update_pass</u>	246
<u>Método update_user</u>	246
<b>Paquete TES Elementos procedurales</b>	248
<u>enrolamiento.php</u>	248
<u>notificacion.php</u>	249
<u>reporteador.php</u>	250
<u>reporte_sincronizacion.php</u>	251
<u>semana_nacional.php</u>	252
<u>servicios.php</u>	253
<u>tableta.php</u>	254
<u>usuario_tableta.php</u>	255
<b>Paquete TES Clases</b>	256
<u>Clase Enrolamiento</u>	256
<u>Constructor_construct</u>	256
<u>Método addForm</u>	256
<u>Método autocomplete</u>	256
<u>Método brothers_search</u>	257
<u>Método brother_found</u>	257
<u>Método catalog_check</u>	257
<u>Método catalog_select</u>	258
<u>Método categoriacie10_select</u>	258
<u>Método checar_session</u>	258
<u>Método cie10_select</u>	259
<u>Método comparar_view</u>	259
<u>Método data_tutor</u>	259
<u>Método file_to_card</u>	260
<u>Método ifCurpExists</u>	260
<u>Método ifCurpTExists</u>	261
<u>Método index</u>	261
<u>Método insert</u>	261
<u>Método paciente_similar</u>	262

<u>Método print_card</u>	262
<u>Método searchageb</u>	262
<u>Método searchum</u>	263
<u>Método tratamiento_select</u>	263
<u>Método update</u>	264
<u>Método update_card</u>	264
<u>Método vacunacion</u>	264
<u>Método validarForm</u>	265
<u>Método validarism</u>	265
<u>Método validate_card</u>	266
<u>Método view</u>	266
<u>Clase Notificacion</u>	266
<u>Constructor_construct</u>	267
<u>Método delete</u>	267
<u>Método index</u>	267
<u>Método insert</u>	267
<u>Método update</u>	268
<u>Método view</u>	268
<u>Clase Reporteador</u>	269
<u>Constructor_construct</u>	269
<u>Método index</u>	269
<u>Método view</u>	269
<u>Clase Reporte sincronizacion</u>	270
<u>Constructor_construct</u>	270
<u>Método index</u>	270
<u>Método lote</u>	270
<u>Método lote_view</u>	271
<u>Método view</u>	271
<u>Clase Semana nacional</u>	271
<u>Constructor_construct</u>	272
<u>Método delete</u>	272
<u>Método getAll</u>	272
<u>Método index</u>	272
<u>Método insert</u>	273
<u>Método update</u>	273
<u>Método view</u>	273
<u>Clase Servicios</u>	274
<u>Constructor_construct</u>	274
<u>Método actualiza_estado_tableta</u>	274
<u>Método catalogos_relevantes</u>	275
<u>Método esquema_incompleto</u>	275
<u>Método is_step_0</u>	275
<u>Método is_step_1</u>	276
<u>Método is_step_2</u>	276
<u>Método is_step_3</u>	277
<u>Método is_step_4</u>	277
<u>Método prueba2</u>	278
<u>Método ss_step_5</u>	278
<u>Método ss_step_6</u>	278

<u>Método Synchronization</u>	279
<u>Clase Tableta</u>	279
<u>Constructor construct</u>	280
<u>Método delete</u>	280
<u>Método index</u>	280
<u>Método insert</u>	281
<u>Método setUM</u>	281
<u>Método update</u>	281
<u>Método uploadFile</u>	282
<u>Método view</u>	282
<u>Método validateMac</u>	282
<u>Clase Usuario_tableta</u>	283
<u>Constructor construct</u>	283
<u>Método delete</u>	283
<u>Método index</u>	283
<u>Método insert</u>	284
<u>enrolamiento_model.php</u>	285
<u>estado_tableta_model.php</u>	286
<u>notificacion_model.php</u>	287
<u>reporteador_model.php</u>	288
<u>reporte_censo_nominal.php</u>	289
<u>reporte_sincronizacion_model.php</u>	290
<u>semana_nacional_model.php</u>	291
<u>tableta_model.php</u>	292
<u>tipo_censo_model.php</u>	293
<u>usuario_tableta_model.php</u>	294
<u>Clase Enrolamiento_model</u>	294
<u>Constructor construct</u>	294
<u>Método autocomplete_tutor</u>	294
<u>Método cns_insert</u>	295
<u>Método cns_update</u>	295
<u>Método cns_update_visita</u>	295
<u>Método data_tutor</u>	296
<u>Método entorno_x_persona</u>	296
<u>Método getaccion_nutricional</u>	297
<u>Método getafiliacion</u>	297
<u>Método getAfiliaciones</u>	297
<u>Método getageb</u>	297
<u>Método getAlergia</u>	297
<u>Método getalergias</u>	298
<u>Método getaltura</u>	298
<u>Método getByCurn</u>	298
<u>Método getById</u>	298
<u>Método getcalle</u>	299
<u>Método getCategoríaCIE10</u>	299
<u>Método getcelular</u>	299
<u>Método getcelularT</u>	299
<u>Método getCIE10</u>	299
<u>Método getcodigo_barras</u>	300

<a href="#">Método getcolonia</a>	300
<a href="#">Método getcompania</a>	300
<a href="#">Método getcompaniaT</a>	300
<a href="#">Método getconsulta</a>	300
<a href="#">Método getControlConsultas</a>	301
<a href="#">Método getcp</a>	301
<a href="#">Método getcurp</a>	301
<a href="#">Método getcurpT</a>	301
<a href="#">Método getestimulacion_capacitado</a>	301
<a href="#">Método getestimulacion_fecha</a>	301
<a href="#">Método getfaccion_nutricional</a>	302
<a href="#">Método getfconsulta</a>	302
<a href="#">Método getfechacivil</a>	302
<a href="#">Método getfecha_peri_cefa</a>	302
<a href="#">Método getfnacimiento</a>	302
<a href="#">Método getfnutricion</a>	302
<a href="#">Método getfolio</a>	303
<a href="#">Método getfvacuna</a>	303
<a href="#">Método gethemoglobina</a>	303
<a href="#">Método getId</a>	303
<a href="#">Método getidtutor</a>	303
<a href="#">Método getListEnrolamiento</a>	304
<a href="#">Método getInacimiento</a>	304
<a href="#">Método getlocalidad</a>	304
<a href="#">Método getlugarcivil</a>	304
<a href="#">Método getmanzana</a>	304
<a href="#">Método getmaterno</a>	305
<a href="#">Método getmaternoT</a>	305
<a href="#">Método getMsgError</a>	305
<a href="#">Método getnacionalidad</a>	305
<a href="#">Método getnombre</a>	305
<a href="#">Método getnombreT</a>	305
<a href="#">Método getnumero</a>	306
<a href="#">Método getNumRows</a>	306
<a href="#">Método getparto</a>	306
<a href="#">Método getpaterno</a>	306
<a href="#">Método getpaternoT</a>	306
<a href="#">Método getperi_cefa</a>	307
<a href="#">Método getpeso</a>	307
<a href="#">Método getprecurp</a>	307
<a href="#">Método getreferencia</a>	307
<a href="#">Método getRegistro_civil</a>	307
<a href="#">Método getsales_cantidad</a>	307
<a href="#">Método getsales_fecha</a>	308
<a href="#">Método getsangre</a>	308
<a href="#">Método getsector</a>	308
<a href="#">Método getsexo</a>	308
<a href="#">Método getsexoT</a>	308
<a href="#">Método gettalla</a>	308

<u>Método gettamiz</u>	309
<u>Método gettbeneficiario</u>	309
<u>Método getconsulta</u>	309
<u>Método gettelefono</u>	309
<u>Método gettelefonoT</u>	309
<u>Método getumt</u>	309
<u>Método getvacuna</u>	309
<u>Método get_catalog</u>	310
<u>Método get_catalog2</u>	310
<u>Método get_catalog_count</u>	311
<u>Método get_catalog_relevante</u>	311
<u>Método get_catalog_tratamiento</u>	311
<u>Método get_catalog_view</u>	312
<u>Método get_cns_cat_persona</u>	312
<u>Método get_cns_cat_persona_count</u>	313
<u>Método get_cns_persona</u>	313
<u>Método get_control_nutricional</u>	313
<u>Método get_datos_grafica</u>	314
<u>Método get_estimulacion</u>	314
<u>Método get_notificacion</u>	314
<u>Método get_pacientes</u>	315
<u>Método get_peri_cefa</u>	315
<u>Método get_persona_x_tutor</u>	315
<u>Método get_sales</u>	316
<u>Método get_transaction_relevante</u>	316
<u>Método get_version</u>	316
<u>Método insert</u>	317
<u>Método setaccion_nutricional</u>	317
<u>Método setafiliacion</u>	317
<u>Método setageb</u>	317
<u>Método setalergias</u>	318
<u>Método setaltura</u>	318
<u>Método setcalle</u>	318
<u>Método setcelular</u>	318
<u>Método setcelularT</u>	319
<u>Método setcodigo_barras</u>	319
<u>Método setcolonia</u>	319
<u>Método setcompania</u>	320
<u>Método setcompaniaT</u>	320
<u>Método setconsulta</u>	320
<u>Método setcp</u>	320
<u>Método setcurp</u>	321
<u>Método setcurpT</u>	321
<u>Método setestimulacion_capacitado</u>	321
<u>Método setestimulacion_fecha</u>	322
<u>Método setfaccion_nutricional</u>	322
<u>Método setfconsulta</u>	322
<u>Método setfechacivil</u>	322
<u>Método setfecha_peri_cefa</u>	323

<a href="#">Método setfnacimiento</a>	323
<a href="#">Método setfnutricion</a>	323
<a href="#">Método setfvacuna</a>	323
<a href="#">Método sethemoglobina</a>	324
<a href="#">Método setId</a>	324
<a href="#">Método setIdtutor</a>	324
<a href="#">Método setInacimiento</a>	325
<a href="#">Método setLocalidad</a>	325
<a href="#">Método setLugarcivil</a>	325
<a href="#">Método setManzana</a>	325
<a href="#">Método setMaterno</a>	326
<a href="#">Método setMaternoT</a>	326
<a href="#">Método setNacionalidad</a>	326
<a href="#">Método setNombre</a>	327
<a href="#">Método setNombreT</a>	327
<a href="#">Método setNumero</a>	327
<a href="#">Método setParto</a>	327
<a href="#">Método setPaterno</a>	328
<a href="#">Método setPaternoT</a>	328
<a href="#">Método setPeri_cefa</a>	328
<a href="#">Método setPeso</a>	329
<a href="#">Método setPrecurp</a>	329
<a href="#">Método setReferencia</a>	329
<a href="#">Método setsales_cantidad</a>	329
<a href="#">Método setsales_fecha</a>	330
<a href="#">Método setsangre</a>	330
<a href="#">Método setSector</a>	330
<a href="#">Método setSexo</a>	330
<a href="#">Método setSexoT</a>	331
<a href="#">Método setTalla</a>	331
<a href="#">Método setTamiz</a>	331
<a href="#">Método setBeneficiario</a>	332
<a href="#">Método setConsulta</a>	332
<a href="#">Método setTelefono</a>	332
<a href="#">Método setTelefonoT</a>	332
<a href="#">Método setUmT</a>	333
<a href="#">Método setVacuna</a>	333
<a href="#">Método tes_pendientes_tarjeta_delete</a>	333
<a href="#">Método update_accion</a>	334
<a href="#">Método update_alergia</a>	334
<a href="#">Método update_basico</a>	334
<a href="#">Método update_Beneficiario</a>	334
<a href="#">Método update_consulta</a>	334
<a href="#">Método update_direccion</a>	335
<a href="#">Método update_estimulacion</a>	335
<a href="#">Método update_nutricion</a>	335
<a href="#">Método update_peri_cefa</a>	335
<a href="#">Método update_recivil</a>	335
<a href="#">Método update_sales</a>	336

<u>Método update_status_tableta</u>	336
<u>Método update_tutor</u>	336
<u>Método update_umt</u>	336
<u>Método update_vacuna</u>	337
<u>Método valid_card</u>	337
<u>Clase Estado_tableta_model</u>	337
<u>Constructor_construct</u>	338
<u>Método getAll</u>	338
<u>Método getById</u>	338
<u>Método getDescripcion</u>	338
<u>Método getId</u>	339
<u>Método getMsgError</u>	339
<u>Método setDescripcion</u>	339
<u>Método setId</u>	339
<u>Clase Notificacion_model</u>	340
<u>Constructor_construct</u>	340
<u>Método addFilter</u>	340
<u>Método delete</u>	341
<u>Método getAll</u>	341
<u>Método getById</u>	341
<u>Método getContenido</u>	342
<u>Método getFechaFin</u>	342
<u>Método getFechalInicio</u>	342
<u>Método getId</u>	342
<u>Método getIdsTabletas</u>	342
<u>Método getMsgError</u>	342
<u>Método getNumRows</u>	343
<u>Método getTitulo</u>	343
<u>Método insert</u>	343
<u>Método setContenido</u>	343
<u>Método setFechaFin</u>	344
<u>Método setFechalInicio</u>	344
<u>Método setId</u>	344
<u>Método setIdsTabletas</u>	345
<u>Método setTitulo</u>	345
<u>Método update</u>	345
<u>Clase Reporteador_model</u>	345
<u>Constructor_construct</u>	346
<u>Método getCensoNominal</u>	346
<u>Método getCoberturaBiologicoListado</u>	346
<u>Método getConcentradoActividades</u>	347
<u>Método getEsquemasIncompletos</u>	347
<u>Método getGrupoVacunas</u>	347
<u>Método getMsgError</u>	347
<u>Método getSeguimientoRV1RV5</u>	348
<u>Método getVacunas</u>	348
<u>Método getVacunasByGrupo</u>	348
<u>Método object_to_array</u>	348
<u>Clase Reporte_censo_nominal</u>	348

<u>Var \$apellido_materno</u>	349
<u>Var \$apellido_paterno</u>	349
<u>Var \$curp</u>	349
<u>Var \$domicilio</u>	349
<u>Var \$fecha_nacimiento</u>	349
<u>Var \$nombre</u>	350
<u>Var \$parto_multiple</u>	350
<u>Var \$sexo</u>	350
<u>Var \$vacunas</u>	350
<u>Constructor construct</u>	350
<u>Clase Reporte sincronizacion model</u>	351
<u>Método getCount</u>	351
<u>Método getListado</u>	351
<u>Método getMsgError</u>	352
<u>Método get_version</u>	352
<u>Clase Semana nacional model</u>	352
<u>Constructor construct</u>	353
<u>Método delete</u>	353
<u>Método getAll</u>	353
<u>Método getById</u>	353
<u>Método getDescripcion</u>	354
<u>Método getFecha_fin</u>	354
<u>Método getFecha_inicio</u>	354
<u>Método getId</u>	354
<u>Método getMsgError</u>	354
<u>Método getNumRows</u>	355
<u>Método insert</u>	355
<u>Método setDescripcion</u>	355
<u>Método setFecha_fin</u>	356
<u>Método setFecha_inicio</u>	356
<u>Método update</u>	356
<u>Clase Tableta model</u>	357
<u>Constructor construct</u>	357
<u>Método delete</u>	357
<u>Método getAll</u>	357
<u>Método getById</u>	358
<u>Método getByMac</u>	358
<u>Método getId</u>	358
<u>Método getIdVersion</u>	359
<u>Método getId_asu_um</u>	359
<u>Método getId_tes_estado_tableta</u>	359
<u>Método getId_tipo_censo</u>	359
<u>Método getMac</u>	359
<u>Método getMsgError</u>	359
<u>Método getNumRows</u>	360
<u>Método getPeriodo_esq_inc</u>	360
<u>Método getLast_update</u>	360
<u>Método getUsuarios_asignados</u>	360
<u>Método insert</u>	360

<u>Método setId</u>	361
<u>Método setIdVersion</u>	361
<u>Método setId_asu_um</u>	361
<u>Método setId_tes_estado_tableta</u>	361
<u>Método setId_tipo_censo</u>	362
<u>Método setMac</u>	362
<u>Método setPeriodo_esq_inc</u>	362
<u>Método setUltima_actualizacion</u>	363
<u>Método setUsuarios_asignados</u>	363
<u>Método update</u>	363
<u>Clase Tipo_censo_model</u>	364
<u>Constructor construct</u>	364
<u>Método getAll</u>	364
<u>Método getById</u>	364
<u>Método getDescripcion</u>	365
<u>Método getId</u>	365
<u>Método getMsgError</u>	365
<u>Método setDescripcion</u>	365
<u>Método setId</u>	366
<u>Clase Usuario_tableta_model</u>	366
<u>Constructor construct</u>	366
<u>Método delete</u>	366
<u>Método getMsgError</u>	367
<u>Método getTableta</u>	367
<u>Método getTabletasByUsuario</u>	367
<u>Método getUsuario</u>	368
<u>Método getUsuariosByTableta</u>	368
<u>Método insert</u>	368
<u>Método setTableta</u>	368
<u>Método setUsuario</u>	369
<u>Appendices</u>	370
<u>Appendix A - Class Trees</u>	371
<u>SIIGS</u>	371
<u>default</u>	379
<u>TES</u>	379
<u>Libreria</u>	383
<u>CodeIgniter</u>	384
<u>Session</u>	384
<u>Appendix C - Source Code</u>	386
<u>Package TES</u>	387
<u>source code: index.php</u>	388
<u>source code: index.php</u>	389
<u>Package TES</u>	389
<u>source code: Form_validation.php</u>	390
<u>source code: Pagination.php</u>	408
<u>source code: template.php</u>	413
<u>Package TES</u>	421
<u>source code: menubuilder.php</u>	422
<u>source code: graph.php</u>	424

<a href="#">source code: obtenercurp.php</a>	426
<a href="#">source code: tree.php</a>	432
<a href="#">Package TES</a>	433
<a href="#">source code: Session.php</a>	434
<a href="#">Package TES</a>	437
<a href="#">source code: ayuda.php</a>	438
<a href="#">source code: accion.php</a>	439
<a href="#">source code: bitacora.php</a>	443
<a href="#">source code: catalogo.php</a>	449
<a href="#">source code: catalogocsv.php</a>	461
<a href="#">source code: catalogo_x_raiz.php</a>	470
<a href="#">source code: cie10.php</a>	474
<a href="#">source code: controlador.php</a>	483
<a href="#">source code: entorno.php</a>	490
<a href="#">source code: errorlog.php</a>	495
<a href="#">source code: grupo.php</a>	498
<a href="#">source code: menu.php</a>	502
<a href="#">source code: permiso.php</a>	508
<a href="#">source code: raiz.php</a>	510
<a href="#">source code: reglavaduna.php</a>	521
<a href="#">source code: usuario.php</a>	527
<a href="#">source code: accion_model.php</a>	540
<a href="#">source code: ageb_model.php</a>	544
<a href="#">source code: arbolesegmentacion_model.php</a>	547
<a href="#">source code: bitacora_model.php</a>	563
<a href="#">source code: catalogocsv_model.php</a>	571
<a href="#">source code: catalogo_model.php</a>	576
<a href="#">source code: catalogo_x_raiz_model.php</a>	582
<a href="#">source code: cie10_model.php</a>	588
<a href="#">source code: controladoraccion_model.php</a>	594
<a href="#">source code: controlador_model.php</a>	597
<a href="#">source code: entorno_model.php</a>	604
<a href="#">source code: errorlog_model.php</a>	609
<a href="#">source code: georeferencia_model.php</a>	615
<a href="#">source code: grupo_model.php</a>	617
<a href="#">source code: hemoglobina_model.php</a>	621
<a href="#">source code: menu_model.php</a>	623
<a href="#">source code: permiso_model.php</a>	632
<a href="#">source code: poblacion_model.php</a>	635
<a href="#">source code: raiz_model.php</a>	637
<a href="#">source code: reglavaduna_model.php</a>	641
<a href="#">source code: usuario_model.php</a>	648
<a href="#">Package TES</a>	656
<a href="#">source code: enrolamiento.php</a>	657
<a href="#">source code: notificacion.php</a>	677
<a href="#">source code: reporteador.php</a>	682
<a href="#">source code: reporte_sincronizacion.php</a>	685
<a href="#">source code: semana_nacional.php</a>	693
<a href="#">source code: servicios.php</a>	698

<a href="#">source code: tableta.php</a>	716
<a href="#">source code: usuario_tableta.php</a>	723
<a href="#">source code: enrolamiento_model.php</a>	726
<a href="#">source code: estado_tableta_model.php</a>	766
<a href="#">source code: notificacion_model.php</a>	769
<a href="#">source code: reporteador_model.php</a>	775
<a href="#">source code: reporte_censo_nominal.php</a>	784
<a href="#">source code: reporte_sincronizacion_model.php</a>	785
<a href="#">source code: semana_nacional_model.php</a>	787
<a href="#">source code: tableta_model.php</a>	792
<a href="#">source code: tipo_censo_model.php</a>	800
<a href="#">source code: usuario_tableta_model.php</a>	803



# Paquete default Elementos procedurales

## index.php

- **Package default**
- **Filesource** [Source Code for this file](#)

# index.php

- **Package default**
- **Filesource** [Source Code for this file](#)

# Paquete default Clases

## Clase Index [line [3](#)]

- **Package** default

Constructor `void` función `Index::__construct()` [line [5](#)]

- **Access** public

`void` función `Index::index()` [line [10](#)]

- **Access** public



# Paquete CodeIgniter Elementos procedurales

## Form\_validation.php

### CodeIgniter

An open source application development framework for PHP 5.1.6 or newer

- **Package** CodeIgniter
- **Author** ExpressionEngine Dev Team
- **Copyright** Copyright (c) 2008 - 2011, EllisLab, Inc.
- **Link** <http://codeigniter.com>
- **Since** Version 1.0
- **Filesource** [Source Code for this file](#)
- **License** [http://codeigniter.com/user\\_guide/license.html](http://codeigniter.com/user_guide/license.html)

# Pagination.php

## CodeIgniter

An open source application development framework for PHP 5.1.6 or newer

- **Package** CodeIgniter
- **Author** ExpressionEngine Dev Team
- **Copyright** Copyright (c) 2008 - 2011, EllisLab, Inc.
- **Link** <http://codeigniter.com>
- **Since** Version 1.0
- **Filesource** [Source Code for this file](#)
- **License** [http://codeigniter.com/user\\_guide/license.html](http://codeigniter.com/user_guide/license.html)

# template.php

## CodeIgniter

An open source application development framework for PHP 4.3.2 or newer

- **Package** CodeIgniter
- **Author** ExpressionEngine Dev Team
- **Copyright** Copyright (c) 2006, EllisLab, Inc.
- **Link** <http://codeigniter.com>
- **Since** Version 1.0
- **Filesource** [Source Code for this file](#)
- **License** [http://codeigniter.com/user\\_guide/license.html](http://codeigniter.com/user_guide/license.html)

# Paquete CodeIgniter Clases

## Clase CI\_Form\_validation [line 27]

### Form Validation Class

- **Package** CodeIgniter
- **Sub-Package** Libraries
- **Author** ExpressionEngine Dev Team
- **Link** [http://codeigniter.com/user\\_guide/libraries/form\\_validation.html](http://codeigniter.com/user_guide/libraries/form_validation.html)

### CI\_Form\_validation::\$CI

*mixed = [line 29]*

- **Access** protected

### CI\_Form\_validation::\$error\_string

*mixed = " [line 36]*

- **Access** protected

### **CI\_Form\_validation::\$\_config\_rules**

*mixed = array() [line 31]*

- **Access** protected

### **CI\_Form\_validation::\$\_error\_array**

*mixed = array() [line 32]*

- **Access** protected

### **CI\_Form\_validation::\$\_error\_messages**

*mixed = array() [line 33]*

- **Access** protected

### **CI\_Form\_validation::\$\_error\_prefix**

*mixed = '<div class="error">' [line 34]*

- **Access** protected

### **CI\_Form\_validation::\$\_error\_suffix**

*mixed = '</div>' [line 35]*

- **Access** protected

### **CI\_Form\_validation::\$\_field\_data**

*mixed = array() [line 30]*

- **Access** protected

### **CI\_Form\_validation::\$\_safe\_form\_data**

*mixed = FALSE [line 37]*

- **Access** protected

Constructor *void* función **CI\_Form\_validation::\_\_construct([&\$rules = array()]) [line 42]**

**Parámetros de la función:**

- **\$rules**

### **Constructor**

- **Access** public

*bool* función **CI\_Form\_validation::alpha(\$str) [line 1099]**

**Parámetros de la función:**

- *string* **\$str**

### **Alpha**

- **Access** public

*bool función CI\_Form\_validation::alpha\_dash(\$str) [line 1127]*

**Parámetros de la función:**

- *string \$str*

## Alpha-numeric with underscores and dashes

- **Access** public

*bool función CI\_Form\_validation::alpha\_numeric(\$str) [line 1113]*

**Parámetros de la función:**

- *string \$str*

## Alpha-numeric

- **Access** public

*bool función CI\_Form\_validation::decimal(\$str) [line 1184]*

**Parámetros de la función:**

- *string \$str*

## Decimal number

- **Access** public

*string función CI\_Form\_validation::encode\_php\_tags(\$str) [line 1373]*

**Parámetros de la función:**

- *string \$str*

## Convert PHP tags to entities

- **Access** public

*void función CI\_Form\_validation::error([\$field = "], [\$prefix = "], [\$suffix = "]) [line 208]*

**Parámetros de la función:**

- *string \$field* the field name
- *\$prefix*
- *\$suffix*

## Get Error Message

Gets the error message associated with a particular field

- **Access** public

*str función CI\_Form\_validation::error\_string([\$prefix = "], [\$suffix = "]) [line 240]*

**Parámetros de la función:**

- *string \$prefix*

- *string \$suffix*

## Error String

Returns the error messages as a string, wrapped in the error delimiters

- **Access** public

*bool función CI\_Form\_validation::exact\_length(\$str, \$val) [line 1019]*

**Parámetros de la función:**

- *string \$str*
- *value \$val*

## Exact Length

- **Access** public

*bool función CI\_Form\_validation::greater\_than(\$str, \$min) [line 1198]*

**Parámetros de la función:**

- *string \$str*
- *\$min*

## Greater than

- **Access** public

*bool función CI\_Form\_validation::integer(\$str) [line 1170]*

**Parámetros de la función:**

- *string \$str*

## **Integer**

- **Access** public

*bool función CI\_Form\_validation::is\_natural(\$str) [line 1234]*

**Parámetros de la función:**

- *string \$str*

## **Is a Natural number (0,1,2,3, etc.)**

- **Access** public

*bool función CI\_Form\_validation::is\_natural\_no\_zero(\$str) [line 1248]*

**Parámetros de la función:**

- *string \$str*

## **Is a Natural number, but not a zero (1,2,3, etc.)**

- **Access** public

*bool función CI\_Form\_validation::is\_numeric(\$str) [line 1156]*

**Parámetros de la función:**

- *string \$str*

## Is Numeric

- **Access** public

*bool función CI\_Form\_validation::is\_unique(\$str, \$field) [line 951]*

**Parámetros de la función:**

- *string \$str*
- *field \$field*

## Match one field to another

- **Access** public

*bool función CI\_Form\_validation::less\_than(\$str, \$max) [line 1216]*

**Parámetros de la función:**

- *string \$str*
- *\$max*

## Less than

- **Access** public

*bool función CI\_Form\_validation::matches(\$str, \$field) [line 929]*

**Parámetros de la función:**

- *string* **\$str**
- *field* **\$field**

## Match one field to another

- **Access** public

*bool función CI\_Form\_validation::max\_length(\$str, \$val) [line 994]*

**Parámetros de la función:**

- *string* **\$str**
- *value* **\$val**

## Max Length

- **Access** public

*bool función CI\_Form\_validation::min\_length(\$str, \$val) [line 969]*

**Parámetros de la función:**

- *string* **\$str**
- *value* **\$val**

## Minimum Length

- **Access** public

*bool función CI\_Form\_validation::numeric(\$str) [line 1141]*

**Parámetros de la función:**

- *string \$str*

## Numeric

- **Access** public

*string función CI\_Form\_validation::prep\_for\_form([\$data = "]) [line 1292]*

**Parámetros de la función:**

- *string \$data*

## Prep data for form

This function allows HTML to be safely shown in a form. Special characters are converted.

- **Access** public

*string función CI\_Form\_validation::prep\_url([\$str = "]) [line 1321]*

**Parámetros de la función:**

- *string \$str*

## Prep URL

- **Access** public

*bool función CI\_Form\_validation::regex\_match(\$str, \$regex) [line 909]*

**Parámetros de la función:**

- *string \$str*
- *regex \$regex*

## Performs a Regular Expression match test.

- **Access** public

*bool función CI\_Form\_validation::required(\$str) [line 887]*

**Parámetros de la función:**

- *string \$str*

## Required

- **Access** public

*bool función CI\_Form\_validation::run([\$group = "]) [line 281]*

**Parámetros de la función:**

- **\$group**

### Run the Validator

This function does all the work.

- **Access** public

*string función CI\_Form\_validation::set\_checkbox([\$field = ""], [\$value = ""], [\$default = FALSE]) [line 847]*

**Parámetros de la función:**

- **string \$field**
- **string \$value**
- **\$default**

### Set Checkbox

Enables checkboxes to be set to the value the user selected in the event of an error

- **Access** public

*void función CI\_Form\_validation::set\_error\_delimiters([\$prefix = '<p>'], [\$suffix = '</p>']) [line 189]*

**Parámetros de la función:**

- **string \$prefix**
- **string \$suffix**

### Set The Error Delimiter

Permits a prefix/suffix to be added to each error message

- **Access** public

*string* función CI\_Form\_validation::set\_message(\$lang, [\$val = "]) [line 165]

**Parámetros de la función:**

- *string* **\$lang**
- *string* **\$val**

## Set Error Message

Lets users set their own error messages on the fly. Note: The key name has to match the function name that it corresponds to.

- **Access** public

*string* función CI\_Form\_validation::set\_radio([\$field = "], [\$value = "], [\$default = FALSE]) [line 803]

**Parámetros de la función:**

- *string* **\$field**
- *string* **\$value**
- **\$default**

## Set Radio

Enables radio buttons to be set to the value the user selected in the event of an error

- **Access** public

*void función CI\_Form\_validation::set\_rules(\$field, [\$label = ""], [\$rules = "]) [line 74]*

**Parámetros de la función:**

- ***mixed \$field***
- ***string \$label***
- ***\$rules***

## **Set Rules**

This function takes an array of field names and validation rules as input, validates the info, and stores it

- **Access** public

*string función CI\_Form\_validation::set\_select([\$field = ""], [\$value = ""], [\$default = FALSE]) [line 759]*

**Parámetros de la función:**

- ***string \$field***
- ***string \$value***
- ***\$default***

## **Set Select**

Enables pull-down lists to be set to the value the user selected in the event of an error

- **Access** public

*void función CI\_Form\_validation::set\_value([\$field = ""], [\$default = "]) [line 729]*

**Parámetros de la función:**

- ***string \$field*** the field name

- *string \$default*

## Get the value from a form

Permits you to repopulate a form field with the value it was submitted with, or, if that value doesn't exist, with the default

- **Access** public

*string función CI\_Form\_validation::strip\_image\_tags(\$str) [line 1345]*

**Parámetros de la función:**

- *string \$str*

## Strip Image Tags

- **Access** public

*bool función CI\_Form\_validation::valid\_base64(\$str) [line 1275]*

**Parámetros de la función:**

- *string \$str*

## Valid Base64

Tests a string for characters outside of the Base64 alphabet as defined by RFC 2045  
<http://www.faqs.org/rfcs/rfc2045>

- **Access** public

*bool función CI\_Form\_validation::valid\_email(\$str) [line 1043]*

**Parámetros de la función:**

- *string \$str*

## Valid Email

- **Access** public

*bool función CI\_Form\_validation::valid\_emails(\$str) [line 1057]*

**Parámetros de la función:**

- *string \$str*

## Valid Emails

- **Access** public

*string función CI\_Form\_validation::valid\_ip(\$ip, [\$which = "]) [line 1085]*

**Parámetros de la función:**

- *string \$ip*
- *string \$which "ipv4" or "ipv6" to validate a specific ip format*

## Validate IP Address

- **Access** public

*string* función CI\_Form\_validation::xss\_clean(\$str) [line 1359]

**Parámetros de la función:**

- *string* \$str

## XSS Clean

- **Access** public

*mixed* función CI\_Form\_validation::\_execute(\$row, \$rules, [\$postdata = NULL], [\$cycles = 0]) [line 470]

**Parámetros de la función:**

- *array* \$row
- *array* \$rules
- *mixed* \$postdata
- *integer* \$cycles

## Executes the Validation routines

- **Access** protected

*mixed* función CI\_Form\_validation::\_reduce\_array(\$array, \$keys, [\$i = 0]) [line 376]

**Parámetros de la función:**

- *array* \$array
- *array* \$keys
- *integer* \$i

## Traverse a multidimensional \$\_POST array index until the data is found

- **Access** protected

*null* función CI\_Form\_validation::\_reset\_post\_array() [line 408]

**Re-populate the \_POST array with our finalized and processed data**

- **Access** protected

*string* función CI\_Form\_validation::\_translate\_fieldname(\$fieldname) [line 697]

**Parámetros de la función:**

- **string \$fieldname** the field name

## Translate a field name

- **Access** protected

## Clase CI\_Pagination [line 27]

### Pagination Class

- **Package** CodeIgniter
- **Sub-Package** Libraries
- **Author** ExpressionEngine Dev Team
- **Link** [http://codeigniter.com/user\\_guide/libraries/pagination.html](http://codeigniter.com/user_guide/libraries/pagination.html)

### **CI\_Pagination::\$anchor\_class**

*mixed = " [line 61]*

### **CI\_Pagination::\$base\_url**

*mixed = " [line 29]*

### **CI\_Pagination::\$cur\_page**

*mixed = 0 [line 36]*

### **CI\_Pagination::\$cur\_tag\_close**

*mixed = '</strong></a>  
' [line 51]*

### **CI\_Pagination::\$cur\_tag\_open**

- *mixed = '  
• <a><strong>' [line 50]*

### **CI\_Pagination::\$display\_pages**

*mixed = TRUE [line 60]*

### **CI\_Pagination::\$first\_link**

*mixed = '&lsquo; First' [line 38]*

### **CI\_Pagination::\$first\_tag\_close**

*mixed = '  
' [line 46]*

### **CI\_Pagination::\$first\_tag\_open**

- *mixed = '  
' [line 45]*

### **CI\_Pagination::\$first\_url**

*mixed = " [line 49]*

### **CI\_Pagination::\$full\_tag\_close**

*mixed = '*

' [line 44]

**CI\_Pagination::\$full\_tag\_open**

*mixed = '<ul class="pagination pagination-sm">' [line 43]*

**CI\_Pagination::\$last\_link**

*mixed = 'Last &rsaquo;' [line 41]*

**CI\_Pagination::\$last\_tag\_close**

*mixed = '  
' [line 48]*

**CI\_Pagination::\$last\_tag\_open**

*mixed = '  
• ' [line 47]*

**CI\_Pagination::\$next\_link**

*mixed = '>' [line 39]*

**CI\_Pagination::\$next\_tag\_close**

*mixed = '  
' [line 53]*

**CI\_Pagination::\$next\_tag\_open**

*mixed = '  
• ' [line 52]*

**CI\_Pagination::\$num\_links**

*mixed = 2 [line 35]*

**CI\_Pagination::\$num\_tag\_close**

*mixed = '  
' [line 57]*

**CI\_Pagination::\$num\_tag\_open**

*mixed = '  
• ' [line 56]*

**CI\_Pagination::\$page\_query\_string**

*mixed = FALSE [line 58]*

**CI\_Pagination::\$per\_page**

*mixed = 10 [line 34]*

**CI\_Pagination::\$prefix**

*mixed = " [line 30]*

### **CI\_Pagination::\$prev\_link**

*mixed = '<' [line 40]*

### **CI\_Pagination::\$prev\_tag\_close**

*mixed = '  
' [line 55]*

### **CI\_Pagination::\$prev\_tag\_open**

*mixed = '  
• ' [line 54]*

### **CI\_Pagination::\$query\_string\_segment**

*mixed = 'per\_page' [line 59]*

### **CI\_Pagination::\$suffix**

*mixed = '' [line 31]*

### **CI\_Pagination::\$total\_rows**

*mixed = 0 [line 33]*

### **CI\_Pagination::\$uri\_segment**

*mixed = 3 [line 42]*

### **CI\_Pagination::\$use\_page\_numbers**

*mixed = FALSE [line 37]*

Constructor void función CI\_Pagination::\_\_construct([\$params = array()]) [line 69]

**Parámetros de la función:**

- **array \$params** initialization parameters

## **Constructor**

- **Access** public

*string función CI\_Pagination::create\_links() [line 115]*

**Generate the pagination links**

- **Access** public

*void función CI\_Pagination::initialize([&\$params = array()]) [line 93]*

**Parámetros de la función:**

- *array \$params* initialization parameters

## Initialize Preferences

- **Access** public

# Clase CI\_Template

*[line 33]*

## CodeIgniter Template Class

This class is and interface to CI's View class. It aims to improve the interaction between controllers and views. Follow @link for more info

- **Package** CodeIgniter
- **Sub-Package** Libraries
- **Author** Colin Williams
- **Version** 1.4.1
- **Copyright** Copyright (c) 2008, Colin Williams.
- **Link** <http://www.williamsconcepts.com/ci/libraries/template/index.html>

## CI\_Template::\$CI

*mixed = [line 35]*

### **CI\_Template::\$config**

*mixed = [line 36]*

### **CI\_Template::\$css**

*mixed = array() [line 45]*

### **CI\_Template::\$js**

*mixed = array() [line 44]*

### **CI\_Template::\$master**

*mixed = [line 38]*

### **CI\_Template::\$output**

*mixed = [line 43]*

### **CI\_Template::\$parser**

*mixed = 'parser' [line 46]*

### **CI\_Template::\$parser\_method**

*mixed = 'parse' [line 47]*

### **CI\_Template::\$parse\_template**

*mixed = FALSE [line 48]*

### **CI\_Template::\$regions**

*mixed = array(  
'\_scripts' => array(), '\_styles' => array(),) [line 39]*

### **CI\_Template::\$template**

*mixed = [line 37]*

Constructor void función CI\_Template::CI\_Template() [line 59]

#### **Constructor**

Loads template configuration, template regions, and validates existence of default template

- **Access** public

*TRUE función CI\_Template::add\_css(\$style, [\$type = 'link'], [\$media = FALSE]) [line 492]*

**Parámetros de la función:**

- *string \$style* CSS file to link, import or embed
- *string \$type* 'link', 'import' or 'embed'
- *string \$media* media attribute to use with 'link' type only, FALSE for none

### Dynamically include CSS in the template

NOTE: This function does NOT check for existence of .css file

- **Access** public

*TRUE función CI\_Template::add\_js(\$script, [\$type = 'import'], [\$defer = FALSE]) [line 433]*

**Parámetros de la función:**

- *string \$script* script to import or embed
- *string \$type* 'import' to load external file or 'embed' to add as-is
- *boolean \$defer* TRUE to use 'defer' attribute, FALSE to exclude it

### Dynamically include javascript in the template

NOTE: This function does NOT check for existence of .js file

- **Access** public

*void función CI\_Template::add\_region(\$name, [\$props = array()]) [line 233]*

**Parámetros de la función:**

- *string \$name* Name to identify the region
- *array \$props* Optional array with region defaults

## Dynamically add region to the currently set template

- **Access** public

*void función CI\_Template::add\_template(\$group, \$template, [\$activate = FALSE]) [line 129]*

**Parámetros de la función:**

- *string* **\$group** array key to access template settings
- *array* **\$template** properly formed
- **\$activate**

## Dynamically add a template and optionally switch to it

- **Access** public

*void función CI\_Template::empty\_region(\$name) [line 260]*

**Parámetros de la función:**

- *string* **\$name** Name to identify the region

## Empty a region's content

- **Access** public

*void función CI\_Template::initialize(\$props) [line 155]*

**Parámetros de la función:**

- *array \$props* configuration array

## Initialize class settings using config settings

- **Access** public

*void función CI\_Template::load([\$region = NULL], [\$buffer = FALSE]) [line 602]*

**Parámetros de la función:**

- **\$region**
- **\$buffer**

## Load the master template or a single region

DEPRECATED!

Use render() to compile and display your template and regions

*void función CI\_Template::parse\_view(\$region, \$view, [\$data = NULL], [\$overwrite = FALSE]) [line 391]*

**Parámetros de la función:**

- *string \$region* region to write to
- *string \$view* view file to parse
- *array \$data* variables to pass into view for parsing
- *boolean \$overwrite* FALSE to append to region, TRUE to overwrite region

## Parse content from a View to a region with the Parser Class

- **Access** public

*void función CI\_Template::render([\$region = NULL], [\$buffer = FALSE], [\$parse = FALSE]) [line 548]*

**Parámetros de la función:**

- *string \$region* optionally opt to render a specific region
- *boolean \$buffer* FALSE to output the rendered template, TRUE to return as a string. Always TRUE when \$region is supplied
- *\$parse*

## Render the master template or a single region

- **Access** public

*void función CI\_Template::set\_master\_template(\$filename) [line 106]*

**Parámetros de la función:**

- *string \$filename* filename of new master template file

## Set master template

- **Access** public

*void función CI\_Template::set\_parser(\$parser, [\$method = NULL]) [line 282]*

**Parámetros de la función:**

- *string \$parser* name of parser class to load and use for parsing methods
- *\$method*

## Set parser

- **Access** public

*void función CI\_Template::set\_parser\_method(\$method) [/line 303]*

**Parámetros de la función:**

- *string \$method* name of parser class member function to call when parsing

## **Set parser method**

- **Access** public

*void función CI\_Template::set\_regions(\$regions) [/line 199]*

**Parámetros de la función:**

- *array \$regions* properly formed regions array

## **Set regions for writing to**

- **Access** public

*void función CI\_Template::set\_template(\$group) [/line 83]*

**Parámetros de la función:**

- *string \$group* array key to access template settings

## **Use given template settings**

- **Access** public

*void función CI\_Template::write(\$region, \$content, [\$overwrite = FALSE]) [line 320]*

**Parámetros de la función:**

- *string \$region* region to write to
- *string \$content* what to write
- *boolean \$overwrite* FALSE to append to region, TRUE to overwrite region

## Write contents to a region

- **Access** public

*void función CI\_Template::write\_view(\$region, \$view, [\$data = NULL], [\$overwrite = FALSE]) [line 352]*

**Parámetros de la función:**

- *string \$region* region to write to
- *string \$view* view file to use
- *array \$data* variables to pass into view
- *boolean \$overwrite* FALSE to append to region, TRUE to overwrite region

## Write content from a View to a region. 'Views within views'

- **Access** public



# Paquete Libreria Elementos procedurales

## menubuilder.php

- **Package** Libreria
- **Sub-Package** Clase
- **Filesource** [Source Code for this file](#)

# Paquete Libreria Clases

## Clase Menubuilder

[line [12](#)]

### Menu Builder

- **Package** Libreria
- **Sub-Package** Clase
- **Author** Pascual

Constructor `void` función Menubuilder::\_\_construct() [line[23](#)]

- **Access** public

`string` función Menubuilder::build([`$todos = false`]) [line[25](#)]

**Parámetros de la función:**

- `boolean $todos` Establece si se debe devolver todos los elementos del menu

### Construye el menu basandose en los permisos del usuario logeado

- **Static**

- **Access** public

void función Menubuilder::crearMenu(&\$strMenu, [\$id\_padre = 'NULL'], [\$todos = false], \$strMenu) [[line 68](#)]  
**Parámetros de la función:**

- *string \$strMenu* Variable pasada por referencia donde se guardará la cadena de ul y li
- *string \$id\_padre* ID del padre de los elementos del arbol de menu
- *&\$strMenu*
- *\$todos*

### Función recursiva que recorre todo el árbol de elementos del menu

- **Static**
- **Access** public

void función Menubuilder::isGranted(\$accion, \$strMenu, \$id\_padre) [[line 123](#)]  
**Parámetros de la función:**

- *string \$strMenu* Variable pasada por referencia donde se guardará la cadena de ul y li
- *string \$id\_padre* ID del padre de los elementos del arbol de menu
- *\$accion*

### Función que valida si se visualiza o no la accion especificada (usada en views)

- **Static**
- **Access** public

## graph.php

- **Package** Libreria
- **Sub-Package** Controlador
- **Filesource** [Source Code for this file](#)

## obtenercurp.php

- **Package** Libreria
- **Sub-Package** Controlador
- **Filesource** [Source Code for this file](#)

# tree.php

- **Package** Libreria
- **Sub-Package** Controlador
- **Filesource** [Source Code for this file](#)

## Clase Graph [line [11](#)]

### Controlador Objetos

- **Package** Libreria
- **Sub-Package** Controlador
- **Author** Eliecer

Constructor **void** función Graph::\_\_construct() [line [13](#)]

- **Access** public

**void** función Graph::graph\_init(\$title, \$titulo, \$array, \$label, [\$grafica = ""], [\$nacimiento = ""]) [line [36](#)]

#### Parámetros de la función:

- **string \$title** Titulo de la pagina en el navegador
- **string \$titulo** titulo o nombre a mostrar en la vista al crear el grafico
- **array \$array** datos que se graficaran ver ejemplo
- **array \$label** datos con las etiquetas que se muestran en cada grafica
- **array \$grafica** tipo de grafica puede ser :time, basic, axis, bars, bars-h, stacked, horizontal ó pie
- **string \$nacimiento** si se necesita mostrar fechas enviar la fecha inicial

## Crea una grafica en el lugar que se llame

- **Access** public

```
void función Graph::map([$lugar = "Chiapas"], [$zoom = 6], [$rewrite = 0], [$datos = ""])
```

[line [93](#)]

**Parámetros de la función:**

- **string \$lugar** Especifica el lugar donde se centra el mapa
- **int \$zoom** Especifica el zoom de acercamiento en el mapa
- **boolean \$rewrite** Si se desea que sea una pagina o estar enbebida en otra 0=embebido  
1=pagina
- **array \$datos** datos a mostrar en el mapa

## crea un objeto mapa con la ayuda de la api de google

- **Access** public

## Clase Obtenercurp

[line [11](#)]

**Controlador Objeto**

- **Package** Libreria
- **Sub-Package** Controlador
- **Author** Eliecer

## Obtenercurp::\$estados

```
$estados = array(
    array(
        "AGUASCALIENTES"=>"AS",
        "BAJA CALIFORNIA NTE"=>"BC",
        "BAJA CALIFORNIA NORTE"=>"BC",
        "BAJA CALIFORNIA"=>"BC",
        "BAJA CALIFORNIA SUR"=>"BS",
        "CAMPECHE"=>"CC",
        "COAHUILA"=>"CL",
        "COLIMA"=>"CM",
        "CHIAPAS"=>"CS",
        "CHIHUAHUA"=>"CH",
        "DISTRITO FEDERAL"=>"DF",
        "DURANGO"=>"DG",
        "GUANAJUATO"=>"GT",
        "GUERRERO"=>"GR",
        "HIDALGO"=>"HG",
        "JALISCO"=>"JC",
        "MEXICO"=>"MC",
        "MICHOACAN"=>"MN",
        "MORELOS"=>"MS",
        "NAYARIT"=>"NT",
        "NUEVO LEON"=>"NL",
        "OAXACA"=>"OC",
        "PUEBLA"=>"PL",
        "QUERETARO"=>"QT",
        "QUINTANA ROO"=>"QR",
        "SAN LUIS POTOSI"=>"SP",
        "SINALOA"=>"SL",
        "SONORA"=>"SR",
        "TABASCO"=>"TC",
        "TAMAULIPAS"=>"TS",
        "TLAXCALA"=>"TL",
        "VERACRUZ"=>"VZ",
        "ZACATECAS"=>"ZS",
        "EXTERIOR MEXICANO"=>"SM",
        "NACIDO EN EL EXTRANJERO"=>"NE"
    )) [line 17]
```

**Arreglo con los estados y sus abreviaturas, sirven para el cálculo de la CURP**

- **Access** public

Constructor void función Obtenercurp::\_\_construct() [line [58](#)]

- **Access** public

`echo función Obtenercurp::calculacurp($paterno, $materno, $nombre, $dia, $mes, $year, $sexo, $estado, [$regresar = ""]) [line 280]`

**Parámetros de la función:**

- *string \$paterno* Apellido paterno de la persona
- *string \$materno* Apellido materno
- *string \$nombre* Nombre o nombres
- *int \$dia* Dia de nacimiento
- *int \$mes* Mes de nacimiento
- *int \$year* Año de nacimiento
- *string \$sexo* Sexo
- *string \$estado* Lugar de nacimiento
- *string \$regresar* Tipo de retorno =1 return array !=1 json

## Calcula la curp y el rfc con los datos proporcionados

- **Access** public

`echo función Obtenercurp::calcular_curb($paterno, $materno, $nombre, $dia, $mes, $year, $sexo, $estado, [$regresar = ""]) [line 207]`

**Parámetros de la función:**

- *string \$paterno* Apellido paterno de la persona
- *string \$materno* Apellido materno
- *string \$nombre* Nombre o nombres
- *int \$dia* Dia de nacimiento
- *int \$mes* Mes de nacimiento
- *int \$year* Año de nacimiento
- *string \$sexo* Sexo
- *string \$estado* Lugar de nacimiento
- *string \$regresar* Tipo de retorno =1 return array !=1 json

## Calcula la curp y el rfc con los datos proporcionados

- **Access** public

```
echo función Obtenercurp::curp($paterno, $materno, $nombre, $dia, $mes, $year, $sexo, $estado, [$regresar =  
""]) [line 79]
```

**Parámetros de la función:**

- **string \$paterno** Apellido paterno de la persona
- **string \$materno** Apellido materno
- **string \$nombre** Nombre o nombres
- **int \$dia** Dia de nacimiento
- **int \$mes** Mes de nacimiento
- **int \$year** Año de nacimiento
- **string \$sexo** Sexo
- **string \$estado** Lugar de nacimiento
- **string \$regresar** Tipo de retorno =1 return array !=1 json

## Consulta si la curp existe en la base de datos de la condusef

- **Access** public

# Clase Tree

[line [11](#)]

## Controlador Objeto

- **Package** Libreria
- **Sub-Package** Controlador
- **Author** Eliecer

Constructor **void** función Tree::\_\_construct() [line[13](#)]

- **Access** public

*void función Tree::create(\$title, \$titulo, \$seleccion, \$tipo, \$menu, \$id, \$text, [\$idarbol = 1], [\$nivel = 1],  
[\$omitidos = array(NULL)], [\$seleccionable = ""])* [line [42](#)]

**Parámetros de la función:**

- *string \$title* Titulo de la pagina en el navegador
- *string \$titulo* titulo o nombre a mostrar en la vista al crear el arbol
- *int \$seleccion* tipo de seleccion 1=select. 2=multiselect. 3=multiselect Parcial (Marcan los padres del hijo seleccionado)
- *string \$tipo* tipo de control radio o check
- *boolean \$menu* si se desea mostrar el menu
- *string \$id* id del campo oculto donde se guarda el id del elemento seleccionado
- *string \$text* id del campo donde se muestra la descripcion del elemento seleccionado
- *string \$idarbol* id del arbol donde comenzara la creacion
- *string \$nivel* nivel en el que se empezara a mostrar informacion
- *string \$omitidos* nodos que no se deben mostrar en la vista al crear el arbol
- *string \$seleccionable* determina si un nodo se puede o no seleccionar

### Crea el arbol y lo muestra en la view

- **Access** public



# Paquete Session Elementos procedurales

## Session.php

- **Package** Session
- **Sub-Package** Libraries
- **Filesource** [Source Code for this file](#)

# Paquete Session Clases

## Clase Session

[line [12](#)]

### CodeIgniter Native Session Library

- **Package** Session
- **Sub-Package** Libraries
- **Author** Bo-Yi Wu (appleboy) < [appleboy.tw@gmail.com](mailto:appleboy.tw@gmail.com)>
- **Author** Marko Martinovi? < [marko@techytalk.info](mailto:marko@techytalk.info)>

#### Session::\$ci

*mixed = [line [16](#)]*

- **Access** protected

#### Session::\$flashdata\_key

*mixed = 'flash' [line [18](#)]*

- **Access** protected

### **Session::\$sess\_expiration**

*mixed = " [line [15](#)]*

- **Access** protected

### **Session::\$sess\_namespace**

*mixed = " [line [14](#)]*

- **Access** protected

### **Session::\$store**

*mixed = array() [line [17](#)]*

- **Access** protected

Constructor *void* función **Session::\_\_construct([&\$config = array()])** [line [28](#)]

**Parámetros de la función:**

- *array* **\$config** config preferences

### **Constructor**

- **Access** public

*array* función **Session::all\_userdata()** [line [190](#)]

**Fetch all session data**

- **Access** public

*string* función Session::flashdata(\$key) [line [245](#)]

**Parámetros de la función:**

- *string* \$key

**Fetch a specific flashdata item from the session array**

- **Access** public

*void* función Session::is\_expired() [line [106](#)]

**Check if session is expired**

- **Access** public

*void* función Session::keep\_flashdata(\$key) [line [225](#)]

**Parámetros de la función:**

- *string* \$key

**Keeps existing flashdata available to next request.**

- **Access** public

*void función Session::sess\_create() [line 85]*

### Create Session

- **Access** public

*void función Session::sess\_destroy() [line 19]*

### Destroy session

- **Access** public

*void función Session::set\_flashdata([\$newdata = array()], [\$newval = "]) [line 204]*

#### Parámetros de la función:

- *mixed \$newdata*
- *string \$newval*

### Add or change flashdata, only available until the next request

- **Access** public

*void función Session::set\_userdata([\$data = array()], [\$value = "]) [line 151]*

#### Parámetros de la función:

- *array \$data* list of data to be stored
- *object value \$value* to be stored if only one element is passed

## **Set value for specific user data element**

- **Access** public

*void función Session::unset\_userdata([\$data = array()]) [line [169](#)]*

**Parámetros de la función:**

- **array \$data** list of data to be removed

## **remove array value for specific user data element**

- **Access** public

*object element función Session::userdata(\$value) [line [131](#)]*

**Parámetros de la función:**

- **string \$value** element key

## **Get specific user data element**

- **Access** public



# Paquete SIIGS Elementos procedurales

## ayuda.php

- **Package** SIIGS
- **Sub-Package** Controlador
- **Filesource** [Source Code for this file](#)

## accion.php

- **Package** SIIGS
- **Sub-Package** Controlador
- **Filesource** [Source Code for this file](#)

## bitacora.php

- **Package** SIIGS
- **Sub-Package** Controlador
- **Filesource** [Source Code for this file](#)

## catalogo.php

- **Package** SIIGS
- **Sub-Package** Controlador
- **Filesource** [Source Code for this file](#)

## catalogocsv.php

- **Package** SIIGS
- **Sub-Package** Controlador
- **Filesource** [Source Code for this file](#)

## **catalogo\_x\_raiz.php**

- **Package** SIIGS
- **Sub-Package** Controlador
- **Filesource** [Source Code for this file](#)

## cie10.php

- **Package** SIIGS
- **Sub-Package** Controlador
- **Filesource** [Source Code for this file](#)

## controlador.php

- **Package** SIIGS
- **Sub-Package** Controlador
- **Filesource** [Source Code for this file](#)

## entorno.php

- **Package** SIIGS
- **Sub-Package** Controlador
- **Filesource** [Source Code for this file](#)

## errorlog.php

- **Package** SIIGS
- **Sub-Package** Controlador
- **Filesource** [Source Code for this file](#)

## grupo.php

- **Package** SIIGS
- **Sub-Package** Controlador
- **Filesource** [Source Code for this file](#)

## menu.php

- **Package** SIIGS
- **Sub-Package** Controlador
- **Filesource** [Source Code for this file](#)

## permiso.php

- **Package** SIIGS
- **Sub-Package** Controlador
- **Filesource** [Source Code for this file](#)

## raiz.php

- **Package** SIIGS
- **Sub-Package** Controlador
- **Filesource** [Source Code for this file](#)

## reglavacuna.php

- **Package** SIIGS
- **Sub-Package** Controlador
- **Filesource** [Source Code for this file](#)

## usuario.php

- **Package** SIIGS
- **Sub-Package** Controlador
- **Filesource** [Source Code for this file](#)

# Paquete SIIGS Clases

## Clase Accion

[line [10](#)]

### Controlador Accion

- **Package** SIIGS
- **Sub-Package** Controlador
- **Author** Geovanni

Constructor `void` función Accion::\_\_construct() [[line12](#)]

- **Access** public

`void` función Accion::delete(\$id) [[line246](#)]

**Parámetros de la función:**

- `int $id` Este parámetro no puede ser nulo

**Acción para eliminar una acción, recibe el id de la acción a eliminar**

- **Access** public

*void función Accion::index([\$pag = 0]) [line 85]*

**Parámetros de la función:**

- *int \$pag* Número de registro para el paginador

**Acción por default del controlador, carga la lista**  
de acciones disponibles y una lista de opciones

- **Access** public

*void función Accion::insert() [line 12]*

**Acción para preparar la inserción de nuevas acciones , realiza la validación del formulario del lado cliente**

- **Access** public

*void función Accion::update(\$id) [line 170]*

**Parámetros de la función:**

- *int \$id* Este parámetro no puede ser nulo

**Acción para preparar la actualización de una acción ya existente,**

recibe un ID para obtener los valores de esa acción y mostrarlos en la vista update ,  
realiza la validación del formulario del lado del cliente

- **Access** public

void función Accion::view(\$id) [line 83]

**Parámetros de la función:**

- *int \$id* Este parametro no puede ser nulo

**Acción para visualizar información de una acción específica, obtiene el objeto acción por medio del id proporcionado.**

- **Access** public

## Clase Ayuda

[line 11]

**Controlador Ayuda**

- **Package** SIIGS
- **Sub-Package** Controlador
- **Author** Pascual

Constructor void función Ayuda::\_\_construct() [line 13]

- **Access** public

void función Ayuda::index([\$id\_controlador\_accion = null]) [line 25]

**Parámetros de la función:**

- *int \$id\_controlador\_accion* Id del controlador accion, es opcional

**Función que renderiza el contenido de la ayuda dependiendo de la sección donde se encuentre**

- **Access** public

## Clase Bitacora [line 11]

### Controlador Bitacora

- **Package** SIIGS
- **Sub-Package** Controlador
- **Author** Pascual

Constructor *void* función Bitacora::\_\_construct() [line 13]

- **Access** public

*void* función Bitacora::index([*\$pag* = 0]) [line 35]

**Parámetros de la función:**

- *int \$pag* Establece el desplazamiento del primer registro a devolver

## **Lista todos los registros de la bitacora, con su correspondiente paginación**

permite hacer filtrados por Usuario, Entorno, Controlador, Acción y Fecha, permite eliminar un conjunto de registro o un elemento individual, muestra enlaces para actualizar y ver detalles de un elemento específico

- **Access** public

*void función Bitacora::validateExistUsuario(\$id\_usuario) [line 867]*

**Parámetros de la función:**

- *int \$id\_usuario* ID del usuario

**callback utilizado por las acciones create y update para validar la existencia de un usuario**

- **Access** public

*void función Bitacora::view(\$id) [line 301]*

**Parámetros de la función:**

- *int \$id* ID del elemento a actualizar

**Muestra los datos del registro especificado por el id**

- **Access** public

# Clase Catalogo

[line 10]

## Controlador Catalogo

- **Package** SIIGS
- **Sub-Package** Controlador
- **Author** Geovanni

Constructor *void* función Catalogo::\_\_construct() [line 12]

- **Access** public

*void* función Catalogo::checkpk(\$campos) [line 745]

**Parámetros de la función:**

- *string* \$campos

**Acción para revisar registros repetidos en las columnas designadas como primary key**

- **Access** public

*void* función Catalogo::checkTypeData(\$campo, \$type) [line 776]

**Parámetros de la función:**

- *string* \$campo Nombre del campo a revisar
- *string* \$type Define el tipo de dato del campo

**Acción para revisar si los tipos de datos coinciden con los datos contenidos en la tabla temporal que fueron tomados del CSV**

- **Access** public

*void función Catalogo::delete(\$nombre) [line 805]*

**Parámetros de la función:**

- *string \$nombre*

**Acción para eliminar un catálogo, recibe el nombre del catalogo a eliminar**

- **Access** public

*void función Catalogo::index() [line 85]*

**Acción por default del controlador, carga la lista de catálogos disponibles y una lista de opciones No recibe parámetros**

- **Access** public

*void función Catalogo::insert() [line 548]*

**Acción para preparar la inserción de nuevos catálogos , realiza la validación del formulario del lado del servidor y crea la estructura para el catálogo, crea la tabla y obtiene los datos a partir de la tabla tmp\_catalogo**

- **Access** public

*void función Catalogo::load(0) [line 130]*

**Parámetros de la función:**

- **\$\_FILES[] \$0** archivocsv Variable pasada por POST con el archivo csv para cargar datos

**Acción para cargar datos desde un archivo CSV, recibe el stream desde las variables PHP Guarda en la tabla tmp\_catalogos toda la estructura del CSV e imprime las columnas del archivo. Solo se permite su acceso por medio de peticiones AJAX**

- **Access** public

*void función Catalogo::loadupdate(\$nombrecat, [\$update = false]) [line 230]*

**Parámetros de la función:**

- **\$\_FILES[] \$nombrecat** archivocsv Variable pasada por POST con el archivo csv para cargar datos
- **\$update**

**Acción para cargar datos desde un archivo CSV, recibe el stream desde las variables PHP compara los datos recibidos con los datos que contiene actualmente el catálogo, regresa como resultado las filas nuevas y las filas a modificar.**  
Solo se permite su acceso por medio de peticiones AJAX

- **Access** public

*void función Catalogo::update(\$nombre, [\$pag = 0]) [line 681]*

**Parámetros de la función:**

- *string \$nombre*
- *int \$pag* Numero de registro para el paginador

### Acción para preparar la actualización de un catálogo ya existente,

recibe un string para obtener los valores del catalogo y mostrarlos en la vista update , realiza la validación del formulario del lado del cliente y servidor

- **Access** public

*void función Catalogo::view(\$nombre, [\$pag = 0]) [line 508]*

#### Parámetros de la función:

- *string \$nombre* Este parámetro no puede ser nulo
- *int \$pag* Numero de registro para el paginador

### Acción para visualizar información de un catálogo específico, obtiene el objeto catálogo por medio del nombre proporcionado

- **Access** public

*void función Catalogo::\_array\_unique\_recursive(\$arr) [line 531]*

#### Parámetros de la función:

- *array \$arr* Arreglo multidimensional

**\_array\_unique\_recursive**      Revisa      valores      duplicados      en      arreglos  
multidimensionales

- **Access** public

## Clase CatalogoCsv

[line [10](#)]

### Controlador CatalogoCsv

- **Package** SIIGS
- **Sub-Package** Controlador
- **Author** Geovanni

Constructor *void* función CatalogoCsv::\_\_construct() [line[12](#)]

- **Access** public

Boolean función CatalogoCsv::ActivaEnCatalogo(\$id, \$catalogo, \$activo) [line[403](#)]

#### Parámetros de la función:

- *Int \$id* Es el id del registro en el catalogo
- *String \$catalogo* para determinar a que catalogo se va a agregar o quitar el registro
- *Boolean \$activo* False para quitar del catalogo, true para agregarlo

\* **Accion para activar o desactivar elementos en los catalogos indicados en el parametro Solo se permite su acceso por medio de peticiones AJAX**

- **Access** public

*void función CatalogoCsv::checkpk(\$campos) [line 494]*

**Parámetros de la función:**

- *string \$campos*

**Acción para revisar registros repetidos en las columnas designadas como primary key**

- **Access** public

*void función CatalogoCsv::createTableAgeb() [line 587]*

**Acción para ejecutar la creación de la tabla Asu Ageb No recibe parámetros**

- **Access** public

*void función CatalogoCsv::createTableGeo() [line 555]*

**Acción para ejecutar la creación de la tabla poblacional No recibe parámetros**

- **Access** public

*void función CatalogoCsv::createTableHemoGlobina() [line 620]*

**Acción para ejecutar la creación de la tabla Asu Ageb No recibe parámetros**

- **Access** public

*void función CatalogoCsv::createTablePob() [line 522]*

**Acción para ejecutar la creación de la tabla poblacional No recibe parámetros**

- **Access** public

*void función CatalogoCsv::index() [line 85]*

**Acción por default del controlador, carga la lista de catálogoscsv disponibles y una lista de opciones No recibe parámetros**

- **Access** public

*void función CatalogoCsv::loadupdate(\$nombrecat, [\$update = false]) [line 19]*

**Parámetros de la función:**

- ***\$\_FILES[] \$nombrecat*** archivocsv Variable pasada por POST con el archivo csv para cargar datos
- ***boolean \$update*** Define si se actualizará la DB o solo se hará la primera revisión de datos

**Acción para cargar datos desde un archivo CSV, recibe el stream desde las variables PHP**

compara los datos recibidos con los datos que contiene actualmente el catálogo, regresa como resultado las filas nuevas y las filas a modificar Solo se permite su acceso por medio de peticiones AJAX

- **Access** public

*void función CatalogoCsv::update(\$nombre, [\$pag = 0]) [line 147]*

**Parámetros de la función:**

- *string \$nombre*
- *int \$pag* Número de registro para el paginador

**Acción para preparar la actualización de un catálogo ya existente,**

recibe un string para obtener los valores del catálogo y mostrarlos en la vista update , realiza la validacion del formulario del lado del cliente y servidor

- **Access** public

*void función CatalogoCsv::view(\$nombre, [\$pag = 0]) [line 68]*

**Parámetros de la función:**

- *string \$nombre* Este parametro no puede ser nulo
- *int \$pag* Número de registro para el paginador

**Acción para visualizar información de un catálogo específico, obtiene el objeto catalogocsv por medio del nombre proporcionado**

- **Access** public

*void función CatalogoCsv::\_\_array\_unique\_recursive(\$arr) [line 377]*

**Parámetros de la función:**

- *array \$arr*

**\_\_array\_unique\_recursive** Revisa valores duplicados en arreglos que contienen arreglos

- **Access** public

## Clase Catalogo\_x\_raiz [line 10]

### Controlador Raiz\_x\_Catalogo

- **Package** SIIGS
- **Sub-Package** Controlador
- **Author** Geovanni

Constructor `void` función Catalogo\_x\_raiz::\_\_construct() [line 12]

- **Access** public

`boolean` función Catalogo\_x\_raiz::check(\$id) [line 174]

**Parámetros de la función:**

- `type $id` Id del catalogo\_x\_raiz

**Acción que sirve para revisar inconsistencias en el arbol de segmentacion Recibe como parámetro el catálogo x raiz y revisa que todos los registros tengan un correspondiente en el catálogo padre.**

Solo se permite su acceso por medio de peticiones AJAX

- **Access** public

*void función Catalogo\_x\_raiz::delete(\$id) [line 226]*

**Parámetros de la función:**

- *int \$id*

**Acción para eliminar un catálogo en el arbol, recibe el id del catálogo en la raiz a eliminar**

- **Access** public

*void función Catalogo\_x\_raiz::insert([\$id = 0]) [line 63]*

**Parámetros de la función:**

- **\$id**

**Acción para preparar la inserción de nuevas raices para catálogos , realiza la validación del formulario del lado cliente**

- **Access** public

*void función Catalogo\_x\_raiz::view(\$id) [line 86]*

**Parámetros de la función:**

- *int \$id* Este parametro no puede ser nulo

**Acción para visualizar información de una raiz\_x\_catalogo específica, obtiene el**

**objeto raiz\_x\_catalogo por medio del id proporcionado.**

- **Access** public

## Clase Cie10

*[line 10]*

### Controlador Cie10

- **Package** SIIGS
- **Sub-Package** Controlador
- **Author** Geovanni

Constructor **void** función Cie10::\_\_construct() *[line 12]*

- **Access** public

Boolean función Cie10::ActivaEnCatalogo(\$id, \$catalogo, \$activo) *[line 166]*

**Parámetros de la función:**

- **Int \$id** Es el id del registro en el catalogo
- **String \$catalogo** para determinar a que catalogo se va a agregar o quitar el registro
- **Boolean \$activo** False para quitar del catalogo, true para agregarlo

\* **Accion para activar o desactivar elementos en los catalogos IRA EDA Consultas**  
**Solo se permite su acceso por medio de peticiones AJAX**

- **Access** public

Boolean función Cie10::AgregaEnCatalogo(\$id, \$catalogo, \$activo) [line 21]

**Parámetros de la función:**

- *Int \$id* Es el id del registro en el catalogo de CIE10
- *String \$catalogo* para determinar a que catalogo se va a agregar o quitar el registro
- *Boolean \$activo* False para quitar del catalogo, true para agregarlo

\* **Accion para agregar elementos del catalogo cie10 a los catalogos de EDA, IRA y Consultas dependiendo de los Solo se permite su acceso por medio de peticiones AJAX**

- **Access** public

void función Cie10::index([\$pag = 0]) [line 86]

**Parámetros de la función:**

- *int \$pag* Numero de registro para el paginador

**Acción por default del controlador, carga la lista de datos disponibles en el cie10 y una lista de opciones**

- **Access** public

void función Cie10::insert([\$update = false]) [line 346]

**Parámetros de la función:**

- *\$\_FILES[] \$update* archivocsv Variable pasada por POST con el archivo csv para cargar

datos

**Acción para cargar datos desde un archivo CSV, recibe el stream desde las variables PHP compara los datos recibidos con los datos que contiene actualmente el catálogo, regresa como resultado las filas nuevas y las filas a modificar**  
Solo se permite su acceso por medio de peticiones AJAX

- **Access** public

void función Cie10::load(0) [[line 209](#)]

**Parámetros de la función:**

- **\$\_FILES[]** 0 archivocsv Variable pasada por POST con el archivo csv para cargar datos

**Acción para cargar datos desde un archivo CSV, recibe el stream desde las variables PHP**

Guarda en la tabla tmp\_catalogos toda la estructura del CSV e imprime las columnas del archivo Solo se permite su acceso por medio de peticiones AJAX

- **Access** public

void función Cie10::update(\$id) [[line 266](#)]

**Parámetros de la función:**

- **int \$id**

**Acción para preparar la actualización de un registro del CIE10,**

recibe un id para obtener los valores del catálogo y mostrarlos en la vista update , realiza la validación del formulario del lado del cliente y servidor

- **Access** public

*void función Cie10::view(\$cat) [line 84]*

**Parámetros de la función:**

- *string \$cat* Nombre del catalogo a mostrar

\* **Accion para mostrar información de los catalogos IDE , ERA y Consultas**

- **Access** public

*void función Cie10::\_\_array\_unique\_recursive(\$arr) [line 578]*

**Parámetros de la función:**

- *array \$arr*

**\_array\_unique\_recursive** Revisa valores duplicados en arreglos que contienen arreglos

- **Access** public

## Clase Controlador

[line 10]

## Controlador Controlador

- **Package** SIIGS
- **Sub-Package** Controlador
- **Author** Geovanni

Constructor `void` función Controlador::`__construct()` [line 12]

- **Access** public

`void` función Controlador::`accion($id)` [line 300]

**Parámetros de la función:**

- `int $id`

**Acción para preparar la actualización de acciones asignadas a un controlador, recibe un ID para obtener las acciones asignadas a ese controlador y mostrarlos en la vista update**

- **Access** public

`void` función Controlador::`delete($id)` [line 365]

**Parámetros de la función:**

- `int $id`

**Acción para eliminar un controlador, recibe el id del controlador a eliminar**

- **Access** public

*Object* función Controlador::getGroupPermissions(\$entorno, \$grupo) [line 400]

**Parámetros de la función:**

- *int* \$entorno
- *int* \$grupo

### **Acción para servir un array de objetos con los permisos asignados a**

un entorno y grupo determinados, esta acción solo es accedida por peticiones AJAX y devuelve un objeto JSON Solo se permite su acceso por medio de peticiones AJAX

- **Access** public

*void* función Controlador::help(\$idControladorAccion, \$id\_controlador\_accion) [line 422]

**Parámetros de la función:**

- *int* \$id\_controlador\_accion
- **\$idControladorAccion**

### **Establece el texto de ayuda para el controlador accion**

- **Access** public

*void* función Controlador::index([\$pag = 0]) [line 86]

**Parámetros de la función:**

- *int \$pag* Número de registro para el paginador

**Acción por default del controlador, carga la lista de controladores disponibles y una lista de opciones Recibe un parametro en caso de filtrado por entornos**

- **Access** public

*void función Controlador::insert([\$id = FALSE]) [line 138]*

**Parámetros de la función:**

- **\$id**

**Acción para preparar la insercion de nuevos controladores , realiza la validacion del formulario del lado cliente**

- **Access** public

*void función Controlador::update(\$id) [line 205]*

**Parámetros de la función:**

- *int \$id*

**Acción para preparar la actualización de un controlador ya existente,**

recibe un ID para obtener los valores de ese controlador y mostrarlos en la vista update , realiza la validación del formulario del lado del cliente

- **Access** public

*void función Controlador::view(\$id) [line 109]*

**Parámetros de la función:**

- **int \$id** Este parametro no puede ser nulo

**Acción para visualizar información de un controlador específico, obtiene el objeto controlador por medio del id proporcionado.**

- **Access** public

## Clase Entorno [line 11]

### Controlador Entorno

- **Package** SIIGS
- **Sub-Package** Controlador
- **Author** Geovanni

Constructor *void función Entorno::\_\_construct() [line 13]*

- **Access** public

*void función Entorno::delete(\$id) [line 290]*

**Parámetros de la función:**

- *int \$id*

**Acción para eliminar un entorno, recibe el id del entorno a eliminar**

- **Access** public

*void función Entorno::index() [line 85]*

**Acción por default del controlador, carga la lista de entornos disponibles y una lista de opciones No recibe parámetros**

- **Access** public

*void función Entorno::insert() [line 94]*

**Acción para preparar la inserción de nuevos entornos , realiza la validación del formulario del lado cliente y del lado servidor para evitar entornos duplicados**

- **Access** public

*void función Entorno::update(\$id) [line 208]*

**Parámetros de la función:**

- *int \$id*

**Acción para preparar la actualización de un entorno ya existente,**

recibe un ID para obtener los valores de ese entorno y mostrarlos en la vista update , realiza la validación del formulario del lado del cliente y del servidor para evitar datos duplicados

- **Access** public

*void función Entorno::view(\$id) [/line65]*

**Parámetros de la función:**

- *int \$id* Este parametro no puede ser nulo

**Acción para visualizar de un entorno específico, obtiene el objeto entorno por medio del id proporcionado.**

- **Access** public

*boolean función Entorno::\_\_ExistEntorno(\$nombre\_entorno) [/line153]*

**Parámetros de la función:**

- *string \$nombre\_entorno* Revisa si este valor ya existe como un entorno

**Acción para validar que no exista previamente el entorno a insertar (Esta acción no puede ser accedida desde el navegador)**

- **Access** public

*boolean función Entorno::\_\_ExistEntornoUpdate(\$nombre\_entorno) [/line178]*

**Parámetros de la función:**

- *string \$nombre\_entorno* Revisa si este valor ya existe como un entorno

### **Acción para validar que no exista previamente el entorno a actualizar**

esta acción revisa si el nombre a usar ya existe en la base excepto el mismo objeto a actualizar (Esta acción no puede ser accedida desde el navegador)

- **Access** public

## **Clase Errorlog**

*[line 11]*

### **Controlador Errorlog**

- **Package** SIIGS
- **Sub-Package** Controlador
- **Author** Pascual

Constructor **void** función Errorlog::\_\_construct() *[line 13]*

- **Access** public

**void** función Errorlog::index([**\$pag** = 0]) *[line 84]*

**Parámetros de la función:**

- **int \$pag** Establece el desplazamiento del primer registro a devolver

### **Lista todos los registros de la tabla error, con su correspondiente paginación**

**permite hacer filtrados por Usuario, Entorno, Controlador, Acción y Fecha, muestra enlaces para ver detalles de un elemento específico**

- **Access** public

*void función Errorlog::view(\$id) [line [141](#)]*

**Parámetros de la función:**

- *int \$id* ID del elemento a actualizar

**Muestra los datos del registro especificado por el id**

- **Access** public

## Clase Grupo

*[line [10](#)]*

**Controlador Grupo**

- **Package** SIIGS
- **Sub-Package** Controlador
- **Author** Rogelio

Constructor *void función Grupo::\_\_construct() [line [12](#)]*

- **Access** public

*void función Grupo::delete(\$id) [line 195]*

**Parámetros de la función:**

- *int \$id* id del grupo a eliminar

### **Solicita la eliminación del grupo recibido**

- **Access** public

*void función Grupo::index([\$pag = 0]) [line 33]*

**Parámetros de la función:**

- *int \$pag* número de página a visualizar (paginación)

### **1) Visualiza los grupos existentes para su interacción CRUD 2) En caso de detectar un texto a buscar se filtran los grupos existentes acorde a la búsqueda**

- **Access** public

*void función Grupo::insert() [line 109]*

### **1) Prepara el formulario para la inserción de un grupo nuevo 2) Realiza las validaciones necesarias sobre cada campo del registro**

- **Access** public

*void función Grupo::update(\$id) [line 152]*

**Parámetros de la función:**

- *int \$id* id del grupo a modificar

**1) Prepara el formulario para la modificación de un grupo existente 2) Realiza las validaciones necesarias sobre cada campo del registro**

- **Access** public

*void función Grupo::view(\$id) [line 81]*

**Parámetros de la función:**

- *int \$id* id del grupo a visualizar

**Visualiza los datos del grupo recibido**

- **Access** public

*boolean función Grupo::\_ifGroupExists(\$name) [line 222]*

**Parámetros de la función:**

- *string \$name* nombre del grupo a validar

**Callback para validar que un nombre de grupo no se duplique**

- **Access** public

## Clase Menu

[line [11](#)]

### Controlador Menu

- **Package** SIIGS
- **Sub-Package** Controlador
- **Author** Pascual

Constructor `void` función `Menu::__construct()` [line [21](#)]

- **Access** public

`void` función `Menu::delete($id)` [line [262](#)]

**Parámetros de la función:**

- `int $id` ID del elemento a eliminar

### Eliminar el registro especificado por el id

- **Access** public

`void` función `Menu::index([$pag = 0])` [line [47](#)]

**Parámetros de la función:**

- *int \$pag* Establece el desplazamiento del primer registro a devolver

### **Lista todos los registros de la menu, con su correspondiente paginación**

permite hacer filtrados por Usuario, Entorno, Controlador, Acción y Fecha, permite eliminar un conjunto de registro o un elemento individual, muestra enlaces para actualizar y ver detalles de un elemento específico

- **Access** public

*void función Menu::insert([\$id = null]) [line 130]*

**Parámetros de la función:**

- **\$id**

**Muestra el formulario para crear un nuevo registro en la menu, las variables se obtienen por el metodo POST**

- **Access** public

*void función Menu::update(\$id) [line 225]*

**Parámetros de la función:**

- *int \$id* ID del elemento a actualizar

**Muestra el formulario con los datos del registro especificado por el id, para actualizar sus datos**

- **Access** public

void función Menu::view(\$id) [line [326](#)]

**Parámetros de la función:**

- *int \$id* ID del elemento a actualizar

**Muestra los datos del registro especificado por el id**

- **Access** public

## Clase Permiso

[line [10](#)]

**Controlador Permiso**

- **Package** SIIGS
- **Sub-Package** Controlador
- **Author** Rogelio

Constructor void función Permiso::\_\_construct() [line [12](#)]

- **Access** public

*void función Permiso::index(\$id) [line 84]*

**Parámetros de la función:**

- *int \$id* id del grupo del cual se obtendrán (y actualizar si aplica) los permisos asignados

**1) Visualiza los entornos existentes para su selección**

2) Al seleccionar un entorno se obtienen los controladores\_x\_accion existentes y se indica sobre cuales el grupo tiene permisos asignados 3) Elimina los permisos asignados al grupo anteriormente e inserta los asignados recientemente

- **Access** public

## Clase Raiz

*[line 11]*

### Controlador Raiz

- **Package** SIIGS
- **Sub-Package** Controlador
- **Author** Geovanni

Constructor *void función Raiz::\_\_construct() [line 13]*

- **Access** public

*void función Raiz::createasu(\$id) [line 316]*

**Parámetros de la función:**

- *int \$id*

**Acción para crear el ASU a partir de una raiz Solo se permite su acceso por medio de peticiones AJAX**

- **Access** public

*void función Raiz::delete(\$id) [line 284]*

**Parámetros de la función:**

- *int \$id*

**Acción para eliminar una raiz, recibe el id de la raiz a eliminar**

- **Access** public

*Object función Raiz::getChildrenFromLevel(\$idarbol, \$nivel, \$claves) [line 632]*

**Parámetros de la función:**

- *Int \$idarbol* parametro pasado por POST y determina el arbol a consultar
- *Int \$nivel* parametro pasado por POST y determina el nivel superior a desglosar en el arbol
- *Array \$claves* Este parametro es pasado por POST y es la lista de valores a preseleccionar en el arbol

**Accion para regresar el arbol de segmentacion determinado, el objeto regresado contiene estructura de arbol y es consumida solamente por peticiones AJAX**

- **Access** public

*Object* función Raiz::getDataKeyValue(\$idarbol, \$nivel, [\$filtro = 0]) [line [738](#)]

**Parámetros de la función:**

- *int \$idarbol*
- *Int \$nivel* Nivel de desglose de información requerida
- *Int \$filtro* (Opcional) filtrar por un valor determinado

**Accion para obtener los registros de un ASU determinado en cierto nivel y con un ID de filtro**

- **Throws** Exception Si ocurre error al recuperar datos de la base de datos
- **Access** public

*Object* función Raiz::getDataTreeFromId(\$claves, \$desglose) [line [596](#)]

**Parámetros de la función:**

- *Array \$claves* Este parametro es pasado por POST y es la lista de valores a consultar
- *Int \$desglose* parametro pasado por POST y determina si se requiere información adicional

**Accion para regresar la descripción e informacion adicional de un arreglo de ID's desde el arbol de segmentacion**

- **Access** public

*Object* función Raiz::getTreeBlock(\$idarbol, \$nivel, \$seleccionados, \$seleccionable, \$seleccionables, \$omitidos) [line [678](#)]

**Parámetros de la función:**

- *int \$idarbol* ID del arbol (el arbol usado por la TES es el 1)
- *int \$nivel* nivel del arbol que se desea obtener

- *array \$seleccionados* se especifica si dentro del arreglo de retorno, hay valores preseleccionados
- *bool \$seleccionable* especifica si los elementos del arbol pueden ser seleccionados
- *array \$seleccionables* especifica que niveles del arbol pueden ser seleccionados
- *array \$omitidos* especifica niveles omitidos dentro del arbol (Si hay un nivel intermedio omitido, los hijos de este nivel son agregados como hijos de su nivel inmediato superior)

### **Sirve para obtener bloques del arbol de segmentación única ASU**

solo se puede acceder por peticiones AJAX, los parametros son pasados por GET

- **Access** public

*void función Raiz::index() [line62]*

**Acción por default del controlador, carga la lista de Raices disponibles y una lista de opciones No recibe parámetros**

- **Access** public

*void función Raiz::iniciarasu(\$id) [line88]*

**Parámetros de la función:**

- *type \$id* ID del asu

### **Crea los archivos JSON necesarios para iniciar el ASU en caché y agilizar su carga**

- **Access** public

*void función Raiz::insert() [line134]*

**Acción para preparar la inserción de nuevas acciones , realiza la validación del formulario del lado cliente**

- **Access** public

*void función Raiz::update(\$id) [line<sup>188</sup>]*

**Parámetros de la función:**

- *int \$id*

**Acción para preparar la actualización de una raiz ya existente,**

recibe un ID para obtener los valores de esa raiz y mostrarlos en la vista update , realiza la validación del formulario del lado del cliente

- **Access** public

*void función Raiz::updateasu(\$id) [line<sup>439</sup>]*

**Parámetros de la función:**

- *int \$id*

**Acción para actualizar el ASU a partir de una raiz Solo se permite su acceso por medio de peticiones AJAX**

- **Access** public

*void función Raiz::view(\$id) [line<sup>94</sup>]*

**Parámetros de la función:**

- *int \$id* Este parametro no puede ser nulo

**Acción para visualizar información de una raiz específica, obtiene el objeto raiz por medio del id proporcionado.**

- **Access** public

## Clase ReglaVacuna [line 10]

### Controlador ReglaVacuna

- **Package** SIIGS
- **Sub-Package** Controlador
- **Author** Geovanni

Constructor *void* función ReglaVacuna::\_\_construct() [line 12]

- **Access** public

*void* función ReglaVacuna::delete(\$id) [line 833]

**Parámetros de la función:**

- *int \$id*

**Acción para eliminar una regla, recibe el id de la regla a eliminar**

- **Access** public

*void función ReglaVacuna::index() [line 85]*

**Acción por default del controlador, carga la lista de reglas de vacunas disponibles y una lista de opciones No recibe parámetros**

- **Access** public

*void función ReglaVacuna::insert() [line 95]*

**Acción para preparar la insercion de nuevas reglas , realiza la validación del formulario del lado cliente**

- **Access** public

*void función ReglaVacuna::update(\$id) [line 203]*

**Parámetros de la función:**

- *int \$id*

**Acción para preparar la actualización de una regla ya existente,**

recibe un ID para obtener los valores de esa regla y mostrarlos en la vista update , realiza la validación del formulario del lado del cliente

- **Access** public

void función ReglaVacuna::view(\$id) [[line 66](#)]

**Parámetros de la función:**

- *int \$id* Este parametro no puede ser nulo

**Acción para visualizar información de una regla específica, obtiene el objeto regla\_vacuna por medio del id proporcionado.**

- **Access** public

## Clase Usuario

[[line 10](#)]

**Controlador Usuario**

- **Package** SIIGS
- **Sub-Package** Controlador
- **Author** Rogelio

**Usuario::\$mas**

*mixed = ""* [[line 773](#)]

**Acción para hacer autologin en otro sistema**

- **Access** public

Constructor void función Usuario::\_\_construct() [line 12]

- **Access** public

void función Usuario::automatic\_access() [line 774]

- **Access** public

redirect función Usuario::cerrar\_etab() [line 809]

**Acción para cerrar el login en otro sistema**

void función Usuario::delete(\$id) [line 825]

**Parámetros de la función:**

- *int \$id* id del usuario a eliminar

**Solicita la eliminación del usuario recibido**

- **Access** public

void función Usuario::form\_init() [line 884]

Object función Usuario::getActivesByGroup(\$grupo) [line 752]

**Parámetros de la función:**

- *int \$grupo*

**Acción para servir un array de objetos con los usuarios activos por grupo AJAX y**

## **devuelve un objeto JSON**

- **Access** public

*cookies* función Usuario::get\_galleta() [[line 864](#)]

### **Obtiene las cookies que se generan en la session**

*token* función Usuario::get\_token(\$url, \$var, [\$valor = ""], [\$num = ""], [\$count = ""], [\$par = ""]) [[line 820](#)]

#### **Parámetros de la función:**

- **\$url**
- **\$var**
- **\$valor**
- **\$num**
- **\$count**
- **\$par**

## **Acción para obtener el token oculto en el login**

*void* función Usuario::index([\$pag = 0]) [[line 126](#)]

#### **Parámetros de la función:**

- **int \$pag** número de página a visualizar (paginación)

**1) Visualiza los usuarios existentes para su interacción CRUD 2) En caso de detectar un texto a buscar se filtran los usuarios existentes acorde a la búsqueda**

- **Access** public

*void* función Usuario::insert() [[line 197](#)]

**1) Prepara el formulario para la inserción de un usuario nuevo    2) Realiza las validaciones necesarias sobre cada campo del registro**

- **Access** public

*void función Usuario::load\_update() [line 655]*

- **Access** public

*void función Usuario::login() [line 83]*

**Ofrece el inicio de sesión**

- **Access** public

*void función Usuario::logout() [line 108]*

**Termina la sesión**

- **Access** public

*void función Usuario::remember() [line 893]*

- **Access** public

*void función Usuario::reset() [line 588]*

- **Access** public

*void función Usuario::send\_mail(\$subject, \$body, \$from, \$rto, \$correo, \$CC, \$CCO, \$adj) [line 556]*

**Parámetros de la función:**

- **\$subject**
- **\$body**
- **\$from**
- **\$rto**
- **\$correo**
- **\$CC**
- **\$CCO**
- **\$adj**

- **Access** public

*void función Usuario::token(\$str) [line 518]*

**Parámetros de la función:**

- **\$str**

- **Access** public

*void función Usuario::update(\$id) [line 862]*

**Parámetros de la función:**

- *int \$id* id del usuario a modificar

**1) Prepara el formulario para la modificación de un usuario existente 2) Realiza las validaciones necesarias sobre cada campo del registro**

- **Access** public

*void función Usuario::update\_info() [line 684]*

- **Access** public

*void función Usuario::view(\$id) [line 170]*

**Parámetros de la función:**

- *int \$id* id del usuario a visualizar

### **Visualiza los datos del usuario recibido**

- **Access** public

*boolean función Usuario::\_ifUserExists(\$username) [line 952]*

**Parámetros de la función:**

- *string \$username* nombre de usuario a validar

### **Callback para validar que un nombre de usuario no se duplique**

- **Access** public

## accion\_model.php

- **Package** SIIGS
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## ageb\_model.php

- **Package** SIIGS
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## arbolsegmentacion\_model.php

- **Package** SIIGS
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## bitacora\_model.php

- **Package** SIIGS
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## catalogocsv\_model.php

- **Package** SIIGS
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## catalogo\_model.php

- **Package** SIIGS
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## catalogo\_x\_raiz\_model.php

- **Package** SIIGS
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## cie10\_model.php

- **Package** SIIGS
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## controladoraccion\_model.php

- **Package** SIIGS
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## controlador\_model.php

- **Package** SIIGS
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## entorno\_model.php

- **Package** SIIGS
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## errorlog\_model.php

- **Package** SIIGS
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## georeferencia\_model.php

- **Package** SIIGS
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## grupo\_model.php

- **Package** SIIGS
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## hemoglobina\_model.php

- **Package** SIIGS
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## menu\_model.php

- **Package** SIIGS
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## permiso\_model.php

- **Package** SIIGS
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## poblacion\_model.php

- **Package** SIIGS
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## **raiz\_model.php**

- **Package** SIIGS
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## reglavaduna\_model.php

- **Package** SIIGS
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## usuario\_model.php

- **Package** SIIGS
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## Clase Accion\_model [line 11]

### Modelo Accion

- **Package** SIIGS
- **Sub-Package** Modelo
- **Author** Geovanni

Constructor `void` función Accion\_model::\_\_construct() [line 129]

- **Access** public

`boolean` función Accion\_model::delete() [line 268]

**Elimina el registro actual de la base de datos**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

`ArrayObject` función Accion\_model::getAll() [line 145]

## Devuelve todos los registros de la tabla acciones

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*Object* función Accion\_model::getById(\$id) [\[line 192\]](#)

**Parámetros de la función:**

- *int \$id* ID (Llave primaria)

## Devuelve la información de una accion por su ID

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*void* función Accion\_model::getDescripcion() [\[line 79\]](#)

- **Access** public

*void* función Accion\_model::getId() [\[line 64\]](#)

\*\*\*\*\*

- **Access** public

*void* función Accion\_model::getMetodo() [\[line 87\]](#)

- **Access** public

*string/boolean* función Accion\_model::getMsgError([*\$value* = 'usr'], *\$value*,) [[line114](#)]  
**Parámetros de la función:**

- *string \$value*, default 'usr' (Tipo mensaje)
- *\$value*

**Devuelve los mensajes de error en caso de ocurrir alguna excepción 'usr' devuelve el mensaje para la vista de usuario 'log' devuelve el mensaje para el log de errores**

- **Access** public

*void* función Accion\_model::getNombre() [[line72](#)]

- **Access** public

*int* función Accion\_model::getNumRows() [[line171](#)]

**Devuelve el numero de registros**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*int* función Accion\_model::insert() [[line213](#)]

**Inserta en la tabla accion la información contenida en el objeto**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

void función Accion\_model::setDescripcion(\$value) [[line 83](#)]

**Parámetros de la función:**

- **\$value**
- **Access** public

void función Accion\_model::setId(\$value) [[line 68](#)]

**Parámetros de la función:**

- **\$value**
- **Access** public

void función Accion\_model::setMetodo(\$value) [[line 91](#)]

**Parámetros de la función:**

- **\$value**
- **Access** public

void función Accion\_model::setNombre(\$value) [[line 75](#)]

**Parámetros de la función:**

- **\$value**

- **Access** public

void función Accion\_model::setOffset(\$value) [line 95]

**Parámetros de la función:**

- **\$value**

- **Access** public

void función Accion\_model::setRows(\$value) [line 98]

**Parámetros de la función:**

- **\$value**

- **Access** public

boolean función Accion\_model::update() [line 240]

**Actualiza el objeto actual en la base de datos**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

# Clase Ageb\_model

[line 11]

## Modelo Ageb

- **Package** SIIGS
- **Sub-Package** Modelo
- **Author** Geovanni

Constructor `void` función Ageb\_model::\_\_construct() [line 36]

- **Access** public

`boolean|string` función Ageb\_model::getMsgError([`$type = 'usr'`]) [line 60]

**Parámetros de la función:**

- `string $type` usr si se quiere devolver el mensaje de error a mostrar en la vista, log obtiene el mensaje de error con mas detalles para depuración, valor por defecto usr

**Devuelve el mensaje de error, en caso de existir un error despues de ejecutar un metodo, de lo contrario false**

- **Access** public

`boolean` función Ageb\_model::process() [line 82]

**Inserta los registros contenidos en la tabla cat\_poblacion a la tabla asu\_poblacion**

- **Access** public

*Object* función Ageb\_model::searchageb(\$idlocalidad, \$like) [[line 193](#)]

**Parámetros de la función:**

- *int \$idlocalidad* Id del ASU de la localidad
- *\$like*

**Devuelve una lista de AGEBS de la localidad pasada como parámetro**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*Object* función Ageb\_model::searchUM(\$idlocalidad, \$ageb) [[line 162](#)]

**Parámetros de la función:**

- *int \$idlocalidad* Id del ASU de la localidad
- *string \$ageb* Ageb

**Devuelve la información de una UM de acuerdo a su localidad y ageb**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

# Clase ArbolSegmentacion\_model

[line 11]

## Modelo ArbolSegmentacion

- **Package** SIIGS
- **Sub-Package** Modelo
- **Author** Geovanni

Constructor **void** función ArbolSegmentacion\_model::\_\_construct() [line 45]

- **Access** public

Array función ArbolSegmentacion\_model::convertType(\$arbol, \$seleccionable, \$seleccionados) [line 398]

### Parámetros de la función:

- **Array \$arbol** fila array de valores
- **bool \$seleccionable** Seleccionable opcion para que este elemento sea seleccionable
- **\$seleccionados**

## Convierte el tipo de arreglo para enviarlo como se debe recibir en el cliente de Javascript

- **Access** public

object con función ArbolSegmentacion\_model::getById(\$item) [line 372]

### Parámetros de la función:

- **int \$item** item seleccionado

## Obtener el elemento del ASU por medio de su ID

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*Object* función ArbolSegmentacion\_model::getChildrenFromId(\$id, [\$omitidos = array()]) [line [831](#)]

**Parámetros de la función:**

- *int \$id* Id del elemento en el ASU
- *Array \$omitidos* int \$omitidos Array de niveles omitidos

**Accion para devolver los hijos de un elemento en el ASU a partir de su ID**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*Object* función ArbolSegmentacion\_model::getChildrenFromLevel(\$idarbol, \$nivel, [\$omitidos = array()], [\$seleccionados = array()], [\$seleccionables = array()]) [line [706](#)]

**Parámetros de la función:**

- *Int \$idarbol* Id del arbol a crear
- *Int \$nivel* Nivel de segmentacion desde la cual se desarrolla el arbol
- *Array \$omitidos* int \$omitidos Array de niveles omitidos
- *Array \$seleccionados* int \$seleccionados Array de elementos seleccionados
- *Array \$seleccionables* int \$seleccionables Array de niveles que son seleccionables

**Accion para devolver el esquema completo del ASU a partir de un nivel especificado, niveles omitidos y elementos preseleccionados**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*void* función ArbolSegmentacion\_model::getCluesFromId(\$id) [[line 299](#)]

**Parámetros de la función:**

- *int \$id* Id del elemento en el asu
  - \* @return Object un arreglo con la estructura del arbol

\*

Obtiene las unidades medicas correspondientes a un ID de un elemento independientemente su nivel en el ASU

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*Object* función ArbolSegmentacion\_model::getDataKeyValue(\$idarbol, \$nivel, [\$filtro = 0]) [[line 1031](#)]

**Parámetros de la función:**

- *int \$idarbol* Id del arbol a buscar
- *Int \$nivel* Nivel de desglose de información requerida
- *Int \$filtro* (Opcional) filtrar por un valor determinado

**Accion para obtener los registros de un ASU determinado en cierto nivel y con un ID de filtro**

- **Throws** Exception Si ocurre error al recuperar datos de la base de datos
- **Access** public

*Object* función ArbolSegmentacion\_model::getDescripcionById(\$claves, [\$desglose = 0]) [[line 935](#)]

**Parámetros de la función:**

- *Int \$desglose* Nivel de desglose de información requerida
- *Array \$claves* int \$claves arreglo de valores a recuperar

### Accion para obtener la descripcion e información adicional del elemento en el ASU

- **Throws** Exception Si ocurre error al recuperar datos de la base de datos
- **Access** public

void función ArbolSegmentacion\_model::getListChildrenLevel(\$resultadotemp, [\$clave2 = 0]) [line 640]

#### Parámetros de la función:

- **\$resultadotemp**
- **\$clave2**

- **Access** public

string/boolean función ArbolSegmentacion\_model::getMsgError([\$value = 'usr'], \$value,) [line 30]

#### Parámetros de la función:

- **string \$value**, default 'usr' (Tipo mensaje)
- **\$value**

Devuelve los mensajes de error en caso de ocurrir alguna excepción 'usr' devuelve el mensaje para la vista de usuario 'log' devuelve el mensaje para el log de errores

- **Access** public

*Object* función ArbolSegmentacion\_model::getTree(\$idarbol, \$nivel, [\$nivelesocultos = array()]) [line 123]

**Parámetros de la función:**

- *int \$idarbol* ID del arbol a crear
- *int \$nivel* Nivel de segmentacion desde el cual se iniciará a desarrollar el arbol
- *\$nivelesocultos \$nivelesocultos* Array con los niveles que se deben ocultar

**Regresa el objeto del arbol de segmentacion**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*Object* función ArbolSegmentacion\_model::getTreeBlock(\$idarbol, \$nivel, [\$seleccionados = array()], \$seleccionable, \$elegido, [\$omitidos = array()], [\$seleccionables = array()]) [line 441]

**Parámetros de la función:**

- *Int \$idarbol* Id del arbol a crear
- *Int \$nivel* Nivel de segmentacion desde la cual se desarrolla el arbol
- *Array \$seleccionados* int \$omitidos Array de niveles omitidos
- *Array \$seleccionable* int \$seleccionados Array de elementos seleccionados
- *Array \$elegido* int \$seleccionables Array de niveles que son seleccionables
- *Array \$omitidos* int \$elegido Clave seleccionada para obtener sus hijos
- *\$seleccionables*

**Accion para devolver un bloque del ASU a partir de un nivel especificado o una clave seleccionada, niveles omitidos y elementos preseleccionados**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*Object* función ArbolSegmentacion\_model::getTreeBlockData(\$idarbol, \$nivel, \$elegido) [line 221]

**Parámetros de la función:**

- *int \$idarbol* ID del arbol a crear
- *int \$nivel* Nivel de segmentacion desde el cual se iniciará a desarrollar el arbol
- *\$nivelesocultos \$elegido* Array con los niveles que se deben ocultar

**Regresa el objeto del arbol de segmentacion por nivel o hijos de un elemento seleccionado**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*Object* función ArbolSegmentacion\_model::getUMParentsById(\$clave) [[line 63](#)]

**Parámetros de la función:**

- *int \$clave* Clave de la unidad medica u elemento en el ASU

**Regresa la información de los padres de una unidad medica en el ASU**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*void* función ArbolSegmentacion\_model::\_addSelectedItems(\$datos, \$nivel, \$seleccionados, \$seleccionables) [[line 863](#)]

**Parámetros de la función:**

- **\$datos**
- **\$nivel**
- **\$seleccionados**
- **\$seleccionables**

- **Access** public

*void función ArbolSegmentacion\_model::\_\_addSelectedItems\_(\$datos, \$seleccionados, \$nivel, \$omitidos, \$seleccionables) [line 894]*

**Parámetros de la función:**

- **\$datos**
- **\$seleccionados**
- **\$nivel**
- **\$omitidos**
- **\$seleccionables**

- **Access** public

## Clase Bitacora\_model

*[line 11]*

**Modelo Bitacora**

- **Package** SIIGS
- **Sub-Package** Modelo
- **Author** Pascual

Constructor *void función Bitacora\_model::\_\_construct() [line 93]*

- **Access** public

*void/boolean función Bitacora\_model::addFilter(\$columna, \$condicion, \$valor) [line 998]*

**Parámetros de la función:**

- *string \$columna* Puede ser cualquier campo del objeto (id, id\_usuario, fecha\_hora, parametros, id\_controlador\_accion)
- *string \$condicion* Establece la condición a evaluar, entre los valores permitidos están: =, !=, >=, <=, like
- *string \$valor* Valor contra el cual se realizará la evaluación del campo

## Agrega una nueva regla de filtrado al arreglo de filtros

- **Access** public

*int función Bitacora\_model::delete([\$id = null]) [line 278]*

### Parámetros de la función:

- *int \$id* Si no se establece el valor de ID, se toma el valor del objeto actual

## Elimina el registro actual de la base de datos

- **Access** public

*int función Bitacora\_model::deleteByFilter() [line 317]*

## Elimina el conjunto de registros que cumplen con el o los criterios de filtrado

- **Access** public

*array función Bitacora\_model::getAll([\$offset = null], [\$row\_count = null]) [line 467]*

### Parámetros de la función:

- *int \$offset* Establece el desplazamiento del primer registro a devolver, si se define solo el valor de offset el valor especifica el número de registros a retornar desde el comienzo del conjunto de resultados.
- *int \$row\_count* Establece la cantidad de registros a devolver

**Obtiene todos los registros de la tabla Bitacora en caso de existir filtros, estos son aplicados a la consulta**

- **Access** public

*object|boolean* función Bitacora\_model::getById(*\$id*) [[line 349](#)]

**Parámetros de la función:**

- *int \$id* Si no se establece el valor de ID, se toma el valor del objeto actual

**Obtiene los datos del registro de la bitacora que tiene el ID especificado**

- **Access** public

*void* función Bitacora\_model::getFecha\_hora() [[line 134](#)]

- **Access** public

*void* función Bitacora\_model::getId() [[line 114](#)]

- **Access** public

*void función Bitacora\_model::getId\_controlador\_accion() [line 154]*

- **Access** public

*void función Bitacora\_model::getId\_usuario() [line 124]*

- **Access** public

*boolean/string función Bitacora\_model::getMsgError([\$type = 'usr']) [line 185]*

**Parámetros de la función:**

- *string \$type* usr si se quiere devolver el mensaje de error a mostrar en la vista, log obtiene el mensaje de error con mas detalles para depuración, valor por defecto usr

**Devuelve el mensaje de error, en caso de existir un error despues de ejecutar un metodo, de lo contrario false**

- **Access** public

*int función Bitacora\_model::getNumRows() [line 514]*

**Obtiene el numero total de registros en la tabla Bitacora en caso de existir filtros, estos son aplicados a la consulta**

- **Access** public

*void función Bitacora\_model::getParametros() [line 144]*

- **Access** public

*boolean función Bitacora\_model::insert(\$path, \$parametros) [line 208]*

**Parámetros de la función:**

- *string \$path* Concatenación de directorio del proyecto, clase y metodo, unidos por dos dobles puntos '::'
- *string \$parametros*

### **Inserta a la bitacora la información proporcionada**

- **Static**
- **Access** public

*void función Bitacora\_model::resetFilter() [line 451]*

### **Elimina todos los filtros registrados**

- **Access** public

*void función Bitacora\_model::setFecha\_hora(\$fecha\_hora) [line 139]*

**Parámetros de la función:**

- **\$fecha\_hora**

- **Access** public

*void función Bitacora\_model::setId(\$id) [/line119]*

**Parámetros de la función:**

- **\$id**

- **Access** public

*void función Bitacora\_model::setId\_accion(\$id\_accion) [/line169]*

**Parámetros de la función:**

- **\$id\_accion**

- **Access** public

*void función Bitacora\_model::setId\_controlador(\$id\_controlador) [/line164]*

**Parámetros de la función:**

- **\$id\_controlador**

- **Access** public

*void función Bitacora\_model::setId\_controlador\_accion(\$id\_controlador\_accion) [/line159]*

**Parámetros de la función:**

- **\$id\_controlador\_accion**

- **Access** public

*void función Bitacora\_model::setId\_usuario(\$id\_usuario) [line 129]*

**Parámetros de la función:**

- **\$id\_usuario**

- **Access** public

*void función Bitacora\_model::setParametros(\$parametros) [line 149]*

**Parámetros de la función:**

- **\$parametros**

- **Access** public

*boolean función Bitacora\_model::update([\$id = null]) [line 241]*

**Parámetros de la función:**

- *int \$id* Si no se establece el valor de ID, se toma el valor del objeto actual

## **Actualiza los datos del objeto actual**

- **Access** public

# Clase CatalogoCsv\_model

[line 11]

## Modelo CatalogoCsv

- **Package** SIIGS
- **Sub-Package** Modelo
- **Author** Geovanni

Constructor `void` función CatalogoCsv\_model::\_\_construct() [line 128]

- **Access** public

`boolean` función CatalogoCsv\_model::activaEnCatalogo(\$id, \$catalogo, \$valor) [line 255]

### Parámetros de la función:

- `int $id` el id del registro en el catalogo
- `string $catalogo` nombre del catalogo donde se realizara la operacion
- `boolean $valor` agregar o eliminar el registro del catalogo

## Accion para activar o desactivar registros de catalogos como el de EDA, IRA y Consultas

- **Throws** Exception Si ocurre algun error al consultar y modificar la base de datos
- **Access** public

`boolean` función CatalogoCsv\_model::checkPk(\$campo) [line 282]

### Parámetros de la función:

- *string \$campo* (varios campos delimitados por | )

**Revisa en la base de datos por registros duplicados en los campos pasados por parametro**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*ArrayObject* función CatalogoCsv\_model::getAll() [[line 165](#)]

**Devuelve una lista con los catalogos existentes en la DB**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*ArrayObject* función CatalogoCsv\_model::getAllData(\$nombrecat) [[line 86](#)]

**Parámetros de la función:**

- *string \$nombrecat* Nombre del catalogo

**Devuelve los datos de un catalogo pasado como parametro**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*Object* función CatalogoCsv\_model::getByName(\$nombre) [[line 213](#)]

**Parámetros de la función:**

- *string \$nombre* (Nombre del catalogo)

### **Devuelve la informacion de un catalogo por su nombre**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*void función CatalogoCsv\_model::getCampos() [line 79]*

- **Access** public

*void función CatalogoCsv\_model::getId() [line 64]*  
\*\*\*\*\*

- **Access** public

*void función CatalogoCsv\_model::getLLave() [line 86]*

- **Access** public

*string/boolean función CatalogoCsv\_model::getMsgError([\$value = 'usr'], \$value,)* [line 113]  
**Parámetros de la función:**

- *string \$value*, default 'usr' (Tipo mensaje)
- *\$value*

### **Devuelve los mensajes de error en caso de ocurrir alguna excepción 'usr' devuelve**

**el mensaje para la vista de usuario 'log' devuelve el mensaje para el log de errores**

- **Access** public

*void función CatalogoCsv\_model::getNombre() [line<sup>72</sup>]*

- **Access** public

*int función CatalogoCsv\_model::getNumRows(\$nombre) [line<sup>45</sup>]*

**Parámetros de la función:**

- *string \$nombre* Nombre del catalogo

**Devuelve el numero de registros**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*void función CatalogoCsv\_model::setCampos(\$value) [line<sup>82</sup>]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función CatalogoCsv\_model::setId(\$value) [line 68]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función CatalogoCsv\_model::setLlave(\$value) [line 89]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función CatalogoCsv\_model::setNombre(\$value) [line 75]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función CatalogoCsv\_model::setOffset(\$value) [line 93]*

**Parámetros de la función:**

- **\$value**

- **Access** public

void función CatalogoCsv\_model::setRows(\$value) [[line 96](#)]

**Parámetros de la función:**

- **\$value**

- **Access** public

## Clase Catalogo\_model [[line 11](#)]

**Modelo Catalogo**

- **Package** SIIGS
- **Sub-Package** Modelo
- **Author** Geovanni

Constructor void función Catalogo\_model::\_\_construct() [[line 128](#)]

- **Access** public

boolean función Catalogo\_model::checkPk(\$campo) [[line 254](#)]

**Parámetros de la función:**

- *string \$campo* (varios campos delimitados por | )

**Revisa en la base de datos por registros duplicados en los campos pasados por parametro**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*boolean función Catalogo\_model::checkTypeData(\$campo, \$type) [line 287]*

**Parámetros de la función:**

- *string \$campo* (varios campos delimitados por | )
- *\$type*

**Revisa en la base de datos por registros que no coincidan con el tipo de dato pasado como parametro en el campo indicado**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*boolean función Catalogo\_model::delete() [line 386]*

**Elimina el registro actual de la base de datos**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*ArrayObject función Catalogo\_model::getAll() [line 144]*

**Devuelve una lista con los catalogos existentes en la DB**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*ArrayObject* función Catalogo\_model::getAllData(\$nombrecat) [[line 186](#)]

**Parámetros de la función:**

- *string* **\$nombrecat** Nombre del catalogo

**Devuelve los datos de un catalogo pasado como parametro**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*Object* función Catalogo\_model::getByName(\$nombre) [[line 213](#)]

**Parámetros de la función:**

- *string* **\$nombre** (Nombre del catalogo)

**Devuelve la informacion de un catalogo por su nombre**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*void* función Catalogo\_model::getCampos() [[line 79](#)]

- **Access** public

```
void función Catalogo_model::getId() [line 64]
*****
```

- **Access** public

```
void función Catalogo_model::getLLave() [line 86]
```

- **Access** public

*string/boolean función Catalogo\_model::getMsgError([\$value = 'usr'], \$value,) [line 113]*  
**Parámetros de la función:**

- **string \$value**, default 'usr' (Tipo mensaje)
- **\$value**

**Devuelve los mensajes de error en caso de ocurrir alguna excepción 'usr' devuelve el mensaje para la vista de usuario 'log' devuelve el mensaje para el log de errores**

- **Access** public

```
void función Catalogo_model::getNombre() [line 72]
```

- **Access** public

*int* función Catalogo\_model::getNumRows(\$nombre) [[line 165](#)]

**Parámetros de la función:**

- *string \$nombre* Nombre del catalogo

### Devuelve el numero de registros

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*boolean* función Catalogo\_model::insert(\$create, \$select) [[line 829](#)]

**Parámetros de la función:**

- *string \$create* (la consulta para crear el catalogo)
- *string \$select* (la consulta para extraer datos de la tabla tmp\_catalogo)

### Inserta en la base datos el catálogo y obtiene los datos de la tabla temporal

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*void* función Catalogo\_model::setCampos(\$value) [[line 82](#)]

**Parámetros de la función:**

- **\$value**
- **Access** public

*void función Catalogo\_model::setId(\$value) [line 68]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Catalogo\_model::setLlave(\$value) [line 89]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Catalogo\_model::setNombre(\$value) [line 75]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Catalogo\_model::setOffset(\$value) [line 94]*

**Parámetros de la función:**

- **\$value**

- **Access** public

void función Catalogo\_model::setRows(\$value) [line [97](#)]

**Parámetros de la función:**

- **\$value**

- **Access** public

boolean función Catalogo\_model::updateComentario(\$nombre, \$comentario) [line [364](#)]

**Parámetros de la función:**

- *string* **\$nombre**
- *string* **\$comentario**

### Cambia el comentario de la tabla indicada

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

## Clase Catalogo\_x\_raiz\_model

[line [11](#)]

### Modelo Raiz\_x\_Catalogo

- **Package** SIIGS
- **Sub-Package** Modelo
- **Author** Geovanni

Constructor `void` función Catalogo\_x\_raiz\_model::\_\_construct() [[line 161](#)]

- **Access** public

`Object` función Catalogo\_x\_raiz\_model::check(\$id) [[line 290](#)]

**Parámetros de la función:**

- `int $id` ID (Llave primaria)

**Revisa inconsistencias en los datos de un catalogo x raiz con respecto a su catalogo padre**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

`boolean` función Catalogo\_x\_raiz\_model::delete() [[line 407](#)]

**Elimina el registro actual de la base de datos**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

`ArrayObject` función Catalogo\_x\_raiz\_model::getByArbol(\$id) [[line 178](#)]

**Parámetros de la función:**

- *int \$id*

## Devuelve todos los registros de la tabla raiz\_x\_catalogo de una raiz determinada

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*Object* función Catalogo\_x\_raiz\_model::getById(\$id) [[line 245](#)]

### Parámetros de la función:

- *int \$id* ID (Llave primaria)

## Devuelve la información de un catalogo x accion por su ID

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*ArrayObject* función Catalogo\_x\_raiz\_model::getByNivel(\$idarbol, \$nivel) [[line 223](#)]

### Parámetros de la función:

- *int \$idarbol*
- *int \$nivel*

## Devuelve el catalogo padre de un elemento raiz\_x\_catalogo

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*void función Catalogo\_x\_raiz\_model::getColumnaDescripcion() [line 113]*

- **Access** public

*void función Catalogo\_x\_raiz\_model::getColumnaLLave() [line 106]*

- **Access** public

*void función Catalogo\_x\_raiz\_model::getGrado() [line 92]*

- **Access** public

*void función Catalogo\_x\_raiz\_model::getId() [line 76]*

\*\*\*\*\*

- **Access** public

*void función Catalogo\_x\_raiz\_model::getIdRaiz() [line 84]*

- **Access** public

*string/boolean función Catalogo\_x\_raiz\_model::getMsgError([\$value = 'usr'], \$value,.) [line 146]*

**Parámetros de la función:**

- *string \$value, default 'usr' (Tipo mensaje)*
- *\$value*

**Devuelve los mensajes de error en caso de ocurrir alguna excepción 'usr' devuelve el mensaje para la vista de usuario 'log' devuelve el mensaje para el log de errores**

- **Access** public

*ArrayObject* función Catalogo\_x\_raiz\_model::getNivel(\$id) [line 200]

**Parámetros de la función:**

- *int \$id*

**Devuelve el nivel siguiente para la tabla raiz\_x\_catalogo de un arbol determinado**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*void* función Catalogo\_x\_raiz\_model::getRelacionHijo() [line 127]

- **Access** public

*void* función Catalogo\_x\_raiz\_model::getRelacionPadre() [line 120]

- **Access** public

*Object* función Catalogo\_x\_raiz\_model::getRelations(\$id) [line 267]

**Parámetros de la función:**

- *int \$id* ID (Llave primaria)

## Devuelve las relaciones de una raiz x catalogo

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*void* función Catalogo\_x\_raiz\_model::getTablaCatalogo() [[line 99](#)]

- **Access** public

*int* función Catalogo\_x\_raiz\_model::insert() [[line 350](#)]

## Inserta en la base datos la información del objeto actual

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*void* función Catalogo\_x\_raiz\_model::setColumnaDescripcion(\$value) [[line 116](#)]

### Parámetros de la función:

- **\$value**

- **Access** public

*void* función Catalogo\_x\_raiz\_model::setColumnaLlave(\$value) [[line 109](#)]

### Parámetros de la función:

- **\$value**

- **Access** public

void función Catalogo\_x\_raiz\_model::setGrado(\$value) [line 95]

**Parámetros de la función:**

- **\$value**

- **Access** public

void función Catalogo\_x\_raiz\_model::setId(\$value) [line 80]

**Parámetros de la función:**

- **\$value**

- **Access** public

void función Catalogo\_x\_raiz\_model::setIdRaiz(\$value) [line 98]

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Catalogo\_x\_raiz\_model::setRelacionHijo(\$value) [line 130]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Catalogo\_x\_raiz\_model::setRelacionPadre(\$value) [line 123]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Catalogo\_x\_raiz\_model::setTablaCatalogo(\$value) [line 102]*

**Parámetros de la función:**

- **\$value**

- **Access** public

# Clase Cie10\_model

[line 11]

## Modelo Cie10

- **Package** SIIGS
- **Sub-Package** Modelo
- **Author** Geovanni

Constructor `void` función `Cie10_model::__construct()` [line 106]

- **Access** public

`boolean` función `Cie10_model::activaEnCatalogo($id, $catalogo, $valor)` [line 293]

### Parámetros de la función:

- `int $id` el id del registro en el catalogo
- `string $catalogo` nombre del catalogo donde se realizara la operacion
- `boolean $valor` agregar o eliminar el registro del catalogo

## Accion para activar o desactivar registros de catalogos como el de EDA, IRA y Consultas

- **Throws** Exception Si ocurre algun error al consultar y modificar la base de datos
- **Access** public

`boolean` función `Cie10_model::agregaEnCatalogo($id, $catalogo, $valor)` [line 243]

### Parámetros de la función:

- `int $id` el id del registro en el catalogo cie10
- `string $catalogo` nombre del catalogo donde se realizara la operacion
- `boolean $valor` agregar o eliminar el registro del catalogo

## Accion para agregar registros del CIE10 a otros catalogos como el de EDA, IRA y Consultas

- **Throws** Exception Si ocurre algun error al consultar y modificar la base de datos
- **Access** public

*boolean función Cie10\_model::checkPk(\$campo) [line 843]*

**Parámetros de la función:**

- *string \$campo* (varios campos delimitados por | )

## Revisa en la base de datos por registros duplicados en los campos pasados por parametro

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*ArrayObject función Cie10\_model::getAll() [line 43]*

**Devuelve una lista con los registros existentes en el catalogo cie10**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*ArrayObject función Cie10\_model::getAllData(\$nombrecat) [line 320]*

**Parámetros de la función:**

- *string \$nombrecat* Nombre del catalogo

## Devuelve los datos de un catalogo pasado como parametro

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*Object* función Cie10\_model::getById(\$id) [line 220]

**Parámetros de la función:**

- *int \$id* ID (Llave primaria)

## Devuelve la información de un registro del catalogo cie10 por su ID

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*ArrayObject* función Cie10\_model::getCatalogoByName(\$cat) [line 194]

**Parámetros de la función:**

- *\$cat*

## Devuelve una lista con los registros existentes en el catalogo requerido

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*ArrayObject* función Cie10\_model::getData() [line 170]

**Devuelve una lista con los registros existentes en el catalogo cie10 omitiendo los ID**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*void función Cie10\_model::getDescripcion() [line 69]*

- **Access** public

*void función Cie10\_model::getId() [line 61]*

\*\*\*\*\*

- **Access** public

*string/boolean función Cie10\_model::getMsgError([\\$value = 'usr'], \\$value,) [line 91]*

**Parámetros de la función:**

- *string \$value*, default 'usr' (Tipo mensaje)
- *\$value*

**Devuelve los mensajes de error en caso de ocurrir alguna excepción 'usr' devuelve el mensaje para la vista de usuario 'log' devuelve el mensaje para el log de errores**

- **Access** public

*int función Cie10\_model::getNumRows() [line 122]*

**Devuelve el numero de registros**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*void función Cie10\_model::setDescripcion(\$value) [line 73]*

**Parámetros de la función:**

- **\$value**
  - **Access** public

*void función Cie10\_model::setId(\$value) [line 65]*

**Parámetros de la función:**

- **\$value**
  - **Access** public

*void función Cie10\_model::setOffset(\$value) [line 51]*

**Parámetros de la función:**

- **\$value**
  - **Access** public

*void* función Cie10\_model::setRows(\$value) [[line 54](#)]

**Parámetros de la función:**

- **\$value**

- **Access** public

*boolean* función Cie10\_model::update() [[line 874](#)]

**Actualiza el objeto actual en la base de datos**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

## Clase ControladorAccion\_model

[[line 11](#)]

**Modelo ControladorAccion**

- **Package** SIIGS
- **Sub-Package** Modelo
- **Author** Geovanni

Constructor *void* función ControladorAccion\_model::\_\_construct() [[line 79](#)]

- **Access** public

*Object* función ControladorAccion\_model::getById(\$id) [line [119](#)]

**Parámetros de la función:**

- *int \$id* ID (Llave primaria)

**Devuelve la información de una accion por controlador de acuerdo a su Id**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*int* función ControladorAccion\_model::getId(\$controlador, \$accion) [line [97](#)]

**Parámetros de la función:**

- *int \$controlador* (Id del controlador)
- *int \$accion* (Id de la accion)

**Devuelve el Id de una accion por controlador de una accion y controlador determinados**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*int* función ControladorAccion\_model::getIdByPath(\$path) [line [43](#)]

**Parámetros de la función:**

- *string \$path*

## **Devuelve el id de una accion por controlador de acuerdo al path**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*string|boolean* función ControladorAccion\_model::getMsgError([*\$value* = 'usr'], *\$value*,) [\[line 64\]](#)

### **Parámetros de la función:**

- *string \$value*, default 'usr' (Tipo mensaje)
- *\$value*

**Devuelve los mensajes de error en caso de ocurrir alguna excepción 'usr' devuelve el mensaje para la vista de usuario 'log' devuelve el mensaje para el log de errores**

- **Access** public

*int* función ControladorAccion\_model::setHelp(*\$id*, *\$textAyuda*) [\[line 181\]](#)

### **Parámetros de la función:**

- *int \$id*
- *string \$textAyuda*

**Establece el mensaje de ayuda**

- **Throws** Exception En caso de algun error al guardar el texto de ayuda
- **Access** public

# Clase Controlador\_model

[line 11]

## Modelo Controlador

- **Package** SIIGS
- **Sub-Package** Modelo
- **Author** Geovanni

Constructor void función Controlador\_model::\_\_construct() [line 159]

- **Access** public

boolean función Controlador\_model::accionesUpdate() [line 874]

## Actualiza las acciones asignadas a un controlador

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

boolean función Controlador\_model::delete() [line 431]

## Elimina el registro actual de la base de datos

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*void función Controlador\_model::getAcción() [line 108]*

- **Access** public

*ArrayObject función Controlador\_model::getAcciones(\$id) [line 271]*

**Parámetros de la función:**

- *int \$id* ID (Llave primaria)

**Devuelve todas las acciones asignadas al controlador por su Id**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*ArrayObject función Controlador\_model::getAll() [line 175]*

**Devuelve todos los registros de la tabla controlador**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*ArrayObject función Controlador\_model:: getByEntorno(\$id) [line 244]*

**Parámetros de la función:**

- *int \$id* ID (Id del entorno)

**Devuelve todos los controladores que pertenecen a un entorno por su Id**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*Object* función Controlador\_model::getById(\$id) [line[222](#)]

**Parámetros de la función:**

- *int \$id* ID (Llave primaria)

## Devuelve la información de un controlador por su ID

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*void* función Controlador\_model::getClase() [line[116](#)]

- **Access** public

*void* función Controlador\_model::getDescripcion() [line[92](#)]

- **Access** public

*void* función Controlador\_model::getId() [line[77](#)]

\*\*\*\*\*

- **Access** public

*void* función Controlador\_model::getIdEntorno() [line[100](#)]

- **Access** public

*string/boolean* función Controlador\_model::getMsgError([*\$value* = 'usr'], *\$value*,) [[line 144](#)]

**Parámetros de la función:**

- *string \$value*, default 'usr' (Tipo mensaje)
- *\$value*

**Devuelve los mensajes de error en caso de ocurrir alguna excepción 'usr' devuelve el mensaje para la vista de usuario 'log' devuelve el mensaje para el log de errores**

- **Access** public

*void* función Controlador\_model::getNombre() [[line 85](#)]

- **Access** public

*int* función Controlador\_model::getNumRows([*\$entorno* = 0]) [[line 201](#)]

**Parámetros de la función:**

- *int \$entorno* , default 0 (Id del entorno)

**Devuelve el numero de registros**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*ArrayObject* función Controlador\_model::getPermisos(\$entorno, \$grupo) [line[296](#)]

**Parámetros de la función:**

- **int \$entorno** (Id del entorno)
- **int \$grupo** (Id del grupo)

**Devuelve los permisos asignados a un grupo sobre un entorno determinado  
(Mapea la información de las acciones asignadas a un controlador y los une con los permisos de un grupo sobre esas acciones)**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*int* función Controlador\_model::insert() [line[817](#)]

**Inserta en la tabla controlador, la información contenida en el objeto**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*void* función Controlador\_model::setAccion(\$value) [line[112](#)]

**Parámetros de la función:**

- **\$value**
  - **Access** public

*void* función Controlador\_model::setClase(\$value) [line[120](#)]

**Parámetros de la función:**

- **\$value**
  - **Access** public

*void función Controlador\_model::setDescripcion(\$value) [line 96]*

**Parámetros de la función:**

- **\$value**
  - **Access** public

*void función Controlador\_model::setId(\$value) [line 81]*

**Parámetros de la función:**

- **\$value**
  - **Access** public

*void función Controlador\_model::setIdEntorno(\$value) [line 104]*

**Parámetros de la función:**

- **\$value**

- **Access** public

void función Controlador\_model::setNombre(\$value) [line 88]

**Parámetros de la función:**

- **\$value**

- **Access** public

void función Controlador\_model::setOffset(\$value) [line 124]

**Parámetros de la función:**

- **\$value**

- **Access** public

void función Controlador\_model::setRows(\$value) [line 127]

**Parámetros de la función:**

- **\$value**

- **Access** public

boolean función Controlador\_model::update() [line 845]

**Actualiza el objeto actual en la base de datos**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

## Clase Entorno\_model [line 11]

### Modelo Entorno

- **Package** SIIGS
- **Sub-Package** Modelo
- **Author** Geovanni

Constructor *void* función Entorno\_model::\_\_construct() [line 148]

- **Access** public

*boolean* función Entorno\_model::delete() [line 811]

### Elimina el registro actual de la base de datos

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*ArrayObject* función Entorno\_model::getAll() [line 164]

### Devuelve todos los registros de la tabla entorno

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*Object función Entorno\_model::getById(\$id) [line 186]*

**Parámetros de la función:**

- *int \$id* ID (Llave primaria)

**Devuelve la información de un entorno por su ID**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*Object función Entorno\_model::getByName(\$nombre) [line 208]*

**Parámetros de la función:**

- *string \$nombre*

**Devuelve la información de un entorno por su nombre**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*void función Entorno\_model::getDescripcion() [line 78]*

- **Access** public

*void función Entorno\_model::getDirectorio() [line 99]*

- **Access** public

*void función Entorno\_model::getHostname() [line 92]*

- **Access** public

*void función Entorno\_model::getId() [line 64]*

\*\*\*\*\*

- **Access** public

*string función Entorno\_model:: getInfo() [line 142]*

**Devuelve la información del objeto en forma de string**

- **Access** public

*void función Entorno\_model::getIp() [line 85]*

- **Access** public

*string/boolean función Entorno\_model::getMsgError([\$value = 'usr'], \$value,) [line 118]*

**Parámetros de la función:**

- *string \$value*, default 'usr' (Tipo mensaje)
- *\$value*

**Devuelve los mensajes de error en caso de ocurrir alguna excepción 'usr' devuelve el mensaje para la vista de usuario 'log' devuelve el mensaje para el log de errores**

- **Access** public

*void* función Entorno\_model::getNombre() [[line 231](#)]

- **Access** public

*Object* función Entorno\_model::getPermissionsByGroup(\$grupo) [[line 230](#)]

**Parámetros de la función:**

- *string \$grupo*

**Obtiene los permisos asignados al grupo**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*int* función Entorno\_model::insert() [[line 252](#)]

**Inserta en la tabla entorno la información contenida en el objeto**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

void función Entorno\_model::setDescripcion(\$value) [line[81](#)]

**Parámetros de la función:**

- **\$value**
- **Access** public

void función Entorno\_model::setDirectorio(\$value) [line[102](#)]

**Parámetros de la función:**

- **\$value**
- **Access** public

void función Entorno\_model::setHostname(\$value) [line[95](#)]

**Parámetros de la función:**

- **\$value**
- **Access** public

void función Entorno\_model::setId(\$value) [line[67](#)]

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Entorno\_model::setIp(\$value) [line 88]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Entorno\_model::setNombre(\$value) [line 74]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*boolean función Entorno\_model::update() [line 281]*

**Actualiza el objeto actual en la base de datos**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

# Clase Errorlog\_model

[line 11]

## Modelo Errorlog

- **Package** SIIGS
- **Sub-Package** Modelo
- **Author** Pascual

Constructor `void` función `Errorlog_model::__construct()` [line 76]

- **Access** public

`void|boolean` función `Errorlog_model::addFilter($columna, $condicion, $valor)` [line 243]

**Parámetros de la función:**

- `string $columna` Puede ser cualquier campo del objeto (id, id\_usuario, fecha\_hora, descripcion, id\_controlador\_accion)
- `string $condicion` Establece la condicion a evaluar, entre los valores permitidos estan: =, !=, >=, <=, like
- `string $valor` Valor contra el cual se realizará la evaluación del campo

## Agrega una nueva regla de filtrado al arreglo de filtros

- **Access** public

`array` función `Errorlog_model::getAll([$offset = null], [$row_count = null])` [line 801]

**Parámetros de la función:**

- *int \$offset* Establece el desplazamiento del primer registro a devolver, si se define solo el valor de offset el valor especifica el número de registros a retornar desde el comienzo del conjunto de resultados.
- *int \$row\_count* Establece la cantidad de registros a devolver

**Obtiene todos los registros de la tabla Error en caso de existir filtros, estos son aplicados a la consulta**

- **Access** public

*object|boolean* función Errorlog\_model::getById(\$id) [[line 196](#)]

**Parámetros de la función:**

- *int \$id* Si no se establece el valor de ID, se toma el valor del objeto actual

**Obtiene los datos del registro del Error que tiene el ID especificado**

- **Access** public

*void* función Errorlog\_model::getDescripcion() [[line 129](#)]

- **Access** public

*void* función Errorlog\_model::getFecha\_hora() [[line 124](#)]

- **Access** public

*void función Errorlog\_model::getId() [line 94]*

- **Access** public

*void función Errorlog\_model::getId\_controlador\_accion() [line 14]*

- **Access** public

*void función Errorlog\_model::getId\_usuario() [line 104]*

- **Access** public

*int|boolean función Errorlog\_model::getNumRows() [line 844]*

**Obtiene el numero total de registros en la tabla Error en caso de existir filtros, estos son aplicados a la consulta**

- **Access** public

*void función Errorlog\_model::insert(\$path, \$descripcion, \$id\_controlador, \$id\_accion) [line 158]*

**Parámetros de la función:**

- *int \$id\_controlador*
- *int \$id\_accion*
- *string \$descripcion*
- *\$path*

**Inserta en la base de datos la informacion del error**

- **Static**
- **Access** public

*void función Errorlog\_model::resetFilter() [line 285]*

**Elimina todos los filtros registrados**

- **Access** public

*string función Errorlog\_model::save(\$model, \$method, \$modelo) [line 383]*

**Parámetros de la función:**

- *string \$modelo* Nombre del modelo que lanzó la excepción
- *string \$method* Contiene el nombre de la clase y metodo donde se originó el error o el mensaje de error para mostrar al usuario final
- *\$model*

**Guardar el mensaje de error descriptivo en la base de datos,**

si no puede insertar el registro a la base de datos, el mensaje de error se guarda en el directorio logs, devuelve el mensaje de error para el usuario final

- **Static**
- **Access** public

*void función Errorlog\_model::setDescripcion(\$descripcion) [line 134]*

**Parámetros de la función:**

- *\$descripcion*

- **Access** public

*void función Errorlog\_model::setId(\$id) [line 99]*

**Parámetros de la función:**

- **\$id**

- **Access** public

*void función Errorlog\_model::setId\_accion(\$id\_accion) [line 144]*

**Parámetros de la función:**

- **\$id\_accion**

- **Access** public

*void función Errorlog\_model::setId\_controlador(\$id\_controlador) [line 139]*

**Parámetros de la función:**

- **\$id\_controlador**

- **Access** public

*void función Errorlog\_model::setId\_controlador\_accion(\$id\_controlador\_accion) [line 119]*

**Parámetros de la función:**

- **\$id\_controlador\_accion**

- **Access** public

void función Errorlog\_model::setId\_usuario(\$id\_usuario) [\[line 109\]](#)

**Parámetros de la función:**

- **\$id\_usuario**

- **Access** public

## Clase Georeferencia\_model [\[line 11\]](#)

**Modelo Georeferencia**

- **Package** SIIGS
- **Sub-Package** Modelo
- **Author** Pascual

Constructor void función Georeferencia\_model::\_\_construct() [\[line 26\]](#)

- **Access** public

*boolean|string* función Georeferencia\_model::getMsgError([*\$type* = 'usr']) [line 60]

**Parámetros de la función:**

- *string \$type* usr si se quiere devolver el mensaje de error a mostrar en la vista, log obtiene el mensaje de error con mas detalles para depuración, valor por defecto usr

**Devuelve el mensaje de error, en caso de existir un error despues de ejecutar un metodo, de lo contrario false**

- **Access** public

*boolean* función Georeferencia\_model::process() [line 82]

**Inserta los registros contenidos en la tabla cat\_georeferencia a la tabla asu\_georeferencia**

- **Access** public

## Clase Grupo\_model

[line 10]

**Modelo Grupo**

- **Package** SIIGS
- **Sub-Package** Modelo
- **Author** Rogelio

Constructor `void` función `Grupo_model::__construct()` [line 42]

- **Access** public

`boolean` función `Grupo_model::delete()` [line 265]

### **Elimina de la base de datos al grupo (id en propiedades)**

- **Access** public

`void|array` función `Grupo_model::getAll([\$keywords = "], [\${$offset} = null], [\${$row_count} = null])` [line 102]

#### **Parámetros de la función:**

- `int $offset` Establece el desplazamiento del primer registro a devolver, si se define solo el valor de offset el valor especifica el número de registros a retornar desde el comienzo del conjunto de resultados.
- `int $row_count` Establece la cantidad de registros a devolver
- `\$keywords`

### **Obtiene todos los grupos existentes**

- **Access** public

`void|object` función `Grupo_model::getById($id)` [line 135]

#### **Parámetros de la función:**

- `int $id` id del grupo

## Obtiene el grupo solicitado

- **Access** public

*void|object* función Grupo\_model::getByName(\$name) [\[line 177\]](#)

**Parámetros de la función:**

- *string* **\$name** nombre del grupo

## Obtiene el grupo solicitado

- **Access** public

*void* función Grupo\_model::getDescripcion() [\[line 68\]](#)

- **Access** public

*void|object* función Grupo\_model::getEntornosById(\$id) [\[line 155\]](#)

**Parámetros de la función:**

- *int* **\$id** id del grupo

## Obtiene el grupo solicitado con sus entornos vinculados

- **Access** public

*void* función Grupo\_model::getId() [*line 49*]

- **Access** public

*boolean* función Grupo\_model::getMsgError([*\$value* = 'usr']) [*line 85*]

**Parámetros de la función:**

- *string* **\$value** tipo de error a visualizar: usr o log, default: usr

**Asigna el mensaje de error a visualizar: para usuario final (usr) o para bitácora (log)**

- **Access** public

*void* función Grupo\_model::getNombre() [*line 58*]

- **Access** public

*int* función Grupo\_model::getNumRows([*\$keywords* = '']) [*line 197*]

**Parámetros de la función:**

- *boolean|string* **\$keywords** false no hay texto a buscar|string con texto a buscar

**Obtiene el numero total de grupos**

- **Access** public

*boolean* función Grupo\_model::insert() [[line 223](#)]

**Inserta en la base de datos los datos del grupo (datos en propiedades)**

- **Access** public

*void* función Grupo\_model::setDescripcion(\$descripcion) [[line 73](#)]

**Parámetros de la función:**

- **\$descripcion**

- **Access** public

*void* función Grupo\_model::setId(\$value) [[line 54](#)]

**Parámetros de la función:**

- **\$value**

- **Access** public

*void* función Grupo\_model::setNombre(\$nombre) [[line 63](#)]

**Parámetros de la función:**

- **\$nombre**

- **Access** public

*boolean* función Grupo\_model::update() [[line 244](#)]

**Actualiza en la base de datos los datos del grupo (datos en propiedades)**

- **Access** public

## Clase Hemoglobina\_model [[line 11](#)]

**Modelo Hemoglobina**

- **Package** SIIGS
- **Sub-Package** Modelo
- **Author** Geovanni

Constructor *void* función Hemoglobina\_model::\_\_construct() [[line 36](#)]

- **Access** public

*boolean/string* función Hemoglobina\_model::getMsgError([*\$type* = 'usr']) [[line 60](#)]

**Parámetros de la función:**

- *string \$type* usr si se quiere devolver el mensaje de error a mostrar en la vista, log obtiene el mensaje de error con mas detalles para depuración, valor por defecto usr

**Devuelve el mensaje de error, en caso de existir un error despues de ejecutar un metodo, de lo contrario false**

- **Access** public

*boolean* función Hemoglobina\_model::process() [[line 82](#)]

**Inserta los registros contenidos en la tabla cat\_georeferencia a la tabla  
asu\_georeferencia**

- **Access** public

## Clase Menu\_model

[[line 11](#)]

**Modelo Menu**

- **Package** SIIGS
- **Sub-Package** Modelo
- **Author** Pascual

Constructor *void* función Menu\_model::\_\_construct() [[line 85](#)]

- **Access** public

*void|boolean* función Menu\_model::addFilter(\$columna, \$condicion, \$valor) [[line 401](#)]

**Parámetros de la función:**

- *string \$columna* Puede ser cualquier campo del objeto (id, id\_usuario, fecha\_hora, parametros, id\_controlador\_accion)
- *string \$condicion* Establece la condicion a evaluar, entre los valores permitidos estan: =, !=, >=, <=, like
- *string \$valor* Valor contra el cual se realizará la evaluación del campo

### Agrega una nueva regla de filtrado al arreglo de filtros

- **Access** public

*int* función Menu\_model::delete([*\$id* = null]) [[line 287](#)]

**Parámetros de la función:**

- *int \$id* Si no se establece el valor de ID, se toma el valor del objeto actual

### Elimina el registro actual de la base de datos

- **Access** public

*int* función Menu\_model::deleteByFilter() [[line 326](#)]

### Elimina el conjunto de registros que cumplen con el o los criterios de filtrado

- **Access** public

*array* función Menu\_model::getAll([*\$offset* = null], [*\$row\_count* = null]) [[line 469](#)]

**Parámetros de la función:**

- *int \$offset* Establece el desplazamiento del primer registro a devolver, si se define solo el valor de offset el valor especifica el número de registros a retornar desde el comienzo del conjunto de resultados.
- *int \$row\_count* Establece la cantidad de registros a devolver

**Obtiene todos los registros de la tabla Menu en caso de existir filtros, estos son aplicados a la consulta**

- **Access** public

*void* función Menu\_model::getAtributo() [[line 163](#)]

- **Access** public

*object|boolean* función Menu\_model::getById(\$id) [[line 358](#)]

**Parámetros de la función:**

- *int \$id* Si no se establece el valor de ID, se toma el valor del objeto actual

**Obtiene los datos del registro de la menu que tiene el ID especificado**

- **Access** public

*boolean* función Menu\_model::getByPadre(\$padre) [[line 566](#)]

**Parámetros de la función:**

- **\$padre**

## Obtiene todos los nodos hijos de un padre

- **Access** public

*void función Menu\_model::getId() [line 98]*

- **Access** public

*void función Menu\_model::getId\_controlador() [line 143]*

- **Access** public

*void función Menu\_model::getId\_padre() [line 108]*

- **Access** public

*void función Menu\_model::getId\_raiz() [line 118]*

- **Access** public

*boolean/string función Menu\_model::getMsgError([\$type = 'usr']) [line 179]*

**Parámetros de la función:**

- *string \$type* usr si se quiere devolver el mensaje de error a mostrar en la vista, log obtiene el mensaje de error con mas detalles para depuración, valor por defecto usr

**Devuelve el mensaje de error, en caso de existir un error despues de ejecutar un metodo, de lo contrario false**

- **Access** public

*void función Menu\_model::getNombre() [line 128]*

- **Access** public

*int función Menu\_model::getNumRows() [line 511]*

**Obtiene el numero total de registros en la tabla Menu en caso de existir filtros, estos son aplicados a la consulta**

- **Access** public

*void función Menu\_model::getRuta() [line 153]*

- **Access** public

*boolean función Menu\_model::hasChild(\$id) [line 550]*

**Parámetros de la función:**

- **\$id**

**Verifica si el nodo actual tiene hijos**

- **Access** public

*boolean función Menu\_model::insert() [line 200]*

**Inserta en la base de datos, la informacion contenida en el objeto**

- **Access** public

*void función Menu\_model::resetFilter() [line 453]*

**Elimina todos los filtros registrados**

- **Access** public

*void función Menu\_model::setAtributo(\$atributo) [line 58]*

**Parámetros de la función:**

- **\$atributo**

- **Access** public

*void función Menu\_model::setId(\$id) [line 103]*

**Parámetros de la función:**

- **\$id**

- **Access** public

void función Menu\_model::setId\_controlador(\$id\_controlador) [line[138](#)]

**Parámetros de la función:**

- **\$id\_controlador**

- **Access** public

void función Menu\_model::setId\_padre(\$id\_padre) [line[113](#)]

**Parámetros de la función:**

- **\$id\_padre**

- **Access** public

void función Menu\_model::setId\_raiz(\$id\_raiz) [line[123](#)]

**Parámetros de la función:**

- **\$id\_raiz**

- **Access** public

void función Menu\_model::setNombre(\$nombre) [line[133](#)]

**Parámetros de la función:**

- **\$nombre**

- **Access** public

*void función Menu\_model::setRuta(\$ruta) [line 148]*

**Parámetros de la función:**

- **\$ruta**

- **Access** public

*boolean función Menu\_model::update([id = null]) [line 244]*

**Parámetros de la función:**

- **int \$id** Si no se establece el valor de ID, se toma el valor del objeto actual

### **Actualiza los datos del objeto actual**

- **Access** public

## **Clase Permiso\_model**

[line 10]

## Modelo Permiso

- **Package** SIIGS
- **Sub-Package** Modelo
- **Author** Rogelio

Constructor `void` función `Permiso_model::__construct()` [line 47]

- **Access** public

`boolean` función `Permiso_model::deletePermissions($entorno, $grupo)` [line 156]

**Parámetros de la función:**

- `int $entorno` id de entorno
- `int $grupo` id del grupo

### Elimina de la base de datos los permisos del entorno y grupo recibidos

- **Access** public

`void` función `Permiso_model::getFecha()` [line 73]

- **Access** public

`void` función `Permiso_model::getId()` [line 54]

- **Access** public

*void función Permiso\_model::getIdControladorAccion() [line 83]*

- **Access** public

*void función Permiso\_model::getIdGrupo() [line 63]*

- **Access** public

*boolean función Permiso\_model::getMsgError([\\$value = 'usr']) [line 100]*

**Parámetros de la función:**

- **\$value** **\$value** tipo de error a visualizar: usr o log, default: usr

**Asigna el mensaje de error a visualizar: para usuario final (usr) o para bitácora (log)**

- **Access** public

*void/object función Permiso\_model::getPermission(\$id) [line 14]*

**Parámetros de la función:**

- **int \$id** id del controlador\_x\_accion

**Obtiene el permiso solicitado**

- **Access** public

*boolean función Permiso\_model::insertBatch(\$data) [line 135]*

**Parámetros de la función:**

- **array \$data** object \$data array con los permisos a insertar

### **Inserta en la base de datos el arreglo de permisos recibido**

- **Access** public

*void función Permiso\_model::setFecha(\$fecha) [line 78]*

**Parámetros de la función:**

- **\$fecha**

- **Access** public

*void función Permiso\_model::setId(\$value) [line 59]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Permiso\_model::setIdControladorAccion(\$id\_controlador\_accion) [line 88]*

**Parámetros de la función:**

- **\$id\_controlador\_accion**

- **Access** public

void función Permiso\_model::setIdGrupo(\$id\_grupo) [[line 68](#)]

**Parámetros de la función:**

- **\$id\_grupo**

- **Access** public

## Clase Poblacion\_model [[line 11](#)]

**Modelo Poblacion**

- **Package** SIIGS
- **Sub-Package** Modelo
- **Author** Pascual

Constructor void función Poblacion\_model::\_\_construct() [[line 86](#)]

- **Access** public

*boolean|string* función Poblacion\_model::getMsgError([*\$type* = 'usr']) [*line 60*]

**Parámetros de la función:**

- *string \$type* usr si se quiere devolver el mensaje de error a mostrar en la vista, log obtiene el mensaje de error con mas detalles para depuración, valor por defecto usr

**Devuelve el mensaje de error, en caso de existir un error despues de ejecutar un metodo, de lo contrario false**

- **Access** public

*boolean* función Poblacion\_model::process() [*line 82*]

**Inserta los registros contenidos en la tabla cat\_poblacion a la tablaasu\_poblacion**

- **Access** public

## Clase Raiz\_model [line 11]

**Modelo Raiz**

- **Package** SIIGS
- **Sub-Package** Modelo
- **Author** Geovanni

Constructor `void` función Raiz\_model::\_\_construct() [line 84]

- **Access** public

`boolean` función Raiz\_model::delete() [line 216]

**Elimina el registro actual de la base de datos**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

`Boolean` función Raiz\_model::ExistInArbol(\$id) [line 122]

**Parámetros de la función:**

- `int $id`

**Revisa si la raiz pasada como parametro existe en el ASU**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

`ArrayObject` función Raiz\_model::getAll() [line 100]

**Devuelve todos los registros de la tabla raiz**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*Object* función Raiz\_model::getById(\$id) [line 144]

**Parámetros de la función:**

- *int \$id* ID (Llave primaria)

**Devuelve la información de una raíz por su ID**

- **Throws** Exception En caso de algún error al consultar la base de datos
- **Access** public

*void* función Raiz\_model::getDescripcion() [line 48]

- **Access** public

*void* función Raiz\_model::getId() [line 40]

\*\*\*\*\*

- **Access** public

*string/boolean* función Raiz\_model::getMsgError([\$value = 'usr'], \$value,) [line 69]

**Parámetros de la función:**

- *string \$value*, default 'usr' (Tipo mensaje)
- *\$value*

**Devuelve los mensajes de error en caso de ocurrir alguna excepción 'usr' devuelve el mensaje para la vista de usuario 'log' devuelve el mensaje para el log de errores**

- **Access** public

*int función Raiz\_model::insert() [line 165]*

**Inserta en la tabla raiz, la información contenida en el objeto**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*void función Raiz\_model::setDescripcion(\$value) [line 52]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Raiz\_model::setId(\$value) [line 44]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*boolean función Raiz\_model::update() [line 190]*

**Actualiza el objeto actual en la base de datos**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

## Clase ReglaVacuna\_model

*[line 11]*

### Modelo ReglaVacuna

- **Package** SIIGS
- **Sub-Package** Modelo
- **Author** Geovanni

Constructor *void* función ReglaVacuna\_model::\_\_construct() *[line 251]*

- **Access** public

*boolean* función ReglaVacuna\_model::delete() *[line 414]*

### Elimina el registro actual de la base de datos

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*void* función ReglaVacuna\_model::getAlergias() *[line 209]*

- **Access** public

*ArrayObject* función ReglaVacuna\_model::getAll() [[line 267](#)]

### **Devuelve todos los registros de la tabla regla\_vacuna**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*Object* función ReglaVacuna\_model::getById(\$id) [[line 288](#)]

#### **Parámetros de la función:**

- *int \$id* ID (Llave primaria)

### **Devuelve la información de una regla de vacuna por su ID**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*void* función ReglaVacuna\_model::getDiaFinNacido() [[line 146](#)]

- **Access** public

*void* función ReglaVacuna\_model::getDiaFinPrevia() [[line 160](#)]

- **Access** public

*void función ReglaVacuna\_model::getDiaInicioNacido() [line 139]*

- **Access** public

*void función ReglaVacuna\_model::getDiaInicioPrevia() [line 153]*

- **Access** public

*void función ReglaVacuna\_model::getDosis() [line 174]*

- **Access** public

*void función ReglaVacuna\_model::getEsqComp() [line 195]*

- **Access** public

*void función ReglaVacuna\_model::getForzarAplicacion() [line 216]*

- **Access** public

*void función ReglaVacuna\_model::getId() [line 118]*

\*\*\*\*\*

- **Access** public

*void función ReglaVacuna\_model::getIdVacuna() [line 125]*

- **Access** public

*void función ReglaVacuna\_model::getIdVacunaPrevia() [line[132](#)]*

- **Access** public

*void función ReglaVacuna\_model::getIdViaVacuna() [line[167](#)]*

- **Access** public

*string/boolean función ReglaVacuna\_model::getMsgError([\$value = 'usr'], \$value,) [line[236](#)]*

**Parámetros de la función:**

- **string \$value**, default 'usr' (Tipo mensaje)
- **\$value**

**Devuelve los mensajes de error en caso de ocurrir alguna excepción 'usr' devuelve el mensaje para la vista de usuario 'log' devuelve el mensaje para el log de errores**

- **Access** public

*void función ReglaVacuna\_model::getObservacionRegion() [line[188](#)]*

- **Access** public

*void función ReglaVacuna\_model::getOrdenEsqComp() [line 202]*

- **Access** public

*void función ReglaVacuna\_model::getRegion() [line 181]*

- **Access** public

*int función ReglaVacuna\_model::insert() [line 327]*

**Inserta en la tabla regla\_vacuna la información contenida en el objeto**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

*void función ReglaVacuna\_model::setAlergias(\$value) [line 212]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función ReglaVacuna\_model::setDiaFinNacido(\$value) [line 149]*

**Parámetros de la función:**

- **\$value**

- **Access** public

void función ReglaVacuna\_model::setDiaFinPrevia(\$value) [line 163]

**Parámetros de la función:**

- **\$value**

- **Access** public

void función ReglaVacuna\_model::setDialInicioNacido(\$value) [line 142]

**Parámetros de la función:**

- **\$value**

- **Access** public

void función ReglaVacuna\_model::setDialInicioPrevia(\$value) [line 156]

**Parámetros de la función:**

- **\$value**

- **Access** public

void función ReglaVacuna\_model::setDosis(\$value) [line 177]

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función ReglaVacuna\_model::setEsqComp(\$value) [line 198]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función ReglaVacuna\_model::setForzarAplicacion(\$value) [line 219]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función ReglaVacuna\_model::setId(\$value) [line 121]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función ReglaVacuna\_model::setIdVacuna(\$value) [line 128]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función ReglaVacuna\_model::setIdVacunaPrevia(\$value) [line 135]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función ReglaVacuna\_model::setIdViaVacuna(\$value) [line 170]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función ReglaVacuna\_model::setObservacionRegion(\$value) [line 191]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función ReglaVacuna\_model::setOrdenEsqComp(\$value) [line 205]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función ReglaVacuna\_model::setRegion(\$value) [line 184]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*boolean función ReglaVacuna\_model::update() [line 370]*

**Actualiza el objeto actual en la base de datos**

- **Throws** Exception En caso de algun error al consultar la base de datos
- **Access** public

# Clase Usuario\_model

[line 10]

## Modelo Usuario

- **Package** SIIGS
- **Sub-Package** Modelo
- **Author** Rogelio

Constructor `void` función `Usuario_model::__construct()` [line 81]

- **Access** public

`null||object` función `Usuario_model::authenticate($username, $password)` [line 423]

**Parámetros de la función:**

- `string $username` nombre de usuario
- `string $password` clave

## Valida las credenciales recibidas

- **Access** public

`void` función `Usuario_model::checkCredentials($path, $pathURL, $group_id)` [line 444]

**Parámetros de la función:**

- `string $path` entorno::controlador::accion
- `int $group_id` id del grupo a validar permisos
- `$pathURL`

**Verifica que el usuario haya iniciado sesión y además tenga permiso en la acción recibida**

- **Static**
- **Access public**

*void función Usuario\_model::check\_data(\$usuario, \$correo) [line 477]*

**Parámetros de la función:**

- **\$usuario**
  - **\$correo**
- 
- **Access public**

*void función Usuario\_model::check\_token(\$correo) [line 490]*

**Parámetros de la función:**

- **\$correo**
- 
- **Access public**

*boolean función Usuario\_model::delete() [line 404]*

**Elimina de la base de datos al usuario (id en propiedades)**

- **Access** public

*void|object* función Usuario\_model::getActivesByGroup(\$group\_id, \$id, \$viewMode) [line[283](#)]

**Parámetros de la función:**

- *int \$id* id del usuario
- *boolean \$viewMode* true obtiene el modo visualización, false o null obtiene el registro normal
- *\$group\_id*

## Obtiene los usuarios activos del grupo solicitado

- **Access** public

*void* función Usuario\_model::getActivo() [line[161](#)]

- **Access** public

*void* función Usuario\_model::getApellidoMaterno() [line[141](#)]

- **Access** public

*void* función Usuario\_model::getApellidoPaterno() [line[131](#)]

- **Access** public

*void|object* función Usuario\_model::getById(\$id, [\$viewMode = FALSE]) [line[253](#)]

**Parámetros de la función:**

- *int \$id* id del usuario
- *boolean \$viewMode* true obtiene el modo visualización, false o null obtiene el registro normal

**Obtiene el usuario solicitado, se puede obtener el registro normal o personalizado para visualización (descripciones en tablas vinculadas)**

- **Access** public

*void|object función Usuario\_model::getByUsername(\$username) [line 303]*

**Parámetros de la función:**

- *string \$username* nombre de usuario

**Obtiene el usuario solicitado**

- **Access** public

*void función Usuario\_model::getClave() [line 11]*

- **Access** public

*void función Usuario\_model::getCorreo() [line 151]*

- **Access** public

*void función Usuario\_model::getgrupo(\$grupo) [line 532]*

**Parámetros de la función:**

- **\$grupo**

- **Access** public

*void función Usuario\_model::getId() [line 92]*

- **Access** public

*void función Usuario\_model::getIdGrupo() [line 171]*

- **Access** public

*boolean función Usuario\_model::getMsgError([\$value = 'usr']) [line 188]*

**Parámetros de la función:**

- *string \$value* tipo de error a visualizar: usr o log, default: usr

**Asigna el mensaje de error a visualizar: para usuario final (usr) o para bitácora (log)**

- **Access** public

*void función Usuario\_model::getNombre() [line 121]*

- **Access** public

*void* función Usuario\_model::getNombreUsuario() [line [101](#)]

- **Access** public

*int* función Usuario\_model::getNumRows([*\$keywords* = ""]) [line [323](#)]

**Parámetros de la función:**

- *boolean|string* **\$keywords** false no hay texto a buscar|string con texto a buscar

### Obtiene el numero total de usuarios

- **Access** public

*void|array* función Usuario\_model::getOnlyActives([*\$keywords* = ""], [*\$onlyActives* = TRUE], [*\$offset* = null], [*\$row\_count* = null]) [line [207](#)]

**Parámetros de la función:**

- *boolean|string* **\$keywords** false no hay texto a buscar|string con texto a buscar
- *boolean* **\$onlyActives** true obtiene solo usuarios activos, false o null obtiene todos los usuarios
- *int* **\$offset** Establece el desplazamiento del primer registro a devolver, si se define solo el valor de offset el valor especifica el número de registros a retornar desde el comienzo del conjunto de resultados.
- *int* **\$row\_count** Establece la cantidad de registros a devolver

### Obtiene todos los usuarios existentes, se puede filtrar por: texto a buscar o solo activos si se desea

- **Access** public

*void función Usuario\_model::getuser(\$id) [line [519](#)]*

**Parámetros de la función:**

- **\$id**

- **Access** public

*void función Usuario\_model::get\_grupo\_entorno(\$nombre) [line [583](#)]*

**Parámetros de la función:**

- **\$nombre**

- **Access** public

*void función Usuario\_model::get\_permiso\_entorno(\$nombre) [line [560](#)]*

**Parámetros de la función:**

- **\$nombre**

- **Access** public

*void función Usuario\_model::get\_usuario\_entorno(\$nombre, [\$inusuario = ""]) [line [607](#)]*

**Parámetros de la función:**

- **\$nombre**
- **\$inusuario**

- **Access** public

*boolean función Usuario\_model::insert() [line[351](#)]*

**Inserta en la base de datos los datos del usuario (datos en propiedades)**

- **Access** public

*void función Usuario\_model::setActivo(\$activo) [line[166](#)]*

**Parámetros de la función:**

- **\$activo**

- **Access** public

*void función Usuario\_model::setApellidoMaterno(\$apellido\_materno) [line[146](#)]*

**Parámetros de la función:**

- **\$apellido\_materno**

- **Access** public

*void función Usuario\_model::setApellidoPaterno(\$apellido\_paterno) [line[136](#)]*

**Parámetros de la función:**

- **\$apellido\_paterno**

- **Access** public

*void función Usuario\_model::setClave(\$clave) [line 16]*

**Parámetros de la función:**

- **\$clave**

- **Access** public

*void función Usuario\_model::setCorreo(\$correo) [line 156]*

**Parámetros de la función:**

- **\$correo**

- **Access** public

*void función Usuario\_model::setId(\$value) [line 97]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Usuario\_model::setIdGrupo(\$id\_grupo) [line 176]*

**Parámetros de la función:**

- **\$id\_grupo**

- **Access** public

*void función Usuario\_model::setNombre(\$nombre) [line 126]*

**Parámetros de la función:**

- **\$nombre**

- **Access** public

*void función Usuario\_model::setNombreUsuario(\$nombre\_usuario) [line 106]*

**Parámetros de la función:**

- **\$nombre\_usuario**

- **Access** public

*boolean función Usuario\_model::update() [line 878]*

**Actualiza en la base de datos los datos del usuario (datos en propiedades)**

- **Access** public

*void función Usuario\_model::update\_pass(\$pass, \$id) [line 503]*

**Parámetros de la función:**

- **\$pass**
- **\$id**

- **Access** public

*void función Usuario\_model::update\_user(\$id, \$clave, \$correo) [line 545]*

**Parámetros de la función:**

- **\$id**
- **\$clave**
- **\$correo**

- **Access** public



# Paquete TES Elementos procedurales

## enrolamiento.php

- **Package** TES
- **Sub-Package** Controlador
- **Filesource** [Source Code for this file](#)

## notificacion.php

- **Package** TES
- **Sub-Package** Controlador
- **Filesource** [Source Code for this file](#)

## reporteador.php

- **Package** TES
- **Sub-Package** Controlador
- **Filesource** [Source Code for this file](#)

## reporte\_sincronizacion.php

- **Package** TES
- **Sub-Package** Controlador
- **Filesource** [Source Code for this file](#)

## semana\_nacional.php

- **Package** TES
- **Sub-Package** Controlador
- **Filesource** [Source Code for this file](#)

# servicios.php

- **Package** TES
- **Sub-Package** Controlador
- **Filesource** [Source Code for this file](#)

## tableta.php

- **Package** TES
- **Sub-Package** Controlador
- **Filesource** [Source Code for this file](#)

## usuario\_tableta.php

- **Package** TES
- **Sub-Package** Controlador
- **Filesource** [Source Code for this file](#)

# Paquete TES Clases

## Clase Enrolamiento

[line [10](#)]

### Controlador de enrolamiento

- **Package** TES
- **Sub-Package** Controlador
- **Author** Eliecer

Constructor **void** función Enrolamiento::\_\_construct() [line [13](#)]

- **Access** public

echo función Enrolamiento::addForm() [line [907](#)]

### Pase de parametros para la insercion o actualizacion

- **Access** public

echo función Enrolamiento::autocomplete() [line [470](#)]

### Crea el autocomplete para facilitar la busqueda de un tutor

- **Access** public

echo función Enrolamiento::brothers\_search(\$tutor) [[line 1088](#)]

**Parámetros de la función:**

- *string \$tutor* id del tutor compartido

**Este metodo extrae la informacion de las personas con las que se comparte el mismo tutor si se selecciona una de estas importa los datos para el apartado direccion**

- **Access** public

echo función Enrolamiento::brother\_found(\$id) [[line 1071](#)]

**Parámetros de la función:**

- *string \$id* id de la persona

**Este metodo verifica si un paciente comparte un mismo tutor**

- **Access** public

echo función Enrolamiento::catalog\_check(\$catalog, \$tipo, [\$col = 1], [\$sel = ""], [\$orden = ""]) [[line 417](#)]

**Parámetros de la función:**

- *string \$catalog* tabla de donde se extrae la informacion
- *string \$tipo* tipo de control radio o check

- *int \$col* numero de columnas en la tabla
- *string \$sel* identifica si un valor ya esta seleccionado
- *string \$orden* columna para hacer el ordenamiento

## Crea un grupo de radio o check con la informacion de los catalogos

- **Access** public

*echo función Enrolamiento::catalog\_select(\$catalog, [\$sel = ""], [\$orden = ""], [\$campo = ""], [\$valor = ""]) [line [338](#)]*

### Parámetros de la función:

- *string \$catalog* tabla de donde se extrae la informacion
- *string \$sel* identifica si un valor ya esta seleccionado
- *string \$orden* columna para hacer el ordenamiento
- *string \$campo* campo de la tabla para hacer el where
- *string \$valor* valor a comparar en el where

## Genera los options de un campo tipo select

- **Access** public

*echo función Enrolamiento::categoriacie10\_select() [line [1333](#)]*

## Genera los options de un campo tipo select para las categorías de CIE10

- **Access** public

*bool función Enrolamiento::chechar\_session() [line [1310](#)]*

## Revisa si la sesión está activa

- **Access** public

echo función Enrolamiento::cie10\_select(\$categoria) [line[1360](#)]

**Parámetros de la función:**

- *string \$categoria* Categoría de la CIE10

**Genera los options de un campo tipo select para los CIE10 correspondientes a una categoría**

- **Access** public

echo función Enrolamiento::comparar\_view(\$id, [\$prod1 = ""], [\$prod2 = ""], [\$prod3 = ""]) [line [1241](#)]

**Parámetros de la función:**

- *string \$id* identificador de la persona
- *\$prod1*
- *\$prod2*
- *\$prod3*

**Crea la pagina para ver la infromacion de la persona y comprararla con la persona capturada**

- **Access** public

echo función Enrolamiento::data\_tutor(\$curp) [line[492](#)]

**Parámetros de la función:**

- *string \$curp* curp del tutor

### Obtiene información del tutor

- **Access** public

*echo función Enrolamiento::file\_to\_card(\$id, [\$tipo = ""])* [line [538](#)]

#### Parámetros de la función:

- *string \$id* identificador de la persona
- *\$tipo*

### crea un archivo descargable el cual se necesita para el envío por nfc a la tarjeta del paciente

- **Access** public

*echo función Enrolamiento::ifCurpExists(\$curp)* [line [985](#)]

#### Parámetros de la función:

- *string \$curp* curp de la persona

### Valida que la curp del paciente no exista

- **Access** public

`echo función Enrolamiento::ifCurpTExists($curp) [line1029]`

**Parámetros de la función:**

- `string $curp` curp de la persona

**valida que la curp del tutor no exista**

- **Access** public

`echo función Enrolamiento::index([$pag = 0], [$id = ""], [$array = ""]) [line38]`

**Parámetros de la función:**

- `string $pag` numero de pagina para la posicion
- `string $id` id de una persona
- `$array`

**Este es el metodo por default, obtiene el listado de las personas**

se recibe el parametro \$pag de tipo int que representa la paginacion

- **Access** public

`echo función Enrolamiento::insert() [line766]`

**prepara los datos para insertarlos**

- **Access** public

```
json($porcentaje_similitud,$persona) función Enrolamiento::paciente_similar($nombre, $paterno, $materno,  
$curp, $nacimiento, $lugar, [$calle = ""], [$referencia = ""], [$colonia = ""],  
[$curpT = ""], $cp, $numero) [line 1190]
```

**Parámetros de la función:**

- *string \$nombre* nombre del paciente que se esta capturando
- *string \$paterno* apellido paterno del paciente
- *string \$materno* apellido materno del paciente
- *string \$curp* curp del paciente
- *string \$nacimiento* fecha de nacimiento del paciente
- *string \$calle* calle del domicilio del paciente
- *string \$referencia* referencia del domicilio del paciente
- *string \$colonia* colonia del paciente
- *string \$cp* cp de la colonia donde vive el paciente
- *string \$numero* numero de la vivienda
- *\$lugar*
- *\$curpT*

**Comprueba la similitud de un paciente que se este capturando con los que ya existe en la base de datos, esto con la finalidad de disminuir datos repetidos**

- **Access** public

```
echo función Enrolamiento::print_card($id) [line 87]
```

**Parámetros de la función:**

- *string \$id* identificador de la persona

**Este metodo estraе la informacion del paciente que sera impreso en la tarjeta**

- **Access** public

```
Object función Enrolamiento::searchageb([$idasulocalidad = ""], [$like = ""]) [line 1158]
```

**Parámetros de la función:**

- *int \$idasulocalidad* Id de la localidad en el ASU
- *\$like*

**Regresa un objeto JSON con la lista de agebs disponibles en la localidad Solo se permite su acceso por medio de peticiones AJAX**

- **Access** public

*Object función Enrolamiento::searchum(\$idasulocalidad, \$ageb) [line 130]*

**Parámetros de la función:**

- *int \$idasulocalidad* Id de la localidad en el ASU
- *string \$ageb* Numero de ageb

**Busca dentro del catalogoasu\_ageb la unidad médica de acuerdo a la localidad y ageb Solo se permite su acceso por medio de peticiones AJAX**

- **Access** public

*echo función Enrolamiento::tratamiento\_select([\$campo = ""], [\$valor = ""], [\$sel = ""], [\$orden = ""]) [line 371]*

**Parámetros de la función:**

- *string \$campo* campo de la tabla para hacer el where
- *string \$valor* valor a comparar en el where
- *string \$sel* identifica si un valor ya esta seleccionado
- *string \$orden* columna para hacer el ordenamiento

**Genera los options de un campo tipo select para los tratamientos de consultas**

- **Access** public

echo función Enrolamiento::update(\$id) [\[line 194\]](#)

**Parámetros de la función:**

- *string \$id* identificador de la persona

### Crea el formulario para editar la informacion de la persona

- **Access** public

echo función Enrolamiento::update\_card(\$persona, \$impreso, [\$fecha = ""], [\$archivo = ""], [\$entorno = '4']) [\[line 739\]](#)

**Parámetros de la función:**

- *string \$persona* id de la persona
- *boolean \$impreso* identifica si el proceso de impresion fue correcto o no
- *int \$fecha* fecha del evento
- *string \$archivo* archivo generado
- *string \$entorno* tipo de entorno

**Este metodo actualiza el estado del archivo descargado si fue escrito correctamente o no en la tarjeta**

- **Access** public

echo función Enrolamiento::vacunacion(\$fecha, [\$id = ""]) [\[line 1291\]](#)

**Parámetros de la función:**

- *date \$fecha*
- *string \$id*

**Obtiene el historial de vacunacion de un paciente cuyo id es pasado por parametro**

- **Access** public

*echo función Enrolamiento::validarForm([\$op = ""])* [line [833](#)]

**Parámetros de la función:**

- *string \$op* bandera que identifica si una seccion entra en validacion

**valida los datos de entrada en el formulario**

- **Access** public

*echo función Enrolamiento::validarisum(\$id)* [line [1107](#)]

**Parámetros de la función:**

- *string \$id* id de arbol de segmentacion

**valida que el nodo seleccionado en el arbol sea una unidad medica**

- **Access** public

`echo función Enrolamiento::validate_card($persona, $archivo) [line 755]`

**Parámetros de la función:**

- `string $persona` id de la persona
- `string $archivo` archivo generado

**valida que un archivo sea valido para enviar a la tarjeta por nfc**

- **Access** public

`echo función Enrolamiento::view($id) [line 134]`

**Parámetros de la función:**

- `string $id` identificador de la persona

**Crea la pagina para ver la infromacion de la persona**

- **Access** public

## Clase Notificacion

*[line 10]*

**Controlador Notificación**

- **Package TES**
- **Sub-Package Controlador**

- **Author** Rogelio

Constructor void función Notificacion::\_\_construct() [line 12]

- **Access** public

void función Notificacion::delete(\$id) [line 255]

**Parámetros de la función:**

- *int \$id* id de notificación a eliminar

### Solicita la eliminación de la notificación recibida

- **Access** public

void función Notificacion::index([\$pag = 0]) [line 34]

**Parámetros de la función:**

- *int \$pag* número de página a visualizar (paginación)

**1) Visualiza las notificaciones existentes para su interacción CRUD 2) En caso de detectar un texto a buscar se filtran las notificaciones existentes acorde a la búsqueda**

- **Access** public

void función Notificacion::insert() [line 39]

**1) Prepara el formulario para la inserción de una notificación nueva 2) Realiza las**

## **validaciones necesarias sobre cada campo del registro**

- **Access** public

*void función Notificacion::update(\$id) [line 197]*

**Parámetros de la función:**

- *int \$id* id de la notificación a modificar

**1) Prepara el formulario para la modificación de una notificación existente 2) Realiza las validaciones necesarias sobre cada campo del registro**

- **Access** public

*void función Notificacion::view(\$id) [line 103]*

**Parámetros de la función:**

- *int \$id* id de notificación a visualizar

**Visualiza los datos de la notificación recibida**

- **Access** public

# Clase Reporteador

[line 10]

## Controlador Reporteador

- **Package** TES
- **Sub-Package** Controlador
- **Author** Rogelio

Constructor `void` función Reporteador::`__construct()` [line 12]

- **Access** public

`void` función Reporteador::`index()` [line 82]

### Visualiza los reportes existentes

- **Access** public

`void` función Reporteador::`view($op, $title, $nivel, $id, [$fecha = "])` [line 73]

#### Parámetros de la función:

- `int $op`
- `string $title`
- `int $nivel`
- `int $id`
- `string $fecha`

### Renderiza la vista del reporte

- **Access** public

# Clase Reporte\_sincronizacion

[line 10]

## Controller Usuario

- **Package** TES
- **Sub-Package** Controlador
- **Author** Eliecer

Constructor `void` función `Reporte_sincronizacion::__construct()` [line 13]

- **Access** public

`void` función `Reporte_sincronizacion::index()` [line 35]

**Este es el metodo por default, crea el listado del reporte de sincronizacion**

- **Access** public

`void` función `Reporte_sincronizacion::lote()` [line 175]

**Genera el item del reporte de lote de vacunacion**

- **Access** public

*void función Reporte\_sincronizacion::lote\_view(\$lote, \$title, \$op, [\$lugar = "Chiapas"]) [line*

[344](#)]

**Parámetros de la función:**

- *string \$lote* Valor del lote del que se esta generando el reporte
- *string \$title* Especifica el titulo a mostrar en el reporte
- *string \$op* Especifica el reporte a mostrar
- *string \$lugar* Especifica el lugar donde se centrara el mapa

**Muestra detallada por cada list del reporte de lote de vacunacion**

- **Access** public

*void función Reporte\_sincronizacion::view(\$op, \$title) [line*[103](#)]

**Parámetros de la función:**

- *string \$op* Especifica el reporte a mostrar
- *string \$title* Especifica el titulo a mostrar en el reporte

**Muestra detallada por cada list del reporte de sincronizacion**

- **Access** public

## Clase Semana\_nacional [line [11](#)]

### Controlador Semana Nacional

- **Package** TES
- **Sub-Package** Controlador
- **Author** Pascual

Constructor `void` función `Semana_nacional::__construct()` [line 13]

- **Access** public

`void` función `Semana_nacional::delete($id)` [line 245]

**Parámetros de la función:**

- `int $id` ID del elemento a eliminar

### **Eliminar el registro especificado por el id**

- **Access** public

`json` función `Semana_nacional::getAll()` [line 278]

**Devuelve un json con todos los registros de semanas nacional**

- **Access** public

`void` función `Semana_nacional::index([$pag = 0])` [line 35]

**Parámetros de la función:**

- `int $pag` Establece el desplazamiento del primer registro a devolver

**Lista todos los registros de semanas nacional, con su correspondiente paginación permite eliminar un elemento individual, muestra enlaces para actualizar y ver detalles de un elemento específico**

- **Access** public

*void función Semana\_nacional::insert() [line 92]*

**Muestra el formulario para crear un nuevo registro en la semana\_nacional, las variables se obtienen por el metodo POST**

- **Access** public

*void función Semana\_nacional::update(\$id) [line 151]*

**Parámetros de la función:**

- *int \$id* ID del elemento a actualizar

**Muestra el formulario con los datos del registro especificado por el id, para actualizar sus datos**

- **Access** public

*void función Semana\_nacional::view(\$id) [line 205]*

**Parámetros de la función:**

- *int \$id* ID del elemento a actualizar

## Muestra los datos del registro especificado por el id

- **Access** public

## Clase Servicios

[line [10](#)]

### Controlador Servicios

- **Package** TES
- **Sub-Package** Controlador
- **Author** Eliecer

Constructor **void** función Servicios::\_\_construct() [line [11](#)]

- **Access** public

echo función Servicios::actualiza\_estado\_tableta(\$id\_sesion, [\$id\_tes\_estado\_tableta = ""],  
[\$version = ""], \$id\_session) [line [1025](#)]

**Parámetros de la función:**

- **int \$id\_session** Representa la session activa para la peticion
- **json \$id\_tes\_estado\_tableta** Representa el status que tomara la tableta
- **int \$version** Representa la version de la apk instalada en la tableta
- **\$id\_sesion**

### Actualiza el estatus de la tableta

- **Access** public

echo función Servicios::catalogos\_relevantes([\$sf = ""]) [line [1102](#)]

**Parámetros de la función:**

- *string \$sf* bandera que activa el filtro de fechas segun el tipo de sincronizacion

## Genera los catalogos relevantes por entorno

- **Access** public

echo función Servicios::esquema\_incompleto(\$id\_persona, \$fecha, \$vacunas) [line [1053](#)]

**Parámetros de la función:**

- *int \$id\_persona* Representa la persona a la que se le calculara su esquema
- *string \$fecha* Representa la fecha de nacimiento de la persona
- *array \$vacunas* Representa las vacunas aplicadas a la persona

## Genera los esquemas incompletos de las personas que correspondan a la unidad medica de la tableta

- **Access** public

void función Servicios::is\_step\_0(\$id\_accion, [\$id\_tab = null], [\$id\_sesion = null], [\$id\_version = null], [\$datos = null], \$id\_session, \$version\_apk) [line [79](#)]

**Parámetros de la función:**

- *int \$id\_accion* Representa el tipo de accion que se ejecutara
- *int \$id\_tab* Representa a la MAC de la tableta

- *int \$id\_session* Representa la session activa para la peticion
- *int \$version\_apk* Representa la version de la apk instalada en la tableta
- *json \$datos* Representa el json que contiene el contenido para la comunicacion tableta - servidor
- **\$id\_sesion**
- **\$id\_version**

### **Paso 0 se procesa las peticiones segun la accion:**

Si la acci?n es 1: Valida la disponibilidad del dispositivo especificado y genera una session que se mantiene activa en toda la sincronizacion Si la acci?n es 2: Regresa la informacion de todos los catalogos Si la acci?n es 3: Recibe un mensaje si es ok actualiza el estado de la tableta si es error se crea un archivo log con la descripcion Si la acci?n es 4: Regresa la informacion de la persona que pertenescan a la unidad medica de la tableta Si la acci?n es 5: Recibe la informacion que envia la tableta y la almacena en sus respectivas tablas Si la acci?n es 6: Regresa la informacion de los catalogos y personas que se actualizaron o agregaron despues de la ultima sincronizacion de la tableta

- **Access public**

*session* función Servicios::is\_step\_1(\$id\_tab, \$id\_version, \$version\_apk) [\[line 128\]](#)

#### **Parámetros de la función:**

- *int \$id\_tab* Representa a la MAC de la tableta
- *int \$version\_apk* Representa la version de la apk instalada en la tableta
- **\$id\_version**

**valida que la tableta este asignada a una unidad medica y que tenga un status valido para la sincronizacion**

recibe parametros por POST

- **Access public**

*echo* función Servicios::is\_step\_2(\$id\_sesion, [\$si = ""], [\$sf = ""], \$id\_session) [\[line](#)

[207\]](#)

**Parámetros de la función:**

- *int \$id\_session* Representa la session activa para la peticion
- *int \$si* Bandera que especifica si este paso es llamado por otro paso
- *json \$sf* Bandera que especifica el comportamiento del armado del json
- **\$id\_sesion**

**Valida que la session este activa, genera los catalogos a enviar en la sincronizacion por primera vez**

- **Access public**

*void función Servicios::is\_step\_3(\$id\_sesion, \$datos, \$id\_session) [line 466]*

**Parámetros de la función:**

- *int \$id\_session* Representa la session activa para la peticion
- *json \$datos* Representa el json que contiene el contenido para la comunicacion tableta - servidor
- **\$id\_sesion**

**Guarda en la base de datos el estado de la sincronizacion**

- **Access public**

*echo función Servicios::is\_step\_4(\$id\_sesion, \$id\_session) [line 502]*

**Parámetros de la función:**

- *int \$id\_session* Representa la session activa para la peticion
- **\$id\_sesion**

**Prepara la informacion de personas con su catalogos transaccionales de cada una**

- **Access** public

void función Servicios::prueba2(\$id\_accion, [\$id\_tab = null], [\$id\_sesion = null], [\$version = null]) [\[line 119\]](#)

**Parámetros de la función:**

- **\$id\_accion**
- **\$id\_tab**
- **\$id\_sesion**
- **\$version**

- **Access** public

echo función Servicios::ss\_step\_5(\$id\_sesion, \$datos, \$id\_session) [\[line 689\]](#)

**Parámetros de la función:**

- **int \$id\_session** Representa la session activa para la peticion
- **json \$datos** Representa el json que contiene el contenido para la comunicacion tableta - servidor
- **\$id\_sesion**

**Recibe los datos que genero la tableta para ser almacenados en la base de datos del servidor**

- **Access** public

void función Servicios::ss\_step\_6(\$id\_sesion, \$id\_session) [\[line 835\]](#)

**Parámetros de la función:**

- **int \$id\_session** Representa la session activa para la peticion

- **\$id\_sesion**

**prepara los datos para la sincronizacion secuencia, envia unicamente aquellos datos modificados despues de la ultima sincronizacion de la tableta**

- **Access public**

*void función Servicios::Synchronization(\$id\_accion, \$id\_tab, \$id\_session, \$version\_apk, \$datos) [line 43]*

**Parámetros de la función:**

- *int \$id\_accion* Representa el tipo de accion que se ejecutara
- *int \$id\_tab* Representa a la MAC de la tableta
- *int \$id\_session* Representa la session activa para la peticion
- *int \$version\_apk* Representa la version de la apk instalada en la tableta
- *json \$datos* Representa el json que contiene el contenido para la comunicacion tableta - servidor

**Metodo principal al que se le hacen las peticiones y es el que se encarga de distribuir la informacion**  
recibe parametros por POST

- **Access public**

## Clase Tableta

[line 11]

**Controlador Tableta**

- **Package** TES
- **Sub-Package** Controlador
- **Author** Pascual

Constructor `void` función Tableta::`__construct()` [line 13]

- **Access** public

`void` función Tableta::`delete($id)` [line 340]

**Parámetros de la función:**

- `int $id` ID del elemento a eliminar

**Eliminar el registro especificado por el id**

- **Access** public

`void` función Tableta::`index([$pag = 0])` [line 35]

**Parámetros de la función:**

- `int $pag` Establece el desplazamiento del primer registro a devolver

**Lista todos los registros de tabletas, con su correspondiente paginación permite eliminar un conjunto de registro o un elemento individual, muestra enlaces para actualizar y ver detalles de un elemento específico**

- **Access** public

*void función Tableta::insert() [line 163]*

**Muestra el formulario para crear un nuevo registro en la tabla, las variables se obtienen por el metodo POST**

- **Access** public

*redirect función Tableta::setUM(\$id) [line 461]*

**Parámetros de la función:**

- *int \$id*

## **Asignar unidad medica y tipo de censo**

- **Access** public

*void función Tableta::update(\$id) [line 220]*

**Parámetros de la función:**

- *int \$id* ID del elemento a actualizar

**Muestra el formulario con los datos del registro especificado por el id, para actualizar sus datos**

- **Access** public

*redirect* función Tableta::uploadFile() [[line 893](#)]

### Registra tabletas desde un archivo csv

- **Access** public

*void* función Tableta::view(\$id) [[line 285](#)]

#### Parámetros de la función:

- *int \$id* ID del elemento a actualizar

### Muestra los datos del registro especificado por el id

- **Access** public

*boolean* función Tableta::\_validateMac(\$mac) [[line 273](#)]

#### Parámetros de la función:

- *type \$mac*

### Valida si existe una MAC en la base de datos

- **Access** public

# Clase Usuario\_tableta

[line 11]

## Controlador Usuario\_tableta

- **Package** TES
- **Sub-Package** Controlador
- **Author** Pascual

Constructor `void` función `Usuario_tableta::__construct()` [line 13]

- **Access** public

`void` función `Usuario_tableta::delete($id_usuario, $id_tableta, $$id_usuario)` [line 143]

**Parámetros de la función:**

- `int $$id_usuario` ID del usuario a eliminar
- `int $id_tableta` ID de la tableta que tiene asignada el usuario especificado
- `$id_usuario`

## Eliminar un usuario de una tableta específica

- **Access** public

`void` función `Usuario_tableta::index($idTableta)` [line 36]

**Parámetros de la función:**

- `int $idTableta` ID de la Tableta

**Lista todos los registros de usuarios correspondientes a una tableta en específico  
permite eliminar un conjunto de registro o un elemento individual, muestra  
enlaces para actualizar y ver detalles de un elemento específico**

- **Access** public

void función Usuario\_tableta::insert(\$id\_tableta) [*linea 102*]

**Parámetros de la función:**

- **\$id\_tableta**

**Muestra el formulario para crear un nuevo registro en la tabla, las variables se obtienen por el metodo POST**

- **Access** public

## enrolamiento\_model.php

- **Package TES**
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## estado\_tableta\_model.php

- **Package** TES
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## notificacion\_model.php

- **Package TES**
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## reporteador\_model.php

- **Package TES**
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## reporte\_censo\_nominal.php

- **Package TES**
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## reporte\_sincronizacion\_model.php

- **Package** TES
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## semana\_nacional\_model.php

- **Package TES**
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## tableta\_model.php

- **Package TES**
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## tipo\_censo\_model.php

- **Package TES**
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

## usuario\_tableta\_model.php

- **Package** TES
- **Sub-Package** Modelo
- **Filesource** [Source Code for this file](#)

### Clase Enrolamiento\_model [line 10]

#### Modelo Usuario

- **Package** TES
- **Sub-Package** Modelo
- **Author** Eliecer

Constructor **void** función Enrolamiento\_model::\_\_construct() [line 18]

- **Access** public

**result()** función Enrolamiento\_model::autocomplete\_tutor(\$keywords) [line 2297]

**Parámetros de la función:**

- **string \$keywords** palabras claves para hacer el filtro

**obtiene informacion del tutor para genberar el autocomplete**

- **Access** public

*result()* función Enrolamiento\_model::cns\_insert(\$tabla, \$array) [*line 727*]

**Parámetros de la función:**

- **string \$tabla** Nombre de la tabla a la que se afectara
- **array \$array** Datos que se guardaran en la tabla

**Hace insert de las tablas cns\_control\_x que se reciben en la sincronizacion secuencial**

- **Access** public

*result()* función Enrolamiento\_model::cns\_update(\$tabla, \$array, \$id, [\$campo = ""], [\$valor = ""], [\$campo2 = ""], [\$valor2 = ""]) [*line 751*]

**Parámetros de la función:**

- **string \$tabla** Nombre de la tabla afectada
- **array \$array** Datos a actualizar
- **string \$id** identificador para el where
- **string \$campo** campo si se necesitaran un segundo where
- **string \$valor** valor del campo a comparar
- **\$campo2**
- **\$valor2**

**Actualiza la tabla especificada**

- **Access** public

*void* función Enrolamiento\_model::cns\_update\_visita(\$id) [*line 767*]

**Parámetros de la función:**

- **\$id**
  - **Access** public

*result()* función Enrolamiento\_model::data\_tutor(\$curp) [line [2275](#)]

**Parámetros de la función:**

- *string \$curp* Curp del tutor

**obtiene informacion del tutor**

- **Access** public

*result()* función Enrolamiento\_model::entorno\_x\_persona(\$entorno, \$persona, \$fecha, \$archivo, \$impreso) [line [1110](#)]

**Parámetros de la función:**

- *string \$entorno* id del entorno
- *string \$persona* id de la persona a la que se le asigna una tarjeta
- *string \$fecha* fecha en que se genera el evento
- *string \$archivo* nombre del archivo que se genero
- *boolena \$impreso* determina si el archivo fue escrita en la tarjeta o no

**Este metodo actualiza o inserta los datos que permiten el envio de la informacion a la tarjeta por nfc**

- **Access** public

*void* función Enrolamiento\_model::getaccion\_nutricional() [*line548*]

- **Access** public

*void* función Enrolamiento\_model::getafiliacion() [*line468*]

- **Access** public

*result()* función Enrolamiento\_model::getAfiliaciones([*\$id* = "], [*\$order* = "]) [*line1830*]  
**Parámetros de la función:**

- *string \$id* identificador de la persona
- *string \$order* nombre del campo para hacer el order by

### Obtiene las afiliaciones asociadas a una persona

- **Access** public

*void* función Enrolamiento\_model::getageb() [*line437*]

- **Access** public

*result()* función Enrolamiento\_model::getAlergia([*\$id* = "], [*\$order* = "]) [*line1799*]  
**Parámetros de la función:**

- *string \$id* identificador de la persona
- *string \$order* nombre del campo para hacer el order by

## Obtiene las alergias asociadas a una persona

- **Access** public

*void función Enrolamiento\_model::getalergias() [line 478]*

- **Access** public

*void función Enrolamiento\_model::getaltura() [line 578]*

- **Access** public

*result() función Enrolamiento\_model:: getByCurp(\$curp, \$tabla, \$id) [line 2328]*

**Parámetros de la función:**

- *string \$curp* Curp a validar
- *string \$tabla* Tabla en la que se debe hacer la validacion
- *string \$id* id de la persona o tutor para excluirse

**valida que no se repita la curp en personas y tutor**

- **Access** public

*result() función Enrolamiento\_model:: getById(\$id) [line 1739]*

**Parámetros de la función:**

- *string \$id* identificado de la persona

## **Obtiene la informacion de la persona**

- **Access** public

*void función Enrolamiento\_model::getcalle() [line[887](#)]*

- **Access** public

*object|boolean función Enrolamiento\_model::getCategoriaCIE10() [line[2717](#)]*

## **Obtiene el listado de categorias de CIE10**

- **Access** public

*void función Enrolamiento\_model::getcelular() [line[637](#)]*

- **Access** public

*void función Enrolamiento\_model::getcelularT() [line[837](#)]*

- **Access** public

*object|boolean función Enrolamiento\_model::getCIE10(\$categoria) [line[2746](#)]*

**Parámetros de la función:**

- **\$categoria**

## Obtiene el listado de CIE10 correspondientes a una CIE10

- **Access** public

*void función Enrolamiento\_model::getcodigo\_barras() [line[508](#)]*

- **Access** public

*void función Enrolamiento\_model::getcolonia() [line[897](#)]*

- **Access** public

*void función Enrolamiento\_model::getcompania() [line[627](#)]*

- **Access** public

*void función Enrolamiento\_model::getcompaniaT() [line[327](#)]*

- **Access** public

*void función Enrolamiento\_model::getconsulta() [line[518](#)]*

- **Access** public

*object|boolean* función Enrolamiento\_model::getControlConsultas(\$idPersona) [line[2776](#)]

**Parámetros de la función:**

- **\$idPersona**

**Obtiene todas las consultas asociadas a un paciente**

- **Access** public

*void* función Enrolamiento\_model::getcp() [line[427](#)]

- **Access** public

*void* función Enrolamiento\_model::getcurp() [line[179](#)]

- **Access** public

*void* función Enrolamiento\_model::getcurpT() [line[298](#)]

- **Access** public

*void* función Enrolamiento\_model::getestimulacion\_capacitado() [line[687](#)]

- **Access** public

*void* función Enrolamiento\_model::getestimulacion\_fecha() [line[677](#)]

- **Access** public

*void función Enrolamiento\_model::getfaccion\_nutricional() [line [558](#)]*

- **Access** public

*void función Enrolamiento\_model::getfconsulta() [line [528](#)]*

- **Access** public

*void función Enrolamiento\_model::getfechacivil() [line [347](#)]*

- **Access** public

*void función Enrolamiento\_model::getfecha\_peri\_cefa() [line [667](#)]*

- **Access** public

*void función Enrolamiento\_model::getfnacimiento() [line [209](#)]*

- **Access** public

*void función Enrolamiento\_model::getfnutricion() [line [607](#)]*

- **Access** public

*result()* función Enrolamiento\_model::getfolio(\$persona) [line 158]

**Parámetros de la función:**

- **strin** \$persona id de la persona a la que se le asigna una tarjeta

**Extrae el folio que se anexa en el envío para la tarjeta**

- **Access** public

*void* función Enrolamiento\_model::getfvacuna() [line 498]

- **Access** public

*void* función Enrolamiento\_model::gethemoglobina() [line 597]

- **Access** public

*void* función Enrolamiento\_model::getId() [line 129]

- **Access** public

*void* función Enrolamiento\_model::getidtutor() [line 259]

- **Access** public

*result()* función Enrolamiento\_model::getListEnrolamiento([**\$keywords** = "], [**\$offset** = null], [**\$row\_count** = null])  
[line [1594](#)]

**Parámetros de la función:**

- **string \$keywords** palabras claves para hacer el filtro
- **string \$offset** inicio del registro
- **string \$row\_count** numero de filas a mostrar por pagina

**Este metodo retorna el list de las personas enroladas**

- **Access** public

*void* función Enrolamiento\_model::getInacimiento() [line [169](#)]

- **Access** public

*void* función Enrolamiento\_model::getlocalidad() [line [407](#)]

- **Access** public

*void* función Enrolamiento\_model::getlugarcivil() [line [257](#)]

- **Access** public

*void* función Enrolamiento\_model::getmanzana() [line [457](#)]

- **Access** public

*void función Enrolamiento\_model::getmaterno() [line 159]*

- **Access** public

*void función Enrolamiento\_model::getmaternoT() [line 289]*

- **Access** public

*void función Enrolamiento\_model::getMsgError([\$value = 'usr']) [line 2386]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::getnacionalidad() [line 647]*

- **Access** public

*void función Enrolamiento\_model::getnombre() [line 139]*

- **Access** public

*void función Enrolamiento\_model::getnombreT() [line 269]*

- **Access** public

*void función Enrolamiento\_model::getnumero() [line 417]*

- **Access** public

*result() función Enrolamiento\_model::getNumRows([\$keywords = "]) [line 1670]*

**Parámetros de la función:**

- *string \$keywords* palabras clave para hacer el filtro

**Devuelve el numero de filas en la tabla cns\_persona**

- **Access** public

*void función Enrolamiento\_model::getparto() [line 229]*

- **Access** public

*void función Enrolamiento\_model::getpaterno() [line 149]*

- **Access** public

*void función Enrolamiento\_model::getpaternoT() [line 279]*

- **Access** public

*void función Enrolamiento\_model::getperi\_cefa() [line 657]*

- **Access** public

*void función Enrolamiento\_model::getpeso() [line 568]*

- **Access** public

*void función Enrolamiento\_model::getprecurp() [line 239]*

- **Access** public

*void función Enrolamiento\_model::getreferencia() [line 877]*

- **Access** public

*result() función Enrolamiento\_model::getRegistro\_civil(\$id) [line 1772]*

**Parámetros de la función:**

- *string \$id* identificador de la persona

**obtiene informacion del registro civil**

- **Access** public

*void función Enrolamiento\_model::getsales\_cantidad() [line 697]*

- **Access** public

*void función Enrolamiento\_model::getsales\_fecha() [line 707]*

- **Access** public

*void función Enrolamiento\_model::getsangre() [line 199]*

- **Access** public

*void función Enrolamiento\_model::getsector() [line 447]*

- **Access** public

*void función Enrolamiento\_model::getsexo() [line 189]*

- **Access** public

*void función Enrolamiento\_model::getsexoT() [line 808]*

- **Access** public

*void función Enrolamiento\_model::gettalla() [line 587]*

- **Access** public

*void* función Enrolamiento\_model::gettamiz() [*line 234*]

- **Access** public

*void* función Enrolamiento\_model::gettbeneficiario() [*line 219*]

- **Access** public

*void* función Enrolamiento\_model::gettconsulta() [*line 538*]

- **Access** public

*void* función Enrolamiento\_model::gettelefono() [*line 617*]

- **Access** public

*void* función Enrolamiento\_model::gettelefonoT() [*line 817*]

- **Access** public

*void* función Enrolamiento\_model::getumt() [*line 867*]

- **Access** public

*void* función Enrolamiento\_model::getvacuna() [*line 488*]

- **Access** public

*result()* función Enrolamiento\_model::get\_catalog(\$catalog, [\$campo = ""], [\$id = ""],  
[\$orden = ""], \$order) [line [1925](#)]

**Parámetros de la función:**

- *string* **\$catalog** Nombre de la tabla
- *string* **\$campo** nombre del campo para hacer el where
- *string* **\$id** valor del campo para el where
- *string* **\$order** nombre del campo para hacer el order by
- **\$orden**

## Hace select de las tablas cns\_x que representa a los catalogos

- **Access** public

*result()* función Enrolamiento\_model::get\_catalog2(\$catalog, [\$campo1 = ""], [\$id1 = ""],  
[\$campo2 = ""], [\$id2 = ""], [\$l1 = ""], [\$l2 = ""]) [line [2022](#)]

**Parámetros de la función:**

- *string* **\$catalog** Nombre de la tabla
- *string* **\$campo1** nombre del campo para hacer el where
- *string* **\$id1** valor del campo para el where
- *string* **\$campo2** nombre del campo para hacer el where
- *string* **\$id2** valor del campo para el where
- *string* **\$l1** nombre del campo para hacer el limit offset
- *string* **\$l2** nombre del campo para hacer el limit count

## Obtiene los datos de una tabla

- **Access** public

*count* función Enrolamiento\_model::get\_catalog\_count(\$catalog, [\$campo = ""], [\$valor = ""]) [line [1995](#)]

**Parámetros de la función:**

- *string \$catalog* Nombre de la tabla
- *\$campo*
- *\$valor*

**Obtiene el numero de resultados de una tabla**

- **Access** public

*result()* función Enrolamiento\_model::get\_catalog\_relevante([\$fecha = ""]) [line [2199](#)]

**Parámetros de la función:**

- *\$fecha*

**obtiene los catalogos relevante x entorno para la sincronizacion**

- **Access** public

*result()* función Enrolamiento\_model::get\_catalog\_tratamiento(\$catalog, \$campo, \$valor, \$orden, \$order) [line [1961](#)]

**Parámetros de la función:**

- *string \$catalog* Nombre de la tabla
- *strin \$campo* nombre del campo para hacer el where
- *string \$valor* valor del campo para el where
- *strin \$order* nombre del campo para hacer el order by
- *\$orden*

## Hace select de los tratamientos de las consultas

- **Access** public

*result()* función Enrolamiento\_model::get\_catalog\_view(\$catalog, \$id, [\$order1 = ""], [\$order2 = ""]) [line [1862](#)]

**Parámetros de la función:**

- *string \$catalog* Nombre de la tabla
- *string \$id* identificador de la persona
- *string \$order1* nombre del campo para hacer el order by
- *string \$order2* nombre del campo para hacer el order by

## Hace select de los catalogos que tengan relacion con una persona para mostrarlos en el view

- **Access** public

*result()* función Enrolamiento\_model::get\_cns\_cat\_persona(\$catalog, \$array, [\$l1 = ""], [\$l2 = ""]) [line [2123](#)]

**Parámetros de la función:**

- *string \$catalog* Nombre de la tabla
- *string \$array* ids de personas
- *string \$l1* offset para el limit
- *string \$l2* count para el limit

## Este metodo obtiene los controles que le corresponde a cada persona y que seran incluidas en la sincronizacion

- **Access** public

*result()* función Enrolamiento\_model::get\_cns\_cat\_persona\_count(\$catalog, \$persona, \$personas) [line [2158](#)]  
**Parámetros de la función:**

- *string \$catalog* Nombre de la tabla
- *string \$personas* personas que cumplen el requisito
- *\$persona*

### Hace el count de personas que se envian en la sincronizacion

- **Access** public

*result()* función Enrolamiento\_model::get\_cns\_persona(\$array, [\$fecha = ""]) [line [2089](#)]  
**Parámetros de la función:**

- *string \$array* arreglo con los ids de las personas que cumplen con los requisitos del envio
- *string \$fecha* fecha que determina si se envia o no una persona

### Este metodo obtiene las personas que seran enviadas en la sincronizacion

- **Access** public

*result()* función Enrolamiento\_model::get\_control\_nutricional(\$id, [\$order = ""]) [line [1893](#)]  
**Parámetros de la función:**

- *string \$id* identificador de la persona
- *string \$order* nombre del campo para hacer el order by

### Obtiene los datos del control nutricional asociados a una persona

- **Access** public

*void función Enrolamiento\_model::get\_datos\_grafica(\$catalogo, \$sexo, \$edad\_meses, \$id\_persona, [\$asu\_locali = "]) [line [2404](#)]*

**Parámetros de la función:**

- *int \$catalogo* Determina el catalogo a consultar
- *int \$sexo* El sexo del paciente (F, M)
- *int \$edad\_meses* Edad del paciente en meses
- *int \$id\_persona* Identificador del paciente
- *int \$asu\_locali* Identificador del asu de la localidad del domicilio

**Obtiene los datos específicos de un catálogo para ser visualizados en una gráfica**

- **Access** public

*result() función Enrolamiento\_model::get\_estimulacion(\$id, [\$order = "]) [line [2661](#)]*

**Parámetros de la función:**

- *string \$id* identificador de la persona
- *string \$order* nombre del campo para hacer el order by

**Obtiene los registros de estimulacion temprana asociados a una persona**

- **Access** public

*result() función Enrolamiento\_model::get\_notificacion(\$id) [line [2062](#)]*

**Parámetros de la función:**

- *string \$id* id del arbol de segmentacion

**Este metodo obtiene las notificaciones que se enviaran en la sincronizacion**

- **Access** public

*result()* función Enrolamiento\_model::get\_pacientes() [line [2368](#)]

**devuelve todos los pacientes de la base de datos**

- **Access** public

*result()* función Enrolamiento\_model::get\_peri\_cefa(\$id, [\$order = ""]) [line [2601](#)]

**Parámetros de la función:**

- *string \$id* identificador de la persona
- *string \$order* nombre del campo para hacer el order by

**Obtiene los registros de perimetro cefalico asociados a una persona**

- **Access** public

*result()* función Enrolamiento\_model::get\_persona\_x\_tutor(\$array) [line [2176](#)]

**Parámetros de la función:**

- *string \$array* tutores que tienen asignado un paciente

**obtiene los tutores de las personas que se envian en la sincronizacion**

- **Access** public

*result()* función Enrolamiento\_model::get\_sales(\$id, [\$order = ""]) [line [2866](#)]

**Parámetros de la función:**

- **string \$id** identificador de la persona
- **string \$order** nombre del campo para hacer el order by

**Obtiene los registros de sales de rehidratación oral asociados a una persona**

- **Access** public

*result()* función Enrolamiento\_model::get\_transaction\_relevante() [line [2225](#)]

**Obtiene las transacciones relevante para la sincronizacion**

- **Access** public

*result()* función Enrolamiento\_model::get\_version() [line [2249](#)]

**obtiene cual es la ultima version de apk de la tablet**

- **Access** public

*result()* función Enrolamiento\_model::insert() [line 787]

### Guarda la persona capturada mediante el formulario web

- **Access** public

*void* función Enrolamiento\_model::setaccion\_nutricional(\$value) [line 553]

#### Parámetros de la función:

- **\$value**

- **Access** public

*void* función Enrolamiento\_model::setafiliacion(\$value) [line 473]

#### Parámetros de la función:

- **\$value**

- **Access** public

*void* función Enrolamiento\_model::setageb(\$value) [line 442]

#### Parámetros de la función:

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setalergias(\$value) [line 483]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setaltura(\$value) [line 583]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setcalle(\$value) [line 892]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setcelular(\$value) [line 642]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setcelularT(\$value) [line 842]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setcodigo\_barras(\$value) [line 513]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setcolonia(\$value) [line 402]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setcompania(\$value) [line 632]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setcompaniaT(\$value) [line 832]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setconsulta(\$value) [line 523]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setcp(\$value) [line 432]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setcurp(\$value) [line 184]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setcurpT(\$value) [line 303]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setestimulacion\_capacitado(\$value) [line 692]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setestimulacion\_fecha(\$value) [line 682]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setfaccion\_nutricional(\$value) [line 563]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setfconsulta(\$value) [line 533]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setfechacivil(\$value) [line 852]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setfecha\_peri\_cefa(\$value) [line 672]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setfnacimiento(\$value) [line 214]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setfnutricion(\$value) [line 612]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setfvacuna(\$value) [line 503]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::sethemoglobina(\$value) [line 602]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setId(\$value) [line 134]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setidtutor(\$value) [line 264]*

**Parámetros de la función:**

- **\$value**

- **Access** public

void función Enrolamiento\_model::setInacimiento(\$value) [line[174](#)]

**Parámetros de la función:**

- **\$value**

- **Access** public

void función Enrolamiento\_model::setlocalidad(\$value) [line[412](#)]

**Parámetros de la función:**

- **\$value**

- **Access** public

void función Enrolamiento\_model::setlugarcivil(\$value) [line[862](#)]

**Parámetros de la función:**

- **\$value**

- **Access** public

void función Enrolamiento\_model::setmanzana(\$value) [line[462](#)]

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setmaterno(\$value) [line 164]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setmaternoT(\$value) [line 293]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setnacionalidad(\$value) [line 652]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setnombre(\$value) [line 144]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setnombreT(\$value) [line 274]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setnumero(\$value) [line 422]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setparto(\$value) [line 249]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setpaterno(\$value) [line 154]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setpaternoT(\$value) [line 284]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setperi\_cefa(\$value) [line 662]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setpeso(\$value) [line 573]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setprecurp(\$value) [line 244]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setreferencia(\$value) [line 882]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setsales\_cantidad(\$value) [line 702]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setsales\_fecha(\$value) [line[712](#)]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setsangre(\$value) [line[204](#)]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setsector(\$value) [line[452](#)]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setsexo(\$value) [line[194](#)]*

**Parámetros de la función:**

- **\$value**
  - **Access** public

*void función Enrolamiento\_model::setsexoT(\$value) [line 813]*

**Parámetros de la función:**

- **\$value**
  - **Access** public

*void función Enrolamiento\_model::settalla(\$value) [line 592]*

**Parámetros de la función:**

- **\$value**
  - **Access** public

*void función Enrolamiento\_model::settamiz(\$value) [line 254]*

**Parámetros de la función:**

- **\$value**

- **Access** public

void función Enrolamiento\_model::settbeneficiario(\$value) [[line 224](#)]

**Parámetros de la función:**

- **\$value**

- **Access** public

void función Enrolamiento\_model::settconsulta(\$value) [[line 543](#)]

**Parámetros de la función:**

- **\$value**

- **Access** public

void función Enrolamiento\_model::settelefono(\$value) [[line 622](#)]

**Parámetros de la función:**

- **\$value**

- **Access** public

void función Enrolamiento\_model::settelefonoT(\$value) [[line 922](#)]

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setumt(\$value) [line 372]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void función Enrolamiento\_model::setvacuna(\$value) [line 493]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*result() función Enrolamiento\_model::tes\_pendientes\_tarjeta\_delete() [line 2350]*

**Elimina los pendientes de las personas que no tengan asignado una tarjeta**

- **Access** public

*result()* función Enrolamiento\_model::update\_accion() [line [1459](#)]

### **Actualiza el control accion nutricional del paciente**

- **Access** public

*result()* función Enrolamiento\_model::update\_alergia() [line [1360](#)]

### **Actualiza los datos de las alergias del paciente**

- **Access** public

*result()* función Enrolamiento\_model::update\_basico() [line [1171](#)]

### **Actualiza los datos basicos del paciente**

- **Access** public

*result()* función Enrolamiento\_model::updateBeneficiario() [line [1528](#)]

### **Actualiza el tipo de beneficiario del paciente**

- **Access** public

*result()* función Enrolamiento\_model::update\_consulta() [line [1424](#)]

### **Actualiza el control consulta del paciente**

- **Access** public

*result()* función Enrolamiento\_model::update\_direccion() [line [1227](#)]  
**actualiza la direccion del paciente**

- **Access** public

*result()* función Enrolamiento\_model::update\_estimulacion() [line [2689](#)]  
**Actualiza los registros de estimulacion temprana**

- **Access** public

*result()* función Enrolamiento\_model::update\_nutricion() [line [492](#)]  
**Actualiza el control nutricional del paciente**

- **Access** public

*result()* función Enrolamiento\_model::update\_peri\_cefa() [line [2629](#)]  
**Actualiza los registros de perimetro cefalico**

- **Access** public

*result()* función Enrolamiento\_model::update\_regcivil() [line [1265](#)]  
**actualiza el registro civil del paciente**

- **Access** public

*result()* función Enrolamiento\_model::update\_sales() [line[2834](#)]

**Actualiza los registros de sales de rehidratacion oral**

- **Access** public

*result()* función Enrolamiento\_model::update\_status\_tableta(\$mac, \$status, \$version, \$fecha) [line[1566](#)]

**Parámetros de la función:**

- *string \$mac* Mac de la tableta
- *string \$status* nuevo status
- *string \$version* version de la apk de la tableta
- *string \$fecha* fecha del evento

**Hace update de la tableta que este sincronizando dependiendo del resultado**

- **Access** public

*result()* función Enrolamiento\_model::update\_tutor() [line[1289](#)]

**Actualiza los datos del tutor del paciente**

- **Access** public

*result()* función Enrolamiento\_model::update\_umt() [line[1204](#)]

**Actualiza la unidad medica tratante del paciente**

- **Access** public

*result()* función Enrolamiento\_model::update\_vacuna() [\[line 1391\]](#)

### **Actualiza las vacunas del paciente**

- **Access** public

*result()* función Enrolamiento\_model::valid\_card(\$persona, \$archivo) [\[line 1144\]](#)

#### **Parámetros de la función:**

- *string* **\$persona** id de la persona a la que se le asigna una tarjeta
- *string* **\$archivo** nombre del archivo que se genero

**Este metodo valida que exista un archivo para enviar a la tarjeta por nfc**

- **Access** public

## **Clase Estado\_tableta\_model**

[\[line 11\]](#)

### **Modelo Estado\_tableta**

- **Package** TES
- **Sub-Package** Modelo
- **Author** Pascual

Constructor `void` función `Estado_tableta_model::__construct()` [line 49]

- **Access** public

`array` función `Estado_tableta_model::getAll()` [line 140]

**Obtiene todos los registros de la tabla**

- **Access** public

`object|boolean` función `Estado_tableta_model::getById($id)` [line 112]

**Parámetros de la función:**

- `int $id` Si no se establece el valor de ID, se toma el valor del objeto actual

**Obtiene los datos del registro que tiene el ID especificado**

- **Access** public

`void` función `Estado_tableta_model::getDescripcion()` [line 67]

- **Access** public

*void función Estado\_tableta\_model::getId() [line 62]*

- **Access** public

*boolean|string función Estado\_tableta\_model::getMsgError([\$type = 'usr']) [line 91]*

**Parámetros de la función:**

- *string \$type* usr si se quiere devolver el mensaje de error a mostrar en la vista, log obtiene el mensaje de error con mas detalles para depuración, valor por defecto usr

**Devuelve el mensaje de error, en caso de existir un error despues de ejecutar un metodo, de lo contrario false**

- **Access** public

*void función Estado\_tableta\_model::setDescripcion(\$descripcion) [line 76]*

**Parámetros de la función:**

- **\$descripcion**

- **Access** public

*void función Estado\_tableta\_model::setId(\$id) [line 71]*

**Parámetros de la función:**

- **\$id**

- **Access** public

## Clase Notificacion\_model

[line [10](#)]

### Modelo Usuario

- **Package** TES
- **Sub-Package** Modelo
- **Author** Rogelio

Constructor `void` función Notificacion\_model::\_\_construct() [line [74](#)]

- **Access** public

`void|boolean` función Notificacion\_model::addFilter(\$columna, \$condicion, \$valor) [line [824](#)]

#### Parámetros de la función:

- **string \$columna** Puede ser cualquier campo del objeto (id, id\_usuario, fecha\_hora, parametros, id\_controlador\_accion)
- **string \$condicion** Establece la condicion a evaluar, entre los valores permitidos estan: =, !=, >=, <=, like
- **string \$valor** Valor contra el cual se realizará la evaluación del campo

### Agrega una nueva regla de filtrado al arreglo de filtros

- **Access** public

*boolean* función Notificacion\_model::delete() [[line 204](#)]

### **Elimina de la base de datos la notificación (id en propiedades)**

- **Access** public

*void|array* función Notificacion\_model::getAll([*\$keywords* = "], [*\$offset* = null], [*\$row\_count* = null]) [[line 170](#)]

#### **Parámetros de la función:**

- *boolean|string \$keywords* false no hay texto a buscar|string con texto a buscar
- *int \$offset* Establece el desplazamiento del primer registro a devolver, si se define solo el valor de offset el valor especifica el número de registros a retornar desde el comienzo del conjunto de resultados.
- *int \$row\_count* Establece la cantidad de registros a devolver

**Obtiene todas las notificaciones existentes, se puede filtrar por: texto a buscar si se desea**

- **Access** public

*void|object* función Notificacion\_model::getById(*\$id*) [[line 209](#)]

#### **Parámetros de la función:**

- *int \$id* id de notificación

**Obtiene la notificación solicitada**

- **Access** public

*void* función Notificacion\_model::getContenido() [[line 105](#)]

- **Access** public

*void* función Notificacion\_model::getFechaFin() [[line 125](#)]

- **Access** public

*void* función Notificacion\_model::getFechainicio() [[line 115](#)]

- **Access** public

*void* función Notificacion\_model::getId() [[line 86](#)]

- **Access** public

*void* función Notificacion\_model::getIdsTabletas() [[line 135](#)]

- **Access** public

*boolean* función Notificacion\_model::getMsgError([*\$value* = 'usr']) [[line 152](#)]

**Parámetros de la función:**

- *string* **\$value** tipo de error a visualizar: usr o log, default: usr

**Asigna el mensaje de error a visualizar: para usuario final (usr) o para bitácora (log)**

- **Access** public

*int función Notificacion\_model::getNumRows([\$keywords = ""]) [line 229]*

**Parámetros de la función:**

- *boolean|string \$keywords* false no hay texto a buscar|string con texto a buscar

## Obtiene el numero total de notificaciones

- **Access** public

*void función Notificacion\_model::getTitulo() [line 95]*

- **Access** public

*boolean función Notificacion\_model::insert() [line 255]*

**Inserta en la base de datos los datos de la notificación (datos en propiedades)**

- **Access** public

*void función Notificacion\_model::setContenido(\$contenido) [line 110]*

**Parámetros de la función:**

- **\$contenido**

- **Access** public

*void función Notificacion\_model::setFechaFin(\$fecha\_fin) [line 130]*

**Parámetros de la función:**

- **\$fecha\_fin**

- **Access** public

*void función Notificacion\_model::setFechalInicio(\$fecha\_inicio) [line 120]*

**Parámetros de la función:**

- **\$fecha\_inicio**

- **Access** public

*void función Notificacion\_model::setId(\$value) [line 91]*

**Parámetros de la función:**

- **\$value**

- **Access** public

*void* función Notificacion\_model::setIdsTabletas(\$id\_arr\_asu) [line [40](#)]

**Parámetros de la función:**

- **\$id\_arr\_asu**

- **Access** public

*void* función Notificacion\_model::setTitle(\$titulo) [line [100](#)]

**Parámetros de la función:**

- **\$titulo**

- **Access** public

*boolean* función Notificacion\_model::update() [line [279](#)]

**Actualiza en la base de datos los datos de la notificación (datos en propiedades)**

- **Access** public

## Clase Reporteador\_model [line [10](#)]

**Modelo Reporteador**

- **Package** TES
- **Sub-Package** Modelo
- **Author** Rogelio

Constructor `void` función `Reporteador_model::__construct()` [[line 27](#)]

- **Access** public

`void` función `Reporteador_model::getCensoNominal($nivel, $id, &$th)` [[line 319](#)]

**Parámetros de la función:**

- **\$nivel**
- **\$id**
- **&\$th**

- **Access** public

`void` función `Reporteador_model::getCoberturaBiologicoListado($nivel, $id, $fecha, [$fechaFin = null])` [[line 59](#)]

**Parámetros de la función:**

- **int \$nivel** Nivel del elemento del arbol ASU
- **int \$id** Identificador del elemento ASU
- **date \$fecha** Fecha de corte de elemento
- **\$fechaFin**

### Obtiene el reporte de cobertura por tipo de biológico

- **Access** public

*void función Reporteador\_model::getConcentradoActividades(\$nivel, \$id, \$fecha) [line 289]*

**Parámetros de la función:**

- **\$nivel**
- **\$id**
- **\$fecha**

- **Access** public

*void función Reporteador\_model::getEsquemasIncompletos(\$nivel, \$id, &\$th) [line 414]*

**Parámetros de la función:**

- **\$nivel**
- **\$id**
- **&\$th**

- **Access** public

*void función Reporteador\_model::getGrupoVacunas() [line 598]*

*boolean función Reporteador\_model::getMsgError([\$value = 'usr']) [line 43]*

**Parámetros de la función:**

- **string \$value** tipo de error a visualizar: usr o log, default: usr

**Asigna el mensaje de error a visualizar: para usuario final (usr) o para bitácora (log)**

- **Access** public

*void función Reporteador\_model::getSeguimientoRV1RV5(\$nivel, \$id, \$fecha) [line 304]*

**Parámetros de la función:**

- **\$nivel**
- **\$id**
- **\$fecha**

- **Access** public

*void función Reporteador\_model::getVacunas() [line 547]*

*void función Reporteador\_model::getVacunasByGrupo(\$grupo) [line 573]*

**Parámetros de la función:**

- **\$grupo**

*void función Reporteador\_model::object\_to\_array(\$data, \$campo1) [line 532]*

**Parámetros de la función:**

- **\$data**
- **\$campo1**

## Clase Reporte\_censo\_nominal [line 11]

**Modelo Reporte\_censo\_nominal**

- **Package** TES
- **Sub-Package** Modelo
- **Author** Rogelio

#### **Reporte\_censo\_nominal::\$apellido\_materno**

*varchar = [line 21]*

- **Access** public

#### **Reporte\_censo\_nominal::\$apellido\_paterno**

*varchar = [line 16]*

- **Access** public

#### **Reporte\_censo\_nominal::\$curp**

*varchar = [line 36]*

- **Access** public

#### **Reporte\_censo\_nominal::\$domicilio**

*varchar = [line 31]*

- **Access** public

#### **Reporte\_censo\_nominal::\$fecha\_nacimiento**

*varchar = [line 41]*

- **Access** public

#### **Reporte\_censo\_nominal::\$nombre**

*varchar = [line 26]*

- **Access** public

#### **Reporte\_censo\_nominal::\$parto\_multiple**

*varchar = [line 46]*

- **Access** public

#### **Reporte\_censo\_nominal::\$sexo**

*varchar = [line 51]*

- **Access** public

#### **Reporte\_censo\_nominal::\$vacunas**

*array = [line 56]*

- **Access** public

Constructor *void* función Reporte\_censo\_nominal::\_\_construct() *[line 58]*

- **Access** public

# Clase Reporte\_sincronizacion\_model

[line 10]

## Modelo Usuario

- **Package** TES
- **Sub-Package** Modelo
- **Author** Eliecer

*num\_rows()* función Reporte\_sincronizacion\_model::getCount(\$tabla, [\$sentencia = ""], \$sentencia) [line 55]

### Parámetros de la función:

- **string \$tabla** nombre de una tabla en la base de datos
- **string \$sentencia** consulta sql
- **\$sentencia**

### obtiene el numero de registros de una tabla o consulta

- **Access** public

*result()* función Reporte\_sincronizacion\_model::getListado(\$sql) [line 81]

### Parámetros de la función:

- **string \$sql** consulta slq a ejecutar

### Obtiene el resultado de una consulta

- **Access** public

*void* función Reporte\_sincronizacion\_model::getMsgError([*\$value* = 'usr']) [*line 88*]

**Parámetros de la función:**

- **\$value**

- **Access** public

*result()* función Reporte\_sincronizacion\_model::get\_version() [*line 71*]

**obtiene la ultima version de la apk de las tabletas**

- **Access** public

## Clase Semana\_nacional\_model [*line 11*]

**Modelo Tableta**

- **Package** TES
- **Sub-Package** Modelo
- **Author** Pascual

Constructor `void` función `Semana_nacional_model::__construct()` [line [61](#)]

- **Access** public

`int` función `Semana_nacional_model::delete([\$id = null])` [line [203](#)]

**Parámetros de la función:**

- `int $id` Si no se establece el valor de ID, se toma el valor del objeto actual

## Elimina el registro actual de la base de datos

- **Access** public

`array` función `Semana_nacional_model::getAll([\$offset = null], [\$row_count = null])` [line [281](#)]

**Parámetros de la función:**

- `int $offset` Establece el desplazamiento del primer registro a devolver, si se define solo el valor de offset el valor especifica el número de registros a retornar desde el comienzo del conjunto de resultados.
- `int $row_count` Establece la cantidad de registros a devolver

## Obtiene todos los registros de la tabla

- **Access** public

`object|boolean` función `Semana_nacional_model::getById($id)` [line [243](#)]

**Parámetros de la función:**

- *int \$id* Si no se establece el valor de ID, se toma el valor del objeto actual

## Obtiene los datos del registro que tiene el ID especificado

- **Access** public

*void función Semana\_nacional\_model::getDescripcion() [line 79]*

- **Access** public

*void función Semana\_nacional\_model::getFecha\_fin() [line 89]*

- **Access** public

*void función Semana\_nacional\_model::getFecha\_inicio() [line 84]*

- **Access** public

*void función Semana\_nacional\_model::getId() [line 74]*

- **Access** public

*boolean/string función Semana\_nacional\_model::getMsgError([\$type = 'usr']) [line 120]*

### Parámetros de la función:

- *string \$type* usr si se quiere devolver el mensaje de error a mostrar en la vista, log obtiene

el mensaje de error con mas detalles para depuración, valor por defecto usr

**Devuelve el mensaje de error, en caso de existir un error despues de ejecutar un metodo, de lo contrario false**

- **Access** public

*int* función Semana\_nacional\_model::getNumRows() [[line 816](#)]

**Obtiene el numero total de registros en la tabla**

- **Access** public

*boolean* función Semana\_nacional\_model::insert() [[line 140](#)]

**Inserta en la base de datos, la informacion contenida en el objeto**

- **Access** public

*void* función Semana\_nacional\_model::setDescripcion(\$descripcion) [[line 94](#)]

**Parámetros de la función:**

- **\$descripcion**

- **Access** public

*void función Semana\_nacional\_model::setFecha\_fin(\$fecha\_fin) [line 104]*

**Parámetros de la función:**

- **\$fecha\_fin**

- **Access** public

*void función Semana\_nacional\_model::setFecha\_inicio(\$fecha\_inicio) [line 99]*

**Parámetros de la función:**

- **\$fecha\_inicio**

- **Access** public

*boolean función Semana\_nacional\_model::update([\$id = null]) [line 171]*

**Parámetros de la función:**

- **int \$id** Si no se establece el valor de ID, se toma el valor del objeto actual

## **Actualiza los datos del objeto actual**

- **Access** public

# Clase Tableta\_model

[line 11]

## Modelo Tableta

- **Package** TES
- **Sub-Package** Modelo
- **Author** Pascual

Constructor `void` función Tableta\_model::\_\_construct() [line 90]

- **Access** public

`int` función Tableta\_model::delete([`$id = null`]) [line 292]

**Parámetros de la función:**

- `int $id` Si no se establece el valor de ID, se toma el valor del objeto actual

## Elimina el registro actual de la base de datos

- **Access** public

`array` función Tableta\_model::getAll([`$offset = null`], [`$row_count = null`], `$filtro`) [line 419]

**Parámetros de la función:**

- `int $offset` Establece el desplazamiento del primer registro a devolver, si se define solo el valor de offset el valor especifica el número de registros a retornar desde el comienzo del conjunto de resultados.
- `int $row_count` Establece la cantidad de registros a devolver
- `$filtro`

## Obtiene todos los registros de la tabla

- **Access** public

*object|boolean* función Tableta\_model::getById(\$id) [line 832]

**Parámetros de la función:**

- *int \$id* Si no se establece el valor de ID, se toma el valor del objeto actual

**Obtiene los datos del registro que tiene el ID especificado**

- **Access** public

*object|boolean* función Tableta\_model:: getByMac(\$mac) [line 874]

**Parámetros de la función:**

- *int \$mac* Direccion MAC

**Obtiene los datos del registro la MAC especificada**

- **Access** public

*void* función Tableta\_model::getId() [line 103]

- **Access** public

*void función Tableta\_model::getIdVersion() [line112]*

- **Access** public

*void función Tableta\_model::getId\_asu\_um() [line132]*

- **Access** public

*void función Tableta\_model::getId\_tes\_estado\_tableta() [line124]*

- **Access** public

*void función Tableta\_model::getId\_tipo\_censo() [line128]*

- **Access** public

*void función Tableta\_model::getMac() [line108]*

- **Access** public

*boolean/string función Tableta\_model::getMsgError([\$type = 'usr']) [line188]*

**Parámetros de la función:**

- *string \$type* usr si se quiere devolver el mensaje de error a mostrar en la vista, log obtiene el mensaje de error con mas detalles para depuración, valor por defecto usr

**Devuelve el mensaje de error, en caso de existir un error despues de ejecutar un metodo, de lo contrario false**

- **Access** public

*int función Tableta\_model::getNumRows() [line 518]*

**Obtiene el numero total de registros en la tabla en caso de existir filtros, estos son aplicados a la consulta**

- **Access** public

*void función Tableta\_model::getPeriodo\_esq\_inc() [line 136]*

- **Access** public

*void función Tableta\_model::getUltima\_actualizacion() [line 116]*

- **Access** public

*void función Tableta\_model::getUsuarios\_asignados() [line 120]*

- **Access** public

*boolean función Tableta\_model::insert() [line 208]*

**Inserta en la base de datos, la informacion contenida en el objeto**

- **Access** public

void función Tableta\_model::setId(\$id) [line[140](#)]

**Parámetros de la función:**

- **\$id**

- **Access** public

void función Tableta\_model::setIdVersion(\$id\_version) [line[149](#)]

**Parámetros de la función:**

- **\$id\_version**

- **Access** public

void función Tableta\_model::setId\_asu\_um(\$id\_asu\_um) [line[169](#)]

**Parámetros de la función:**

- **\$id\_asu\_um**

- **Access** public

void función Tableta\_model::setId\_tes\_estado\_tableta(\$id\_tes\_estado\_tableta) [line[161](#)]

**Parámetros de la función:**

- **\$id\_tes\_estado\_tableta**

- **Access** public

*void función Tableta\_model::setId\_tipo\_censo(\$id\_tipo\_censo) [line 165]*

**Parámetros de la función:**

- **\$id\_tipo\_censo**

- **Access** public

*void función Tableta\_model::setMac(\$mac) [line 145]*

**Parámetros de la función:**

- **\$mac**

- **Access** public

*void función Tableta\_model::setPeriodo\_esq\_inc(\$periodo\_esq\_inc) [line 173]*

**Parámetros de la función:**

- **\$periodo\_esq\_inc**

- **Access** public

*void función Tableta\_model::setUltima\_actualizacion(\$ultima\_actualizacion) [line 153]*

**Parámetros de la función:**

- **\$ultima\_actualizacion**

- **Access** public

*void función Tableta\_model::setUsuarios\_asignados(\$usuarios\_asignados) [line 157]*

**Parámetros de la función:**

- **\$usuarios\_asignados**

- **Access** public

*boolean función Tableta\_model::update([\$id = null]) [line 239]*

**Parámetros de la función:**

- **int \$id** Si no se establece el valor de ID, se toma el valor del objeto actual

### **Actualiza los datos del objeto actual**

- **Access** public

## Clase Tipo\_censo\_model [line 11]

### Modelo Tipo\_censo

- **Package** TES
- **Sub-Package** Modelo
- **Author** Pascual

Constructor `void` función `Tipo_censo_model::__construct()` [line 49]

- **Access** public

`array` función `Tipo_censo_model::getAll()` [line 141]

**Obtiene todos los registros de la tabla**

- **Access** public

`object|boolean` función `Tipo_censo_model::getById($id)` [line 113]

**Parámetros de la función:**

- `int $id` Si no se establece el valor de ID, se toma el valor del objeto actual

**Obtiene los datos del registro que tiene el ID especificado**

- **Access** public

*void función Tipo\_censo\_model::getDescripcion() [line 67]*

- **Access** public

*void función Tipo\_censo\_model::getId() [line 62]*

- **Access** public

*boolean|string función Tipo\_censo\_model::getMsgError([\$type = 'usr']) [line 92]*

**Parámetros de la función:**

- *string \$type* usr si se quiere devolver el mensaje de error a mostrar en la vista, log obtiene el mensaje de error con mas detalles para depuración, valor por defecto usr

**Devuelve el mensaje de error, en caso de existir un error despues de ejecutar un metodo, de lo contrario false**

- **Access** public

*void función Tipo\_censo\_model::setDescripcion(\$descripcion) [line 77]*

**Parámetros de la función:**

- **\$descripcion**

- **Access** public

*void función Tipo\_censo\_model::setId(\$id) [line 72]*

**Parámetros de la función:**

- **\$id**
  - **Access** public

## Clase Usuario\_tableta\_model [line 11]

**Modelo Estado\_tableta**

- **Package** TES
- **Sub-Package** Modelo
- **Author** Pascual

Constructor *void función Usuario\_tableta\_model::\_\_construct() [line 19]*

- **Access** public

*boolean función Usuario\_tableta\_model::delete(\$id\_usuario, [\$id\_tableta = null]) [line 161]*

**Parámetros de la función:**

- *int \$id\_usuario*
- *int \$id\_tableta* Si no se proporciona el parametro, se toma el id del objeto actual

## Elimina la relacion entre usuario y tableta

- **Access** public

*boolean|string* función Usuario\_tableta\_model::getMsgError([*\$type* = 'usr']) [[line 91](#)]

**Parámetros de la función:**

- *string \$type* usr si se quiere devolver el mensaje de error a mostrar en la vista, log obtiene el mensaje de error con mas detalles para depuración, valor por defecto usr

**Devuelve el mensaje de error, en caso de existir un error despues de ejecutar un metodo, de lo contrario false**

- **Access** public

*void* función Usuario\_tableta\_model::getTableta() [[line 62](#)]

- **Access** public

*object|boolean* función Usuario\_tableta\_model::getTabletasByUsuario([*\$id\_usuario* = null]) [[line 136](#)]

**Parámetros de la función:**

- *int \$id\_usuario* Si no se establece el valor de ID, se toma el valor del objeto actual

**Obtiene el id de todas las tabletas relacionadas con el usuario especificado**

- **Access** public

*void función Usuario\_tableta\_model::getUsuario() [line 67]*

- **Access** public

*object|boolean función Usuario\_tableta\_model::getUsuariosByTableta([\$id\_tes\_tableta = null]) [line 12]*

**Parámetros de la función:**

- **int \$id\_tes\_tableta** Si no se establece el valor de ID, se toma el valor del objeto actual

**Obtiene el id de todos los usuarios asignados a la tableta especificada**

- **Access** public

*boolean función Usuario\_tableta\_model::insert([\$id\_usuario = null], [\$id\_tableta = null]) [line 204]*

**Parámetros de la función:**

- **\$id\_usuario**
- **\$id\_tableta**

**Inserta en la base de datos, la informacion contenida en el objeto**

- **Access** public

*void función Usuario\_tableta\_model::setTableta(\$id\_tes\_tableta) [line 21]*

**Parámetros de la función:**

- **\$id\_tes\_tableta**

- **Access** public

*void función Usuario\_tableta\_model::setUsuario(\$id\_usuario) [line 76]*

**Parámetros de la función:**

- **\$id\_usuario**

- **Access** public

# Apéndices

# Apéndice A - Arbol de clases

## Paquete SIIGS

### Accion

- CI\_Controller
  - [Accion](#)

### Accion\_model

- CI\_Model
  - [Accion\\_model](#)

### Ageb\_model

- CI\_Model
  - [Ageb\\_model](#)

### ArbolSegmentacion\_model

- CI\_Model
  - [ArbolSegmentacion\\_model](#)

## Ayuda

- CI\_Controller
  - [Ayuda](#)

## Bitacora

- CI\_Controller
  - [Bitacora](#)

## Bitacora\_model

- CI\_Model
  - [Bitacora\\_model](#)

## Catalogo

- CI\_Controller
  - [Catalogo](#)

## CatalogoCsv

- CI\_Controller
  - [CatalogoCsv](#)

## CatalogoCsv\_model

- CI\_Model
  - [CatalogoCsv\\_model](#)

## Catalogo\_model

- CI\_Model
  - [Catalogo\\_model](#)

## Catalogo\_x\_raiz

- CI\_Controller
  - [Catalogo\\_x\\_raiz](#)

## Catalogo\_x\_raiz\_model

- CI\_Model
  - [Catalogo\\_x\\_raiz\\_model](#)

## Cie10

- CI\_Controller
  - [Cie10](#)

## Cie10\_model

- CI\_Model
  - [Cie10\\_model](#)

## Controlador

- CI\_Controller
  - [Controlador](#)

## ControladorAccion\_model

- CI\_Model
  - [ControladorAccion\\_model](#)

## Controlador\_model

- CI\_Model
  - [Controlador\\_model](#)

## Entorno

- CI\_Controller
  - [Entorno](#)

## Entorno\_model

- CI\_Model
  - [Entorno\\_model](#)

## Errorlog

- CI\_Controller
  - [Errorlog](#)

## Errorlog\_model

- CI\_Model
  - [Errorlog\\_model](#)

## Georeferencia\_model

- CI\_Model
  - [Georeferencia\\_model](#)

## Grupo

- CI\_Controller
  - [Grupo](#)

## Grupo\_model

- CI\_Model
  - [Grupo\\_model](#)

## Hemoglobina\_model

- CI\_Model
  - [Hemoglobina\\_model](#)

## Menu

- CI\_Controller
  - [Menu](#)

## Menu\_model

- CI\_Model
  - [Menu\\_model](#)

## Permiso

- CI\_Controller
  - [Permiso](#)

## Permiso\_model

- CI\_Model
  - [Permiso\\_model](#)

## Poblacion\_model

- CI\_Model
  - [Poblacion\\_model](#)

## Raiz

- CI\_Controller
  - [Raiz](#)

## Raiz\_model

- CI\_Model
  - [Raiz\\_model](#)

## ReglaVacuna

- CI\_Controller
  - [ReglaVacuna](#)

## ReglaVacuna\_model

- CI\_Model
  - [ReglaVacuna\\_model](#)

## Usuario

- CI\_Controller
  - [Usuario](#)

## Usuario\_model

- CI\_Model
  - [Usuario\\_model](#)

Paquete default

## Index

- CI\_Controller
  - [Index](#)

## Index

- CI\_Controller
  - [Index](#)

Paquete TES

## Enrolamiento

- CI\_Controller
  - [Enrolamiento](#)

## Enrolamiento\_model

- CI\_Model
  - [Enrolamiento\\_model](#)

## Estado\_tableta\_model

- CI\_Model
  - [Estado\\_tableta\\_model](#)

## Notificacion

- CI\_Controller
  - [Notificacion](#)

## Notificacion\_model

- CI\_Model
  - [Notificacion\\_model](#)

## Reporteador

- CI\_Controller
  - [Reporteador](#)

## Reporteador\_model

- CI\_Model
  - [Reporteador\\_model](#)

## Reporte\_censo\_nominal

- CI\_Model
  - [Reporte\\_censo\\_nominal](#)

## Reporte\_sincronizacion

- CI\_Controller
  - [Reporte\\_sincronizacion](#)

## Reporte\_sincronizacion\_model

- CI\_Model
  - [Reporte\\_sincronizacion\\_model](#)

## Semana\_nacional

- CI\_Controller
  - [Semana\\_nacional](#)

## Semana\_nacional\_model

- CI\_Model
  - [Semana\\_nacional\\_model](#)

## Servicios

- CI\_Controller
  - [Servicios](#)

## Tableta

- CI\_Controller
  - [Tableta](#)

## Tableta\_model

- CI\_Model
  - [Tableta\\_model](#)

## Tipo\_censo\_model

- CI\_Model
  - [Tipo\\_censo\\_model](#)

## Usuario\_tableta

- CI\_Controller
  - [Usuario\\_tableta](#)

## Usuario\_tableta\_model

- CI\_Model
  - [Usuario\\_tableta\\_model](#)

## Paquete Libreria

## Graph

- CI\_Controller
  - [Graph](#)

## Menubuilder

- [Menubuilder](#)

## Obtenercurp

- CI\_Controller
  - [Obtenercurp](#)

## Tree

- CI\_Controller
  - [Tree](#)

## Paquete CodeIgniter

### CI\_Form\_validation

- [CI\\_Form\\_validation](#)

### CI\_Pagination

- [CI\\_Pagination](#)

### CI\_Template

- [CI\\_Template](#)

## Paquete Session

# Session

- [Session](#)

# Apéndice C - Código fuente

# Paquete TES

# Archivo fuente para index.php

*La documentación para este archivo está disponible en [index.php](#)*

```
1  <?php
2
3  class Index extends CI_Controller {
4
5      public function __construct()
6      {
7          parent::__construct();
8      }
9
10     public function index()
11     {
12         $this->template->write_view('content','home');
13         $this->template->render();
14     }
15 }
```

# Archivo fuente para index.php

*La documentación para este archivo está disponible en [index.php](#)*

```
1      <?php
2
3  class Index extends CI_Controller {
4
5      public function __construct()
6      {
7          parent::__construct();
8      }
9
10     public function index()
11     {
12         $this->template->render();
13     }
14 }
```

Paquete TES

# Archivo fuente para Form\_validation.php

La documentación para este archivo está disponible en [Form\\_validation.php](#)

```
1  <?php      if ( ! defined('BASEPATH')) exit('No direct script access allowed');
2  /**
3   * CodeIgniter
4   *
5   * An open source application development framework for PHP 5.1.6 or newer
6   *
7   * @package      CodeIgniter
8   * @author       ExpressionEngine Dev Team
9   * @copyright    Copyright (c) 2008 - 2011, EllisLab, Inc.
10  * @license      http://codeigniter.com/user_guide/license.html
11  * @link         http://codeigniter.com
12  * @since        Version 1.0
13  * @filesource
14  */
15
16 // -----
17
18 /**
19  * Form Validation Class
20  *
21  * @package      CodeIgniter
22  * @subpackage   Libraries
23  * @category    Validation
24  * @author       ExpressionEngine Dev Team
25  * @link         http://codeigniter.com/user_guide/libraries/form_validation.html
26  */
27 class CI_Form_validation {
28
29     protected $CI;
30     protected $field_data      = array();
31     protected $config_rules    = array();
32     protected $error_array     = array();
33     protected $error_messages  = array();
34     protected $error_prefix    = '<div class="error">';
35     protected $error_suffix    = '</div>';
36     protected $error_string    = '';
37     protected $safe_form_data  = FALSE;
38
39     /**
40      * Constructor
41     */
42     public function __construct($rules = array())
43     {
44         $this-> CI =& get_instance();
45
46         // Validation rules can be stored in a config file.
47         $this-> config_rules = $rules;
48
49         // Automatically load the form helper
50         $this-> CI-> load-> helper('form');
51
52         // Set the character encoding in MB.
53         if (function_exists('mb_internal_encoding'))
54         {
55             mb_internal_encoding($this-> CI-> config-> item('charset'));
56         }
57
58         log_message('debug', "Form Validation Class Initialized");
59     }
60
61 // -----
62
63 /**
64  * Set Rules
65  *
66  * This function takes an array of field names and validation
67  * rules as input, validates the info, and stores it
68  */
```

```

68
69     * @access public
70     * @param mixed
71     * @param string
72     * @return void
73     */
74     public function set_rules($field, $label = '', $rules = '')
75     {
76         // No reason to set rules if we have no POST data
77         if ($count($_POST) == 0)
78         {
79             return $this;
80         }
81
82         // If an array was passed via the first parameter instead of individual string
83         // values we cycle through it and recursively call this function.
84         if (is_array($field))
85         {
86             foreach ($field as $row)
87             {
88                 // Houston, we have a problem...
89                 if (!isset($row['field']) OR !isset($row['rules']))
90                 {
91                     continue;
92                 }
93
94                 // If the field label wasn't passed we use the field name
95                 $label = (!isset($row['label'])) ? $row['field'] : $row['label'];
96
97                 // Here we go!
98                 $this-> set_rules($row['field'], $label, $row['rules']);
99             }
100            return $this;
101        }
102
103        // No fields? Nothing to do...
104        if (!is_string($field) OR !is_string($rules) OR $field == '')
105        {
106            return $this;
107        }
108
109        // If the field label wasn't passed we use the field name
110        $label = ($label == '') ? $field : $label;
111
112        // Is the field name an array? We test for the existence of a bracket "[" in
113        // the field name to determine this. If it is an array, we break it apart
114        // into its components so that we can fetch the corresponding POST data later
115        if (strpos($field, '[') !== FALSE AND preg_match_all('/\[(.*?)\]/', $field, $matches))
116        {
117            // Note: Due to a bug in current() that affects some versions
118            // of PHP we can not pass function call directly into it
119            $x = explode('[', $field);
120            $indexes[] = current($x);
121
122            for ($i = 0; $i < count($matches['0']); $i++)
123            {
124                if ($matches['1'][$i] != '')
125                {
126                    $indexes[] = $matches['1'][$i];
127                }
128            }
129
130            $is_array = TRUE;
131        }
132        else
133        {
134            $indexes = array();
135            $is_array = FALSE;
136        }
137
138        // Build our master array
139        $this-> _field_data[$field] = array(
140            'field'          => $field,
141            'label'          => $label,
142            'rules'          => $rules,
143            'is_array'       => $is_array,
144            'keys'           => $indexes,
145            'postdata'       => NULL,
146            'error'          => ''
147        );

```

```

148         return $this;
149     }
150
151
152 // -----
153 /**
154 * Set Error Message
155 *
156 * Lets users set their own error messages on the fly. Note: The key
157 * name has to match the function name that it corresponds to.
158 *
159 * @access public
160 * @param string
161 * @param string
162 * @return string
163 */
164 public function set_message($lang, $val = '')
165 {
166     if ( ! is_array($lang) )
167     {
168         $lang = array($lang => $val);
169     }
170
171     $this-> _error_messages = array_merge($this-> _error_messages, $lang);
172
173     return $this;
174 }
175
176
177 // -----
178 /**
179 * Set The Error Delimiter
180 *
181 * Permits a prefix/suffix to be added to each error message
182 *
183 * @access public
184 * @param string
185 * @param string
186 * @return void
187 */
188 public function set_error_delimiters($prefix = '<p>', $suffix = '</p>')
189 {
190     $this-> _error_prefix = $prefix;
191     $this-> _error_suffix = $suffix;
192
193     return $this;
194 }
195
196
197 // -----
198 /**
199 * Get Error Message
200 *
201 * Gets the error message associated with a particular field
202 *
203 * @access public
204 * @param string the field name
205 * @return void
206 */
207 public function error($field = '', $prefix = '', $suffix = '')
208 {
209     if ( ! isset($this-> _field_data[$field]['error']) OR $this-
210 > _field_data[$field]['error'] == '' )
211     {
212         return '';
213     }
214
215     if ($prefix == '')
216     {
217         $prefix = $this-> _error_prefix;
218     }
219
220     if ($suffix == '')
221     {
222         $suffix = $this-> _error_suffix;
223     }
224
225     return $prefix.$this-> _field_data[$field]['error'].$suffix;
226 }

```

```

227
228 // -----
229
230 /**
231 * Error String
232 *
233 * Returns the error messages as a string, wrapped in the error delimiters
234 *
235 * @access public
236 * @param string
237 * @param string
238 * @return str
239 */
240 public function error_string($prefix = '', $suffix = '')
241 {
242     // No errors, validation passes!
243     if (count($this-> _error_array) === 0)
244     {
245         return '';
246     }
247
248     if ($prefix == '')
249     {
250         $prefix = $this-> _error_prefix;
251     }
252
253     if ($suffix == '')
254     {
255         $suffix = $this-> _error_suffix;
256     }
257
258     // Generate the error string
259     $str = '';
260     foreach ($this-> _error_array as $val)
261     {
262         if ($val != '')
263         {
264             $str .= $prefix.$val.$suffix."\n";
265         }
266     }
267
268     return $str;
269 }
270 // -----
271
272 /**
273 * Run the Validator
274 *
275 * This function does all the work.
276 *
277 * @access public
278 * @return bool
279 */
280 public function run($group = '')
281 {
282     // Do we even have any data to process? Mm?
283     if (count($_POST) == 0)
284     {
285         return FALSE;
286     }
287
288     // Does the _field_data array containing the validation rules exist?
289     // If not, we look to see if they were assigned via a config file
290     if (count($this-> _field_data) == 0)
291     {
292         // No validation rules? We're done...
293         if (count($this-> _config_rules) == 0)
294         {
295             return FALSE;
296         }
297
298         // Is there a validation rule for the particular URI being accessed?
299         $uri = ($group == '') ? trim($this-> CI-> uri-> ruri_string(), '/') : $group;
300
301         if ($uri != '' AND isset($this-> _config_rules[$uri]))
302         {
303             $this-> set_rules($this-> _config_rules[$uri]);
304         }
305     else

```

```

307     {
308         $this-> set_rules($this-> config_rules);
309     }
310
311     // We're we able to set the rules correctly?
312     if (count($this-> field_data) == 0)
313     {
314         log_message('debug', "Unable to find validation rules");
315         return FALSE;
316     }
317 }
318
319     // Load the language file containing error messages
320     $this-> CI-> lang-> load('form_validation');
321
322     // Cycle through the rules for each field, match the
323     // corresponding $_POST item and test for errors
324     foreach ($this-> field_data as $field => $row)
325     {
326         // Fetch the data from the corresponding $_POST array and cache it in the
327         // _field_data array.
328         // Depending on whether the field name is an array or a string will determine where
329         // we get it from.
330
331         if ($row['is_array'] == TRUE)
332         {
333             $this-> field_data[$field]['postdata'] = $this-> reduce_array($_POST,
334 $row['keys']);
335         }
336         else
337         {
338             if (isset($_POST[$field]) AND $_POST[$field] != "")
339             {
340                 $this-> field_data[$field]['postdata'] = $_POST[$field];
341             }
342         }
343
344         $this-> execute($row, explode('|', $row['rules']), $this-
345 > field_data[$field]['postdata']);
346     }
347
348     // Did we end up with any errors?
349     $total_errors = count($this-> error_array);
350
351     if ($total_errors > 0)
352     {
353         $this-> safe_form_data = TRUE;
354     }
355
356     // Now we need to re-set the POST data with the new, processed data
357     $this-> reset_post_array();
358
359     // No errors, validation passes!
360     if ($total_errors == 0)
361     {
362         return TRUE;
363     }
364
365     // -----
366
367 /**
368 * Traverse a multidimensional $_POST array index until the data is found
369 *
370 * @access private
371 * @param array
372 * @param array
373 * @param integer
374 * @return mixed
375 */
376 protected function _reduce_array($array, $keys, $i = 0)
377 {
378     if (is_array($array))
379     {
380         if (isset($keys[$i]))
381         {
382             if (isset($array[$keys[$i]]))

```

```

383
384     {
385         $array = $this-> _reduce_array($array[$keys[$i]], $keys, ($i+1));
386     }
387     else
388     {
389         return NULL;
390     }
391     else
392     {
393         return $array;
394     }
395 }
396
397 return $array;
398 }
399
400 // -----
401 /**
402 * Re-populate the _POST array with our finalized and processed data
403 *
404 * @access private
405 * @return null
406 */
407 protected function _reset_post_array()
408 {
409     foreach ($this-> _field_data as $field => $row)
410     {
411         if ( ! is_null($row['postdata']))
412         {
413             if ($row['is_array'] == FALSE)
414             {
415                 if (isset($_POST[$row['field']])))
416                 {
417                     $_POST[$row['field']] = $this-> prep_for_form($row['postdata']);
418                 }
419             }
420             else
421             {
422                 // start with a reference
423                 $post_ref =& $_POST;
424
425                 // before we assign values, make a reference to the right POST key
426                 if (count($row['keys']) == 1)
427                 {
428                     $post_ref =& $post_ref[current($row['keys'])];
429                 }
430                 else
431                 {
432                     foreach ($row['keys'] as $val)
433                     {
434                         $post_ref =& $post_ref[$val];
435                     }
436                 }
437
438                 if (is_array($row['postdata']))
439                 {
440                     $array = array();
441                     foreach ($row['postdata'] as $k => $v)
442                     {
443                         $array[$k] = $this-> prep_for_form($v);
444                     }
445
446                     $post_ref = $array;
447                 }
448                 else
449                 {
450                     $post_ref = $this-> prep_for_form($row['postdata']);
451                 }
452             }
453         }
454     }
455 }
456
457
458 // -----
459 /**
460 * Executes the Validation routines
461 */

```

```

463 * @access private
464 * @param array
465 * @param array
466 * @param mixed
467 * @param integer
468 * @return mixed
469 */
470 protected function _execute($row, $rules, $postdata = NULL, $cycles = 0)
471 {
472     // If the $_POST data is an array we will run a recursive call
473     if (is_array($postdata))
474     {
475         foreach ($postdata as $key => $val)
476         {
477             $this-> _execute($row, $rules, $val, $cycles);
478             $cycles++;
479         }
480     }
481     return;
482 }
483
484 // -----
485
486 // If the field is blank, but NOT required, no further tests are necessary
487 $callback = FALSE;
488 if ( ! in_array('required', $rules) AND is_null($postdata))
489 {
490     // Before we bail out, does the rule contain a callback?
491     if (preg_match("/(callback_\w+([.*?\\])?)/", implode(' ', $rules),
492 $match))
493     {
494         $callback = TRUE;
495         $rules = (array('1' => $match[1]));
496     }
497     else
498     {
499         return;
500     }
501 }
502
503 // -----
504 // Isset Test. Typically this rule will only apply to checkboxes.
505 if (is_null($postdata) AND $callback == FALSE)
506 {
507     if (in_array('isset', $rules, TRUE) OR in_array('required', $rules))
508     {
509         // Set the message type
510         $type = (in_array('required', $rules)) ? 'required' : 'isset';
511
512         if ( ! isset($this-> _error_messages[$type]))
513         {
514             if (FALSE === ($line = $this-> CI-> lang-> line($type)))
515             {
516                 $line = 'The field was not set';
517             }
518         }
519         else
520         {
521             $line = $this-> _error_messages[$type];
522         }
523
524         // Build the error message
525         $message = sprintf($line, $this-> _translate_fieldname($row['label']));
526
527         // Save the error message
528         $this-> _field_data[$row['field']]['error'] = $message;
529
530         if ( ! isset($this-> _error_array[$row['field']]))
531         {
532             $this-> _error_array[$row['field']] = $message;
533         }
534     }
535
536     return;
537 }
538
539 // -----
540 // Cycle through each rule and run it

```



```

618
619         {
620             $result = $rule($postdata);
621
622             if ($in_array == TRUE)
623             {
624                 $this-> _field_data[$row['field']] ['postdata'][$cycles] =
625             ($is_bool($result)) ? $postdata : $result;
626             }
627             else
628             {
629                 $this-> _field_data[$row['field']] ['postdata'] =
630             ($is_bool($result)) ? $postdata : $result;
631             }
632             else
633             {
634                 log_message('debug', "Unable to find validation rule:
635             ". $rule);
636             }
637             continue;
638         }
639
640         $result = $this-> $rule($postdata, $param);
641
642         if ($in_array == TRUE)
643         {
644             $this-> _field_data[$row['field']] ['postdata'][$cycles] =
645             ($is_bool($result)) ? $postdata : $result;
646             else
647             {
648                 $this-> _field_data[$row['field']] ['postdata'] = ($is_bool($result)) ?
649             $postdata : $result;
650             }
651
652         // Did the rule test negatively? If so, grab the error.
653         if ($result === FALSE)
654         {
655             if ( ! isset($this-> _error_messages[$rule]))
656             {
657                 if (FALSE === ($line = $this-> CI-> lang-> line($rule)))
658                 {
659                     $line = 'Unable to access an error message corresponding to your field
660 name.';
661                 }
662             else
663             {
664                 $line = $this-> _error_messages[$rule];
665             }
666
667             // Is the parameter we are inserting into the error message the name
668             // of another field? If so we need to grab its "field label"
669             if (isset($this-> _field_data[$param]) AND isset($this-
670             > _field_data[$param]['label']))
671             {
672                 $param = $this-> _translate_fieldname($this-
673             > _field_data[$param]['label']);
674             }
675
676             // Build the error message
677             $message = sprintf($line, $this-> _translate_fieldname($row['label']),
678             $param);
679
680             // Save the error message
681             $this-> _field_data[$row['field']] ['error'] = $message;
682
683             if ( ! isset($this-> _error_array[$row['field']]))
684             {
685                 $this-> _error_array[$row['field']] = $message;
686             }
687
688             return;
689         }
690     }
691
692     // -----

```

```

689
690     /**
691      * Translate a field name
692      *
693      * @access    private
694      * @param     string    the field name
695      * @return    string
696      */
697     protected function _translate_fieldname($fieldname)
698     {
699         // Do we need to translate the field name?
700         // We look for the prefix lang: to determine this
701         if (substr($fieldname, 0, 5) == 'lang:')
702         {
703             // Grab the variable
704             $line = substr($fieldname, 5);
705
706             // Were we able to translate the field name? If not we use $line
707             if (FALSE === ($fieldname = $this-> CI-> lang-> line($line)))
708             {
709                 return $line;
710             }
711         }
712
713         return $fieldname;
714     }
715
716     // -----
717
718     /**
719      * Get the value from a form
720      *
721      * Permits you to repopulate a form field with the value it was submitted
722      * with, or, if that value doesn't exist, with the default
723      *
724      * @access    public
725      * @param     string    the field name
726      * @param     string
727      * @return    void
728      */
729     public function set_value($field = '', $default = '')
730     {
731         if ( ! isset($this-> _field_data[$field]))
732         {
733             return $default;
734         }
735
736         // If the data is an array output them one at a time.
737         // E.g: form_input('name[]', set_value('name[]'));
738         if (is_array($this-> _field_data[$field]['postdata']))
739         {
740             return array_shift($this-> _field_data[$field]['postdata']);
741         }
742
743         return $this-> _field_data[$field]['postdata'];
744     }
745
746     // -----
747
748     /**
749      * Set Select
750      *
751      * Enables pull-down lists to be set to the value the user
752      * selected in the event of an error
753      *
754      * @access    public
755      * @param     string
756      * @param     string
757      * @return    string
758      */
759     public function set_select($field = '', $value = '', $default = FALSE)
760     {
761         if ( ! isset($this-> _field_data[$field]) OR ! isset($this-
762 > _field_data[$field]['postdata']))
763         {
764             if ($default === TRUE AND count($this-> _field_data) === 0)
765             {
766                 return ' selected="selected" ';
767             }
768             return '';
769         }
770     }

```

```

768     }
769
770     $field = $this-> _field_data[$field]['postdata'];
771
772     if (is_array($field))
773     {
774         if ( ! in_array($value, $field))
775         {
776             return '';
777         }
778     else
779     {
780         if (( $field == '' OR $value == '') OR ($field != $value))
781         {
782             return '';
783         }
784     }
785
786     return ' selected="selected"' ;
787 }
788
789 // -----
790 /**
791 * Set Radio
792 *
793 * Enables radio buttons to be set to the value the user
794 * selected in the event of an error
795 *
796 * @access public
797 * @param string
798 * @param string
799 * @return string
800 */
801 public function set_radio($field = '', $value = '', $default = FALSE)
802 {
803     if ( ! isset($this-> _field_data[$field]) OR ! isset($this-
804 > _field_data[$field]['postdata']))
805     {
806         if ($default === TRUE AND count($this-> _field_data) === 0)
807         {
808             return ' checked="checked"' ;
809         }
810         return '';
811     }
812
813     $field = $this-> _field_data[$field]['postdata'];
814
815     if (is_array($field))
816     {
817         if ( ! in_array($value, $field))
818         {
819             return '';
820         }
821     else
822     {
823         if (( $field == '' OR $value == '') OR ($field != $value))
824         {
825             return '';
826         }
827     }
828
829     }
830
831     return ' checked="checked"' ;
832 }
833
834 // -----
835 /**
836 * Set Checkbox
837 *
838 * Enables checkboxes to be set to the value the user
839 * selected in the event of an error
840 *
841 * @access public
842 * @param string
843 * @param string
844 * @return string
845 */

```

```

847     public function set_checkbox($field = '', $value = '', $default = FALSE)
848     {
849         if ( ! isset($this-> _field_data[$field]) OR ! isset($this-
850 > _field_data[$field]['postdata']))
851         {
852             if ($default === TRUE AND count($this-> _field_data) === 0)
853             {
854                 return ' checked="checked" ';
855             }
856             return '';
857         }
858         $field = $this-> _field_data[$field]['postdata'];
859
860         if (is_array($field))
861         {
862             if ( ! in_array($value, $field))
863             {
864                 return '';
865             }
866         }
867         else
868         {
869             if ((($field == '' OR $value == '') OR ($field != $value)))
870             {
871                 return '';
872             }
873         }
874
875         return ' checked="checked" ';
876     }
877
878 // -----
879
880 /**
881 * Required
882 *
883 * @access public
884 * @param string
885 * @return bool
886 */
887 public function required($str)
888 {
889     if ( ! is_array($str))
890     {
891         return (trim($str) == '') ? FALSE : TRUE;
892     }
893     else
894     {
895         return ( ! empty($str));
896     }
897 }
898
899 // -----
900
901 /**
902 * Performs a Regular Expression match test.
903 *
904 * @access public
905 * @param string
906 * @param regex
907 * @return bool
908 */
909 public function regex_match($str, $regex)
910 {
911     if ( ! preg_match($regex, $str))
912     {
913         return FALSE;
914     }
915
916     return TRUE;
917 }
918
919 // -----
920
921 /**
922 * Match one field to another
923 *
924 * @access public
925 * @param string

```

```

926     * @param    field
927     * @return   bool
928     */
929     public function matches($str, $field)
930     {
931         if ( ! isset($_POST[$field]))
932         {
933             return FALSE;
934         }
935
936         $field = $_POST[$field];
937
938         return ($str !== $field) ? FALSE : TRUE;
939     }
940
941     // -----
942
943     /**
944      * Match one field to another
945      *
946      * @access  public
947      * @param   string
948      * @param   field
949      * @return  bool
950      */
951     public function is_unique($str, $field)
952     {
953         list($table, $field)=explode('.', $field);
954         $query = $this-> CI-> db-> limit(1)-> get_where($table, array($field =>
955 $str));
956
957         return $query-> num_rows() === 0;
958     }
959
960     // -----
961     /**
962      * Minimum Length
963      *
964      * @access  public
965      * @param   string
966      * @param   value
967      * @return  bool
968      */
969     public function min_length($str, $val)
970     {
971         if (preg_match("/[^0-9]/" , $val))
972         {
973             return FALSE;
974         }
975
976         if (function_exists('mb_strlen'))
977         {
978             return (mb_strlen($str) < $val) ? FALSE : TRUE;
979         }
980
981         return (strlen($str) < $val) ? FALSE : TRUE;
982     }
983
984     // -----
985
986     /**
987      * Max Length
988      *
989      * @access  public
990      * @param   string
991      * @param   value
992      * @return  bool
993      */
994     public function max_length($str, $val)
995     {
996         if (preg_match("/[^0-9]/" , $val))
997         {
998             return FALSE;
999         }
1000
1001         if (function_exists('mb_strlen'))
1002         {
1003             return (mb_strlen($str) > $val) ? FALSE : TRUE;
1004         }

```

```

1005     return (strlen($str) > $val) ? FALSE : TRUE;
1006 }
1007
1008 // -----
1009 /**
1010 * Exact Length
1011 *
1012 * @access public
1013 * @param string
1014 * @param value
1015 * @return bool
1016 */
1017 public function exact_length($str, $val)
1018 {
1019     if (preg_match("/[^0-9]/" , $val))
1020     {
1021         return FALSE;
1022     }
1023
1024     if (function_exists('mb_strlen'))
1025     {
1026         return (mb_strlen($str) != $val) ? FALSE : TRUE;
1027     }
1028
1029     return (strlen($str) != $val) ? FALSE : TRUE;
1030 }
1031
1032 // -----
1033 /**
1034 * Valid Email
1035 *
1036 * @access public
1037 * @param string
1038 * @return bool
1039 */
1040 public function valid_email($str)
1041 {
1042     return (! preg_match("/^([a-z0-9\+_\-]+)(\.[a-z0-9\+_\-]+)*@[a-z0-9\_-]+\.[a-zA-Z]{2,6}$/ix" , $str)) ? FALSE : TRUE;
1043 }
1044
1045 // -----
1046 /**
1047 * Valid Emails
1048 *
1049 * @access public
1050 * @param string
1051 * @return bool
1052 */
1053 public function valid_emails($str)
1054 {
1055     if (strpos($str, ',') === FALSE)
1056     {
1057         return $this-> valid_email(trim($str));
1058     }
1059
1060     foreach (explode(',', $str) as $email)
1061     {
1062         if (trim($email) != '' && $this-> valid_email(trim($email)) === FALSE)
1063         {
1064             return FALSE;
1065         }
1066     }
1067
1068     return TRUE;
1069 }
1070
1071 // -----
1072 /**
1073 * Validate IP Address
1074 *
1075 * @access public
1076 * @param string
1077 * @param string "ipv4" or "ipv6" to validate a specific ip format
1078 * @return string
1079 */
1080 public function validate_ip($ip, $format)
1081 {
1082     if ($format == 'ipv4')
1083     {
1084         if (! filter_var($ip, FILTER_VALIDATE_IP, FILTER_FLAG_IPV4))
1085         {
1086             return FALSE;
1087         }
1088
1089         return TRUE;
1090     }
1091
1092     if ($format == 'ipv6')
1093     {
1094         if (! filter_var($ip, FILTER_VALIDATE_IP, FILTER_FLAG_IPV6))
1095         {
1096             return FALSE;
1097         }
1098
1099         return TRUE;
1100     }
1101
1102     return FALSE;
1103 }

```

```

1084     */
1085     public function valid_ip($ip, $which = '')
1086     {
1087         return $this-> CI-> input-> valid_ip($ip, $which);
1088     }
1089     // -----
1090
1091 /**
1092 * Alpha
1093 *
1094 * @access public
1095 * @param string
1096 * @return bool
1097 */
1098 public function alpha($str)
1099 {
1100     return ( ! preg_match("/^([a-z])+$/i" , $str)) ? FALSE : TRUE;
1101 }
1102 // -----
1103
1104 /**
1105 * Alpha-numeric
1106 *
1107 * @access public
1108 * @param string
1109 * @return bool
1110 */
1111 public function alpha_numeric($str)
1112 {
1113     return ( ! preg_match("/^([a-z0-9])+$/i" , $str)) ? FALSE : TRUE;
1114 }
1115 // -----
1116
1117 /**
1118 * Alpha-numeric with underscores and dashes
1119 *
1120 * @access public
1121 * @param string
1122 * @return bool
1123 */
1124 public function alpha_dash($str)
1125 {
1126     return ( ! preg_match("/^([-a-z0-9_-])+$/i" , $str)) ? FALSE : TRUE;
1127 }
1128 // -----
1129
1130 /**
1131 * Numeric
1132 *
1133 * @access public
1134 * @param string
1135 * @return bool
1136 */
1137 public function numeric($str)
1138 {
1139     return (bool)preg_match( '/^[-+]?[0-9]*\.[0-9]+$/i' , $str);
1140 }
1141 // -----
1142
1143 /**
1144 * Is Numeric
1145 *
1146 * @access public
1147 * @param string
1148 * @return bool
1149 */
1150 public function is_numeric($str)
1151 {
1152     return ( ! is_numeric($str)) ? FALSE : TRUE;
1153 }
1154 // -----
1155
1156 /**
1157 * Is Numeric
1158 *
1159 * @access public
1160 * @param string
1161 * @return bool
1162 */
1163

```

```

1164 * Integer
1165 *
1166 * @access public
1167 * @param string
1168 * @return bool
1169 */
1170 public function integer($str)
1171 {
1172     return (bool) preg_match('/^[-+]?[0-9]+$/i', $str);
1173 }
1174 //
1175 // -----
1176 /**
1177 * Decimal number
1178 *
1179 * @access public
1180 * @param string
1181 * @return bool
1182 */
1183 public function decimal($str)
1184 {
1185     return (bool) preg_match('/^[-+]?[0-9]+\.[0-9]+$/i', $str);
1186 }
1187 //
1188 // -----
1189 /**
1190 * Greater than
1191 *
1192 * @access public
1193 * @param string
1194 * @return bool
1195 */
1196 public function greater_than($str, $min)
1197 {
1198     if ( ! is_numeric($str))
1199     {
1200         return FALSE;
1201     }
1202     return $str >    $min;
1203 }
1204 //
1205 // -----
1206 /**
1207 * Less than
1208 *
1209 * @access public
1210 * @param string
1211 * @return bool
1212 */
1213 public function less_than($str, $max)
1214 {
1215     if ( ! is_numeric($str))
1216     {
1217         return FALSE;
1218     }
1219     return $str <    $max;
1220 }
1221 //
1222 // -----
1223 /**
1224 * Is a Natural number (0,1,2,3, etc.)
1225 *
1226 * @access public
1227 * @param string
1228 * @return bool
1229 */
1230 public function is_natural($str)
1231 {
1232     return (bool) preg_match( '/^0-9]+$/i', $str);
1233 }
1234 //
1235 /**
1236 * Is a Natural number, but not a zero (1,2,3, etc.)
1237 *
1238 */
1239 //
1240 /**
1241 * Is a Natural number, but not a zero (1,2,3, etc.)
1242 *
1243 */

```

```

1244 * @access public
1245 * @param string
1246 * @return bool
1247 */
1248 public function is_natural_no_zero($str)
1249 {
1250     if ( ! preg_match( '/^[\d]+$/ ', $str) )
1251     {
1252         return FALSE;
1253     }
1254     if ($str == 0)
1255     {
1256         return FALSE;
1257     }
1258     return TRUE;
1259 }
1260
1261 /**
1262 * Valid Base64
1263 *
1264 * Tests a string for characters outside of the Base64 alphabet
1265 * as defined by RFC 2045 http://www.faqs.org/rfcs/rfc2045
1266 *
1267 * @access public
1268 * @param string
1269 * @return bool
1270 */
1271 public function valid_base64($str)
1272 {
1273     return (bool) ! preg_match('/[^a-zA-Z0-9\/+]/', $str);
1274 }
1275
1276 /**
1277 * Prep data for form
1278 *
1279 * This function allows HTML to be safely shown in a form.
1280 * Special characters are converted.
1281 *
1282 * @access public
1283 * @param string
1284 * @return string
1285 */
1286 public function prep_for_form($data = '')
1287 {
1288     if (is_array($data))
1289     {
1290         foreach ($data as $key => $val)
1291         {
1292             $data[$key] = $this-> prep_for_form($val);
1293         }
1294     }
1295     return $data;
1296 }
1297
1298 if ($this-> _safe_form_data == FALSE OR $data === '')
1299 {
1300     return $data;
1301 }
1302
1303 return str_replace(array("''", '"', '<', '>'), stripslashes($data));
1304 array("'");
1305     , "&quot;" , '&lt;' , '&gt;' ), stripslashes($data));
1306 }
1307
1308 /**
1309 * Prep URL
1310 *
1311 * @access public
1312 * @param string
1313 * @return string
1314 */
1315 public function prep_url($str = '')
1316 {

```

```

1323     if ($str == 'http://' OR $str == '')
1324     {
1325         return '';
1326     }
1327     if (substr($str, 0, 7) != 'http://' && substr($str, 0, 8) != 'https://')
1328     {
1329         $str = 'http://'.$str;
1330     }
1331     return $str;
1332 }
1333
1334 // -----
1335 /**
1336 * Strip Image Tags
1337 *
1338 * @access public
1339 * @param string
1340 * @return string
1341 */
1342 public function strip_image_tags($str)
1343 {
1344     return $this-> CI-> input-> strip_image_tags($str);
1345 }
1346
1347 // -----
1348 /**
1349 * XSS Clean
1350 *
1351 * @access public
1352 * @param string
1353 * @return string
1354 */
1355 public function xss_clean($str)
1356 {
1357     return $this-> CI-> security-> xss_clean($str);
1358 }
1359
1360 // -----
1361 /**
1362 * Convert PHP tags to entities
1363 *
1364 * @access public
1365 * @param string
1366 * @return string
1367 */
1368 public function encode_php_tags($str)
1369 {
1370     return str_replace(array('<?php' , '<?PHP' , '<?' , '?>' ),
1371 array('&lt;?php' , '&lt;?PHP' , '&lt;?' , '?&gt;' ), $str);
1372 }
1373
1374 }
1375 // END Form Validation Class
1376
1377 /*
1378 * End of file Form_validation.php */
1379 /* Location: ./system/libraries/Form_validation.php */
1380
1381
1382

```

# Archivo fuente para Pagination.php

La documentación para este archivo está disponible en [Pagination.php](#)

```
1  <?php      if ( ! defined('BASEPATH')) exit('No direct script access allowed');
2  /**
3   * CodeIgniter
4   *
5   * An open source application development framework for PHP 5.1.6 or newer
6   *
7   * @package      CodeIgniter
8   * @author       ExpressionEngine Dev Team
9   * @copyright    Copyright (c) 2008 - 2011, EllisLab, Inc.
10  * @license      http://codeigniter.com/user_guide/license.html
11  * @link         http://codeigniter.com
12  * @since        Version 1.0
13  * @filesource
14  */
15
16 // -----
17
18 /**
19  * Pagination Class
20  *
21  * @package      CodeIgniter
22  * @subpackage   Libraries
23  * @category    Pagination
24  * @author       ExpressionEngine Dev Team
25  * @link         http://codeigniter.com/user_guide/libraries/pagination.html
26  */
27 class CI_Pagination {
28
29     var $base_url           = '';
30     var $prefix              = '';
31     var $suffix              = '';
32
33     var $total_rows          = 0;
34     var $per_page             = 10;
35     var $num_links            = 2;
36
37     var $cur_page             = 0;
38     var $use_page_numbers      = FALSE;
39     var $first_link           = '&laquo; First';
40     var $next_link             = '&gt;';
41     var $prev_link             = '&lt;';
42     var $last_link             = 'Last &rsaquo;';
43
44     var $uri_segment          = 3;
45
46     var $full_tag_open         = '<ul class="pagination pagination-sm">';
47     var $full_tag_close        = '</ul>';
48     var $first_tag_open        = '<li>';
49     var $first_tag_close       = '</li>';
50
51     var $cur_tag_open          = '<li><a><strong>';
52     var $cur_tag_close         = '</strong></a></li>';
53
54     var $next_tag_open         = '<li>';
55     var $next_tag_close        = '</li>';
56
57     var $prev_tag_open         = '<li>';
58     var $prev_tag_close        = '</li>';
59
60     var $num_tag_open          = '<li>';
61     var $num_tag_close         = '</li>';
62
63     var $page_query_string     = FALSE;
64     var $query_string_segment  = 'per_page';
65     var $display_pages          = TRUE;
66     var $anchor_class           = '';
67
68 /**
69  * Constructor
70  *
71  * @access public
72  */
```

```

67     * @param array initialization parameters
68     */
69     public function __construct($params = array())
70     {
71         if (count($params) > 0)
72         {
73             $this-> initialize($params);
74         }
75
76         if ($this-> anchor_class != '')
77         {
78             $this-> anchor_class = 'class=""' . $this-> anchor_class . '';
79         }
80
81         log_message('debug', "Pagination Class Initialized");
82     }
83
84     // -----
85
86     /**
87      * Initialize Preferences
88      *
89      * @access public
90      * @param array initialization parameters
91      * @return void
92      */
93     function initialize($params = array())
94     {
95         if (count($params) > 0)
96         {
97             foreach ($params as $key => $val)
98             {
99                 if (isset($this-> $key))
100                 {
101                     $this-> $key = $val;
102                 }
103             }
104         }
105     }
106
107     // -----
108
109    /**
110     * Generate the pagination links
111     *
112     * @access public
113     * @return string
114     */
115    function create_links()
116    {
117        // If our item count or per-page total is zero there is no need to continue.
118        if ($this-> total_rows == 0 OR $this-> per_page == 0)
119        {
120            return '';
121        }
122
123        // Calculate the total number of pages
124        $num_pages = ceil($this-> total_rows / $this-> per_page);
125
126        // Is there only one page? Hm... nothing more to do here then.
127        if ($num_pages == 1)
128        {
129            return '';
130        }
131
132        // Set the base page index for starting page number
133        if ($this-> use_page_numbers)
134        {
135            $base_page = 1;
136        }
137        else
138        {
139            $base_page = 0;
140        }
141
142        // Determine the current page number.
143        $CI =& get_instance();
144
145        if ($CI-> config-> item('enable_query_strings') === TRUE OR $this-
> page_query_string === TRUE)

```

```

146
147     {
148         if ($CI-> input-> get($this-> query_string_segment) != $base_page)
149         {
150             $this-> cur_page = $CI-> input-> get($this-> query_string_segment);
151             // Prep the current page - no funny business!
152             $this-> cur_page = (int) $this-> cur_page;
153         }
154     }
155     else
156     {
157         if ($CI-> uri-> segment($this-> uri_segment) != $base_page)
158         {
159             $this-> cur_page = $CI-> uri-> segment($this-> uri_segment);
160             // Prep the current page - no funny business!
161             $this-> cur_page = (int) $this-> cur_page;
162         }
163     }
164 }
165
166 // Set current page to 1 if using page numbers instead of offset
167 if ($this-> use_page_numbers AND $this-> cur_page == 0)
168 {
169     $this-> cur_page = $base_page;
170 }
171
172 $this-> num_links = (int)$this-> num_links;
173
174 if ($this-> num_links < 1)
175 {
176     show_error('Your number of links must be a positive number.');
177 }
178
179 if ( ! is_numeric($this-> cur_page))
180 {
181     $this-> cur_page = $base_page;
182 }
183
184 // Is the page number beyond the result range?
185 // If so we show the last page
186 if ($this-> use_page_numbers)
187 {
188     if ($this-> cur_page > $num_pages)
189     {
190         $this-> cur_page = $num_pages;
191     }
192     else
193     {
194         if ($this-> cur_page > $this-> total_rows)
195         {
196             $this-> cur_page = ($num_pages - 1) * $this-> per_page;
197         }
198     }
199 }
200
201 $uri_page_number = $this-> cur_page;
202
203 if ( ! $this-> use_page_numbers)
204 {
205     $this-> cur_page = floor(($this-> cur_page/$this-> per_page) + 1);
206 }
207
208 // Calculate the start and end numbers. These determine
209 // which number to start and end the digit links with
210 $start = (( $this-> cur_page - $this-> num_links) > 0) ? $this-> cur_page -
($this-> num_links - 1) : 1;
211 $end = (( $this-> cur_page + $this-> num_links) < $num_pages) ? $this-
> cur_page + $this-> num_links : $num_pages;
212
213 // Is pagination being used over GET or POST? If get, add a per_page query
214 // string. If post, add a trailing slash to the base URL if needed
215 if ($CI-> config-> item('enable_query_strings') === TRUE OR $this-
> page_query_string === TRUE)
216 {
217     $this-> base_url = rtrim($this-> base_url). '&' . $this-
> query_string_segment . '=';
218 }
219 else
220 {
221     $this-> base_url = rtrim($this-> base_url, '/') . '/';

```

```

222     }
223
224     // And here we go...
225     $output = '';
226
227     // Render the "First" link
228     if ($this-> first_link !== FALSE AND $this-> cur_page > (    $this-> num_links +
1))
229     {
230         $first_url = ($this-> first_url == '') ? $this-> base_url : $this-> first_url;
231         $output .= $this-> first_tag_open.'<a ' . $this-
> anchor_class.'href=""' . $first_url.'">' . $this-> first_link.'</a>' . $this-
> first_tag_close;
232     }
233
234     // Render the "previous" link
235     if ($this-> prev_link !== FALSE AND $this-> cur_page != 1)
236     {
237         if ($this-> use_page_numbers)
238         {
239             $i = $uri_page_number - 1;
240         }
241         else
242         {
243             $i = $uri_page_number - $this-> per_page;
244         }
245
246         if ($i == 0 && $this-> first_url != '')
247         {
248             $output .= $this-> prev_tag_open.'<a ' . $this-
> anchor_class.'href=""' . $this-> first_url.'">' . $this-
> prev_link.'</a>' . $this-> prev_tag_close;
249         }
250         else
251         {
252             $i = ($i == 0) ? '' : $this-> prefix.$i.$this-> suffix;
253             $output .= $this-> prev_tag_open.'<a ' . $this-
> anchor_class.'href=""' . $this-> base_url.$i.'">' . $this-
> prev_link.'</a>' . $this-> prev_tag_close;
254         }
255     }
256
257     // Render the pages
258     if ($this-> display_pages !== FALSE)
259     {
260         // Write the digit links
261         for ($loop = $start -1; $loop <= $end; $loop++)
262         {
263             if ($this-> use_page_numbers)
264             {
265                 $i = $loop;
266             }
267             else
268             {
269                 $i = ($loop * $this-> per_page) - $this-> per_page;
270             }
271
272             if ($i >= $base_page)
273             {
274                 if ($this-> cur_page == $loop)
275                 {
276                     $output .= $this-> cur_tag_open.$loop.$this-> cur_tag_close; // Current page
277                 }
278                 else
279                 {
280                     $n = ($i == $base_page) ? '' : $i;
281
282                     if ($n == '' && $this-> first_url != '')
283                     {
284                         $output .= $this-> num_tag_open.'<a ' . $this-
> anchor_class.'href=""' . $this-> first_url.'">' . $loop.'</a>' . $this-
> num_tag_close;
285                     }
286                     else
287                     {
288                         $n = ($n == '') ? '' : $this-> prefix.$n.$this-> suffix;
289                         $output .= $this-> num_tag_open.'<a ' . $this-

```

```

>     anchor_class.'href="' . $this-> base_url.$n. '">' . $loop.'</a>' . $this-
> num_tag_close;
292             }
293         }
294     }
295 }
296 }
297
298 // Render the "next" link
299 if ($this-> next_link != FALSE AND $this-> cur_page < $num_pages)
300 {
301     if ($this-> use_page_numbers)
302     {
303         $i = $this-> cur_page + 1;
304     }
305     else
306     {
307         $i = ($this-> cur_page * $this-> per_page);
308     }
309
310     $output .= $this-> next_tag_open.'<a ' . $this-
> anchor_class.'href="' . $this-> base_url.$this-> prefix.$i.$this-
> suffix.'">' . $this-> next_link.'</a>' . $this-> next_tag_close;
311 }
312
313 // Render the "Last" link
314 if ($this-> last_link != FALSE AND ($this-> cur_page + $this-> num_links) <
$num_pages)
315 {
316     if ($this-> use_page_numbers)
317     {
318         $i = $num_pages;
319     }
320     else
321     {
322         $i = ((($num_pages * $this-> per_page) - $this-> per_page));
323     }
324     $output .= $this-> last_tag_open.'<a ' . $this-
> anchor_class.'href="' . $this-> base_url.$this-> prefix.$i.$this-
> suffix.'">' . $this-> last_link.'</a>' . $this-> last_tag_close;
325 }
326
327 // Kill double slashes. Note: Sometimes we can end up with a double slash
328 // in the penultimate link so we'll kill all double slashes.
329 $output = preg_replace("#([^\:]//+#+") , "\\\1/" , $output);
330
331 // Add the wrapper HTML if exists
332 $output = $this-> full_tag_open.$output.$this-> num_tag_open.'<span> Total de
registros: '.number_format($this-> total_rows).'.</span>' . $this-> num_tag_close.$this-
> full_tag_close;
333
334     return $output;
335 }
336 }
337 // END Pagination Class
338
339 /* End of file Pagination.php */
340 /* Location: ./system/libraries/Pagination.php */

```

# Archivo fuente para template.php

La documentación para este archivo está disponible en [template.php](#)

```
1  <?php      if (!defined('BASEPATH')) exit('No direct script access allowed');
2  /**
3   * CodeIgniter
4   *
5   * An open source application development framework for PHP 4.3.2 or newer
6   *
7   * @package CodeIgniter
8   * @author ExpressionEngine Dev Team
9   * @copyright Copyright (c) 2006, EllisLab, Inc.
10  * @license http://codeigniter.com/user_guide/license.html
11  * @link http://codeigniter.com
12  * @since Version 1.0
13  * @filesource
14  */
15
16 // -----
17
18 /**
19  * CodeIgniter Template Class
20  *
21  * This class is and interface to CI's View class. It aims to improve the
22  * interaction between controllers and views. Follow @link for more info
23  *
24  * @package      CodeIgniter
25  * @author       Colin Williams
26  * @subpackage   Libraries
27  * @category    Libraries
28  * @link        http://www.williamsconcepts.com/ci/libraries/template/index.html
29  * @copyright   Copyright (c) 2008, Colin Williams.
30  * @version     1.4.1
31  *
32  */
33 class CI_Template {
34
35     var $CI;
36     var $config;
37     var $template;
38     var $master;
39     var $regions = array(
40         '_scripts' => array(),
41         '_styles'  => array(),
42     );
43     var $output;
44     var $js = array();
45     var $css = array();
46     var $parser = 'parser';
47     var $parser_method = 'parse';
48     var $parse_template = FALSE;
49
50     /**
51      * Constructor
52      *
53      * Loads template configuration, template regions, and validates existence of
54      * default template
55      *
56      * @access public
57     */
58
59     function CI_Template()
60     {
61         // Copy an instance of CI so we can use the entire framework.
62         $this-> CI =& get_instance();
63
64         // Load the template config file and setup our master template and regions
65         include(APPPATH.'config/template'.EXT);
66         if (isset($template))
67     {
```

```

68         $this-> config = $template;
69         $this-> set_template($template['active_template']);
70     }
71 }
72 /**
73 * -----
74 /**
75 * Use given template settings
76 *
77 * @access public
78 * @param string array key to access template settings
79 * @return void
80 */
81
82 function set_template($group)
83 {
84     if (isset($this-> config[$group]))
85     {
86         $this-> template = $this-> config[$group];
87     }
88     else
89     {
90         show_error('The "' . $group . '" template group does not exist. Provide a
91 valid group name or add the group first.');
92     }
93     $this-> initialize($this-> template);
94 }
95 /**
96 * -----
97 /**
98 * Set master template
99 *
100 * @access public
101 * @param string filename of new master template file
102 * @return void
103 */
104
105 function set_master_template($filename)
106 {
107     if (file_exists(APPPATH . 'views/' . $filename) or file_exists(APPPATH . 'views/' . $filename
108 . EXT))
109     {
110         $this-> master = $filename;
111     }
112     else
113     {
114         show_error('The filename provided does not exist in <strong>' . APPPATH
115 . 'views</strong>. Remember to include the extension if other than ".php"' );
116     }
117 }
118 /**
119 * -----
120 /**
121 * Dynamically add a template and optionally switch to it
122 *
123 * @access public
124 * @param string array key to access template settings
125 * @param array properly formed
126 * @return void
127 */
128
129 function add_template($group, $template, $activate = FALSE)
130 {
131     if ( ! isset($this-> config[$group]))
132     {
133         $this-> config[$group] = $template;
134         if ($activate === TRUE)
135         {
136             $this-> initialize($template);
137         }
138     }
139     else
140     {
141         show_error('The "' . $group . '" template group already exists. Use a
142 different group name.');
143     }
}

```

```

144
145 // -----
146
147 /**
148 * Initialize class settings using config settings
149 *
150 * @access public
151 * @param array configuration array
152 * @return void
153 */
154
155 function initialize($props)
156 {
157     // Set master template
158     if (isset($props['template']))
159         && ( file_exists(APPPATH . 'views/' . $props['template']) or file_exists(APPPATH
160 . 'views/' . $props['template'] . EXT)))
161     {
162         $this-> master = $props['template'];
163     }
164     else
165     {
166         // Master template must exist. Throw error.
167         show_error('Either you have not provided a master template or the one provided does
168 not exist in <strong>' . APPPATH . 'views</strong>. Remember to include the extension if
169 other than ".php"');
170     }
171
172     // Load our regions
173     if (isset($props['regions']))
174     {
175         $this-> set_regions($props['regions']);
176     }
177
178     // Set parser and parser method
179     if (isset($props['parser']))
180     {
181         $this-> set_parser($props['parser']);
182     }
183     if (isset($props['parser_method']))
184     {
185         $this-> set_parser_method($props['parser_method']);
186     }
187
188     // -----
189
190 /**
191 * Set regions for writing to
192 *
193 * @access public
194 * @param array properly formed regions array
195 * @return void
196 */
197
198 function set_regions($regions)
199 {
200     if (count($regions))
201     {
202         $this-> regions = array(
203             '_scripts' => array(),
204             '_styles' => array(),
205         );
206         foreach ($regions as $key => $region)
207         {
208             // Regions must be arrays, but we take the burden off the template
209             // developer and insure it here
210             if ( ! is_array($region))
211             {
212                 $this-> add_region($region);
213             }
214             else {
215                 $this-> add_region($key, $region);
216             }
217         }
218     }
219 }

```

```

220 }
221
222 // -----
223
224 /**
225 * Dynamically add region to the currently set template
226 *
227 * @access public
228 * @param string Name to identify the region
229 * @param array Optional array with region defaults
230 * @return void
231 */
232
233 function add_region($name, $props = array())
234 {
235     if ( ! is_array($props) )
236     {
237         $props = array();
238     }
239
240     if ( ! isset($this-> regions[$name]) )
241     {
242         $this-> regions[$name] = $props;
243     }
244     else
245     {
246         show_error('The "' . $name . '" region has already been defined.' );
247     }
248 }
249
250 // -----
251
252 /**
253 * Empty a region's content
254 *
255 * @access public
256 * @param string Name to identify the region
257 * @return void
258 */
259
260 function empty_region($name)
261 {
262     if (isset($this-> regions[$name]['content']))
263     {
264         $this-> regions[$name]['content'] = array();
265     }
266     else
267     {
268         show_error('The "' . $name . '" region is undefined.' );
269     }
270 }
271
272 // -----
273
274 /**
275 * Set parser
276 *
277 * @access public
278 * @param string name of parser class to load and use for parsing methods
279 * @return void
280 */
281
282 function set_parser($parser, $method = NULL)
283 {
284     $this-> parser = $parser;
285     $this-> CI-> load-> library($parser);
286
287     if ($method)
288     {
289         $this-> set_parser_method($method);
290     }
291 }
292
293 // -----
294
295 /**
296 * Set parser method
297 *
298 * @access public
299 * @param string name of parser class member function to call when parsing

```

```

300     * @return void
301     */
302
303     function set_parser_method($method)
304     {
305         $this-> parser_method = $method;
306     }
307
308 // -----
309
310 /**
311     * Write contents to a region
312     *
313     * @access public
314     * @param string region to write to
315     * @param string what to write
316     * @param boolean FALSE to append to region, TRUE to overwrite region
317     * @return void
318 */
319
320     function write($region, $content, $overwrite = FALSE)
321     {
322         if (isset($this-> regions[$region]))
323         {
324             if ($overwrite === TRUE) // Should we append the content or overwrite it
325             {
326                 $this-> regions[$region]['content'] = array($content);
327             } else {
328                 $this-> regions[$region]['content'][] = $content;
329             }
330         }
331
332         // Regions MUST be defined
333         else
334         {
335             show_error(" Cannot write to the '{$region}' region. The region is
undefined.");
336         }
337     }
338
339 // -----
340
341 /**
342     * Write content from a View to a region. 'Views within views'
343     *
344     * @access public
345     * @param string region to write to
346     * @param string view file to use
347     * @param array variables to pass into view
348     * @param boolean FALSE to append to region, TRUE to overwrite region
349     * @return void
350 */
351
352     function write_view($region, $view, $data = NULL, $overwrite = FALSE)
353     {
354         $args = func_get_args();
355
356         // Get rid of non-views
357         unset($args[0], $args[2], $args[3]);
358
359         // Do we have more view suggestions?
360         if (count($args) > 1)
361         {
362             foreach ($args as $suggestion)
363             {
364                 if (file_exists(APPPATH . 'views/' . $suggestion . EXT) or file_exists(APPPATH
.'views/' . $suggestion))
365                 {
366                     // Just change the $view arg so the rest of our method works as normal
367                     $view = $suggestion;
368                     break;
369                 }
370             }
371         }
372
373         $content = $this-> CI-> load-> view($view, $data, TRUE);
374         $this-> write($region, $content, $overwrite);
375
376     }
377

```

```

378 // -----
379 /**
380 * Parse content from a View to a region with the Parser Class
381 *
382 * @access public
383 * @param string region to write to
384 * @param string view file to parse
385 * @param array variables to pass into view for parsing
386 * @param boolean FALSE to append to region, TRUE to overwrite region
387 * @return void
388 */
389
390 function parse_view($region, $view, $data = NULL, $overwrite = FALSE)
391 {
392     $this-> CI-> load-> library('parser');
393
394     $args = func_get_args();
395
396     // Get rid of non-views
397     unset($args[0], $args[2], $args[3]);
398
399     // Do we have more view suggestions?
400     if (count($args) > 1)
401     {
402         foreach ($args as $suggestion)
403         {
404             if (file_exists(APPPATH . 'views/' . $suggestion . EXT) or file_exists(APPPATH
405 . 'views/' . $suggestion))
406             {
407                 // Just change the $view arg so the rest of our method works as normal
408                 $view = $suggestion;
409                 break;
410             }
411         }
412     }
413
414     $content = $this-> CI->{ $this-> parser}->{ $this-> parser_method}($view,
415     $data, TRUE);
416     $this-> write($region, $content, $overwrite);
417 }
418
419 // -----
420 /**
421 * Dynamically include javascript in the template
422 *
423 * NOTE: This function does NOT check for existence of .js file
424 *
425 * @access public
426 * @param string script to import or embed
427 * @param string 'import' to load external file or 'embed' to add as-is
428 * @param boolean TRUE to use 'defer' attribute, FALSE to exclude it
429 * @return TRUE on success, FALSE otherwise
430 */
431
432 function add_js($script, $type = 'import', $defer = FALSE)
433 {
434     $success = TRUE;
435     $js = NULL;
436
437     $this-> CI-> load-> helper('url');
438
439     switch ($type)
440     {
441         case 'import':
442             $filepath = base_url() . $script;
443             $js = '<script type="text/javascript" src=' . $filepath . "'";
444             if ($defer)
445             {
446                 $js .= ' defer="defer" ';
447             }
448             $js .= "></script>";
449             break;
450
451         case 'embed':
452             $js = '<script type="text/javascript" ';
453             if ($defer)
454             {
455

```

```

456         $js .= ' defer="defer"' ;
457     }
458     $js .= ">" ;
459     $js .= $script;
460     $js .= '</script>' ;
461     break;
462 
463     default:
464         $success = FALSE;
465         break;
466     }
467 
468 // Add to js array if it doesn't already exist
469 if ($js != NULL && ! in_array($js, $this-> js))
470 {
471     $this-> js[] = $js;
472     $this-> write('_scripts', $js);
473 }
474 
475 return $success;
476 }
477 
478 // -----
479 /**
480 * Dynamically include CSS in the template
481 *
482 * NOTE: This function does NOT check for existence of .css file
483 *
484 * @access public
485 * @param string CSS file to link, import or embed
486 * @param string 'link', 'import' or 'embed'
487 * @param string media attribute to use with 'link' type only, FALSE for none
488 * @return TRUE on success, FALSE otherwise
489 */
490 
491 function add_css($style, $type = 'link', $media = FALSE)
492 {
493     $success = TRUE;
494     $css = NULL;
495 
496     $this-> CI-> load-> helper('url');
497     $filepath = base_url() . $style;
498 
499     switch ($type)
500     {
501         case 'link':
502 
503             $css = '<link type="text/css" rel="stylesheet" href="' .
504 $filepath . '"';
505             ;
506             if ($media)
507             {
508                 $css .= ' media=' . $media . '"';
509             }
510             $css .= ' />' ;
511             break;
512 
513         case 'import':
514             $css = '<style type="text/css">@import url(' . $filepath .
515 . ')</style>' ;
516             ;
517             break;
518 
519         case 'embed':
520             $css = '<style type="text/css">' ;
521             $css .= $style;
522             $css .= '</style>' ;
523             break;
524 
525         default:
526             $success = FALSE;
527             break;
528     }
529 
530 // Add to js array if it doesn't already exist
531 if ($css != NULL && ! in_array($css, $this-> css))
532 {
533     $this-> css[] = $css;
534     $this-> write('_styles', $css);
535 }

```

```

534     return $success;
535 }
536
537 // -----
538 /**
539  * Render the master template or a single region
540 *
541  * @access public
542  * @param string optionally opt to render a specific region
543  * @param boolean FALSE to output the rendered template, TRUE to return as a string.
544 Always TRUE when $region is supplied
545  * @return void or string (result of template build)
546 */
547
548 function render($region = NULL, $buffer = FALSE, $parse = FALSE)
549 {
550     // Just render $region if supplied
551     if ($region) // Display a specific regions contents
552     {
553         if (isset($this-> regions[$region]))
554         {
555             $output = $this-> _build_content($this-> regions[$region]);
556         }
557         else
558         {
559             show_error(" Cannot render the '{$region}' region. The region is
undefined.");
560         }
561     }
562
563     // Build the output array
564     else
565     {
566         foreach ($this-> regions as $name => $region)
567         {
568             $this-> output[$name] = $this-> _build_content($region);
569         }
570
571         if ($this-> parse_template === TRUE or $parse === TRUE)
572         {
573             // Use provided parser class and method to render the template
574             $output = $this-> CI->{ $this-> parser}->{ $this-> parser_method}($this-
> master, $this-> output, TRUE);
575
576             // Parsers never handle output, but we need to mimick it in this case
577             if ($buffer === FALSE)
578             {
579                 $this-> CI-> output-> set_output($output);
580             }
581             else
582             {
583                 // Use CI's loader class to render the template with our output array
584                 $output = $this-> CI-> load-> view($this-> master, $this-> output,
585 $buffer);
586             }
587         }
588
589         return $output;
590     }
591
592 // -----
593 /**
594  * Load the master template or a single region
595 *
596  * DEPRECATED!
597 *
598  * Use render() to compile and display your template and regions
599 */
600
601 function load($region = NULL, $buffer = FALSE)
602 {
603     $region = NULL;
604     $this-> render($region, $buffer);
605 }
606
607 // -----

```

```

610 /**
611  * Build a region from it's contents. Apply wrapper if provided
612 *
613 * @access private
614 * @param string region to build
615 * @param string HTML element to wrap regions in; like '<div>'
616 * @param array Multidimensional array of HTML elements to apply to $wrapper
617 * @return string Output of region contents
618 */
619
620 function _build_content($region, $wrapper = NULL, $attributes = NULL)
621 {
622     $output = NULL;
623
624     // Can't build an empty region. Exit stage left
625     if ( ! isset($region['content']) or ! count($region['content']) )
626     {
627         return FALSE;
628     }
629
630     // Possibly overwrite wrapper and attributes
631     if ($wrapper)
632     {
633         $region['wrapper'] = $wrapper;
634     }
635     if ($attributes)
636     {
637         $region['attributes'] = $attributes;
638     }
639
640     // Open the wrapper and add attributes
641     if (isset($region['wrapper']))
642     {
643         // This just trims off the closing angle bracket. Like '<p>' to '<p'
644         $output .= substr($region['wrapper'], 0, strlen($region['wrapper']) - 1);
645
646         // Add HTML attributes
647         if (isset($region['attributes']) && is_array($region['attributes']))
648         {
649             foreach ($region['attributes'] as $name => $value)
650             {
651                 // We don't validate HTML attributes. Imagine someone using a custom XML
652                 // template...
653                 $output .= "      $name=\"$value\"      ";
654             }
655
656             $output .= ">";
657         }
658
659         // Output the content items.
660         foreach ($region['content'] as $content)
661         {
662             $output .= $content;
663         }
664
665         // Close the wrapper tag
666         if (isset($region['wrapper']))
667         {
668             // This just turns the wrapper into a closing tag. Like '<p>' to '</p>'
669             $output .= str_replace('<', '</', $region['wrapper']) . "\n";
670         }
671
672         return $output;
673     }
674
675 }
676 // END Template Class
677
678 /* End of file Template.php */
679 /* Location: ./system/application/libraries/Template.php */

```

## Paquete TES

# Archivo fuente para menubuilder.php

La documentación para este archivo está disponible en [menubuilder.php](#)

```
1  ?<?php    if ( ! defined('BASEPATH')) exit('No se permite acceso directo al script');
2
3 /**
4  * Menu Builder
5  *
6  * @package    Libreria
7  * @subpackage Clase
8  * @author     Pascual
9  * @created    2013-01-14
10 */
11
12 class Menubuilder
13 {
14 /**
15  * Guarda la instancia del objeto global CodeIgniter
16  * para utilizarlo en la función estática
17  *
18  * @access private
19  * @var    instance
20  */
21 private static $CI;
22
23 public function __construct()
24 {
25     self::$CI = & get_instance();
26 }
27
28 /**
29  * Construye el menu basandose en los permisos del usuario logeado
30  *
31  * @access public
32  * @param boolean $todos Establece si se debe devolver todos los elementos del menu
33  * @return string
34  */
35 public static function build($todos=false)
36 {
37     $strMenu = '<ul class="nav">' ;
38
39     if(self::$CI-> session-> userdata(GROUP_ID)) {
40         self::$CI-> load-> model(DIR_SIIGS.'/Controlador_model');
41         self::$CI-> load-> model(DIR_SIIGS.'/Entorno_model');
42         self::$CI-> load-> model(DIR_SIIGS.'/Menu_model');
43         self::$CI-> load-> model(DIR_SIIGS.'/Bitacora_model');
44         self::$CI-> load-> model(DIR_SIIGS.'/Usuario_model');
45
46         if(!$todos)
47             $strMenu .= '<li><a href="/index'
id="0">Inicio</a></li>' ;
48
49         self::crearMenu($strMenu, 'NULL', $todos);
50
51         if(!$todos)
52             $strMenu .= '<li><a href="/siigs/usuario/logout">Cerrar
sesión</a></li>' ;
53
54         $strMenu .= '</ul>' ;
55     }
56
57     return $strMenu;
58 }
59
60 /**
61  * Función recursiva que recorre todo el árbol de elementos del menu
62  *
63  * @access public
64  * @param string $strMenu Variable pasada por referencia donde se guardará la cadena de
ul y li
```

```

65     * @param string $id_padre ID del padre de los elementos del arbol de menu
66     * @return void
67     */
68     public static function crearMenu(&      $strMenu, $id_padre='NULL', $todos=false)
69     {
70         $items = self::$CI->    Menu_model->    getByPadre($id_padre);
71         $controlador = '';
72         $entorno = '';
73
74         foreach($items as $item) {
75             $ruta = '#';
76
77             if($item->    ruta &&          $item->    ruta!='#') {
78                 if(strpos($item->    ruta, 'http') === FALSE)
79                     $ruta = '/'. $item->    ruta;
80                 else
81                     $ruta = $item->    ruta;
82             }
83
84             if($item->    id_controlador) {
85                 $controlador = self::$CI->    Controlador_model->    getById($item-
86 > id_controlador); // $this->Controlador_model->getId($item->id_controlador);
87                 $entorno = self::$CI->    Entorno_model->    getById($controlador-
88 > id_entorno); // $this->Entorno_model->getId($controlador->id_entorno);
89
90                 $ruta = '/'. $entorno->    directorio.'/'. $controlador->    clase;
91
92                 if(!$todos) {
93                     // Revisa permisos de acceso para el controlador especifico
94                     if(!Usuario_model::checkCredentials($entorno-
95 > directorio.'::'. $controlador->    clase.'::index', '')) {
96                         continue; // Ignorar la secuencia normal y Seguir con la iteraccion del
97                         foreach
98                             }
99                         }
100
101                     $strMenu .= '<li class="expanded" id="'. $item-
102 > id.'">' . $item->    nombre.( $hijos ? ' >' : '');
103                     else
104                         $strMenu .= '<li class="expanded" id="'. $item-
105 > id.'"><a href="'. $ruta.'"' . $item->    atributo.'>' . $item-
106 > nombre.( $hijos ? ' >' : '').'</a>' ;
107
108                     if ( $hijos ) {
109                         $strMenu .= '<ul>';
110                         self::crearMenu($strMenu, $item->    id, $todos);
111                     }
112                     $strMenu .= '</li>' ;
113                 }
114
115             /**
116             * Función que valida si se visualiza o no la accion especificada (usada en views)
117             *
118             * @access public
119             * @param string $strMenu Variable pasada por referencia donde se guardará la cadena de
120             * ul y li
121             * @param string $id_padre ID del padre de los elementos del arbol de menu
122             * @return void
123             */
124             public static function isGranted($accion)
125             {
126                 if (self::$CI->    session->    userdata(PERMISSIONS)){
127                     foreach(self::$CI->    session->    userdata(PERMISSIONS) as $k=>    $v) {
128                         if(array_search($accion, $v)) {
129                             return true;
130                         }
131                     }
132                 }
133             }
134         }

```

# Archivo fuente para graph.php

La documentación para este archivo está disponible en [graph.php](#)

```
1  ?<?php
2  /**
3   * Controlador Objetos
4   *
5   * @package      Libreria
6   * @subpackage   Controlador
7   * @author       Eliecer
8   * @created      2013-12-10
9   */
10 if ( ! defined('BASEPATH')) exit('No direct script access allowed');
11 class Graph extends CI_Controller
12 {
13     public function __construct()
14     {
15         parent::__construct();
16         $this->load->helper('url');
17     }
18     /**
19      *
20      * Crea una grafica en el lugar que se llame
21      *
22      * @param        string      $title           Titulo de la pagina en el navegador
23      * @param        string      $titulo          titulo o nombre a mostrar en la vista al
24      * @param        array       $array            datos que se graficaran ver ejemplo
25      * @param        array       $label            datos con las etiquetas que se muestran en
26      * @param        array       $grafica          tipo de grafica puede ser :time, basic,
27      * @param        string      $nacimiento      si se necesita mostrar fechas enviar la
28      * @param        array       $axis, bars, bars-h, stacked, horizontal ó pie
29      * @param        string      $nacimiento
30      * fecha inicial
31      *
32      * @return       void
33      *
34      * array ejemplo
35      * array(array("d1"=>"[1,2]", "d2"=>"[1,2]", "d3"=&
36      * gt;"[1,2]"..."dn"=>"[b,h]"))
37      * b= valor en el eje de las x h= valor en el eje de las y
38      * crea grafica solicitud por url array con urlencode()
39      */
40     public function
41     graph_init($title,$titulo,$array,$label,$grafica="",
42     ,$nacimiento="") )
43     {
44         if($grafica=="todos" )
45             ;
46         $data["graficas"]           ='"time","basic","axis","bars",&quo
47         t;bars-h","stacked","horizontal","pie"';
48         ;
49         else
50             {
51                 $cadena="";
52                 $grafica="";
53                 . $grafica;
54                 if(strpos($grafica,"time" ))
55                     $cadena.= '"time",';
56                 if(strpos($grafica,"basic" ))
57                     $cadena.= '"basic",';
58                 if(strpos($grafica,"axis" ))
59                     $cadena.= '"axis",';
60                 if(strpos($grafica,"axis" ))
61                     $cadena.= '"bar,$graficas",';
62                 if(strpos($grafica,"bars" ))
63                     $cadena.= '"bars-h",';
64             ;
65         ;
66     ;
67     ;
68     ;
69     ;
70     ;
71     ;
72     ;
73     ;
74     ;
75     ;
76     ;
77     ;
78     ;
79     ;
80     ;
81     ;
82     ;
83     ;
84     ;
85     ;
86     ;
87     ;
88     ;
89     ;
90     ;
91     ;
92     ;
93     ;
94     ;
95     ;
96     ;
97     ;
98     ;
99     ;
100    ;
101    ;
102    ;
103    ;
104    ;
105    ;
106    ;
107    ;
108    ;
109    ;
110    ;
111    ;
112    ;
113    ;
114    ;
115    ;
116    ;
117    ;
118    ;
119    ;
120    ;
121    ;
122    ;
123    ;
124    ;
125    ;
126    ;
127    ;
128    ;
129    ;
130    ;
131    ;
132    ;
133    ;
134    ;
135    ;
136    ;
137    ;
138    ;
139    ;
140    ;
141    ;
142    ;
143    ;
144    ;
145    ;
146    ;
147    ;
148    ;
149    ;
150    ;
151    ;
152    ;
153    ;
154    ;
155    ;
156    ;
157    ;
158    ;
159    ;
160    ;
161    ;
162    ;
163    ;
164    ;
165    ;
166    ;
167    ;
168    ;
169    ;
170    ;
171    ;
172    ;
173    ;
174    ;
175    ;
176    ;
177    ;
178    ;
179    ;
180    ;
181    ;
182    ;
183    ;
184    ;
185    ;
186    ;
187    ;
188    ;
189    ;
190    ;
191    ;
192    ;
193    ;
194    ;
195    ;
196    ;
197    ;
198    ;
199    ;
200    ;
201    ;
202    ;
203    ;
204    ;
205    ;
206    ;
207    ;
208    ;
209    ;
210    ;
211    ;
212    ;
213    ;
214    ;
215    ;
216    ;
217    ;
218    ;
219    ;
220    ;
221    ;
222    ;
223    ;
224    ;
225    ;
226    ;
227    ;
228    ;
229    ;
230    ;
231    ;
232    ;
233    ;
234    ;
235    ;
236    ;
237    ;
238    ;
239    ;
240    ;
241    ;
242    ;
243    ;
244    ;
245    ;
246    ;
247    ;
248    ;
249    ;
250    ;
251    ;
252    ;
253    ;
254    ;
255    ;
256    ;
257    ;
258    ;
259    ;
260    ;
261    ;
262    ;
263    ;
264    ;
265    ;
266    ;
267    ;
268    ;
269    ;
270    ;
271    ;
272    ;
273    ;
274    ;
275    ;
276    ;
277    ;
278    ;
279    ;
280    ;
281    ;
282    ;
283    ;
284    ;
285    ;
286    ;
287    ;
288    ;
289    ;
290    ;
291    ;
292    ;
293    ;
294    ;
295    ;
296    ;
297    ;
298    ;
299    ;
300    ;
301    ;
302    ;
303    ;
304    ;
305    ;
306    ;
307    ;
308    ;
309    ;
310    ;
311    ;
312    ;
313    ;
314    ;
315    ;
316    ;
317    ;
318    ;
319    ;
320    ;
321    ;
322    ;
323    ;
324    ;
325    ;
326    ;
327    ;
328    ;
329    ;
330    ;
331    ;
332    ;
333    ;
334    ;
335    ;
336    ;
337    ;
338    ;
339    ;
340    ;
341    ;
342    ;
343    ;
344    ;
345    ;
346    ;
347    ;
348    ;
349    ;
350    ;
351    ;
352    ;
353    ;
354    ;
355    ;
356    ;
357    ;
358    ;
359    ;
360    ;
361    ;
362    ;
363    ;
364    ;
365    ;
366    ;
367    ;
368    ;
369    ;
370    ;
371    ;
372    ;
373    ;
374    ;
375    ;
376    ;
377    ;
378    ;
379    ;
380    ;
381    ;
382    ;
383    ;
384    ;
385    ;
386    ;
387    ;
388    ;
389    ;
390    ;
391    ;
392    ;
393    ;
394    ;
395    ;
396    ;
397    ;
398    ;
399    ;
400    ;
401    ;
402    ;
403    ;
404    ;
405    ;
406    ;
407    ;
408    ;
409    ;
410    ;
411    ;
412    ;
413    ;
414    ;
415    ;
416    ;
417    ;
418    ;
419    ;
420    ;
421    ;
422    ;
423    ;
424    ;
425    ;
426    ;
427    ;
428    ;
429    ;
430    ;
431    ;
432    ;
433    ;
434    ;
435    ;
436    ;
437    ;
438    ;
439    ;
440    ;
441    ;
442    ;
443    ;
444    ;
445    ;
446    ;
447    ;
448    ;
449    ;
450    ;
451    ;
452    ;
453    ;
454    ;
455    ;
456    ;
457    ;
458    ;
459    ;
460    ;
461    ;
462    ;
463    ;
464    ;
465    ;
466    ;
467    ;
468    ;
469    ;
470    ;
471    ;
472    ;
473    ;
474    ;
475    ;
476    ;
477    ;
478    ;
479    ;
480    ;
481    ;
482    ;
483    ;
484    ;
485    ;
486    ;
487    ;
488    ;
489    ;
490    ;
491    ;
492    ;
493    ;
494    ;
495    ;
496    ;
497    ;
498    ;
499    ;
500    ;
501    ;
502    ;
503    ;
504    ;
505    ;
506    ;
507    ;
508    ;
509    ;
510    ;
511    ;
512    ;
513    ;
514    ;
515    ;
516    ;
517    ;
518    ;
519    ;
520    ;
521    ;
522    ;
523    ;
524    ;
525    ;
526    ;
527    ;
528    ;
529    ;
530    ;
531    ;
532    ;
533    ;
534    ;
535    ;
536    ;
537    ;
538    ;
539    ;
540    ;
541    ;
542    ;
543    ;
544    ;
545    ;
546    ;
547    ;
548    ;
549    ;
550    ;
551    ;
552    ;
553    ;
554    ;
555    ;
556    ;
557    ;
558    ;
559    ;
560    ;
561    ;
562    ;
563    ;
564    ;
565    ;
566    ;
567    ;
568    ;
569    ;
570    ;
571    ;
572    ;
573    ;
574    ;
575    ;
576    ;
577    ;
578    ;
579    ;
580    ;
581    ;
582    ;
583    ;
584    ;
585    ;
586    ;
587    ;
588    ;
589    ;
590    ;
591    ;
592    ;
593    ;
594    ;
595    ;
596    ;
597    ;
598    ;
599    ;
599    ;
600    ;
601    ;
602    ;
603    ;
604    ;
605    ;
606    ;
607    ;
608    ;
609    ;
610    ;
611    ;
612    ;
613    ;
614    ;
615    ;
616    ;
617    ;
618    ;
619    ;
620    ;
621    ;
622    ;
623    ;
624    ;
625    ;
626    ;
627    ;
628    ;
629    ;
630    ;
631    ;
632    ;
633    ;
634    ;
635    ;
636    ;
637    ;
638    ;
639    ;
640    ;
641    ;
642    ;
643    ;
644    ;
645    ;
646    ;
647    ;
648    ;
649    ;
650    ;
651    ;
652    ;
653    ;
654    ;
655    ;
656    ;
657    ;
658    ;
659    ;
660    ;
661    ;
662    ;
663    ;
664    ;
665    ;
666    ;
667    ;
668    ;
669    ;
670    ;
671    ;
672    ;
673    ;
674    ;
675    ;
676    ;
677    ;
678    ;
679    ;
680    ;
681    ;
682    ;
683    ;
684    ;
685    ;
686    ;
687    ;
688    ;
689    ;
690    ;
691    ;
692    ;
693    ;
694    ;
695    ;
696    ;
697    ;
698    ;
699    ;
700    ;
701    ;
702    ;
703    ;
704    ;
705    ;
706    ;
707    ;
708    ;
709    ;
710    ;
711    ;
712    ;
713    ;
714    ;
715    ;
716    ;
717    ;
718    ;
719    ;
720    ;
721    ;
722    ;
723    ;
724    ;
725    ;
726    ;
727    ;
728    ;
729    ;
729    ;
730    ;
731    ;
732    ;
733    ;
734    ;
735    ;
736    ;
737    ;
738    ;
739    ;
739    ;
740    ;
741    ;
742    ;
743    ;
744    ;
745    ;
746    ;
747    ;
748    ;
749    ;
749    ;
750    ;
751    ;
752    ;
753    ;
754    ;
755    ;
756    ;
757    ;
758    ;
759    ;
759    ;
760    ;
761    ;
762    ;
763    ;
764    ;
765    ;
766    ;
767    ;
768    ;
769    ;
769    ;
770    ;
771    ;
772    ;
773    ;
774    ;
775    ;
776    ;
777    ;
778    ;
779    ;
779    ;
780    ;
781    ;
782    ;
783    ;
784    ;
785    ;
786    ;
787    ;
788    ;
789    ;
789    ;
790    ;
791    ;
792    ;
793    ;
794    ;
795    ;
796    ;
797    ;
798    ;
799    ;
799    ;
800    ;
801    ;
802    ;
803    ;
804    ;
805    ;
806    ;
807    ;
808    ;
809    ;
809    ;
810    ;
811    ;
812    ;
813    ;
814    ;
815    ;
816    ;
817    ;
818    ;
819    ;
819    ;
820    ;
821    ;
822    ;
823    ;
824    ;
825    ;
826    ;
827    ;
828    ;
829    ;
829    ;
830    ;
831    ;
832    ;
833    ;
834    ;
835    ;
836    ;
837    ;
838    ;
839    ;
839    ;
840    ;
841    ;
842    ;
843    ;
844    ;
845    ;
846    ;
847    ;
848    ;
849    ;
849    ;
850    ;
851    ;
852    ;
853    ;
854    ;
855    ;
856    ;
857    ;
858    ;
859    ;
859    ;
860    ;
861    ;
862    ;
863    ;
864    ;
865    ;
866    ;
867    ;
868    ;
869    ;
869    ;
870    ;
871    ;
872    ;
873    ;
874    ;
875    ;
876    ;
877    ;
878    ;
879    ;
879    ;
880    ;
881    ;
882    ;
883    ;
884    ;
885    ;
886    ;
887    ;
888    ;
889    ;
889    ;
890    ;
891    ;
892    ;
893    ;
894    ;
895    ;
896    ;
897    ;
898    ;
899    ;
899    ;
900    ;
901    ;
902    ;
903    ;
904    ;
905    ;
906    ;
907    ;
908    ;
909    ;
909    ;
910    ;
911    ;
912    ;
913    ;
914    ;
915    ;
916    ;
917    ;
918    ;
919    ;
919    ;
920    ;
921    ;
922    ;
923    ;
924    ;
925    ;
926    ;
927    ;
928    ;
929    ;
929    ;
930    ;
931    ;
932    ;
933    ;
934    ;
935    ;
936    ;
937    ;
938    ;
939    ;
939    ;
940    ;
941    ;
942    ;
943    ;
944    ;
945    ;
946    ;
947    ;
948    ;
949    ;
949    ;
950    ;
951    ;
952    ;
953    ;
954    ;
955    ;
956    ;
957    ;
958    ;
959    ;
959    ;
960    ;
961    ;
962    ;
963    ;
964    ;
965    ;
966    ;
967    ;
968    ;
969    ;
969    ;
970    ;
971    ;
972    ;
973    ;
974    ;
975    ;
976    ;
977    ;
978    ;
979    ;
979    ;
980    ;
981    ;
982    ;
983    ;
984    ;
985    ;
986    ;
987    ;
988    ;
989    ;
989    ;
990    ;
991    ;
992    ;
993    ;
994    ;
995    ;
996    ;
997    ;
998    ;
999    ;
999    ;
1000   ;
1001   ;
1002   ;
1003   ;
1004   ;
1005   ;
1006   ;
1007   ;
1008   ;
1009   ;
1009   ;
1010   ;
1011   ;
1012   ;
1013   ;
1014   ;
1015   ;
1016   ;
1017   ;
1018   ;
1019   ;
1019   ;
1020   ;
1021   ;
1022   ;
1023   ;
1024   ;
1025   ;
1026   ;
1027   ;
1028   ;
1029   ;
1029   ;
1030   ;
1031   ;
1032   ;
1033   ;
1034   ;
1035   ;
1036   ;
1037   ;
1038   ;
1039   ;
1039   ;
1040   ;
1041   ;
1042   ;
1043   ;
1044   ;
1045   ;
1046   ;
1047   ;
1048   ;
1049   ;
1049   ;
1050   ;
1051   ;
1052   ;
1053   ;
1054   ;
1055   ;
1056   ;
1057   ;
1058   ;
1059   ;
1059   ;
1060   ;
1061   ;
1062   ;
1063   ;
1064   ;
1065   ;
1066   ;
1067   ;
1068   ;
1069   ;
1069   ;
1070   ;
1071   ;
1072   ;
1073   ;
1074   ;
1075   ;
1076   ;
1077   ;
1078   ;
1079   ;
1079   ;
1080   ;
1081   ;
1082   ;
1083   ;
1084   ;
1085   ;
1086   ;
1087   ;
1088   ;
1089   ;
1089   ;
1090   ;
1091   ;
1092   ;
1093   ;
1094   ;
1095   ;
1096   ;
1097   ;
1098   ;
1099   ;
1099   ;
1100   ;
1101   ;
1102   ;
1103   ;
1104   ;
1105   ;
1106   ;
1107   ;
1108   ;
1109   ;
1109   ;
1110   ;
1111   ;
1112   ;
1113   ;
1114   ;
1115   ;
1116   ;
1117   ;
1118   ;
1119   ;
1119   ;
1120   ;
1121   ;
1122   ;
1123   ;
1124   ;
1125   ;
1126   ;
1127   ;
1128   ;
1129   ;
1129   ;
1130   ;
1131   ;
1132   ;
1133   ;
1134   ;
1135   ;
1136   ;
1137   ;
1138   ;
1139   ;
1139   ;
1140   ;
1141   ;
1142   ;
1143   ;
1144   ;
1145   ;
1146   ;
1147   ;
1148   ;
1149   ;
1149   ;
1150   ;
1151   ;
1152   ;
1153   ;
1154   ;
1155   ;
1156   ;
1157   ;
1158   ;
1159   ;
1159   ;
1160   ;
1161   ;
1162   ;
1163   ;
1164   ;
1165   ;
1166   ;
1167   ;
1168   ;
1169   ;
1169   ;
1170   ;
1171   ;
1172   ;
1173   ;
1174   ;
1175   ;
1176   ;
1177   ;
1178   ;
1179   ;
1179   ;
1180   ;
1181   ;
1182   ;
1183   ;
1184   ;
1185   ;
1186   ;
1187   ;
1188   ;
1189   ;
1189   ;
1190   ;
1191   ;
1192   ;
1193   ;
1194   ;
1195   ;
1196   ;
1197   ;
1198   ;
1199   ;
1199   ;
1200   ;
1201   ;
1202   ;
1203   ;
1204   ;
1205   ;
1206   ;
1207   ;
1208   ;
1209   ;
1209   ;
1210   ;
1211   ;
1212   ;
1213   ;
1214   ;
1215   ;
1216   ;
1217   ;
1218   ;
1219   ;
1219   ;
1220   ;
1221   ;
1222   ;
1223   ;
1224   ;
1225   ;
1226   ;
1227   ;
1228   ;
1229   ;
1229   ;
1230   ;
1231   ;
1232   ;
1233   ;
1234   ;
1235   ;
1236   ;
1237   ;
1238   ;
1239   ;
1239   ;
1240   ;
1241   ;
1242   ;
1243   ;
1244   ;
1245   ;
1246   ;
1247   ;
1248   ;
1249   ;
1249   ;
1250   ;
1251   ;
1252   ;
1253   ;
1254   ;
1255   ;
1256   ;
1257   ;
1258   ;
1259   ;
1259   ;
1260   ;
1261   ;
1262   ;
1263   ;
1264   ;
1265   ;
1266   ;
1267   ;
1268   ;
1269   ;
1269   ;
1270   ;
1271   ;
1272   ;
1273   ;
1274   ;
1275   ;
1276   ;
1277   ;
1278   ;
1278   ;
1279   ;
1280   ;
1281   ;
1282   ;
1283   ;
1284   ;
1285   ;
1286   ;
1287   ;
1287   ;
1288   ;
1289   ;
1290   ;
1291   ;
1292   ;
1293   ;
1294   ;
1295   ;
1296   ;
1297   ;
1298   ;
1299   ;
1299   ;
1300   ;
1301   ;
1302   ;
1303   ;
1304   ;
1305   ;
1306   ;
1307   ;
1308   ;
1309   ;
1309   ;
1310   ;
1311   ;
1312   ;
1313   ;
1314   ;
1315   ;
1316   ;
1317   ;
1318   ;
1319   ;
1319   ;
1320   ;
1321   ;
1322   ;
1323   ;
1324   ;
1325   ;
1326   ;
1327   ;
1328   ;
1329   ;
1329   ;
1330   ;
1331   ;
1332   ;
1333   ;
1334   ;
1335   ;
1336   ;
1337   ;
1338   ;
1339   ;
1339   ;
1340   ;
1341   ;
1342   ;
1343   ;
1344   ;
1345   ;
1346   ;
1347   ;
1348   ;
1349   ;
1349   ;
1350   ;
1351   ;
1352   ;
1353   ;
1354   ;
1355   ;
1356   ;
1357   ;
1358   ;
1359   ;
1359   ;
1360   ;
1361   ;
1362   ;
1363   ;
1364   ;
1365   ;
1366   ;
1367   ;
1368   ;
1369   ;
1369   ;
1370   ;
1371   ;
1372   ;
1373   ;
1374   ;
1375   ;
1376   ;
1377   ;
1378   ;
1378   ;
1379   ;
1380   ;
1381   ;
1382   ;
1383   ;
1384   ;
1385   ;
1386   ;
1387   ;
1387   ;
1388   ;
1389   ;
1390   ;
1391   ;
1392   ;
1393   ;
1394   ;
1395   ;
1396   ;
1397   ;
1398   ;
1399   ;
1399   ;
1400   ;
1401   ;
1402   ;
1403   ;
1404   ;
1405   ;
1406   ;
1407   ;
1408   ;
1409   ;
1409   ;
1410   ;
1411   ;
1412   ;
1413   ;
1414   ;
1415   ;
1416   ;
1417   ;
1418   ;
1419   ;
1419   ;
1420   ;
1421   ;
1422   ;
1423   ;
1424   ;
1425   ;
1426   ;
1427   ;
1428   ;
1429   ;
1429   ;
1430   ;
1431   ;
1432   ;
1433   ;
1434   ;
1435   ;
1436   ;
1437   ;
1438   ;
1439   ;
1439   ;
1440   ;
1441   ;
1442   ;
1443   ;
1444   ;
1445   ;
1446   ;
1447   ;
1448   ;
1449   ;
1449   ;
1450   ;
1451   ;
1452   ;
1453   ;
1454   ;
1455   ;
1456   ;
1457   ;
1458   ;
1459   ;
1459   ;
1460   ;
1461   ;
1462   ;
1463   ;
1464   ;
1465   ;
1466   ;
1467   ;
1468   ;
1469   ;
1469   ;
1470   ;
1471   ;
1472   ;
1473   ;
1474   ;
1475   ;
1476   ;
1477   ;
1478   ;
1478   ;
1479   ;
1480   ;
1481   ;
1482   ;
1483   ;
1484   ;
1485   ;
1486   ;
1487   ;
1487   ;
1488   ;
1489   ;
1490   ;
1491   ;
1492   ;
1493   ;
1494   ;
1495   ;
1496   ;
1497   ;
1498   ;
1498   ;
1499   ;
1500   ;
1501   ;
1502   ;
1503   ;
1504   ;
1505   ;
1506   ;
1507   ;
1508   ;
1509   ;
1509   ;
1510   ;
1511   ;
1512   ;
1513   ;
1514   ;
1515   ;
1516   ;
1517   ;
1518   ;
1519   ;
1519   ;
1520   ;
1521   ;
1522   ;
1523   ;
1524   ;
1525   ;
1526   ;
1527   ;
1528   ;
1529   ;
1529   ;
1530   ;
1531   ;
1532   ;
1533   ;
1534   ;
1535   ;
1536   ;
1537   ;
1538   ;
1539   ;
1539   ;
1540   ;
1541   ;
1542   ;
1543   ;
1544   ;
1545   ;
1546   ;
1547   ;
1548   ;
1549   ;
1549   ;
1550   ;
1551   ;
1552   ;
1553   ;
1554   ;
1555   ;
1556   ;
1557   ;
1558   ;
1559   ;
1559   ;
1560   ;
1561   ;
1562   ;
1563   ;
1564   ;
1565   ;
1566   ;
1567   ;
1568   ;
1569   ;
1569   ;
1570   ;
1571   ;
1572   ;
1573   ;
1574   ;
1575   ;
1576   ;
1577   ;
1578   ;
1578   ;
1579   ;
1580   ;
1581   ;
1582   ;
1583   ;
1584   ;
1585   ;
1586   ;
1587   ;
1587   ;
1588   ;
1589   ;
1590   ;
1591   ;
1592   ;
1593   ;
1594   ;
1595   ;
1596   ;
1597   ;
1598   ;
1599   ;
1599   ;
1600   ;
1601   ;
1602   ;
1603   ;
1604   ;
1605   ;
1606   ;
1607   ;
1608   ;
1609   ;
1609   ;
1610   ;
1611   ;
1612   ;
1613   ;
1614   ;
1615   ;
1616   ;
1617   ;
1618   ;
1619   ;
1619   ;
1620   ;
1621   ;
1622   ;
1623   ;
1624   ;
1625   ;
1626   ;
1627   ;
1628   ;
1629   ;
1629   ;
1630   ;
1631   ;
1632   ;
1633   ;
1634   ;
1635   ;
1636   ;
1637   ;
1638   ;
1639   ;
1639   ;
1640   ;
1641   ;
1642   ;
1643   ;
1644   ;
1645   ;
1646   ;
1647   ;
1648   ;
1649   ;
1649   ;
1650   ;
1651   ;
1652   ;
1653   ;
1654   ;
1655   ;
1656   ;
1657   ;
1658   ;
1659   ;
1659   ;
1660   ;
1661   ;
1662   ;
1663   ;
1664   ;
1665   ;
1666   ;
1667   ;
1668   ;
1669   ;
1669   ;
1670   ;
1671   ;
1672   ;
1673   ;
1674   ;
1675   ;
1676   ;
1677   ;
1678   ;
1678   ;
1679   ;
1680   ;
1681   ;
1682   ;
1683   ;
1684   ;
1685   ;
1686   ;
1687   ;
1687   ;
1688   ;
1689   ;
1690   ;
1691   ;
1692   ;
1693   ;
1694   ;
1695   ;
1696   ;
1697   ;
1698   ;
1699   ;
1699   ;
1700   ;
1701   ;
1702   ;
1703   ;
1704   ;
1705   ;
1706   ;
1707   ;
1708   ;
1709   ;
1709   ;
1710   ;
1711   ;
1712   ;
1713   ;
1714   ;
1715   ;
1716   ;
1717   ;
1718   ;
1719   ;
1719   ;
1720   ;
1721   ;
1722   ;
1723   ;
1724   ;
1725   ;
1726   ;
1727   ;
1728   ;
1729   ;
1729   ;
1730   ;
1731   ;
1732   ;
1733   ;
1734   ;
1735   ;
1736   ;
1737   ;
1738   ;
1739   ;
1739   ;
1740   ;
1741   ;
1742   ;
1743   ;
1744   ;
1745   ;
1746   ;
1747   ;
1748   ;
1749   ;
1749   ;
1750   ;
1751   ;
1752   ;
1753   ;
1754   ;
1755   ;
1756   ;
1757   ;
1758   ;
1759   ;
1759   ;
1760   ;
1761   ;
1762   ;
1763   ;
1764   ;
1765   ;
1766   ;
1767   ;
1768   ;
1769   ;
1769   ;
1770   ;
1771   ;
1772   ;
1773   ;
1774   ;
1775   ;
1776   ;
1777   ;
1778   ;
1778   ;
1779   ;
1780   ;
1781   ;
1782   ;
1783   ;
1784   ;
1785   ;
1786   ;
1787   ;
1787   ;
1788   ;
1789   ;
1790   ;
1791   ;
1792   ;
1793   ;
1794   ;
1795   ;
1796   ;
1797   ;
1798   ;
1799   ;
1799   ;
1800   ;
1801   ;
1802   ;
1803   ;
1804   ;
1805   ;
1806   ;
1807   ;
1808   ;
1809   ;
1809   ;
1810   ;
1811   ;
1812   ;
1813   ;
1814   ;
1815   ;
1816   ;
1817   ;
1818   ;
1819   ;
1819   ;
1820   ;
1821   ;
1822   ;
1823   ;
1824   ;
1825   ;
1826   ;
1827   ;
1828   ;
1829   ;
1829   ;
1830   ;
1831   ;
1832   ;
1833   ;
1834   ;
1835   ;
1836   ;
1837   ;
1838   ;
1839   ;
1839   ;
1840   ;
1841   ;
1842   ;
1843   ;
1844   ;
1845   ;
1846   ;
1847   ;
1848   ;
1849   ;
1849   ;
1850   ;
1851   ;
1852   ;
1853   ;
1854   ;
1855   ;
1856   ;
1857   ;
1858   ;
1859   ;
1859   ;
1860   ;
1861   ;
1862   ;
1863   ;
1864   ;
1865   ;
1866   ;
1867   ;
1868   ;
1869   ;
1869   ;
1870   ;
1871   ;
1872   ;
1873   ;
1874
```

```

59         if(strpos($grafica,"bars-h")           ))  
60             $cadena.= "stacked", '  
61  
62         if(strpos($grafica,"stacked")        ))  
63             $cadena.= "horizontal", '  
64  
65         if(strpos($grafica,"pie")            ))  
66             $cadena.= "pie" '  
67  
68     $data[ "graficas" ] = $cadena;  
69 }  
70  
71 $url= "/grafica/graph" ;  
72 $data[ "title" ] = $title;  
73 $data[ "title" ] = $title;  
74 $data[ "nacimiento" ] = $nacimiento;  
75 $data[ "titulo" ] = str_replace("%20", " ", $titulo);  
76 $data[ "array" ] = json_decode(urldecode($array));  
77 $data[ "etiqueta" ] = json_decode(urldecode($label));  
78 $this-> load-> view(DIR_TES.$url,$data);  
79 }  
80  
81 /**
82 * crea un objeto mapa con la ayuda de la api de google
83 *
84 * @param string $lugar Especifica el lugar donde se centra el mapa
85 * @param int $zoom Especifica el zoom de acercamiento en el mapa
86 * @param boolean $rewrite Si se desea que sea una pagina o estar
87 enbebida en otra 0=embebido 1=pagina
88 * @param array $datos datos a mostrar en el mapa
89 *
90 * @return void
91 */
92 public function map($lugar="Chiapas", $zoom=6,$rewrite=0,$datos="")  
{
93     if($datos=="") $datos=$this-> input-> post('datos');
94     $cadena="";  
95     $data[ "zoom" ] = $zoom;  
96     $data[ "lugar" ] = $lugar;  
97     if($datos)  
98         foreach($datos as $x)
99     {
100         $cadena.= " " . $x[ "localidad" ]. " , " ;
101         $cadena.= " " . $x[ "lat" ]. " , " ;
102         $cadena.= " " . $x[ "lon" ]. " , " ;
103         $cadena.= " " . $x[ "descripcion" ]. " , " ;
104         $cadena.= " " . $x[ "imagen" ]. " , " ;
105         $cadena.= " " . $x[ "icono" ]. " , " ;
106     }
107     $data[ "array" ] = $cadena;
108     if($rewrite==0)
109     {
110         echo ".";
111         $data[ "api" ]='<script type="text/javascript"
src="http://maps.google.com/maps/api/js?sensor=false&language=es"></script>';
112     }
113     $this-> template-> write('header','','true');
114     $this-> template-> write('footer','','true');
115     $this-> template-> write('menu','','true');
116     $this-> template-> write('sala_prensa','','true');
117     $this-> template-> write_view('content',DIR_TES.'/grafica/map', $data);
118     $this-> template-> render();
119 }
120 }
121 else
122 {
123     $data[ "api" ]='';
124     $this-> load-> view(DIR_TES.'/grafica/map', $data);
125 }
126 }
127 }
128 ?>

```

# Archivo fuente para obtenercurp.php

La documentación para este archivo está disponible en [obtenercurp.php](#)

```
1  <?php
2  /**
3   * Controlador Objeto
4   *
5   * @package      Libreria
6   * @subpackage   Controlador
7   * @author       Eliecer
8   * @created      2013-12-10
9   */
10 if ( ! defined('BASEPATH')) exit('No direct script access allowed');
11 class Obtenercurp extends CI_Controller
12 {
13     /**
14      *Arreglo con los estados y sus abreviaturas, sirven para el cálculo de la CURP
15      * @var $estados
16      */
17     public $estados=
18     array(
19         array(
20             "AGUASCALIENTES"      => "AS"
21             "BAJA CALIFORNIA NTE" => "BC"
22             "BAJA CALIFORNIA NORTE" => "BC"
23             "BAJA CALIFORNIA"      => "BC"
24             "BAJA CALIFORNIA SUR"  => "BS"
25             "CAMPECHE"            => "CC"
26             "COAHUILA"             => "CL"
27             "COLIMA"               => "CM"
28             "CHIAPAS"              => "CS"
29             "CHIHUAHUA"            => "CH"
30             "DISTRITO FEDERAL"    => "DF"
31             "DURANGO"              => "DG"
32             "GUANAJUATO"           => "GT"
33             "GUERRERO"              => "GR"
34             "HIDALGO"              => "HG"
35             "JALISCO"              => "JC"
36             "MEXICO"                => "MC"
37             "MICHOACAN"             => "MN"
38             "MORELOS"              => "MS"
39             "NAYARIT"              => "NT"
40             "NUEVO LEON"            => "NL"
41             "OAXACA"                => "OC"
42             "PUEBLA"                => "PL"
43             "QUERETARO"              => "QT"
44             "QUINTANA ROO"          => "QR"
45             "SAN LUIS POTOSI"        => "SP"
46             "SINALOA"                => "SL"
47             "SONORA"                 => "SR"
48             "TABASCO"                => "TC"
49             "TAMAULIPAS"              => "TS"
50             "TLAXCALA"                => "TL"
51             "VERACRUZ"                => "VZ"
52             "ZACATECAS"              => "ZS"
53             "EXTERIOR MEXICANO"      => "SM"
54             "NACIDO EN EL EXTRANJERO" => "NE"
55         )
56     );
57
58     public function __construct()
59     {
60         parent::__construct();
61         $this-> load-> helper('url');
62     }
63     /**
64      *
65      * Consulta si la curp existe en la base de datos de la condusef
66      *
67      * @param      string      $paterno      Apellido paterno de la persona

```

```

68      * @param      string      $materno      Apellido materno
69      * @param      string      $nombre       Nombre o nombres
70      * @param      int         $dia        Dia de nacimiento
71      * @param      int         $mes        Mes de nacimiento
72      * @param      int         $year       Año de nacimiento
73      * @param      string      $sexo        Sexo
74      * @param      string      $estado      Lugar de nacimiento
75      * @param      string      $regresar    Tipo de retorno =1 return array !=1 json
76
77      * @return
78      */
79      public function
80      curp($paterno,$materno,$nombre,$dia,$mes,$year,$sexo,$estado,$regresar="") {
81      {
82          $estados=$this-> estados;
83          $ap=strtoupper($paterno);
84          $am=strtoupper($materno);
85          $na=strtoupper($nombre);
86
87          $d=$dia;
88          if($d< 10) $d="0".(int)$d;
89          $m=$mes;
90          if($m< 10) $m="0".(int)$m;
91
92          $y=$year;
93          $se=$sexo;
94          $sse=strtoupper($se);
95          if($sse=="HOMBRE" || $sse=="MASCULINO" || $sse=="M")
96              $se="H";
97          if($sse=="MUJER" || $sse=="FEMENINO" || $sse=="F")
98              $se="M";
99          $estado=strtoupper($estado);
100
101         $edo=$estados[0][$estado];
102         if
103             ($ap!="") && ($am!="") && ($y!="") && ($na!="") && ($se!="") && ($d!="") && ($edo!="")
104         {
105             $ch = curl_init();
106             curl_setopt($ch, CURLOPT_URL,
107             "http://consultas.curp.gob.mx/CurpSP/curp11.do?strPrimerApellido=$ap&strSegundoApellido=$am&
108             ;strNombre=$na&strdia=$d&strmes=$m&stranio=$y&sSexoA=$se&sEntidadA=$edo&rdbBD=
109             myoracle&strTipo=A&codigo=bf139");
110
111             curl_setopt($ch, CURLOPT_HTTPHEADER, array("Cookie:
112             JSESSIONID=XrQFT2YSf8BMmwnbJ7HyFlnfttYcjqp3dtJDjQ7HM2NRz84GGW12!-767651644"));
113
114             curl_setopt($ch, CURLOPT_FOLLOWLOCATION, 1);
115             curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);
116             curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
117             curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
118
119             curl_setopt($ch, CURLOPT_USERAGENT, "Mozilla/5.0 (Windows; U; Windows NT 6.0;
120             en-US; rv:1.9.0.6) Gecko/2009011913 Firefox/3.0.6 (.NET CLR 3.5.30729)");
121             $html = curl_exec($ch);
122             curl_close($ch);
123
124             echo
125             "http://consultas.curp.gob.mx/CurpSP/curp11.do?strPrimerApellido=$ap&strSegundoApellido=$am&
126             ;strNombre=$na&strdia=$d&strmes=$m&stranio=$y&sSexoA=$se&sEntidadA=$edo&rdbBD=
127             myoracle&strTipo=A";
128             echo $html;
129
130             $pos=stripos($html,'<td class="TablaTitulo2"><span
131             class="NotaBlanca">Curp</span></td>
132             <td><b class="Nota">' );
133             $t=34;
134             $html=substr($html,$pos,strlen($html)-$pos);
135
136             //echo
137             "$http://consultas.curp.gob.mx/CurpSP/curp11.do?strPrimerApellido=$ap&strSegundoApellido=$am&
138             ;strNombre=$na&strdia=$d&strmes=$m&stranio=$y&sSexoA=$se&sEntidadA=$edo&rdbBD=
139             myoracle&strTipo=A";
140             //print($html);
141
142             $cu=substr($html,stripos($html,'>Curp' )+($t+6),18);
143
144             $ap=substr($html,stripos($html,'>Primer Apellido' )+($t+17),38);
145             $ap=substr($ap,0,stripos($ap,'<' ));

```

```

132
133     $am=substr($html,stripos($html,'>Segundo Apellido'      )+($t+20),38);
134     $am=substr($am,0,stripos($am,'<'      ));
135
136     $na=substr($html,stripos($html,'>Nombre(s)'      )+($t+11),18);
137     $na=substr($na,0,stripos($na,'<'      ));
138
139     $se=substr($html,stripos($html,'>Sexo'      )+($t+6),18);
140     $se=substr($se,0,stripos($se,'<'      ));
141
142     $fn=substr($html,stripos($html,'>Fecha de Nacimiento'      )+($t+33),12);
143
144     $se=substr($html,stripos($html,'>Sexo'      )+($t+9),18);
145     $se=substr($se,0,stripos($se,'<'      ));
146
147     $nw=substr($html,stripos($html,'>Nacionalidad'      )+($t+16),18);
148     $nw=substr($nw,0,stripos($nw,'<'      ));
149
150     $ed=substr($html,stripos($html,'>Entidad de Nacimiento'      )+($t+25),28);
151     $ed=substr($ed,0,stripos($ed,'<'      ));
152
153     $dc=substr($html,stripos($html,'>Tipo Doc. Probatorio'      )+($t+25),28);
154     $dc=substr($dc,0,stripos($dc,'<'      ));
155
156     $if=substr($html,stripos($html,'<table'      ),strlen($html)-
157     stripos($html,'</b></td>
158         </tr>
159             </table>'      )+40);
160     $if=str_replace("\r"      , "      ,$if);
161     $if=str_replace("\n"      , "      ,$if);
162     $if=str_replace("\t"      , "      ,$if);
163
164     $cp=substr($html,stripos($html,'>Historicas'      )+($t+15),18);
165     $array=
166     array(
167         array(
168             "curp"      =>      $cu,
169             "paterno"      =>      $ap,
170             "materno"      =>      $am,
171             "nombre"      =>      $na,
172             "nacimiento"      =>      $fn,
173             "sexo"      =>      $se,
174             "nacionalidad"      =>      $nw,
175             "entidad"      =>      $ed,
176             "documento"      =>      $dc,
177             "cuerpo"      =>      $cp,
178             "informacion"      =>      utf8_encode(trim($if))
179         );
180
181 if(!stripos($cu,'Curp')&&!      stripos($cu,'ink')&&!      stripos($cu,"<"      ))
182 {
183     if($regresar==1)
184         return $array;
185     else
186         echo json_encode($array);
187
188 }
189 }
190 /**
191 *
192 * Calcula la curp y el rfc con los datos proporcionados
193 *
194 * @param string $paterno Apellido paterno de la persona
195 * @param string $materno Apellido materno
196 * @param string $nombre Nombre o nombres
197 * @param int $dia Dia de nacimiento
198 * @param int $mes Mes de nacimiento
199 * @param int $year Año de nacimiento
200 * @param string $sexo Sexo
201 * @param string $estado Lugar de nacimiento
202 * @param string $regresar Tipo de retorno =1 return array !=1 json
203 *
204 * @return echo
205 */
206
207 public function
calcular_curp($paterno,$materno,$nombre,$dia,$mes,$year,$sexo,$estado,$regresar=" "
208 {

```

```

209     $estados=$this-> estados;
210     $ap=strtoupper($paterno);
211     $am=strtoupper($materno);
212     $na=strtoupper($nombre);
213
214     $d=$dia;
215     if($d< 10) $d= "0" .(int)$d;
216     $m=$mes;
217     if($m< 10) $m= "0" .(int)$m;
218
219     $y=$year;
220     $se=$sexo;
221     $se=strtoupper($se);
222     if($se=="HOMBRE" || $se=="MASCULINO" || $se=="M")
223         $se="H";
224     if($se=="MUJER" || $se=="FEMENINO" || $se=="F")
225         $se="M";
226     $estado=strtoupper($estado);
227
228     $edo=$estados[0][$estado];
229     if
230     ($ap!=""
231     && $am!=""
232     && $y!=""
233     && $na!=""
234     && $se!=""
235     && $d!=""
236     && $edo!=""
237     ;)
238
239     {
240         $ch = curl_init();
241         curl_setopt($ch, CURLOPT_URL, "http://www.conisia.com.mx/calcular.php");
242         curl_setopt($ch, CURLOPT_POST, 1);
243         curl_setopt($ch, CURLOPT_HEADER, 0);
244         curl_setopt($ch, CURLOPT_FOLLOWLOCATION, 1);
245         curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);
246         curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
247         curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
248         curl_setopt($ch,
249         CURLOPT_POSTFIELDS, "PATERNO=$ap&MATERO=$am&NOMBRES=$na&DIA=$d&MES=$m&ANIO=$y
250         &SEXO=$se&ESTADO=$edo");
251         curl_setopt($ch, CURLOPT_USERAGENT, "Mozilla/5.0 (Windows; U; Windows NT 6.0;
252         en-US; rv:1.9.0.6) Gecko/2009011913 Firefox/3.0.6 (.NET CLR 3.5.30729)");
253         $html = curl_exec($ch);
254         curl_close($ch);
255         //modificados los valores de offset para recortar los TAGS del DOM
256         //correspondientes a RFC y CURP (Podría cambiar con el paso del tiempo)
257         $cur=substr($html,strpos($html,'CURP      </span>')+160),18);
258         $rfc=substr($html,strpos($html,'RFC      </span>')+159),13);
259         $array=
260         array(
261             array(
262                 "curp" => $cur,
263                 "rfc" => $rfc,
264             );
265             if(strlen($cur)> 10&&
266                 strpos($cur,<" )
267             {
268                 if($regresar==1)
269                     return $array;
270                 else
271                     echo json_encode($array);
272             }
273         }
274
275         /**
276         * Calcula la curp y el rfc con los datos proporcionados
277         *
278         * @param string $paterno Apellido paterno de la persona
279         * @param string $materno Apellido materno
280         * @param string $nombre Nombre o nombres
281         * @param int $dia Dia de nacimiento
282         * @param int $mes Mes de nacimiento
283         * @param int $year Año de nacimiento
284         * @param string $sexo Sexo
285         * @param string $estado Lugar de nacimiento
286         * @param string $regresar Tipo de retorno =1 return array !=1 json
287         *
288         * @return echo
289         */
290     public function
291     calcularcurp($paterno,$materno,$nombre,$dia,$mes,$year,$sexo,$estado,$regresar=" ")
292     {

```

```

282     $estados=$this-> estados;
283     $ap=strtoupper($paterno);
284     $am=strtoupper($materno);
285     $na=strtoupper($nombre);
286
287     $d=$dia;
288     if($d< 10) $d="0" .(int)$d;
289     $m=$mes;
290     if($m< 10) $m="0" .(int)$m;
291
292     $y=$year;
293     $se=$sexo;
294     $se=strtoupper($se);
295     if($se=="HOMBRE" || $se=="MASCULINO" || $se=="M")
296         $se="H";
297     if($se=="MUJER" || $se=="FEMENINO" || $se=="F")
298         $se="M";
299     $estado=strtoupper($estado);
300
301     $edo=$estados[0][$estado];
302     if
303     ($ap!=""
304      && $m!=""
305      ;)
306     {
307         $ch = curl_init();
308         curl_setopt($ch, CURLOPT_URL, "http://losimpuestos.com.mx/rfc/calcular-
rfc.php");
309         curl_setopt($ch, CURLOPT_POST, 1);
310         curl_setopt($ch, CURLOPT_HEADER, 0);
311         curl_setopt($ch, CURLOPT_FOLLOWLOCATION, 1);
312         curl_setopt($ch, CURLOPT_SSL_VERIFYPeer, false);
313         curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
314         curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
315         curl_setopt($ch,
316         CURLOPT_POSTFIELDS,
317             paterno=$ap&materno= $am&nombre= $na&dia= $d&mes= $m&anno=
318             &sexo=$se&entidad= $edo );
319         curl_setopt($ch, CURLOPT_USERAGENT, "Mozilla/5.0 (Windows; U; Windows NT 6.0;
320             en-US; rv:1.9.0.6) Gecko/2009011913 Firefox/3.0.6 (.NET CLR 3.5.30729)" );
321         $html = curl_exec($ch);
322         curl_close($ch);
323
324         //modificados los valores de offset para recortar los TAGS del DOM
325         //correspondientes a RFC y CURP (Podría cambiar con el paso del tiempo)
326         $infoinicio=substr($html,stripos($html,'<table>'));
327         $info=substr($infoinicio,0,stripos($infoinicio,'</table>'));
328         $info = str_replace(' ', ' ', $info);
329
330         $rfc = substr($info, stripos($info,
331             '<strong>RFC</strong>'));
332         $cur = substr($info, stripos($info,
333             '<strong>CURP</strong>'));
334
335         $rfc = substr($rfc, 0,stripos($rfc,'</span></strong>'));
336         $cur = substr($cur, 0,stripos($cur,'</span></strong>'));
337
338         $rfc = preg_replace('/[^ A-Za-z0-9_-ñÑ]/', ' ', $rfc);
339         $cur = preg_replace('/[^ A-Za-z0-9_-ñÑ]/', ' ', $cur);
340
341         $replaces = array(
342             'strong' => '',
343             'td' => '',
344             'RFC'=> '',
345             'CURP'=> '',
346             'span' => '',
347             'style' => '',
348             'color'=> '',
349             'foo' => ''
350         );
351
352         $rfc = str_replace(array_keys($replaces),array_values($replaces), $rfc);
353         $cur = str_replace(array_keys($replaces),array_values($replaces), $cur);
354
355         $array=
356         array(
357             array(
358                 "curp" => $cur,
359                 "rfc" => $rfc,
360             )
361         );

```

```
353     if(strlen($cur)> 10&&! strip_pos($cur, "<"))
354     {
355         if($regresar==1)
356             return $array;
357         else
358             echo json_encode($array);
359     }
360 }
361 }
362 ?>
363 }
364
```

# Archivo fuente para tree.php

La documentación para este archivo está disponible en [tree.php](#)

```
1  <?php
2  /**
3   * Controlador Objeto
4   *
5   * @package      Libreria
6   * @subpackage   Controlador
7   * @author       Eliecer
8   * @created      2013-12-10
9   */
10 if ( ! defined('BASEPATH')) exit('No direct script access allowed');
11 class Tree extends CI_Controller
12 {
13     public function __construct()
14     {
15         parent::__construct();
16         $this->load->helper('url');
17     }
18     /**
19      *
20      * Crea el arbol y lo muestra en la view
21      *
22      * @param string $title           Titulo de la pagina en el navegador
23      * @param string $titulo          titulo o nombre a mostrar en la vista al
24      *                                crear el arbol
25      * @param int    $seleccion        tipo de seleccion 1=select. 2=multiselect.
26      * @param string $tipo            tipo de control radio o check
27      * @param boolean $menu           si se desea mostrar el menu
28      * @param string $id              id del campo oculto donde se guarda el id
29      * @param string $text            id del campo donde se muestra la
30      *                                descripcion del elemento seleccionado
31      * @param string $idarbol         id del arbol donde comenzara la creacion
32      * @param string $nivel           nivel en el que se empezara a mostrar
33      *                                informacion
34      * @param string $omitidos        nodos que no se deben mostrar en la vista
35      *                                al crear el arbol
36      * @param string $seleccionable   determina si un nodo se puede o no
37      *                                seleccionar
38      *
39      * @return void
40      *
41      * ejemplo en views :
42      <a href="/?php echo DIR_TES?>/Tree/tree/TES/Lugar de
43      Nacimiento/1/radio/0/id/text/1/1/<?php echo urlencode(json_encode(array(2,3,4,5)));?>/<?php
44      echo urlencode(json_encode(array(2)));?>" id="fba1"
45      class="cat">Seleccionar</a>
46      <input type='hidden' id='id'>
47      <input type='text' id='text'>
48      Trae Estados ->Municipio y solo deja seleccionar municipios
49      */
50     public function
51     create($title,$titulo,$seleccion,$tipo,$menu,$id,$text,$idarbol=1,$nivel=1,$omitidos=array(NULL),$sele
52     ccionable=" ")
53     {
54         $data["title"]           = $title;
55         $data["titulo"]          = str_replace("%20", " ", $titulo);
56         $data["seleccion"]        = $seleccion;
57         $data["tipo"]             = $tipo;
58         $data["menu"]             = $menu;
59         $data["id"]               = $id;
60         $data["text"]             = $text;
61         $data["idarbol"]          = $idarbol;
62         $data["nivel"]            = $nivel;
63         $data["omitidos"]         = json_decode(urldecode($omitidos));
64
65         if($seleccionable!="")
66     }
```

```
56     $sel=json_decode(urldecode($seleccionable));
57     else $sel="";
58     $data["seleccionables"]=$sel;
59     $this->load->view(DIR_TES.'/tree/tree',$data);
60   }
61 }
?>
```

## Paquete TES

# Archivo fuente para Session.php

La documentación para este archivo está disponible en [Session.php](#)

```
1  <?php      if ( ! defined('BASEPATH')) exit('No direct script access allowed');
2  /**
3   * CodeIgniter Native Session Library
4   *
5   * @package    Session
6   * @subpackage Libraries
7   * @category   Session
8   * @author     Bo-Yi Wu (appleboy) <appleboy.tw@gmail.com>
9   * @author     Marko Martinović <marko@techytalk.info>
10  */
11
12 class Session
13 {
14     protected $sess_namespace = '';
15     protected $sess_expiration = '';
16     protected $ci;
17     protected $store = array();
18     protected $flashdata_key = 'flash';
19
20     /**
21      * Constructor
22      *
23      * @access public
24      * @param array config preferences
25      *
26      * @return void
27      */
28     public function __construct($config = array())
29     {
30         $this-> ci = get_instance();
31
32         if ( ! isset($_SESSION)) {
33             session_start();
34         }
35         $this-> initialize($config);
36
37         // Delete 'old' flashdata (from last request)
38         $this-> _flashdata_sweep();
39
40         // Mark all new flashdata as old (data will be deleted before next request)
41         $this-> _flashdata_mark();
42     }
43
44     /**
45      * Initialize the configuration options
46      *
47      * @access private
48      * @param array config options
49      * @return void
50      */
51     private function initialize($config)
52     {
53         $this-> ci-> load-> config('session');
54         $config = array_merge(
55             (
56                 array
57                 (
58                     'sess_namespace' => $this-> ci-> config-> item('sess_namespace'),
59                     'sess_expiration' => $this-> ci-> config-> item('sess_expiration')
60                 ),
61                 $config
62             );
63             foreach ($config as $key => $val) {
64                 if (method_exists($this, 'set_'.$key)) {
65                     $this->{ 'set_'.$key}($val);
66                 elseif (isset($this-> $key)) {
67                     $this-> $key = $val;
```

```

68         }
69     }
70     if (isset($_SESSION[$this-> sess_namespace])) {
71         $this-> store = $_SESSION[$this-> sess_namespace];
72         if (! $this-> is_expired()) {
73             return;
74         }
75     }
76     $this-> sess_create();
77 }
78 /**
79 * Create Session
80 *
81 * @access public
82 * @return void
83 */
84 public function sess_create()
85 {
86     // Set the session length. If the session expiration is
87     // set to zero we'll set the expiration two years from now.
88     if ($this-> sess_expiration == 0) {
89         $this-> sess_expiration = (60*60*24*365*2);
90     }
91     $expire_time = time() + intval($this-> sess_expiration);
92     $_SESSION[$this-> sess_namespace] = array(
93         'session_id' => md5(microtime()),
94         'expire_at' => $expire_time
95     );
96     $this-> store = $_SESSION[$this-> sess_namespace];
97 }
98 }
99 /**
100 * Check if session is expired
101 *
102 * @access public
103 * @return void
104 */
105 public function is_expired()
106 {
107     if ( ! isset($this-> store['expire_at'])) {
108         return TRUE;
109     }
110     return (time() > $this-> store['expire_at']);
111 }
112 }
113 /**
114 * Destroy session
115 *
116 * @access public
117 */
118 public function sess_destroy()
119 {
120     $this-> sess_create();
121 }
122 }
123 /**
124 * Get specific user data element
125 *
126 * @access public
127 * @param string element key
128 * @return object element value
129 */
130 public function userdata($value)
131 {
132     if ($value == 'session_id') {
133         return $this-> store['session_id'];
134     }
135     if (isset($this-> store[$value])) {
136         return $this-> store[$value];
137     } else {
138         return FALSE;
139     }
140 }
141 }
142 /**
143 * Set value for specific user data element
144 *
145 * @access public
146 * @param array list of data to be stored

```

```

148 * @param object value to be stored if only one element is passed
149 * @return void
150 */
151 public function set_userdata($data = array(), $value = '')
152 {
153     if (is_string($data)) {
154         $data = array($data => $value);
155     }
156     foreach ($data as $key => $val) {
157         $this-> store[$key] = $val;
158     }
159     $_SESSION[$this-> sess_namespace] = $this-> store;
160 }
161 /**
162 * remove array value for specific user data element
163 *
164 * @access public
165 * @param array list of data to be removed
166 * @return void
167 */
168 public function unset_userdata($data = array())
169 {
170     if (is_string($data)) {
171         $data = array($data => '');
172     }
173
174     if (count($data) > 0) {
175         foreach ($data as $key => $val) {
176             unset($this-> store[$key]);
177         }
178     }
179
180     $_SESSION[$this-> sess_namespace] = $this-> store;
181 }
182
183 /**
184 * Fetch all session data
185 *
186 * @access public
187 * @return array
188 */
189 public function all_userdata()
190 {
191     return $this-> store;
192 }
193
194 /**
195 * Add or change flashdata, only available
196 * until the next request
197 *
198 * @access public
199 * @param mixed
200 * @param string
201 * @return void
202 */
203 public function set_flashdata($newdata = array(), $newval = '')
204 {
205     if (is_string($newdata)) {
206         $newdata = array($newdata => $newval);
207     }
208
209     if (count($newdata) > 0) {
210         foreach ($newdata as $key => $val) {
211             $flashdata_key = $this-> flashdata_key.'::new:'.$key;
212             $this-> set_userdata($flashdata_key, $val);
213         }
214     }
215 }
216
217 /**
218 * Keeps existing flashdata available to next request.
219 *
220 * @access public
221 * @param string
222 * @return void
223 */
224 public function keep_flashdata($key)
225 {
226     // 'old' flashdata gets removed. Here we mark all

```

```

228     // flashdata as 'new' to preserve it from _flashdata_sweep()
229     // Note the function will return FALSE if the $key
230     // provided cannot be found
231     $old_flashdata_key = $this->  flashdata_key.':old:'.$key;
232     $value = $this->  userdata($old_flashdata_key);
233
234     $new_flashdata_key = $this->  flashdata_key.':new:'.$key;
235     $this->  set_userdata($new_flashdata_key, $value);
236 }
237
238 /**
239 * Fetch a specific flashdata item from the session array
240 *
241 * @access public
242 * @param string
243 * @return string
244 */
245 public function flashdata($key)
246 {
247     $flashdata_key = $this->  flashdata_key.':old:'.$key;
248     return $this->  userdata($flashdata_key);
249 }
250
251 /**
252 * Identifies flashdata as 'old' for removal
253 * when _flashdata_sweep() runs.
254 *
255 * @access private
256 * @return void
257 */
258 private function _flashdata_mark()
259 {
260     $userdata = $this->  all_userdata();
261     foreach ($userdata as $name => $value) {
262         $parts = explode(':new:', $name);
263         if (is_array($parts) && count($parts) === 2) {
264             $new_name = $this->  flashdata_key.':old:'.$parts[1];
265             $this->  set_userdata($new_name, $value);
266             $this->  unset_userdata($name);
267         }
268     }
269 }
270
271 /**
272 * Removes all flashdata marked as 'old'
273 *
274 * @access private
275 * @return void
276 */
277 private function _flashdata_sweep()
278 {
279     $userdata = $this->  all_userdata();
280     foreach ($userdata as $key => $value) {
281         if (strpos($key, ':old:')) {
282             $this->  unset_userdata($key);
283         }
284     }
285 }
286 }

```

## Paquete TES

# Archivo fuente para ayuda.php

La documentación para este archivo está disponible en [ayuda.php](#)

```
1  <?php
2  /**
3   * Controlador Ayuda
4   *
5   * @package    SIIGS
6   * @subpackage Controlador
7   * @author     Pascual
8   * @created    2013-09-26
9   */
10
11 class Ayuda extends CI_Controller {
12
13     public function __construct()
14     {
15         parent::__construct();
16     }
17
18     /**
19      * Función que renderiza el contenido de la ayuda dependiendo de la sección donde se encuentre
20      *
21      * @access public
22      * @param int $id_controlador_accion Id del controlador accion, es opcional
23      * @return void
24      */
25     public function index($id_controlador_accion=null)
26     {
27         $data['contenido_ayuda'] = 'No se encontró ayuda para esta sección';
28         $ruta = explode('/', str_replace('http://', '', $_SERVER['HTTP_REFERER']));
29
30         if(count($ruta)>= 3) {
31             $path = $ruta[1] . ':' . $ruta[2] . ':' . (isset($ruta[3]) ? $ruta[3] : 'index');
32
33             $this-> load-> model('siigs/ControladorAccion_model');
34             $idPath = $this-> ControladorAccion_model-> getIdByPath($path);
35
36             if(!empty($idPath)) {
37                 $registro = $this-> ControladorAccion_model-> getById( $idPath );
38
39                 if(!empty($registro-> ayuda))
40                     $data['contenido_ayuda'] = $registro-> ayuda;
41             }
42         }
43
44         $this-> template-> write('header','','true');
45         $this-> template-> write('menu','<h2 style="font-size: 25px;">
46              Ayuda</h2>',
48             ,true);
49         $this-> template-> write('sala_prensa','','true');
50         $this-> template-> write('seccion_ayuda','','true');
51         $this-> template-> write_view('content', 'ayuda', $data);
52         $this-> template-> render();
53     }
54 }
```

# Archivo fuente para accion.php

La documentación para este archivo está disponible en [accion.php](#)

```
1  <?php
2  /**
3   * Controlador Accion
4   *
5   * @package    SIIGS
6   * @subpackage Controlador
7   * @author     Geovanni
8   * @created    2013-09-26
9   */
10  class Accion extends CI_Controller {
11
12      public function __construct()
13      {
14          parent::__construct();
15
16          try
17          {
18              $this-> load-> helper('url');
19              $this-> load-> model(DIR_SIIGS.'/Accion_model');
20          }
21          catch (Exception $e)
22          {
23              $this-> template-> write("content" , $e-> getMessage());
24              $this-> template-> render();
25          }
26      }
27
28      /**
29       * Acción por default del controlador, carga la lista
30       * de acciones disponibles y una lista de opciones
31       * @param int $pag Número de registro para el paginador
32       *
33       * @return void
34       */
35      public function index($pag = 0)
36      {
37          if (empty($this-> Accion_model))
38              return false;
39              if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
40              show_error('', 403, 'Acceso denegado');
41
42          $this-> load-> library('pagination');
43          $this-> load-> helper('form');
44
45          //Configuracion para la paginacion
46          $configPag['base_url'] = '/'. DIR_SIIGS.'/accion/index/';
47          $configPag['first_link'] = 'Primero';
48          $configPag['last_link'] = '&Uacute;ltimo';
49          $configPag['total_rows'] = $this-> Accion_model-> getNumRows();
50          $configPag['uri_segment'] = '4';
51          $configPag['per_page'] = 20;
52
53          $this-> pagination-> initialize($configPag);
54          $this-> Accion_model-> setOffset($pag);
55          $this-> Accion_model-> setRows($configPag['per_page']);
56
57          try
58          {
59
60              $data['title'] = 'Lista de acciones disponibles';
61              $data['acciones'] = $this-> Accion_model-> getAll();
62              $data['msgResult'] = $this-> session->flashdata('msgResult');
63                  $data['clsResult'] = $this-> session->flashdata('clsResult');
64                  $data['pag'] = $pag;
65          }
66          catch (Exception $e)
67          {
```

```

68             $data['clsResult'] = "error";
69             $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
70         }
71
72         $this-> template-> write_view('content',DIR_SIIGS.'/accion/index', $data);
73         $this-> template-> render();
74     }
75
76 /**
77 *Acción para visualizar información de una acción específica, obtiene el objeto
78 *acción por medio del id proporcionado.
79 */
80 * @param int $id Este parametro no puede ser nulo
81 * @return void
82 */
83 public function view($id)
84 {
85     if (empty($this-> Accion_model))
86         return false;
87         if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
88     show_error('', 403, 'Acceso denegado');
89     try
90     {
91         $data['title'] = "Detalles de la acción";
92         $data['accion_item'] = $this-> Accion_model-> getById($id);
93     }
94     catch (Exception $e)
95     {
96         $data['clsResult'] = 'error';
97         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
98     }
99
100    $this-> template-> write_view('content',DIR_SIIGS.'/accion/view', $data);
101    $this-> template-> write('menu','','true');
102    $this-> template-> write('sala_prensa','','true');
103    $this-> template-> render();
104 }
105
106 /**
107 *Acción para preparar la inserción de nuevas acciones , realiza la validación
108 *del formulario del lado cliente
109 */
110 * @return void
111 */
112 public function insert()
113 {
114     if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
115     show_error('', 403, 'Acceso denegado');
116     $error = false;
117     $this-> load-> helper('form');
118     $this-> load-> library('form_validation');
119
120     $data['title'] = 'Crear una nueva acción';
121     $this-> form_validation-> set_rules('nombre', 'Nombre',
122     'trim|xss_clean|required|max_length[30]');
123     $this-> form_validation-> set_rules('descripcion', 'Descripción',
124     'trim|xss_clean|required|max_length[100]');
125     $this-> form_validation-> set_rules('metodo', 'Método',
126     'trim|xss_clean|required|max_length[30]');
127     // $this->form_validation->set_message('alpha','Los campos Nombre y Método
128     // solo pueden contener caracteres del alfabeto');
129
130     if ($this-> form_validation-> run() === FALSE)
131     {
132         $this-> template-> write_view('content',DIR_SIIGS.'/accion/insert',$data);
133         $this-> template-> render();
134     }
135     else
136     {
137         try
138         {
139             $this-> load-> helper('url');
140
141             $this-> Accion_model-> setNombre($this-> input-> post('nombre'));
142             $this-> Accion_model-> setDescripcion($this-> input-
143             > post('descripcion'));
144             $this-> Accion_model-> setMetodo($this-> input-> post('metodo'));
145             $this-> Accion_model-> insert();
146         }

```

```

143         catch (Exception $e)
144     {
145         $data['clsResult'] = 'error';
146         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
147         $this-> template-> write_view('content',DIR_SIIGS.'/accion/insert', $data);
148         $this-> template-> render();
149         $error = true;
150     }
151
152     if ($error == false)
153     {
154         $this-> session-> set_flashdata('msgResult', 'Registro insertado
correctamente');
155         $this-> session-> set_flashdata('clsResult', 'success');
156         redirect(DIR_SIIGS.'/accion/index','refresh');
157     }
158 }
159
160 /**
161 *Acción para preparar la actualización de una acción ya existente,
162 *recibe un ID para obtener los valores de esa acción y mostrarlos
163 *en la vista update , realiza la validación del formulario del lado
164 *del cliente
165 */
166
167 * @param int $id Este parámetro no puede ser nulo
168 * @return void
169 */
170 public function update($id)
171 {
172     if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
173     show_error('', 403, 'Acceso denegado');
174     $this-> load-> helper('form');
175     $this-> load-> helper('url');
176     $this-> load-> library('form_validation');
177
178     $error = false;
179
180     $data['title'] = 'Modificar acción';
181     $this-> form_validation-> set_rules('nombre', 'Nombre',
182     'trim|xss_clean|required|max_length[30]');
183     $this-> form_validation-> set_rules('descripcion', 'Descripción',
184     'trim|xss_clean|required|max_length[100]');
185     $this-> form_validation-> set_rules('metodo', 'Método',
186     'trim|xss_clean|required|max_length[30]');
187
188     if ($this-> form_validation-> run() === FALSE)
189     {
190         try
191         {
192             $data['accion_item'] = $this-> Accion_model-> getById($id);
193         }
194         catch (Exception $e)
195         {
196             $data['clsResult'] = 'error';
197             $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
198         }
199
200     else
201     {
202         try
203         {
204             $this-> Accion_model-> setNombre($this-> input-> post('nombre'));
205             $this-> Accion_model-> setDescripcion($this-> input-
206             > post('descripcion'));
207             $this-> Accion_model-> setMetodo($this-> input-> post('metodo'));
208             $this-> Accion_model-> setId($this-> input-> post('id'));
209
210             $this-> Accion_model-> update();
211         }
212         catch (Exception $e)
213         {
214             try
215             {
216                 $data['accion_item'] = $this-> Accion_model-> getById($id);
217             }
218             catch (Exception $e)

```

```

218
219         {
220             $data['clsResult'] = 'error';
221             $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
222         }
223             $data['clsResult'] = 'error';
224             $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
225             $this-> template-> write_view('content',DIR_SIIGS.'/accion/update', $data);
226             $this-> template-> render();
227
228         $error = true;
229     }
230
231     if ($error == false)
232     {
233         $this-> session-> set_flashdata('msgResult', 'Registro
actualizado correctamente');
234         $this-> session-> set_flashdata('clsResult', 'success');
235         redirect(DIR_SIIGS.'/accion','refresh');
236     }
237 }
238
239 /**
240 *
241 * Acción para eliminar una acción, recibe el id de la acción a eliminar
242 *
243 * @param int $id Este parámetro no puede ser nulo
244 * @return void
245 */
246 public function delete($id)
247 {
248     try
249     {
250         if (empty($this-> Accion_model))
251             return false;
252         if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__,
current_url()))
253             show_error('', 403, 'Acceso denegado');
254
255         $this-> load-> helper('url');
256         $this-> Accion_model-> setId($id);
257         $this-> Accion_model-> delete();
258         $this-> session-> set_flashdata('msgResult', 'Registro eliminado exitosamente');
259         $this-> session-> set_flashdata('clsResult', 'success');
260     }
261     catch(Exception $e)
262     {
263         $this-> session-> set_flashdata('clsResult', 'error');
264         $this-> session-> set_flashdata('msgResult', Errorlog_model::save($e-
> getMessage(), __METHOD__));
265     }
266     redirect(DIR_SIIGS.'/accion','refresh');
267 }
268 }
```

# Archivo fuente para bitacora.php

La documentación para este archivo está disponible en [bitacora.php](#)

```
1      <?php
2
3  /**
4   * Controlador Bitacora
5   *
6   * @package    SIIGS
7   * @subpackage Controlador
8   * @author     Pascual
9   * @created    2013-09-26
10  */
11 class Bitacora extends CI_Controller {
12
13     public function __construct()
14     {
15         parent::__construct();
16
17         if(!$this-> db-> conn_id) {
18             $this-> template-> write('content', 'Error no se puede conectar a la Base de
Datos');
19             $this-> template-> render();
20         }
21
22         $this-> load-> helper('url');
23     }
24
25 /**
26  * Lista todos los registros de la bitacora, con su correspondiente paginación
27  * permite hacer filtrados por Usuario, Entorno, Controlador, Acción y Fecha,
28  * permite eliminar un conjunto de registro o un elemento individual,
29  * muestra enlaces para actualizar y ver detalles de un elemento específico
30  *
31  * @access public
32  * @param int    $pag Establece el desplazamiento del primer registro a devolver
33  * @return void
34  */
35 public function index($pag = 0)
36 {
37     if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url())) {
38         show_error('', 403, 'Acceso denegado');
39         return false;
40     }
41
42     if(!isset($this-> Bitacora_model))
43         return false;
44
45     try {
46         $this-> load-> library('pagination');
47         $this-> load-> helper(array('form', 'formatFecha'));
48         $this-> load-> model( array(DIR_SIIGS.'/Usuario_model',
DIR_SIIGS.'/Entorno_model', DIR_SIIGS.'/Controlador_model', DIR_SIIGS.'/Accion_model') );
49
50         $filtros = array();
51         $data = array();
52
53         $data['pag'] = $pag;
54         $data['msgResult'] = $this-> session->flashdata('msgResult');
55         $data['clsResult'] = $this-> session->flashdata('clsResult');
56         $data['title'] = 'Bitacora';
57
58         /** Inicia Campos para Filtros ***/
59         $usuarios = $this-> Usuario_model-> getOnlyActives();
60         $data['usuarios'][0] = 'Todos';
61         foreach ($usuarios as $user) {
62             $data['usuarios'][$user-> id] = $user-> nombre . ' ' . $user-
> apellido_paterno . ' ' . $user-> apellido_materno;
63         }
64     }
```

```

65      $entornos = $this-> Entorno_model-> getAll();
66      $data['entornos'][0] = 'Todos';
67      foreach ($entornos as $sent) {
68          $data['entornos'][$sent-> id] = $sent-> nombre;
69      }
70
71      $data['controladores'][0] = 'Todos';
72
73      if($this-> input-> post('entorno')) {
74          $controladores = $this-> Controlador_model-> getByEntorno($this-> input-
75 > post('entorno'));
76          foreach ($controladores as $contr) {
77              $data['controladores'][$contr-> id] = $contr-> nombre;
78          }
79
80      $data['acciones'][0] = 'Todos';
81      $acciones = $this-> Accion_model-> getAll();
82      foreach ($acciones as $acci) {
83          $data['acciones'][$acci-> id] = $acci-> nombre;
84      }
85      /*** Fin Campos para Filtros ***/
86
87      $registroEliminar = $this-> input-> post('registroEliminar');
88
89      if( !empty($registroEliminar) ) {
90          $this-> Bitacora_model-> delete($registroEliminar);
91          $data['clsResult'] = 'success';
92          $data['msgResult'] = 'Registros Eliminados exitosamente';
93      }
94
95      if($this-> input-> post('filtrar')) {
96          // Eliminar el campo hidden y el boton
97          unset($_POST['filtrar'], $_POST['btnFiltrar']);
98          $filtros = array_filter($this-> input-> post());
99
100         if(!empty($filtros)) {
101             foreach ($filtros as $campo => $valor) {
102                 switch ($campo) {
103                     case 'usuario':
104                         $this-> Bitacora_model-> addFilter('id_usuario', '=',
$valor);
105                         break;
106                     case 'fechaIni':
107                         $this-> Bitacora_model-> addFilter('fecha_hora', '>='
formatFecha($valor, "Y-m-d");
108                         break;
109                     case 'fechaFin':
110                         $this-> Bitacora_model-> addFilter('fecha_hora', '<='
formatFecha($valor, "Y-m-d");
111                         break;
112                     case 'entorno':
113                         $this-> Bitacora_model-> addFilter('id_entorno', '=',
$valor);
114                         break;
115                     case 'controlador':
116                         $this-> Bitacora_model-> addFilter('id_controlador', '=',
$valor);
117                         break;
118                     case 'accion':
119                         $this-> Bitacora_model-> addFilter('id_accion', '=',
$valor);
120                         break;
121                 }
122             }
123         }
124
125         $data = array_merge($data, $filtros);
126     }
127     // Configuración para el Paginador
128     $configPag['base_url'] = site_url().DIR_SIIGS.'/bitacora/index/';
129     $configPag['first_link'] = 'Primero';
130     $configPag['last_link'] = '&Uacute;ltim' ;
131     $configPag['uri_segment'] = 4;
132     $configPag['total_rows'] = $this-> Bitacora_model-> getNumRows();
133     $configPag['per_page'] = 50;
134
135     $this-> pagination-> initialize($configPag);
136
137     $data['registros'] = $this-> Bitacora_model-> getAll($configPag['per_page'],

```

```

$pag);
138     } catch (Exception $e) {
139         $data['clsResult'] = 'error';
140         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
141     }
142
143     $this-> template-> write_view('content',DIR_SIIGS.'/bitacora/index', $data);
144     $this-> template-> render();
145 }
146
147 /**
148 * Muestra el formulario para crear un nuevo registro en la bitacora,
149 * las variables se obtienen por el metodo POST
150 *
151 * @access public
152 * @return void
153 */
154 /*public function insert()
155 {
156     if (!Usuario_model::checkCredentials(DIR_SIIGS.'::__METHOD__', current_url())) {
157         show_error('', 403, 'Acceso denegado');
158         return false;
159     }
160
161     if(!isset($this->Bitacora_model))
162         return false;
163
164     try {
165         $this->load->helper('form');
166         $this->load->model( array(DIR_SIIGS.'/Usuario_model',
DIR_SIIGS.'/Entorno_model', DIR_SIIGS.'/Controlador_model', DIR_SIIGS.'/Accion_model') );
167
168         $usuarios = $this->Usuario_model->getOnlyActives();
169         $data['usuarios'][0] = 'Elegir';
170         $data['title'] = 'Crear nuevo registro';
171         foreach ($usuarios as $user) {
172             $data['usuarios'][$user->id] = $user->nombre.' '.$user-
>apellido_paterno.' '.$user->apellido_materno;
173         }
174
175         $entornos = $this->Entorno_model->getAll();
176         $data['entornos'][0] = 'Elegir';
177         foreach ($entornos as $ent) {
178             $data['entornos'][$ent->id] = $ent->nombre;
179         }
180
181         $data['controladores'][0] = 'Elegir';
182
183         if($this->input->post('entorno')) {
184             $controladores = $this->Controlador_model->getByEntorno($this->input-
>post('entorno'));
185             foreach ($controladores as $contr) {
186                 $data['controladores'][$contr->id] = $contr->nombre;
187             }
188         }
189         $data['acciones'][0] = 'Elegir';
190         $acciones = $this->Accion_model->getAll();
191         foreach ($acciones as $acci) {
192             $data['acciones'][$acci->id] = $acci->nombre;
193         }
194
195         $datos = $this->input->post();
196         $data['title'] = 'Crear un nuevo registro';
197         $data['clsResult'] = $this->session->flashdata('clsResult');
198         $data['msgResult'] = $this->session->flashdata('msgResult');
199
200         if(!empty($datos)) {
201             $this->load->library('form_validation');
202
203             $this->form_validation->set_rules('usuario', 'Usuario',
'is_natural_no_zero|required|callback_validateExistUsuario');
204             $this->form_validation->set_rules('parametros', 'Parametros',
'trim|xss_clean|max_length[200]|required');
205             $this->form_validation->set_rules('controlador', 'Controlador',
'is_natural_no_zero|required');
206             $this->form_validation->set_rules('accion', 'Acción',
'is_natural_no_zero|required');
207
208             if ($this->form_validation->run() === true) {
209                 if(Bitacora_model::insert(DIR_SIIGS.'::__METHOD__', $datos['parametros']))

```

```

{
210         $this->session->set_flashdata('msgResult', 'Registro guardado
exitosamente');
211         redirect(DIR_SIIGS.'/bitacora/' , 'refresh');
212         die();
213     } else {
214         $data['clsResult'] = 'error';
215         $data['msgResult'] = 'Ocurrió un error al intentar guardar el registro';
216     }
217 }
218 }
219 } catch (Exception $e) {
220     $data['clsResult'] = 'error';
221     $data['msgResult'] = Errorlog_model::save($e->getMessage(), __METHOD__);
222 }
223
224 $this->template->write_view('content',DIR_SIIGS.'/bitacora/insert', $data);
225 $this->template->render();
226 */
227 /**
228 * Muestra el formulario con los datos del registro especificado por el id,
229 * para actualizar sus datos
230 *
231 * @access public
232 * @param int $id ID del elemento a actualizar
233 * @return void
234 */
235 /*public function update($id)
236 {
237     if (!Usuario_model::checkCredentials(DIR_SIIGS::__METHOD__, current_url())) {
238         show_error('', 403, 'Acceso denegado');
239         return false;
240     }
241
242     if(!isset($this->Bitacora_model))
243         return false;
244
245     try {
246         $this->load->helper('form');
247         $this->load->model(DIR_SIIGS.'/Usuario_model');
248
249         $usuarios = $this->Usuario_model->getOnlyActives();
250         $data['usuarios'][0] = 'Elegir';
251         foreach ($usuarios as $user) {
252             $data['usuarios'][$user->id] = $user->nombre.' '.$user-
>apellido_paterno.' '.$user->apellido_materno;
253         }
254
255         $data['title'] = 'Actualizar datos del registro';
256         $datos = $this->input->post();
257         $data['registro'] = $this->Bitacora_model->getById($id);
258
259         if( empty($data['registro']) )
260             $data['clsResult'] = 'error';
261             $data['msgResult'] = 'El registro solicitado no existe';
262
263         if(!empty($datos)) {
264             $this->load->library('form_validation');
265
266             $this->form_validation->set_rules('usuario', 'Usuario',
'is_natural_no_zero|required|callback_validateExistUsuario');
267             $this->form_validation->set_rules('parametros', 'Parametros',
'trim/xss_clean/max_length[200]|required');
268             $this->form_validation->set_rules('controlador', 'Controlador',
'is_natural_no_zero|required');
269             $this->form_validation->set_rules('accion', 'Acción',
'is_natural_no_zero|required');
270
271             $this->Bitacora_model->setId_usuario($datos['usuario']);
272             $this->Bitacora_model->setParametros($datos['parametros']);
273             $this->Bitacora_model->setId_controlador($datos['controlador']);
274             $this->Bitacora_model->setId_accion($datos['accion']);
275
276             if ($this->form_validation->run() === true) {
277                 $this->Bitacora_model->update($id);
278
279                 $this->session->set_flashdata('msgResult', 'Registro actualizado
correctamente');
280
281                 redirect(DIR_SIIGS.'/bitacora/' , 'refresh');
}

```

```

282             die();
283         }
284     }
285     } catch (Exception $e) {
286         $data['clsResult'] = 'error';
287         $data['msgResult'] = Errorlog_model::save($e->getMessage(), __METHOD__);
288     }
289
290     $this->template->write_view('content',DIR_SIIGS.'/bitacora/update', $data);
291     $this->template->render();
292 }/*
293 /**
294 * Muestra los datos del registro especificado por el id
295 *
296 * @access public
297 * @param int    $id ID del elemento a actualizar
298 * @return void
299 */
300
301 public function view($id)
302 {
303     if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url())) {
304         show_error('', 403, 'Acceso denegado');
305         return false;
306     }
307
308     if(!isset($this-> Bitacora_model))
309         return false;
310
311     try {
312         $data['registro'] = $this-> Bitacora_model-> getById($id);
313         $data['title'] = 'Datos del registro';
314
315         if( empty($data['registro']) ) {
316             $data['msgResult'] = 'ERROR: El registro solicitado no existe';
317             $data['clsResult'] = 'error';
318         }
319     } catch (Exception $e) {
320         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
321         $data['clsResult'] = 'error';
322     }
323
324     $this-> template-> write_view('content',DIR_SIIGS.'/bitacora/view', $data);
325     $this-> template-> write('menu','','true');
326     $this-> template-> write('sala_prensa','','true');
327     $this-> template-> render();
328 }
329
330 /**
331 * Elimina el registro especificado por el id
332 *
333 * @access public
334 * @param int    $id ID del elemento a eliminar
335 * @return void
336 */
337 /*public function delete($id)
338 {
339     if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url())) {
340         show_error('', 403, 'Acceso denegado');
341         return false;
342     }
343
344     if(!isset($this->Bitacora_model))
345         return false;
346
347     try {
348         $this->Bitacora_model->delete($id);
349         $this->session->set_flashdata('msgResult', 'Registro eliminado
exitosamente');
350         $this->session->set_flashdata('clsResult', 'success');
351     } catch (Exception $e) {
352         $this->session->set_flashdata('msgResult', Errorlog_model::save($e-
>getMessage(), __METHOD__));
353         $this->session->set_flashdata('clsResult', 'error');
354     }
355
356     redirect(DIR_SIIGS.'/bitacora/', 'refresh');
357     die();
358 }/*
359

```

```
360     /**
361      * callback utilizado por las acciones create y update para validar la existencia de un
362      *
363      * @access public
364      * @param int    $id_usuario ID del usuario
365      * @return void
366      */
367     public function validateExistUsuario($id_usuario)
368     {
369         $this-> load-> model(DIR_SIIGS.'/Usuario_model');
370         $result = $this-> Usuario_model-> getById($id_usuario);
371
372         if( empty($result) ) {
373             $this-> form_validation-> set_message('validateExistUsuario', 'El usuario no
374 existe');
375             return false;
376         } else {
377             return true;
378         }
379     }
380 }
381 ?>
```

# Archivo fuente para catalogo.php

La documentación para este archivo está disponible en [catalogo.php](#)

```
1  <?php
2  /**
3   * Controlador Catalogo
4   *
5   * @package    SIIGS
6   * @subpackage Controlador
7   * @author     Geovanni
8   * @created    2013-10-07
9   */
10  class Catalogo extends CI_Controller {
11
12      public function __construct()
13      {
14          parent::__construct();
15
16          try
17          {
18              $this-> load-> helper('url');
19              $this-> load-> model(DIR_SIIGS.'/Catalogo_model');
20          }
21          catch (Exception $e)
22          {
23              $this-> template-> write("content" , $e-> getMessage());
24              $this-> template-> render();
25          }
26      }
27
28      /**
29       * Acción por default del controlador, carga la lista
30       * de catálogos disponibles y una lista de opciones
31       * No recibe parámetros
32       *
33       * @return void
34       */
35      public function index()
36      {
37          if (empty($this-> Catalogo_model))
38              return false;
39              if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
40                  show_error('', 403, 'Acceso denegado');
41          try
42          {
43
44              $data['title'] = 'Lista de catálogos disponibles';
45              $data['catalogos'] = $this-> Catalogo_model-> getAll();
46              $data['msgResult'] = $this-> session-> flashdata('msgResult');
47              $data['clsResult'] = $this-> session-> flashdata('clsResult');
48          }
49          catch (Exception $e)
50          {
51              $data['clsResult'] = "error" ;
52              $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
53          }
54
55          $this-> template-> write_view('content',DIR_SIIGS.'/catalogo/index' , $data);
56
57          $this-> template-> render();
58      }
59
60      /**
61       * Acción para visualizar información de un catálogo específico, obtiene el objeto
62       * catálogo por medio del nombre proporcionado
63       *
64       * @param string $nombre Este parámetro no puede ser nulo
65       * @param int $pag Número de registro para el paginador
66       * @return void
67       */
68  }
```

```

68     public function view($nombre, $pag = 0)
69     {
70         if (empty($this-> Catalogo_model))
71             return false;
72
73         if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
74             show_error(' ', 403, 'Acceso denegado');
75
76         if ($this-> input-> is_ajax_request())
77         {
78             try
79             {
80                 $data['catalogo_item'] = $this-> Catalogo_model-> getByName($nombre);
81                 echo json_encode($data['catalogo_item']);
82
83             } catch (Exception $e)
84             {
85                 echo 'false';
86             }
87         exit;
88     }
89
90
91         $this-> load-> library('pagination');
92         $this-> load-> helper('form');
93
94         //Configuracion para la paginacion
95         $configPag['base_url'] = '/' . DIR_SIIGS.'/catalogo/view/' . $nombre . '/';
96         $configPag['first_link'] = 'Primero';
97         $configPag['last_link'] = '&Uacute;ltimo' ;
98         $configPag['total_rows'] = $this-> Catalogo_model-> getNumRows($nombre);
99         $configPag['uri_segment'] = '5';
100        $configPag['per_page'] = 50;
101
102        $this-> pagination-> initialize($configPag);
103        $this-> Catalogo_model-> setOffset($pag);
104        $this-> Catalogo_model-> setRows($configPag['per_page']);
105
106        try
107        {
108            $data['title'] = "Detalles del catálogo" ;
109            $data['catalogo_item'] = $this-> Catalogo_model-> getByName($nombre);
110            $data['datos_cat'] = $this-> Catalogo_model-> getAllData($nombre);
111        }
112        catch (Exception $e)
113        {
114            $data['clsResult'] = "error" ;
115            $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
116        }
117
118        $this-> template-> write_view('content',DIR_SIIGS.'/catalogo/view', $data);
119        $this-> template-> render();
120    }
121
122 /**
123 *Acción para cargar datos desde un archivo CSV, recibe el stream desde las variables PHP
124 *Guarda en la tabla tmp_catalogos toda la estructura del CSV e imprime las columnas del
125 *archivo. Solo se permite su acceso por medio de peticiones AJAX
126 */
127     * @param $_FILES[] archivocsv Variable pasada por POST con el archivo csv para cargar
128     * @return void
129     */
130     public function load()
131     {
132
133         if (!$this-> input-> is_ajax_request())
134             show_error(' ', 403, 'Acceso denegado');
135
136         if (isset($_FILES["archivocsv"])) &&
137         is_uploaded_file($_FILES['archivocsv']['tmp_name']))
138         {
139             $fp = fopen($_FILES['archivocsv']['tmp_name'], "r" );
140             // $fp = fopen('catalogos/estados.csv', "r");
141             $cont = 0;
142             $columnas = array();
143             $resultado = array();
144             $rows = array();
145

```

```

146 $consulta = 'select id_cat_tipo_columna as id,descripcion as descripcion from
147 asu_tipo_columna';
148 $dbtiposdatos = $this-> db-> query($consulta);
149 $tiposdatos = array();
150 foreach ($dbtiposdatos-> result() as $item)
151 {
152     array_push($tiposdatos, array('clave' => $item-> id , 'valor' => $item-
153 > descripcion));
154 }
155 //while (!feof($fp))
156 while (($data = fgetcsv($fp)) !== FALSE)
157 {
158     $utf8_encode = function($val)
159     {
160         return utf8_encode($val);
161     };
162     $data = array_map($utf8_encode,$data);
163     $cont +=1;
164     $data = preg_replace("!\r?\n!" , " " , $data);
165     if ($cont == 1)
166     {
167         $columnas = $data;
168         //Revisar si hay columnas con el mismo nombre
169         if (count($columnas) <>
170 count(array_unique($columnas)))
171         {
172             echo json_encode(array("Error" , "El
173 archivo contiene columnas con el mismo nombre"));
174             return;
175         }
176         if (in_array('id' , $columnas))
177         {
178             echo json_encode(array("Error" , "No
179 puede usar el nombre de columna 'id', por favor intente con id_{nombre del catalogo}"));
180             return;
181         }
182         //elimina la tabla temporal para crear catalogos
183         $consulta = 'drop table if exists tmp_catalogo;';
184         $query = $this-> db-> query($consulta);
185         //crea la tabla temporal para catalogos con la estructura del CSV
186         $consulta = 'create table if not exists tmp_catalogo (' ;
187         foreach ($columnas as $col)
188         {
189             array_push($resultado, array('columnName' => $col , 'tiposData'-
190 => $tiposdatos));
191             $consulta .= $col.' varchar(50),';
192         }
193         $consulta = substr($consulta, 0,count($consulta)-2);
194         $consulta .= ')';
195         $query = $this-> db-> query($consulta);
196         //enviar el resultado con el numero de columnas del csv
197         echo json_encode((object)$resultado);
198     }
199     else
200     {
201         //crea los rows con las filas que cumplen con la estructura en el CSV
202         if (count($columnas) == count($data))
203         {
204             $item = array_combine($columnas, $data);
205             array_push($rows, $item);
206         }
207     }
208     fclose($fp);
209     //Inserta los datos en lotes a la tabla temporal
210     $this-> db-> insert_batch('tmp_catalogo',$rows);
211 }
212 else
213 {
214     echo json_encode("Error:el archivo no se cargó correctamente");
215     return;
216 }
217 }
218 }
219

```

```

220
221     /**
222      *Acción para cargar datos desde un archivo CSV, recibe el stream desde las variables PHP
223      *compara los datos recibidos con los datos que contiene actualmente el catálogo, regresa
224      *resultado las filas nuevas y las filas a modificar.
225      * Solo se permite su acceso por medio de peticiones AJAX
226      *
227      * @param $_FILES[] archivocsv Variable pasada por POST con el archivo csv para cargar
228      * @return void
229      */
230     public function loadupdate($nombrecat , $update = false)
231     {
232
233         if (!$this-> input-> is_ajax_request())
234             show_error(' ', 403, 'Acceso denegado');
235
236         ini_set('max_execution_time',10000);
237
238         if (isset($_FILES["archivocsv"] ) &&
239             is_uploaded_file($_FILES['archivocsv']['tmp_name']))
240         {
241             $fp = fopen($_FILES['archivocsv']['tmp_name'], "r");
242             // $fp = fopen('catalogos/estados.csv', "r");
243             $cont = 0;
244             $columnas = array();
245             $resultado = array();
246             $rows = array();
247             $nuevos = 0; $modificados = 0; $iguales = 0; $errores = 0; $lineaserrores = array();
248
249             // obtiene los datos del catalogo
250             try
251             {
252                 $rowscat = $this-> Catalogo_model-> getAllData($nombrecat);
253                 if (count($rowscat)> 0)
254                 {
255                     // Si el catalogo contiene una columna ID
256                     // se hace el recorrido del arreglo para quitar esa columna
257                     if (array_key_exists('id', $rowscat[0]))
258                     {
259                         $temprowscat = array();
260                         foreach ($rowscat as $row)
261                         {
262                             $row = get_object_vars($row);
263                             $nuevorow = array();
264                             foreach (array_keys($row) as $rowkey)
265                             {
266                                 if ($rowkey != 'id')
267                                     $nuevorow[$rowkey] =
268                                     $row[$rowkey];
269
270                         }
271                         array_push($temprowscat, (object)$nuevorow);
272
273                     }
274                 }
275             }
276             catch (Exception $e)
277             {
278                 Errorlog_model::save($e-> getMessage(), __METHOD__);
279                 return;
280             }
281
282             try
283             {
284                 // array que contiene todos los registros llave
285                 $datallaves = array();
286                 // obtiene la estructura del catalogo
287                 $catalogo = $this-> Catalogo_model-> getByName($nombrecat);
288
289                 // obtiene los nombres de los campos con su tipo de dato y otros valores
290                 $campostemp = explode('|||', $catalogo-> campos);
291                 $llavestemp = explode('|||', $catalogo-> llave);
292
293                 // array para hacer modificaciones por lotes
294                 $consultamodificar = array();
295                 // array para hacer inserciones por lotes

```

```

296     $consultaagregar = array();
297
298     //filtro solo los nombres de los campos
299     $campos = array();
300     $llaves = array();
301     //obtiene el nombre de los campos y las llaves
302     foreach($campostemp as $item)
303         array_push($campos, explode(' | ', $item)[0]);
304
305     foreach($llavestemp as $item)
306     {
307         $key = explode(' | ', $item)[0];
308         if ($key != 'id')
309             array_push($llaves, $key);
310     }
311     if (count($llaves) == 0)
312     {
313         // se obtienen todos los campos que contengan el sufijo id_ y se toman
314         // como llaves primarias para el mapeo de datos repetidos en caso de que
315         // la llave primaria sea 'id' creada por el sistema
316         foreach($campostemp as $item)
317         {
318             $key = explode(' | ', $item)[0];
319             if (strpos($key, 'id_') !== false)
320                 array_push($llaves, $key);
321         }
322     }
323     else
324     {
325         $colstemp = $llaves;
326         foreach ($campos as $campo)
327         {
328             array_push($colstemp, $campo);
329         }
330         $campos = $colstemp;
331     }
332     //agrega las llaves como campos normales para obtener el numero total de
333     //columnas
334     //echo "select
335     ".implode(", ", $llaves)." from ".$nombrecat;
336     //obtiene las llaves primarias del catalogo
337     $rowsllaves = $this-> db-> query("select
338         , $llaves)." from "
339         . $nombrecat);
340     $rowsllaves = $rowsllaves-> result();
341     //var_dump($rowsllaves);
342     }
343     catch (Exception $e)
344     {
345         Errorlog_model::save($e-> getMessage(), __METHOD__);
346         die();
347     }
348     ini_set('memory_limit', '1024M');
349     $error = false;
350     //while (!feof($fp))
351     while (($data = fgetcsv($fp)) !== FALSE)
352     {
353         // $data = explode(", ", fgets($fp));
354
355         $utf8_encode = function($val)
356         {
357             return utf8_encode(addslashes($val));
358         };
359         $data = array_map($utf8_encode, $data);
360
361         $cont +=1;
362         $data = preg_replace("!\r?\n!", " ", $data);
363         if ($cont == 1)
364         {
365             $errorcols = false;
366             $columnas = $data;
367             if (count($campos) != count($columnas))
368                 $errorcols = true;
369             for ($i = 0; $i < count($campos); $i++)
370             {
371                 if (!isset($columnas[$i]) || !isset($campos[$i]) || $campos[$i] !=
$columnas[$i])
372                     $errorcols = true;
373             }
374             if ($errorcols)

```





```

510             {
511                 $this-> db-> trans_commit();
512                 echo json_encode(array("Ok" , "Los datos del catalogo se han
513 modifiedo correctamente" ));
514             }
515         }
516     }
517     else
518     {
519         echo json_encode(array("Error" , "El archivo no ha sido cargado
520 correctly." ));
521         die();
522     }
523 }
524
525 /**
526 * _array_unique_recursive
527 * Revisa valores duplicados en arreglos multidimensionales
528 *
529 * @param array $arr Arreglo multidimensional
530 */
531 public function _array_unique_recursive($arr)
532 {
533     foreach($arr as $key=> $value)
534         if(gettype($value)=='array')
535         {
536             $arr[$key]=implode("", $value);
537         }
538     return array_unique($arr,SORT_REGULAR);
539 }
540
541 /**
542 *Acción para preparar la inserción de nuevos catálogos , realiza la validación
543 *del formulario del lado del servidor y crea la estructura para el catálogo, crea
544 *la tabla y obtiene los datos a partir de la tabla tmp_catalogo
545 */
546 *@return void
547 */
548 public function insert()
549 {
550     if (!Usuario_model::checkCredentials(DIR_SIIGS::__METHOD__, current_url()))
551     show_error('' , 403 , 'Acceso denegado');
552
553     $error = false;
554     $this-> load-> helper('form');
555     $this-> load-> library('form_validation');
556
557     $data['title'] = 'Crear un nuevo catálogo';
558     $this-> form_validation-> set_rules('nombre' , 'Nombre',
559                                         'trim|xss_clean|required|alpha|max_length[30]');
560     $this-> form_validation-> set_rules('campos[]' , 'Campos',
561                                         'trim|xss_clean|required');
562
563     if ($this-> form_validation-> run() === FALSE)
564     {
565         $this-> template-> write_view('content' ,DIR_SIIGS.'/catalogo/insert' ,$data);
566         $this-> template-> render();
567     }
568     else
569     {
570         try
571         {
572             $this-> load-> helper('url');
573
574             $nombrep = $this-> input-> post('nombre');
575             $camposp = $this-> input-> post('campos');
576             $llavesp = $this-> input-> post('llaves');
577
578             $querycreate = '';
579
580             $queryselect = 'INSERT INTO cat_.'.$nombrep;
581             $querycreate = 'create table cat_.'.$nombrep. ' (' ;
582             $camposquery = '';
583             $campos = array();
584             $llaves = array();
585
586             //guarda el tamaño de la clave en caso de ser compuesta
587             $lenclave = 0;

```

```

586
587     foreach($campospost as $campo)
588     {
589         //valida si el campo fue asignado como llave primaria
590         if ($this-> input-> post('pk' . $campo) == 'on')
591         {
592             if ($llavespost == '1')
593             {
594                 //Si la llave primaria es de un campo entonces solo se agrega al
595                 //arreglo llave
596                 array_push($llaves, $campo);
597             }
598             else
599             {
600                 //se agrega a la lista de campos
601                 array_push($campos, $campo);
602                 //se agrega a la lista de llaves con LPAD para llenar dependiendo
603                 //el tamano del campo
604                 array_push($llaves, 'LPAD(' . $campo . ', ' . $this-> input-
605                             );
606                 //obtener el tamano final de la llave compuesta
607                 $lencclave += $this-> input-> post('len' . $campo);
608                 //se agrega a la lista de campos para el query con su tipo de dato
609                 //y tamano
610                 $camposquery .= $campo . ' ' . $this-> input-
611                             ;
612             }
613             else
614             {
615                 //se agrega a la lista de campos
616                 array_push($campos, $campo);
617                 //se agrega a la lista de campos para el query con su tipo de dato y
618                 //tamano
619                 $camposquery .= $campo . ' ' . $this-> input-
620                             ;
621             }
622             if (count($llaves) > 1)
623             {
624                 //si la llave es compuesta se crea el arreglo de llaves en tmpllave
625                 $tmpllave = implode(", ", $llaves);
626                 //se limpia el arreglo de llaves
627                 $llaves = array();
628                 //se asigna la llave compuesta al arreglo de llaves
629                 //array_push($llaves, 'concat('. $tmpllave.') as id');
630                 //se asigna la llave al query para crear la tabla
631                 $querycreate .= 'id int('. $lencclave .') primary key auto_increment,';
632             }
633             else
634             {
635                 foreach ($llaves as $key=> $value)
636                 {
637                     //se asigna la unica llave existente al query para crear la tabla
638                     $querycreate .= $value . ' ' . $this-> input-
639                             ;
640                     $camposquery = substr($camposquery, 0, count($camposquery)-2);
641                     $querycreate .= $camposquery . ') comment "' . $this-> input-
642                             . '"';
643                     $queryselect .= '('. (($llavespost == 1) ? implode(", ", $llaves) . ',' :
644                                     $campos) . ') select ';
645                     //se asignan las llaves al select
646                     $queryselect .= implode(", ", $llaves);
647                     //se asignan los campos al select
648                     $queryselect .= ((($llavespost==2) ? '' : ',')). implode(", ", $campos);
649                     $queryselect .= ' from tmp_catalogo';
650                     $this-> Catalogo_model-> insert($querycreate, $queryselect);
651                 }
652             }
653             catch (Exception $e)
654             {
655                 $data['clsResult'] = "error";

```

```

656             $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
657             $this-> template-> write_view('content',DIR_SIIGS.'/catalogo/insert',
658             );
659             $this-> template-> render();
660             $error = true;
661         }
662         if ($error == false)
663         {
664             $this-> session-> set_flashdata('clsResult', 'success');
665             $this-> session-> set_flashdata('msgResult', 'Registro insertado
correctamente');
666             redirect(DIR_SIIGS.'/catalogo/index','refresh');
667         }
668     }
669 }
670 */
671 /**
672 * Acción para preparar la actualización de un catálogo ya existente,
673 * recibe un string para obtener los valores del catálogo y mostrarlos
674 * en la vista update , realiza la validación del formulario del lado
675 * del cliente y servidor
676 *
677 * @param string $nombre
678 * @param int $pag Número de registro para el paginador
679 * @return void
680 */
681 public function update($nombre, $pag=0)
682 {
683     if (empty($this-> Catalogo_model))
684         return false;
685         if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
686 show_error('', 403, 'Acceso denegado');
687
688
689     $this-> load-> library('pagination');
690     $this-> load-> helper('form');
691
692     //Configuracion para la paginacion
693     $configPag['base_url'] = '/'. DIR_SIIGS.'/catalogo/update/'.$nombre '/';
694     $configPag['first_link'] = 'Primero';
695     $configPag['last_link'] = '&Uacute;ltimo' ;
696     $configPag['total_rows'] = $this-> Catalogo_model-> getNumRows($nombre);
697     $configPag['uri_segment'] = '5';
698     $configPag['per_page'] = 50;
699
700     $this-> pagination-> initialize($configPag);
701     $this-> Catalogo_model-> setOffset($pag);
702     $this-> Catalogo_model-> setRows($configPag['per_page']);
703
704     if ($this-> input-> post('comentario'))
705     {
706         try
707         {
708             $this-> Catalogo_model-> updateComentario($nombre, $this-
> input-> post('comentario'));
709             $data['clsResult'] = 'success';
710             $data['msgResult'] = 'Se modificó correctamente el comentario del
catalogo';
711         }
712         catch (Exception $e)
713         {
714             $data['clsResult'] = 'error';
715             $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
716         }
717     }
718
719     try
720     {
721         $data['title'] = "Modificar datos del catálogo" ;
722         $data['catalogo_item'] = $this-> Catalogo_model-> getByName($nombre);
723             $data['catalogo_item']-> nombre = $nombre;
724             $data['datos'] = $this-> Catalogo_model-> getAllData($nombre);
725     }
726     catch (Exception $e)
727     {
728             $data['clsResult'] = 'error';
729             $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
730     }
731

```

```

732     $this-> template-> write('ajustaAncho',1,true);
733     $this-> template-> write_view('content',DIR_SIIGS.'/catalogo/update', $data);
734     $this-> template-> render();
735 }
736 /**
737 *
738 * Acción para revisar registros repetidos en las columnas designadas como primary key
739 *
740 * @param string $campos
741 * @return void
742 */
743 public function checkPk($campos)
744 {
745     try
746     {
747         if (empty($this-> Catalogo_model))
748             return false;
749
750         $this-> load-> helper('url');
751         $result = $this-> Catalogo_model-> checkPk($campos);
752         if (count($result) == 0)
753             echo "true";
754         else
755             echo "false";
756
757     }
758     catch(Exception $e)
759     {
760         Errorlog_model::save($e-> getMessage(), __METHOD__);
761         echo "false";
762     }
763 }
764 /**
765 *
766 * Acción para revisar si los tipos de datos coinciden con los datos contenidos en
767 * la tabla temporal que fueron tomados del CSV
768 *
769 * @param string $campo Nombre del campo a revisar
770 * @param string $type Define el tipo de dato del campo
771 * @return void
772 */
773 public function checkTypeData($campo,$type)
774 {
775     try
776     {
777         if (empty($this-> Catalogo_model))
778             return false;
779
780         if (!$this-> input-> is_ajax_request())
781             show_error('', 403, 'Acceso denegado');
782
783         $result = $this-> Catalogo_model-> checkTypeData($campo,$type);
784
785         echo $result;
786
787     }
788     catch(Exception $e)
789     {
790         Errorlog_model::save($e-> getMessage(), __METHOD__);
791         echo "false";
792     }
793 }
794 /**
795 *
796 * Acción para eliminar un catálogo, recibe el nombre del catalogo a eliminar
797 *
798 * @param string $nombre
799 * @return void
800 */
801 public function delete($nombre)
802 {
803     try
804     {
805         if (empty($this-> Catalogo_model))
806             return false;
807
808
809
810
811

```

```

812
813             if (!Usuario_model::checkCredentials(DIR_SIIGS::__METHOD__,
814                                         current_url()))
815                 show_error('', 403, 'Acceso denegado');
816             $this-> load-> helper('url');
817
818             $existe = $this-> db-> query('select * fromasu_raiz_x_catalogo where
819                                         tabla_catalogo=' . $nombre . '');
820             if ($existe-> num_rows() > 0)
821             {
822                 $this-> session-> set_flashdata('msgResult', 'No se puede eliminar el catálogo
823                                         porque forma parte de un Arbol');
824                 $this-> session-> set_flashdata('clsResult', 'warning');
825                 redirect(DIR_SIIGS().'/catalogo','refresh');
826                 die();
827             }
828
829             $this-> Catalogo_model-> setNombre($nombre);
830             $this-> Catalogo_model-> delete();
831             $this-> session-> set_flashdata('msgResult', 'Catálogo eliminado exitosamente');
832             $this-> session-> set_flashdata('clsResult', 'success');
833         }
834     catch(Exception $e)
835     {
836         $this-> session-> set_flashdata('clsResult', 'error');
837         $this-> session-> set_flashdata('msgResult', Errorlog_model::save($e-
> getMessage(), __METHOD__));
838     }
839 }
840 }
```

# Archivo fuente para catalogocsv.php

La documentación para este archivo está disponible en [catalogocsv.php](#)

```
1  <?php
2  /**
3   * Controlador CatalogoCsv
4   *
5   * @package    SIIGS
6   * @subpackage Controlador
7   * @author     Geovanni
8   * @created    2013-12-06
9   */
10  class CatalogoCsv extends CI_Controller {
11
12      public function __construct()
13      {
14          parent::__construct();
15
16          try
17          {
18              $this-> load-> helper('url');
19              $this-> load-> model(DIR_SIIGS.'/CatalogoCsv_model');
20          }
21          catch (Exception $e)
22          {
23              $this-> template-> write("content" , $e-> getMessage());
24              $this-> template-> render();
25          }
26      }
27
28      /**
29       * Acción por default del controlador, carga la lista
30       * de catálogoscsv disponibles y una lista de opciones
31       * No recibe parámetros
32       *
33       * @return void
34       */
35      public function index()
36      {
37          if (empty($this-> CatalogoCsv_model))
38              return false;
39              if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
40              show_error('', 403, 'Acceso denegado');
41          try
42          {
43
44              $data['title'] = 'Lista de catálogos disponibles';
45              $data['catalogos'] = $this-> CatalogoCsv_model-> getAll();
46              $data['msgResult'] = $this-> session-> flashdata('msgResult');
47              $data['clsResult'] = $this-> session-> flashdata('clsResult');
48          }
49          catch (Exception $e)
50          {
51              $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
52              $data['clsResult'] = 'error';
53          }
54
55          $this-> template-> write_view('content',DIR_SIIGS.'/catalogocsv/index' , $data);
56
57          $this-> template-> render();
58      }
59
60      /**
61       * Acción para visualizar información de un catálogo específico, obtiene el objeto
62       * catalogocsv por medio del nombre proporcionado
63       *
64       * @param string $nombre Este parametro no puede ser nulo
65       * @param int $pag Número de registro para el paginador
66       * @return void
67       */
68  }
```

```

68     public function view($nombre, $pag = 0)
69     {
70         if (empty($this-> CatalogoCsv_model))
71             return false;
72
73         if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
74             show_error('', 403, 'Acceso denegado');
75
76
77         $this-> load-> library('pagination');
78         $this-> load-> helper('form');
79
80         //Configuracion para la paginacion
81         $configPag['base_url'] = '/'. DIR_SIIGS .'/catalogocsv/view/'.$nombre '/';
82         $configPag['first_link'] = 'Primero';
83         $configPag['last_link'] = '&Uacute;ltimo';
84         $configPag['total_rows'] = $this-> CatalogoCsv_model-> getNumRows($nombre);
85         $configPag['uri_segment'] = '5';
86         $configPag['per_page'] = 50;
87
88         $this-> pagination-> initialize($configPag);
89         $this-> CatalogoCsv_model-> setOffset($pag);
90         $this-> CatalogoCsv_model-> setRows($configPag['per_page']);
91
92         try
93         {
94             $data['title'] = "Detalles del catálogo";
95             $data['catalogo_item'] = $this-> CatalogoCsv_model-> getByName($nombre);
96             $data['datos_cat'] = $this-> CatalogoCsv_model-
97             > getAllData($nombre);
98         }
99         catch (Exception $e)
100        {
101            $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
102            $data['clsResult'] = 'error';
103        }
104
105        $this-> template-> write_view('content',DIR_SIIGS .'/catalogocsv/view', $data);
106        $this-> template-> render();
107    }
108
109    /**
110     *Acción para cargar datos desde un archivo CSV, recibe el stream desde las variables PHP
111     *compara los datos recibidos con los datos que contiene actualmente el catálogo, regresa
112     *resultado las filas nuevas y las filas a modificar
113     *Solo se permite su acceso por medio de peticiones AJAX
114     * @param $_FILES[] archivocsv Variable pasada por POST con el archivo csv para cargar
115     * @param string $nombrecat Nombre del catalogo a modificarle datos
116     * @param boolean $update Define si se actualizará la DB o solo se hará la primera
117     * revisión de datos
118     */
119    public function loadupdate($nombrecat , $update = false)
120    {
121        if (!$this-> input-> is_ajax_request())
122            show_error('', 403, 'Acceso denegado');
123
124            ini_set('max_execution_time',1000);
125
126        if (isset($_FILES["archivocsv"] ) &&
127        is_uploaded_file($_FILES['archivocsv']['tmp_name']))
128        {
129            $fp = fopen($_FILES['archivocsv']['tmp_name'], "r" );
130            // $fp = fopen('localidades_lat_lon_alt.csv', "r");
131            $cont = 0;
132            $columnas = array();
133            $resultado = array();
134            $rows = array();
135            $nuevos = 0; $modificados = 0; $iguales = 0;$errores = 0;
136
137            try
138            {
139                //array que contiene todos los registros llave
140                $datallaves = array();
141                //obtiene la estructura del catalogo
142                $catalogo = $this-> CatalogoCsv_model-> getByName($nombrecat);

```

```

143 //obtiene los nombres de los campos con su tipo de dato y otros valores
144
145 $campostemp = (strlen($catalogo-> campos)==0) ? array() : explode('||',
146 $catalogo-> campos);
147 $llavestemp = (strlen($catalogo-> llave)==0) ? array() :
148 explode('||',$catalogo-> llave);
149
150 //array para hacer modificaciones por lotes
151 $consultamodificar = array();
152 //array para hacer inserciones por lotes
153 $consultaagregar = array();
154
155 //filtro solo los nombres de los campos
156 $campos = array();
157 $llaves = array();
158 //obtiene el nombre de los campos y las llaves
159 if (count($campostemp)> 0)
160 foreach($campostemp as $item)
161 {
162     array_push($campos, explode('||',$item)[0]);
163     if (count($llavestemp)> 0)
164 foreach($llavestemp as $item)
165 {
166     $key = explode('||',$item)[0];
167     array_push($llaves, $key);
168 }
169
170 //agrega las llaves como campos normales para obtener el numero
171 //total de columnas
172
173 $colstemp = $llaves;
174 foreach ($campos as $campo)
175 {
176     array_push($colstemp, $campo);
177 }
178 $campos = $colstemp;
179
180 //obtiene las llaves primarias del catalogo
181 $rowsllaves = array();
182 if (count($llaves)> 0)
183 {
184     $rowsllaves = $this-> db-> query("select
185         , $llaves)." from ". $nombrecat);
186     $rowsllaves = $rowsllaves-> result();
187 }
188
189 //obtiene los datos excepto
190
191 $rowscat = array();
192 if (count($campos)> 0)
193 {
194     $rowscat = $this-> db-> query("select
195         , $campos)." from ". $nombrecat);
196     //echo "select
197     //var_dump($campos);
198     $rowscat = $rowscat-> result_array();
199 }
200
201 catch (Exception $e)
202 {
203     echo Errorlog_model::save($e-> getMessage(), __METHOD__);
204     die();
205 }
206 ini_set('memory_limit', '1024M');
207 $error = false;
208 while (($datatemp = fgetcsv($fp)) !== FALSE)
209 {
210     $utf8_encode = function($val)
211     {
212         return utf8_encode(addslashes($val));
213     };
214     // $datatemp = explode(", ", fgets($fp));
215     $data = array_map($utf8_encode,$datatemp);
216     $cont +=1;
217     $data = preg_replace("!\r?\n!" , " , $data);
218     if ($cont == 1)
219     {
220         $errorcols = false;
221         $columnas = $data;
222         if (count($campos) != count($columnas))
223             $errorcols = true;
224         for ($i = 0; $i < count($campos); $i++)
225     }

```

```

216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
$rowsllaves))
260
261
262
263
264
'.$nombrecat. ' set ';
265
266
267
268
269
270
'" . $value. " , ";
271
272
273
274
'" . $value. "" ;
275
276
277
$count($consultaupdate)-2);
278
279
280
281
1000)
282
283
as $sql)
284
> query($sql))
{
    if (!isset($columnas[$i]) || !isset($campos[$i]) || $campos[$i] !=
$columnas[$i])
        $errorcols = true;
    }
    if ($errorcols)
    {
        echo json_encode(array("Error" , "Las columnas del CSV
no coinciden con la estructura de la tabla" ));
        die();
    }
    else
    {
        $indiceellaves = array();
        //Obtiene los indices de columnas que corresponden a las llaves en el
CSV
        for ($i = 0; $i < count($columnas); $i++)
        {
            if (in_array($columnas[$i],$llaves))
            {
                array_push($indiceellaves,$i);
            }
        }
    }
    else
    {
        if (count($columnas) == count($data))
        {
            $datallave = array();
            foreach ($indiceellaves as $i)
            {
                array_push($datallave, $data[$i]);
            }
            //agrega el registro llave a la lista de llaves
            array_push($datallaves,$datallave);
            //agrega las claves con nombres de campo
            $procesada = array_combine($columnas,$data);
            //si el registro existe en el catalogo
            if (in_array($procesada, $rowscat))
            {
                $iguales += 1;
            }
            else
            {
                //si la clave existe en el catalogo
                if (in_array((object)array_combine($llaves,$datallave),
{
                    if ($update == true)
{
$consultaupdate = 'update
$consultaupdatewhere = ' where 1=1 ';
foreach ($procesada as $key => $value)
{
if (!in_array($key, $llaves))
{
$consultaupdate .= $key. " =
}
else
{
$consultaupdatewhere .= ' and ' . $key. " =
}
}
$consultaupdate = substr($consultaupdate,0,
$consultaupdate .= $consultaupdatewhere;
array_push($consultamodificar, $consultaupdate);
if (count($consultamodificar)==
{
foreach ($consultamodificar
if (!$this-> db-

```

```

285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353

    $error = true;
    $consultamodificar =
    }
}

    $modificados += 1;
}
else
{
    if ($update == true)
    {
        $arraytemp = array();
        foreach($campos as
        {
            $arraytemp[$valor] =
        }
        array_push($consultaagregar,
        if (count($consultaagregar) ==
        {
            if ($this-> db-
                $error = true;
                $consultaagregar = array();
        }
    }
}

    $nuevos += 1;
}
}
}
else {
    $errores +=1;
}
}
}
}
fclose($fp);
//var_dump($datallaves);
if (count($datallaves) > count($this-> array_unique_recursive($datallaves)))
{
    //echo count($datallaves) . " ". count($this-
>array_unique_recursive($datallaves));
    echo json_encode(array("Error" , "El archivo contiene llaves
primarias duplicadas" ));
    die();
}
else
{
    array_push($resultado,array('Numero de registros anteriores',count($rowscat)));
    array_push($resultado,array('Numero de registros actuales',$cont-1));
    array_push($resultado,array('Numero de registros a insertar',$nuevos));
    array_push($resultado,array('Numero de registros a modificar',$modificados));
    array_push($resultado,array('Numero de registros sin cambios',$iguales));
    array_push($resultado,array('Numero de registros con
errores' ,$errores));
}
if ($update == false)
    echo json_encode($resultado);
else
{
    if (count($consultaagregar)> 0)
    {
        if ($this-> db-
> insert_batch($nombrecat,$consultaagregar)==0)
            $error = true;
    }
    if (count($consultamodificar)> 0)
    {

        foreach ($consultamodificar as $sql)
            if (!$this-> db-> query($sql))
                $error = true;
    }
}
if ($error == true)
{
    // $this->db->trans_rollback();
}

```

```

354             echo json_encode(array("Error" , "Ha ocurrido un error al
hacer el volcado, los datos no se modificaron."));
355         }
356     }
357     {
358         // $this->db->trans_commit();
359         echo json_encode(array("Ok" , "Los datos del catalogo se han
modificado correctamente"));
360         $this-> db-> query("update cns_tabla_catalogo set
fecha_actualizacion = NOW() where descripcion='"
.$nombrecat."");
361     }
362     }
363     else
364     {
365         echo json_encode(array("Error" , "El archivo no ha sido cargado
correctamente."));
366         die();
367     }
368 }
369 }
370 /**
371 * _array_unique_recursive
372 * Revisa valores duplicados en arreglos que contienen arreglos
373 *
374 * @param array $arr
375 */
376 public function _array_unique_recursive($arr)
377 {
378     foreach($arr as $key=> $value)
379     {
380         if(gettype($value)=='array')
381         {
382             $arr[$key]=implode(", " , $value);
383         }
384         if (count($arr)> count(array_unique($arr)))
385         {
386             //var_dump(array_diff($arr, array_unique($arr)));
387             //echo count($arr) . " . " . count(array_unique($arr));
388         }
389     }
390     return array_unique($arr);
391 }
392 /**
393 * Accion para activar o desactivar elementos en los catalogos indicados en el parametro
394 * Solo se permite su acceso por medio de peticiones AJAX
395 *
396 * @param Int $id Es el id del registro en el catalogo
397 * @param String $catalogo para determinar a que catalogo se va a agregar o quitar el
registro
398 * @param Boolean $activo False para quitar del catalogo, true para agregarlo
399 *
400 * @return Boolean En caso de error, o errores de referencia, etc.
401 */
402 public function ActivaEnCatalogo(){
403
404     if (!$this-> input-> is_ajax_request())
405         show_error('' , 403 , 'Acceso denegado');
406
407     try
408     {
409         if ($this-> input-> is_ajax_request())
410     {
411         $id = $this-> input-> post('id');
412         $catalogo = $this-> input-> post('catalogo');
413         $activo = $this-> input-> post('activo');
414         // $id = 6110;
415         // $catalogo = "eda";
416         // $activo = true;
417         if ($id && $catalogo)
418         {
419             $resultado = $this-> CatalogoCsv_model-
> activaEnCatalogo($id,$catalogo,$activo);
420             if ($resultado == true)
421                 echo "ok" ;
422             else
423                 echo "error" ;
424         }
425         else
426             echo "Parametros incorrectos" ;
427     }

```

```

428     }
429     else echo 'Acceso denegado';
430     }
431     catch(Exception $e)
432     {
433     echo $e-> getMessage();
434     }
435     }
436     }
437 /**
438 * Acción para preparar la actualización de un catálogo ya existente,
439 * recibe un string para obtener los valores del catalogo y mostrarlos
440 * en la vista update , realiza la validacion del formulario del lado
441 * del cliente y servidor
442 *
443 * @param string $nombre
444 *      @param int $pag Número de registro para el paginador
445 * @return void
446 */
447 public function update($nombre, $pag = 0)
448 {
449     if (empty($this-> CatalogoCsv_model))
450         return false;
451         if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
452 show_error('', 403, 'Acceso denegado');
453
454     $this-> load-> library('pagination');
455     $this-> load-> helper('form');
456
457     //Configuracion para la paginacion
458     $configPag['base_url'] = '/'. DIR_SIIGS.'/catalogocsv/update/'. $nombre . '/';
459     $configPag['first_link'] = 'Primero';
460     $configPag['last_link'] = '&Uacute;ltimo';
461     $configPag['total_rows'] = $this-> CatalogoCsv_model-> getNumRows($nombre);
462     $configPag['uri_segment'] = '5';
463     $configPag['per_page'] = 50;
464
465     $this-> pagination-> initialize($configPag);
466     $this-> CatalogoCsv_model-> setOffset($pag);
467     $this-> CatalogoCsv_model-> setRows($configPag['per_page']);
468
469     try
470     {
471         $data['title'] = "Modificar datos del catálogo";
472         $data['catalogo_item'] = $this-> CatalogoCsv_model-> getByName($nombre);
473             $data['datos'] = $this-> CatalogoCsv_model-> getAllData($nombre);
474     }
475     catch (Exception $e)
476     {
477         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
478         $data['clsResult'] = 'error';
479     }
480
481 //        $this->template->write('ajustaAncho',1,true);
482 $this-> template-> write_view('content',DIR_SIIGS.'/catalogocsv/update', $data);
483 $this-> template-> render();
484
485 }
486
487 /**
488 *
489 * Acción para revisar registros repetidos en las columnas designadas como primary key
490 *
491 * @param string $campos
492 * @return void
493 */
494 public function checkpk($campos)
495 {
496     try
497     {
498         if (empty($this-> CatalogoCsv_model))
499             return false;
500
501         $this-> load-> helper('url');
502         $result = $this-> CatalogoCsv_model-> checkPk($campos);
503         if (count($result) == 0)
504             echo "true";
505         else
506             echo "false";
507

```

```

508     }
509     catch(Exception $e)
510     {
511         Errorlog_model::save($e-> getMessage(), __METHOD__);
512         echo "false";
513     }
514 }
515
516 /**
517 *Acción para ejecutar la creación de la tabla poblacional
518 *No recibe parámetros
519 *
520 *@return void
521 */
522 public function createTablePob()
523 {
524     $this-> load-> model(DIR_SIIGS.'/Poblacion_model');
525
526     if (empty($this-> Poblacion_model)) {
527         echo 'No se puede cargar el modelo Poblacion';
528         return false;
529     }
530     if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
531         show_error('', 403, 'Acceso denegado');
532     try
533     {
534         $this-> Poblacion_model-> process();
535
536         $this-> session-> set_flashdata('msgResult', 'Datos procesados correctamente');
537         $this-> session-> set_flashdata('clsResult', 'success');
538     }
539     catch (Exception $e)
540     {
541         $this-> session-> set_flashdata('msgResult', Errorlog_model::save($e-
> getMessage(), __METHOD__));
542         $this-> session-> set_flashdata('clsResult', 'error');
543     }
544
545     redirect(DIR_SIIGS.'/catalogocsv/', 'refresh');
546     die();
547 }
548
549 /**
550 *Acción para ejecutar la creación de la tabla poblacional
551 *No recibe parámetros
552 *
553 *@return void
554 */
555 public function createTableGeo()
556 {
557     $this-> load-> model(DIR_SIIGS.'/Georeferencia_model');
558
559     if (empty($this-> Georeferencia_model)) {
560         echo 'No se puede cargar el modelo Georeferencia';
561         return false;
562     }
563     if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
564         show_error('', 403, 'Acceso denegado');
565     try
566     {
567         $this-> Georeferencia_model-> process();
568
569         $this-> session-> set_flashdata('msgResult', 'Datos procesados correctamente');
570         $this-> session-> set_flashdata('clsResult', 'success');
571     }
572     catch (Exception $e) {
573         $this-> session-> set_flashdata('msgResult', Errorlog_model::save($e-
> getMessage(), __METHOD__));
574         $this-> session-> set_flashdata('clsResult', 'error');
575     }
576
577     redirect(DIR_SIIGS.'/catalogocsv/', 'refresh');
578     die();
579 }
580
581 /**
582 *Acción para ejecutar la creación de la tabla Asu Ageb
583 *No recibe parámetros
584 *
585 *@return void

```

```

586     */
587     public function createTableAgeb()
588     {
589         ini_set('max_execution_time',1000);
590
591         $this-> load-> model(DIR_SIIGS.'/Ageb_model');
592
593         if (empty($this-> Ageb_model)) {
594             echo 'No se puede cargar el modelo Ageb';
595             return false;
596         }
597         if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
598             show_error('', 403, 'Acceso denegado');
599
600         try
601         {
602             $this-> Ageb_model-> process();
603             $this-> session-> set_flashdata('msgResult', 'Datos procesados
correctamente');
604             $this-> session-> set_flashdata('clsResult', 'success');
605         }
606         catch (Exception $e)
607         {
608             $this-> session-> set_flashdata('msgResult', Errorlog_model::save($e-
> getMessage(), __METHOD__));
609             $this-> session-> set_flashdata('clsResult', 'error');
610         }
611
612         redirect(DIR_SIIGS.'/catalogocsv/', 'refresh');
613         //die();
614     }
615
616     /**
617      *Acción para ejecutar la creación de la tabla Asu Ageb
618      *No recibe parámetros
619      */
620     public function createTableHemoGlobina()
621     {
622         ini_set('max_execution_time',1000);
623
624         $this-> load-> model(DIR_SIIGS.'/Hemoglobina_model');
625
626         if (empty($this-> Hemoglobina_model)) {
627             echo 'No se puede cargar el modelo Ageb';
628             return false;
629         }
630         if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
631             show_error('', 403, 'Acceso denegado');
632
633         try
634         {
635             $this-> Hemoglobina_model-> process();
636             $this-> session-> set_flashdata('msgResult', 'Datos procesados
correctamente');
637             $this-> session-> set_flashdata('clsResult', 'success');
638         }
639         catch (Exception $e)
640         {
641             $this-> session-> set_flashdata('msgResult', Errorlog_model::save($e-
> getMessage(), __METHOD__));
642             $this-> session-> set_flashdata('clsResult', 'error');
643         }
644
645         redirect(DIR_SIIGS.'/catalogocsv/', 'refresh');
646         //die();
647     }

```

# Archivo fuente para catalogo\_x\_raiz.php

La documentación para este archivo está disponible en [catalogo\\_x\\_raiz.php](#)

```
1  <?php
2  /**
3   * Controlador Raiz_x_Catalogo
4   *
5   * @package    SIIGS
6   * @subpackage Controlador
7   * @author     Geovanni
8   * @created    2013-10-16
9   */
10  class Catalogo_x_raiz extends CI_Controller {
11
12      public function __construct()
13      {
14          parent::__construct();
15
16          try
17          {
18              $this-> load-> helper('url');
19              $this-> load-> model(DIR_SIIGS.'/Catalogo_x_raiz_model');
20              $this-> load-> model(DIR_SIIGS.'/Catalogo_model');
21          }
22          catch (Exception $e)
23          {
24              $this-> template-> write("content" , $e-> getMessage());
25              $this-> template-> render();
26          }
27      }
28
29      /**
30      *Acción para visualizar información de una raiz_x_catalogo específica, obtiene el objeto
31      *raiz_x_catalogo por medio del id proporcionado.
32      *
33      * @param int $id Este parametro no puede ser nulo
34      * @return void
35      */
36      public function view($id)
37      {
38          if (empty($this-> Catalogo_x_raiz_model))
39          {
40              if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
41                  show_error('', 403, 'Acceso denegado');
42          try
43          {
44              $data['title'] = "Detalles del raiz x catálogo";
45              $data['catalogo_item'] = $this-> Catalogo_x_raiz_model-> getById($id);
46          }
47          catch (Exception $e)
48          {
49              $data['clsResult']='error';
50              $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
51          }
52
53          $this-> template-> write_view('content',DIR_SIIGS.'/catalogo_x_raiz/view', $data);
54          $this-> template-> render();
55      }
56
57      /**
58      *Acción para preparar la inserción de nuevas raices para catálogos , realiza la validación
59      *del formulario del lado cliente
60      *
61      * @return void
62      */
63      public function insert($id = 0)
64      {
65          if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
66              show_error('', 403, 'Acceso denegado');
67      }
}
```

```

68     $error = false;
69     $this-> load-> helper('form');
70     $this-> load-> library('form_validation');
71
72     $data['title'] = 'Agregar un nuevo catálogo al arbol';
73     $this-> form_validation-> set_rules('tabla_catalogo', 'Catálogo',
74                                         'trim|xss_clean|required');
74     $this-> form_validation-> set_rules('columna_llave', 'Columna llave',
75                                         'trim|xss_clean|required');
75     $this-> form_validation-> set_rules('columnas_descripcion', 'Descripción',
76                                         'trim|xss_clean|required');
76     $this-> form_validation-> set_rules('grado', 'Grado de segmentación',
77                                         'trim|xss_clean|required');
77
78     if ($this-> form_validation-> run() === FALSE && $id > 0)
79     {
80         try
81         {
82             $data['nivel'] = $this-> Catalogo_x_raiz_model-> getNivel($id)-> nivel;
83
84             if ($data['nivel'] > 1)
85             {
86                 $catalogo_padre = $this-> Catalogo_x_raiz_model-
86                 > getByNivel($id,$data['nivel']-1)-> nombre;
87                 $data['catalogo_padre'] = $catalogo_padre;
88             }
89             else
90                 $data['catalogo_padre'] = '';
91
92             $catalogos = $this-> Catalogo_model-> getAll();
93             $data['catalogos']['''] = 'Todos';
94             foreach ($catalogos as $item) {
95                 $data['catalogos'][$item-> nombre] = $item-> nombre;
96             }
97
98             $data['id_raiz'] = $id;
99         }
100        catch (Exception $e)
101        {
102            $data['clsResult']='error';
103            $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
104        }
105
106        $this-> template-
106        > write_view('content',DIR_SIIGS.'/catalogo_x_raiz/insert',$data);
107        $this-> template-> render();
108    }
109    else if ($this-> form_validation-> run() === TRUE)
110    {
111        try
112        {
113            //var_dump($this->input->post());
114            $this-> load-> helper('url');
115
116            $this-> Catalogo_x_raiz_model-> setGrado($this-> input-> post('grado'));
117            $this-> Catalogo_x_raiz_model-> setTablaCatalogo($this-> input-
117             > post('tabla_catalogo'));
118            $this-> Catalogo_x_raiz_model-> setColumnaLlave($this-> input-
118             > post('columna_llave'));
119            $this-> Catalogo_x_raiz_model-> setColumnaDescripcion($this-> input-
119             > post('columnas_descripcion'));
120            $this-> Catalogo_x_raiz_model-> setIdRaiz($this-> input-
120             > post('id_raiz'));
121
122            $relaciones = $this-> input-> post('relaciones');
123
124            $relacionpadre = array();
125            $relacionhijo = array();
126            for ($i = 1;$i<= $relaciones;$i++)
127            {
128                array_push($relacionpadre, $this-> input-> post('relacionpadre'.'$i'));
129                array_push($relacionhijo, $this-> input-> post('relacionhijo'.'$i'));
130            }
131            //var_dump($relacionpadre);
132            //var_dump($relacionhijo);
133
134            $this-> Catalogo_x_raiz_model-> setRelacionPadre($relacionpadre);
135            $this-> Catalogo_x_raiz_model-> setRelacionHijo($relacionhijo);
136
137

```

```

138         //die();
139         $this-> Catalogo_x_raiz_model-> insert();
140     }
141     catch (Exception $e)
142     {
143         $data['clsResult'] = 'error';
144         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
145         //this->template-
146         >write_view('content', DIR_SIIGS.'/catalogo_x_raiz/insert', $data);
147         //this->template->render();
148         $error = true;
149         //var_dump(Errorlog_model::save($e->getMessage(), __METHOD__));
150         //die();
151         $this-> session-> set_flashdata('clsResult', 'error');
152         $this-> session-> set_flashdata('msgResult', Errorlog_model::save($e-
153         > getMessage(), __METHOD__));
154         redirect(DIR_SIIGS.'/raiz/update/'. $this-> input-
155         > post('id_raiz'), 'refresh');
156     }
157     if ($error == false)
158     {
159         $this-> session-> set_flashdata('clsResult', 'success');
160         $this-> session-> set_flashdata('msgResult', 'Registro insertado
correctamente');
161         redirect(DIR_SIIGS.'/raiz/update/'. $this-> input-
162         > post('id_raiz'), 'refresh');
163     }
164 }
165 /**
166 * Acción que sirve para revisar inconsistencias en el arbol de segmentación
167 * Recibe como parámetro el catálogo x raiz y revisa que todos los registros tengan
168 * un correspondiente en el catálogo padre.
169 * Solo se permite su acceso por medio de peticiones AJAX
170 *
171 * @param type $id Id del catalogo_x_raiz
172 * @return boolean
173 */
174 public function check($id)
175 {
176     if (!$this-> input-> is_ajax_request())
177         show_error('', 403, 'Acceso denegado');
178     try
179     {
180         $this-> load-> helper('form');
181         if ($this-> input-> is_ajax_request())
182         {
183             $nivel = $this-> Catalogo_x_raiz_model-> getById($id);
184             if (empty($nivel))
185             {
186                 echo json_encode(array('false'));
187                 return;
188             }
189             if ($nivel-> grado_segmentacion == 1)
190             {
191                 echo json_encode(array('true'));
192                 return;
193             }
194             else
195             {
196                 $inconsistencias = $this-> Catalogo_x_raiz_model-> check($id);
197                 if (count($inconsistencias) == 0)
198                 {
199                     echo json_encode(array('true'));
200                     return;
201                 }
202                 else
203                 {
204                     echo json_encode($inconsistencias);
205                 }
206             }
207         }
208     }
209     catch (Exception $e)
210 }

```

```

213     {
214         // $data['msgResult'] = Errorlog_model::save($e->getMessage(), __METHOD__);
215         echo 'false' . $data['msgResult'];
216     }
217 }
218 /**
219 *
220 * Acción para eliminar un catálogo en el arbol, recibe el id del catálogo en la raiz a
221 * eliminar
222 *
223 * @param int $id
224 * @return void
225 */
226 public function delete($id)
227 {
228     if (!Usuario_model::checkCredentials(DIR_SIIGS::__METHOD__, current_url()))
229         show_error('', 403, 'Acceso denegado');
230
231     try
232     {
233         if (empty($this-> Catalogo_x_raiz_model))
234             return false;
235
236         $element = $this-> Catalogo_x_raiz_model-> getById($id);
237         $id_raiz = $element-> id_raiz_arbol;
238         //var_dump($id_raiz);
239         //die();
240         $this-> load-> helper('url');
241         $this-> Catalogo_x_raiz_model-> setId($id);
242         $this-> Catalogo_x_raiz_model-> setGrado(
243             grado_segmentacion);
244         $this-> Catalogo_x_raiz_model-> setIdRaiz($id_raiz);
245         $this-> Catalogo_x_raiz_model-> delete();
246         $this-> session-> set_flashdata('clsResult', 'success');
247         $this-> session-> set_flashdata('msgResult', 'Catálogo eliminado
248 exitosamente');
249     }
250     catch(Exception $e)
251     {
252         $this-> session-> set_flashdata('msgResult', Errorlog_model::save($e-
253 getMessage(), __METHOD__));
254         $this-> session-> set_flashdata('clsResult', 'error');
255     }
256 }

```

# Archivo fuente para cie10.php

La documentación para este archivo está disponible en [cie10.php](#)

```
1  <?php
2  /**
3   * Controlador Cie10
4   *
5   * @package    SIIGS
6   * @subpackage Controlador
7   * @author     Geovanni
8   * @created    2013-12-02
9   */
10  class Cie10 extends CI_Controller {
11
12      public function __construct()
13      {
14          parent::__construct();
15
16          try
17          {
18              $this-> load-> helper('url');
19              $this-> load-> model(DIR_SIIGS.'/Cie10_model');
20          }
21          catch (Exception $e)
22          {
23              $this-> template-> write("content" , $e-> getMessage());
24              $this-> template-> render();
25          }
26      }
27
28      /**
29       *Acción por default del controlador, carga la lista
30       *de datos disponibles en el cie10 y una lista de opciones
31       *
32       * @param int $pag Número de registro para el paginador
33       *
34       * @return void
35       */
36      public function index($pag = 0)
37      {
38          if (empty($this-> Cie10_model))
39              return false;
40
41          if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
42              show_error('', 403, 'Acceso denegado');
43
44          try
45          {
46              $this-> load-> library('pagination');
47
48                  //Configuracion para la paginacion
49              $configPag['base_url'] = '/'. DIR_SIIGS.'/cie10/index/';
50              $configPag['first_link'] = 'Primero';
51              $configPag['last_link'] = '&Uacute;ltimo' ;
52              $configPag['total_rows'] = $this-> Cie10_model-> getNumRows();
53              $configPag['uri_segment'] = '4';
54              $configPag['per_page'] = 100;
55
56              $this-> pagination-> initialize($configPag);
57
58              $this-> Cie10_model-> setOffset($pag);
59              $this-> Cie10_model-> setRows($configPag['per_page']);
60
61              $data['title'] = 'Lista de datos en el catálogo CIE10';
62              $data['datos'] = $this-> Cie10_model-> getAll();
63              $data['msgResult'] = $this-> session-> flashdata('msgResult');
64              $data['clsResult'] = $this-> session-> flashdata('clsResult');
65
66          } catch (Exception $e)
67          {
```

```

68         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
69         $data['clsResult'] = 'error';
70     }
71
72     $this-> template-> write_view('content',DIR_SIIGS.'/cie10/index', $data);
73
74     $this-> template-> render();
75
76
77     /**
78      * Accion para mostrar informacion de los catalogos IDE , ERA y Consultas
79      *
80      * @param string $cat Nombre del catalogo a mostrar
81      * @return void
82      */
83
84     public function view($cat){
85         if (empty($this-> Cie10_model))
86             return false;
87
88         if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
89             show_error('', 403, 'Acceso denegado');
90
91         try
92     {
93             $data['title'] = 'Datos en el catálogo '.$cat;
94             $data['datos'] = $this-> Cie10_model-> getCatalogoByName($cat);
95             $data['catalogo'] = $cat;
96             $data['msgResult'] = $this-> session-> flashdata('msgResult');
97             $data['clsResult'] = 'success';
98         }
99         catch (Exception $e)
100     {
101         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
102         $data['clsResult'] = 'error';
103     }
104
105     $this-> template-> write_view('content',DIR_SIIGS.'/cie10/view', $data);
106
107     $this-> template-> render();
108 }
109
110 /**
111  * Accion para agregar elementos del catalogo cie10 a los catalogos de EDA, IRA y
Consultas dependiendo de los
112  * Solo se permite su acceso por medio de peticiones AJAX
113  *
114  * @param Int $id Es el id del registro en el catalogo de CIE10
115  * @param String $catalogo para determinar a que catalogo se va a agregar o quitar el
registro
116  * @param Boolean $activo False para quitar del catalogo, true para agregarlo
117  *
118  * @return Boolean En caso de error, o errores de referencia, etc.
119  */
120
121     public function AgregaEnCatalogo(){
122
123         if (!$this-> input-> is_ajax_request())
124             show_error('', 403, 'Acceso denegado');
125
126         try
127     {
128         if ($this-> input-> is_ajax_request())
129     {
130             $id = $this-> input-> post('id');
131             $catalogo = $this-> input-> post('catalogo');
132             $activo = $this-> input-> post('activo');
133             // $id = 6110;
134             // $catalogo = "eda";
135             // $activo = true;
136             if ($id && $catalogo)
137             {
138                 $resultado = $this-> Cie10_model-
> agregaEnCatalogo($id,"cns_". $catalogo,$activo);
139                 if ($resultado == true)
140                     echo "ok"
141                 else
142                     echo "error"
143             }
144         }
145     }
146
147     }

```

```

145                     echo "Parametros incorrectos" ;
146                 }
147             else echo 'Acceso denegado';
148         }
149         catch(Exception $e)
150     {
151         echo $e-> getMessage();
152     }
153 }
154 /**
155 * Accion para activar o desactivar elementos en los catalogos IRA EDA Consultas
156 * Solo se permite su acceso por medio de peticiones AJAX
157 *
158 * @param Int $id Es el id del registro en el catalogo
159 * @param String $catalogo para determinar a que catalogo se va a agregar o quitar el
160 registro
161 * @param Boolean $activo False para quitar del catalogo, true para agregarlo
162 *
163 * @return Boolean En caso de error, o errores de referencia, etc.
164 */
165 public function ActivaEnCatalogo(){
166
167     if (!$this-> input-> is_ajax_request())
168         show_error('', 403, 'Acceso denegado');
169
170     try
171     {
172         if ($this-> input-> is_ajax_request())
173     {
174         $id = $this-> input-> post('id');
175         $catalogo = $this-> input-> post('catalogo');
176         $activo = $this-> input-> post('activo');
177         // $id = 6110;
178         // $catalogo = "eda";
179         // $activo = true;
180         if ($id && $catalogo)
181         {
182             $resultado = $this-> Cielo_model-
183             activaEnCatalogo($id, "cns_" . $catalogo, $activo);
184             if ($resultado == true)
185                 echo "ok" ;
186             else
187                 echo "error" ;
188         }
189         else
190             echo "Parametros incorrectos" ;
191     }
192     else echo 'Acceso denegado';
193
194     catch(Exception $e)
195     {
196         echo $e-> getMessage();
197     }
198 }
199 /**
200 * Acción para cargar datos desde un archivo CSV, recibe el stream desde las variables PHP
201 * Guarda en la tabla tmp_catalogos toda la estructura del CSV e imprime las columnas del
202 * archivo
203 * Solo se permite su acceso por medio de peticiones AJAX
204 *
205 * @param $_FILES[] archivocsv Variable pasada por POST con el archivo csv para cargar
206 datos
207 * @return void
208 */
209 public function load()
210 {
211
212     if (!$this-> input-> is_ajax_request())
213         show_error('', 403, 'Acceso denegado');
214
215     if (isset($_FILES["archivocsv"] ) &&
216     is_uploaded_file($_FILES['archivocsv']['tmp_name']))
217     {
218         $fp = fopen($_FILES['archivocsv']['tmp_name'], "r" );
219         // $fp = fopen('catalogos/estados.csv', "r");
220         $columnas = array('ciel0','descripcion');

```

```

221 //while (!feof($fp))
222         while (($data = fgetcsv($fp)) !== FALSE)
223     {
224         // $data = explode(", ", fgets($fp));
225         if (count($data) == 2)
226         {
227             $utf8_encode = function($val)
228             {
229                 return utf8_encode(addslashes($val));
230             };
231             $data = array_map($utf8_encode,$data);
232             $data = preg_replace("!\r?\n!", "", $data);
233         }
234     }
235     // Crea los rows con las filas que cumplen con la
236     // estructura en el CSV
237     if (count($columnas) == count($data))
238     {
239         $item = array_combine($columnas, $data);
240         array_push($rows, $item);
241     }
242 }
243 }
244 fclose($fp);
245
246 // Inserta los datos en lotes a la tabla temporal
247 $this-> db-> insert_batch('cns_cie10', $rows);
248
249 }
250 else
251 {
252     echo "Error: el archivo no se cargó correctamente";
253 }
254 }
255
256
257 /**
258 * Acción para preparar la actualización de un registro del CIE10,
259 * recibe un id para obtener los valores del catálogo y mostrarlos
260 * en la vista update , realiza la validación del formulario del lado
261 * del cliente y servidor
262 *
263 * @param int $id
264 * @return void
265 */
266 public function update($id)
267 {
268     if (empty($this-> Cie10_model))
269         return false;
270
271     if (!Usuario_model::checkCredentials(DIR_SIIIGS::__METHOD__, current_url()))
272         show_error('', 403, 'Acceso denegado');
273
274     $this-> load-> helper('form');
275     $this-> load-> helper('url');
276     $this-> load-> library('form_validation');
277
278     $error = false;
279
280     $data['title'] = "Modificar datos del catálogo CIE10";
281     $this-> form_validation-> set_rules('descripcion', 'Descripción',
282                                         'trim|xss_clean|required|max_length[100]');
283
284     if ($this-> form_validation-> run() === FALSE)
285     {
286         try
287         {
288             $data['catalogo_item'] = $this-> Cie10_model-> getById($id);
289         }
290         catch (Exception $e)
291         {
292             $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
293             $data['clsResult'] = 'error';
294         }
295
296         $this-> template-> write_view('content', DIR_SIIIGS.'/cie10/update', $data);
297         $this-> template-> render();
298     }

```

```

298     else
299     {
300         try
301         {
302             $this-> Cielo_model-> setDescripcion($this-> input-
303 > post('descripcion'));
304             $this-> Cielo_model-> setId($this-> input-> post('id'));
305             $this-> Cielo_model-> update();
306         }
307         catch (Exception $e)
308         {
309             try
310             {
311                 $data['catalogo_item'] = $this-> Cielo_model-> getById($id);
312             }
313             catch (Exception $e)
314             {
315                 $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
316                 $data['clsResult'] = 'error';
317             }
318             $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
319             $data['clsResult'] = 'error';
320             $this-> template-> write_view('content',DIR_SIIGS.'/ciel0/update', $data);
321             $this-> template-> render();
322
323             $error = true;
324         }
325
326         if ($error == false)
327         {
328             $this-> session-> set_flashdata('msgResult', 'Registro actualizado
correctamente');
329             $this-> session-> set_flashdata('clsResult', 'success');
330             redirect(DIR_SIIGS.'/ciel0','refresh');
331         }
332     }
333 }
334
335 /**
336 *Acción para cargar datos desde un archivo CSV, recibe el stream desde las variables PHP
337 *compara los datos recibidos con los datos que contiene actualmente el catálogo, regresa
como
338 *resultado las filas nuevas y las filas a modificar
339 *
340 *Solo se permite su acceso por medio de peticiones AJAX
341 *
342 * @param $_FILES[] archivocsv Variable pasada por POST con el archivo csv para cargar
datos
343
344 * @param boolean $update Indica si se modifica la base de datos o solo una revisión de
campos
345 */
346 public function insert($update = false)
347 {
348     if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
349         show_error('', 403, 'Acceso denegado');
350
351     ini_set('max_execution_time',1000);
352
353     if (isset($_FILES["archivocsv"])) &&
354     is_uploaded_file($_FILES['archivocsv'][tmp_name]))
355     //if (TRUE)
356     {
357         $fp = fopen($_FILES['archivocsv'][tmp_name], "r");
358         // $fp = fopen('ciel0.csv', "r");
359         $cont = 0;
360         $columnas = array();
361         $resultado = array();
362         $rows = array();
363         $nuevos = 0; $modificados = 0; $iguales = 0; $errores = 0;
364
365         //obtiene los datos del catalogo
366         try
367         {
368             $rowscat = $this-> Cielo_model-> getData();
369         }
370         catch (Exception $e)
371         {
372             echo Errorlog_model::save($e-> getMessage(), __METHOD__);

```

```

372                     return;
373     }
374     try
375     {
376         //array que contiene todos los registros llave
377         $datallaves = array();
378
379         //array para hacer modificaciones por lotes
380         $consultamodificar = array();
381         //array para hacer inserciones por lotes
382         $consultaagregar = array();
383
384         //filtro solo los nombres de los campos
385         $campos = array('id_categoria','id_ciel0','descripcion','activo');
386         $llaves = array('id_ciel0');
387
388         //obtiene las llaves primarias del catalogo
389         $rowsllaves = $this-> db-> query("select
390             , $llaves)." from cns_ciel0" );
391         $rowsllaves = $rowsllaves-> result();
392         //var_dump($rowsllaves);
393     }
394     catch (Exception $e)
395     {
396         echo Errorlog_model::save($e-> getMessage(), __METHOD__);
397         die();
398     }
399     ini_set('memory_limit', '1024M');
400     $error = false;
401     //while (!feof($fp))
402     while ($data = fgetcsv($fp)) !== FALSE)
403     {
404         $utf8_encode = function($val)
405         {
406             return utf8_encode(addslashes($val));
407         };
408         $data = array_map($utf8_encode,$data);
409
410         $cont +=1;
411         $data = preg_replace("!\r?\n!", " ", $data);
412         if ($cont == 1)
413         {
414             $errorcols = false;
415             $columnas = $data;
416             if (count($campos) != count($columnas))
417                 $errorcols = true;
418             for ($i = 0; $i < count($campos); $i++)
419             {
420                 if (!isset($columnas[$i]) || !isset($campos[$i]) || $campos[$i] !=
421                     $columnas[$i])
422                     $errorcols = true;
423             }
424             if ($errorcols)
425             {
426                 echo json_encode(array("Error" . implode(',', $campos). " .
427                     . implode(',', $columnas)));
428                 die();
429             }
430             else
431             {
432                 $indicellaves = array();
433                 //Obtiene los indices de columnas que corresponden a las llaves en el
434                 //CSV
435                 for ($i = 0; $i < count($columnas); $i++)
436                 {
437                     if (in_array($columnas[$i],$llaves))
438                     {
439                         array_push($indicellaves,$i);
440                     }
441                 }
442                 if (count($columnas) == count($data))
443                 {
444                     $datallave = array();
445                     foreach ($indicellaves as $i)
446                     {

```

```

447                         array_push($datallave, $data[$i]);
448                     }
449                     //agrega el registro llave a la lista de llaves
450                     array_push($datallaves,$datallave);
451                     //agrega las claves con nombres de campo
452                     $procesada = array_combine($columnas,$data);
453                     //si el registro existe en el catalogo
454                     if (in_array((object)$procesada, $rowscat))
455                     {
456                         $iguales += 1;
457                     }
458                     else
459                     {
460                         //si la clave existe en el catalogo
461                         if (in_array((object)array_combine($llaves,$datallave),
462                                     $rowsllaves))
463                         {
464                             $update = true;
465                         }
466                         cns_ciel0 set ';
467                         1=1 ';
468                         =>      $value)
469
470                         $llaves))
471
472                         $key." = '" . addslashes(utf8_encode($value))."' ,"
473
474                         .$value."'";
475
476                         $consultaupdatewhere .= ' and '.$key." = '" . $value."'";
477
478
479                         substr($consultaupdate,0, count($consultaupdate)-2);
480
481                         $consultaupdatewhere;
482                         $consultaupdate;
483
484                         1000)
485
486                         as $sql)
487                         > query($sql))
488
489                         array();
490
491
492                         }
493                         else
494                         {
495
496
497
498                         $clave=> $valor)
499
500                         $data[$clave];
501
502                         $arraytemp;
503
504                         1000)
505                         > insert_batch('cns_ciel0',$consultaagregar)==0)
506
507
array_push($datallave, $data[$i]);
}
//agrega el registro llave a la lista de llaves
array_push($datallaves,$datallave);
//agrega las claves con nombres de campo
$procesada = array_combine($columnas,$data);
//si el registro existe en el catalogo
if (in_array((object)$procesada, $rowscat))
{
    $iguales += 1;
}
else
{
    //si la clave existe en el catalogo
    if (in_array((object)array_combine($llaves,$datallave),
        $rowsllaves))
    {
        if ($update == true)
        {
            $consultaupdate = 'update
$consultaupdatewhere = ' where
foreach ($procesada as $key
{
    if (!in_array($key,
        {
            $consultaupdate .=
}
else
{
}
}
}
$consultaupdate =
$consultaupdate .=
array_push($consultamodificar,
if (count($consultamodificar) ==
{
    foreach ($consultamodificar
        if (!$this-> db-
            $error = true;
$consultamodificar =
}
}
$modificados += 1;

if ($update == true)
{
    $arraytemp = array();
    foreach($campos as
{
    $arraytemp[$valor] =
}
array_push($consultaagregar,
if (count($consultaagregar) ==
{
    if ($this-> db-
        $error = true;
$consultaagregar = array();
}
}

```



```
582             {
583                 $arr[$key]=implode(" ", "
584             }
585         return array_unique($arr,SORT_REGULAR);
586     }
587 }
588 }
```

# Archivo fuente para controlador.php

La documentación para este archivo está disponible en [controlador.php](#)

```
1  <?php
2  /**
3   * Controlador Controlador
4   *
5   * @package    SIIGS
6   * @subpackage Controlador
7   * @author     Geovanni
8   * @created    2013-09-26
9   */
10  class Controlador extends CI_Controller {
11
12      public function __construct()
13      {
14          parent::__construct();
15
16          try
17          {
18              $this-> load-> helper('url');
19              $this-> load-> model(DIR_SIIGS.'/Controlador_model');
20              $this-> load-> model(DIR_SIIGS.'/Entorno_model');
21          }
22          catch (Exception $e)
23          {
24              $this-> template-> write("content" , $e-> getMessage());
25              $this-> template-> render();
26          }
27      }
28
29      /**
30       * Acción por default del controlador, carga la lista
31       * de controladores disponibles y una lista de opciones
32       * Recibe un parametro en caso de filtrado por entornos
33       *
34       * @param int $pag Número de registro para el paginador
35       * @return void
36       */
37      public function index($pag = 0)
38      {
39          if (empty($this-> Controlador_model))
40          {
41              if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
42                  show_error('', 403, 'Acceso denegado');
43          try
44          {
45              $this-> load-> library('pagination');
46              $this-> load-> helper('form');
47
48              $id = $this-> input-> post('id_entorno') ? $this-> input-
49              > post('id_entorno') : 0;
50
51              //Configuracion para la paginacion
52              $configPag['base_url'] = '/' . DIR_SIIGS.'/controlador/index/';
53              $configPag['first_link'] = 'Primero';
54              $configPag['last_link'] = '&Uacute;ltimo';
55              $configPag['total_rows'] = $this-> Controlador_model-> getNumRows($id);
56              $configPag['uri_segment'] = '4';
57              $configPag['per_page'] = 20;
58
59              $this-> pagination-> initialize($configPag);
60
61              if (!$this-> input-> is_ajax_request())
62              {
63                  $this-> Controlador_model-> setOffset($pag);
64                  $this-> Controlador_model-> setRows($configPag['per_page']);
65              }
66
67              $data['title'] = 'Lista de Controladores disponibles' .
68              ' (' . $this-> Entorno_model-> getById($id)-> nombre . ')';
69          }
70      }
```

```

);
67     $data['msgResult'] = $this-> session-> flashdata('msgResult');
68     $data['clsResult'] = $this-> session-> flashdata('clsResult');
69     $data['id_entorno'] = ($id) ? $id : 0;
70     $data['pag'] = $pag;
71
72     if ($id == 0)
73     {
74         $data['controladores'] = $this-> Controlador_model-> getAll();
75     }
76     else
77         $data['controladores'] = $this-> Controlador_model-> getByEntorno($id);
78
79     if ($this-> input-> is_ajax_request())
80     {
81         echo json_encode($data["controladores"]);
82         die();
83     }
84
85     $entornos = $this-> Entorno_model-> getAll();
86     $data['entornos'][0] = 'Todos';
87     foreach ($entornos as $item)
88     {
89         $data['entornos'][$item-> id] = $item-> nombre;
90     }
91
92     catch (Exception $e)
93     {
94         $data['clsResult'] = 'error';
95         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
96     }
97
98     //Dibujar la vista en el navegador
99     $this-> template-> write_view('content',DIR_SIIGS.'/controlador/index', $data);
100    $this-> template-> render();
101
102 /**
103 *Acción para visualizar información de un controlador específico, obtiene el objeto
104 *controlador por medio del id proporcionado.
105 */
106 * @param int $id Este parametro no puede ser nulo
107 * @return void
108 */
109 public function view($id)
110 {
111     if (empty($this-> Controlador_model))
112         return false;
113         if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
114     show_error('', 403, 'Acceso denegado');
115     try
116     {
117         $data['title'] = "Detalles del controlador";
118         $data['controlador_item'] = $this-> Controlador_model-> getById($id);
119     }
120     catch(Exception $e)
121     {
122         $data['clsResult'] = 'error';
123         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
124     }
125
126     $this-> template-> write_view('content',DIR_SIIGS.'/controlador/view', $data);
127     $this-> template-> write('menu','','true');
128     $this-> template-> write('sala_prensa','','true');
129     $this-> template-> render();
130 }
131
132 /**
133 *Acción para preparar la insercion de nuevos controladores , realiza la validacion
134 *del formulario del lado cliente
135 */
136 * @return void
137 */
138 public function insert($id = FALSE)
139 {
140         if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
141     show_error('', 403, 'Acceso denegado');
142     $error = false;
143     $this-> load-> helper('form');
144     $this-> load-> library('form_validation');
145

```

```

146      $data['title'] = 'Crear un nuevo controlador';
147      $this-> form_validation-> set_rules('id_entorno', 'Entorno',
148      'required|is_natural_no_zero');
148      $this-> form_validation-> set_rules('nombre', 'Nombre',
149      'trim|xss_clean|required|max_length[40]');
149      $this-> form_validation-> set_rules('descripcion', 'Descripcion',
150      'trim|xss_clean|required|max_length[100]');
150      $this-> form_validation-> set_rules('clase', 'Clase',
151      'trim|xss_clean|required|max_length[30]');
151
152      $data['id_entorno'] = ($id != FALSE) ? $id : $this-> input-> post('id_entorno');
153      $entornos = $this-> Entorno_model-> getAll();
154      $data['entornos']['''] = 'Elige un entorno';
155      foreach ($entornos as $item) {
156          $data['entornos'][$item-> id] = $item-> nombre;
157      }
158
159      if ($this-> form_validation-> run() == FALSE)
160      {
161
162          $this-> template-> write_view('content',DIR_SIIGS.'/controlador/insert',$data);
163          $this-> template-> render();
164      }
165      else
166      {
167          try
168          {
169              $this-> load-> helper('url');
170
171              $this-> Controlador_model-> setNombre($this-> input-> post('nombre'));
172              $this-> Controlador_model-> setDescripcion($this-> input-
173 > post('descripcion'));
173              $this-> Controlador_model-> setClase($this-> input-> post('clase'));
174              $this-> Controlador_model-> setIdEntorno($this-> input-
174 > post('id_entorno'));
175
176              $this-> Controlador_model-> insert();
177          }
178          catch (Exception $e)
179          {
180              $data['clsResult'] = 'error';
181              $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
182              $this-> template-> write_view('content',DIR_SIIGS.'/controlador/insert',
182 $data);
183              $this-> template-> render();
184              $error = true;
185          }
186
187          if ($error == false)
188          {
189              $this-> session-> set_flashdata('msgResult', 'Registro insertado
correctamente');
190              $this-> session-> set_flashdata('clsResult', 'success');
191              redirect(DIR_SIIGS.'/controlador/index','refresh');
192          }
193      }
194  }
195
196 /**
197 *Acción para preparar la actualización de un controlador ya existente,
198 *recibe un ID para obtener los valores de ese controlador y mostrarlos
199 *en la vista update , realiza la validación del formulario del lado
200 *del cliente
201 *
202 * @param int $id
203 * @return void
204 */
205 public function update($id)
206 {
207     if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
208     show_error('', 403, 'Acceso denegado');
209
210     $this-> load-> helper('form');
211     $this-> load-> library('form_validation');
212
213     $error = false;
214
215     $data['title'] = 'Modificar controlador';
216     $this-> form_validation-> set_rules('id_entorno', 'Entorno',
216 'required|is_natural_no_zero');

```

```

217      $this-> form_validation-> set_rules('nombre', 'Nombre',
218      'trim|xss_clean|required|max_length[40]');
219      $this-> form_validation-> set_rules('descripcion', 'Descripción',
220      'trim|xss_clean|required|max_length[100]');
221      $this-> form_validation-> set_rules('clase', 'Clase',
222      'trim|xss_clean|required|max_length[30]');
223
224      if ($this-> form_validation-> run() === FALSE)
225      {
226          try
227          {
228              $data["controlador_item"] = $this-> Controlador_model-
229              > getById($id);
230
231              $entornos = $this-> Entorno_model-> getAll();
232              $data['entornos'][0] = 'Elige un entorno';
233              foreach ($entornos as $item) {
234                  $data['entornos'][$item-> id] = $item-> nombre;
235              }
236          }
237          catch (Exception $e)
238          {
239              $data['clsResult'] = 'error';
240              $data['msgResult'] = Errorlog_model::save($e-> getMessage(),
241 _METHOD__);
242          }
243      }
244      else
245      {
246          try
247          {
248              $this-> load-> helper('url');
249              $this-> Controlador_model-> setNombre($this-> input-> post('nombre'));
250              $this-> Controlador_model-> setDescripcion($this-> input-
251              > post('descripcion'));
252              $this-> Controlador_model-> setClase($this-> input-> post('clase'));
253              $this-> Controlador_model-> setIdEntorno($this-> input-
254              > post('id_entorno'));
255              $this-> Controlador_model-> setId($this-> input-> post('id'));
256              $this-> Controlador_model-> update();
257          }
258          catch (Exception $e)
259          {
260              $data['title'] = 'Modificar controlador';
261              try
262              {
263                  $data['controlador_item'] = $this-> Controlador_model-> getById($id);
264
265                  $entornos = $this-> Entorno_model-> getAll();
266                  $data['entornos'][0] = 'Elige un entorno';
267                  foreach ($entornos as $item) {
268                      $data['entornos'][$item-> id] = $item-
269                      > nombre;
270                  }
271              }
272              catch (Exception $e)
273              {
274                  $data['clsResult'] = 'error';
275                  $data['msgResult'] = Errorlog_model::save($e-> getMessage(), _METHOD__);
276              }
277          }
278
279          $data;
280      }
281
282      if ($error == false)
283      {
284          $this-> session-> set_flashdata('clsResult', 'success');
285          $this-> session-> set_flashdata('msgResult', 'Registro actualizado
correctamente');
286      }

```

```

287             redirect(DIR_SIIGS.'/controlador','refresh');
288         }
289     }
290 }
291 /**
292 *Acción para preparar la actualización de acciones asignadas a un controlador,
293 *recibe un ID para obtener las acciones asignadas a ese controlador y mostrarlos
294 *en la vista update
295 *
296 * @param int $id
297 * @return void
298 */
299 public function accion($id)
300 {
301     if (!Usuario_model::checkCredentials(DIR_SIIGS::__METHOD__, current_url()))
302         show_error('', 403, 'Acceso denegado');
303
304     $this-> load-> helper(array('form','url'));
305     $this-> load-> library('form_validation');
306     $this-> load-> model('siigs/ControladorAccion_model');
307
308     $data['title'] = 'Acciones asignadas al controlador';
309     $data['id_controlador'] = $id;
310     $this-> form_validation-> set_rules('acciones', 'Acciones', 'required');
311
312     if ($this-> form_validation-> run() === FALSE)
313     {
314         try
315         {
316             $data["controlador_acciones"] = $this-> Controlador_model-
317             > getAcciones($id);
318         }
319         catch (Exception $e)
320         {
321             $data['clsResult'] = 'error';
322             $data['msgResult'] = Errorlog_model::save($e-> getMessage(),
323             __CLASS__. '::update');
324         }
325
326         $this-> template-> write_view('content',DIR_SIIGS.'/controlador/accion', $data);
327         $this-> template-> render();
328     }
329     else
330     {
331         try
332         {
333             $error = false;
334
335             $this-> session-> set_flashdata('clsResult', 'success');
336             $this-> session-> set_flashdata('msgResult', 'Se modificaron correctamente
las acciones permitidas al controlador');
337
338             $this-> Controlador_model-> setAccion($this-> input-> post('acciones'));
339             $this-> Controlador_model-> setId($this-> input-> post('id'));
340             $this-> Controlador_model-> accionesUpdate();
341         }
342         catch (Exception $e)
343         {
344             $data['msgResult'] = 'error';
345             $data['msgResult'] = Errorlog_model::save($e-> getMessage(),
346             __CLASS__. '::update');
347             $error = true;
348         }
349
350         if ($error == true)
351         {
352             $this-> template-> write_view('content',DIR_SIIGS.'/controlador/accion',
353             $data);
354             $this-> template-> render();
355         }
356     }
357
358 /**
359 *
360 *Acción para eliminar un controlador, recibe el id del controlador a eliminar
361 */

```

```

362     * @param int $id
363     * @return void
364     */
365    public function delete($id)
366    {
367        if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
368            show_error('', 403, 'Acceso denegado');
369
370        try
371        {
372            if (empty($this-> Controlador_model))
373                return false;
374
375            $this-> load-> helper('url');
376            $this-> Controlador_model-> setId($id);
377            $this-> Controlador_model-> delete();
378            $this-> session-> set_flashdata('msgResult', 'Registro eliminado exitosamente');
379            $this-> session-> set_flashdata('clsResult', 'success');
380        }
381        catch(Exception $e)
382        {
383            $this-> session-> set_flashdata('clsResult', 'error');
384            $this-> session-> set_flashdata('msgResult', Errorlog_model::save($e-> getMessage(), __METHOD__));
385        }
386        redirect(DIR_SIIGS.'/controlador','refresh');
387    }
388
389
390 /**
391 * Acción para servir un array de objetos con los permisos asignados a
392 * un entorno y grupo determinados, esta acción solo es accedida por peticiones
393 * AJAX y devuelve un objeto JSON
394 *      *Solo se permite su acceso por medio de peticiones AJAX
395 *
396 * @param int $entorno
397 * @param int $grupo
398 * @return Object JSON
399 */
400 public function getGroupPermissions($entorno, $grupo)
401 {
402     try {
403         if ($this-> input-> is_ajax_request())
404         {
405             $data['permisos'] = $this-> Controlador_model-> getPermisos($entorno , $grupo);
406             echo json_encode($data['permisos']);
407             exit;
408         }
409         else echo 'Acceso denegado';
410     }
411     catch(Exception $e){
412         echo $e-> getMessage();
413     }
414 }
415
416 /**
417 * Establece el texto de ayuda para el controlador acción
418 *
419 * @param int $id_controlador_accion
420 * @return void
421 */
422 public function help($idControladorAccion)
423 {
424     $data['id_controlador_accion'] = $idControladorAccion;
425
426     $this-> load-> helper('form');
427
428     if($this-> input-> post('ayuda_controlador_accion')) {
429         $this-> load-> model('ControladorAccion_model');
430
431         try {
432             $this-> ControladorAccion_model-> setHelp($idControladorAccion, $this-> input-> post('ayuda_controlador_accion'));
433             $data['msgResult'] = 'Registro actualizado exitosamente';
434             $data['clsResult'] = 'success';
435         }
436         catch(Exception $e)
437         {
438             $data['msgResult'] = $e-> getMessage();
439             $data['clsResult'] = 'error';

```

```
440 }
441 } else {
442     $datos = $this-> ControladorAccion_model-> getById($idControladorAccion);
443     $data['ayuda'] = $datos-> ayuda;
444 }
445
446 $this-> template-> write('sala_prensa', '', true);
447 $this-> template-> write('menu', '', true);
448 $this-> template-> write_view('content',DIR_SIIGS.'/controlador/help', $data);
449 $this-> template-> render();
450 }
451 }
```

# Archivo fuente para entorno.php

La documentación para este archivo está disponible en [entorno.php](#)

```
1      <?php
2
3  /**
4   * Controlador Entorno
5   *
6   * @package    SIIGS
7   * @subpackage Controlador
8   * @author     Geovanni
9   * @created    2013-09-26
10  */
11 class Entorno extends CI_Controller {
12
13     public function __construct()
14     {
15         parent::__construct();
16         try
17         {
18             $this-> load-> helper('url');
19             $this-> load-> model(DIR_SIIGS.'/Entorno_model');
20         }
21         catch (Exception $e)
22         {
23             $this-> template-> write("content" , $e-> getMessage());
24             $this-> template-> render();
25         }
26     }
27
28     /**
29      *Acción por default del controlador, carga la lista
30      *de entornos disponibles y una lista de opciones
31      *No recibe parámetros
32      *
33      *@return void
34      */
35     public function index()
36     {
37         if (empty($this-> Entorno_model))
38             return false;
39             if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
40             show_error('', 403, 'Acceso denegado');
41         try
42         {
43             $data['title'] = 'Lista de Entornos disponibles';
44             $data['msgResult'] = $this-> session-> flashdata('msgResult');
45             $data['clsResult'] = $this-> session-> flashdata('clsResult');
46             $data['entornos'] = $this-> Entorno_model-> getAll();
47         }
48         catch (Exception $e)
49         {
50             $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
51             $data['clsResult'] = 'error';
52         }
53
54         $this-> template-> write_view('content',DIR_SIIGS.'/entorno/index' , $data);
55         $this-> template-> render();
56     }
57
58     /**
59      *Acción para visualizar de un entorno específico, obtiene el objeto
60      *entorno por medio del id proporcionado.
61      *
62      * @param int $id Este parametro no puede ser nulo
63      * @return void
64      */
65     public function view($id)
66     {
67         if (empty($this-> Entorno_model))
```

```

68        return false;
69        if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
70        show_error('', 403, 'Acceso denegado');
71    try
72    {
73        $data['title'] = "Detalles del entorno";
74        $data['entorno_item'] = $this-> Entorno_model-> getById($id);
75    }
76    catch(Exception $e)
77    {
78        $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
79        $data['clsResult'] = 'error';
80    }
81
82    $this-> template-> write_view('content',DIR_SIIGS.'/entorno/view', $data);
83    $this-> template-> write('menu','','true');
84    $this-> template-> write('sala_prensa','','true');
85    $this-> template-> render();
86}
87
88 /**
89 *Acción para preparar la inserción de nuevos entornos , realiza la validación
90 *del formulario del lado cliente y del lado servidor para evitar entornos duplicados
91 */
92
93 /**
94 public function insert()
95 {
96     if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
97     show_error('', 403, 'Acceso denegado');
98
99     $error = false;
100    $this-> load-> helper('form');
101    $this-> load-> library('form_validation');
102
103    $data['title'] = 'Crear un nuevo entorno';
104    $this-> form_validation-> set_rules('nombre', 'Nombre',
105    'trim|xss_clean|required|max_length[15]|callback_ExistEntorno');
106    $this-> form_validation-> set_rules('ip', 'Ip',
107    'trim|xss_clean|required|min_length[7]|max_length[15]|valid_ip');
108    $this-> form_validation-> set_rules('descripcion', 'Descripción',
109    'trim|xss_clean|required|max_length[100]');
110    $this-> form_validation-> set_rules('hostname', 'Hostname',
111    'trim|xss_clean|required|max_length[100]');
112    $this-> form_validation-> set_rules('directorio', 'Directorio',
113    'trim|xss_clean|required|max_length[20]');
114
115    if ($this-> form_validation-> run() === FALSE)
116    {
117        $this-> template-> write_view('content',DIR_SIIGS.'/entorno/insert',$data);
118        $this-> template-> render();
119    }
120    else
121    {
122        try
123        {
124            $this-> load-> helper('url');
125
126            $this-> Entorno_model-> setNombre($this-> input-> post('nombre'));
127            $this-> Entorno_model-> setDescripcion($this-> input-
128            > post('descripcion'));
129            $this-> Entorno_model-> setIp($this-> input-> post('ip'));
130            $this-> Entorno_model-> setHostname($this-> input-> post('hostname'));
131            $this-> Entorno_model-> setDirectorio($this-> input-
132            > post('directorio'));
133
134            $this-> Entorno_model-> insert();
135        }
136        catch (Exception $e)
137        {
138            $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
139            $data['clsResult'] = 'error';
140            $this-> template-> write_view('content',DIR_SIIGS.'/entorno/insert', $data);
141            $this-> template-> render();
142            $error = true;
143        }
144        if ($error == false)
145        {
146            $this-> session-> set_flashdata('msgResult', 'Registro insertado
correctamente');
147        }
148    }
149}

```

```

140             $this-> session-> set_flashdata('clsResult', 'success');
141             redirect(DIR_SIIGS.'/entorno/index','refresh');
142         }
143     }
144 }
145 /**
146 *Acción para validar que no exista previamente el entorno a insertar
147 *(Esta acción no puede ser accedida desde el navegador)
148 *
149 * @param string $nombre_entorno Revisa si este valor ya existe como un entorno
150 * @return boolean
151 */
152 public function _ExistEntorno($nombre_entorno) {
153
154     $exist = $this-> Entorno_model-> getByName($nombre_entorno);
155
156     if ($exist)
157     {
158         $this-> form_validation-> set_message(
159             '_ExistEntorno', 'Este entorno ya existe en la base de datos, intente con
otro nombre.');
160             );
161             return false;
162     } else
163     {
164         return true;
165     }
166 }
167 }
168 /**
169 *Acción para validar que no exista previamente el entorno a actualizar
170 *esta acción revisa si el nombre a usar ya existe en la base excepto
171 *el mismo objeto a actualizar
172 *(Esta acción no puede ser accedida desde el navegador)
173 *
174 * @param string $nombre_entorno Revisa si este valor ya existe como un entorno
175 * @return boolean
176 */
177 public function _ExistEntornoUpdate($nombre_entorno) {
178
179     $where = 'select * from sis_entorno where nombre = "' .
180             $nombre_entorno.'" and id<>"'
181             $this-> input-> post('id')."'" ;
182
183     $exist = $this-> db-> query($where);
184
185     if ($exist-> num_rows() > 0)
186     {
187         $this-> form_validation-> set_message(
188             '_ExistEntornoUpdate', 'Este entorno ya existe en la base de datos, intente
con otro nombre.');
189             );
190             return false;
191     } else
192     {
193         return true;
194     }
195 }
196 }
197 }
198 /**
199 *Acción para preparar la actualización de un entorno ya existente,
200 *recibe un ID para obtener los valores de ese entorno y mostrarlos
201 *en la vista update , realiza la validación del formulario del lado
202 *del cliente y del servidor para evitar datos duplicados
203 *
204 * @param int $id
205 * @return void
206 */
207 public function update($id)
208 {
209     if (!Usuario_model::checkCredentials(DIR_SIIGS::__METHOD__, current_url()))
210         show_error('', 403, 'Acceso denegado');
211     //Load helpers and libraries
212     $this-> load-> helper('form');
213     $this-> load-> helper('url');
214     $this-> load-> library('form_validation');
215
216     $error = false;

```

```

218
219     $data['title'] = 'Modificar entorno';
220     $this-> form_validation-> set_rules('nombre', 'Nombre',
221 'trim|xss_clean|required|max_length[15]|callback_ExistEntornoUpdate');
222     $this-> form_validation-> set_rules('ip', 'Ip',
223 'trim|xss_clean|required|min_length[7]|max_length[15]|valid_ip');
224     $this-> form_validation-> set_rules('descripcion', 'Descripción',
225 'trim|xss_clean|required|max_length[100]');
226     $this-> form_validation-> set_rules('hostname', 'Hostname',
227 'trim|xss_clean|required|max_length[100]');
228     $this-> form_validation-> set_rules('directorio', 'Directorio',
229 'trim|xss_clean|required|max_length[20]');
230     if ($this-> form_validation-> run() === FALSE)
231     {
232         try
233         {
234             $data['entorno_item'] = $this-> Entorno_model-> getById($id);
235         }
236         catch (Exception $e)
237         {
238             $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
239             $data['clsResult'] = 'error';
240         }
241     }
242     try
243     {
244         $this-> Entorno_model-> setNombre($this-> input-> post('nombre'));
245         $this-> Entorno_model-> setDescripcion($this-> input-
246 > post('descripcion'));
247         $this-> Entorno_model-> setIp($this-> input-> post('ip'));
248         $this-> Entorno_model-> setHostname($this-> input-> post('hostname'));
249         $this-> Entorno_model-> setDirectorio($this-> input-
250 > post('directorio'));
251         $this-> Entorno_model-> setId($this-> input-> post('id'));
252         $this-> Entorno_model-> update();
253     }
254     catch (Exception $e)
255     {
256         $data['title'] = 'Modificar entorno';
257         try
258         {
259             $data['entorno_item'] = $this-> Entorno_model-> getById($id);
260         }
261         catch (Exception $e)
262         {
263             $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
264             $data['clsResult'] = 'error';
265         }
266         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
267         $data['clsResult'] = 'error';
268         $this-> template-> write_view('content',DIR_SIIGS.'/entorno/update', $data);
269         $this-> template-> render();
270
271         $error = true;
272     }
273
274     if ($error == false)
275     {
276         $this-> session-> set_flashdata('msgResult', 'Registro actualizado
277 correctamente');
278         $this-> session-> set_flashdata('clsResult', 'success');
279         redirect(DIR_SIIGS.'/entorno','refresh');
280     }
281 }
282
283 /**
284 *
285 * Acción para eliminar un entorno, recibe el id del entorno a eliminar
286 *
287 * @param int $id
288 * @return void
289 */

```

```

290     public function delete($id)
291     {
292         if (!Usuario_model::checkCredentials(DIR_SIIGS::__METHOD__, current_url()))
293             show_error(' ', 403, 'Acceso denegado');
294         try
295         {
296             if (empty($this-> Entorno_model))
297                 return false;
298
299             $this-> load-> helper('url');
300             $this-> Entorno_model-> setId($id);
301             $this-> Entorno_model-> delete();
302             $this-> session-> set_flashdata('msgResult', 'Registro eliminado exitosamente');
303             $this-> session-> set_flashdata('clsResult', 'success');
304         }
305         catch(Exception $e)
306         {
307             $this-> session-> set_flashdata('msgResult', Errorlog_model::save($e-> getMessage(), __METHOD__));
308             $this-> session-> set_flashdata('clsResult', 'error');
309         }
310         redirect(DIR_SIIGS. '/entorno', 'refresh');
311     }
312 }
```

# Archivo fuente para errorlog.php

La documentación para este archivo está disponible en [errorlog.php](#)

```
1      <?php
2
3      /**
4       * Controlador Errorlog
5       *
6       * @package    SIIGS
7       * @subpackage Controlador
8       * @author     Pascual
9       * @created    2013-10-02
10      */
11     class Errorlog extends CI_Controller {
12
13         public function __construct()
14     {
15             parent::__construct();
16
17             if(!$this-> db-> conn_id) {
18                 $this-> template-> write('content', 'Error no se puede conectar a la Base de
Datos');
19                 $this-> template-> render();
20             }
21
22             $this-> load-> helper('url');
23         }
24
25         /**
26          * Lista todos los registros de la tabla error, con su correspondiente paginación
27          * permite hacer filtrados por Usuario, Entorno, Controlador, Acción y Fecha,
28          * muestra enlaces para ver detalles de un elemento específico
29          *
30          * @access public
31          * @param int    $pag Establece el desplazamiento del primer registro a devolver
32          * @return void
33          */
34         public function index($pag = 0)
35     {
36             if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url())) {
37                 show_error('', 403, 'Acceso denegado');
38                 return false;
39             }
40
41             if(!isset($this-> Errorlog_model))
42                 return false;
43
44             try {
45                 $this-> load-> library('pagination');
46                 $this-> load-> helper(array('form','formatFecha'));
47                 $this-> load-> model( array(DIR_SIIGS.'/usuario_model',
DIR_SIIGS.'/entorno_model', DIR_SIIGS.'/controlador_model', DIR_SIIGS.'/Accion_model') );
48
49                 $filtros = array();
50                 $data = array();
51
52                 $data['pag'] = $pag;
53                 $data['title'] = 'Error Log';
54
55                 /*** Inicia Campos para Filtros ***/
56                 $usuarios = $this-> usuario_model-> getOnlyActives();
57                 $data['usuarios'][0] = 'Todos';
58                 foreach ($usuarios as $user) {
59                     $data['usuarios'][$user-> id] = $user-> nombre.' '.$user-
> apellido_paterno.' '.$user-> apellido_materno;
60                 }
61
62                 $entornos = $this-> entorno_model-> getAll();
63                 $data['entornos'][0] = 'Todos';
64                 foreach ($entornos as $ent) {
```

```

65             $data['entornos'][$sent-> id] = $sent-> nombre;
66         }
67
68     $data['controladores'][0] = 'Todos';
69
70     if($this-> input-> post('entorno')) {
71         $controladores = $this-> controlador_model-> getByEntorno($this-> input-
72 > post('entorno'));
73         foreach ($controladores as $contr) {
74             $data['controladores'][$contr-> id] = $contr-> nombre;
75         }
76     $data['acciones'][0] = 'Todos';
77     $acciones = $this-> Accion_model-> getAll();
78     foreach ($acciones as $acci) {
79         $data['acciones'][$acci-> id] = $acci-> nombre;
80     }
81     /** Fin Campos para Filtros ***/
82
83     if($this-> input-> post('filtrar')) {
84         // Eliminar el campo hidden y el boton
85         unset($_POST['filtrar'], $_POST['btnFiltrar']);
86         $filtros = array_filter($this-> input-> post());
87
88         if(!empty($filtros)) {
89             foreach ($filtros as $campo => $valor) {
90                 switch ($campo) {
91                     case 'usuario':
92                         $this-> Errorlog_model-> addFilter('id_usuario', '=',
$valor);
93                         break;
94                     case 'fechaIni':
95                         $this-> Errorlog_model-> addFilter('fecha_hora', '>='
formatFecha($valor, "Y-m-d");
96                         break;
97                     case 'fechaFin':
98                         $this-> Errorlog_model-> addFilter('fecha_hora', '<='
formatFecha($valor, "Y-m-d");
99                         break;
100                    case 'entorno':
101                        $this-> Errorlog_model-> addFilter('id_entorno', '=',
$valor);
102                        break;
103                    case 'controlador':
104                        $this-> Errorlog_model-> addFilter('id_controlador', '=',
$valor);
105                        break;
106                    case 'accion':
107                        $this-> Errorlog_model-> addFilter('id_accion', '=',
$valor);
108                        break;
109                }
110            }
111        }
112
113        $data = array_merge($data, $filtros);
114    }
115    // Configuración para el Paginador
116    $configPag['base_url'] = site_url().DIR_SIIGS.'/errorlog/index/';
117    $configPag['first_link'] = 'Primero';
118    $configPag['last_link'] = '&Uacute;ltimo' ;
119    $configPag['uri_segment'] = 4;
120    $configPag['total_rows'] = $this-> Errorlog_model-> getNumRows();
121    $configPag['per_page'] = 50;
122
123    $this-> pagination-> initialize($configPag);
124
125    $data['registros'] = $this-> Errorlog_model-> getAll($configPag['per_page'],
$pag);
126 } catch (Exception $e) {
127     echo $e-> getTraceAsString();
128 }
129
130 $this-> template-> write_view('content',DIR_SIIGS.'/errorlog/index', $data);
131 $this-> template-> render();
132 }
133
134 /**
135 * Muestra los datos del registro especificado por el id
136 */

```

```

137     * @access public
138     * @param int      $id ID del elemento a actualizar
139     * @return void
140     */
141    public function view($id)
142    {
143        if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url())) {
144            show_error('', 403, 'Acceso denegado');
145            return false;
146        }
147
148        if(!isset($this-> Errorlog_model))
149            return false;
150
151        try {
152            $data['registro'] = $this-> Errorlog_model-> getById($id);
153            $data['title'] = 'Datos del error';
154
155            if( empty($data['registro']) ) {
156                $data['msgResult'] = 'ERROR: El registro de error solicitado no existe';
157                $data['clsResult'] = 'error';
158            }
159        } catch (Exception $e) { }
160
161        $this-> template-> write_view('content',DIR_SIIGS.'/errorlog/view', $data);
162        $this-> template-> write('menu','','true');
163        $this-> template-> write('sala_prensa','','true');
164        $this-> template-> render();
165    }
166
167}
168
169?>
```

# Archivo fuente para grupo.php

La documentación para este archivo está disponible en [grupo.php](#)

```
1  <?php
2  /**
3   * Controlador Grupo
4   *
5   * @package      SIIGS
6   * @subpackage   Controlador
7   * @author       Rogelio
8   * @created      2013-09-25
9   */
10  class Grupo extends CI_Controller {
11
12      public function __construct()
13      {
14          parent::__construct();
15          try{
16              $this-> load-> helper('url');
17              $this-> load-> model(DIR_SIIGS.'/Grupo_model');
18          }
19          catch(Exception $e){
20              $this-> template-> write('content', $e-> getMessage());
21              $this-> template-> render();
22          }
23      }
24
25      /**
26      * 1) Visualiza los grupos existentes para su interacción CRUD
27      * 2) En caso de detectar un texto a buscar se filtran los grupos existentes acorde a la
búsqueda
28      *
29      * @access      public
30      * @param       int      $pag     número de página a visualizar (paginación)
31      * @return      void
32      */
33      public function index($pag = 0)
34      {
35          try {
36              if (empty($this-> Grupo_model))
37                  return false;
38              if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
39                  show_error('', 403, 'Acceso denegado');
40              // validar permisos (buscar dinámicamente las acciones?)
41              $data['permisos'] = Usuario_model::checkCredentials(DIR_SIIGS.'::Permiso::index',
current_url());
42              $data['view'] = Usuario_model::checkCredentials(DIR_SIIGS.'::Grupo::view',
current_url());
43              $data['update'] = Usuario_model::checkCredentials(DIR_SIIGS.'::Grupo::update',
current_url());
44              $data['delete'] = Usuario_model::checkCredentials(DIR_SIIGS.'::Grupo::delete',
current_url());
45              $data['title'] = 'Catálogo de Grupos';
46              $this-> load-> helper('form');
47              $this-> load-> library('pagination');
48
49              $data['pag'] = $pag;
50              $data['msgResult'] = $this-> session-> flashdata('msgResult');
51              $data['clsResult'] = $this-> session-> flashdata('clsResult');
52
53              // Configuración para el Paginador
54              $configPag['base_url'] = '/'.DIR_SIIGS.'/grupo/index/';
55              $configPag['first_link'] = 'Primero';
56              $configPag['last_link'] = '&Uacute;ltim' ;
57              $configPag['uri_segment'] = '4';
58              $configPag['total_rows'] = $this-> Grupo_model-> getNumRows($this-> input-
> post('busqueda'));
59              $configPag['per_page'] = 15;
60              $this-> pagination-> initialize($configPag);
61              if ($this-> input-> post('busqueda'))
```

```

62             $data['groups'] = $this-> Grupo_model-> getAll($this-> input-
> post('busqueda'), $configPag['per_page'], $pag);
63         else
64             $data['groups'] = $this-> Grupo_model-> getAll('', $configPag['per_page'],
$pag);
65     }
66     catch(Exception $e){
67         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
68         $data['clsResult'] = 'error';
69     }
70     $this-> template-> write_view('content',DIR_SIIGS.'/grupo/index', $data);
71     $this-> template-> render();
72 }
73 /**
74 * Visualiza los datos del grupo recibido
75 *
76 * @access public
77 * @param int $id id del grupo a visualizar
78 * @return void
79 */
80 public function view($id)
81 {
82     try {
83         if (empty($this-> Grupo_model))
84             return false;
85         if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
86             show_error('', 403, 'Acceso denegado');
87         $data['title'] = 'Ver detalles de grupo';
88         $data['group_item'] = $this-> Grupo_model-> getById($id);
89         $data['entornos'] = $this-> Grupo_model-> getEntornosById($id);
90     }
91     catch(Exception $e){
92         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
93         $data['clsResult'] = 'error';
94     }
95     $this-> template-> write_view('content',DIR_SIIGS.'/grupo/view', $data);
96     $this-> template-> write('menu','','true');
97     $this-> template-> write('sala_prensa','','true');
98     $this-> template-> render();
99 }
100 /**
101 * 1) Prepara el formulario para la inserción de un grupo nuevo
102 * 2) Realiza las validaciones necesarias sobre cada campo del registro
103 *
104 * @access public
105 * @return void
106 */
107 public function insert(){
108     if (empty($this-> Grupo_model))
109         return false;
110     if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
111         show_error('', 403, 'Acceso denegado');
112     $data['title'] = 'Crear un nuevo grupo';
113     $this-> load-> helper('form');
114     $this-> load-> library('form_validation');
115     $this-> form_validation-> set_rules('nombre', 'Nombre',
116                                         'trim|xss_clean|required|callback__ifGroupExists|max_length[30]');
117     $this-> form_validation-> exists('descripcion', 'Descripcion',
118                                         'trim|xss_clean|max_length[100]');
119     if ($this-> form_validation-> run() === FALSE)
120     {
121         $this-> template-> write_view('content',DIR_SIIGS.'/grupo/insert', $data);
122         $this-> template-> render();
123     }
124     else
125     {
126         try {
127             $this-> Grupo_model-> setNombre($this-> input-> post('nombre'));
128             $this-> Grupo_model-> setDescripcion($this-> input-
> post('descripcion'));
129             $this-> Grupo_model-> insert();
130             $this-> session-> set_flashdata('msgResult', 'Registro agregado
exitosamente');
131             $this-> session-> set_flashdata('clsResult', 'success');
132             Bitacora_model::insert(DIR_SIIGS.'::'.__METHOD__, 'Grupo agregado:
'.strtoupper($this->
133             input-> post('nombre')));
134             redirect(DIR_SIIGS.'/grupo','refresh');
135         }

```

```

135         catch(Exception $e){
136             $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
137             $data['clsResult'] = 'error';
138             $this-> template-> write_view('content',DIR_SIIGS.'/grupo/insert', $data);
139             $this-> template-> render();
140         }
141     }
142 }
143 */
144 /**
145 * 1) Prepara el formulario para la modificación de un grupo existente
146 * 2) Realiza las validaciones necesarias sobre cada campo del registro
147 *
148 * @access public
149 * @param int $id id del grupo a modificar
150 * @return void
151 */
152 public function update($id)
153 {
154     if (empty($this-> Grupo_model))
155         return false;
156     if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
157         show_error('', 403, 'Acceso denegado');
158     $data['title'] = 'Modificar grupo';
159     $this-> load-> helper('form');
160     $this-> load-> library('form_validation');
161     $this-> form_validation-> set_rules('descripcion', 'Descripcion',
162                                         'trim|xss_clean|max_length[100]');
163     $data['group_item'] = $this-> Grupo_model-> getById($id);
164     if ($this-> form_validation-> run() === FALSE)
165     {
166         $this-> template-> write_view('content',DIR_SIIGS.'/grupo/update', $data);
167         $this-> template-> render();
168     }
169     else
170     {
171         try {
172             $this-> Grupo_model-> setId($id);
173             $this-> Grupo_model-> setDescripcion($this-> input-
> post('descripcion'));
174             $this-> Grupo_model-> update();
175             $this-> session-> set_flashdata('msgResult', 'Registro actualizado
exitosamente');
176             $this-> session-> set_flashdata('clsResult', 'success');
177             Bitacora_model::insert(DIR_SIIGS.'::'.__METHOD__, 'Grupo actualizado: '.$id);
178             redirect(DIR_SIIGS.'/grupo', 'refresh');
179         }
180         catch(Exception $e){
181             $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
182             $data['clsResult'] = 'error';
183             $this-> template-> write_view('content',DIR_SIIGS.'/grupo/update', $data);
184         }
185     }
186 }
187 */
188 /**
189 * Solicita la eliminación del grupo recibido
190 *
191 * @access public
192 * @param int $id id del grupo a eliminar
193 * @return void
194 */
195 public function delete($id)
196 {
197     try {
198         if (empty($this-> Grupo_model))
199             return false;
200         if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
201             show_error('', 403, 'Acceso denegado');
202         $this-> Grupo_model-> setId($id);
203         $this-> Grupo_model-> delete();
204         $this-> session-> set_flashdata('msgResult', 'Registro eliminado exitosamente');
205         $this-> session-> set_flashdata('clsResult', 'success');
206         Bitacora_model::insert(DIR_SIIGS.'::'.__METHOD__, 'Grupo eliminado: '.$id);
207     }
208     catch(Exception $e){
209         $this-> session-> set_flashdata('clsResult', 'error');
210         $this-> session-> set_flashdata('msgResult', Errorlog_model::save($e-
> getMessage(), __METHOD__));

```

```

211         }
212         redirect(DIR_SIIGS.'/grupo','refresh');
213     }
214
215     /**
216      * Callback para validar que un nombre de grupo no se duplique
217      *
218      * @access      public
219      * @param        string      $name      nombre del grupo a validar
220      * @return       boolean      false si el nombre del grupo ya existe, true
221      * si el nombre del grupo está disponible
222      */
223     public function _ifGroupExists($name) {
224         if (empty($this-> Grupo_model))
225             return false;
226         $is_exist = null;
227         try{
228             $is_exist = $this-> Grupo_model-> getByName($name);
229         } catch(Exception $e){
230         }
231         if ($is_exist) {
232             $this-> form_validation-> set_message(
233                 '_ifGroupExists', 'El nombre de grupo seleccionado ya existe, intente con
234                 otro.');
235             );
236             return false;
237         }
238         else
239         {
240             if (!$this-> Grupo_model-> getMsgError())
241                 return true;
242             else{
243                 $this-> form_validation-> set_message(
244                     '_ifGroupExists', $this-> Grupo_model-> getMsgError()
245                     );
246                 return false;
247             }
248         }
249     }

```

# Archivo fuente para menu.php

La documentación para este archivo está disponible en [menu.php](#)

```
1      <?php
2
3  /**
4   * Controlador Menu
5   *
6   * @package    SIIGS
7   * @subpackage Controlador
8   * @author     Pascual
9   * @created    2013-10-07
10  */
11 class Menu extends CI_Controller {
12     /**
13      * Guarda la instancia del objeto global CodeIgniter
14      * para utilizarlo en la función estática
15      *
16      * @access private
17      * @var    instance
18      */
19     private static $CI;
20
21     public function __construct()
22     {
23         parent::__construct();
24
25         self::$CI = & get_instance();
26
27         $this-> load-> model( DIR_SIIGS.'/Menu_model' );
28
29         if(!$this-> db-> conn_id) {
30             $this-> template-> write('content', 'Error no se puede conectar a la Base de
Datos');
31             $this-> template-> render();
32         }
33
34         $this-> load-> helper('url');
35     }
36
37     /**
38      * Lista todos los registros de la menu, con su correspondiente paginación
39      * permite hacer filtrados por Usuario, Entorno, Controlador, Acción y Fecha,
40      * permite eliminar un conjunto de registro o un elemento individual,
41      * muestra enlaces para actualizar y ver detalles de un elemento específico
42      *
43      * @access public
44      * @param int    $pag Establece el desplazamiento del primer registro a devolver
45      * @return void
46      */
47     public function index($pag = 0)
48     {
49         if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url())) {
50             show_error('', 403, 'Acceso denegado');
51             return false;
52         }
53
54         if(!isset($this-> Menu_model))
55             return false;
56
57         try {
58             $this-> load-> library(array('pagination', 'menubuilder'));
59             $this-> load-> helper('form');
60
61             $filtros = array();
62             $data = array();
63
64             $data['pag'] = $pag;
65             $data['msgResult'] = $this-> session->flashdata('msgResult');
66             $data['clsResult'] = $this-> session->flashdata('clsResult');
```

```

67         $data['title'] = 'Menu';
68
69         $registroEliminar = $this-> input-> post('registroEliminar');
70
71         if( !empty($registroEliminar) ) {
72             $this-> Menu_model-> delete($registroEliminar);
73             $data['clsResult'] = 'success';
74             $data['msgResult'] = 'Registros Eliminados exitosamente';
75         }
76
77         $menus = $this-> Menu_model-> getAll();
78         $data['menus'][0] = 'Elegir';
79         foreach ($menus as $men) {
80             $data['menus'][$men-> id] = $men-> nombre;
81         }
82
83         if($this-> input-> post('filtrar')) {
84             // Eliminar el campo hidden y el boton
85             unset($_POST['filtrar'], $_POST['btnFiltrar']);
86             $filtros = array_filter($this-> input-> post());
87
88             if(!empty($filtros)) {
89                 foreach ($filtros as $campo => $valor) {
90                     switch ($campo) {
91                         case 'raiz':
92                             $this-> Menu_model-> addFilter('id_raiz', '=', $valor);
93                             break;
94                     }
95                 }
96             }
97
98             $data = array_merge($data, $filtros);
99         }
100
101        // Configuración para el Paginador
102        $configPag['base_url'] = site_url().DIR_SIIGS.'/menu/index/';
103        $configPag['first_link'] = 'Primero';
104        $configPag['last_link'] = '&Uacute;ltimo' ;
105        $configPag['uri_segment'] = 4;
106        $configPag['total_rows'] = $this-> Menu_model-> getNumRows();
107        $configPag['per_page'] = 30;
108
109        $this-> pagination-> initialize($configPag);
110
111        $data['menuTree'] = Menubuilder::build(true);
112
113        // $data['registros'] = $this->Menu_model->getAll($configPag['per_page'],
114        $pag);
115        } catch (Exception $e) {
116            $data['clsResult'] = 'error';
117            $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
118        }
119        $this-> template-> write_view('content',DIR_SIIGS.'/menu/index', $data);
120        $this-> template-> render();
121    }
122
123    /**
124     * Muestra el formulario para crear un nuevo registro en la menu,
125     * las variables se obtienen por el metodo POST
126     *
127     * @access public
128     * @return void
129     */
130    public function insert($id = null)
131    {
132        if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url())) {
133            show_error('', 403, 'Acceso denegado');
134            return false;
135        }
136
137        if(!isset($this-> Menu_model))
138            return false;
139
140        try {
141            $this-> load-> helper('form');
142            $this-> load-> model( array(DIR_SIIGS.'/Entorno_model',
143 DIR_SIIGS.'/Controlador_model' ) );
144            $data['menuSeleccionado'] = $this-> Menu_model-> getById($id);

```

```

145
146     $menus = $this->  Menu_model->  getAll();
147     $data['menus'][0] = 'Elegir';
148     foreach ($menus as $men) {
149         $data['menus'][$men->  id] = $men->  nombre;
150     }
151
152     $entornos = $this->  Entorno_model->  getAll();
153     $data['entornos'][0] = 'Elegir';
154     foreach ($entornos as $ent) {
155         $data['entornos'][$ent->  id] = $ent->  nombre;
156     }
157
158     $entornos = $this->  Entorno_model->  getAll();
159     $data['entornos'][0] = 'Elegir';
160     foreach ($entornos as $ent) {
161         $data['entornos'][$ent->  id] = $ent->  nombre;
162     }
163
164     $data['controladores'][0] = 'Elegir';
165     if($this->  input->  post('entorno')) {
166         $controladores = $this->  Controlador_model->  getByEntorno($this->  input-
167 >  post('entorno'));
168         foreach ($controladores as $contr) {
169             $data['controladores'][$contr->  id] = $contr->  nombre;
170         }
171
172     $datos = $this->  input->  post();
173     $data['title'] = 'Menu - Crear un nuevo registro';
174     $data['msgResult'] = $this->  session->  flashdata('msgResult');
175
176     if(!empty($datos)) {
177         $this->  load->  library('form_validation');
178
179         $this->  form_validation->  set_rules('nombre', 'Nombre',
180 'trim|xss_clean|max_length[45]|required');
181
182         if ($this->  form_validation->  run() === true) {
183             if( !empty($datos['padre']) )
184                 $this->  Menu_model->  setId_padre($datos['padre']);
185
186             if( !empty($datos['raiz']) )
187                 $this->  Menu_model->  setId_raiz($datos['raiz']);
188
189             if( !empty($datos['controlador']) )
190                 $this->  Menu_model->  setId_controlador($datos['controlador']);
191
192             if( !empty($datos['ruta']) )
193                 $this->  Menu_model->  setRuta($datos['ruta']);
194
195             if( !empty($datos['atributo']) )
196                 $this->  Menu_model->  setAtributo($datos['atributo']);
197
198             $this->  Menu_model->  setNombre($datos['nombre']);
199
200             $this->  Menu_model->  insert();
201             $this->  session->  set_flashdata('msgResult', 'Registro guardado
exitosamente');
202
203             $this->  session->  set_flashdata('clsResult', 'success');
204
205             Bitacora_model::insert(DIR_SIIGS.'::'.__METHOD__, 'Registro creado: '. $this-
206 >  Menu_model->  getId());
206             redirect(DIR_SIIGS.'/menu/', 'refresh');
207             die();
208         }
209     } catch (Exception $e) {
210         $data['clsResult'] = 'error';
211         $data['msgResult'] = Errorlog_model::save($e->  getMessage(), __METHOD__);
212     }
213
214     $this->  template->  write_view('content',DIR_SIIGS.'/menu/insert', $data);
215     $this->  template->  render();
216 }
217 /**
218 * Muestra el formulario con los datos del registro especificado por el id,
219 * para actualizar sus datos
220 */

```

```

221     * @access public
222     * @param int      $id ID del elemento a actualizar
223     * @return void
224     */
225    public function update($id)
226    {
227        if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url())) {
228            show_error('', 403, 'Acceso denegado');
229            return false;
230        }
231
232        if(!isset($this-> Menu_model))
233            return false;
234
235        try {
236            $this-> load-> helper('form');
237            $this-> load-> model( array(DIR_SIIGS.'/Entorno_model',
238                DIR_SIIGS.'/Controlador_model' ) );
239
240            $datos = $this-> input-> post();
241            $data['title'] = 'Menu - Actualizar datos del registro';
242            $data['registro'] = $this-> Menu_model-> getById($id);
243
244            if(empty($data['registro']))
245                throw new Exception('Registro no encontrado');
246
247            $menus = $this-> Menu_model-> getAll();
248            $data['menus'][0] = 'Elegir';
249            foreach ($menus as $men) {
250                $data['menus'][$men-> id] = $men-> nombre;
251            }
252
253            $entornos = $this-> Entorno_model-> getAll();
254            $data['entornos'][0] = 'Elegir';
255            foreach ($entornos as $ent) {
256                $data['entornos'][$ent-> id] = $ent-> nombre;
257            }
258
259            $entornos = $this-> Entorno_model-> getAll();
260            $data['entornos'][0] = 'Elegir';
261            foreach ($entornos as $ent) {
262                $data['entornos'][$ent-> id] = $ent-> nombre;
263            }
264
265            $data['controladores'][0] = 'Elegir';
266            if($data['registro']-> id_entorno) {
267                $controladores = $this-> Controlador_model-> getByEntorno($data['registro']-
268                > id_entorno);
269                foreach ($controladores as $contr) {
270                    $data['controladores'][$contr-> id] = $contr-> nombre;
271                }
272
273                if(!empty($datos)) {
274                    $this-> load-> library('form_validation');
275
276                    $this-> form_validation-> set_rules('nombre', 'Nombre',
277                        'trim|xss_clean|max_length[45]|required');
278
279                    if ($this-> form_validation-> run() === true) {
280                        if( !empty($datos['padre']) )
281                            $this-> Menu_model-> setId_padre($datos['padre']);
282                        else
283                            $this-> Menu_model-> setId_padre(NULL);
284
285                        if( !empty($datos['raiz']) )
286                            $this-> Menu_model-> setId_raiz($datos['raiz']);
287                        else
288                            $this-> Menu_model-> setId_raiz(NULL);
289
290                        if( !empty($datos['controlador']) )
291                            $this-> Menu_model-> setId_controlador($datos['controlador']);
292                        else
293                            $this-> Menu_model-> setId_controlador(NULL);
294
295                        if( !empty($datos['ruta']) )
296                            $this-> Menu_model-> setRuta($datos['ruta']);
297
298                        if( !empty($datos['atributo']) )
299                            $this-> Menu_model-> setAtributo($datos['atributo']);

```

```

298             $this-> Menu_model-> setNombre($datos['nombre']);
299
300             $this-> Menu_model-> update($id);
301
302             Bitacora_model::insert(DIR_SIIGS.'::'.__METHOD__, 'Registro actualizado:
303 .' . $id);
304             $this-> session-> set_flashdata('msgResult', 'Registro guardado
305 exitosamente');
306             $this-> session-> set_flashdata('clsResult', 'success');
307             redirect(DIR_SIIGS.'/menu/', 'refresh');
308             die();
309         }
310     } catch (Exception $e) {
311         $data['clsResult'] = 'error';
312         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
313     }
314
315     $this-> template-> write_view('content', DIR_SIIGS.'/menu/update', $data);
316     $this-> template-> render();
317 }
318
319 /**
320 * Muestra los datos del registro especificado por el id
321 *
322 * @access public
323 * @param int    $id ID del elemento a actualizar
324 * @return void
325 */
326 public function view($id)
327 {
328     if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url())) {
329         show_error('', 403, 'Acceso denegado');
330         return false;
331     }
332
333     if (!isset($this-> Menu_model))
334         return false;
335
336     try {
337         $data['registro'] = $this-> Menu_model-> getById($id);
338         $data['title'] = 'Menu - Datos del registro';
339
340         if( empty($data['registro']) ) {
341             $data['clsResult'] = 'error';
342             $data['msgResult'] = 'ERROR: El registro solicitado no existe';
343         }
344     } catch (Exception $e) {
345         $data['clsResult'] = 'error';
346         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
347     }
348
349     $this-> template-> write_view('content', DIR_SIIGS.'/menu/view', $data);
350     $this-> template-> write('menu','','true');
351     $this-> template-> write('sala_prensa','','true');
352     $this-> template-> render();
353 }
354
355 /**
356 * Eliminar el registro especificado por el id
357 *
358 * @access public
359 * @param int    $id ID del elemento a eliminar
360 * @return void
361 */
362 public function delete($id)
363 {
364     if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url())) {
365         show_error('', 403, 'Acceso denegado');
366         return false;
367     }
368
369     if (!isset($this-> Menu_model))
370         return false;
371
372     try {
373         $this-> Menu_model-> delete($id);
374         $this-> session-> set_flashdata('clsResult', 'success');
375         $this-> session-> set_flashdata('msgResult', 'Registro eliminado exitosamente');

```

```
376     } catch (Exception $e) {
377         $this-> session-> set_flashdata('clsResult', 'error');
378         $this-> session-> set_flashdata('msgResult', Errorlog_model::save($e-
> getMessage(), __METHOD__));
379     }
380
381     Bitacora_model::insert(DIR_SIIGS.'::'.__METHOD__, 'Registro eliminado: '.$id);
382
383     redirect(DIR_SIIGS.'/menu/', 'refresh');
384     die();
385 }
386
387 }
388 ?>
```

# Archivo fuente para permiso.php

La documentación para este archivo está disponible en [permiso.php](#)

```
1  <?php
2  /**
3   * Controlador Permiso
4   *
5   * @package      SIIGS
6   * @subpackage   Controlador
7   * @author       Rogelio
8   * @created      2013-10-01
9   */
10  class Permiso extends CI_Controller {
11
12      public function __construct()
13  {
14          parent::__construct();
15          try{
16              $this-> load-> model(DIR_SIIGS.'/Permiso_model');
17          }
18          catch(Exception $e){
19              $this-> template-> write('content', $e-> getMessage());
20              $this-> template-> render();
21          }
22      }
23
24      /**
25      * 1) Visualiza los entornos existentes para su selección
26      * 2) Al seleccionar un entorno se obtienen los controladores_x_accion existentes y se
27      indica sobre cuales
28      * el grupo tiene permisos asignados
29      * 3) Elimina los permisos asignados al grupo anteriormente e inserta los asignados
30      * recientemente
31      * @access      public
32      * @param        int           $id      id del grupo del cual se obtendrán (y actualizar si
33      * aplica) los permisos asignados
34      * @return       void
35      */
36      public function index($id)
37  {
38          try
39  {
40              if (empty($this-> Permiso_model))
41                  return false;
42              $this-> load-> helper('url');
43              if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
44                  show_error('', 403, 'Acceso denegado');
45              $this-> load-> helper('form');
46              $this-> load-> library('form_validation');
47              $this-> load-> model(DIR_SIIGS.'/accion_model');
48              $this-> load-> model(DIR_SIIGS.'/entorno_model');
49
50              $data['msgResult'] = $this-> session-> flashdata('msgResult');
51              $data['clsResult'] = $this-> session-> flashdata('clsResult');
52              $data['title'] = 'Lista de Permisos disponibles';
53              $data['actions'] = $this-> accion_model-> getAll();
54              $arrEntornos = $this-> entorno_model-> getAll();
55              $data['entornos'][-1] = '-- Seleccione una opción --';
56              foreach ($arrEntornos as $entorno)
57  {
58                  $data['entornos'][$entorno-> id] = $entorno-> nombre;
59              }
60              $this-> form_validation-> set_rules('id_entorno', 'Entorno',
61 'required|is_natural_no_zero');
62              $this-> form_validation-> set_message('is_natural_no_zero', 'Debe seleccionar
un entorno válido');
63              $data['id_grupo'] = $id;
64              if ($this-> form_validation-> run() === FALSE)
65  {
```

```

63             $this-> template-> write_view('content',DIR_SIIGS.'/permiso/index', $data);
64             $this-> template-> render();
65         }
66     }
67     else
68     {
69         // se eliminan los permisos actuales
70         if ($this-> Permiso_model-> deletePermissions($this-> input-
71 post('id_entorno'), $id))
72         {
73             // si hay permisos seleccionados se guardan
74             if ($this-> input-> post('permisos'))
75             {
76                 $i = 0;
77                 $data = array();
78                 foreach ($this-> input-> post('permisos') as $permiso)
79                 {
80                     $data[$i] = array(
81                         'id_grupo' => $id ,
82                         'id_controlador_accion' => $permiso
83                     );
84                     $i++;
85                 }
86                 $this-> Permiso_model-> insertBatch($data);
87             }
88             $this-> session-> set_flashdata('msgResult', 'Registro actualizado
exitosamente');
89             $this-> session-> set_flashdata('clsResult', 'success');
90             redirect(DIR_SIIGS.'/grupo','refresh');
91         }
92     }
93     catch(Exception $e){
94         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
95         $data['clsResult'] = 'error';
96         $this-> template-> write_view('content',DIR_SIIGS.'/permiso/index', $data);
97         $this-> template-> render();
98     }
99 }
100 }
```

# Archivo fuente para raiz.php

La documentación para este archivo está disponible en [raiz.php](#)

```
1  <?php
2  /**
3   * Controlador Raiz
4   *
5   * @package    SIIGS
6   * @subpackage Controlador
7   * @author     Geovanni
8   * @created    2013-10-07
9   */
10
11 class Raiz extends CI_Controller {
12
13     public function __construct()
14     {
15         parent::__construct();
16
17         try
18         {
19             $this-> load-> helper('url');
20             $this-> load-> model(DIR_SIIGS.'/Raiz_model');
21             $this-> load-> model(DIR_SIIGS.'/ArbolSegmentacion_model');
22             $this-> load-> model(DIR_SIIGS.'/Catalogo_model');
23             $this-> load-> model(DIR_SIIGS.'/Catalogo_x_raiz_model');
24         }
25         catch (Exception $e)
26         {
27             $this-> template-> write("content" , $e-> getMessage());
28             $this-> template-> render();
29         }
30     }
31
32     /**
33      * Crea los archivos JSON necesarios para iniciar el ASU en caché y agilizar su carga
34      *
35      * @param type $id ID del asu
36      */
37
38     public function iniciarasu($id){
39         error_reporting(E_ALL);
40         try
41         {
42             if (($this-> ArbolSegmentacion_model-
> getChildrenFromLevel($id,1,array(),array(), array()) != "false"
43                         echo "true" ;
44             else
45                 echo "false" ;
46         }
47         catch (Exception $e)
48         {
49             Errorlog_model::save($e-> getMessage(), __METHOD__);
50             echo "false" ;
51         }
52         //echo json_encode($this->ArbolSegmentacion_model->getCluesFromId(781));
53     }
54
55     /**
56      *Acción por default del controlador, carga la lista
57      *de Raices disponibles y una lista de opciones
58      *No recibe parámetros
59      *
60      *@return void
61      */
62     public function index()
63     {
64         if (empty($this-> Raiz_model))
65             return false;
66         if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
67     }
```

```

67     show_error('', 403, 'Acceso denegado');
68
69     try
70     {
71
72         $data['title'] = 'Lista de raices disponibles';
73         $data['raices'] = $this-> Raiz_model-> getAll();
74         $data['msgResult'] = $this-> session-> flashdata('msgResult');
75         $data['clsResult'] = $this-> session-> flashdata('clsResult');
76     }
77     catch (Exception $e)
78     {
79         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
80         $data['clsResult'] = 'error';
81     }
82
83     $this-> template-> write_view('content',DIR_SIIGS.'/raiz/index', $data);
84     $this-> template-> render();
85 }
86
87 /**
88 *Acción para visualizar información de una raiz específica, obtiene el objeto
89 *raiz por medio del id proporcionado.
90 */
91 * @param int $id Este parametro no puede ser nulo
92 * @return void
93 */
94 public function view($id)
95 {
96     if (empty($this-> Raiz_model))
97         return false;
98         if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
99     show_error('', 403, 'Acceso denegado');
100
101    try
102    {
103        $data['title'] = "Detalles de la raiz";
104        $data['raiz_item'] = $this-> Raiz_model-> getById($id);
105    }
106    catch (Exception $e)
107    {
108        $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
109        $data['clsResult'] = 'error';
110    }
111
112    try
113    {
114        $data['catalogos'] = $this-> Catalogo_x_raiz_model-> getByArbol($id);
115    }
116    catch (Exception $e)
117    {
118        $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
119        $data['clsResult'] = 'error';
120    }
121
122    $this-> template-> write_view('content',DIR_SIIGS.'/raiz/view', $data);
123    $this-> template-> write('menu','','true');
124    $this-> template-> write('sala_prensa','','true');
125    $this-> template-> render();
126 }
127
128 /**
129 *Acción para preparar la inserción de nuevas acciones , realiza la validación
130 *del formulario del lado cliente
131 */
132 * @return void
133 */
134 public function insert()
135 {
136     if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
137     show_error('', 403, 'Acceso denegado');
138
139     $error = false;
140     $this-> load-> helper('form');
141     $this-> load-> library('form_validation');
142
143     $data['title'] = 'Crear una nueva raiz';
144     $this-> form_validation-> set_rules('descripcion', 'Descripción',
145     'trim|xss_clean|required|max_length[45]');
145

```

```

146     if ($this-> form_validation-> run() === FALSE)
147     {
148         $this-> template-> write_view('content',DIR_SIIGS.'/raiz/insert', $data);
149         $this-> template-> render();
150     }
151     else
152     {
153         try
154         {
155             $this-> load-> helper('url');
156
157             $this-> Raiz_model-> setDescripcion($this-> input-
158 > post('descripcion'));
159
160             $this-> Raiz_model-> insert();
161         }
162         catch (Exception $e)
163         {
164             $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
165             $data['clsResult'] = 'error';
166             $this-> template-> write_view('content',DIR_SIIGS.'/raiz/insert', $data);
167             $this-> template-> render();
168             $error = true;
169         }
170
171         if ($error == false)
172         {
173             $this-> session-> set_flashdata('msgResult', 'Registro insertado
correctamente');
174             $this-> session-> set_flashdata('clsResult', 'success');
175             redirect(DIR_SIIGS.'/raiz/index','refresh');
176         }
177     }
178 }
179 /**
180 *Acción para preparar la actualización de una raíz ya existente,
181 *recibe un ID para obtener los valores de esa raíz y mostrarlos
182 *en la vista update , realiza la validación del formulario del lado
183 *del cliente
184 *
185 * @param int $id
186 * @return void
187 */
188 public function update($id)
189 {
190     if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
191     show_error('', 403, 'Acceso denegado');
192
193     $this-> load-> helper('form');
194     $this-> load-> helper('url');
195     $this-> load-> library('form_validation');
196
197     $error = false;
198
199     $data['title'] = 'Modificar raíz';
200     $this-> form_validation-> set_rules('descripcion', 'Descripción',
201 'trim|xss_clean|required|max_length[45]');
202
203     if ($this-> form_validation-> run() === FALSE)
204     {
205         //obtiene el modelo
206         try
207         {
208             $data['raiz_item'] = $this-> Raiz_model-> getById($id);
209         }
210         catch (Exception $e)
211         {
212             $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
213             $data['clsResult'] = 'error';
214         }
215         //obtiene los catálogos relacionados a la raíz
216         try
217         {
218             $data['catalogos'] = $this-> Catalogo_x_raiz_model-> getByArbol($id);
219         }
220         catch (Exception $e)
221         {
222             $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
223             $data['clsResult'] = 'error';

```

```

223
224     }
225     //revisa si ya existen los registros de esta raiz en el arbol de segmentacion unica
226     try
227     {
228         $data['existe'] = $this-> Raiz_model-> ExistInArbol($id);
229     }
230     catch (Exception $e)
231     {
232         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
233         $data['clsResult'] = 'error';
234     }
235
236     $this-> template-> write_view('content',DIR_SIIGS.'/raiz/update', $data);
237     $this-> template-> render();
238 }
239 else
240 {
241     try
242     {
243         $this-> Raiz_model-> setDescripcion($this-> input->
244 > post('descripcion'));
245         $this-> Raiz_model-> setId($this-> input-> post('id'));
246
247         $this-> Raiz_model-> update();
248     }
249     catch (Exception $e)
250     {
251         try
252         {
253             $data['raiz_item'] = $this-> Raiz_model-> getById($id);
254         }
255         catch (Exception $e)
256         {
257             $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
258             $data['clsResult'] = 'error';
259         }
260
261         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
262         $data['clsResult'] = 'error';
263         $this-> template-> write_view('content',DIR_SIIGS.'/raiz/update', $data);
264         $this-> template-> render();
265
266         $error = true;
267     }
268
269     if ($error == false)
270     {
271         $this-> session-> set_flashdata('msgResult', 'Registro actualizado
correctamente');
272         $this-> session-> set_flashdata('clsResult', 'success');
273     }
274 }
275 }
276 /**
277 *
278 *Acción para eliminar una raiz, recibe el id de la raiz a eliminar
279 *
280 * @param int $id
281 * @return void
282 */
283
284 public function delete($id)
285 {
286     if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
287     show_error('', 403, 'Acceso denegado');
288
289     try
290     {
291         if (empty($this-> Raiz_model))
292             return false;
293
294         $this-> load-> helper('url');
295         $this-> Raiz_model-> setId($id);
296         $this-> Raiz_model-> delete();
297         $this-> session-> set_flashdata('msgResult', 'Registro eliminado exitosamente');
298         $this-> session-> set_flashdata('clsResult', 'success');
299     }
300     catch(Exception $e)

```

```

301         {
302             $this-> session-> set_flashdata('msgResult', Errorlog_model::save($e-
303             > getMessage(), __METHOD__));
304             $this-> session-> set_flashdata('clsResult', 'error');
305         }
306         redirect(DIR_SIIGS.'/raiz','refresh');
307     }
308     /**
309      *
310      *Acción para crear el ASU a partir de una raiz
311      *Solo se permite su acceso por medio de peticiones AJAX
312      *
313      * @param int $id
314      * @return void
315      */
316     public function createasu($id)
317     {
318         if ($this-> input-> is_ajax_request())
319         {
320             if ($this-> db-> query("select count(*) as count from
321             asu_arbol_segmentacion where id_raiz='".$id)-> result()[0]-> count> 0)
322             {
323                 echo "El arbol ya ha sido creado anteriormente";
324                 die();
325             }
326             try
327             {
328                 ini_set('max_execution_time',10000);
329                 ini_set('memory_limit', '-1');
330                 $catalogos = $this-> Catalogo_x_raiz_model-> getByArbol($id);
331
332                 foreach ($catalogos as $item) {
333
334                     $iditem = $item-> id;
335                     $tabla = $item-> tabla_catalogo;
336                     $nivel = $item-> grado_segmentacion;
337                     $llave = $item-> nombre_columna_llave;
338                     $descripcion = $item-> nombre_columna_descripcion;
339
340                     $descripcion = explode('+', $descripcion);
341                     foreach ($descripcion as $item => $valor)
342                         $descripcion[$item] = $tabla.".".$valor;
343
344                     if (count($descripcion)> 1)
345                         $descripcion = implode(", ", $descripcion);
346                     else
347                         $descripcion = $descripcion[0];
348
349                     $descripcion = 'concat('. $descripcion.')';
350
351                     if ($nivel > 1)
352                     {
353                         $consulta = "select
354                         ".$tabla.".".$llave." as llave, ";
355                         $consulta .=
356                         $descripcion." as descripcion, ";
357                         $consulta .= "asu_arbol_segmentacion.id as padre
358                         ";
359
360                         $padre = $this-> Catalogo_x_raiz_model-
361                         $relaciones = $this-> Catalogo_x_raiz_model-
362
363                         $consulta .= " join " . $padre-
364                         foreach ($relaciones as $relacion)
365                         {
366                             $consulta .= " and
367                             ".$relacion-> columna_hijo." = " . $padre-
368                             $relacion-> columna_padre;
369                         }
370                         $consulta .= " join asu_arbol_segmentacion on
371                         $consulta .= " and
372                         $consulta .= " and
373                         ".$padre-> nombre."." . $padre-> llave;
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
779
780
781
782
783
784
785
786
787
788
789
789
790
791
792
793
794
795
796
797
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
818
819
819
820
821
822
823
824
825
826
827
828
829
829
830
831
832
833
834
835
836
837
838
839
839
840
841
842
843
844
845
846
847
848
849
849
850
851
852
853
854
855
856
857
858
859
859
860
861
862
863
864
865
866
867
868
869
869
870
871
872
873
874
875
876
877
878
879
879
880
881
882
883
884
885
886
887
888
889
889
890
891
892
893
894
895
896
897
898
899
899
900
901
902
903
904
905
906
907
908
909
909
910
911
912
913
914
915
916
917
918
919
919
920
921
922
923
924
925
926
927
928
929
929
930
931
932
933
934
935
936
937
938
939
939
940
941
942
943
944
945
946
947
948
949
949
950
951
952
953
954
955
956
957
958
959
959
960
961
962
963
964
965
966
967
968
969
969
970
971
972
973
974
975
976
977
978
979
979
980
981
982
983
984
985
986
987
988
989
989
990
991
992
993
994
995
996
997
998
999
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2289
2290
2291
2292
2293
2294
2295
2296
2297
2297
2298
2299
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2429
2430
2431
2432
2433
2434
2435
```

```

368
369             $filas = $this-> db-> query($consulta);
370
371             $datosdump = array();
372             foreach ($filas-> result() as $value) {
373
374                 array_push($datosdump, array(
375                     'grado_segmentacion' => $nivel,
376                     'id_raiz'=> $id,
377                     'id_padre' => $value-> padre,
378                     'id_tabla_original' => $value-> llave,
379                     'orden' => '0',
380                     'visible' => 'true',
381                     'descripcion' => $value-> descripcion
382                 )));
383             }
384             if (count($datosdump)> 0)
385                 if ($this-> db-
386                 {
387                     echo 'false';
388                 }
389             }
390             else
391             {
392                 $consulta = "select " . $llave." as llave,
393
394                 $consulta .= $descripcion." as descripcion "
395                 $consulta .= " from ". $tabla;
396                 $filas = $this-> db-> query($consulta);
397                 $datosdump = array();
398                 foreach ($filas-> result() as $key => $value) {
399
400                     array_push($datosdump, array(
401                         'grado_segmentacion' => $nivel,
402                         'id_raiz'=> $id,
403                         'id_padre' => '0',
404                         'id_tabla_original' => $value-> llave,
405                         'orden' => '0',
406                         'visible' => 'true',
407                         'descripcion' => $value-> descripcion
408                     )));
409             }
410             if (count($datosdump)> 0)
411                 if ($this-> db-
412                 {
413                     echo 'false';
414                 }
415                 echo "true" ;
416             }
417         }
418     }
419     catch(Exception $e)
420     {
421         echo Errorlog_model::save($e-> getMessage(), __METHOD__);
422     }
423     else
424     {
425         echo "Acceso denegado" ;
426     }
427 }
428 */
429 /**
430 * Acción para actualizar el ASU a partir de una raíz
431 * Solo se permite su acceso por medio de peticiones AJAX
432 *
433 * @param int $id
434 * @return void
435 */
436
437
438     public function updateasu($id)
439     {
440         if ($this-> input-> is_ajax_request())
441         {
442             try
443             {

```

```

445         ini_set('max_execution_time',10000);
446         ini_set('memory_limit', '-1');
447         $catalogos = $this-> Catalogo_x_raiz_model-> getByArbol($id);
448         $nuevos_registros = array();
449         $cambios_registros = array();
450         foreach ($catalogos as $item) {
451
452             $iditem = $item-> id;
453             $tabla = $item-> tabla_catalogo;
454             $snivel = $item-> grado_segmentacion;
455             $llave = $item-> nombre_columna_llave;
456             $descripcion = $item-> nombre_columna_descripcion;
457
458             $descripcion = explode('+', $descripcion);
459             foreach ($descripcion as $item => $valor)
460                 $descripcion[$item] = "a." . $valor;
461
462             if (count($descripcion)> 1)
463                 $descripcion = implode(", ", $descripcion);
464             else
465                 $descripcion = $descripcion[0];
466
467             $descripcion = 'concat('. $descripcion . ')';
468
469             //obtener todos los ID que ya estan agregados al ASU
470             //correspondiente al nivel actual
471             $filas = $this-> db-> query("select
472             concat('\'',id_tabla_original,'\'') as llave from asu_arbol_segmentacion where
473             id_raiz=" . $id . " and grado_segmentacion=" . $snivel);
474             $get_array = function($val)
475             {
476                 return $val['llave'];
477             };
478             $datos = array_map($get_array,$filas-> result_array());
479
480             if ($nivel> 1)
481             {
482                 //obtener las relaciones para el arbol
483                 $padre = $this-> Catalogo_x_raiz_model-
484                 > getRelations($iditem);
485
486                 //crear la consulta basica
487                 $consulta_base = "select a." . $llave . " as
488                 llave, " . $descripcion . " as descripcion, " . $padre . "
489                 > llave." as padre from "
490
491                 > nombre." on l=1" ;
492
493                 foreach ($relaciones as $relacion)
494                 {
495                     $consulta_base .= " and a." . $relacion-> columna_padre;
496
497                     . $padre-> nombre . ". " . $relacion-> columna_padre;
498
499                     else
500
501                     {
502                         $consulta_base = "select a." . $llave . " as
503                         llave, " . $descripcion . " as descripcion , 0 as padre" ;
504
505                         $consulta_base .= " from " . $tabla . " a" ;
506
507                         $consulta_nuevos = $consulta_base . " where
508                         . implode(',', $datos). "")" ;
509
510                         //Nuevos registros en los catalogos para agregar al ASU
511                         $resultado = $this-> db-> query($consulta_nuevos);
512                         if ($resultado && $resultado-> num_rows()> 0)
513                         {
514                             foreach($resultado-> result_array() as $fila)
515                                 array_push($nuevos_registros, array(
516                                     'id_raiz' => $id, 'grado_segmentacion' => $snivel,
517                                     'id_padre' =>
518                                     $fila["padre"] , 'id_tabla_original' => $fila["llave"]
519                                     , 'orden' => 0, 'visible' => '1', 'descripcion' =>
520                                     $fila["descripcion"]
521                                     ));
522                         }
523                     }
524                 }
525             }
526         }
527     }
528
529     $nuevos_registros = array();
530
531     foreach ($nuevos_registros as $item)
532     {
533         $nuevos_registros[] = array(
534             'id_raiz' => $id, 'grado_segmentacion' => $snivel,
535             'id_padre' =>
536             $fila["padre"] , 'id_tabla_original' => $fila["llave"]
537             , 'orden' => 0, 'visible' => '1', 'descripcion' =>
538             $fila["descripcion"]
539             );
540     }
541
542     return $nuevos_registros;
543 }

```

```

512
513
514
515
>    getByNivel($id,$nivel-1);
516
>    getRelations($iditem);
517
518
a."      .$llave." as llave, "
519
asu_arbol_segmentacion where id_raiz="
1)." and id_tabla_original = "
padre from "      .$tabla." a"
520
521
>    nombre." on l=1" ;
522
523
524
>    columna_hijo." = "      .$padre-
525
526
527
asu_arbol_segmentacion b on b.id_raiz="
and b.id_tabla_original = a."      .$llave." where ( b.descripcion <>
".$descripcion." or b.id_padre <> (select id from asu_arbol_segmentacion where
id_raiz="      .$id." and grado_segmentacion = "      .($nivel-1)." and id_tabla_original =
"      .$padre->  nombre.".")      .$padre->  llave.") )" ;
528
529
530
531
a."      .$llave." as llave, "
532
533
asu_arbol_segmentacion b on b.id_raiz="
and b.id_tabla_original = a."      .$llave." and ( b.descripcion <>
".$descripcion.")" ;
534
535
536
537
538
539
540
541
542
543
544
$fila["padre"      , 'id_tabla_original' => $fila["llave"]      , 'orden' => 0, 'visible' => '1', 'descripcion' =>
$fila["descripcion"      ]
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
if ($nivel> 1)
{
    //obtener las relaciones para el arbol
    $padre = $this-> Catalogo_x_raiz_model-
    $relaciones = $this-> Catalogo_x_raiz_model-
        //crear la consulta basica
        $consulta_base = "select b.id as id,
        $descripcion." as descripcion, "      ;
        $consulta_base .= "(select id from
        .$id." and grado_segmentacion = "      .($nivel-
        .$padre->  nombre.".")      .$padre->  llave.") as
        ;
        //crear las relaciones
        $consulta_base .= " join "      .$padre-
            foreach ($relaciones as $relacion)
            {
                $consulta_base .= " and a."      .$relacion->  columna_padre;
            }
        $consulta_modificaciones = $consulta_base." join
        .$id." and b.grado_segmentacion="      .$nivel."      ;
        and b.id_tabla_original = a."      .$llave." where ( b.descripcion <>
        ".$descripcion." or b.id_padre <> (select id from asu_arbol_segmentacion where
        id_raiz="      .$id." and grado_segmentacion = "      .($nivel-1)." and id_tabla_original =
        "      .$padre->  nombre.".")      .$padre->  llave.") )" ;
    }
    else
    {
        $consulta_base = "select b.id as id,
        $descripcion." as descripcion , 0 as padre"      ;
        $consulta_base .= " from "      .$tabla." a"
        $consulta_modificaciones = $consulta_base." join
        .$id." and b.grado_segmentacion="      .$nivel."      ;
        and b.id_tabla_original = a."      .$llave." and ( b.descripcion <>
        ".$descripcion.")" ;
    }
}

//Nuevos registros en los catalogos para agregar al ASU
$resultado = $this-> db-> query($consulta_modificaciones);
if ($resultado && $resultado-> num_rows()> 0)
{
    foreach($resultado-> result_array() as $fila)
        array_push($cambios_registros, array(
            'id' => $fila['id'],
            'id_raiz' => $id, 'grado_segmentacion' => $nivel,
            'id_padre' =>
            $fila["padre"      , 'id_tabla_original' => $fila["llave"]      , 'orden' => 0, 'visible' => '1', 'descripcion' =>
            $fila["descripcion"      ]
        ));
    if (count($nuevos_registros)> 0)
    {
        if ($this-> db-
> insert_batch('asu_arbol_segmentacion',$nuevos_registros) != 1)
            echo "true" ;
        else
            echo "false" ;
    }
    else
    {
        echo "true" ;
    }
    if(count($cambios_registros)> 0)
    {
        if ($this-> db-
> update_batch('asu_arbol_segmentacion',$cambios_registros,'id') != 1)
            echo "true" ;
        else
            echo "false" ;
    }
    else
    {
        echo "true" ;
    }
}

```

```

571     }
572     catch(Exception $e)
573     {
574         echo Errorlog_model::save($e-> getMessage(), __METHOD__);
575     }
576     else
577     {
578         echo "Acceso denegado";
579     }
580 }
581
582
583
584
585
586 /**
587 * Accion para regresar la descripción e informacion adicional de un arreglo
588 * de ID's desde el arbol de segmentacion
589 *
590 * @param Array $claves Este parametro es pasado por POST y es la lista de valores a
591 consultar
592 * @param Int $desglose parametro pasado por POST y determina si se requiere informacion
593 *
594 * @return Object JSON con la informacion requerida
595 */
596 public function getDataTreeFromId()
597 {
598     try
599     {
600         if ($this-> input-> is_ajax_request())
601     {
602             $claves = $this-> input-> post('claves');
603             $desglose = $this-> input-> post('desglose');
604             // $claves = array(775,776);
605             // $claves = array(895,896);
606             // $desglose = 2;
607             if ($claves)
608                 echo json_encode($this-> ArbolSegmentacion_model-
609 getDescripcionById($claves,$desglose));
610             else
611                 echo "Parametros incorrectos";
612         }
613         else echo 'Acceso denegado';
614     }
615     catch(Exception $e)
616     {
617         echo $e-> getMessage();
618     }
619 }
620
621 /**
622 * Accion para regresar el arbol de segmentacion determinado, el objeto regresado
623 * estructura de arbol y es consumida solamente por peticiones AJAX
624 *
625 * @param Int $idarbol parametro pasado por POST y determina el arbol a consultar
626 * @param Int $nivel parametro pasado por POST y determina el nivel superior a
627 desglosar en el arbol
628 * @param Array $claves Este parametro es pasado por POST y es la lista de niveles a
629 omitir en el arbol
630 * @param Array $claves Este parametro es pasado por POST y es la lista de valores a
631 preseleccionar en el arbol
632 *
633 * @return Object JSON con la informacion requerida
634 */
635
636 public function getChildrenFromLevel()
637 {
638     try
639     {
640         if ($this-> input-> is_ajax_request())
641     {
642             $idarbol = $this-> input-> post('idarbol');
643             $nivel = $this-> input-> post('nivel');
644             $omitidos = $this-> input-> post('omitidos');
645             $seleccionados = $this-> input-> post('seleccionados');
646             $seleccionables = array();
647     }
648 }
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
999

```

```

644             if (($this-> input-> post('seleccionables')))
645             $seleccionables = $this-> input-> post('seleccionables');
646             $idarbol = 1;
647             $nivel = 1;
648             $omitidos = array(null);
649             $seleccionados = array(775,776);
650             if ($idarbol && $nivel && $omitidos &&
651                 $seleccionados)
652             {
653                 echo json_encode($this-> ArbolSegmentacion_model-
654 > getChildrenFromLevel($idarbol,$nivel,$omitidos,$seleccionados,$seleccionables),JSON_UNESCAPED_UNICODE);
655             }
656             else
657                 echo "Parámetros incorrectos";
658             }
659             catch(Exception $e)
660             {
661                 echo $e-> getMessage();
662             }
663         }
664     }
665     /**
666      * Sirve para obtener bloques del arbol de segmentación única ASU
667      * solo se puede acceder por peticiones AJAX, los parametros son pasados por GET
668      * @param int $idarbol ID del arbol (el arbol usado por la TES es el 1)
669      * @param int $nivel nivel del arbol que se desea obtener
670      * @param array $seleccionados se especifica si dentro del arreglo de retorno, hay
671      * valores preseleccionados
672      * @param bool $seleccionable especifica si los elementos del arbol pueden ser
673      * seleccionados
674      * @param array $seleccionables especifica que niveles del arbol pueden ser
675      * seleccionados
676      * @param array $omitidos especifica niveles omitidos dentro del arbol (Si hay un nivel
677      * intermedio omitido, los hijos de este nivel son agregados como hijos de su nivel inmediato superior)
678      * @return Object JSON
679     */
680     public function getTreeBlock()
681     {
682         try
683         {
684             if ($this-> input-> is_ajax_request() || true)
685             {
686                 $seleccionables = array();
687                 $omitidos = array();
688                 $seleccionados = array();
689                 $idarbol=0;
690                 $nivel=0;
691                 $elejido=0;
692
693                 if (($this-> input-> get('idarbol',TRUE)))
694                     $idarbol = $this-> input-> get('idarbol',TRUE);
695                 if (($this-> input-> get('nivel',TRUE)))
696                     $nivel = $this-> input-> get('nivel',TRUE);
697                 if (($this-> input-> get('elejido',TRUE)))
698                     $elejido = $this-> input-> get('elejido',TRUE);
699                 if (($this-> input-> get('omitidos',TRUE)))
700                     $omitidos = $this-> input-> get('omitidos',TRUE);
701                 if (($this-> input-> get('seleccionados',TRUE)))
702                     $seleccionados = $this-> input-> get('seleccionados',TRUE);
703                 if (($this-> input-> get('seleccionable',TRUE)))
704                     $seleccionable = $this-> input-> get('seleccionable',TRUE);
705                 if (($this-> input-> get('seleccionables',TRUE)))
706                     $seleccionables = $this-> input-> get('seleccionables',TRUE);
707
708                 // $idarbol = 1;
709                 // $nivel = 1;
710                 $seleccionable = (($seleccionable == 'true') ? true : false);
711
712                 // $seleccionables = array(1);
713                 // $omitidos = array(null);
714                 // $seleccionados = array(775,776);
715                 if ($idarbol> 0 && ($nivel> 0 || $elejido> 0))
716                 {
717                     echo json_encode($this-> ArbolSegmentacion_model-
718 > getTreeBlock($idarbol,$nivel,$seleccionados,$seleccionable,$elejido,$omitidos,$seleccionables),JSON_UNESCAPED_UNICODE);

```

```

715
716
717
718
719
720
721
722
723
724
725
726
727
728
de filtro
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
    }
    else
        echo "Parámetros incorrectos" ;
    }
    else echo 'Acceso denegado';
}
catch(Exception $e)
{
echo $e-> getMessage();
}
}

/**
 * Accion para obtener los registros de un ASU determinado en cierto nivel y con un ID
 * @param int $idarbol
 * @param Int $nivel Nivel de desglose de información requerida
 * @param Int $filtro (Opcional) filtrar por un valor determinado
 *
 * @return Object
 * @throws Exception Si ocurre error al recuperar datos de la base de datos
 */
public function getDataKeyValue($idarbol,$nivel,$filtro = 0)
{
    if (!$this-> input-> is_ajax_request())
        show_error('', 403, 'Acceso denegado');

    echo json_encode($this-> ArbolSegmentacion_model-
>getDataKeyValue($idarbol,$nivel,$filtro));
}

/*
public function prueba ($id)
{
    header('Content-type: application/json; charset=utf-8');
    echo json_encode($this->ArbolSegmentacion_model-
>getChildrenFromLevel($id,1,array(null),array(null)),JSON_UNESCAPED_UNICODE);
    //echo json_encode($this->ArbolSegmentacion_model->getCluesFromId(20063));
}
*/
}

```

# Archivo fuente para reglavaduna.php

La documentación para este archivo está disponible en [reglavaduna.php](#)

```
1  <?php
2  /**
3   * Controlador ReglaVacuna
4   *
5   * @package    SIIGS
6   * @subpackage Controlador
7   * @author     Geovanni
8   * @created    2013-12-09
9   */
10  class ReglaVacuna extends CI_Controller {
11
12      public function __construct()
13      {
14          parent::__construct();
15
16          try
17          {
18              $this-> load-> helper('url');
19              $this-> load-> model(DIR_SIIGS.'/ReglaVacuna_model');
20          }
21          catch (Exception $e)
22          {
23              $this-> template-> write("content" , $e-> getMessage());
24              $this-> template-> render();
25          }
26      }
27
28      /**
29       * Acción por default del controlador, carga la lista
30       * de reglas de vacunas disponibles y una lista de opciones
31       * No recibe parámetros
32       *
33       * @return void
34       */
35      public function index()
36      {
37          if (empty($this-> ReglaVacuna_model))
38              return false;
39              if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
40                  show_error('', 403, 'Acceso denegado');
41          try
42          {
43
44              $data['title'] = 'Lista de reglas para vacunas';
45              $data['reglas'] = $this-> ReglaVacuna_model-> getAll();
46              $data['msgResult'] = $this-> session-> flashdata('msgResult');
47              $data['clsResult'] = $this-> session-> flashdata('clsResult');
48          }
49          catch (Exception $e)
50          {
51              $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
52              $data['clsResult'] = 'error';
53          }
54
55          $this-> template-> write_view('content',DIR_SIIGS.'/reglavaduna/index', $data);
56          $this-> template-> render();
57      }
58
59      /**
60       * Acción para visualizar información de una regla específica, obtiene el objeto
61       * regla_vacuna por medio del id proporcionado.
62       *
63       * @param int $id Este parametro no puede ser nulo
64       * @return void
65       */
66      public function view($id)
67      {
```

```

68      if (empty($this-> ReglaVacuna_model))
69          return false;
70          if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
71      show_error('', 403, 'Acceso denegado');
72  try
73  {
74      $data['title'] = "Detalles de la regla";
75      $data['regla_item'] = $this-> ReglaVacuna_model-> getById($id);
76  }
77  catch (Exception $e)
78  {
79      $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
80      $data['clsResult'] = 'error';
81  }
82
83      $this-> template-> write_view('content',DIR_SIIGS.'/reglavacuna/view', $data);
84      $this-> template-> write('menu','','true');
85      $this-> template-> write('sala_prensa','','true');
86      $this-> template-> render();
87  }
88
89 /**
90 *Acción para preparar la inserción de nuevas reglas , realiza la validación
91 *del formulario del lado cliente
92 */
93 *@return void
94 */
95 public function insert()
96 {
97     if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
98     show_error('', 403, 'Acceso denegado');
99     $error = false;
100    $this-> load-> helper('form');
101    $this-> load-> library('form_validation');
102
103    $data['title'] = 'Crear una nueva regla para vacuna';
104    $this-> form_validation-> set_rules('aplicacion_inicio', 'Inicio aplicación',
105 'trim|xss_clean|required|is_natural');
106    $this-> form_validation-> set_rules('aplicacion_fin', 'Fin aplicación',
107 'trim|xss_clean|required|is_natural_no_zero');
108    $this-> form_validation-> set_rules('id_vacuna', 'Vacuna',
109 'trim|xss_clean|required|is_natural_no_zero');
110    $this-> form_validation-> set_rules('id_via_vacuna', 'Vía Vacuna',
111 'trim|xss_clean|required|is_natural_no_zero');
112    $this-> form_validation-> set_rules('dosis', 'Dosis',
113 'trim|xss_clean|decimal');
114    $this-> form_validation-> set_rules('region', 'Región',
115 'trim|xss_clean|max_length[90]');
116    $this-> form_validation-> set_rules('observacion_region', 'Observación de la
117 'región', 'trim|xss_clean|max_length[450]');
118    //$/this->form_validation->set_rules('tipo_aplicacion', 'Tipo de
119 'aplicación', 'trim|xss_clean|required');
120
121    $vacunas = $this-> db-> query("select id,descripcion from cns_vacuna
122 where activo=1" )-> result();
123    $data['vacunas']['''] = 'Elige una vacuna';
124    foreach ($vacunas as $item) {
125        $data['vacunas'][$item-> id] = $item-> descripcion;
126    }
127
128    $vias_vacuna = $this-> db-> query("select id,descripcion from
129 cns_via_vacuna where activo=1" )-> result();
130    $data['vias_vacuna']['''] = 'Elige una vía de vacuna';
131    foreach ($vias_vacuna as $item) {
132        $data['vias_vacuna'][$item-> id] = $item-> descripcion;
133    }
134
135    $orden = $this-> db-> query("SELECT a.id FROM (SELECT 1 AS id UNION
136 SELECT 2 AS id UNION SELECT 3 AS id UNION SELECT 4 AS id UNION SELECT 5 AS id UNION SELECT 6 AS id
137 UNION SELECT 7 AS id
138 UNION SELECT 8 AS id UNION SELECT 9 AS id UNION SELECT 10 AS id UNION SELECT 11 AS id UNION
139 SELECT 12 AS id UNION SELECT 13 AS id UNION SELECT 14 AS id UNION SELECT 15 AS id
140 UNION SELECT 16 AS id UNION SELECT 17 AS id UNION SELECT 18 AS id UNION SELECT 19 AS id UNION
141 SELECT 20 AS id
142 UNION SELECT 21 AS id UNION SELECT 22 AS id UNION SELECT 23 AS id UNION SELECT 24 AS id
143 UNION SELECT 25 AS id UNION SELECT 26 AS id
144 UNION SELECT 27 AS id UNION SELECT 28 AS id UNION SELECT 29 AS id UNION SELECT 30 AS id) AS
145 a WHERE a.id NOT IN (SELECT orden_esq_com FROM cns_regla_vacuna WHERE orden_esq_com IS NOT
146 NULL)" )-> result();
130    $data['orden']['''] = 'Orden de la vacuna';

```

```

131     foreach ($orden as $item) {
132         $data['orden'][$item-> id] = $item-> id;
133     }
134
135     $alergias = $this-> db-> query("select id,descripcion from cns_alergia
where activo=1");
136     $data['alergias'] = $alergias;
137
138     if ($this-> form_validation-> run() === FALSE)
139     {
140         $this-> template-> write_view('content',DIR_SIIGS.'/reglavaduna/insert',$data);
141         $this-> template-> render();
142     }
143     else
144     {
145         try
146         {
147             $this-> load-> helper('url');
148
149             $this-> ReglaVacuna_model-> setIdVacuna($this-> input-
> post('id_vacuna'));
150             $this-> ReglaVacuna_model-> setIdVacunaPrevia($this-> input-
> post('id_vacuna_previa'));
151             //if ($this->input->post('tipo_aplicacion') == 'nacimiento')
152             //{
153                 $this-> ReglaVacuna_model-> setDiaInicioNacido($this-
> input-> post('aplicacion_inicio'));
154                 $this-> ReglaVacuna_model-> setDiaFinNacido($this-
> input-> post('aplicacion_fin'));
155                 //}
156             //
157             //
158             //
159             //
160             //
161             $this-> ReglaVacuna_model-> setIdViaVacuna($this-> input-
> post('id_via_vacuna'));
162             $this-> ReglaVacuna_model-> setDosis($this-> input-
> post('dosis'));
163             $this-> ReglaVacuna_model-> setRegion($this-> input-
> post('region'));
164             $this-> ReglaVacuna_model-> setObservacionRegion($this-
> input-> post('observacion_region'));
165             $this-> ReglaVacuna_model-> setAlergias($this-> input-
> post('alergias'));
166             $this-> ReglaVacuna_model-> setEsqComp((($this-> input-
> post('esq_com')) == false) ? false : true);
167             $this-> ReglaVacuna_model-> setForzarAplicacion((($this-
> input-> post('forzar_aplicacion')) == false) ? false : true));
168
169             if ($this-> input-> post('esq_com') == true)
170             {
171                 $this-> ReglaVacuna_model-> setOrdenEsqComp($this-
> input-> post('orden_esq_com'));
172             }
173
174             $this-> ReglaVacuna_model-> insert();
175         }
176         catch (Exception $e)
177         {
178             $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
179             $data['clsResult'] = 'error';
180             $this-> template-> write_view('content',DIR_SIIGS.'/reglavaduna/insert',
$data);
181             $this-> template-> render();
182             $error = true;
183         }
184
185         if ($error == false)
186         {
187             $this-> session-> set_flashdata('msgResult', 'Registro insertado
correctamente');
188             $this-> session-> set_flashdata('clsResult', 'success');
189             redirect(DIR_SIIGS.'/reglavaduna/index','refresh');
190         }
191     }
192 }
193

```

```

194 /**
195  *Acción para preparar la actualización de una regla ya existente,
196  *recibe un ID para obtener los valores de esa regla y mostrarlos
197  *en la vista update , realiza la validación del formulario del lado
198  *del cliente
199 *
200  * @param int $id
201  * @return void
202  */
203 public function update($id)
204 {
205     if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
206         show_error('', 403, 'Acceso denegado');
207     $this-> load-> helper('form');
208     $this-> load-> helper('url');
209     $this-> load-> library('form_validation');
210
211     $error = false;
212
213     $data['title'] = 'Modificar regla para vacuna';
214     $this-> form_validation-> set_rules('aplicacion_inicio', 'Inicio aplicación',
215                                         'trim|xss_clean|required|is_natural');
216     $this-> form_validation-> set_rules('aplicacion_fin', 'Fin aplicación',
217                                         'trim|xss_clean|required|is_natural_no_zero');
218     $this-> form_validation-> set_rules('id_vacuna', 'Vacuna',
219                                         'trim|xss_clean|required|is_natural_no_zero');
220     $this-> form_validation-> set_rules('tipo_aplicacion', 'Tipo de
221                                         aplicación', 'trim|xss_clean|required');
222     $this-> form_validation-> set_rules('id_via_vacuna', 'Vía Vacuna',
223                                         'trim|xss_clean|required|is_natural_no_zero');
224     $this-> form_validation-> set_rules('dosis', 'Dosis',
225                                         'trim|xss_clean|decimal');
226     $this-> form_validation-> set_rules('region', 'Región',
227                                         'trim|xss_clean|max_length[90]');
228     $this-> form_validation-> set_rules('observacion_region', 'Observación de la
229                                         región', 'trim|xss_clean|max_length[450]');
230
231     $vacunas = $this-> db-> query("select id,descripcion from cns_vacuna
232 where activo=1" )-> result();
233     $data['vacunas']['''] = 'Elige una vacuna';
234     foreach ($vacunas as $item) {
235         $data['vacunas'][$item-> id] = $item-> descripcion;
236
237         $vias_vacuna = $this-> db-> query("select id,descripcion from
238 cns_via_vacuna where activo=1" )-> result();
239         $data['vias_vacuna']['''] = 'Elige una vía de vacuna';
240         foreach ($vias_vacuna as $item) {
241             $data['vias_vacuna'][$item-> id] = $item-> descripcion;
242         }
243
244         $orden = $this-> db-> query("SELECT a.id FROM (SELECT 1 AS id UNION
245 SELECT 2 AS id UNION SELECT 3 AS id UNION SELECT 4 AS id UNION SELECT 5 AS id UNION SELECT 6 AS id
246 UNION SELECT 7 AS id
247 UNION SELECT 8 AS id UNION SELECT 9 AS id UNION SELECT 10 AS id UNION SELECT 11 AS id UNION
248 SELECT 12 AS id UNION SELECT 13 AS id UNION SELECT 14 AS id UNION SELECT 15 AS id
249 UNION SELECT 16 AS id UNION SELECT 17 AS id UNION SELECT 18 AS id UNION SELECT 19 AS id UNION
250 SELECT 20 AS id
251 UNION SELECT 21 AS id UNION SELECT 22 AS id UNION SELECT 23 AS id UNION SELECT 24 AS id
252 UNION SELECT 25 AS id UNION SELECT 26 AS id
253 UNION SELECT 27 AS id UNION SELECT 28 AS id UNION SELECT 29 AS id UNION SELECT 30 AS id) AS
254 a WHERE a.id NOT IN (SELECT orden_esq_com FROM cns_regla_vacuna WHERE orden_esq_com IS NOT NULL and
255 id <> "
256 .$id." )-> result();
257         $data['orden']['''] = 'Orden de la vacuna';
258         foreach ($orden as $item) {
259             $data['orden'][$item-> id] = $item-> id;
260         }
261
262         $alergias = $this-> db-> query("select id,descripcion from cns_alergia
263 where activo=1" )-> result();
264         $data['alergias'] = $alergias;
265
266     }
267     if ($this-> form_validation-> run() === FALSE)
268     {
269         try
270         {
271             $data['regla_item'] = $this-> ReglaVacuna_model-> getById($id);
272         }
273         catch (Exception $e)
274     }

```

```

256
257     {
258         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
259         $data['clsResult'] = 'error';
260     }
261     $this-> template-> write_view('content',DIR_SIIGS.'/reglaviruscuna/update', $data);
262     $this-> template-> render();
263 }
264 else
265 {
266     try
267     {
268         $this-> ReglaVacuna_model-> setId($this-> input-
269 > post('id'));
270         $this-> ReglaVacuna_model-> setIdVacuna($this-> input-
271 > post('id_vacuna'));
272         $this-> ReglaVacuna_model-> setIdVacunaPrevia($this-> input-
273 > post('id_vacuna_previa'));
274         // if ($this->input->post('tipo_aplicacion') == 'nacimiento')
275         // $this-> ReglaVacuna_model-> setDiaInicioNacido($this-
276         // $this-> input-> post('aplicacion_inicio'));
277         // $this-> ReglaVacuna_model-> setDiaFinNacido($this-> input-
278         // $this->ReglaVacuna_model->setDiaInicioPrevia($this-
279         // $this->ReglaVacuna_model->setDiaFinPrevia($this-
280         // $this->ReglaVacuna_model->update());
281
282         $this-> ReglaVacuna_model-> setIdViaVacuna($this-> input-
283 > post('id_via_vacuna'));
284         $this-> ReglaVacuna_model-> setDosis($this-> input-
285 > post('region'));
286         $this-> ReglaVacuna_model-> setRegion($this-> input-
287 > input-> post('observacion_region'));
288         $this-> ReglaVacuna_model-> setAlergias($this-> input-
289 > post('alergias'));
290         $this-> ReglaVacuna_model-> setEsqComp((($this-> input-
291 > post('esq_com')) == false) ? false : true);
292         $this-> ReglaVacuna_model-> setForzarAplicacion((($this-
293 > input-> post('forzar_aplicacion')) == false) ? false : true));
294
295         if (!($this-> input-> post('esq_com') == false))
296         {
297             $this-> ReglaVacuna_model-> setOrdenEsqComp($this-> input-
298 > post('orden_esq_com'));
299         }
300
301         $this-> ReglaVacuna_model-> update();
302     }
303     catch (Exception $e)
304     {
305         $data['regla_item'] = $this-> ReglaVacuna_model-> getById($id);
306     }
307     catch (Exception $e)
308     {
309         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
310         $data['clsResult'] = 'error';
311     }
312     $this-> template-> write_view('content',DIR_SIIGS.'/reglaviruscuna/update',
313     $this-> template-> render();
314
315     $error = true;
316 }
317 if ($error == false)
318 {
319     $this-> session-> set_flashdata('msgResult', 'Registro actualizado

```

```

correctamente');
320         $this-> session-> set_flashdata('clsResult', 'success');
321         redirect(DIR_SIIGS.'/reglavaduna','refresh');
322     }
323 }
324 }
325 /**
326 *
327 * Acción para eliminar una regla, recibe el id de la regla a eliminar
328 *
329 * @param int $id
330 * @return void
331 */
332 public function delete($id)
333 {
334     try
335     {
336         if (empty($this-> ReglaVacuna_model))
337             return false;
338         if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__,
339         current_url()))
340             show_error('', 403, 'Acceso denegado');
341
342         $this-> load-> helper('url');
343         $this-> ReglaVacuna_model-> setId($id);
344         $this-> ReglaVacuna_model-> delete();
345         $this-> session-> set_flashdata('msgResult', 'Registro eliminado exitosamente');
346         $this-> session-> set_flashdata('clsResult', 'success');
347     }
348     catch(Exception $e)
349     {
350         $this-> session-> set_flashdata('msgResult', Errorlog_model::save($e-
> getMessage(), __METHOD__));
351         $this-> session-> set_flashdata('clsResult', 'error');
352     }
353     redirect(DIR_SIIGS.'/reglavaduna','refresh');
354 }
355 }

```

# Archivo fuente para usuario.php

La documentación para este archivo está disponible en [usuario.php](#)

```
1  <?php
2  /**
3   * Controlador Usuario
4   *
5   * @package    SIIGS
6   * @subpackage Controlador
7   * @author     Rogelio
8   * @created    2013-09-25
9   */
10  class Usuario extends CI_Controller {
11
12      public function __construct()
13  {
14          parent::__construct();
15          try{
16              $this-> load-> helper('url');
17              $this-> load-> library('Correo');
18              $this-> load-> library('session');
19          }
20          catch(Exception $e)
21  {
22              $this-> template-> write("content" , $e-> getMessage());
23              $this-> template-> render();
24          }
25      }
26
27      /**
28       * Ofrece el inicio de sesión
29       *
30       * @access    public
31       * @return    void
32       */
33      public function login()
34  {
35          try{
36              if (empty($this-> Usuario_model))
37                  return false;
38              $data['title'] = 'Inicio de sesión';
39              $this-> load-> helper('form');
40              $this-> load-> helper('url');
41              $this-> load-> library('form_validation');
42              $this-> form_validation-> set_rules('nombre_usuario', 'Nombre de Usuario',
43 'trim|required');
44              $this-> form_validation-> set_rules('clave', 'Clave', 'trim|required');
45              $data['msgResult'] = $this-> session-> flashdata('msgResult');
46              $data['clsResult'] = $this-> session-> flashdata('clsResult');
47
48              if ($this-> form_validation-> run() === FALSE)
49  {
50                  $this-> template-> write_view('content',DIR_SIIGS.'/usuario/login', $data);
51                  $this-> template-> render();
52              }
53              else
54  {
55                  $rowUser = $this-> Usuario_model-> authenticate($this-> input-
> post('nombre_usuario'), md5($this-> input-> post('clave')));
56                  if ($rowUser)
57  {
58                      if (!$rowUser-> activo)
59  {
60                          $data['msgResult'] = 'La cuenta de usuario proporcionada se encuentra
inactiva.';
61                          $data['clsResult'] = 'warning';
62                      }
63                  else
64  {
```

```

65                                // almacena en session las variables necesarias
66                                $this-> session-> set_userdata(USERNAME, strtoupper($rowUser-
> nombre_usuario));
67                                $this-> session-> set_userdata(PASSWORD, $this-> input-
> post('clave'));
68                                $this-> session-> set_userdata(USER_LOGGED, $rowUser-> id);
69                                $this-> session-> set_userdata(GROUP_ID, strtoupper($rowUser-
> id_grupo));
70                                // obtiene los permisos del grupo al que pertenece el usuario logueado
71                                $this-> load-> model(DIR_SIIGS.'/Entorno_model');
72                                $this-> session-> set_userdata(PERMISSIONS, $this-> Entorno_model-
> getPermissionsByGroup($rowUser-> id_grupo));
73                                //var_dump($this->session->userdata(PERMISSIONS)); return;
74                                Bitacora_model::insert(DIR_SIIGS.'::'.__CLASS__.':index', 'Sesion
iniciada: '.strtoupper($rowUser-> nombre_usuario));
75                                // redirige a la url de donde provino o a la predeterminada del sistema
76                                if (!$this-> session-> userdata(REDIRECT_TO))
77                                {
78                                    $this-> session-> set_flashdata('msgResult', 'Inicio de sesión
exitoso');
79                                    $this-> session-> set_flashdata('clsResult', 'success');
80                                    redirect('/', 'refresh'); // aca se debe poner la pagina HOME
81                                }
82                                else
83                                    redirect($this-> session-> userdata(REDIRECT_TO, 'refresh'));
84                                }
85                                }
86                                else
87                                {
88                                    $data['msgResult'] = 'Nombre de usuario o clave incorrecta.';
89                                    $data['clsResult'] = 'warning';
90                                }
91                            }
92                        }
93                    catch(Exception $e){
94                        $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
95                        $data['clsResult'] = 'error';
96                    }
97                    $this-> template-> write_view('content',DIR_SIIGS.'/usuario/login', $data);
98                    $this-> template-> render();
99                }
100            }
101        /**
102         * Termina la sesión
103         *
104         * @access      public
105         * @return      void
106         */
107        public function logout()
108    {
109        $this-> load-> helper('url');
110        if ($this-> session-> userdata(USERNAME))
111            Bitacora_model::insert(DIR_SIIGS.'::'.__CLASS__.':index', 'Sesion finalizada:
'.strtoupper($this-> session-> userdata(USERNAME)));
112        // destruye la sesión y redirige al login
113        $this-> session-> sess_destroy();
114        redirect('/', 'refresh');
115    }
116}
117 /**
118  * 1) Visualiza los usuarios existentes para su interacción CRUD
119  * 2) En caso de detectar un texto a buscar se filtran los usuarios existentes acorde a la
120 búsqueda
121 *
122 * @access      public
123 * @param       int      $pag     número de página a visualizar (paginación)
124 * @return      void
125 */
126 public function index($pag = 0)
127 {
128    try{
129        if (empty($this-> Usuario_model))
130            return false;
131        if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
132            show_error('', 403, 'Acceso denegado');
133        $data['title'] = 'Catálogo de Usuarios';
134        $this-> load-> helper('form');
135        $this-> load-> library('pagination');
136    }

```

```

137     $data['pag'] = $pag;
138     $data['msgResult'] = $this-> session-> flashdata('msgResult');
139     $data['clsResult'] = $this-> session-> flashdata('clsResult');
140
141     // Configuración para el Paginador
142     $configPag['base_url'] = '/'.DIR_SIIGS.'/usuario/index/';
143     $configPag['first_link'] = 'Primero';
144     $configPag['last_link'] = '&Uacute;ltimo' ;
145     $configPag['uri_segment'] = '4';
146     $configPag['total_rows'] = $this-> Usuario_model-> getNumRows($this-> input-
> post('busqueda'));
147     $configPag['per_page'] = 20;
148     $this-> pagination-> initialize($configPag);
149     if ($this-> input-> post('busqueda'))
150         $data['users'] = $this-> Usuario_model-> getOnlyActives($this-> input-
> post('busqueda'), FALSE, $configPag['per_page'], $pag);
151     else
152         $data['users'] = $this-> Usuario_model-> getOnlyActives('', FALSE,
153         $configPag['per_page'], $pag);
154     }
155     catch(Exception $e){
156         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
157         $data['clsResult'] = 'error';
158     }
159     $this-> template-> write_view('content',DIR_SIIGS.'/usuario/index', $data);
160     $this-> template-> render();
161 }
162 /**
163 * Visualiza los datos del usuario recibido
164 *
165 * @access public
166 * @param int $id id del usuario a visualizar
167 * @return void
168 */
169 public function view($id)
170 {
171     try {
172         if (empty($this-> Usuario_model))
173             return false;
174         if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
175             show_error('', 403, 'Acceso denegado');
176         $data['title'] = 'Ver detalles de usuario';
177         $data['user_item'] = $this-> Usuario_model-> getById($id, true);
178     }
179     catch(Exception $e){
180         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
181         $data['clsResult'] = 'error';
182     }
183     $this-> template-> write_view('content',DIR_SIIGS.'/usuario/view', $data);
184     $this-> template-> write('menu','','true');
185     $this-> template-> write('sala_prensa','','true');
186     $this-> template-> render();
187 }
188 /**
189 * 1) Prepara el formulario para la inserción de un usuario nuevo
190 * 2) Realiza las validaciones necesarias sobre cada campo del registro
191 *
192 * @access public
193 * @return void
194 */
195 public function insert()
196 {
197     if (empty($this-> Usuario_model))
198         return false;
199     if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
200         show_error('', 403, 'Acceso denegado');
201     $data['title'] = 'Crear un nuevo usuario';
202     $this-> load-> model(DIR_SIIGS.'/grupo_model');
203     $this-> load-> helper('form');
204     $this-> load-> library('form_validation');
205     $this-> form_validation-> set_rules('id_grupo', 'Grupo', 'is_natural_no_zero');
206     $this-> form_validation-> set_message('is_natural_no_zero', 'Debe seleccionar un
207 grupo válido');
208     $this-> form_validation-> set_rules('nombre_usuario', 'Nombre de Usuario',
209     'trim|xss_clean|required|min_length[5]|max_length[15]|callback_ifUserExists');
210     $this-> form_validation-> set_rules('clave', 'Clave',
211     'trim|xss_clean|required|min_length[5]|max_length[12]|matches[repitoclave]|md5');
```

```

211     $this-> form_validation-> set_rules('repiteclave', 'Repetir Clave',
212     'trim|xss_clean|required');
213     $this-> form_validation-> set_rules('nombre', 'Nombre',
214     'trim|xss_clean|required|max_length[40]');
215     $this-> form_validation-> set_rules('apellido_paterno', 'Apellido Paterno',
216     'trim|xss_clean|required|max_length[25]');
217     $this-> form_validation-> set_rules('apellido_materno', 'Apellido Materno',
218     'trim|xss_clean|max_length[25]');
219     $this-> form_validation-> set_rules('correo', 'Email',
220     'trim|required|valid_email|xss_clean|max_length[50]');
221     $arrGrupos = $this-> grupo_model-> getAll();
222     $data['grupos'][''] = '-- Seleccione una opción --';
223     foreach ($arrGrupos as $grupo)
224     {
225         $data['grupos'][$grupo-> id] = $grupo-> nombre;
226     }
227
228     if ($this-> form_validation-> run() === FALSE)
229     {
230         $this-> template-> write_view('content',DIR_SIIGS.'/usuario/insert', $data);
231         $this-> template-> render();
232     }
233     else
234     {
235         try {
236             $this-> Usuario_model-> setNombreUsuario(strtoupper($this-> input-
237 > post('nombre_usuario')));
238             $this-> Usuario_model-> setClave($this-> input-> post('clave'));
239             $this-> Usuario_model-> setNombre($this-> input-> post('nombre'));
240             $this-> Usuario_model-> setApellidoPaterno($this-> input-
241 > post('apellido_paterno'));
242             $this-> Usuario_model-> setApellidoMaterno($this-> input-
243 > post('apellido_materno'));
244             $this-> Usuario_model-> setCorreo($this-> input-> post('correo'));
245             $this-> Usuario_model-> setActivo(true);
246             $this-> Usuario_model-> setIdGrupo($this-> input-> post('id_grupo'));
247             $this-> Usuario_model-> insert();
248             $this-> session-> set_flashdata('msgResult', 'Registro agregado
249 exitosamente');
250             $this-> session-> set_flashdata('clsResult', 'success');
251             Bitacora_model::insert(DIR_SIIGS.'::'.__METHOD__, 'Usuario agregado:
252             strtoupper($this->
253             redirect(DIR_SIIGS.'/usuario','refresh'));
254         }
255         catch (Exception $e){
256             $data['clsResult'] = 'error';
257             $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
258             $this-> template-> write_view('content',DIR_SIIGS.'/usuario/insert', $data);
259             $this-> template-> render();
260         }
261     }
262
263     /**
264      * 1) Prepara el formulario para la modificación de un usuario existente
265      * 2) Realiza las validaciones necesarias sobre cada campo del registro
266      */
267     public function update($id)
268     {
269         if (empty($this-> Usuario_model))
270             return false;
271         if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
272             show_error('', 403, 'Acceso denegado');
273         $data['title'] = 'Modificar usuario';
274         $this-> load-> model(DIR_SIIGS.'/grupo_model');
275         $this-> load-> helper('form');
276         $this-> load-> library('form_validation');
277         $this-> form_validation-> set_rules('id_grupo', 'Grupo', 'is_natural_no_zero');
278         $this-> form_validation-> set_message('is_natural_no_zero', 'Debe seleccionar un
279 grupo válido');
280         $this-> form_validation-> set_rules('nombre', 'Nombre',
281     'trim|xss_clean|required|max_length[40]');
282         $this-> form_validation-> set_rules('apellido_paterno', 'Apellido Paterno',
283     'trim|xss_clean|required|max_length[25]');
284         $this-> form_validation-> set_rules('apellido_materno', 'Apellido Materno',
285     'trim|xss_clean|max_length[25]');

```

```

277         $this-> form_validation-> set_rules('correo', 'Email',
278 'trim|xss_clean|required|valid_email|max_length[50]');
279     $arrGrupos = $this-> grupo_model-> getAll();
280     $data['grupos'][''] = '-- Seleccione una opcion --';
281     foreach ($arrGrupos as $grupo)
282     {
283         $data['grupos'][$grupo-> id] = $grupo-> nombre;
284     }
285     $data['user_item'] = $this-> Usuario_model-> getById($id);
286
287     if ($this-> form_validation-> run() === FALSE)
288     {
289         $this-> template-> write_view('content',DIR_SIIGS.'/usuario/update', $data);
290         $this-> template-> render();
291     }
292     else
293     {
294         try {
295             $this-> Usuario_model-> setId($id);
296             $this-> Usuario_model-> setNombre($this-> input-> post('nombre'));
297             $this-> Usuario_model-> setApellidoPaterno($this-> input-> post('apellido_paterno'));
298             $this-> Usuario_model-> setApellidoMaterno($this-> input-> post('apellido_materno'));
299             $this-> Usuario_model-> setCorreo($this-> input-> post('correo'));
300             $this-> Usuario_model-> setActivo(0);
301             if ($this-> input-> post('activo') == 'on')
302                 $this-> Usuario_model-> setActivo(1);
303             $this-> Usuario_model-> setIdGrupo($this-> input-> post('id_grupo'));
304             $this-> Usuario_model-> update();
305             $this-> session-> set_flashdata('msgResult', 'Registro actualizado exitosamente');
306             $this-> session-> set_flashdata('clsResult', 'success');
307             Bitacora_model::insert(DIR_SIIGS.'::'.__METHOD__, 'Usuario actualizado: '.$id);
308             redirect(DIR_SIIGS.'/usuario', 'refresh');
309         }
310         catch (Exception $e){
311             $data['clsResult'] = 'error';
312             $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
313             $this-> template-> write_view('content',DIR_SIIGS.'/usuario/update', $data);
314             $this-> template-> render();
315         }
316     }
317
318 /**
319 * Solicita la eliminación del usuario recibido
320 *
321 * @access public
322 * @param int $id id del usuario a eliminar
323 * @return void
324 */
325 public function delete($id)
326 {
327     try {
328         if (empty($this-> Usuario_model))
329             return false;
330         if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
331             show_error('', 403, 'Acceso denegado');
332         $this-> Usuario_model-> setId($id);
333         $this-> Usuario_model-> delete();
334         $this-> session-> set_flashdata('msgResult', 'Registro eliminado exitosamente');
335         $this-> session-> set_flashdata('clsResult', 'success');
336         Bitacora_model::insert(DIR_SIIGS.'::'.__METHOD__, 'Usuario eliminado: '.$id);
337     }
338     catch (Exception $e){
339         $this-> session-> set_flashdata('clsResult', 'error');
340         $this-> session-> set_flashdata('msgResult', Errorlog_model::save($e-> getMessage(), __METHOD__));
341     }
342     redirect(DIR_SIIGS.'/usuario', 'refresh');
343 }
344
345 /**
346 * Callback para validar que un nombre de usuario no se duplique
347 *
348 * @access public
349 * @param string $username nombre de usuario a validar
350 * @return boolean false si el nombre de usuario ya existe, true
351 si el nombre de usuario está disponible

```

```

351     */
352     public function _ifUserExists($username)
353     {
354         if (empty($this-> Usuario_model))
355             return false;
356         $is_exist = null;
357         try {
358             $is_exist = $this-> Usuario_model-> getByUsername($username);
359         }
360         catch(Exception $e){
361         }
362         if ($is_exist)
363         {
364             $this-> form_validation-> set_message(
365                 '_ifUserExists', 'El nombre de usuario seleccionado ya existe, intente con
otro.');
366             );
367             return false;
368         }
369         else
370         {
371             if (!$this-> Usuario_model-> getMsgError())
372                 return true;
373             else{
374                 $this-> form_validation-> set_message(
375                     '_ifUserExists', $this-> Usuario_model-> getMsgError()
376                 );
377                 return false;
378             }
379         }
380     }
381
382     //////////////////// Recuperar contraseña y modificar datos de usuario
383
384     function form_init()
385     {
386         $this-> load-> helper('form');
387         $this-> load-> helper('url');
388         $data['title'] = 'Recordar Datos';
389         $data['info'] = '';
390         $this-> template-> write_view('content',DIR_SIIGS.'/usuario/recordar_datos',$data);
391
392         $this-> template-> render();
393     }
394     public function remember()
395     {
396         $this-> load-> helper('form');
397         $this-> load-> helper('url');
398         $this-> load-> library('form_validation');
399         $this-> form_validation-> set_rules('nombre_usuario', 'Nombre de Usuario',
'trim|required');
400         $this-> form_validation-> set_rules('correo', 'Correo', 'trim|required');
401         $data['title'] = 'Recordar Datos';
402         $data['info'] = '';
403         if ($this-> form_validation-> run() === FALSE)
404         {
405             $this-> template-
> write_view('content',DIR_SIIGS.'/usuario/recordar_datos',$data);
406             $this-> template-> render();
407         }
408         else
409         {
410             $this-> load-> model(DIR_SIIGS.'/usuario_model');
411             $datos=$this-> usuario_model-> check_data(strtoupper($this-> input-
> post('nombre_usuario')),$this-> input-> post('correo'));
412             if(sizeof($datos)==0)
413             {
414                 $data['infoclass']= 'error';
415                 $data['msgResult']= 'No se pudo comprobar sus datos';
416
417                 $this-> template-
> write_view('content',DIR_SIIGS.'/usuario/recordar_datos',$data);
418                 $this-> template-> render();
419             }
420             else
421             {
422
423                 $todos =
"1234567890abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ"
;

```

```

424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492

```

**\$valor= " " ;  
for(\$i=0;\$i< 64;\$i++)  
\$valor.= substr(\$todos,rand(0,62),1);  
  
\$\_SERVER['HTTP\_HOST']."' /siigs/usuario/token/\$valor" ;  
//\$url="http://www.siigs.com/siigs/usuario/token/\$valor";  
\$subject="Cómo restablecer su contraseña de SIIGS" ;  
\$adj[0]="resources/images/logo.png" ;  
\$body="<html>  
<head>  
<title></title>  
<style type='text/css'>  
.ph, .pf{color:#666666;font-family:Arial, Helvetica, sans-serif;font-size:13px;padding-top:15px;padding-left:20px;margin-bottom:0px;}  
.p{color:#666666;font-family:Arial, Helvetica, sans-serif;font-size:13px; margin-bottom:0px;}  
#content{width:650px;height:400px;}  
#imgheader{display:block;border:none;width:650px;height:50px;}  
#imgfooter{display:block;border:none;width:650px;height:50px;margin-top:10px;}  
#content\_campos{width:650px;height:auto;min-width:650px;}  
ul {margin-top:0px;margin-bottom:0px;padding-left: 2.5em;line-height: 1.2em; font-size:13px}  
ul li{color:#666666;font-family:Tahoma, Geneva, sans-serif !important;font-size:13px}  
a {color:#3333;}  
</style>  
</head>  
<body>  
<div id='content'>  
<img id='imgheader' src='logo.png' />  
<div id='content\_campos'>  
<p class='ph'>  
Hola, " . \$datos-> nombre. " . \$datos-> apellido\_paterno."  
". \$datos-> apellido\_materno." :<br><br>  
Para recuperar su cuenta del sistema SIIGS, deberá crear una nueva contraseña.  
</p>  
<table style='width:600px; margin-top:5px; margin-left:20px;'>  
<tr><td class='ph'>  
<b>Es fácil:</b><br><br>  
1. Haga clic en el siguiente vínculo para abrir una ventana del navegador segura.<br>  
2. Confirme que usted es el propietario de la cuenta y, a continuación, siga las instrucciones.  
</td></tr>  
</table>  
<br />  
<table style='width:600px; margin-top:5px; margin-left:20px;'>  
<tr><td><p class='p'><b>Restablecer su contraseña ahora:</b> </p></td></tr>  
<tr><td>  
<ul>  
<li><a href='".\$url'>Restablecer</a></li>  
</ul>  
</td>  
</tr>  
</table>  
<table style='width:600px; margin-top:5px; margin-left:20px;'>  
<tr>  
<td align='left'><p class='pf'>No responda este correo electrónico, ya que no estamos controlando esta bandeja de entrada. Para comunicarse con nosotros, inicie sesión en su cuenta y haga clic en «Contáctenos» en la parte inferior de cualquier página.  
Copyright © 2013. Todos los derechos reservados.</p></td>  
</tr>  
</table>  
<br />  
<img id='imgfooter' src='footer\_mail.png' />  
</div>  
</body>  
</html>  
\$from="SIIGS@siigs.com" ;  
\$rto=" " ;**

```

493             $correo=$datos-> correo;
494             $CC="";
495             $CCO="";
496
497             $fecha=date("d/m/Y H:i:s");
498             $fp = fopen(APPPATH."logs/recuperalog.siigs" , "a");
499             fputs($fp, "[ $fecha ][$valor][$correo]\r\n" );
500             fclose($fp);
501             $envio=$this-> send_mail($subject,$body,$from,$rto,$correo,$cc,$cco,$adj);
502             if($envio)
503             {
504                 $data['infoclass']= 'success';
505                 $data['msgResult']= 'Se ha enviado un correo con la informacion necesaria
para restablecer sus datos';
506             }
507             else
508             {
509                 $data['infoclass']= 'error';
510                 $data['msgResult']= 'No se pudo enviar el correo con la informacion
necesaria para restablecer sus datos';
511             }
512             $this-> template-
> write_view('content',DIR_SIIGS.'/usuario/enviar_correo',$data);
513             $this-> template-> render();
514         }
515     }
516 }
517 public function token($str)
518 {
519     $variable="";
520
521     $file = fopen(APPPATH."logs/recuperalog.siigs" , "r" ) or
522 exit("No se puedo abrir!");
523     while(!feof($file))
524     {
525         $variable.=fgets($file);
526     }
527     fclose($file);
528     if(strpos($variable,$str))
529     {
530         $cad=substr($variable,strpos($variable,$str)+strlen($str),100);
531         if(strpos($cad,'['))
532             $cad=substr($cad,1,100);
533
534         $correo=substr($cad,strpos($cad,"[") +1,strpos($cad,']')-1);
535         $correo=str_replace("[","",$correo);
536         $correo=str_replace("]","", $correo);
537         $this-> load-> helper('form');
538         $this-> load-> helper('url');
539         $this-> load-> library('form_validation');
540         $this-> load-> model(DIR_SIIGS.'/usuario_model');
541         $datos=$this-> usuario_model-> check_token($correo);
542         $data["error"]="";
543         if(sizeof($datos)!=0)
544         {
545             $data["title"]      ="Reset Contraseña" ;
546             $data["info"]       ="1.- Compruebe sus datos " ;
547             $data["info2"]      ="2.- Escriba su nueva contraseña" ;
548             $data["c"]          =$str;
549             $this-> template-
> write_view('content',DIR_SIIGS.'/usuario/reset_pass',$data);
550             $this-> template-> render();
551         }
552     }
553     else
554         show_error('', 403, 'El enlace ha caducado');
555 }
556 public function send_mail($subject,$body,$from,$rto,$correo,$cc,$cco,$adj)
557 {
558     $mail = new PHPMailer();
559     $mail-> IsSMTP();                                     // establecemos que utilizaremos
SMTP
560     $mail-> SMTPAuth = true;                            // habilitamos la autenticación
SMTP
561     $mail-> SMTPSecure = "ssl";                         // establecemos el
prefijo del protocolo
562     $mail-> Host   = "smtp.gmail.com";                  // establecemos
nuestro servidor
563     $mail-> Port   = 465;                                // establecemos el puerto

```

```

564     $mail-> Username = "ehlhihehchehrh@gmail.com" ; // la cuenta de correo
565 para el envio
565     $mail-> Password = "mimoj98i" ; // password de la
566 cuenta
566     $mail-> CharSet = 'utf-8'; //Especifica el tipo de
567 codificacion de caracteres.
567     $mail-> Timeout = 20; //Tiempo maximo de espera para
envio de correo.
568     $mail-> IsHTML(true); //El correo esta en formato html.
569
570     $mail-> Subject = $subject; // Asunto del mensaje
571     $mail-> Body = $body; // Mensaje
572     $mail-> SetFrom ($from); // Nombre a mostrar en el envio
573     $mail-> AddReplyTo($rto); // Si hay respuesta
574     $mail-> AddAddress($correo); // Destinatario
575     if($CC!="") // Con copia a
576         $mail-> AddCC($CC); // Con copia oculta a
577     if($CCO!="") // Añade archivos adjuntos
578         $mail-> AddBCC($CCO);
579     if(sizeof($adj)> 0)
580     {
581         for($i=0;$i< sizeof($adj);$i++)
582             $mail-> AddAttachment($adj[$i]);
583     }
584
585     $exito = $mail-> Send();
586     return $exito;
587 }
588 public function reset()
589 {
590     $data[ "error" ]="";
591     $str=$this-> input-> post('c');
592     $data[ "c" ]=$str;
593     $this-> load-> helper('form');
594     $this-> load-> helper('url');
595     $this-> load-> library('form_validation');
596     $this-> form_validation-> set_rules('nombre_usuario', 'Nombre de Usuario',
'trim|required');
597     $this-> form_validation-> set_rules('correo', 'Correo', 'trim|required');
598     $this-> form_validation-> set_rules('pass','Contraseña',
'trim|required[min_length[5]|max_length[12]|matches[repiteclave]|md5']);
599     $data['title'] = 'Reset Contraseña';
600     $data["info"] = "1.- Compruebe sus datos ";
601     $data["info2"] = "2.- Escriba su nueva contraseña";
602
603     if ($this-> form_validation-> run() === FALSE)
604     {
605         $this-> template-> write_view('content',DIR_SIIGS.'/usuario/reset_pass',$data);
606         $this-> template-> render();
607     }
608     else
609     {
610         $variable="";
611
612         $file = fopen(APPPATH."logs/recuperalog.siigs" , "r" ) or
612 exit("No se pudo abrir!");
613         while(!feof($file))
614         {
615             $variable.=fgets($file);
616         }
617         fclose($file);
618         if(strpos($variable,$str))
619         {
620             $this-> load-> model(DIR_SIIGS.'/usuario_model');
621             $datos=$this-> usuario_model-> check_data(strtoupper($this-> input-
622 > post('nombre_usuario')),$this-> input-> post('correo'));
623
624             if(sizeof($datos)==0)
625             {
626                 $data[ "error" ]="";
626                 $data['infoclass']='error';
627                 $data['msgResult']= 'No se pudo comprobar sus datos';
628                 $this-> template-
629 > write_view('content',DIR_SIIGS.'/usuario/reset_pass',$data);
629                 $this-> template-> render();
630             }
631             else
632             {
633                 $this-> usuario_model-> update_pass($this-> input-

```

```

>     post('pass'),$datos-> id);
634
635             $pt=stripos($variable,$str);
636             $cad=substr($variable,0,$pt);
637             $cad.=substr($variable,($pt+64),strlen($variable)-($pt+64));
638             $fp = fopen(APPPATH."logs/recuperalog.siigs" , "w" );
639             fwrite($fp, $cad);
640             fclose($fp);
641         }
642         $data["info"] = "";
643         $data['infoclass']= 'success';
644         $data['msgResult']= 'Se ha restablecido la contraseña';
645         $this-> template-
646         > write_view('content',DIR_SIIGS.'/usuario/enviar_correo',$data);
647         $this-> template-> render();
648     }
649     else
650         show_error('', 403, 'El enlace ha caducado');
651     }
652
653 ////////////// modificar datos de usuario
654
655 public function load_update()
656 {
657     $this-> load-> helper('url');
658     $grupo=$this-> session-> userdata(GROUP_ID);
659     $name=$this-> session-> userdata(USERNAME);
660     $id=$this-> session-> userdata(USER_LOGGED);
661     if ($id == "")
662     {
663         $this-> template-> write_view('content',DIR_SIIGS.'/usuario/login', $data);
664         $this-> template-> render();
665         return;
666     }
667     else
668     {
669         $this-> load-> model(DIR_SIIGS.'/usuario_model');
670         $datos=$this-> usuario_model-> getuser($id);
671         $grup=$this-> usuario_model-> getgrupo($grupo);
672         $this-> load-> helper('form');
673         $this-> load-> helper('url');
674         $data['title'] = 'Cambiar Datos';
675         $data['info'] = '';
676         $data['error'] = '';
677         $data['correo'] = $datos-> correo;
678         $data['grupo'] = $grup-> nombre;
679         $data['nombre'] = $datos-> nombre." ". $datos-
680         > apellido_paterno." ". $datos-> apellido_materno;
681         $this-> template-
682         > write_view('content',DIR_SIIGS.'/usuario/cuenta_usuario',$data);
683         $this-> template-> render();
684     }
685     public function update_info()
686     {
687         $this-> load-> model(DIR_SIIGS.'/usuario_model');
688         $grupo=$this-> session-> userdata(GROUP_ID);
689         $name=$this-> session-> userdata(USERNAME);
690         $id=$this-> session-> userdata(USER_LOGGED);
691         $datos=$this-> usuario_model-> getuser($id);
692         $grup=$this-> usuario_model-> getgrupo($grupo);
693
694         $this-> load-> helper('form');
695         $this-> load-> helper('url');
696         $this-> load-> library('form_validation');
697         $this-> form_validation-> set_rules('correo', 'Correo',
698         'trim|required|valid_email|max_length[50]');
699         $this-> form_validation-> set_rules('pass', 'Contraseña Actual', 'trim|required');
700         $this-> form_validation-> set_rules('newpass', 'Nueva Contraseña',
701         'trim|required|min_length[5]|max_length[12]|matches[repiteclave]|md5');
702         $data['title'] = 'Cambiar Datos';
703         $data['info'] = '';
704         $data['error'] = '';
705         $data['correo'] = $grup-> nombre;
706         $data['nombre'] = $datos-> nombre." ". $datos-> apellido_paterno."
707         . $datos-> apellido_materno;
708         if ($this-> form_validation-> run() === FALSE)
709     {

```

```

707     $this-> template-
708     write_view('content',DIR_SIIGS.'/usuario/cuenta_usuario',$data);
709   }
710   else
711   {
712     $pass = md5($this-> input-> post('pass'));
713     $data['title'] = 'Cambiar Datos';
714     $data['info'] = '';
715     $data['correo'] = '';
716     $data['grupo'] = $grup-> nombre;
717     $data['nombre'] = $datos-> nombre." ".$datos-
718     apellido_paterno." ".$datos-> apellido_materno;
719     if($datos-> clave==$pass)
720     {
721       $update=$this-> usuario_model-> update_user($id,$this-> input-
722       > post('newpass'),$this-> input-> post('correo'));
723       if($update)
724       {
725         $data['error'] = '';
726         $data['infoclass']= 'success';
727         $data['msgResult']= 'Actualización correcta';
728       }
729       else
730       {
731         $data['error']= '';
732         $data['infoclass']= 'error';
733         $data['msgResult']= 'Ocurrio un error al actualizar';
734       }
735     }
736     else
737     {
738       $data['error'] = '';
739       $data['infoclass']= 'warning';
740       $data['msgResult']= 'No coincide la contraseña actual';
741     }
742     $this-> template-
743     write_view('content',DIR_SIIGS.'/usuario/cuenta_usuario',$data);
744   }
745 /**
746 *Acción para servir un array de objetos con los usuarios activos por grupo
747 *AJAX y devuelve un objeto JSON
748 */
749 * @param int $grupo
750 * @return Object JSON
751 */
752 public function getActivesByGroup($grupo)
753 {
754   try {
755     if ($this-> input-> is_ajax_request())
756     {
757       $data['usuarios'] = $this-> Usuario_model-> getActivesByGroup($grupo);
758       echo json_encode($data['usuarios']);
759       exit;
760     }
761     else echo 'Acceso denegado';
762   }
763   catch(Exception $e){
764     echo $e-> getMessage();
765   }
766 }
767 /**
768 *Acción para hacer autologin en otro sistema
769 */
770 */
771 * @return redirect
772 */
773 public $mas= "";
774 public function automatic_access()
775 {
776   if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
777     show_error('', 403, 'Acceso denegado');
778
779   $user=$this-> session-> userdata(USERNAME);
780   $pass=$this-> session-> userdata(PASSWORD);
781   if(strtoupper($user)=="ADMIN")
782     $pass="adminSM2015";

```

```

783     if($this-> session-> userdata('session_etab')=="iniciado" ) )
784     {
785         $this-> session-> unset_userdata('session_etab');
786         echo '<script>
787             document.location.href="http://etab.sm2015.com.mx/admin/";
788             </script>';
789         $this-> cerrar_etab();
790     }
791     $token= "";
792     //while(strlen(str_replace(" ","",$token))<40)
793     $token=$this-
794     > get_token('http://etab.sm2015.com.mx/admin/login',true,'name=_csrf_token'
value="" ,26,40);
794     //valida=$this-
795     >get_token('http://etab.sm2015.com.mx/admin/login_check',true,'<ul
class="nav">',906,20,"_username=$user&_password=$pass&_csrf_token=$token&
;_submit=Entrar");
795
796     $this-> session-> set_userdata( 'session_etab' , "iniciado" );
797
798     $data[ "user" ]=$user;
799     $data[ "pass" ]=$pass;
800     $data[ "token" ]=$token;
801     $this-> load-> view(DIR_TES.'/enrolamiento/etab',$data);
802 }
803 /**
804 *Acción para cerrar el login en otro sistema
805 *
806 *
807 * @return redirect
808 */
809 function cerrar_etab()
810 {
811     $token=$this-
811     > get_token('http://etab.sm2015.com.mx/admin/logout',false,'name=_csrf_token'
value="" ,26,40);
812     $this-> session-> unset_userdata('session_etab');
813 }
814 /**
815 *Acción para obtener el token oculto en el login
816 *
817 *
818 * @return token
819 */
820 function
821 get_token($url,$var,$valor=""
,$num="" ,,$count="" ,,$par="")
{
822     global $mas;
823
824     $ch = curl_init();
825     curl_setopt($ch, CURLOPT_URL, $url);
826
827     curl_setopt($ch, CURLOPT_POST, 1);
828     curl_setopt($ch, CURLOPT_HEADER, 0);
829     curl_setopt($ch, CURLOPT_FOLLOWLOCATION, 1);
830     curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);
831     curl_setopt($ch, CURLOPT_COOKIEJAR,
str_replace('/','/','dirname(__FILE__).'/fb_cookies'.'$mas.'.txt'));
832     curl_setopt($ch, CURLOPT_COOKIEFILE,
str_replace('/','/','dirname(__FILE__).'/fb_cookies'.'$mas.'.txt'));
833     curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
834     curl_setopt($ch, CURLOPT_FRESH_CONNECT, true);
835     curl_setopt($ch, CURLOPT_COOKIESESSION,
str_replace('/','/','dirname(__FILE__).'/fb_cookies'.'$mas.'.txt'));
836     if($var)
837     {
838         $galleta=$this-> get_galleta();
839         curl_setopt($ch,
CURLOPT_POSTFIELDS,$par.'PHPSESSID='.$galleta.'&_submit=Entrar' );
840     }
841     curl_setopt($ch, CURLOPT_USERAGENT, "Mozilla/5.0 (Windows; U; Windows NT 6.0; en-US; rv:1.9.0.6) Gecko/2009011913 Firefox/3.0.6 (.NET CLR 3.5.30729)" );
842     $html = curl_exec($ch);
843     $token=substr($html,stripos($html,$valor)+$num,$count);
844
845     $err = 0;
846     $err = curl_errno($ch);
847     curl_close($ch);
848     if ($err != 0)
849     {

```

```

850         echo 'error=' . $err . "\n" ;
851         return $err;
852     }
853     else
854     {
855         return $token;
856     }
857 }
858 /**
859 *Obtiene las cookies que se generan en la session
860 *
861 *
862 * @return cookies
863 */
864 function get_galleta()
865 {
866     global $mas; $variable="";
867     $file = fopen(str_replace('/', '/', dirname(__FILE__)).'/fb_cookies' . $mas . '.txt',
868 "r") or exit("No se pudo abrir!");
869     while(!feof($file))
870     {
871         $variable.= fgets($file);
872     }
873     fclose($file);
874     return substr($variable, strpos($variable, "PHPSESSID"
875 )+10, 26);
}

```

# Archivo fuente para accion\_model.php

La documentación para este archivo está disponible en [accion\\_model.php](#)

```
1  <?php
2
3  /**
4   * Modelo Accion
5   *
6   * @package    SIIGS
7   * @subpackage Modelo
8   * @author     Geovanni
9   * @created    2013-09-26
10  */
11 class Accion_model extends CI_Model {
12
13     /**
14      * @access private
15      * @var int
16      */
17     private $id;
18
19     /**
20      * @access private
21      * @var string
22      */
23     private $nombre;
24
25     /**
26      * @access private
27      * @var string
28      */
29     private $descripcion;
30
31     /**
32      * @access private
33      * @var string
34      */
35     private $metodo;
36
37     /**
38      * @access private
39      * @var int
40      */
41     private $offset;
42
43     /**
44      * @access private
45      * @var int
46      */
47     private $rows;
48
49     /**
50      * @access private
51      * @var string
52      */
53     private $msg_error_log;
54
55     /**
56      * @access private
57      * @var string
58      */
59     private $msg_error_usr;
60
61     /*****
62     *Getters and setters block*/
63     *****/
64     public function getId() {
65         return $this-> id;
66     }
67 }
```

```

68     public function setId($value) {
69         $this-> id = $value;
70     }
71
72     public function getNombre() {
73         return $this-> nombre;
74     }
75     public function setNombre($value) {
76         $this-> nombre = $value;
77     }
78
79     public function getDescripcion() {
80         return $this-> descripcion;
81     }
82
83     public function setDescripcion($value) {
84         $this-> descripcion = $value;
85     }
86
87     public function getMetodo() {
88         return $this-> metodo;
89     }
90
91     public function setMetodo($value) {
92         $this-> metodo = $value;
93     }
94
95     public function setOffset($value) {
96         $this-> offset = $value;
97     }
98     public function setRows($value) {
99         $this-> rows = $value;
100    }
101   /**
102    *Getters and setters block END*/
103   /**
104   */
105  /**
106   * Devuelve los mensajes de error en caso de ocurrir alguna excepción
107   * 'usr' devuelve el mensaje para la vista de usuario
108   * 'log' devuelve el mensaje para el log de errores
109   *
110  * @access public
111  * @return string|boolean
112  * @param string $value, default 'usr' (Tipo mensaje)
113  */
114  public function getMsgError($value = 'usr')
115  {
116      if (!empty($this-> msg_error_usr))
117      {
118          if ($value == 'usr')
119              return $this-> msg_error_usr;
120          else if ($value == 'log')
121              return $this-> msg_error_log;
122      }
123      else
124      {
125          return false;
126      }
127  }
128
129  public function __construct()
130  {
131      $this-> load-> database();
132      if (!$this-> db-> conn_id)
133      {
134          throw new Exception("No se pudo conectar a la base de datos");
135      }
136  }
137
138  /**
139   *Devuelve todos los registros de la tabla acciones
140   *
141   * @access public
142   * @return ArrayObject
143   * @throws Exception En caso de algun error al consultar la base de datos
144   */
145  public function getAll()
146  {
147      $string = 'select * from sis_accion';

```

```

148      if ((!empty($this-> offset) || $this-> offset == 0) && !empty($this-
149 > rows))
150      {
151          $string .= ' limit ' . $this-> offset . ',' . $this-> rows;
152          $query = $this-> db-> query($string);
153
154          if (!$query)
155          {
156              $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
157 > db-> _error_number() . ': ' . $this-> db-> _error_message();
158              $this-> msg_error_usr = "Ocurrió un error al obtener los datos de
159 acciones";
160          }
161          else
162          {
163              return $query-> result();
164          }
165
166          /**
167          * Devuelve el numero de registros
168          *
169          * @access public
170          * @return int
171          * @throws Exception En caso de algun error al consultar la base de datos
172          */
173          public function getNumRows()
174          {
175              $query = $this-> db-> query('select count(*) as num from sis_accion');
176              if (!$query)
177              {
178                  $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
179 > db-> _error_number() . ': ' . $this-> db-> _error_message();
180                  $this-> msg_error_usr = "Ocurrió un error al obtener los datos de las
181 acciones";
182              }
183              else
184              {
185                  return $query-> row()-> num;
186              }
187
188          /**
189          * Devuelve la informacion de una accion por su ID
190          *
191          * @access public
192          * @return Object
193          * @param int $id ID (Llave primaria)
194          * @throws Exception En caso de algun error al consultar la base de datos
195          */
196          public function getById($id)
197          {
198              $query = $this-> db-> get_where('sis_accion', array('id' => $id));
199
200              if (!$query)
201              {
202                  $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
203 > db-> _error_number() . ': ' . $this-> db-> _error_message();
204                  $this-> msg_error_usr = "Ocurrió un error al obtener la informacion de la
205 accion";
206              }
207              else
208              {
209                  return $query-> row();
210              }
211
212          /**
213          * Inserta en la tabla accion la informacion contenida en el objeto
214          *
215          * @access public
216          * @return int (Id de la insercion si no hubo errores al actualizar)
217          * @throws Exception En caso de algun error al consultar la base de datos
218          */
219          public function insert()
220          {
221              $data = array(
222                  'nombre' => $this-> nombre,
223                  'descripcion' => $this-> descripcion,
224                  'metodo' => $this-> metodo
225              );

```

```

221     $query = $this-> db-> insert('sis_accion', $data);
222
223     if (!$query)
224     {
225         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
226 > _error_number().': '.$this-> db-> _error_message();
227         $this-> msg_error_usr = "Ocurrió un error al insertar la acción" ;
228         throw new Exception(__CLASS__);
229     }
230     else
231     {
232         return $this-> db-> insert_id($query);
233     }
234
235 /**
236 *Actualiza el objeto actual en la base de datos
237 */
238 /**
239 * @access public
240 * @return boolean (Si no hubo errores al actualizar)
241 * @throws Exception En caso de algun error al consultar la base de datos
242 */
243 public function update()
244 {
245     $data = array(
246         'nombre' => $this-> nombre,
247         'descripcion' => $this-> descripcion,
248         'metodo' => $this-> metodo
249     );
250
251     $this-> db-> where('id' , $this-> getId());
252     $query = $this-> db-> update('sis_accion', $data);
253
254     if (!$query)
255     {
256         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
257 > _error_number().': '.$this-> db-> _error_message();
258         $this-> msg_error_usr = "Ocurrió un error al actualizar los datos de la
259 acción" ;
260         throw new Exception(__CLASS__);
261     }
262     else
263     {
264         return true;
265     }
266
267 /**
268 * Elimina el registro actual de la base de datos
269 */
270 public function delete()
271 {
272     $query = $this-> db-> delete('sis_accion', array('id' => $this-> getId()));
273
274     if (!$query)
275     {
276         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
277 > _error_number().': '.$this-> db-> _error_message();
278         $this-> msg_error_usr = "Ocurrió un error al eliminar la acción" ;
279     }
280     else
281     {
282         return true;
283     }
284 }

```

# Archivo fuente para ageb\_model.php

*La documentación para este archivo está disponible en [ageb\\_model.php](#)*

```
1      <?php
2
3  /**
4   * Modelo Ageb
5   *
6   * @package    SIIGS
7   * @subpackage Modelo
8   * @author     Geovanni
9   * @created    2014-01-31
10  */
11 class Ageb_model extends CI_Model
12 {
13     /*****
14     * Estas variables no pertenecen a la tabla *
15     * *****/
16
17     /**
18     * @access private
19     * @var boolean
20     */
21     private $error;
22
23     /**
24     * @access private
25     * @var string
26     */
27     private $msg_error_usr;
28
29     /**
30     * @access private
31     * @var string
32     */
33     private $msg_error_log;
34
35
36     public function __construct()
37     {
38         parent::__construct();
39         $this-> load-> database();
40         $this-> error = false;
41         $this-> msg_error_usr = '';
42         $this-> msg_error_log = '';
43
44         /*if( !$this->db->conn_id ) {
45             throw new Exception ('ERROR: No se puede conectar con la Base de Datos');
46         }*/
47     }
48
49     /**
50     * Devuelve el mensaje de error,
51     * en caso de existir un error despues de ejecutar un metodo,
52     * de lo contrario false
53     *
54     * @access public
55     * @param string $type usr si se quiere devolver el mensaje de error a mostrar en la vista,
56     *                      log obtiene el mensaje de error con mas detalles para depuración,
57     *                      valor por defecto usr
58     * @return boolean/string
59     */
60     public function getMsgError($type = 'usr')
61     {
62         if($this-> error) {
63             if($type == 'usr')
64                 return $this-> msg_error_usr;
65             else if($type == 'log')
66                 return $this-> msg_error_log;
67             else
68         }
```

```

68             return false;
69         }
70     }
71     return false;
72 }
73
74 /**
75 * Inserta los registros contenidos en la tabla cat_poblacion
76 * a la tablaasu_poblacion
77 *
78 * @access public
79 * @return boolean false Si no se ejecutó la inserción, true si se ejecutó la inserción
80 */
81 public function process()
82 {
83     $result = false;
84
85     // $sql = "REPLACE INTO
86     //     asu_ageb (id_asu_localidad, ageb,id_asu_um)
87     //     SELECT
88     //         c.id AS id_asu_localidad,
89     //         LPAD(a.ageb,4,'0000') AS ageb,
90     //         CASE WHEN
91     //             IFNULL(d.id , '') = ''
92     //             THEN
93     //                 a.clues
94     //             ELSE
95     //                 d.id
96     //             END
97     //             AS id_asu_um
98     //             FROM
99     //             cat_ageb a
100    //             JOIN
101    //             cat_localidad b
102    //             ON
103    //                 b.id_localidad = a.id_localidad
104    //                 AND b.id_municipio = a.id_municipio
105    //                 AND a.id_estado = b.id_estado
106    //             LEFT OUTER JOIN
107    //                 asu_arbol_segmentacion c
108    //                 ON
109    //                     b.id = c.id_tabla_original
110    //                     AND c.grado_segmentacion = 4
111    //                     AND c.id_raiz = 1
112    //             LEFT OUTER JOIN
113    //                 asu_arbol_segmentacion d
114    //                 ON
115    //                     d.grado_segmentacion = 5
116    //                     AND d.id_raiz = 1
117    //                     AND d.id_tabla_original = a.clues";
118
119
120     $sql = "REPLACE INTO
121         asu_ageb (id_asu_localidad, ageb,id_asu_um)
122         SELECT
123             d.id_padre AS id_asu_localidad,
124             LPAD(a.ageb,4,'0000') AS ageb,
125             d.id AS id_asu_um
126             FROM
127             cat_ageb a
128             JOIN
129             cat_localidad b
130             ON
131             b.id_localidad = a.id_localidad
132             AND b.id_municipio = a.id_municipio
133             AND a.id_estado = b.id_estado
134             JOIN
135             asu_arbol_segmentacion d
136             ON
137             d.grado_segmentacion = 5
138             AND d.id_raiz = 1
139             AND d.id_tabla_original = a.clues" ;
140
141     $result = $this-> db-> query($sql);
142
143     if( $this-> db-> _error_number() ) {
144         $this-> error = true;
145         $this-> msg_error_usr = 'Error al procesar los ageb';
146         $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
147 > _error_number() . ': ' . $this-> db-> _error_message();

```

```

147             throw new Exception(' (' . __METHOD__ . ') => ' . $this-> db-> _error_number() . ':');
148         }
149     }
150     return $result;
151 }
152 /**
153 *Devuelve la información de una UM de acuerdo a su localidad y ageb
154 *
155 *@access public
156 *@return Object
157 *@param int $idlocalidad Id del ASU de la localidad
158 * @param string Ageb
159 * @throws Exception En caso de algun error al consultar la base de datos
160 */
161 public function searchUM($idlocalidad,$ageb)
162 {
163     $query = $this-> db-> get_where('asu_ageb', array('id_asu_localidad' =>
164 $idlocalidad, 'ageb' => $ageb));
165     if (!$query)
166     {
167         $this-> msg_error_log = " (" . __METHOD__ . ") => " . $this-
168 > db-> _error_number() . ': ' . $this-> db-> _error_message();
169         $this-> msg_error_usr = "Ocurrió un error al obtener la información de la UM
por medio de la AGEB y Localidad";
170         throw new Exception(__CLASS__);
171     }
172     else
173     {
174         if ($query-> num_rows() > 0)
175             // if ($query->row()->id_asu_um == 0)
176             // return -1;
177             // else
178             return $query-> row()-> id_asu_um;
179         else
180             return -1;
181     }
182 }
183 /**
184 *Devuelve una lista de AGEBS de la localidad pasada como parámetro
185 *
186 *@access public
187 *@return Object
188 *@param int $idlocalidad Id del ASU de la localidad
189 * @throws Exception En caso de algun error al consultar la base de datos
190 */
191 public function searchageb($idlocalidad,$like)
192 {
193     $query = $this-> db-> query("select ageb from asu_ageb where id_asu_localidad
= '$idlocalidad.' and ageb like '%" . addslashes($like) . "%'");
194     if (!$query)
195     {
196         $this-> msg_error_log = " (" . __METHOD__ . ") => " . $this-
197 > db-> _error_number() . ': ' . $this-> db-> _error_message();
198         $this-> msg_error_usr = "Ocurrió un error al obtener la lista de AGEBs por
localidad";
199         throw new Exception(__CLASS__);
200     }
201     else
202     {
203         return $query-> result();
204     }
205 }
206 }
207 }
208 }

```

# Archivo fuente para arbolsegmentacion\_model.php

La documentación para este archivo está disponible en [arbolsegmentacion\\_model.php](#)

```
1  <?php
2
3  /**
4   * Modelo ArbolSegmentacion
5   *
6   * @package    SIIGS
7   * @subpackage Modelo
8   * @author     Geovanni
9   * @created    2013-12-02
10  */
11 class ArbolSegmentacion_model extends CI_Model {
12
13     private $msg_error_log;
14
15     /**
16      * @access private
17      * @var string
18      */
19     private $msg_error_usr;
20
21     /**
22      * Devuelve los mensajes de error en caso de ocurrir alguna excepción
23      * 'usr' devuelve el mensaje para la vista de usuario
24      * 'log' devuelve el mensaje para el log de errores
25      *
26      * @access public
27      * @return string/boolean
28      * @param string $value, default 'usr' (Tipo mensaje)
29      */
30     public function getMsgError($value = 'usr')
31     {
32         if (!empty($this-> msg_error_usr))
33         {
34             if ($value == 'usr')
35                 return $this-> msg_error_usr;
36             else if ($value == 'log')
37                 return $this-> msg_error_log;
38         }
39         else
40         {
41             return false;
42         }
43     }
44
45     public function __construct()
46     {
47         $this-> load-> database();
48         if (!$this-> db-> conn_id)
49         {
50             throw new Exception("No se pudo conectar a la base de datos");
51         }
52     }
53
54     /**
55      * Regresa la información de los padres de una unidad medica en el ASU
56      *
57      * @access public
58      * @param int $clave Clave de la unidad medica u elemento en el ASU
59      * @return Object arreglo con padres de la um
60      * @throws Exception En caso de algun error al consultar la base de datos
61      */
62
63     public function getUMParentsById($clave)
64     {
```

```

65         $desglose = $this-> db-> query('select grado_segmentacion - 1 as desglose from
66 asu_arbol_segmentacion where id='.$clave);
67
68         if (!$desglose)
69     {
70             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
71 > db-> _error_number().': '.$this-> db-> _error_message();
72             $this-> msg_error_usr = "Ocurrió un error al obtener los datos del arbol
de segmentacion por id";
73             throw new Exception(__CLASS__);
74         }
75
76         $desglose = $desglose-> result()[0]-> desglose;
77
78         //si se van a hacer joins para informacion adicional, se empieza a crear la
79         //estructura de la consulta
80         $consultavalues = 'select a.id as parent0';
81         $consultafrom = ' from asu_arbol_segmentacion a ';
82         $consultawhere = ' where a.id = '.$clave;
83
84         //crear los joins a partir del numero de niveles de desglose requeridos
85         for($i=1;$i<= $desglose;$i++)
86     {
87             $stablajoin = ($i==1) ? array("a" , "tb" . $i) :
88             array("tb" . ($i-1), "tb" . $i);
89             $consultavalues .= ",case when ifnull(tb" . $i." .descripcion,'') =
90             ' then '' else tb" . $i." .id end as parent" . $i;
91             $consultafrom .= " left outer join asu_arbol_segmentacion
92             ".$stablajoin[1]." on " . $stablajoin[0]. ".id_padre =
93             ".$stablajoin[1]. ".id" ;
94             $consulta = $consultavalues . $consultafrom . $consultawhere;
95
96             $query = $this-> db-> query($consulta);
97
98             if (!$query)
99         {
100                 $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
101 > db-> _error_number().': '.$this-> db-> _error_message();
102                 $this-> msg_error_usr = "Ocurrió un error al obtener los datos del arbol
de segmentacion por id";
103                 throw new Exception(__CLASS__);
104             }
105             else
106             {
107                 $resultado = array();
108                 foreach ($query-> result()[0] as $fila => $clave)
109             {
110                 array_push($resultado, $clave);
111             }
112
113             return $resultado;
114         }
115     }
116
117     /**
118      * Regresa el objeto del arbol de segmentacion
119      *
120      * @access public
121      *
122      * @param int $idarbol ID del arbol a crear
123      * @param int $nivel Nivel de segmentacion desde el cual se iniciará a desarrollar el
124      * @param $nivelesocultos Array con los niveles que se deben ocultar
125      *
126      * @return Object un arreglo con la estructura del arbol
127      * @throws Exception En caso de algun error al consultar la base de datos
128      */
129
130     public function getTree($idarbol , $nivel, $nivelesocultos = array())
131     {
132         $consultavalues = " select distinct ";
133         $consultafrom = " from ";
134         //consulta todos los catalogos que forman el arbol de segmentacion requerido
135         $query = $this-> db-> query('select * from asu_raiz_x_catalogo where
136         id_raiz_arbol = ' . $idarbol . ' and grado_segmentacion >= ' . $nivel.' order by
137         grado_segmentacion');
138         if (!$query)
139         {
140             $this-> msg_error_log = "(" . __METHOD__ . ") => "

```

```

.$this-> db-> _error_number()."': '$this-> db-> _error_message();
132                                     $this-> msg_error_usr = "Ocurrió un error al obtener los datos del
arbol de segmentacion";
133                                     ;
134                                     throw new Exception(__CLASS__);
135                                     }
136                                     {
137                                         $cont = 0;
138                                         $padreactivo = "";
139                                         if ($query-> num_rows() == 0)
140                                             return array();
141                                         foreach($query-> result() as $fila)
142                                         {
143                                             //si el valor del grado de segmentacion existe en la lista de niveles
144                                             //ocultos, entonces se aumenta el contador y se guarda el valor del
145                                             //padre
146                                             if (!in_array($fila-> grado_segmentacion,$nivelesocultos))
147                                             {
148                                                 $cont += 1;
149                                             //si no existe ningun catalogo padre activo, se asigna el grado de
150                                             //segmentacion actual
151                                             if ($padreactivo == "")
152                                                 $padreactivo = $fila-> grado_segmentacion;
153                                             //se agrega a la lista de valores el grado de segmentacion del
154                                             //catalogo
155                                             $consultavalue .= " tabla" . $fila-> grado_segmentacion.
156                                             ".grado_segmentacion as nivel_" . $cont.", ";
157                                             if ($cont == 1)
158                                             {
159                                                 //se agrega a la lista de valores el id padre del registro
160                                                 $consultavalue .= "tabla" . $padreactivo.
161                                                 ".id_padre as padre_" . $cont.", ";
162                                             }
163                                             else
164                                             {
165                                                 //se agrega a la lista de valores el id padre del registro
166                                                 $consultavalue .= "tabla" . $padreactivo.
167                                                 ".id as padre_" . $cont.", ";
168                                             }
169                                             $consultavalue .= "tabla" . $fila-> grado_segmentacion.
170                                             ".id as id_" . $cont.", ";
171                                             "tabla" . $fila-> grado_segmentacion.
172                                             ".descripcion as descripcion_" . $cont.", ";
173                                             $padreactivo = $fila-> grado_segmentacion;
174                                         }
175                                         if ($fila-> grado_segmentacion == $nivel)
176                                         {
177                                             $consultafrom .= " asu_arbol_segmentacion tabla" . $fila-
178                                         > grado_segmentacion;
179                                         }
180                                         else
181                                         {
182                                             $consultafrom .= " left outer join asu_arbol_segmentacion
183                                             tabla" . $fila-> grado_segmentacion;
184                                             $consultafrom .= " on tabla" . $fila-
185                                         > grado_segmentacion.".id_padre = tabla" . ($fila-> grado_segmentacion-1)." . id
186                                         ;
187                                         }
188                                         $consultavalue = substr($consultavalue, 0, count($consultavalue)-3);
189                                         $consulta = $consultavalue.$consultafrom. " where
190                                         .$nivel.".grado_segmentacion = " . $nivel. " and
191                                         .$nivel.".id_raiz=" . $idarbol;
192                                         //var_dump($consulta);
193                                         //die();
194                                         $resultado = $this-> db-> query($consulta);
195                                         if (!$resultado)
196                                         {
197                                             $this-> msg_error_log = "(" . __METHOD__ . ") =>
.$this-> db-> _error_number()."': '$this-> db-> _error_message();

```

```

198             $this-> msg_error_usr = "Ocurrió un error al obtener los datos
del arbol de segmentacion" ;
199             throw new Exception(__CLASS__);
200         }
201     }
202     {
203         return array('niveles' => $cont,'resultado' => $resultado-
> result());
204     }
205   }
206 }
207 }
208 /**
209 * Regresa el objeto del arbol de segmentacion por nivel o hijos de un elemento seleccionado
210 *
211 * @access public
212 *
213 * @param int $idarbol ID del arbol a crear
214 * @param int $nivel Nivel de segmentacion desde el cual se iniciará a desarrollar el
arbol
215 * @param $nivelesocultos Array con los niveles que se deben ocultar
216 *
217 * @return Object un arreglo con la estructura del arbol
218 * @throws Exception En caso de algun error al consultar la base de datos
219 */
220
221     public function getTreeBlockData($idarbol , $nivel,$elegido)
222     {
223         if ($nivel> 0)
224         {
225             if ($elegido > 0)
226             {
227                 $nivelpadre = $this-> db-> query("select grado_segmentacion as
nivel fromasu_arbol_segmentacion where id=" . $elegido)-> result()[0]-> nivel;
228                 $consulta = "select tabla" . ($nivel).".id_padre as parent,
tabla" . ($nivel).".id as id, tabla" . ($nivel).".grado_segmentacion as nivel,
tabla" . ($nivel).".descripcion as descripcion fromasu_arbol_segmentacion
.tabla" . $nivelpadre;
229                 $tempnivel = $nivelpadre;
230                 while ($nivelpadre< $nivel)
231                 {
232                     $nivelpadre +=1;
233                     $consulta .= " joinasu_arbol_segmentacion
.tabla" . ($nivelpadre).".on tabla" . ($nivelpadre).".id_padre =
.tabla" . ($nivelpadre-1).".id" ;
234                     }
235                     $consulta .= " where
.tabla" . ($tempnivel).".id=" . $elegido . " order by
.tabla" . ($nivelpadre).".descripcion" ;
236
237 //var_dump($consulta);
238 $query = $this-> db-> query($consulta);
239 if (!$query)
240 {
241     $this-> msg_error_log = "(" . __METHOD__ . ") =>
" . $this-> db-> _error_number().':'. $this-> db-> _error_message();
242     $this-> msg_error_usr = "Ocurrió un error al obtener los datos
del arbol de segmentacion" ;
243     throw new Exception(__CLASS__);
244 }
245 else
246 {
247     return array('resultado' => $query-> result());
248 }
249
250 //var_dump($consulta);
251 }
252 else
253 {
254     $consulta = "select id_padre as parent, id,grado_segmentacion as
nivel,descripcion fromasu_arbol_segmentacion where id_raiz = " . $idarbol . " and
grado_segmentacion = " . $nivel. " order by descripcion" ;
255     $query = $this-> db-> query($consulta);
256     if (!$query)
257     {
258         $this-> msg_error_log = "(" . __METHOD__ . ") =>
" . $this-> db-> _error_number().':'. $this-> db-> _error_message();
259         $this-> msg_error_usr = "Ocurrió un error al obtener los datos
del arbol de segmentacion" ;

```

```

261             throw new Exception(__CLASS__);
262         }
263     }
264     {
265         return array('resultado' => $query-> result());
266     }
267 }
268 else if ($elegido > 0)
269 {
270     $consulta = "select id,grado_segmentacion as nivel,descripcion from
271 asu_arbol_segmentacion where id_raiz = " . $idarbol . " and id_padre =
272 " . $elegido . " order by descripcion";
273     $query = $this-> db-> query($consulta);
274     if (!$query)
275     {
276         $this-> msg_error_log = "(" . __METHOD__ . ") => "
277 . $this-> db-> _error_number().":'." . $this-> db-> _error_message();
278         $this-> msg_error_usr = "Ocurrió un error al obtener los datos del
279 arbol de segmentacion";
280         throw new Exception(__CLASS__);
281     }
282     else
283     {
284         return array('resultado' => $query-> result());
285     }
286 }
287 /**
288 *
289 * Obtiene las unidades medicas correspondientes a un ID de un elemento
290 * independientemente su nivel en el ASU
291 *
292 * @param int $id Id del elemento en el asu
293 *
294 * * @return Object un arreglo con la estructura del arbol
295 * @throws Exception En caso de algun error al consultar la base de datos
296 */
297
298 public function getCluesFromId($id)
299 {
300     $datos = $this-> db-> query("select * from asu_arbol_segmentacion where
301 id=" . $id);
302     if (!$datos)
303     {
304         $this-> msg_error_log = "(" . __METHOD__ . ") => "
305 . $this-> db-> _error_number().":'." . $this-> db-> _error_message();
306         $this-> msg_error_usr = "Ocurrió un error al obtener los datos del arbol
307 de segmentacion por ID";
308         throw new Exception(__CLASS__);
309     }
310     else
311     {
312         $nivelmaximo = $this-> db-> query("select max(grado_segmentacion) as
313 maximo from asu_arbol_segmentacion where id_raiz=" . $datos-> result()[0]-> id_raiz);
314         $nivel = $datos-> result()[0]-> grado_segmentacion;
315         $nivelmaximo = $nivelmaximo-> result()[0]-> maximo;
316         $omitidos = array();
317         for($i = $nivel+1 ; $i < $nivelmaximo ; $i++)
318             array_push($omitidos, $i);
319         return $this-> getChildrenFromId($id , $omitidos);
320     }
321 /**
322 * Accion para devolver los hijos de un elemento en el ASU a partir de su ID
323 *
324 * @param int $id Id del elemento en el ASU
325 * @param Array int $omitidos Array de niveles omitidos
326 *
327 * @return Object un arreglo con la estructura del arbol
328 * @throws Exception En caso de algun error al consultar la base de datos
329 */
330
331 public function getChildrenFromId($id , $omitidos = array())
332 {

```

```

333         $datos = $this-> db-> query("select * fromasu_arbol_segmentacion where
334         .$id);
335
336         if (!$datos)
337         {
338             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
339             > db-> _error_number().': ' . $this-> db-> _error_message();
340             $this-> msg_error_usr = "Ocurrió un error al obtener los datos del arbol
341             de segmentacion por ID" ;
342             throw new Exception(__CLASS__);
343         }
344         else
345         {
346             if (count($datos-> result()) == 0)
347                 return array();
348
349             //var_dump($datos->result()[0]->id_raiz);
350             //var_dump($datos->result()[0]->grado_segmentacion);
351             //var_dump($omitidos);
352
353             $arregloparcial = $this-> getChildrenFromLevel($datos-> result()[0]-
354             > id_raiz, $datos->
355             result()[0]-> grado_segmentacion,$omitidos);
356             $resultado = array();
357             //var_dump($arregloparcial);
358
359             foreach ($arregloparcial as $arreglo)
360             {
361                 if ($arreglo["key"] == $id)
362                     return $arreglo;
363             }
364
365         /**
366          * Obtener el elemento del ASU por medio de su ID
367          * @param int item seleccionado
368          * @return object con los datos del elemento
369          *
370          * regresa array vacio si el elemento no existe en el ASU
371          *
372          * @throws Exception En caso de algun error al consultar la base de datos
373          */
374
375         public function getById($item)
376         {
377             $resultado = $this-> db-> query("select * fromasu_arbol_segmentacion
378             where id=" . $item);
379             if (!$resultado)
380             {
381                 $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
382                 > db-> _error_number().': ' . $this-> db-> _error_message();
383                 $this-> msg_error_usr = "Ocurrió un error al obtener el elemento
384                 padre" ;
385                 throw new Exception(__CLASS__);
386             }
387             else
388             {
389                 if ($resultado-> num_rows() > 0)
390                     return $resultado-> result();
391                 else
392                     return array();
393             }
394
395         /**
396          * Convierte el tipo de arreglo para enviarlo como se debe recibir en el cliente de
397          * @param Array fila array de valores
398          * @param bool Seleccionable opcion para que este elemento sea seleccionable
399          * @return Array Preparado para el javascript cliente
400          */
401
402         public function convertType($arbol,$seleccionable, $seleccionados)
403         {
404             $resultado = array();
405             foreach($arbol as $fila)
406             {
407                 $fila = (array) $fila;
408                 if ($fila['id'] != null)

```

```

405
406         {
407             $arraytemp = array('key' => $fila['id'], 'title'=>
408             $fila['descripcion'], 'tooltip'=> $fila['nivel']);
409             if (isset($fila['children']))
410                 $arraytemp["children"] = $fila['children'];
411             else
412                 $arraytemp["isLazy"] = true;
413             if (!$seleccionable)
414             {
415                 $arraytemp["unselectable"] = true;
416                 $arraytemp["hideCheckbox"] = true;
417             }
418             if (in_array($fila["id"], $seleccionados))
419             {
420                 $arraytemp["select"] = true;
421             }
422             array_push($resultado, $arraytemp);
423         }
424     }
425
426 /**
427 * Accion para devolver un bloque del ASU a partir de un nivel especificado o una clave
428 seleccionada, niveles omitidos y elementos preseleccionados
429 *
430 * @param Int $idarbol Id del arbol a crear
431 * @param Int $nivel Nivel de segmentacion desde la cual se desarrolla el arbol
432 * @param Array int $omitidos Array de niveles omitidos
433 * @param Array int $seleccionados Array de elementos seleccionados
434 * @param Array int $seleccionables Array de niveles que son seleccionables
435 * @param Array int $elegido Clave seleccionada para obtener sus hijos
436 *
437 * @return Object un arreglo con la estructura del arbol
438 * @throws Exception En caso de algun error al consultar la base de datos
439 */
440
441 public function getTreeBlock($idarbol, $nivel, $seleccionados = array(),
442 $seleccionable, $elegido, $omitidos = array(), $seleccionables = array())
443 {
444     try
445     {
446         $fecha_update_asu = $this-> db-> query("select max(fecha_update) as
447 fecha fromasu_arbol_segmentacion where id_raiz=" . $idarbol);
448         if (!$fecha_update_asu)
449         {
450             $this-> msg_error_log = "(" . __METHOD__ . ") => "
451             . $this-> db-> _error_number() . ':' . $this-> db-> _error_message();
452             $this-> msg_error_usr = "Ocurrió un error al obtener la ultima
453 actualización delasu";
454             return "false";
455         }
456         $fecha_update_asu = $fecha_update_asu-> result()[0]-> fecha;
457         $ruta =
458             __DIR__.DIRECTORY_SEPARATOR.'...'.DIRECTORY_SEPARATOR.'...'.DIRECTORY_SEPARATOR.'json'.DIRECTORY_SEPARATOR;
459         $archivo =
460             'asu_data_'.$idarbol.'_'.$nivel.'_'.$elegido.'_'.$seleccionable.'_'.$strtotime($fecha_update_asu).'.json';
461         $ruta_temp = $ruta;
462         if (!is_dir($ruta))
463         {
464             if (!mkdir($ruta, 0777, true))
465                 return "false";
466         }
467         $ruta.= $archivo;
468         if (file_exists($ruta) && (
469             ) && count($seleccionados)==0 ||
470             $seleccionados[0]=="
471         {
472             $str_datos = file_get_contents($ruta);
473             $datos = json_decode($str_datos,true);
474             //if (count($seleccionados)>0)
475             $datos = $this-> addSelectedItems($datos, $nivel, $seleccionados,
476             $seleccionable);
477             //var_dump($datos);
478             return $datos;

```

```

473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
>     grado_segmentacion, $item[0]-> id_padre);
507
508
509
510
511
512
513
514
515
516
517
foreach($resultadotemp[$ultimaclave]["resultado"] as $clave => $valor)
518
{
519
//      echo
520
$resultadotemp[$ultimaclave]["resultado"][$clave]->parent."...".$itempadre[0]-
>id_padre."<br/>" ;
521
}
522
523
//      echo
524
$resultadotemp[$ultimaclave];
525
//      resultadotemp[$ultimaclave]["resultado"] = null;
526
}
527
}
528
}
529
}
530
}
531
}
532
$resultado = array();
533
//      foreach ($resultadotemp as $clave => $valor)
534
//{
535
//      echo $clave;
536
//      var_dump($resultadotemp[$clave]);
537
//      echo "<br/><br/>";
538
//      die();
539
//      $search = 0;
540
541
542
//Nivel de busqueda para ir agregando hijos a padres
543

```

```

544 //el nivel mayor en la lista de items
545 $search = $this-> getListChildrenLevel($resultadotemp);
546 $contador = 0;
547 //
548 //
549 //
550 while($search["nivel"] > 1/* && $contador<200*/ )
551 {
552     $contador +=1;
553     $clave1 = $search["id"];
554     foreach($resultadotemp as $clave1 => $valor1)
555     {
556         $childrens = array();
557         $childrens = $this-> getListChildrenLevel($resultadotemp,$clave1);
558         var_dump($childrens);
559         echo "<br/><br/>";
560         foreach($childrens as $clave2)
561         {
562             //
563             //echo
564             $clave1.".....".$clave2."<br/><br/>";
565             foreach($resultadotemp[$clave2]["resultado"] as
566             $clave=> $valor)
567             {
568                 if (!empty($resultadotemp[$clave1]["resultado"]
569                 && $resultadotemp[$clave2]["resultado"]
570                 )[$clave]-> id ==
571                 $resultadotemp[$clave1]["resultado"]
572                 )[$0]-> parent)
573                 {
574                     //echo
575                     $resultadotemp[$clave2]["resultado"]
576                     .".$clave1."<br/>";
577                     $resultadotemp[$clave2]["resultado"]
578                     [$clave]->
579                     children = $this-
580                     convertType($resultadotemp[$clave1]["resultado"]
581                     ,in_array($resultadotemp[$clave1][ "resultado"
582                     ][0]-> nivel, $seleccionables),$seleccionados);
583                     $resultadotemp[$clave2]["resultado"]
584                     [0] = null;
585                     continue;
586                 }
587             }
588         }
589     }
590     $search = $this-> getListChildrenLevel($resultadotemp);
591     //var_dump($search);
592 }
593 $resultadotemp[0]["resultado"] = $this-
594 convertType($resultadotemp[0][ "resultado"
595 ,in_array($resultadotemp[0][ "resultado" &quot;
596 ot;][0]-> nivel, $seleccionables),$seleccionados);
597 return $resultadotemp[0][ "resultado" ];
598 }
599 else
600 {
601     $arbol = $this-> getTreeBlockData($idarbol, $nivel,$elegido);
602     if (count($arbol['resultado']) == 0)
603     {
604         return array();
605     }
606     $resultado = $this-> convertType($arbol[ "resultado" ,
607     $seleccionable, $seleccionados);
608     try
609     {
610         $fh = fopen($ruta, 'c');
611         if(!$fh)
612         {
613             return "false";
614         }
615         fwrite($fh, json_encode($resultado,JSON_UNESCAPED_UNICODE));
616         fclose($fh);
617     }
618     $fechas = $this->db->query("select distinct
619 fecha_update as fecha from asu_arbol_segmentacion where id_raiz=".$idarbol);
620     if (!$fechas)
621     {

```

```

611    //                                     $this->msg_error_log = "(" . __METHOD__ . ")"
=> " . $this->db->_error_number()." : '$this->db->_error_message()';
612    //                                     $this->msg_error_usr = "Ocurrió un error al obtener el
registro de actualizaciones del asu";
613    //                                     return "false";
614    //
615    //                                     else
616    //
617    //                                     foreach ($fechas->result() as $item)
618    //
619    //                                     $archivotemp =
'asu_data_'.$idarbol.'_'.$nivel.'_'.$selegido.'_'.$seleccionable.'_'.$strtotime($item-
>fecha).'_.json';
620    //
621    //                                     if ($archivo != $archivotemp)
622    //                                         if (file_exists($ruta_temp.$archivotemp))
623    //                                             unlink($ruta_temp.$archivotemp);
624    //
625    //                                     }
626    //                                     catch(Exception $e)
627    //                                     {
628    //                                         $this->msg_error_log = "(" . __METHOD__ . ") =>
".ASU'.': '".No se pudo crear el archivo JSON para el asu (".$ruta.")
: "'.$e->getMessage();
629    //
630    //                                         return "false";
631    }
632    //
633    }
634    catch(Exception $e)
635    {
636        return "false";
637    }
638}
639
640 public function getListChildrenLevel($resultadotemp, $clave2=0)
641 {
642     if ($clave2==0)
643     {
644         $nivel = array("nivel"      => 0, "id"           => 0);
645         foreach($resultadotemp as $clavel => $valor1)
646         {
647             if (!empty($resultadotemp[$clavel]["resultado"] ))
648             {
649                 if ($resultadotemp[$clavel]["resultado"] )
650                 > nivel> $nivel["nivel"] );
651                 $nivel =
array("nivel"      => $resultadotemp[$clavel]["resultado"]
) );
652                 > nivel,"id"
653                 => $clavel);
654                 }
655                 return $nivel;
656             }
657             else
658             {
659                 $lista = array();
660                 foreach($resultadotemp as $clavel => $valor1)
661                 {
662                     if (!empty($resultadotemp[$clavel]["resultado"] ) &&
!empty($resultadotemp[$clave2]["resultado"] ))
663                     {
664                         if($resultadotemp[$clavel]["resultado"] )
665                         > nivel< $resultadotemp[$clave2]["resultado"] );
666                         //echo "coincidencia...".$clavel;
667                         array_push($lista, $clavel);
668                         }
669                         }
670                         }
671                         return $lista;
672                     }
673                 }
674             }
675             public function checkChildren($arreglo,$indice,$seleccionables,$seleccionados) {
676                 foreach($arreglo[0]["resultado"] as $clave2 => $valor2)
677                 {
678                     if ($arreglo[0]["resultado"][$clave2]->id ==

```

```

$arreglo[$clave1]["resultado"])[0]->parent)
679 // {
680 //     $arreglo[0]["resultado"][$clave2]->children = $this-
>convertType($arreglo[$clave1]["resultado"],in_array($arreglo[$clave1]["resultado"]&qu
ot;)[0]->nivel, $seleccionables,$seleccionados);
681 //     $arreglo[$clave1]["resultado"] = null;
682 //     break;
683 //
684 //     else if (isset($arreglo[0]["resultado"][$clave2]->children))
685 //     {
686 //         $arreglo[0]["resultado"][$clave2]->children = $this-
>checkChildren($arreglo, $clave2, $seleccionables, $seleccionados) ;
687 //     }
688 // }
689 // return $arreglo;
690 //}
691
692 /**
693 * Accion para devolver el esquema completo del ASU a partir de un nivel especificado,
694 niveles omitidos y elementos preseleccionados
695 *
696 * @param Int $idarbol Id del arbol a crear
697 * @param Int $nivel Nivel de segmentacion desde la cual se desarrolla el arbol
698 * @param Array int $omitidos Array de niveles omitidos
699 * @param Array int $seleccionados Array de elementos seleccionados
700 * @param Array int $seleccionables Array de niveles que son seleccionables
701 *
702 * @return Object un arreglo con la estructura del arbol
703 * @throws Exception En caso de algun error al consultar la base de datos
704 */
705
706 public function getChildrenFromLevel($idarbol, $nivel , $omitidos = array() ,
707 $seleccionados = array(), $seleccionables = array())
708 {
709     try
710     {
711         $fecha_update_asu = $this-> db-> query("select max(fecha_update) as
712 fecha fromasu_arbol_segmentacion where id_raiz=" . $idarbol);
713         if (!$fecha_update_asu)
714         {
715             $this-> msg_error_log = " (" . __METHOD__ . ") => "
716 . $this-> db-> _error_number(). ': ' . $this-> db-> _error_message();
717             $this-> msg_error_usr = "Ocurrió un error al obtener la ultima
718 actualizacion del asu";
719             return "false";
720         }
721         $fecha_update_asu = $fecha_update_asu-> result()[0]-> fecha;
722         $ruta =
723         __DIR__.DIRECTORY_SEPARATOR.'...'.DIRECTORY_SEPARATOR.'...'.DIRECTORY_SEPARATOR.'json'.DIRECTORY_SEPARAT
724 OR;
725         $archivo = 'asu_data_'.$idarbol.'_'.$nivel.'_'.$_implode(',',$_
726 omitidos).'.'.strtotime($fecha_update_asu).'.json';
727         $ruta_temp = $ruta;
728         if (!is_dir($ruta))
729         {
730             if (!mkdir($ruta, 0777, true))
731                 return "false";
732         }
733         $ruta.= $archivo;
734         if (file_exists($ruta))
735         {
736             $str_datos = file_get_contents($ruta);
737             $datos = json_decode($str_datos,true);
738             //if (count($seleccionados)>0)
739             $datos = $this-> addSelectedItems($datos,$seleccionados, $nivel ,
740 $omitidos, $seleccionables);
741             //var_dump($datos);
742             ini_set('max_execution_time',1000);
743             ini_set('memory_limit', '1024M');
744             $arbol = $this-> getTree($idarbol, $nivel, $omitidos);
745         }
    }
}

```

```

746
747 //var_dump($omitidos);
748 //echo $nivel."<br/><br/>";
749 //var_dump(json_encode($arbol));
750 //die();
751
752
753     if (count($arbol) == 0)
754     {
755         return array();
756     }
757
758     if ($nivel <= $arbol['niveles'] || true)
759     {
760         $resultado = array();
761         $niveltemp = array();
762
763         for($i = 1 ; $i <= $arbol['niveles'];$i++)
764         {
765             $esseleccionable = false;
766             if (count($seleccionables)==0)
767                 $esseleccionable = true;
768             else if (in_array($i, $seleccionables))
769                 $esseleccionable = true;
770
771             foreach($arbol['resultado'] as $fila)
772             {
773                 $fila = (array) $fila;
774                 if ($fila['id_'][$i] != null)
775                 {
776                     if ($i == $arbol['niveles'])
777                         $arraytemp = array('key' => $fila['id_'][$i],
778 'parent' => $fila['padre_'][$i], 'title'=> ($fila['descripcion_'][$i]));
779                     else
780                         $arraytemp = array('key' => $fila['id_'][$i],
781 'parent' => $fila['padre_'][$i], 'title'=> ($fila['descripcion_'][$i]), 'children'=>array());
782
783                     if (in_array($fila['id_'][$i],$seleccionados))
784                         $arraytemp["select"] = true;
785
786                     if (!$esseleccionable)
787                     {
788                         $arraytemp["unselectable"] = true;
789                         $arraytemp["hideCheckbox"] = true;
790                     }
791
792                     if (!isset($resultado[$i]))
793                         $resultado[$i] = array();
794
795                     if ( !in_array($arraytemp,$resultado[$i]))
796                     {
797                         array_push($resultado[$i], $arraytemp);
798                     }
799                 }
800             }
801             for($i = count($resultado) ; $i > 1;$i--)
802             {
803                 $arreglohijo = $resultado[$i];
804                 $arreglopadre = $resultado[$i-1];
805                 foreach ($arreglohijo as $clave1 => $hijo)
806                 {
807                     foreach ($arreglopadre as $clave2 => $padre)
808                     {
809                         if ($hijo['parent'] == $padre['key'])
810                         {
811                             array_push($arreglopadre[$clave2]['children'],
812 $resultado[$i][$clave1]);
813                         }
814                     }
815                 }
816                 $resultado[$i-1] = $arreglopadre;
817             }
818
819             try
820             {
821                 $fh = fopen($ruta, 'c');
822                 if (!$fh)

```

```

822             {
823                 return "false" ;
824             }
825
826
827             fwrite($fh, json_encode($resultado[1],JSON_UNESCAPED_UNICODE));
828             fclose($fh);
829
830             $fechas = $this-> db-> query("select distinct fecha_update
831             as fecha fromasu_arbol_segmentacion where id_raiz=" . $idarbol);
832             if (!$fechas)
833             {
834                 $this-> msg_error_log = "(" . __METHOD__ . ")"
835                 . $this-> db-> _error_number() . ":" . $this-> db-> _error_message();
836                 $this-> msg_error_usr = "Ocurrió un error al obtener el
837                 registro de actualizaciones del asu" ;
838                 return "false" ;
839             }
840             else
841             {
842                 foreach ($fechas-> result() as $item)
843                 {
844                     $archivotemp = 'asu_data_' . $idarbol . '_' . $nivel . '_'.
845                     implode(' ', $omitidos) . '_' . strtotime($item-> fecha) . '.json';
846                     if ($archivo != $archivotemp)
847                         if (file_exists($ruta_temp . $archivotemp))
848                             unlink($ruta_temp . $archivotemp);

849                 }
850             }
851             catch(Exception $e)
852             {
853                 $this-> msg_error_log = "(" . __METHOD__ . ") =>
854                 ." . $ruta . ")";
855                 . $e-> getMessage();
856                 return "false" ;
857             }
858             return $resultado[1];
859         }
860     }
861
862
863     public function _addSelectedItems($datos,$nivel, $seleccionados, $seleccionables)
864     {
865
866         foreach($datos as $clave => $dato)
867         {
868             if (in_array($dato["key"], $seleccionados))
869             {
870                 $datos[$clave]["select"] = true;
871             }
872
873             if(gettype($seleccionables)=='array')
874             {
875                 if (!in_array($nivel,$seleccionables))
876                 {
877                     $datos[$clave]["unselectable"] = true;
878                     $datos[$clave]["hideCheckbox"] = true;
879                 }
880             }
881             else
882             {
883                 if ($seleccionables==false)
884                 {
885                     $datos[$clave]["unselectable"] = true;
886                     $datos[$clave]["hideCheckbox"] = true;
887                 }
888             }
889         }
890         return $datos;
891     }
892
893
894     public function _addSelectedItems($datos,$seleccionados, $nivel, $omitidos,

```

```

$seleccionables)
895     {
896         while(in_array($nivel, $omitidos))
897             $nivel += 1;
898
899         //var_dump($nivel);
900
901         foreach($datos as $clave => $dato)
902         {
903             if (array_key_exists('children', $dato) &&
904                 count($dato['children']) > 0)
905             {
906                 $nivel += 1;
907                 $datos[$clave]['children'] = $this->
908                 > _addSelectedItems_($datos[$clave]['children'], $seleccionados, $nivel, $omitidos,
909                 $seleccionables);
910                 $nivel -= 1;
911             }
912             if (in_array($dato["key"] , $seleccionados))
913             {
914                 $datos[$clave]["select"] = true;
915             }
916             if (count($seleccionables) > 0)
917             if (!in_array($nivel, $seleccionables))
918             {
919                 // echo $nivel;
920                 // var_dump($seleccionables);
921                 // echo "<br/><br/><br/>";
922                 $datos[$clave]["unselectable"] = true;
923                 $datos[$clave]["hideCheckbox"] = true;
924             }
925         }
926     }
927     /**
928      * Accion para obtener la descripcion e informacion adicional del elemento en el ASU
929      *
930      * @param Array int $claves arreglo de valores a recuperar
931      * @param Int $desglose Nivel de desglose de informacion requerida
932      * @return Object
933      * @throws Exception Si ocurre error al recuperar datos de la base de datos
934      */
935     public function getDescripcionById($claves, $desglose = 0)
936     {
937         if (count($claves) > 0)
938         {
939             if ($desglose == 0)
940                 $consulta = 'select id,descripcion fromasu_arbol_segmentacion where id in
941 (' . implode(',', $claves) . ')';
942             else
943                 {
944                     //si se van a hacer joins para informacion adicional, se empieza a crear la
945                     //estructura de la consulta
946                     $consultavalue = 'select a.id,concat("",a.descripcion' ;
947                     $consultafrom = ' ) as descripcion fromasu_arbol_segmentacion a ' ;
948                     $consultawhere = ' where a.id in (' . implode(',', $claves) . ')';
949                     //crear los joins a partir del numero de niveles de desglose requeridos
950                     for($i=1;$i<= $desglose;$i++)
951                     {
952                         $stablajoin = ($i==1) ? array("a" , "tb" . $i) :
953                         array("tb" . ($i-1), "tb" . $i);
954                         $consultavalue .= ",case when
955                         ifnull(tb" . $i . ".descripcion,'') = '' then '' else concat('
956                         ,tb" . $i . ".descripcion) end" ;
957                         $consultafrom .= " left outer joinasu_arbol_segmentacion
958                         ." . $stablajoin[1] . " on " . $stablajoin[0] . ".id_padre =
959                         ." . $stablajoin[1] . ".id" ;
960                         $consultawhere = $consultavalue . $consultafrom . $consultawhere;
961                         //var_dump($consulta);
962                     }
963                     $query = $this-> db-> query($consulta);
964
965                     if (!$query)
966                     {
967                         $this-> msg_error_log = "(" . __METHOD__ . ") => "
968                         . $this-> db-> _error_number() . ': ' . $this-> db-> _error_message();
969

```

```

963             $this-> msg_error_usr = "Ocurrió un error al obtener los datos del
964 arbol de segmentacion por id";
965         }
966     }
967     {
968         return $query-> result();
969     }
970 }
971 }
972 */
973 /**
974 * Accion para obtener la descripcion e informacion adicional del elemento en el ASU
975 * @param Array int $claves arreglo de valores a recuperar
976 * @param Int $desglose Nivel de desglose de informacion requerida
977 * @return Object
978 * @throws Exception Si ocurre error al recuperar datos de la base de datos
979 */
980 public function isChild($claves,$desglose = 0)
981 {
982     if (count($claves)>0)
983     {
984         if ($desglose == 0)
985             $consulta = 'select id,descripcion fromasu_arbol_segmentacion where id in
986 ('.$implode(',',$claves).')';
987         else
988             {
989                 //si se van a hacer joins para informacion adicional, se empieza a crear la
990                 //estructura de la consulta
991                 $consultavalues = 'select a.id,concat("",a.descripcion',
992                 $consultafrom = ' ) as descripcion fromasu_arbol_segmentacion a ';
993                 $consultawhere = ' where a.id in ('.$implode(',',$claves).')';
994
995                 //crear los joins a partir del numero de niveles de desglose requeridos
996                 for($i=1;$i<=$desglose;$i++)
997                 {
998                     $stablajoin = ($i==1) ? array("a","tb".$i) :
999                     array("tb".($i-1),"tb".$i);
1000                     $consultavalues .= ",case when
1001                     ifnull(tb".$i.".descripcion,'') = '' then '' else concat(
1002                     ',tb".$i.".descripcion) end";
1003                     $consultafrom .= " left outer joinasu_arbol_segmentacion
1004                     ".$stablajoin[1]." on ".$stablajoin[0].".id_padre =
1005                     ".$stablajoin[1].".id";
1006                     }
1007                     $consulta = $consultavalues . $consultafrom . $consultawhere;
1008                     //var_dump($consulta);
1009
1010                     $query = $this->db->query($consulta);
1011
1012                     if (!$query)
1013                     {
1014                         $this->msg_error_log = (".". __METHOD__ ."") =>
1015                         ".$this->db->error_number().":'".$this->db->error_message();
1016                         $this->msg_error_usr = "Ocurrió un error al obtener los datos del
1017                         arbol de segmentacion por id";
1018                         throw new Exception(__CLASS__);
1019                     }
1020                     else
1021                     {
1022                         return $query->result();
1023                     }
1024
1025 /**
1026 * Accion para obtener los registros de un ASU determinado en cierto nivel y con un ID
1027 de filtro
1028 */
1029
1030 public function getDataKeyValue($idarbol,$nivel,$filtro = 0)

```

```

1032     {
1033         $consulta = 'select id,descripcion from asu_arbol_segmentacion where
id_raiz='.$idarbol." and grado_segmentacion=" . $nivel.((&$filtro != 0) ? " and
id_padre=" . $filtro : '').' order by descripcion';
1034         $query = $this-> db-> query($consulta);
1035
1036         if (!$query)
1037         {
1038             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
1039             $this-> msg_error_usr = "Ocurrió un error al obtener los datos del arbol
de segmentacion por nivel y filtro";
1040             throw new Exception(__CLASS__);
1041         }
1042         else
1043         {
1044             return $query-> result();
1045         }
1046     }
1047 }

```

# Archivo fuente para bitacora\_model.php

La documentación para este archivo está disponible en [bitacora\\_model.php](#)

```
1  <?php
2
3  /**
4   * Modelo Bitacora
5   *
6   * @package    SIIGS
7   * @subpackage Modelo
8   * @author     Pascual
9   * @created    2013-09-26
10  */
11 class Bitacora_model extends CI_Model
12 {
13     /**
14      * Guarda la instancia del objeto global CodeIgniter
15      * para utilizarlo en la función estática
16      *
17      * @access private
18      * @var    instance
19      */
20     private static $CI;
21
22     /**
23      * @access private
24      * @var    int
25      */
26     private $id;
27
28     /**
29      * @access private
30      * @var    int
31      */
32     private $id_usuario;
33
34     /**
35      * @access private
36      * @var    datetime
37      */
38     private $fecha_hora;
39
40     /**
41      * @access private
42      * @var    string
43      */
44     private $parametros;
45
46     /**
47      * @access private
48      * @var    int
49      */
50     private $id_controlador_accion;
51
52     /**
53      * Estas variables no pertenecen a la tabla *
54      * ****
55      */
56     /**
57      * @access private
58      * @var    int
59      */
60     private $id_controlador;
61
62     /**
63      * @access private
64      * @var    int
65      */
66     private $id_accion;
67
```

```

68     /**
69      * @access private
70      * @var array
71     */
72    private $filters;
73
74    /**
75     * @access private
76     * @var boolean
77    */
78    private $error;
79
80    /**
81     * @access private
82     * @var string
83    */
84    private $msg_error_usr;
85
86    /**
87     * @access private
88     * @var string
89    */
90    private $msg_error_log;
91
92
93    public function __construct()
94    {
95        parent::__construct();
96
97        self::$CI = & get_instance();
98
99        $this-> load-> database();
100       $this-> fecha_hora = date('Y-m-d H:i:s');
101       $this-> error = false;
102       $this-> msg_error_usr = '';
103       $this-> msg_error_log = '';
104       $this-> filters = array();
105
106      if( !$this-> db-> conn_id ) {
107          echo '<div class="error">ERROR: No se puede conectar con la Base de
Datos</div>';
108          ;
109          die();
110          //return false;
111          //throw new Exception ('ERROR: No se puede conectar con la Base de Datos');
112      }
113
114      public function getId()
115      {
116          return $this-> id;
117      }
118
119      public function setId($id)
120      {
121          return $this-> id = $id;
122      }
123
124      public function getId_usuario()
125      {
126          return $this-> id_usuario;
127      }
128
129      public function setId_usuario($id_usuario)
130      {
131          $this-> id_usuario = $id_usuario;
132      }
133
134      public function getFecha_hora()
135      {
136          return $this-> fecha_hora;
137      }
138
139      public function setFecha_hora($fecha_hora)
140      {
141          $this-> fecha_hora = $fecha_hora;
142      }
143
144      public function getParametros()
145      {
146          return $this-> parametros;

```

```

147     }
148
149     public function setParametros($parametros)
150     {
151         $this-> parametros = $parametros;
152     }
153
154     public function getId_controlador_accion()
155     {
156         return $this-> id_controlador_accion;
157     }
158
159     public function setId_controlador_accion($id_controlador_accion)
160     {
161         $this-> id_controlador_accion = $id_controlador_accion;
162     }
163
164     public function setId_controlador($id_controlador)
165     {
166         $this-> id_controlador = $id_controlador;
167     }
168
169     public function setId_accion($id_accion)
170     {
171         $this-> id_accion = $id_accion;
172     }
173
174 /**
175 * Devuelve el mensaje de error,
176 * en caso de existir un error despues de ejecutar un metodo,
177 * de lo contrario false
178 *
179 * @access public
180 * @param string $type usr si se quiere devolver el mensaje de error a mostrar en la vista,
181 *                      log obtiene el mensaje de error con mas detalles para depuración,
182 *                      valor por defecto usr
183 * @return boolean/string
184 */
185 public function getMsgError($type = 'usr')
186 {
187     if($this-> error) {
188
189         if($type == 'usr')
190             return $this-> msg_error_usr;
191         else if($type == 'log')
192             return $this-> msg_error_log;
193         else
194             return false;
195     }
196
197     return false;
198 }
199
200 /**
201 * Inserta a la bitacora la información proporcionada
202 *
203 * @access public static
204 * @param string $path Concatenación de directorio del proyecto, clase y metodo, unidos por
205 * dos dobles puntos '::'
206 * @param string $parametros
207 * @return boolean false Si no se ejecutó la inserción, true si se ejecutó la inserción
208 */
209 public static function insert($path, $parametros)
210 {
211     self::$CI-> load-> model('siigs/ControladorAccion_model');
212     $id_controlador_accion = self::$CI-> ControladorAccion_model-> getIdByPath($path);
213
214     if(empty($id_controlador_accion)) {
215         Errorlog_model::insert(DIR_SIIGS.'::::'.__METHOD__, '(Bitacora_model::insert) No se
216 encuentra la relación entre el controlador y la acción: '.$path.', Error '.self::$CI-> db-
217 > _error_number().': '.self::$CI-> db-> _error_message());
218
219     $data = array(
220         'id_usuario' => self::$CI-> session-> userdata(USER_LOGGED),
221         'fecha_hora' => date('Y-m-d H:i:s'), // inserta con la fecha y hora actual del
222         'sistema',
223         'parametros' => $parametros,
224         'id_controlador_accion' => $id_controlador_accion
225     );

```

```

223         self::$CI-> db-> insert('sis_bitacora', $data);
224
225     if(self::$CI-> db-> _error_number()) {
226         log_message('error', '(Bitacora_model::insert) Usuario: '.self::$CI-> session-
227 > userdata(USER_LOGGED).', Path: '$path.', Parametros: '$parametros.', Error '.self::$CI-> db-
228 > _error_number().' : '.self::$CI-> db-> _error_message());
229         return false;
230     }
231
232     return true;
233 }
234 /**
235 * Actualiza los datos del objeto actual
236 *
237 * @access public
238 * @param int $id Si no se establece el valor de ID, se toma el valor del objeto actual
239 * @return boolean false Si no se ejecutó la actualización, true si se ejecutó la actualización
240 */
241 public function update($id = null)
242 {
243     $result = false;
244
245     // Obtener el id_controlador_accion a partir del id_controlador y el id_accion
246     if(empty($this-> id_controlador_accion)) {
247         $this-> load-> model('siigs/ControladorAccion_model');
248         $this-> id_controlador_accion = $this-> ControladorAccion_model-> getId($this-
249 > id_controlador,$this-> id_accion);
250     }
251
252     $data = array(
253         'id_usuario' => $this-> id_usuario,
254         'fecha_hora' => $this-> fecha_hora,
255         'parametros' => $this-> parametros,
256         'id_controlador_accion' => $this-> id_controlador_accion
257     );
258
259     $id = is_null($id) ? $this-> id : $id;
260     $result = $this-> db-> update('sis_bitacora', $data, array('id' => $id));
261
262     if(empty($result)) {
263         $this-> error = true;
264         $this-> msg_error_usr = 'No se puede actualizar el registro';
265         $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
266 > _error_number().' : '.$this-> db-> _error_message();
267         throw new Exception(__CLASS__);
268     }
269
270     return $result;
271 }
272 /**
273 * Elimina el registro actual de la base de datos
274 *
275 * @access public
276 * @param int $id Si no se establece el valor de ID, se toma el valor del objeto actual
277 * @return int    false Si no se eliminó el registro, true si se ejecutó la eliminación
278 */
279 public function delete($id = null)
280 {
281     $result = false;
282
283     $id = is_null($id) ? $this-> id : $id;
284
285     if(is_array($id)) {
286         // Eliminar un conjunto de registros
287         foreach ($id as $idx) {
288             $result = $this-> db-> delete('sis_bitacora', array('id' => $idx));
289
290             if(empty($result)) {
291                 $this-> error = true;
292                 $this-> msg_error_usr = 'No se puede eliminar el registro';
293                 $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
294 > _error_number().' : '.$this-> db-> _error_message();
295                 throw new Exception(__CLASS__);
296             }
297         }
298     } else {
299         // Eliminar un solo registro

```

```

298         $result = $this-> db-> delete('sis_bitacora', array('id' => $id));
299
300         if(empty($result)) {
301             $this-> error = true;
302             $this-> msg_error_usr = 'No se puede eliminar el registro';
303             $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
> _error_number(). ': ' . $this-> db-> _error_message();
304             throw new Exception(__CLASS__);
305         }
306     }
307
308     return $result;
309 }
310
311 /**
312 * Elimina el conjunto de registros que cumplen con el o los criterios de filtrado
313 *
314 * @access public
315 * @return int false Si no se ejecutó la inserción, true si se ejecutó la inserción
316 */
317 public function deleteByFilter()
318 {
319     $result = false;
320
321     if(empty($this-> filters)) {
322         $this-> error = true;
323         $this-> msg_error_usr = 'No se encontraron filtros definidos';
324         $this-> msg_error_log = '('.__METHOD__.') => No se encontraron filtros
definidos para la eliminación';
325
326         throw new Exception(__CLASS__);
327     }
328
329     $this-> db-> where($this-> filters);
330     $result = $this-> db-> delete('sis_bitacora');
331
332     if(empty($result)) {
333         $this-> error = true;
334         $this-> msg_error_usr = 'No se pueden eliminar los registros';
335         $this-> msg_error_log = '('.__METHOD__.') => No se pueden eliminar los
registros';
336
337         throw new Exception(__CLASS__);
338     }
339
340     return $result;
341 }
342
343 /**
344 * Obtiene los datos del registro de la bitacora que tiene el ID especificado
345 *
346 * @access public
347 * @param int $id      Si no se establece el valor de ID, se toma el valor del objeto
actual
348 * @return object/boolean Devuelve el objeto con sus datos correspondientes, de lo
contrario, false Si no se encontró el registro
349 */
350 public function getById($id)
351 {
352     $result = false;
353
354     $this-> db-> select('sis_entorno.nombre AS entorno, sis_controlador.nombre AS
controlador, sis_accion.nombre AS accion, sis_bitacora.*,
sis_usuario.nombre_usuario AS usuario, sis_usuario.nombre,
sis_usuario.apellido_paterno, sis_usuario.apellido_materno');
355     $this-> db-> from('sis_bitacora');
356     $this-> db-> join('sis_usuario', 'sis_usuario.id = sis_bitacora.id_usuario');
357     $this-> db-> join('sis_controlador_x_accion', 'sis_controlador_x_accion.id =
sis_bitacora.id_controlador_accion');
358     $this-> db-> join('sis_controlador', 'sis_controlador.id =
sis_controlador_x_accion.id_controlador');
359     $this-> db-> join('sis_accion', 'sis_accion.id =
sis_controlador_x_accion.id_accion');
360     $this-> db-> join('sis_entorno', 'sis_entorno.id = sis_controlador.id_entorno');
361     $this-> db-> where('sis_bitacora.id', $id);
362
363     $query = $this-> db-> get();
364     $result = $query-> row();
365
366     if($this-> db-> _error_number()) {
367         $this-> error = true;
368         $this-> msg_error_usr = 'No se encontraron registros en la búsqueda';

```

```

368     $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
> _error_number().': '.$this-> db-> _error_message();
369     throw new Exception(__CLASS__);
370 }
371
372 if(!empty($result)) {
373     $this-> load-> model('siigs/ControladorAccion_model');
374
375     $this-> id = $id;
376     $this-> id_usuario = $result-> id_usuario;
377     $this-> fecha_hora = $result-> fecha_hora;
378     $this-> parametros = $result-> parametros;
379     $this-> id_controlador_accion = $result-> id_controlador_accion;
380
381     $controlador_accion = $this-> ControladorAccion_model-> getById($result-
> id_controlador_accion);
382     $this-> id_controlador = $controlador_accion-> id_controlador;
383     $this-> id_accion = $controlador_accion-> id_accion;
384 }
385
386 return $result;
387 }
388
389 /**
390 * Agrega una nueva regla de filtrado al arreglo de filtros
391 *
392 * @access public
393 * @param string $columna Puede ser cualquier campo del objeto (id, id_usuario,
394 * fecha_hora, parametros, id_controlador_accion)
395 * @param string $condicion Establece la condicion a evaluar, entre los valores permitidos
396 * estan: =, !=, >=, <=, like
397 * @param string $valor Valor contra el cual se realizará la evaluación del campo
398 * @return void/boolean Devuelve falso en caso de no poder establecer el filtro
399 */
400 public function addFilter($columna, $condicion, $valor)
401 {
402     $columnasPermitidas = array(
403         'id',
404         'id_usuario',
405         'fecha_hora',
406         'parametros',
407         'id_entorno',
408         'id_controlador',
409         'id_accion'
410     );
411
412     $condicionesPermitidas = array('=', '>', '<', '!=', '>=' , '<=' , 'like');
413
414     if(!in_array($columna, $columnasPermitidas)) {
415         $this-> error = true;
416         $this-> msg_error_usr = 'ERROR: Columna no permitida en el filtro ('.$columna.')';
417         $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> msg_error_usr;
418
419         throw new Exception(__CLASS__);
420     }
421
422     if(!in_array($condicion, $condicionesPermitidas)) {
423         $this-> error = true;
424         $this-> msg_error_usr = 'ERROR: Condición no permitida en el filtro
425 ('.$condicion.')';
426         $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> msg_error_usr;
427
428         throw new Exception(__CLASS__);
429     }
430
431     if(empty($valor)) {
432         $this-> error = true;
433         $this-> msg_error_usr = 'ERROR: Debe definir un valor para el filtro';
434         $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> msg_error_usr;
435
436         throw new Exception(__CLASS__);
437     }
438
439     // Ejemplo de filtros permitidos por where de active records
440     // $filtros = array(
441     //     'name !=' => $name,
442     //     'id <'    => $id,
443     //     'date >'  => $date
444     // );
445     $this-> filters[$columna.'.'.$condicion] = $valor;

```

```

443     }
444
445     /**
446      * Elimina todos los filtros registrados
447      *
448      * @access public
449      * @return void
450      */
451     public function resetFilter()
452     {
453         $this-> filters = array();
454     }
455
456     /**
457      * Obtiene todos los registros de la tabla Bitacora
458      * en caso de existir filtros, estos son aplicados a la consulta
459      *
460      * @access public
461      * @param int $offset      Establece el desplazamiento del primer registro a devolver,
462      *                         si se define sólo el valor de offset
463      *                         el valor especifica el número de registros a retornar desde el
464      * comienzo del conjunto de resultados.
465      * @param int $row_count   Establece la cantidad de registros a devolver
466      * @return array object    Devuelve un arreglo de objetos obtenidos de la base de datos
467     */
468     public function getAll($offset = null, $row_count = null)
469     {
470         $result = 0;
471
472         $this-> db-> select('sis_entorno.nombre AS entorno, sis_controlador.nombre AS
controlador, sis_accion.nombre AS accion, sis_bitacora.*,
sis_usuario.nombre_usuario AS usuario, sis_usuario.nombre,
sis_usuario.apellido_paterno, sis_usuario.apellido_materno');
473         $this-> db-> from('sis_bitacora');
474         $this-> db-> join('sis_usuario', 'sis_usuario.id = sis_bitacora.id_usuario');
475         $this-> db-> join('sis_controlador_x_accion', 'sis_controlador_x_accion.id =
sis_bitacora.id_controlador_accion');
476         $this-> db-> join('sis_controlador', 'sis_controlador.id =
sis_controlador_x_accion.id_controlador');
477         $this-> db-> join('sis_accion', 'sis_accion.id =
sis_controlador_x_accion.id_accion');
478         $this-> db-> join('sis_entorno', 'sis_entorno.id = sis_controlador.id_entorno');
479         $this-> db-> order_by('fecha_hora', 'desc');
480         $this-> db-> order_by('id_entorno', 'desc');
481         $this-> db-> order_by('id_controlador', 'desc');
482         $this-> db-> order_by('id_accion', 'desc');
483
484         if( !empty($this-> filters) )
485             $this-> db-> where($this-> filters);
486
487         if(!empty($offset) && !empty($row_count))
488             $this-> db-> limit($offset, $row_count);
489         else if (!empty($offset))
490             $this-> db-> limit($offset);
491
492         $query = $this-> db-> get();
493
494         if($this-> db-> _error_number()) {
495             $this-> error = true;
496             $this-> msg_error_usr = 'No se encontraron registros en la busqueda';
497             $this-> msg_error_log = '('.__METHOD__.') => '. $this-> db-
> _error_number().': '.$this-> db-> _error_message();
498             throw new Exception(__CLASS__);
499         } else {
500             $result = $query-> result();
501         }
502
503         return $result;
504     }
505
506
507     /**
508      * Obtiene el numero total de registros en la tabla Bitacora
509      * en caso de existir filtros, estos son aplicados a la consulta
510      *
511      * @access public
512      * @return int
513      */
514     public function getNumRows()
515     {

```

```

516     $result = 0;
517
518     $this-> db-> select('sis_entorno.nombre AS entorno, sis_controlador.nombre AS
controlador, sis_accion.nombre AS accion, sis_bitacora.*,
sis_usuario.nombre_usuario AS usuario, sis_usuario.nombre,
sis_usuario.apellido_paterno, sis_usuario.apellido_materno');
519
520     $this-> db-> from('sis_bitacora');
521     $this-> db-> join('sis_usuario', 'sis_usuario.id = sis_bitacora.id_usuario');
522     $this-> db-> join('sis_controlador_x_accion', 'sis_controlador_x_accion.id =
sis_bitacora.id_controlador_accion');
523     $this-> db-> join('sis_controlador', 'sis_controlador.id =
sis_controlador_x_accion.id_controlador');
524     $this-> db-> join('sis_accion', 'sis_accion.id =
sis_controlador_x_accion.id_accion');
525     $this-> db-> join('sis_entorno', 'sis_entorno.id = sis_controlador.id_entorno');
526
527     if(!empty($this-> filters))
528         $this-> db-> where($this-> filters);
529
530     $result = $this-> db-> count_all_results();
531
532     if($this-> db-> _error_number()) {
533         $this-> error = true;
534         $this-> msg_error_usr = 'No se encontraron registros en la busqueda';
535         $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
> _error_number().': '.$this-> db-> _error_message();
536         throw new Exception(__CLASS__);
537     }
538
539     return $result;
540 }
541
542 }
543 ?>

```

# Archivo fuente para catalogocsv\_model.php

*La documentación para este archivo está disponible en [catalogocsv\\_model.php](#)*

```
1  <?php
2
3  /**
4   * Modelo CatalogoCsv
5   *
6   * @package      SIIGS
7   * @subpackage   Modelo
8   * @author       Geovanni
9   * @created      2013-10-07
10  */
11 class CatalogoCsv_model extends CI_Model {
12
13     /**
14      * @access private
15      * @var    int
16      */
17     private $id;
18
19     /**
20      * @access private
21      * @var    string
22      */
23     private $nombre;
24
25     /**
26      * @access private
27      * @var    string
28      */
29     private $campos;
30
31     /**
32      * @access private
33      * @var    string
34      */
35     private $llave;
36
37     /**
38      * @access private
39      * @var    int
40      */
41     private $offset;
42
43     /**
44      * @access private
45      * @var    int
46      */
47     private $rows;
48
49     /**
50      * @access private
51      * @var    string
52      */
53     private $msg_error_log;
54
55     /**
56      * @access private
57      * @var    string
58      */
59     private $msg_error_usr;
60
61     /*****
62     /*Getters and setters block*/
63     *****/
64     public function getId() {
```

```

65     return $this-> id;
66 }
67
68 public function setId($value) {
69     $this-> id = $value;
70 }
71
72 public function getNombre() {
73     return $this-> nombre;
74 }
75 public function setNombre($value) {
76     $this-> nombre = $value;
77 }
78
79 public function getCampos() {
80     return $this-> campos;
81 }
82 public function setCampos($value) {
83     $this-> campos = $value;
84 }
85
86 public function getLlave() {
87     return $this-> llave;
88 }
89 public function setLlave($value) {
90     $this-> llave = $value;
91 }
92
93 public function setOffset($value) {
94     $this-> offset = $value;
95 }
96 public function setRows($value) {
97     $this-> rows = $value;
98 }
99
100 /*****
101 /*Getters and setters block END*/
102 /*****
103 /**
104 * Devuelve los mensajes de error en caso de ocurrir alguna excepción
105 * 'usr' devuelve el mensaje para la vista de usuario
106 * 'log' devuelve el mensaje para el log de errores
107 *
108 * @access public
109 * @return string|boolean
110 * @param string $value, default 'usr' (Tipo mensaje)
111 */
112 public function getMsgError($value = 'usr')
113 {
114     if (!empty($this-> msg_error_usr))
115     {
116         if ($value == 'usr')
117             return $this-> msg_error_usr;
118         else if ($value == 'log')
119             return $this-> msg_error_log;
120     }
121     else
122     {
123         return false;
124     }
125 }
126
127 public function __construct()
128 {
129     $this-> load-> database();
130     if (!$this-> db-> conn_id)
131     {
132         throw new Exception("No se pudo conectar a la base de datos");
133     }
134 }
135
136 /**
137 *Devuelve el numero de registros
138 *
139 *@access public
140 *@return int
141 *@param string $nombre Nombre del catalogo
142 * @throws Exception En caso de algun error al consultar la base de datos
143 */

```

```

145     public function getNumRows($nombre)
146     {
147         $query = $this-> db-> query('select count(*) as num from '.$nombre);
148         if (!$query)
149         {
150             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
151             $this-> msg_error_usr = "Ocurrió un error al obtener los datos del catalogo
" . $nombre;
152             //throw new Exception(__CLASS__);
153         }
154         else
155             return $query-> row()-> num;
156     }
157
158 /**
159 * Devuelve una lista con los catalogos existentes en la DB
160 *
161 * @access public
162 * @return ArrayObject
163 * @throws Exception En caso de algun error al consultar la base de datos
164 */
165 public function getAll()
166 {
167     $catalogos = $this-> db-> query('SELECT table_name as nombre, table_comment as
comentario FROM INFORMATION_SCHEMA.TABLES WHERE table_schema = "' . $this-> db-
> database.'" AND table_name in (' . CATALOGOSCSV.')');
168     if (!$catalogos)
169     {
170         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
171         $this-> msg_error_usr = "Ocurrió un error al obtener los datos de los
catálogos" ;
172         throw new Exception(__CLASS__);
173     }
174     else
175         return $catalogos-> result();
176 }
177
178 /**
179 * Devuelve los datos de un catalogo pasado como parametro
180 *
181 * @access public
182 * @param string $nombrecat Nombre del catalogo
183 * @return ArrayObject
184 * @throws Exception En caso de algun error al consultar la base de datos
185 */
186 public function getAllData($nombrecat)
187 {
188     $string = 'select * from '.$nombrecat;
189
190     if (!empty($this-> offset) || $this-> offset == 0) && !empty( $this-
> rows))
191         $string .= ' limit ' . $this-> offset . ',' . $this-> rows;
192
193     $catalogos = $this-> db-> query($string);
194
195     if (!$catalogos)
196     {
197         $this-> msg_error_log = "(" . __METHOD__ . ") => "
. $this-> db-> _error_number().': '.$this-> db-> _error_message();
198         $this-> msg_error_usr = "Ocurrió un error al obtener los datos de los
catálogos" ;
199         throw new Exception(__CLASS__);
200     }
201     else
202         return $catalogos-> result();
203 }
204
205 /**
206 * Devuelve la informacion de un catalogo por su nombre
207 *
208 * @access public
209 * @return Object
210 * @param string $nombre (Nombre del catalogo)
211 * @throws Exception En caso de algun error al consultar la base de datos
212 */
213 public function getByName($nombre)
214 {
215     $result = array(

```

```

216         'nombre' => $nombre
217     );
218     $tblcampos = $this-> db-> query('describe '.$nombre);
219     if (!$tblcampos)
220     {
221         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
222 > _error_number().': '.$this-> db-> _error_message();
223         $this-> msg_error_usr = "Ocurrió un error al obtener los datos de los
224 catálogos";
225     ;
226     throw new Exception(__CLASS__);
227     }
228     else
229     {
230         $campos = '';
231         $llave = '';
232         foreach ($tblcampos-> result() as $campo)
233         {
234             if ($campo-> Key == '')
235                 $campos .= $campo-> Field.'|'.$campo-> Type.'|'.$campo-> Null.'|NO|';
236             else
237                 $llave .= $campo-> Field.'|'.$campo-> Type.'|'.$campo-> Null.'|YES|';
238         }
239         $result['campos'] = substr($campos, 0,count($campos)-3);
240         $result['llave'] = substr($llave, 0,count($llave)-3);
241         $result['comentario'] = $this-> db-> query('SELECT table_comment as
242 comentario FROM INFORMATION_SCHEMA.TABLES WHERE table_schema = "' . $this-> db-
243 > database.'" AND table_name = "' . $nombre.'" )-> result()[0]-> comentario;
244     }
245     /**
246     * Accion para activar o desactivar registros de catalogos como el de EDA, IRA y
247 Consultas
248     * @param int $id el id del registro en el catalogo
249     * @param string $catalogo nombre del catalogo donde se realizara la operacion
250     * @param boolean $valor agregar o eliminar el registro del catalogo
251     * @return boolean como el resultado de la operación
252     * @throws Exception Si ocurre algun error al consultar y modificar la base de datos
253     */
254     public function activaEnCatalogo($id,$catalogo,$valor)
255     {
256
257         $consulta = "update " . $catalogo. " set activo = " . (($valor ==
258 true) ? 1 : 0). " where id = '" . $id. "' ";
259         $datos = $this-> db-> query($consulta);
260
261         if (!$datos)
262         {
263             $this-> msg_error_log = "(" . __METHOD__ . ") => "
264 . $this-> db-> _error_number().': '.$this-> db-> _error_message();
265             $this-> msg_error_usr = "Ocurrió un error al activar el elemento en
266 el catalogo" . $catalogo;
267             throw new Exception(__CLASS__);
268         }
269         else
270         {
271             $this-> db-> query("update cns_tabla_catalogo set fecha_actualizacion
272 = NOW() where descripcion=' " . $catalogo. "' ");
273             return true;
274         }
275     }
276     /**
277     * Revisa en la base de datos por registros duplicados en los campos pasados por parametro
278     *
279     * @access public
280     * @param string $campo (varios campos delimitados por | )
281     * @return boolean (Si no hubo errores al eliminar)
282     * @throws Exception En caso de algun error al consultar la base de datos
283     */
284     public function checkPk($campo)
285     {
286         $campo = urldecode($campo);
287         $campos = explode('|', $campo);
288         $querycampos = " " ;

```

```

287
288     foreach($campos as $c)
289         $querycampos .= $c.', ';
290
291     $consulta = "select " . $querycampos . " count(*) from tmp_catalogo group by
" . substr($querycampos, 0, strlen($querycampos)-2) . " having count(*) > 1" ;
292     $query = $this-> db-> query($consulta);
293
294     if (!$query)
295     {
296         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number() . ': ' . $this-> db-> _error_message();
297         $this-> msg_error_usr = "Ocurrió un error al revisar la llave primaria de los
catalogos";
298         ;
299         throw new Exception(__CLASS__);
300     }
301     else
302         return $query-> result();
303     }
304 }
```

# Archivo fuente para catalogo\_model.php

*La documentación para este archivo está disponible en [catalogo\\_model.php](#)*

```
1  <?php
2
3  /**
4   * Modelo Catalogo
5   *
6   * @package    SIIGS
7   * @subpackage Modelo
8   * @author     Geovanni
9   * @created    2013-10-07
10  */
11 class Catalogo_model extends CI_Model {
12
13     /**
14      * @access private
15      * @var int
16      */
17     private $id;
18
19     /**
20      * @access private
21      * @var string
22      */
23     private $nombre;
24
25     /**
26      * @access private
27      * @var string
28      */
29     private $campos;
30
31     /**
32      * @access private
33      * @var string
34      */
35     private $llave;
36
37     /**
38      * @access private
39      * @var int
40      */
41     private $offset;
42
43     /**
44      * @access private
45      * @var int
46      */
47     private $rows;
48
49     /**
50      * @access private
51      * @var string
52      */
53     private $msg_error_log;
54
55     /**
56      * @access private
57      * @var string
58      */
59     private $msg_error_usr;
60
61     /*****
62     *Getters and setters block*/
63     *****/
64     public function getId() {
65         return $this-> id;
66     }
67 }
```

```

68     public function setId($value) {
69         $this-> id = $value;
70     }
71
72     public function getNombre() {
73         return $this-> nombre;
74     }
75     public function setNombre($value) {
76         $this-> nombre = $value;
77     }
78
79     public function getCampos() {
80         return $this-> campos;
81     }
82     public function setCampos($value) {
83         $this-> campos = $value;
84     }
85
86     public function getLlave() {
87         return $this-> llave;
88     }
89     public function setLlave($value) {
90         $this-> llave = $value;
91     }
92
93
94     public function setOffset($value) {
95         $this-> offset = $value;
96     }
97     public function setRows($value) {
98         $this-> rows = $value;
99     }
100    /*****
101    /*Getters and setters block END*/
102    *****/
103
104    /**
105     * Devuelve los mensajes de error en caso de ocurrir alguna excepción
106     * 'usr' devuelve el mensaje para la vista de usuario
107     * 'log' devuelve el mensaje para el log de errores
108     *
109     * @access public
110     * @return string/boolean
111     * @param string $value, default 'usr' (Tipo mensaje)
112     */
113     public function getMsgError($value = 'usr')
114     {
115         if (!empty($this-> msg_error_usr))
116         {
117             if ($value == 'usr')
118                 return $this-> msg_error_usr;
119             else if ($value == 'log')
120                 return $this-> msg_error_log;
121         }
122         else
123         {
124             return false;
125         }
126     }
127
128     public function __construct()
129     {
130         $this-> load-> database();
131         if (!$this-> db-> conn_id)
132         {
133             throw new Exception("No se pudo conectar a la base de datos");
134         }
135     }
136
137    /**
138     *Devuelve una lista con los catalogos existentes en la DB
139     *
140     *@access public
141     *@return ArrayObject
142     * @throws Exception En caso de algun error al consultar la base de datos
143     */
144     public function getAll()
145     {
146         $catalogos = $this-> db-> query('SELECT table_name as nombre, table_comment as
comentario FROM INFORMATION_SCHEMA.TABLES WHERE table_schema = "' . $this-> db-

```

```

>   database.'" AND table_name LIKE "cat_%" and table_name <>
"cat_georeferencia" and table_name <> "cat_poblacion" and table_name
<> "cat_ageb";
147           if (!$catalogos)
148           {
149               $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
150               _error_number().': '.$this-> db-> _error_message();
151               $this-> msg_error_usr = "Ocurrió un error al obtener los datos de los
catálogos";
152               throw new Exception(__CLASS__);
153           }
154           else
155               return $catalogos-> result();
156       }
157 /**
158 *Devuelve el numero de registros
159 *
160 *@access public
161 *@return int
162 *@param string $nombre Nombre del catalogo
163 * @throws Exception En caso de algun error al consultar la base de datos
164 */
165 public function getNumRows($nombre)
166 {
167     $query = $this-> db-> query('select count(*) as num from ' . $nombre);
168     if (!$query)
169     {
170         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
171         _error_number().': '.$this-> db-> _error_message();
172         $this-> msg_error_usr = "Ocurrió un error al obtener los datos del catalogo
" . $nombre;
173         //throw new Exception(__CLASS__);
174     }
175     else
176         return $query-> row()-> num;
177 }
178 /**
179 *Devuelve los datos de un catalogo pasado como parametro
180 *
181 *@access public
182 * @param string $nombrecat Nombre del catalogo
183 *@return ArrayObject
184 * @throws Exception En caso de algun error al consultar la base de datos
185 */
186 public function getAllData($nombrecat)
187 {
188     $string = 'select * from ' . $nombrecat;
189
190     if (!empty($this-> offset) || $this-> offset == 0) && !empty(
> rows))
191         $string .= ' limit ' . $this-> offset . ',' . $this-> rows;
192
193     $catalogos = $this-> db-> query($string);
194
195     if (!$catalogos)
196     {
197         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
198         _error_number().': '.$this-> db-> _error_message();
199         $this-> msg_error_usr = "Ocurrió un error al obtener los datos de los
catálogos";
200         throw new Exception(__CLASS__);
201     }
202     else
203         return $catalogos-> result();
204 }
205 /**
206 *Devuelve la informacion de un catalogo por su nombre
207 *
208 *@access public
209 *@return Object
210 *@param string $nombre (Nombre del catalogo)
211 * @throws Exception En caso de algun error al consultar la base de datos
212 */
213 public function getByName($nombre)
214 {
215     $result = array(
216         'nombre' => $nombre

```

```

217     );
218     $tblcampos = $this-> db-> query('describe '.$nombre);
219     if (!$tblcampos)
220     {
221         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
222 > _error_number().": ".$this-> db-> _error_message();
223         $this-> msg_error_usr = "Ocurrió un error al obtener los datos de los
224         catálogos";
225     }
226     else
227     {
228         $campos = '';
229         $llave = '';
230         foreach ($tblcampos-> result() as $campo)
231         {
232             if ($campo-> Key == '')
233                 $campos .= $campo-> Field.'|'.$campo-> Type.'|'.$campo-> Null.'|NO|';
234             else
235                 $llave .= $campo-> Field.'|'.$campo-> Type.'|'.$campo-> Null.'|YES|';
236         }
237         $result['campos'] = substr($campos, 0,count($campos)-3);
238         $result['llave'] = substr($llave, 0,count($llave)-3);
239     }
240
241     $result['comentario'] = $this-> db-> query('SELECT table_comment as
242 comentario FROM INFORMATION_SCHEMA.TABLES WHERE table_schema = "' . $this-> db-
243 > database.'" AND table_name = "' . $nombre . '"')-> result()[0]-> comentario;
244
245     return (object)$result;
246 }
247 /**
248 *Revisa en la base de datos por registros duplicados en los campos pasados por parametro
249 *
250 * @access public
251 * @param string $campo (varios campos delimitados por / )
252 * @return boolean (Si no hubo errores al eliminar)
253 * @throws Exception En caso de algun error al consultar la base de datos
254 */
255 public function checkPk($campo)
256 {
257     $campo = urldecode($campo);
258     $campos = explode('/', $campo);
259     $querycampos = "";
260
261     foreach($campos as $c)
262         $querycampos .= $c. ', ';
263
264     $consulta = "select " . $querycampos . " count(*) from tmp_catalogo group by
265     ".substr($querycampos, 0,strlen($querycampos)-2). " having count(*) > 1";
266     $query = $this-> db-> query($consulta);
267
268     //var_dump($consulta);
269
270     if (!$query)
271     {
272         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
273 > _error_number().": ".$this-> db-> _error_message();
274         $this-> msg_error_usr = "Ocurrió un error al eliminar el catálogo";
275         throw new Exception(__CLASS__);
276     }
277     else
278         return $query-> result();
279 }
280 /**
281 *Revisa en la base de datos por registros que no coincidan con el tipo de dato
282 * pasado como parametro en el campo indicado
283 *
284 * @access public
285 * @param string $campo (varios campos delimitados por / )
286 * @return boolean (Si no hubo errores al eliminar)
287 * @throws Exception En caso de algun error al consultar la base de datos
288 */
289 public function checkTypeData($campo,$type)
290 {
291     $consulta = '';
292     switch ($type)

```

```

291         {
292             case 'int':
293                 $consulta = "SELECT * FROM tmp_catalogo WHERE not " . $campo . "
294             REGEXP '^-[0-9]+$'";
295             case 'decimal':
296                 $consulta = "SELECT * FROM tmp_catalogo WHERE not " . $campo . "
297             REGEXP '^-[0-9]+\([0-9]\)?$'";
298             if ($consulta=='')
299                 echo 'true';
300             else
301                 {
302                     $query = $this-> db-> query($consulta);
303                     if (!$query)
304                         {
305                             $this-> msg_error_log = "(" . __METHOD__ . ") =>
306                         . $this-> db-> _error_number().':'. $this-> db-> _error_message();
307                         $this-> msg_error_usr = "Ocurrió un error al obtener tipos de
308                         datos de la columna en tmp_catalogo";
309                         //throw new Exception(__CLASS__);
310                         echo 'false';
311                     }
312                     else
313                         {
314                             if ($query-> num_rows() > 0)
315                                 echo 'false';
316                             else
317                                 echo 'true';
318                         }
319             }
320             /**
321             *Inserta en la base datos el catálogo y obtiene los datos
322             *de la tabla temporal
323             */
324             *@access public
325             *@param string $create (la consulta para crear el catalogo)
326             *@param string $select (la consulta para extraer datos de la tabla tmp_catalogo)
327             *@return boolean (si la insercion es correcta)
328             */
329             public function insert($create,$select)
330             {
331                 $query = $this-> db-> query($create);
332                 if (!$query)
333                     {
334                         $this-> msg_error_log = "(" . __METHOD__ . ") =>
335                         . $this-> db-> _error_number().':'. $this-> db-> _error_message();
336                         $this-> msg_error_usr = "Ocurrió un error al insertar el catalogo";
337                         throw new Exception(__CLASS__);
338                     }
339                 else
340                     {
341                         $query = $this-> db-> query($select);
342                         if (!$query)
343                             {
344                                 $this-> msg_error_log = "(" . __METHOD__ . ") =>
345                                 . $this-> db-> _error_number().':'. $this-> db-> _error_message();
346                                 $this-> msg_error_usr = "Ocurrió un error al insertar el catálogo";
347                                 throw new Exception(__CLASS__);
348                             }
349                         else
350                             return true;
351                     }
352             }
353         }
354     }
355     /**
356     *Cambia el comentario de la tabla indicada
357     */
358     *@access public
359     *@param string $nombre
360     *@param string $comentario
361     *@return boolean (si la insercion es correcta)
362     */
363     public function updateComentario($nombre, $comentario)

```

```

365     {
366
367         $query = $this-> db-> query('alter table '.$nombre.' comment
368 " '
369             . $comentario. ' ');
370
371         if (!$query)
372         {
373             $this-> msg_error_log = " ( " . __METHOD__ . " ) => "
374             . $this-> db-> _error_number(). ': ' . $this-> db-> _error_message();
375             $this-> msg_error_usr = "Ocurrió un error al modificar el comentario
376 del catálogo" ;
377             throw new Exception(__CLASS__);
378         }
379         else
380             return true;
381     }
382
383     /**
384      * Elimina el registro actual de la base de datos
385      *
386      * @access public
387      * @return boolean (Si no hubo errores al eliminar)
388      * @throws Exception En caso de algun error al consultar la base de datos
389      */
390     public function delete()
391     {
392
393         $query = $this-> db-> query('drop table '.$this-> nombre);
394
395         if (!$query)
396         {
397             $this-> msg_error_log = " ( " . __METHOD__ . " ) => "
398             . $this-> db-> _error_number(). ': ' . $this-> db-> _error_message();
399             $this-> msg_error_usr = "Ocurrió un error al eliminar el catálogo" ;
400             throw new Exception(__CLASS__);
401         }
402     }

```

# Archivo fuente para catalogo\_x\_raiz\_model.php

*La documentación para este archivo está disponible en [catalogo\\_x\\_raiz\\_model.php](#)*

```
1  <?php
2
3  /**
4   * Modelo Raiz_x_Catalogo
5   *
6   * @package      SIIGS
7   * @subpackage   Modelo
8   * @author       Geovanni
9   * @created      2013-10-16
10  */
11 class Catalogo_x_raiz_model extends CI_Model {
12
13     /**
14      * @access private
15      * @var    int
16      */
17     private $id;
18
19     /**
20      * @access private
21      * @var    int
22      */
23     private $id_raiz;
24
25     /**
26      * @access private
27      * @var    int
28      */
29     private $grado;
30
31     /**
32      * @access private
33      * @var    string
34      */
35     private $tabla_catalogo;
36
37     /**
38      * @access private
39      * @var    string
40      */
41     private $columna_llave;
42
43     /**
44      * @access private
45      * @var    string
46      */
47     private $columna_descripcion;
48
49     /**
50      * @access private
51      * @var    Array
52      */
53     private $relacionpadre;
54
55     /**
56      * @access private
57      * @var    Array
58      */
59     private $relacionhijo;
60
61     /**
62      * @access private
63      * @var    string
64      */
```

```

65     private $msg_error_log;
66
67     /**
68      * @access private
69      * @var string
70      */
71     private $msg_error_usr;
72
73     /*****
74     * Getters and setters block*/
75     /*****
76     public function getId() {
77         return $this-> id;
78     }
79
80     public function setId($value) {
81         $this-> id = $value;
82     }
83
84     public function getIdRaiz() {
85         return $this-> id_raiz;
86     }
87
88     public function setIdRaiz($value) {
89         $this-> id_raiz = $value;
90     }
91
92     public function getGrado() {
93         return $this-> grado;
94     }
95     public function setGrado($value) {
96         $this-> grado = $value;
97     }
98
99     public function getTablaCatalogo() {
100        return $this-> tabla_catalogo;
101    }
102    public function setTablaCatalogo($value) {
103        $this-> tabla_catalogo = $value;
104    }
105
106    public function getColumnaLlave() {
107        return $this-> columna_llave;
108    }
109    public function setColumnaLlave($value) {
110        $this-> columna_llave = $value;
111    }
112
113    public function getColumnaDescripcion() {
114        return $this-> columna_descripcion;
115    }
116    public function setColumnaDescripcion($value) {
117        $this-> columna_descripcion = $value;
118    }
119
120    public function getRelacionPadre() {
121        return $this-> relacionpadre;
122    }
123    public function setRelacionPadre($value) {
124        $this-> relacionpadre = $value;
125    }
126
127    public function getRelacionHijo() {
128        return $this-> relacionhijo;
129    }
130    public function setRelacionHijo($value) {
131        $this-> relacionhijo = $value;
132    }
133    /*****
134    * Getters and setters block END*/
135    /*****
136
137    /**
138     * Devuelve los mensajes de error en caso de ocurrir alguna excepción
139     * 'usr' devuelve el mensaje para la vista de usuario
140     * 'log' devuelve el mensaje para el log de errores
141     *
142     * @access public
143     * @return string|boolean
144     * @param string $value, default 'usr' (Tipo mensaje)

```

```

145     */
146     public function getMsgError($value = 'usr')
147     {
148         if (!empty($this->msg_error_usr))
149         {
150             if ($value == 'usr')
151                 return $this->msg_error_usr;
152             else if ($value == 'log')
153                 return $this->msg_error_log;
154         }
155         else
156         {
157             return false;
158         }
159     }
160
161     public function __construct()
162     {
163         $this->load->database();
164         if (!$this->db->conn_id)
165         {
166             throw new Exception("No se pudo conectar a la base de datos");
167         }
168     }
169
170 /**
171 * Devuelve todos los registros de la tabla raiz_x_catalogo de una raiz determinada
172 *
173 * @access public
174 * @param int $id
175 * @return ArrayObject
176 * @throws Exception En caso de algun error al consultar la base de datos
177 */
178 public function getByArbol($id)
179 {
180     $catalogos = $this->db->query('SELECT * from asu_raiz_x_catalogo where id_raiz_arbol='.$id.' order by grado_segmentacion');
181
182     if (!$catalogos)
183     {
184         $this->msg_error_log = "(" . __METHOD__ . ") => " . $this-
185 > db->_error_number().': '.$this->db->_error_message();
186         $this->msg_error_usr = "Ocurrió un error al obtener los datos de los
187 catálogos";
188         throw new Exception(__CLASS__);
189     }
190     else
191         return $catalogos->result();
192 }
193
194 /**
195 * Devuelve el nivel siguiente para la tabla raiz_x_catalogo de un arbol determinado
196 *
197 * @access public
198 * @param int $id
199 * @return ArrayObject
200 * @throws Exception En caso de algun error al consultar la base de datos
201 */
202 public function getNivel($id)
203 {
204     $nivel = $this->db->query('SELECT ifnull(max(grado_segmentacion),0) +1 as nivel
205 from asu_raiz_x_catalogo where id_raiz_arbol='.$id);
206
207     if (!$nivel)
208     {
209         $this->msg_error_log = "(" . __METHOD__ . ") => " . $this-
210 > db->_error_number().': '.$this->db->_error_message();
211         $this->msg_error_usr = "Ocurrió un error al obtener los datos de los
212 catálogos";
213         throw new Exception(__CLASS__);
214     }
215     else
216         return $nivel->row();
217 }
218
219 /**
220 * Devuelve el catalogo padre de un elemento raiz_x_catalogo
221 *
222 * @access public
223 * @param int $idarbol

```

```

219         *@param    int $nivel
220         *@return   ArrayObject
221         * @throws Exception En caso de algun error al consultar la base de datos
222         */
223     public function getByNivel($id_arbol,$nivel)
224     {
225         $nivel = $this-> db-> query('SELECT tabla_catalogo as nombre,nombre_columna_llave
226         as llave from asu_raiz_x_catalogo where grado_segmentacion='.$nivel.' and
227         id_raiz_arbol="'.$id_arbol");
228         if (!$nivel)
229         {
230             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
231             > db-> _error_number().': '.$this-> db-> _error_message();
232             $this-> msg_error_usr = "Ocurrió un error al obtener los datos de los
233             catálogos";
234             throw new Exception(__CLASS__);
235         }
236         else
237             return $nivel-> row();
238     }
239
240     /**
241      *Devuelve la información de un catalogo x accion por su ID
242      *
243      *@access  public
244      *@return   Object
245      *@param    int $id ID (Llave primaria)
246      * @throws Exception En caso de algun error al consultar la base de datos
247      */
248     public function getById($id)
249     {
250         $query = $this-> db-> query('select * from asu_raiz_x_catalogo where id='.$id);
251         if (!$query)
252         {
253             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
254             > db-> _error_number().': '.$this-> db-> _error_message();
255             $this-> msg_error_usr = "Ocurrió un error al obtener los datos de los
256             catálogos";
257             throw new Exception(__CLASS__);
258         }
259         else
260             return $query-> row();
261     }
262
263     /**
264      *Devuelve las relaciones de una raiz x catalogo
265      *
266      *@access  public
267      *@return   Object
268      *@param    int $id ID (Llave primaria)
269      * @throws Exception En caso de algun error al consultar la base de datos
270      */
271     public function getRelations($id)
272     {
273         $query = $this-> db-> query('select * from asu_relacion_catalogo where
274         id_raiz_x_catalogo='.$id);
275         if (!$query)
276         {
277             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
278             > db-> _error_number().': '.$this-> db-> _error_message();
279             $this-> msg_error_usr = "Ocurrió un error al obtener las relaciones entre los
280             catálogos";
281             throw new Exception(__CLASS__);
282         }
283         else
284             return $query-> result();
285     }
286
287     /**
288      *Revisa inconsistencias en los datos de un catalogo x raiz con
289      *respecto a su catalogo padre
290      *
291      *@access  public
292      *@return   Object
293      *@param    int $id ID (Llave primaria)
294      * @throws Exception En caso de algun error al consultar la base de datos
295      */

```

```

290     public function check($id)
291     {
292
293         $cathijo = $this-> getById($id);
294         if (!$cathijo || count($cathijo) == 0)
295         {
296             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
297             > db-> _error_number().': '.$this-> db-> _error_message();
298             $this-> msg_error_usr = "Ocurrió un error al obtener los datos de los
299             catálogos x raiz" ;
300             throw new Exception(__CLASS__);
301         }
302         else
303         {
304             $catpadre = $this-> db-> query('select * from asu_raiz_x_catalogo where
305             id_raiz_arbol='.$cathijo-> id_raiz_arbol.' and grado_segmentacion='.$cathijo-
306             > grado_segmentacion-1));
307             if (!$catpadre || $catpadre-> num_rows() == 0)
308             {
309                 $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
310             > db-> _error_number().': '.$this-> db-> _error_message();
311             $this-> msg_error_usr = "Ocurrió un error al obtener los datos de los
312             catálogos x raiz" ;
313             throw new Exception(__CLASS__);
314         }
315         else
316         {
317             $relaciones = $this-> getRelations($id);
318             if (!$relaciones || count($relaciones) == 0)
319             {
320                 $consulta = "select * from " . $cathijo-> tabla_catalogo.' where
321                 $consulta .= ' and '.$cathijo-> tabla_catalogo.'. '$cathijo-
322                 > nombre_columna_llave.' not in (select '.$cathijo-> tabla_catalogo.'. '$cathijo-
323                 > nombre_columna_llave.' from '.$cathijo-> tabla_catalogo.' join '.$catpadre-> row()-+
324                 > tabla_catalogo.' on l=1 ';
325                 foreach ($relaciones as $relacion)
326                 {
327                     // $consulta .= ' and '.$relacion-> columna_hijo.' not in (select
328                     // distinct '.$relacion-> columna_padre.' from '.$catpadre-> row()-> tabla_catalogo.');
329                     $consulta .= ' and '.$cathijo-> tabla_catalogo.'. '$relacion-
330                 > columna_hijo.'='.$catpadre-> row()-> tabla_catalogo.'. '$relacion-> columna_padre.' ';
331                 }
332                 $consulta .= " )" ;
333                 // var_dump($consulta);
334                 $errores = $this-> db-> query($consulta);
335                 if (!$errores)
336                 {
337                     $this-> msg_error_log = "(" . __METHOD__ . ") =>
338                     > _error_number().': '.$this-> db-> _error_message();
339                     $this-> msg_error_usr = "Ocurrió un error al obtener los datos de
340                     los catálogos x raiz" ;
341                     throw new Exception(__CLASS__);
342                 }
343             }
344             /**
345             * Inserta en la base datos la información del objeto actual
346             */
347             *@access public
348             *@return int (Id de la inserción si no hubo errores al actualizar)
349             * @throws Exception En caso de algun error al consultar la base de datos
350             */
351             public function insert()
352             {
353                 $datos = array(
354                     'grado_segmentacion' => $this-> grado,

```

```

354     'tabla_catalogo' => $this-> tabla_catalogo,
355     'nombre_columna_llave' => $this-> columna_llave,
356     'nombre_columna_descripcion' => $this-> columna_descripcion,
357     'id_raiz_arbol' => $this-> id_raiz,
358   );
359
360   $query = $this-> db-> insert('asu_raiz_x_catalogo',$datos);
361
362   if (!$query)
363   {
364     $this-> msg_error_log = " ( " . __METHOD__ . " ) => " . $this-
365 > db-> _error_number().': '.$this-> db-> _error_message();
366     $this-> msg_error_usr = "Ocurrió un error al insertar el catalogo x
raiz" ;
367     throw new Exception(__CLASS__);
368   }
369   else
370   {
371     $id = $this-> db-> insert_id();
372     $datos = array();
373
374     if (count($this-> relacionpadre) > 0)
375     {
376       for($i=0;$i< count($this-> relacionpadre);$i++)
377       {
378         array_push($datos, array(
379           'id_raiz_x_catalogo'=> $id,
380           'columna_padre'=> $this-> relacionpadre[$i],
381           'columna_hijo' => $this-> relacionhijo[$i]
382         ));
383       }
384
385       $query = $this-> db-> insert_batch('asu_relacion_catalogo',$datos);
386
387       if (!$query)
388       {
389         $this-> msg_error_log = " ( " . __METHOD__ . " ) => "
390 . $this-> db-> _error_number().': '.$this-> db-> _error_message();
391         $this-> msg_error_usr = "Ocurrió un error al insertar el catalogo x
raiz" ;
392         throw new Exception(__CLASS__);
393       }
394       else
395       {
396         return true;
397       }
398     }
399   }
400 /**
401 * Elimina el registro actual de la base de datos
402 *
403 * @access public
404 * @return boolean (Si no hubo errores al eliminar)
405 * @throws Exception En caso de algun error al consultar la base de datos
406 */
407 public function delete()
408 {
409   $query_relaciones = $this-> db-> delete('asu_relacion_catalogo',
array('id_raiz_x_catalogo' => $this-> getId()));
410   $query_datos_asu = $this-> db-
> delete('asu_arbol_segmentacion',array('grado_segmentacion'=> $this-> getGrado(), 'id_raiz' => $this-> getIdRaiz()));
411   $query = $this-> db-> delete('asu_raiz_x_catalogo', array('id' => $this-
> getId()));
412
413   if (!$query_relaciones || !$query_datos_asu || !$query)
414   {
415     $this-> msg_error_log = " ( " . __METHOD__ . " ) => " . $this-
416 > db-> _error_number().': '.$this-> db-> _error_message();
417     $this-> msg_error_usr = "Ocurrió un error al eliminar la raiz x
catalogo" ;
418     throw new Exception(__CLASS__);
419   }
420   else
421   {
422     return true;
423   }
424 }

```

# Archivo fuente para cie10\_model.php

La documentación para este archivo está disponible en [cie10\\_model.php](#)

```
1  <?php
2
3  /**
4   * Modelo Cie10
5   *
6   * @package    SIIGS
7   * @subpackage Modelo
8   * @author     Geovanni
9   * @created    2013-12-02
10  */
11 class Cie10_model extends CI_Model {
12
13
14     /**
15      * @access private
16      * @var int
17     */
18     private $id;
19
20     /**
21      * @access private
22      * @var string
23     */
24     private $descripcion;
25
26     /**
27      * @access private
28      * @var string
29     */
30     private $msg_error_log;
31
32     /**
33      * @access private
34      * @var string
35     */
36     private $msg_error_usr;
37
38     /**
39      * @access private
40      * @var int
41     */
42     private $offset;
43
44     /**
45      * @access private
46      * @var int
47     */
48     private $rows;
49
50     public function setOffset($value) {
51         $this-> offset = $value;
52     }
53     public function setRows($value) {
54         $this-> rows = $value;
55     }
56
57     /*****
58     /*Getters and setters block*/
59     *****/
60     public function getId() {
61         return $this-> id;
62     }
63
64     public function setId($value) {
65         $this-> id = $value;
66     }
67 }
```

```

68
69     public function getDescripcion() {
70         return $this-> descripcion;
71     }
72
73     public function setDescripcion($value) {
74         $this-> descripcion = $value;
75     }
76
77     /*****
78     /*Getters and setters block END*/
79     *****/
80
81
82     /**
83      * Devuelve los mensajes de error en caso de ocurrir alguna excepción
84      * 'usr' devuelve el mensaje para la vista de usuario
85      * 'log' devuelve el mensaje para el log de errores
86      *
87      * @access public
88      * @return string|boolean
89      * @param string $value, default 'usr' (Tipo mensaje)
90      */
91     public function getMsgError($value = 'usr')
92     {
93         if (!empty($this-> msg_error_usr))
94         {
95             if ($value == 'usr')
96                 return $this-> msg_error_usr;
97             else if ($value == 'log')
98                 return $this-> msg_error_log;
99         }
100        else
101        {
102            return false;
103        }
104    }
105
106    public function __construct()
107    {
108        $this-> load-> database();
109        if (!$this-> db-> conn_id)
110        {
111            throw new Exception("No se pudo conectar a la base de datos");
112        }
113    }
114
115    /**
116     *Devuelve el numero de registros
117     *
118     *@access public
119     *@return int
120     * @throws Exception En caso de algun error al consultar la base de datos
121     */
122    public function getNumRows()
123    {
124        $query = $this-> db-> query('select count(*) as num from cns_cielo');
125        if (!$query)
126        {
127            $this-> msg_error_log = " ". __METHOD__." => ". $this-
> db-> _error_number().': '.$this-> db-> _error_message();
128            $this-> msg_error_usr = "Ocurrió un error al obtener los datos del catalogo
cie10";
129            throw new Exception(__CLASS__);
130        }
131        else
132            return $query-> row()-> num;
133    }
134
135
136    /**
137     *Devuelve una lista con los registros existentes en el catalogo cie10
138     *
139     *@access public
140     *@return ArrayObject
141     * @throws Exception En caso de algun error al consultar la base de datos
142     */
143    public function getAll()
144    {
145        $consulta = 'SELECT id_cielo AS cie10, b.`descripcion` AS categoria,

```

```

a.descripcion AS descripcion, a.activo FROM cns_ciel0 a, cns_categoria_ciel0 b WHERE a.id_categoria =
b.id AND b.`activo` = 1';
146     if ((!empty($this-> offset) || $this-> offset == 0) && !empty(          $this-
> rows))
147         $consulta .= ' limit '.$this-> offset.', '.$this-> rows;
148
149         $datos = $this-> db-> query($consulta);
150
151         if (!$datos)
152     {
153         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
154         $this-> msg_error_usr = "Ocurrió un error al obtener los datos del catalogo
cie10";
155         throw new Exception(__CLASS__);
156     }
157     else
158     {
159         return $datos-> result();
160     }
161 }
162
163 /**
164 * Devuelve una lista con los registros existentes en el catalogo cie10 omitiendo los ID
165 *
166 * @access public
167 * @return ArrayObject
168 * @throws Exception En caso de algun error al consultar la base de datos
169 */
170 public function getData()
171 {
172     $consulta = 'select id_ciel0,descripcion from cns_ciel0';
173     $datos = $this-> db-> query($consulta);
174
175     if (!$datos)
176     {
177         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
178         $this-> msg_error_usr = "Ocurrió un error al obtener los datos del catalogo
cie10";
179         throw new Exception(__CLASS__);
180     }
181     else
182     {
183         return $datos-> result();
184     }
185 }
186
187 /**
188 * Devuelve una lista con los registros existentes en el catalogo requerido
189 *
190 * @access public
191 * @return ArrayObject
192 * @throws Exception En caso de algun error al consultar la base de datos
193 */
194 public function getCatalogoByName($cat)
195 {
196     $consulta = 'select * from cns_'.$cat;
197     $datos = $this-> db-> query($consulta);
198
199     if (!$datos)
200     {
201         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
202         $this-> msg_error_usr = "Ocurrió un error al obtener los datos del catalogo
cie10";
203         throw new Exception(__CLASS__);
204     }
205     else
206     {
207         return $datos-> result();
208     }
209 }
210
211 /**
212 * Devuelve la informacion de un registro del catalogo cie10 por su ID
213 *
214 * @access public
215 * @return Object

```

```

217     * @param int $id ID (Llave primaria)
218     * @throws Exception En caso de algun error al consultar la base de datos
219     */
220    public function getById($id)
221    {
222        $query = $this-> db-> get_where('cns_cielo', array('id_cielo' => $id));
223
224        if (!$query)
225        {
226            $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
227            $this-> msg_error_usr = "Ocurrió un error al obtener la información del
registro cielo";
228            throw new Exception(__CLASS__);
229        }
230        else
231            return $query-> row();
232    }
233
234    /**
235     * Accion para agregar registros del CIE10 a otros catalogos como el de EDA, IRA y
Consultas
236     * @param int $id el id del registro en el catalogo cielo
237     * @param string $catalogo nombre del catalogo donde se realizara la operacion
238     * @param boolean $valor agregar o eliminar el registro del catalogo
239     * @return boolean como el resultado de la operación
240     * @throws Exception Si ocurre algun error al consultar y modificar la base de datos
241     */
242
243    public function agregaEnCatalogo($id,$catalogo,$valor)
244    {
245        $existe = $this-> db-> query("select * from " . $catalogo . " where
id_cielo='"
246        . $id . "'");
247
248        if (!$existe)
249        {
250            $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this->
db-> _error_number().': '.$this-> db-> _error_message();
251            $this-> msg_error_usr = "Ocurrió un error al obtener los datos del
catalogo cielo";
252            throw new Exception(__CLASS__);
253        }
254        else
255        {
256            $existe = $existe-> num_rows();
257
258            if ($valor == 1)
259            {
260                if ($existe == 0)
261                    $consulta = "insert into " . $catalogo . "
(id_cielo,descripcion,activo,clave) values (" . $id . ",(select descripcion from cns_cielo
where id='"
262                    . $id . ') , 1,(select cielo from cns_cielo where id='"
263                    . $id . '))";
264                else
265                {
266                    $consulta = "update " . $catalogo . " set descripcion=
(select descripcion from cns_cielo where id='"
267                    . $id . ') , activo = 1, clave =(select cielo
from cns_cielo where id='"
268                    . $id . ') where id_cielo = '"
269                    . $id . '";
270
271                    $datos = $this-> db-> query($consulta);
272
273                    if (!$datos)
274                    {
275                        $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
276                        $this-> msg_error_usr = "Ocurrió un error al agregar el elemento en el
catalogo " . $catalogo;
277                        throw new Exception(__CLASS__);
278                    }
279                    else
280                    {
281                        $this-> db-> query("update cns_tabla_catalogo set
fecha_actualizacion = NOW() where descripcion='"
282                            . $catalogo . '");
283
284                    }
285                }
286            }
287        }
288    }

```

```

282     }
283
284     /**
285      * Accion para activar o desactivar registros de catalogos como el de EDA, IRA y
286      * @param int $id el id del registro en el catalogo
287      * @param string $catalogo nombre del catalogo donde se realizara la operacion
288      * @param boolean $valor agregar o eliminar el registro del catalogo
289      * @return boolean como el resultado de la operación
290      * @throws Exception Si ocurre algun error al consultar y modificar la base de datos
291      */
292
293     public function activaEnCatalogo($id,$catalogo,$valor)
294     {
295
296         $consulta = "update " . $catalogo . " set activo = " . ($valor ==
297 true) ? 1 : 0 . " where id = '" . $id . "' ";
298         $datos = $this-> db-> query($consulta);
299
300         if (!$datos)
301         {
302             $this-> db-> _error_number() .': ' . $this-> db-> _error_message();
303             $this-> msg_error_usr = "Ocurrió un error al activar el elemento en
304             el catalogo" . $catalogo;
305             throw new Exception(__CLASS__);
306         }
307         else
308         {
309             $this-> db-> query("update cns_tabla_catalogo set fecha_actualizacion
310             = NOW() where descripcion= '" . $catalogo . "'");
311             return true;
312         }
313     }
314
315     /**
316      * Devuelve los datos de un catalogo pasado como parametro
317      *
318      * @access public
319      * @param string $nombrecat Nombre del catalogo
320      * @return ArrayObject
321      * @throws Exception En caso de algun error al consultar la base de datos
322      */
323     public function getAllData($nombrecat)
324     {
325         $catalogos = $this-> db-> query('select * from ' . $nombrecat);
326
327         if (!$catalogos)
328         {
329             $this-> msg_error_log = "(" . __METHOD__ . ") => "
330             . $this-> db-> _error_number() .': ' . $this-> db-> _error_message();
331             $this-> msg_error_usr = "Ocurrió un error al obtener los datos de los
332             catálogos" ;
333             throw new Exception(__CLASS__);
334         }
335         else
336             return $catalogos-> result();
337
338     /**
339      * Revisa en la base de datos por registros duplicados en los campos pasados por parametro
340      *
341      * @access public
342      * @param string $campo (varios campos delimitados por | )
343      * @return boolean (Si no hubo errores al eliminar)
344      * @throws Exception En caso de algun error al consultar la base de datos
345      */
346     public function checkPk($campo)
347     {
348         $campo = urldecode($campo);
349         $campos = explode(' | ', $campo);
350         $querycampos = " ";
351
352         foreach($campos as $c)
353             $querycampos .= $c . ' , ';
354
355         $consulta = "select " . $querycampos . " count(*) from tmp_catalogo group by
356         " . substr($querycampos, 0, strlen($querycampos)-2) . " having count(*) > 1" ;
357         $query = $this-> db-> query($consulta);

```

```

354
355     //var_dump($consulta);
356
357     if (!$query)
358     {
359         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number(). ': ' . $this-> db-> _error_message();
360         $this-> msg_error_usr = "Ocurrió un error al eliminar el catálogo";
361         throw new Exception(__CLASS__);
362     }
363     else
364         return $query-> result();
365 }
366
367 /**
368 *Actualiza el objeto actual en la base de datos
369 *
370 *@access public
371 *@return boolean (Si no hubo errores al actualizar)
372 * @throws Exception En caso de algun error al consultar la base de datos
373 */
374 public function update()
375 {
376     $data = array('descripcion' => $this-> descripcion);
377
378     $this-> db-> where('id', $this-> getId());
379     $query = $this-> db-> update('cns_cielo', $data);
380
381     if (!$query)
382     {
383         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number(). ': ' . $this-> db-> _error_message();
384         $this-> msg_error_usr = "Ocurrió un error al actualizar los datos del
catálogo";
385         throw new Exception(__CLASS__);
386     }
387     else
388         return true;
389 }
390 }
391 }
```

# Archivo fuente para controladoraccion\_model.php

La documentación para este archivo está disponible en [controladoraccion\\_model.php](#)

```
1  <?php
2
3  /**
4   * Modelo ControladorAccion
5   *
6   * @package    SIIGS
7   * @subpackage Modelo
8   * @author     Geovanni
9   * @created    2013-09-26
10  */
11 class ControladorAccion_model extends CI_Model {
12
13     /**
14      * @access private
15      * @var int
16      */
17     private $id;
18
19     /**
20      * @access private
21      * @var int
22      */
23     private $id_controlador;
24
25     /**
26      * @access private
27      * @var int
28      */
29     private $id_accion;
30
31     /**
32      * @access private
33      * @var boolean
34      */
35     private $activo;
36
37     /**
38      * @access private
39      * @var array
40      */
41     private $acciones;
42
43     /**
44      * @access private
45      * @var string
46      */
47     private $msg_error_log;
48
49     /**
50      * @access private
51      * @var string
52      */
53     private $msg_error_usr;
54
55     /**
56      * Devuelve los mensajes de error en caso de ocurrir alguna excepción
57      * 'usr' devuelve el mensaje para la vista de usuario
58      * 'log' devuelve el mensaje para el log de errores
59      *
60      * @access public
61      * @return string/boolean
62      * @param string $value, default 'usr' (Tipo mensaje)
63      */
64     public function getMsgError($value = 'usr')
```

```

65
66    {
67        if (!empty($this-> msg_error_usr))
68        {
69            if ($value == 'usr')
70                return $this-> msg_error_usr;
71            else if ($value == 'log')
72                return $this-> msg_error_log;
73        }
74    }
75    {
76        return false;
77    }
78}
79 public function __construct()
80 {
81     $this-> load-> database();
82     if (!$this-> db-> conn_id)
83     {
84         throw new Exception("No se pudo conectar a la base de datos");
85     }
86 }
87 /**
88 *Devuelve el Id de una accion por controlador de una accion y controlador determinados
89 *
90 *@access public
91 *@return int
92 *@param int $controlador (Id del controlador)
93 *@param int $accion (Id de la accion)
94 * @throws Exception En caso de algun error al consultar la base de datos
95 */
96 public function getId($controlador , $accion)
97 {
98     $query = $this-> db-> get_where('sis_controlador_x_accion', array('id_accion' => $accion , 'id_controlador' => $controlador));
100
101    if (!$query)
102    {
103        $this-> msg_error_log = " (" . __METHOD__ . " ) => " . $this-
104        > db-> _error_number().':'. $this-> db-> _error_message();
105        $this-> msg_error_usr = "Ocurrió un error al obtener los datos de
acciones";
106        ;
107        throw new Exception(__CLASS__);
108    }
109    else
110        return @$query-> row()-> id;
111 }
112 /**
113 *Devuelve la informacion de una accion por controlador de acuerdo a su Id
114 *
115 *@access public
116 *@return Object
117 *@param int $id ID (Llave primaria)
118 * @throws Exception En caso de algun error al consultar la base de datos
119 */
120 public function getById($id)
121 {
122     $query = $this-> db-> get_where('sis_controlador_x_accion', array('id' => $id));
123
124    if (!$query)
125    {
126        $this-> msg_error_log = " (" . __METHOD__ . " ) => " . $this-
127        > db-> _error_number().':'. $this-> db-> _error_message();
128        $this-> msg_error_usr = "Ocurrió un error al obtener los datos de
acciones";
129        ;
130        throw new Exception(__CLASS__);
131    }
132    else
133        return $query-> row();
134 }
135 /**
136 *Devuelve el id de una accion por controlador
137 *de acuerdo al path
138 *
139 *@access public
140 *@return int

```

```

140     * @param string $path
141     * @throws Exception En caso de algun error al consultar la base de datos
142     */
143     public function getIdByPath($path)
144     {
145
146         $arreglo = explode('::', $path);
147
148         if (count($arreglo) != 3)
149         {
150             $this-> msg_error_log = "(" . __METHOD__ . ") => : el formato de
la cadena es incorrecta";
151             $this-> msg_error_usr = "El formato de la cadena no es correcto";
152             throw new Exception(__CLASS__);
153         }
154
155         $query = $this-> db-> query("SELECT c.id FROM sis_entorno a join
sis_controlador b on a.id = b.id_entorno join sis_controlador_x_accion c on b.id = c.id_controlador
join sis_accion d on c.id_accion = d.id where a.directorio = '" . $arreglo[0] . "' and b.clase
= '" . $arreglo[1] . "' and d.metodo = '" . $arreglo[2] . "'");
156
157         if (!$query)
158         {
159             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number() . ":" . $this-> db-> _error_message();
160             $this-> msg_error_usr = "Ocurrió un error al obtener los datos de
acciones";
161             throw new Exception(__CLASS__);
162         }
163         else
164         {
165             if ($query-> num_rows() == 0)
166                 return '0';
167             else
168                 return $query-> row()-> id;
169         }
170     }
171
172 /**
173 * Establece el mensaje de ayuda
174 *
175 * @access public
176 * @return int
177 * @param int $id
178 * @param string $textAyuda
179 * @throws Exception En caso de algun error al guardar el texto de ayuda
180 */
181 public function setHelp($id, $textAyuda)
182 {
183     $result = false;
184
185     $data['ayuda'] = $textAyuda;
186
187     $result = $this-> db-> update('sis_controlador_x_accion', $data, array('id' =>
$id));
188
189     if( $this-> db-> _error_number() ) {
190         $this-> error = true;
191         $this-> msg_error_usr = 'No se puede actualizar el registro';
192         $this-> msg_error_log = '(' . __METHOD__ . ') => ' . $this-> db-
> _error_number() . ":" . $this-> db-> _error_message();
193         throw new Exception('(' . __METHOD__ . ') => ' . $this-> db-> _error_number() . ":" .
$this-> db-> _error_message());
194     }
195
196     return $result;
197 }
198 }
```

# Archivo fuente para controlador\_model.php

*La documentación para este archivo está disponible en [controlador\\_model.php](#)*

```
1  <?php
2
3  /**
4   * Modelo Controlador
5   *
6   * @package    SIIGS
7   * @subpackage Modelo
8   * @author     Geovanni
9   * @created    2013-09-26
10  */
11 class Controlador_model extends CI_Model {
12
13     /**
14      * @access private
15      * @var    int
16      */
17     private $id;
18
19     /**
20      * @access private
21      * @var    int
22      */
23     private $id_entorno;
24
25     /**
26      * @access private
27      * @var    string
28      */
29     private $nombre;
30
31     /**
32      * @access private
33      * @var    string
34      */
35     private $descripcion;
36
37     /**
38      * @access private
39      * @var    string
40      */
41     private $clase;
42
43     /**
44      * @access private
45      * @var    string
46      */
47     private $msg_error_log;
48
49     /**
50      * @access private
51      * @var    string
52      */
53     private $msg_error_usr;
54
55
56     /**
57      * @access private
58      * @var    array
59      */
60     private $acciones;
61
62     /**
63      * @access private
64      * @var    int
```

```

65      */
66  private $offset;
67
68 /**
69  * @access private
70  * @var int
71  */
72 private $rows;
73
74 ****
75 /*Getters and setters block*/
76 ****
77 public function getId() {
78     return $this-> id;
79 }
80
81 public function setId($value) {
82     $this-> id = $value;
83 }
84
85 public function getNombre() {
86     return $this-> nombre;
87 }
88 public function setNombre($value) {
89     $this-> nombre = $value;
90 }
91
92 public function getDescripcion() {
93     return $this-> descripcion;
94 }
95
96 public function setDescripcion($value) {
97     $this-> descripcion = $value;
98 }
99
100 public function getIdEntorno() {
101     return $this-> id_entorno;
102 }
103
104 public function setIdEntorno($value) {
105     $this-> id_entorno = $value;
106 }
107
108 public function getAccion() {
109     return $this-> acciones;
110 }
111
112 public function setAccion($value) {
113     $this-> acciones = $value;
114 }
115
116 public function getClase() {
117     return $this-> clase;
118 }
119
120 public function setClase($value) {
121     $this-> clase = $value;
122 }
123
124 public function setOffset($value) {
125     $this-> offset = $value;
126 }
127 public function setRows($value) {
128     $this-> rows = $value;
129 }
130
131 ****
132 /*Getters and setters block END*/
133 ****
134
135 /**
136  * Devuelve los mensajes de error en caso de ocurrir alguna excepción
137  * 'usr' devuelve el mensaje para la vista de usuario
138  * 'log' devuelve el mensaje para el log de errores
139  *
140  * @access public
141  * @return string/boolean
142  * @param string $value, default 'usr' (Tipo mensaje)
143  */
144 public function getMsgError($value = 'usr')

```

```

145     {
146         if (!empty($this-> msg_error_usr))
147         {
148             if ($value == 'usr')
149                 return $this-> msg_error_usr;
150             else if ($value == 'log')
151                 return $this-> msg_error_log;
152         }
153     }
154     {
155         return null;
156     }
157 }
158
159     public function __construct()
160     {
161         $this-> load-> database();
162         if (!$this-> db-> conn_id)
163         {
164             throw new Exception("No se pudo conectar a la base de datos");
165         }
166     }
167
168 /**
169 *Devuelve todos los registros de la tabla controlador
170 *
171 *@access public
172 *@return ArrayObject
173 * @throws Exception En caso de algun error al consultar la base de datos
174 */
175     public function getAll()
176     {
177         $string = 'select a.id as id_entorno,a.nombre as entorno, b.* from sis_entorno a join
sis_controlador b on a.id = b.id_entorno order by a.nombre ,b.nombre';
178
179         if ((!empty($this-> offset) || $this-> offset == 0) && !empty($this->
rows))
180             $string .= ' limit '.$this-> offset.', '.$this-> rows;
181         $query = $this-> db-> query($string);
182
183         if (!$query)
184         {
185             $this-> msg_error_log = "(" . __METHOD__ . ". " . __METHOD__ . ") => "
.$this-> _error_number().': ' . $this-> db-> _error_message();
186             $this-> msg_error_usr = "Ocurrió un error al obtener los datos de
entornos";
187             throw new Exception(__CLASS__);
188         }
189         else
190             return $query-> result();
191     }
192
193 /**
194 *Devuelve el numero de registros
195 *
196 *@access public
197 *@return int
198 *@param int $entorno , default 0 (Id del entorno)
199 * @throws Exception En caso de algun error al consultar la base de datos
200 */
201     public function getNumRows($entorno = 0)
202     {
203         $query = $this-> db-> query('select count(*) as num from sis_entorno a join
sis_controlador b on a.id = b.id_entorno ' . ((($entorno == 0) ? '' : ' where b.id_entorno =
'$entorno)));
204         if (!$query)
205         {
206             $this-> msg_error_log = "(" . __METHOD__ . ". " . __METHOD__ . ") => "
.$this-> _error_number().': ' . $this-> db-> _error_message();
207             $this-> msg_error_usr = "Ocurrió un error al obtener los datos de
entornos";
208             throw new Exception(__CLASS__);
209         }
210         else
211             return $query-> row()-> num;
212     }
213
214 /**
215 *Devuelve la informacion de un controlador por su ID
216 *

```

```

217     * @access public
218     * @return Object
219     * @param int $id ID (Llave primaria)
220     * @throws Exception En caso de algun error al consultar la base de datos
221     */
222     public function getById($id)
223     {
224         $query = $this-> db-> get_where('sis_controlador', array('id' => $id));
225
226         if (!$query)
227         {
228             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
229             > db-> _error_number() . ': ' . $this-> db-> _error_message();
230             $this-> msg_error_usr = "Ocurrió un error al obtener los datos de
231             controladores";
232             ;
233             throw new Exception(__CLASS__);
234         }
235         else
236             return $query-> row();
237
238     /**
239      * Devuelve todos los controladores que pertenecen a un entorno por su Id
240      *
241      * @access public
242      * @return ArrayObject
243      * @param int $id ID (Id del entorno)
244      * @throws Exception En caso de algun error al consultar la base de datos
245      */
246     public function getByEntorno($id)
247     {
248         $string = 'select a.id as id_entorno,a.nombre as entorno, b.* from sis_entorno a join
249             sis_controlador b on a.id = b.id_entorno where b.id_entorno = ' . $id . ' order by a.nombre ,b.nombre';
250
251         if (!empty($this-> offset) || $this-> offset == 0) && !empty( $this-
252             > rows))
253             $string .= ' limit ' . $this-> offset . ',' . $this-> rows;
254
255         $query = $this-> db-> query($string);
256
257         if (!$query)
258         {
259             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
260             > db-> _error_number() . ': ' . $this-> db-> _error_message();
261             $this-> msg_error_usr = "Ocurrió un error al obtener los datos del
262             controlador";
263             ;
264             throw new Exception(__CLASS__);
265         }
266         else
267             return $query-> result();
268
269     /**
270      * Devuelve todas las acciones asignadas al controlador por su Id
271      *
272      * @access public
273      * @return ArrayObject
274      * @param int $id ID (Llave primaria)
275      * @throws Exception En caso de algun error al consultar la base de datos
276      */
277     public function getAcciones($id)
278     {
279         $query = $this-> db-> query("select a.id as id, a.nombre as accion , case when
280             b.id_accion is null or b.activo = FALSE then false else true end as activo from sis_accion a left
281             join sis_controlador_x_accion b on a.id = b.id_accion and b.id_controlador = " . $id);
282
283         if (!$query)
284         {
285             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
286             > db-> _error_number() . ': ' . $this-> db-> _error_message();
287             $this-> msg_error_usr = "Ocurrió un error al obtener las acciones asignadas
288             al controlador";
289             ;
290             throw new Exception(__CLASS__);
291         }
292         else
293             return $query-> result();
294
295     /**
296      * Devuelve los permisos asignados a un grupo sobre un entorno determinado

```

```

287     *(Mapea la información de las acciones asignadas a un controlador y los une con
288     * los permisos de un grupo sobre esas acciones)
289     *
290     *@access public
291     *@return ArrayObject
292     *@param int $entorno (Id del entorno)
293     *@param int $grupo (Id del grupo)
294     * @throws Exception En caso de algun error al consultar la base de datos
295     */
296     public function getPermisos($entorno , $grupo)
297     {
298         $query = $this-> db-> query("select a.id_entorno,d.id, a.nombre as
299         controlador, b.nombre as accion, case when isnull(c.id) or c.activo = 0 or isnull(c.activo) then 0
300         else c.id end as id , case when isnull(e.id) then 0 else 1 end as activo from sis_controlador a cross
301         join sis_accion b left join sis_controlador_x_accion c on c.id_accion = b.id and c.id_controlador =
302         a.id left join (sis_grupo d join sis_permiso e on d.id = e.id_grupo and d.id = " . $grupo . " )
303         on e.id_controlador_accion = c.id where a.id_entorno = " . $entorno . " order by
304         a.nombre,b.nombre" );
305         if (!$query)
306         {
307             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
308             > db-> _error_number().': '.$this-> db-> _error_message();
309             $this-> msg_error_usr = "Ocurrió un error al obtener los permisos del
310             controlador";
311             throw new Exception(__CLASS__);
312         }
313         else
314             return $query-> result();
315     }
316     /**
317      *Inserta en la tabla controlador, la información contenida en el objeto
318      *
319      *@access public
320      *@return int (Id de la inserción si no hubo errores al actualizar)
321      * @throws Exception En caso de algun error al consultar la base de datos
322      */
323     public function insert()
324     {
325         $data = array(
326             'nombre' => $this-> nombre,
327             'descripcion' => $this-> descripcion,
328             'id_entorno' => $this-> id_entorno,
329             'clase' => $this-> clase
330         );
331         $query = $this-> db-> insert('sis_controlador', $data);
332         if (!$query)
333         {
334             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
335             > db-> _error_number().': '.$this-> db-> _error_message();
336             $this-> msg_error_usr = "Ocurrió un error al insertar el controlador";
337             throw new Exception(__CLASS__);
338         }
339         else
340             return $this-> db-> insert_id($query);
341     }
342     /**
343      *Actualiza el objeto actual en la base de datos
344      *
345      *@access public
346      *@return boolean (Si no hubo errores al actualizar)
347      * @throws Exception En caso de algun error al consultar la base de datos
348      */
349     public function update()
350     {
351         $data = array(
352             'nombre' => $this-> nombre,
353             'descripcion' => $this-> descripcion,
354             'clase' => $this-> clase,
355             'id_entorno' => $this-> id_entorno
356         );
357         $this-> db-> where('id' , $this-> getId());
358         $query = $this-> db-> update('sis_controlador', $data);
359         if (!$query)
360     }

```

```

358         {
359             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
360             $this-> msg_error_usr = "Ocurrió un error al actualizar el controlador";
361             throw new Exception(__CLASS__);
362         }
363         else
364             return true;
365     }
366
367 /**
368 *Actualiza las acciones asignadas a un controlador
369 *
370 *@access public
371 *@return boolean (Si no hubo errores al actualizar)
372 * @throws Exception En caso de algun error al consultar la base de datos
373 */
374 public function accionesUpdate()
375 {
376
377     $this-> db-> query('delete from sis_permiso where id_controlador_accion in (select
id from sis_controlador_x_accion where id_controlador = '. $this-> getId() . ' and activo = 1 and
id_accion not in ('.implode(", ", $this-> acciones).'))');
378     $this-> db-> query('update sis_controlador_x_accion set activo = 0 where
id_controlador = '. $this-> getId());
379
380     foreach ($this-> acciones as $accion)
381     {
382         $item = array(
383             'id_controlador' => $this-> getId(),
384             'id_accion' => $accion
385         );
386
387         $exists = $this-> db-> get_where("sis_controlador_x_accion" , $item);
388
389         if (!$exists)
390         {
391             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
392             $this-> msg_error_usr = "Ocurrió un error al actualizar las acciones del
controlador (obtener acciones)";
393             throw new Exception(__CLASS__);
394         }
395
396         if ($exists-> num_rows() > 0)
397         {
398             $this-> db-> where($item);
399             $query = $this-> db-> update('sis_controlador_x_accion', array('activo'
=> '1'));
400
401             if (!$query && $remove)
402             {
403                 $this-> msg_error_log = "(" . __METHOD__ . ") => "
.$this-> db-> _error_number().': '.$this-> db-> _error_message();
404                 $this-> msg_error_usr = "Ocurrió un error al actualizar las acciones
del controlador (actualizar accion)";
405                 throw new Exception(__CLASS__);
406             }
407         }
408         else
409         {
410             $item['activo'] = '1';
411             $query = $this-> db-> insert('sis_controlador_x_accion', $item);
412
413             if (!$query)
414             {
415                 $this-> msg_error_log = "(" . __METHOD__ . ") => "
.$this-> db-> _error_number().': '.$this-> db-> _error_message();
416                 $this-> msg_error_usr = "Ocurrió un error al actualizar las acciones
del controlador (insertar accion)";
417                 throw new Exception(__CLASS__);
418             }
419         }
420     }
421     return true;
422 }
423
424 /**
425 * Elimina el registro actual de la base de datos
426 */

```

```

427     * @access public
428     * @return boolean (Si no hubo errores al eliminar)
429     * @throws Exception En caso de algun error al consultar la base de datos
430     */
431     public function delete()
432     {
433         $query = $this-> db-> delete('sis_controlador', array('id' => $this-
> getId()));
434
435         if (!$query)
436         {
437             $this-> msg_error_log = " (" . __METHOD__ . " ) => " . $this-
> db-> _error_number() . ': ' . $this-> db-> _error_message();
438             $this-> msg_error_usr = "Ocurrió un error al eliminar el controlador";
439             throw new Exception(__CLASS__);
440         }
441         else
442             return true;
443
444     }
445 }
446

```

# Archivo fuente para entorno\_model.php

*La documentación para este archivo está disponible en [entorno\\_model.php](#)*

```
1  <?php
2
3  /**
4   * Modelo Entorno
5   *
6   * @package    SIIGS
7   * @subpackage Modelo
8   * @author     Geovanni
9   * @created    2013-09-26
10  */
11 class Entorno_model extends CI_Model {
12
13     /**
14      * @access private
15      * @var int
16      */
17     private $id;
18
19     /**
20      * @access private
21      * @var string
22      */
23     private $nombre;
24
25     /**
26      * @access private
27      * @var string
28      */
29     private $descripcion;
30
31     /**
32      * @access private
33      * @var string
34      */
35     private $ip;
36
37     /**
38      * @access private
39      * @var string
40      */
41     private $hostname;
42
43     /**
44      * @access private
45      * @var string
46      */
47     private $directorio;
48
49     /**
50      * @access private
51      * @var string
52      */
53     private $msg_error_log;
54
55     /**
56      * @access private
57      * @var string
58      */
59     private $msg_error_usr;
60
61     /*****
62     /*Getters and setters block*/
63     *****/
64     public function getId() {
65         return $this-> id;
66     }
67     public function setId($value) {
```

```

68     $this-> id = $value;
69 }
70
71 public function getNombre() {
72     return $this-> nombre;
73 }
74 public function setNombre($value) {
75     $this-> nombre = $value;
76 }
77
78 public function getDescripcion() {
79     return $this-> descripcion;
80 }
81 public function setDescripcion($value) {
82     $this-> descripcion = $value;
83 }
84
85 public function getIp() {
86     return $this-> ip;
87 }
88 public function setIp($value) {
89     $this-> ip = $value;
90 }
91
92 public function getHostname() {
93     return $this-> hostname;
94 }
95 public function setHostname($value) {
96     $this-> hostname = $value;
97 }
98
99 public function getDirectorio() {
100    return $this-> directorio;
101 }
102 public function setDirectorio($value) {
103    $this-> directorio = $value;
104 }
105 /**
106  *Getters and setters block END*/
107 /**
108 /**
109 /**
110 * Devuelve los mensajes de error en caso de ocurrir alguna excepción
111 * 'usr' devuelve el mensaje para la vista de usuario
112 * 'log' devuelve el mensaje para el log de errores
113 *
114 * @access public
115 * @return string/boolean
116 * @param string $value, default 'usr' (Tipo mensaje)
117 */
118 public function getMsgError($value = 'usr')
119 {
120     if (!empty($this-> msg_error_usr))
121     {
122         if ($value == 'usr')
123             return $this-> msg_error_usr;
124         else if ($value == 'log')
125             return $this-> msg_error_log;
126     }
127     else
128     {
129         return null;
130     }
131 }
132
133 /**
134 * Devuelve la información del objeto en forma de string
135 *
136 * @access public
137 * @return string
138 *
139 */
140
141 public function getInfo()
142 {
143     $info = '';
144     $info .= (!empty($this-> id) ? $this-> id : '');
145 }
146
147

```

```

148     public function __construct()
149     {
150         $this-> load-> database();
151         if(!$this-> db-> conn_id)
152         {
153             throw new Exception("No se pudo conectar a la base de datos");
154         }
155     }
156
157 /**
158 *Devuelve todos los registros de la tabla entorno
159 */
160 *@access public
161 *@return ArrayObject
162 * @throws Exception En caso de algun error al consultar la base de datos
163 */
164 public function getAll()
165 {
166     $query = $this-> db-> get('sis_entorno');
167
168     if (!$query)
169     {
170         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
171 > db-> _error_number() . ': ' . $this-> db-> _error_message();
172         $this-> msg_error_usr = "Ocurrió un error al obtener los datos de
entornos";
173         throw new Exception(__CLASS__);
174     }
175     else
176         return $query-> result();
177 }
178
179 /**
180 *Devuelve la información de un entorno por su ID
181 */
182 *@access public
183 *@return Object
184 *@param int $id ID (Llave primaria)
185 * @throws Exception En caso de algun error al consultar la base de datos
186 */
187 public function getById($id)
188 {
189     $query = $this-> db-> get_where('sis_entorno', array('id' => $id));
190
191     if (!$query)
192     {
193         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
194 > db-> _error_number() . ': ' . $this-> db-> _error_message();
195         $this-> msg_error_usr = "Ocurrió un error al obtener la información del
entorno";
196         throw new Exception(__CLASS__);
197     }
198     else
199         return $query-> row();
200 }
201
202 /**
203 *Devuelve la información de un entorno por su nombre
204 */
205 *@access public
206 *@return Object
207 *@param string $nombre
208 * @throws Exception En caso de algun error al consultar la base de datos
209 */
210 public function getByNombre($nombre)
211 {
212     $query = $this-> db-> get_where('sis_entorno', array('nombre' => $nombre));
213
214     if (!$query)
215     {
216         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
217 > db-> _error_number() . ': ' . $this-> db-> _error_message();
218         $this-> msg_error_usr = "Ocurrió un error al obtener la información del
entorno (nombre)";
219         throw new Exception(__CLASS__);
220     }
221     else
222         return $query-> row();
223 }

```

```

222         /**
223          *Obtiene los permisos asignados al grupo
224          *
225          *@access public
226          *@return Object
227          *@param string $grupo
228          * @throws Exception En caso de algun error al consultar la base de datos
229          */
230         public function getPermissionsByGroup($grupo)
231     {
232         $query = $this-> db-> query("SELECT
233             CONCAT(e.directorio,':::',c.clase,':::',a.metodo) AS modulo FROM sis_controlador_x_accion ca
234             INNER JOIN sis_permiso p ON
235             ca.id=p.id_controlador_accion INNER JOIN sis_controlador c ON ca.id_controlador=c.id INNER JOIN
236             sis_entorno e ON c.id_entorno=e.id
237             INNER JOIN sis_accion a ON ca.id_accion=a.id WHERE
238             p.id_grupo=" . $grupo);
239             if (!$query)
240             {
241                 $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
242             > db-> _error_number().': ' . $this-> db-> _error_message();
243                 $this-> msg_error_usr = "Ocurrió un error al obtener la información de los
244                 permisos por grupo (getPermissionsByGrupo)";
245                 throw new Exception(__CLASS__);
246             }
247             else
248                 return $query-> result_array();
249         }
250
251         /**
252          *Inserta en la tabla entorno la información contenida en el objeto
253          *
254          *@access public
255          *@return int (Id de la inserción si no hubo errores al actualizar)
256          * @throws Exception En caso de algun error al consultar la base de datos
257          */
258         public function insert()
259     {
260         $data = array(
261             'nombre' => $this-> nombre,
262             'descripcion' => $this-> descripcion,
263             'ip' => $this-> ip,
264             'hostname' => $this-> hostname,
265             'directorio' => $this-> directorio
266         );
267
268         $query = $this-> db-> insert('sis_entorno', $data);
269
270         if (!$query)
271         {
272             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
273             > db-> _error_number().': ' . $this-> db-> _error_message();
274             $this-> msg_error_usr = "Ocurrió un error al insertar el entorno";
275             throw new Exception(__CLASS__);
276         }
277         else
278             return $this-> db-> insert_id($query);
279     }
280
281         /**
282          *Actualiza el objeto actual en la base de datos
283          *
284          *@access public
285          *@return boolean (Si no hubo errores al actualizar)
286          * @throws Exception En caso de algun error al consultar la base de datos
287          */
288         public function update()
289     {
290         $data = array(
291             'nombre' => $this-> nombre,
292             'descripcion' => $this-> descripcion,
293             'ip' => $this-> ip,
294             'hostname' => $this-> hostname,
295             'directorio' => $this-> directorio
296         );
297
298         $this-> db-> where('id', $this-> getId());
299         $query = $this-> db-> update('sis_entorno', $data);
300
301         if (!$query)
302             throw new Exception(__CLASS__);
303     }

```

```

295      {
296          $this-> msg_error_log = " ( " . __METHOD__ . " ) => " . $this-
297 > db-> _error_number().': '.$this-> db-> _error_message();
298          $this-> msg_error_usr = "Ocurrió un error al actualizar los datos del
299 entorno";
300      ;
301      throw new Exception(__CLASS__);
302  }
303
304 /**
305 * Elimina el registro actual de la base de datos
306 *
307 * @access public
308 * @return boolean (Si no hubo errores al eliminar)
309 * @throws Exception En caso de algun error al consultar la base de datos
310 */
311 public function delete()
312 {
313     $query = $this-> db-> delete('sis_entorno', array('id' => $this-> getId()));
314
315     if (!$query)
316     {
317         $this-> msg_error_log = " ( " . __METHOD__ . " ) => " . $this-
318 > db-> _error_number().': '.$this-> db-> _error_message();
319         $this-> msg_error_usr = "Ocurrió un error al eliminar el entorno" ;
320     }
321     else
322         return true;
323 }
324 }
```

# Archivo fuente para errorlog\_model.php

*La documentación para este archivo está disponible en [errorlog\\_model.php](#)*

```
1  <?php
2
3  /**
4   * Modelo Errorlog
5   *
6   * @package    SIIGS
7   * @subpackage Modelo
8   * @author     Pascual
9   * @created    2013-10-02
10  */
11 class Errorlog_model extends CI_Model
12 {
13     /**
14      * Guarda la instancia del objeto global CodeIgniter
15      * para utilizarlo en la función estática
16      *
17      * @access private
18      * @var    instance
19      */
20     private static $CI;
21
22     /**
23      * @access private
24      * @var    int
25      */
26     private $id;
27
28     /**
29      * @access private
30      * @var    int
31      */
32     private $id_usuario;
33
34     /**
35      * @access private
36      * @var    int
37      */
38     private $id_controlador_accion;
39
40     /**
41      * @access private
42      * @var    datetime
43      */
44     private $fecha_hora;
45
46     /**
47      * @access private
48      * @var    string
49      */
50     private $descripcion;
51
52
53     /**
54      * Estas variables no pertenecen a la tabla *
55      * ****
56     */
57     /**
58      * @access private
59      * @var    int
60      */
61     private $id_controlador;
62
63     /**
64      * @access private
65      * @var    int
66      */
67     private $id_accion;
```

```

68
69     /**
70      * @access private
71      * @var array
72     */
73     private $filters;
74
75
76     public function __construct()
77     {
78         parent::__construct();
79
80         self::$CI = & get_instance();
81
82         $this-> load-> database();
83         $this-> fecha_hora = date('Y-m-d H:i:s');
84         $this-> filters = array();
85
86         if( !$this-> db-> conn_id ) {
87             echo '<div class="error">ERROR: No se puede conectar con la Base de
Datos</div>';
88             ;
89             die();
90             //return false;
91             //throw new Exception ('ERROR: No se puede conectar con la Base de Datos');
92         }
93
94     public function getId()
95     {
96         return $this-> id;
97     }
98
99     public function setId($id)
100    {
101        return $this-> id = $id;
102    }
103
104    public function getId_usuario()
105    {
106        return $this-> id_usuario;
107    }
108
109    public function setId_usuario($id_usuario)
110    {
111        $this-> id_usuario = $id_usuario;
112    }
113
114    public function getId_controlador_accion()
115    {
116        return $this-> id_controlador_accion;
117    }
118
119    public function setId_controlador_accion($id_controlador_accion)
120    {
121        $this-> id_controlador_accion = $id_controlador_accion;
122    }
123
124    public function getFecha_hora()
125    {
126        return $this-> fecha_hora;
127    }
128
129    public function getDescripcion()
130    {
131        return $this-> descripcion;
132    }
133
134    public function setDescripcion($descripcion)
135    {
136        $this-> descripcion = $descripcion;
137    }
138
139    public function setId_controlador($id_controlador)
140    {
141        $this-> id_controlador = $id_controlador;
142    }
143
144    public function setId_accion($id_accion)
145    {
146        $this-> id_accion = $id_accion;

```

```

147     }
148
149     /**
150      * Inserta en la base de datos la informacion del error
151      *
152      * @access public static
153      * @param int $id_controlador
154      * @param int $id_accion
155      * @param string $descripcion
156      * @return void
157      */
158     public static function insert($path, $descripcion)
159     {
160         // Obtener el id_controlador_accion a partir del id_controlador y el id_accion
161         self::$CI-> load-> model(DIR_SIIGS."/ControladorAccion_model");
162         try{
163             $id_controlador_accion = self::$CI-> ControladorAccion_model-
164             > getIdByPath($path);
165
166             if(empty($id_controlador_accion)) {
167                 log_message('error', '(Errorlog_model::insert) No se encuentra la relación entre
168                 el controlador y la acción: '.$path.', Error '.
169                 self::$CI-> db-> _error_number().'');
170                 self-> _error_message();
171             }
172
173             $data = array(
174                 'id_usuario' => self::$CI-> session-> userdata(USER_LOGGED),
175                 'id_controlador_accion' => $id_controlador_accion,
176                 'fecha_hora' => date('Y-m-d H:i:s'), // inserta con la fecha y hora actual
177                 'descripcion' => $descripcion
178             );
179
180             self::$CI-> db-> insert('sis_error', $data);
181
182             if(self::$CI-> db-> _error_number()) {
183                 log_message('error', '(Errorlog_model::insert) Usuario: '.self::$CI-> session-
184                 > userdata(USER_LOGGED).', Path: '.$path.', Descripción: !
185                 $descripcion.', Error '.self::$CI-> db-> _error_number().'':
186                 self::$CI-> db-> _error_message());
187             }
188
189             /**
190              * Obtiene los datos del registro del Error que tiene el ID especificado
191              *
192              * @access public
193              * @param int $id           Si no se establece el valor de ID, se toma el valor del objeto
194              *                            actual
195              * @return object/booleanDevuelve un objeto con los datos del elemento solicitado, de lo
196              *                            contrario, false Si no se encontró el registro
197              */
198             public function getById($id)
199             {
200                 $result = false;
201
202                 $this-> db-> select('sis_entorno.nombre AS entorno, sis_controlador.nombre AS
203                 controlador, sis_accion.nombre AS accion, sis_error.*,
204                 sis_usuario.nombre_usuario AS usuario, sis_usuario.nombre,
205                 sis_usuario.apellido_paterno, sis_usuario.apellido_materno');
206                 $this-> db-> from('sis_error');
207                 $this-> db-> join('sis_usuario', 'sis_usuario.id = sis_error.id_usuario');
208                 $this-> db-> join('sis_controlador_x_accion', 'sis_controlador_x_accion.id =
209                 sis_error.id_controlador_accion');
210                 $this-> db-> join('sis_controlador', 'sis_controlador.id =
211                 sis_controlador_x_accion.id_controlador');
212                 $this-> db-> join('sis_accion', 'sis_accion.id =
213                 sis_controlador_x_accion.id_accion');
214                 $this-> db-> join('sis_entorno', 'sis_entorno.id = sis_controlador.id_entorno');
215                 $this-> db-> where('sis_error.id', $id);
216                 $query = $this-> db-> get();
217                 $result = $query-> row();
218
219                 if($this-> db-> _error_number()) {

```

```

213         log_message('sis_error', __METHOD__, Error '.self::$CI-> db-
> _error_number().': '.self::$CI-> db-> _error_message());
214     }
215 }
216
217 if(!empty($result)) {
218     $this-> load-> model(DIR_SIIGS.'/ControladorAccion_model');
219
220     $this-> id = $id;
221     $this-> id_usuario = $result-> id_usuario;
222     $this-> fecha_hora = $result-> fecha_hora;
223     $this-> descripcion = $result-> descripcion;
224     $this-> id_controlador_accion = $result-> id_controlador_accion;
225
226     $controlador_accion = $this-> ControladorAccion_model-> getById($result-
> id_controlador_accion);
227     $this-> id_controlador = $controlador_accion-> id_controlador;
228     $this-> id_accion = $controlador_accion-> id_accion;
229 }
230
231 return $result;
232 }
233
234 /**
235 * Agrega una nueva regla de filtrado al arreglo de filtros
236 *
237 * @access public
238 * @param string $columna Puede ser cualquier campo del objeto (id, id_usuario,
fecha_hora, descripcion, id_controlador_accion)
239 * @param string $condicion Establece la condicion a evaluar, entre los valores permitidos
estan: =, !=, >=, <=, like
240 * @param string $valor Valor contra el cual se realizará la evaluación del campo
241 * @return void/boolean Devuelve falso en caso de no poder establecer el filtro
242 */
243 public function addFilter($columna, $condicion, $valor)
244 {
245     $columnasPermitidas = array(
246         'id',
247         'id_usuario',
248         'fecha_hora',
249         'descripcion',
250         'id_entorno',
251         'id_controlador',
252         'id_accion'
253         // 'id_controlador_accion',
254     );
255
256     $condicionesPermitidas = array('=', '>', '<', '!=', '>=', '<=' , 'like');
257
258     if(!in_array($columna, $columnasPermitidas)) {
259         return false;
260     }
261
262     if(!in_array($condicion, $condicionesPermitidas)) {
263         return false;
264     }
265
266     if(empty($valor)) {
267         return false;
268     }
269
270     // Ejemplo de filtros permitidos por where de active records
271     // $filtros = array(
272     //     'name !=' => $name,
273     //     'id <'    => $id,
274     //     'date >'  => $date
275     // );
276     $this-> filters[$columna . '.' . $condicion] = $valor;
277 }
278
279 /**
280 * Elimina todos los filtros registrados
281 *
282 * @access public
283 * @return void
284 */
285 public function resetFilter()
286 {
287     $this-> filters = array();
288 }

```

```

289
290    /**
291     * Obtiene todos los registros de la tabla Error
292     * en caso de existir filtros, estos son aplicados a la consulta
293     *
294     * @access public
295     * @param int $offset      Establece el desplazamiento del primer registro a devolver,
296     *                         si se define solo el valor de offset
297     *                         el valor especifica el n mero de registros a retornar desde el
298     * comienzo del conjunto de resultados.
299     * @param int $row_count   Establece la cantidad de registros a devolver
300     * @return array           Devuelve conjunto de datos, como arreglo, obtenidos de la tabla
301
302     public function getAll($offset = null, $row_count = null)
303     {
304         $result = 0;
305
306         $this-> db-> select('sis_entorno.nombre AS entorno, sis_controlador.nombre AS
307 controlador, sis_accion.nombre AS accion, sis_error.*,
308                     sis_usuario.nombre_usuario AS usuario, sis_usuario.nombre,
309                     sis_usuario.apellido_paterno, sis_usuario.apellido_materno');
310         $this-> db-> from('sis_error');
311         $this-> db-> join('sis_usuario', 'sis_usuario.id = sis_error.id_usuario');
312         $this-> db-> join('sis_controlador_x_accion', 'sis_controlador_x_accion.id =
313                     sis_error.id_controlador_accion');
314         $this-> db-> join('sis_controlador', 'sis_controlador.id =
315                     sis_controlador_x_accion.id_controlador');
316         $this-> db-> join('sis_accion', 'sis_accion.id =
317                     sis_controlador_x_accion.id_accion');
318         $this-> db-> join('sis_entorno', 'sis_entorno.id = sis_controlador.id_entorno');
319         $this-> db-> order_by('fecha_hora', 'desc');
320         $this-> db-> order_by('id_entorno', 'desc');
321         $this-> db-> order_by('id_controlador', 'desc');
322         $this-> db-> order_by('id_accion', 'desc');
323
324         if( !empty($this-> filters) )
325             $this-> db-> where($this-> filters);
326
327         if(!empty($offset) && !empty($row_count))
328             $this-> db-> limit($offset, $row_count);
329         else if (!empty($offset))
330             $this-> db-> limit($offset);
331
332         $query = $this-> db-> get();
333
334         if($this-> db-> _error_number())
335             return false;
336         else {
337             $result = $query-> result();
338         }
339
340         return $result;
341     }
342
343     /**
344      * Obtiene el numero total de registros en la tabla Error
345      * en caso de existir filtros, estos son aplicados a la consulta
346      *
347      * @access public
348      * @return int/booleanNumero de registros encontrados o false en caso de algun error
349      */
350     public function getNumRows()
351     {
352         $result = false;
353
354         $this-> db-> select('sis_entorno.nombre AS entorno, sis_controlador.nombre AS
355 controlador, sis_accion.nombre AS accion, sis_error.*,
356                     sis_usuario.nombre_usuario AS usuario, sis_usuario.nombre,
357                     sis_usuario.apellido_paterno, sis_usuario.apellido_materno');
358         $this-> db-> from('sis_error');
359         $this-> db-> join('sis_usuario', 'sis_usuario.id = sis_error.id_usuario');
360         $this-> db-> join('sis_controlador_x_accion', 'sis_controlador_x_accion.id =
361                     sis_error.id_controlador_accion');
362         $this-> db-> join('sis_controlador', 'sis_controlador.id =
363                     sis_controlador_x_accion.id_controlador');
364         $this-> db-> join('sis_accion', 'sis_accion.id =
365                     sis_controlador_x_accion.id_accion');
366         $this-> db-> join('sis_entorno', 'sis_entorno.id = sis_controlador.id_entorno');
367
368         if(!empty($this-> filters))
369     }

```

```

358         $this-> db-> where($this-> filters);
359
360         $result = $this-> db-> count_all_results();
361
362         if($this-> db-> _error_number()) {
363             return false;
364         }
365
366         return $result;
367     }
368
369
370 /**
371 * Guardar el mensaje de error descriptivo en la base de datos,
372 * si no puede insertar el registro a la base de datos,
373 * el mensaje de error se guarda en el directorio logs,
374 * devuelve el mensaje de error para el usuario final
375 *
376 * @access public
377 * @param string $modelo Nombre del modelo que lanzó la excepción
378 * @param string $method Contiene el nombre de la clase y metodo donde se originó el error o
379 *                      el mensaje de error para mostrar al usuario final
380 * @return string Mensaje de error
381 */
382 public static function save($model, $method)
383 {
384     if (isset(self::$CI-> $model))
385     {
386         $msgErrUsr = self::$CI-> $model-> getMsgError();
387         $msgErrLog = self::$CI-> $model-> getMsgError('log');
388
389         Errorlog_model::insert(DIR_SIIGS.'::::'. $method, $msgErrLog);
390     }
391     else
392     {
393         $msgErrUsr = $model;
394         Errorlog_model::insert(DIR_SIIGS.'::::'. $method, $msgErrUsr);
395     }
396
397     return $msgErrUsr;
398 }
399
400 }
401 ?>
```

# Archivo fuente para georeferencia\_model.php

La documentación para este archivo está disponible en [georeferencia\\_model.php](#)

```
1  <?php
2
3  /**
4   * Modelo Georeferencia
5   *
6   * @package    SIIGS
7   * @subpackage Modelo
8   * @author     Pascual
9   * @created    2013-12-24
10  */
11 class Georeferencia_model extends CI_Model
12 {
13     /*****
14     * Estas variables no pertenecen a la tabla *
15     * *****/
16
17     /**
18     * @access private
19     * @var boolean
20     */
21     private $error;
22
23     /**
24     * @access private
25     * @var string
26     */
27     private $msg_error_usr;
28
29     /**
30     * @access private
31     * @var string
32     */
33     private $msg_error_log;
34
35
36     public function __construct()
37     {
38         parent::__construct();
39         $this-> load-> database();
40         $this-> error = false;
41         $this-> msg_error_usr = '';
42         $this-> msg_error_log = '';
43
44         /*if( !$this->db->conn_id ) {
45             throw new Exception ('ERROR: No se puede conectar con la Base de Datos');
46         }*/
47     }
48
49     /**
50      * Devuelve el mensaje de error,
51      * en caso de existir un error despues de ejecutar un metodo,
52      * de lo contrario false
53      *
54      * @access public
55      * @param string $type usr si se quiere devolver el mensaje de error a mostrar en la vista,
56      *                      log obtiene el mensaje de error con mas detalles para depuración,
57      *                      valor por defecto usr
58      * @return boolean/string
59      */
60     public function getMsgError($type = 'usr')
61     {
62         if($this-> error) {
63             if($type == 'usr')
64                 return $this-> msg_error_usr;
```

```

65         else if($type == 'log')
66             return $this->  msg_error_log;
67         else
68             return false;
69     }
70
71     return false;
72 }
73
74 /**
75 * Inserta los registros contenidos en la tabla cat_georeferencia
76 * a la tablaasu_georeferencia
77 *
78 * @access public
79 * @return boolean false Si no se ejecutó la inserción, true si se ejecutó la inserción
80 */
81 public function process()
82 {
83     $result = false;
84     $sql = 'REPLACE INTO
85             asu_georeferencia (id_asu, lat_dec, lon_dec, altitud)
86             SELECT
87                 b.id AS id_asu,
88                 cat_georeferencia.lat_dec,
89                 cat_georeferencia.lon_dec,
90                 cat_georeferencia.altitud
91             FROM
92                 cat_localidad a
93             JOIN
94                 asu_arbol_segmentacion b
95             ON a.id = b.id_tabla_original AND
96                 b.grado_segmentacion = 4 AND
97                 b.id_raiz = 1
98             JOIN
99                 cat_georeferencia
100            ON
101                b.id_raiz=1 AND
102                a.id_estado = cat_georeferencia.id_estado AND
103                a.id_municipio = cat_georeferencia.id_municipio AND
104                a.id_localidad = cat_georeferencia.id_localidad';
105
106     $result = $this-> db-> query($sql);
107
108     if( $this-> db-> _error_number() ) {
109         $this-> error = true;
110         $this-> msg_error_usr = 'Error al procesar la tabla georeferencia';
111         $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
112 > _error_number().': '.$this-> db-> _error_message();
113         throw new Exception('('.__METHOD__.') => ' . $this-> db-> _error_number().':
114         '.$this-> db-> _error_message());
115     }
116
117     return $result;
118 }
119 }
```

# Archivo fuente para grupo\_model.php

La documentación para este archivo está disponible en [grupo\\_model.php](#)

```
1  <?php
2  /**
3   * Modelo Grupo
4   *
5   * @package      SIIGS
6   * @subpackage   Modelo
7   * @author       Rogelio
8   * @created      2013-09-25
9   */
10  class Grupo_model extends CI_Model {
11      /**
12      * @access private
13      * @var int
14      */
15      private $id;
16      /**
17      * @access private
18      * @var string
19      */
20      private $nombre;
21      /**
22      * @access private
23      * @var string
24      */
25      private $descripcion;
26
27      /*****
28      * Estas variables no pertenecen a la tabla *
29      *****/
30
31      /**
32      * @access private
33      * @var string
34      */
35      private $msg_error_usr;
36      /**
37      * @access private
38      * @var string
39      */
40      private $msg_error_log;
41
42      public function __construct()
43      {
44          $this-> load-> database();
45          if (!$this-> db-> conn_id)
46              throw new Exception("No se pudo conectar a la base de datos");
47      }
48
49      public function getId()
50      {
51          return $this-> id;
52      }
53
54      public function setId($value) {
55          $this-> id = $value;
56      }
57
58      public function getNombre()
59      {
60          return $this-> nombre;
61      }
62
63      public function setNombre($nombre)
64      {
65          $this-> nombre = $nombre;
66      }
67
```

```

68     public function getDescripcion()
69     {
70         return $this-> descripcion;
71     }
72
73     public function setDescripcion($descripcion)
74     {
75         $this-> descripcion = $descripcion;
76     }
77
78     /**
79      * Asigna el mensaje de error a visualizar: para usuario final (usr) o para bitácora (log)
80      *
81      * @access public
82      * @param string $value tipo de error a visualizar: usr o log,
83      * @return boolean false si ocurrió algún error, true si se ejecutó
84      */
85     public function getMsgError($value = 'usr')
86     {
87         if ($value == 'log')
88             return $this-> msg_error_log;
89         return $this-> msg_error_usr;
90     }
91
92     /**
93      * Obtiene todos los grupos existentes
94      *
95      * @access public
96      * @param int $offset Establece el desplazamiento del
97      * primer registro a devolver,
98      *                      si se define solo el valor de offset
99      *                      el valor especifica el número de
100     * registros a retornar desde el comienzo del conjunto de resultados.
101    * @param int $row_count Establece la cantidad de registros a
102    * devolver
103    * @return void/arrayobject false si ocurrió algún error, array object si se ejecutó
104    * correctamente
105    */
106    public function getAll($keywords = '', $offset = null, $row_count = null)
107    {
108        if(!empty($offset) && !empty($row_count))
109            $this-> db-> limit($offset, $row_count);
110        else if (!empty($offset))
111            $this-> db-> limit($offset);
112        if (empty($keywords))
113            $query = $this-> db-> get('sis_grupo');
114        else
115        {
116            $this-> db-> select('*');
117            $this-> db-> from('sis_grupo');
118            $this-> db-> like('nombre', $keywords);
119            $this-> db-> or_like('descripcion', $keywords);
120            $query = $this-> db-> get();
121
122            if (!$query){
123                $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
124                $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
125                > db-> _error_number() . ': ' . $this-> db-> _error_message();
126                throw new Exception(__CLASS__);
127            }
128            else
129                return $query-> result();
130        }
131
132        /**
133         * Obtiene el grupo solicitado
134         *
135         * @access public
136         * @param int $id id del grupo
137         * @return void/object false si ocurrió algún error, object si se ejecutó
138         * correctamente
139         */
140        public function getById($id)
141        {
142            $query = $this-> db-> get_where('sis_grupo', array('id' => $id));
143            if (!$query){
144                $this-> msg_error_usr = "Servicio temporalmente no disponible." ;

```

```

140           $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
141           throw new Exception(__CLASS__);
142       }
143   }
144   else
145       return $query-> row();
146   return;
147 }
148 /**
149 * Obtiene el grupo solicitado con sus entornos vinculados
150 *
151 * @access public
152 * @param int $id id del grupo
153 * @return void/object false si ocurrió algún error, object si se ejecutó
correctamente
154 */
155 public function getEntornosById($id)
156 {
157     $sql = "SELECT DISTINCT e.nombre as entorno FROM sis_grupo g INNER JOIN
sis_permiso p ON g.id=p.id_grupo INNER JOIN sis_controlador_x_accion ca ON
p.id_controlador_accion=ca.id
158         INNER JOIN sis_controlador c ON ca.id_controlador=c.id INNER JOIN sis_entorno e
ON c.id_entorno=e.id WHERE g.id=" . $id . " ORDER BY e.nombre" ;
159     $query = $this-> db-> query($sql);
160     if (!$query){
161         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
162         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
163         throw new Exception(__CLASS__);
164     }
165     else
166         return $query-> result();
167     return;
168 }
169 /**
170 * Obtiene el grupo solicitado
171 *
172 * @access public
173 * @param string $name nombre del grupo
174 * @return void/object false si ocurrió algún error, object si se ejecutó
correctamente
175 */
176 public function getByNome($name)
177 {
178     $query = $this-> db-> get_where('sis_grupo', array('nombre' => $name));
179
180     if (!$query){
181         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
182         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
183         throw new Exception(__CLASS__);
184     }
185     else
186         return $query-> row();
187     return;
188 }
189 /**
190 * Obtiene el numero total de grupos
191 *
192 * @access public
193 * @param boolean/string $keywords false no hay texto a buscar/string con
texto a buscar
194 * @return int
195 */
196 public function getNumRows($keywords = '')
197 {
198     if (!$keywords)
199         $query = $this-> db-> get('sis_grupo');
200     else
201     {
202         $this-> db-> select('*');
203         $this-> db-> from('sis_grupo');
204         $this-> db-> like('nombre', $keywords);
205         $this-> db-> or_like('descripcion', $keywords);
206         $query = $this-> db-> get();
207     }
208     if(!$query) {

```

```

210             $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
211             $this-> msg_error_log = "(" . __METHOD__ . ") => ' . $this-> db-
> _error_number().': '$this-> db-> _error_message();"
212             throw new Exception(__CLASS__);
213         }
214     }
215 }
216
217 /**
218 * Inserta en la base de datos los datos del grupo (datos en propiedades)
219 *
220 * @access public
221 * @return boolean false si ocurrió algún error, true si se
ejecutó correctamente
222 */
223 public function insert()
224 {
225     $data = array(
226         'nombre' => $this-> nombre,
227         'descripcion' => $this-> descripcion
228     );
229     $result = $this-> db-> insert('sis_grupo', $data);
230     if (!$result){
231         $this-> msg_error_usr = "Servicio temporalmente no disponible."
232         $this-> msg_error_log = "(" . __METHOD__ . ") => "
> $this-> db-> _error_number().': '$this-> db-> _error_message();"
233         throw new Exception(__CLASS__);
234     }
235     return $result;
236 }
237
238 /**
239 * Actualiza en la base de datos los datos del grupo (datos en propiedades)
240 *
241 * @access public
242 * @return boolean false si ocurrió algún error, true si se
ejecutó correctamente
243 */
244 public function update()
245 {
246     $data = array(
247         'descripcion' => $this-> descripcion
248     );
249     $this-> db-> where('id', $this-> id);
250     $result = $this-> db-> update('sis_grupo', $data);
251     if (!$result){
252         $this-> msg_error_usr = "Servicio temporalmente no disponible."
253         $this-> msg_error_log = "(" . __METHOD__ . ") => "
> $this-> db-> _error_number().': '$this-> db-> _error_message();"
254         throw new Exception(__CLASS__);
255     }
256     return $result;
257 }
258
259 /**
260 * Elimina de la base de datos al grupo (id en propiedades)
261 *
262 * @access public
263 * @return boolean false si ocurrió algún error, true si se
ejecutó correctamente
264 */
265 public function delete()
266 {
267     $result = $this-> db-> delete('sis_grupo', array('id' => $this-> getId()));
268     if (!$result){
269         $this-> msg_error_usr = "Servicio temporalmente no disponible."
270         if (strpos($this-> db-> _error_message(), 'Cannot delete or update a parent
row') !== false) {
271             $this-> msg_error_usr = "No se puede eliminar debido a que ese registro
contiene información vinculada." ;
272         }
273         $this-> msg_error_log = "(" . __METHOD__ . ") => "
> $this-> db-> _error_number().': '$this-> db-> _error_message();"
274         throw new Exception(__CLASS__);
275     }
276     return $result;
277 }
278
279 }
280 ?>
```

# Archivo fuente para hemoglobina\_model.php

La documentación para este archivo está disponible en [hemoglobina\\_model.php](#)

```
1  <?php
2
3  /**
4   * Modelo Hemoglobina
5   *
6   * @package    SIIGS
7   * @subpackage Modelo
8   * @author     Geovanni
9   * @created    2014-07-21
10  */
11 class Hemoglobina_model extends CI_Model
12 {
13     /*****
14     * Estas variables no pertenecen a la tabla *
15     * *****/
16
17     /**
18     * @access private
19     * @var boolean
20     */
21     private $error;
22
23     /**
24     * @access private
25     * @var string
26     */
27     private $msg_error_usr;
28
29     /**
30     * @access private
31     * @var string
32     */
33     private $msg_error_log;
34
35
36     public function __construct()
37     {
38         parent::__construct();
39         $this-> load-> database();
40         $this-> error = false;
41         $this-> msg_error_usr = '';
42         $this-> msg_error_log = '';
43
44         /*if( !$this->db->conn_id ) {
45             throw new Exception ('ERROR: No se puede conectar con la Base de Datos');
46         }*/
47     }
48
49     /**
50      * Devuelve el mensaje de error,
51      * en caso de existir un error despues de ejecutar un metodo,
52      * de lo contrario false
53      *
54      * @access public
55      * @param string $type usr si se quiere devolver el mensaje de error a mostrar en la vista,
56      *                      log obtiene el mensaje de error con mas detalles para depuración,
57      *                      valor por defecto usr
58      * @return boolean/string
59      */
60     public function getMsgError($type = 'usr')
61     {
62         if($this-> error) {
63             if($type == 'usr')
64                 return $this-> msg_error_usr;
```

```

65             else if($type == 'log')
66                 return $this->  msg_error_log;
67             else
68                 return false;
69         }
70     }
71 }
72
73
74 /**
75 * Inserta los registros contenidos en la tabla cat_georeferencia
76 * a la tablaasu_georeferencia
77 *
78 * @access public
79 * @return boolean false Si no se ejecutó la inserción, true si se ejecutó la inserción
80 */
81 public function process()
82 {
83     $result = false;
84     $sql = 'REPLACE INTO
85             asu_hemoglobina_altitud (id_localidad_asu, altitud, mujer_no_embarazada,
86 mujer_embarazada_ninio_6_59_meses)
87             SELECT
88                 b.id AS id_localidad_asu,
89                 cat_hemoglobina_altitud.altitud,
90                 cat_hemoglobina_altitud.mujer_no_embarazada,
91                 cat_hemoglobina_altitud.mujer_embarazada_ninio_6_59_meses
92             FROM
93                 cat_localidad a
94             JOIN
95                 asu_arbol_segmentacion b
96             ON a.id = b.id_tabla_original AND
97                 b.grado_segmentacion = 4 AND
98                 b.id_raiz = 1
99             JOIN
100                cat_hemoglobina_altitud
101            ON
102                b.id_raiz=1 AND
103                a.id_estado = cat_hemoglobina_altitud.id_estado AND
104                a.id_municipio = cat_hemoglobina_altitud.id_municipio AND
105                a.id_localidad = cat_hemoglobina_altitud.id_localidad';
106
107     $result = $this-> db-> query($sql);
108
109     if( $this-> db-> _error_number() ) {
110         $this-> error = true;
111         $this-> msg_error_usr = 'Error al procesar la tabla georeferencia';
112         $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
> _error_number().': '.$this-> db-> _error_message();
113         throw new Exception('('.__METHOD__.') => ' . $this-> db-> _error_number().':
$this-> db-> _error_message());
114     }
115
116     return $result;
117 }
118 }
119 }
```

# Archivo fuente para menu\_model.php

La documentación para este archivo está disponible en [menu\\_model.php](#)

```
1  <?php
2
3  /**
4   * Modelo Menu
5   *
6   * @package    SIIGS
7   * @subpackage Modelo
8   * @author     Pascual
9   * @created    2013-10-03
10  */
11 class Menu_model extends CI_Model
12 {
13     /**
14      * @access private
15      * @var int
16      */
17     private $id;
18
19     /**
20      * @access private
21      * @var int
22      */
23     private $id_padre;
24
25     /**
26      * @access private
27      * @var int
28      */
29     private $id_raiz;
30
31     /**
32      * @access private
33      * @var string
34      */
35     private $nombre;
36
37     /**
38      * @access private
39      * @var int
40      */
41     private $id_controlador;
42
43     /**
44      * @access private
45      * @var boolean
46      */
47     private $ruta;
48
49     /**
50      * @access private
51      * @var string
52      */
53     private $atributo;
54
55
56     /**
57      * Estas variables no pertenecen a la tabla *
58      * ****
59     */
60     /**
61      * @access private
62      * @var array
63      */
64     private $filters;
65
66     /**
67      * @access private
```

```

68     * @var    boolean
69     */
70 private $error;
71
72 /**
73 * @access private
74 * @var    string
75 */
76 private $msg_error_usr;
77
78 /**
79 * @access private
80 * @var    string
81 */
82 private $msg_error_log;
83
84
85 public function __construct()
86 {
87     parent::__construct();
88     $this-> load-> database();
89     $this-> error = false;
90     $this-> msg_error_usr = '';
91     $this-> msg_error_log = '';
92
93     /*if( !$this->db->conn_id ) {
94         throw new Exception ('ERROR: No se puede conectar con la Base de Datos');
95     }*/
96 }
97
98 public function getId()
99 {
100    return $this-> id;
101}
102
103 public function setId($id)
104 {
105    return $this-> id = $id;
106}
107
108 public function getId_padre()
109 {
110    return $this-> id_padre;
111}
112
113 public function setId_padre($id_padre)
114 {
115    $this-> id_padre = $id_padre;
116}
117
118 public function getId_raiz()
119 {
120    return $this-> id_raiz;
121}
122
123 public function setId_raiz($id_raiz)
124 {
125    return $this-> id_raiz = $id_raiz;
126}
127
128 public function getNombre()
129 {
130    return $this-> nombre;
131}
132
133 public function setNombre($nombre)
134 {
135    $this-> nombre = $nombre;
136}
137
138 public function setId_controlador($id_controlador)
139 {
140    $this-> id_controlador = $id_controlador;
141}
142
143 public function getId_controlador()
144 {
145    $this-> id_controlador;
146}
147

```

```

148     public function setRuta($ruta)
149     {
150         $this-> ruta = $ruta;
151     }
152
153     public function getRuta()
154     {
155         return $this-> ruta;
156     }
157
158     public function setAtributo($atributo)
159     {
160         $this-> atributo = $atributo;
161     }
162
163     public function getAtributo()
164     {
165         return $this-> atributo;
166     }
167
168     /**
169      * Devuelve el mensaje de error,
170      * en caso de existir un error despues de ejecutar un metodo,
171      * de lo contrario false
172      *
173      * @access public
174      * @param string $type usr si se quiere devolver el mensaje de error a mostrar en la vista,
175      *                      log obtiene el mensaje de error con mas detalles para depuración,
176      *                      valor por defecto usr
177      * @return boolean/string
178      */
179     public function getMsgError($type = 'usr')
180     {
181         if($this-> error) {
182             if($type == 'usr')
183                 return $this-> msg_error_usr;
184             else if($type == 'log')
185                 return $this-> msg_error_log;
186             else
187                 return false;
188         }
189
190         return false;
191     }
192
193
194     /**
195      * Inserta en la base de datos, la informacion contenida en el objeto
196      *
197      * @access public
198      * @return boolean false Si no se ejecutó la inserción, true si se ejecutó la inserción
199      */
200     public function insert()
201     {
202         $result = false;
203         $data = array();
204
205         if( iset($this-> id_padre) )
206             $data['id_padre'] = $this-> id_padre;
207
208         if( iset($this-> id_raiz) )
209             $data['id_raiz'] = $this-> id_raiz;
210
211         if( iset($this-> id_controlador) )
212             $data['id_controlador'] = $this-> id_controlador;
213
214         if( iset($this-> ruta) )
215             $data['ruta'] = $this-> ruta;
216
217         if( iset($this-> atributo) )
218             $data['atributo'] = $this-> atributo;
219
220         $data['nombre'] = $this-> nombre;
221
222         $result = $this-> db-> insert('sis_menu', $data);
223
224         if($this-> db-> _error_number() ) {
225             $this-> error = true;
226             $this-> msg_error_usr = 'No se puede insertar el registro';
227             $this-> msg_error_log = '('.__METHOD__.') => '. $this-> db-

```

```

>     _error_number().' : '.$this-> db-> _error_message();
228     throw new Exception();
229 } else {
230     // Obtiene el id asignado a la ultima inserción
231     $this-> id = $this-> db-> insert_id();
232 }
233
234     return $result;
235 }
236
237 /**
238 * Actualiza los datos del objeto actual
239 *
240 * @access public
241 * @param int $id Si no se establece el valor de ID, se toma el valor del objeto actual
242 * @return boolean false Si no se ejecutó la actualización, true si se ejecutó la actualización
243 */
244 public function update($id = null)
245 {
246     $result = false;
247
248     //if( isset($this->id_padre) )
249     $data['id_padre'] = $this-> id_padre;
250
251     //if( isset($this->id_raiz) )
252     $data['id_raiz'] = $this-> id_raiz;
253
254     //if( isset($this->id_controlador) )
255     $data['id_controlador'] = $this-> id_controlador;
256
257     //if( isset($this->ruta) )
258     $data['ruta'] = $this-> ruta;
259
260     $data['atributo'] = $this-> atributo;
261
262     $data['nombre'] = $this-> nombre;
263
264     $id = is_null($id) ? $this-> id : $id;
265     $result = $this-> db-> update('sis_menu', $data, array('id' => $id));
266
267     if( $this-> db-> _error_number() ) {
268         $this-> error = true;
269         $this-> msg_error_usr = 'No se puede insertar el registro';
270         $this-> msg_error_log = '(' . __METHOD__ . ') => ' . $this-> db-
> _error_number().' : '.$this-> db-> _error_message();
271         throw new Exception('(' . __METHOD__ . ') => ' . $this-> db-> _error_number().' :
'. $this-> db-> _error_message());
272     } /*else {
273         // Obtiene el id asignado a la ultima inserción
274         $this->id = $this->db->insert_id();
275     }*/
276
277     return $result;
278 }
279
280 /**
281 * Elimina el registro actual de la base de datos
282 *
283 * @access public
284 * @param int $id Si no se establece el valor de ID, se toma el valor del objeto actual
285 * @return int    false Si no se eliminó el registro, true si se ejecutó la eliminación
286 */
287 public function delete($id = null)
288 {
289     $result = false;
290
291     $id = is_null($id) ? $this-> id : $id;
292
293     if(is_array($id)) {
294         // Eliminar un conjunto de registros
295         foreach ($id as $idx) {
296             $result = $this-> db-> delete('sis_menu', array('id' => $idx));
297
298             if(empty($result)) {
299                 $this-> error = true;
300                 $this-> msg_error_usr = 'No se puede eliminar el registro';
301                 $this-> msg_error_log = '(' . __METHOD__ . ') => ' . $this-> db-
> _error_number().' : '.$this-> db-> _error_message();
302                 throw new Exception();
303             }
304         }
305     }
306
307     return $result;
308 }

```

```

304
305     }
306     } else {
307         // Eliminar un solo registro
308         $result = $this-> db-> delete('sis_menu', array('id' => $id));
309
310         if(empty($result)) {
311             $this-> error = true;
312             $this-> msg_error_usr = 'No se puede eliminar el registro';
313             $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
314             > _error_number().': '.$this-> db-> _error_message();
315             throw new Exception();
316         }
317
318         return $result;
319     }
320
321     /**
322      * Elimina el conjunto de registros que cumplen con el o los criterios de filtrado
323      *
324      * @access public
325      * @return int false Si no se ejecutó la inserción, true si se ejecutó la inserción
326      */
327     public function deleteByFilter()
328     {
329         $result = false;
330
331         if(empty($this-> filters)) {
332             $this-> error = true;
333             $this-> msg_error_usr = 'No se encontraron filtros definidos';
334             $this-> msg_error_log = '('.__METHOD__.') => No se encontraron filtros
335             definidos para la eliminación';
336
337             throw new Exception();
338         }
339
340         $this-> db-> where($this-> filters);
341         $result = $this-> db-> delete('sis_menu');
342
343         if(empty($result)) {
344             $this-> error = true;
345             $this-> msg_error_usr = 'No se pueden eliminar los registros';
346             $this-> msg_error_log = '('.__METHOD__.') => No se pueden eliminar los
347             registros';
348
349             throw new Exception();
350         }
351
352         /**
353          * Obtiene los datos del registro de la menu que tiene el ID especificado
354          *
355          * @access public
356          * @param int $id      Si no se establece el valor de ID, se toma el valor del objeto
357          * actual
358          * @return object/boolean Devuelve el objeto con sus datos correspondientes, de lo
359          * contrario, false Si no se encontró el registro
360          */
361         public function getById($id)
362         {
363             $result = false;
364
365             $this-> db-> select('sis_menu.id, sis_menu.id_padre, sis_menu.id_raiz,
366             sis_menu.nombre, sis_menu.atributo, sis_menu.id_controlador, sis_entorno.nombre AS nombre_entorno,
367             sis_entorno.id AS id_entorno,
368             sis_menu.ruta, raiz.nombre AS nombre_raiz, padre.nombre AS
369             nombre_padre, sis_controlador.nombre AS nombre_controlador');
370             $this-> db-> from('sis_menu');
371             $this-> db-> join('sis_menu raiz', 'raiz.id=sis_menu.id_raiz', 'left');
372             $this-> db-> join('sis_menu padre', 'padre.id=sis_menu.id_padre', 'left');
373             $this-> db-> join('sis_controlador', 'sis_controlador.id=sis_menu.id_controlador',
374             'left');
375             $this-> db-> join('sis_entorno', 'sis_entorno.id=sis_controlador.id_entorno',
376             'left');
377             $this-> db-> where('sis_menu.id', $id);
378             $query = $this-> db-> get();
379             $result = $query-> row();
380
381             if($this-> db-> _error_number()) {

```

```

374         $this-> error = true;
375         $this-> msg_error_usr = 'No se encontraron registros en la busqueda';
376         $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
> _error_number();
377         throw new Exception();
378     }
379
380     if(!empty($result)) {
381         $this-> id = $id;
382         $this-> id_padre = $result-> id_padre;
383         $this-> id_raiz = $result-> id_raiz;
384         $this-> id_controlador = $result-> id_controlador;
385         $this-> ruta = $result-> ruta;
386         $this-> nombre = $result-> nombre;
387     }
388
389     return $result;
390 }
391
392 /**
393 * Agrega una nueva regla de filtrado al arreglo de filtros
394 *
395 * @access public
396 * @param string $columna Puede ser cualquier campo del objeto (id, id_usuario,
397 * fecha_hora, parametros, id_controlador_accion)
398 * @param string $condicion Establece la condicion a evaluar, entre los valores permitidos
399 * estan: =, !=, >, <, like
400 * @param string $valor Valor contra el cual se realizará la evaluación del campo
401 * @return void/boolean Devuelve falso en caso de no poder establecer el filtro
402 */
403 public function addFilter($columna, $condicion, $valor)
404 {
405     $columnasPermitidas = array(
406         'id',
407         'id_padre',
408         'id_raiz',
409         'id_controlador',
410         'ruta',
411         'atributo',
412     );
413
414     $condicionesPermitidas = array('=', '>', '<', '!=', '>=',
415                                     '<=');
416
417     if(!in_array($columna, $columnasPermitidas)) {
418         $this-> error = true;
419         $this-> msg_error_usr = 'ERROR: Columna no permitida en el filtro';
420         $this-> msg_error_log = '('.__METHOD__.') => Columna no permitida en el
421         filtro';
422
423         throw new Exception();
424     }
425
426     if(!in_array($condicion, $condicionesPermitidas)) {
427         $this-> error = true;
428         $this-> msg_error_usr = 'ERROR: Condición no permitida en el filtro';
429         $this-> msg_error_log = '('.__METHOD__.') => Condición no permitida en el
430         filtro';
431
432         throw new Exception();
433     }
434
435     if(empty($valor)) {
436         $this-> error = true;
437         $this-> msg_error_usr = 'ERROR: Debe definir un valor para el filtro';
438         $this-> msg_error_log = '('.__METHOD__.') => Debe definir un valor para el
439         filtro';
440
441         throw new Exception();
442     }
443
444     // Ejemplo de filtros permitidos por where de active records
445     // $filtros = array(
446     //     'name !=' => $name,
447     //     'id <'    => $id,
448     //     'date >'   => $date
449     // );
450     $this-> filters['sis_menu.' . $columna . ' ' . $condicion] = $valor;
451 }
452
453 /**

```

```

448     * Elimina todos los filtros registrados
449     *
450     * @access public
451     * @return void
452     */
453     public function resetFilter()
454     {
455         $this-> filters = array();
456     }
457
458 /**
459     * Obtiene todos los registros de la tabla Menu
460     * en caso de existir filtros, estos son aplicados a la consulta
461     *
462     * @access public
463     * @param int $offset      Establece el desplazamiento del primer registro a devolver,
464     *                         si se define solo el valor de offset
465     *                         el valor especifica el n mero de registros a retornar desde el
466     * comienzo del conjunto de resultados.
467     * @param int $row_count   Establece la cantidad de registros a devolver
468     * @return array object    Devuelve un arreglo de objetos obtenidos de la base de datos
469
470     public function getAll($offset = null, $row_count = null)
471     {
472         $result = 0;
473
474         $this-> db-> select('sis_menu.id, sis_menu.id_padre, sis_menu.id_raiz,
475         sis_menu.atributo, sis_menu.nombre, sis_menu.id_controlador,
476         sis_menu.ruta, raiz.nombre AS nombre_raiz, padre.nombre AS
477         nombre_padre, sis_controlador.nombre AS nombre_controlador');
478         $this-> db-> from('sis_menu');
479         $this-> db-> join('sis_menu raiz', 'raiz.id=sis_menu.id_raiz', 'left');
480         $this-> db-> join('sis_menu padre', 'padre.id=sis_menu.id_padre', 'left');
481         $this-> db-> join('sis_controlador', 'sis_controlador.id=sis_menu.id_controlador',
482         'left');
483
484         if( !empty($this-> filters) )
485             $this-> db-> where($this-> filters);
486
487         if(!empty($offset) && !empty(
488             $row_count))
489             $this-> db-> limit($offset, $row_count);
490         else if (!empty($offset))
491             $this-> db-> limit($offset);
492
493         $query = $this-> db-> get();
494         $result = $query-> result();
495
496         if($this-> db-> _error_number()) {
497             $this-> error = true;
498             $this-> msg_error_usr = 'No se encontraron registros en la busqueda';
499             $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
500             > _error_number().': '.$this-> db-> _error_message();
501             throw new Exception();
502         } else if(empty ($result)) {
503             $this-> msg_error_usr = 'No se encontraron registros en la busqueda';
504         }
505
506         return $result;
507     }
508
509 /**
510     * Obtiene el numero total de registros en la tabla Menu
511     * en caso de existir filtros, estos son aplicados a la consulta
512     *
513     * @access public
514     * @return int
515     */
516     public function getNumRows()
517     {
518         $result = 0;
519
520         $this-> db-> select('sis_menu.id, sis_menu.id_padre, sis_menu.id_raiz,
521         sis_menu.atributo, sis_menu.nombre, sis_menu.id_controlador,
522         sis_menu.ruta, raiz.nombre AS nombre_raiz, padre.nombre AS
523         nombre_padre, sis_controlador.nombre AS nombre_controlador');
524         $this-> db-> from('sis_menu');
525         $this-> db-> join('sis_menu raiz', 'raiz.id=sis_menu.id_raiz', 'left');
526         $this-> db-> join('sis_menu padre', 'padre.id=sis_menu.id_padre', 'left');
527         $this-> db-> join('sis_controlador', 'sis_controlador.id=sis_menu.id_controlador',

```

```

'left');

521     if( !empty($this-> filters) )
522         $this-> db-> where($this-> filters);
524
525     if(!empty($offset) && !empty($row_count))
526         $this-> db-> limit($offset, $row_count);
527     else if (!empty($offset))
528         $this-> db-> limit($offset);
529
530     $result = $this-> db-> count_all_results();
531
532     if($this-> db-> _error_number()) {
533         $this-> error = true;
534         $this-> msg_error_usr = 'No se encontraron registros en la busqueda';
535         $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
> _error_number().': '.$this-> db-> _error_message();
536         throw new Exception();
537     } else if(empty($result)) {
538         $this-> msg_error_usr = 'No se encontraron registros en la busqueda';
539     }
540
541     return $result;
542 }
543
544 /**
545 * Verifica si el nodo actual tiene hijos
546 *
547 * @access public
548 * @return boolean
549 */
550 public function hasChild($id) {
551     $query = $this-> db-> query('SELECT COUNT(id) as num_children FROM sis_menu WHERE
552 id_padre='.$id);
553     $result = $query-> row();
554
555     if($result-> num_children == 0)
556         return false;
557
558     return true;
559 }
560
561 /**
562 * Obtiene todos los nodos hijos de un padre
563 *
564 * @access public
565 * @return boolean
566 */
567 public function getByPadre($padre) {
568     $cond = '';
569
570     // Obtiene todas las raices
571     if($padre=='NULL' || $padre=='null' || $padre==0)
572         $cond = ' IS NULL';
573     else
574         $cond = ' = '.$padre;
575
576     $query = $this-> db-> query('SELECT * FROM sis_menu WHERE id_padre '.$cond);
577     $result = $query-> result();
578
579     return $result;
580 }
581
582 /*public function getMenuTree(){
583     $query = $this->db->query('SELECT
584         menu.id_padre,
585         menu_padre.nombre AS nombre_padre,
586         menu.id,
587         menu.nombre,
588         menu.nivel
589     FROM
590         menu
591     LEFT JOIN menu AS menu_raiz ON menu.id_raiz = menu_raiz.id
592     LEFT JOIN menu AS menu_padre ON menu.id_padre = menu_padre.id');
593
594     $result = $query->result_array();
595
596     return $result;
597 }*/

```

```
598    }
599 ?>
```

# Archivo fuente para permiso\_model.php

La documentación para este archivo está disponible en [permiso\\_model.php](#)

```
1  <?php
2  /**
3   * Modelo Permiso
4   *
5   * @package      SIIGS
6   * @subpackage   Modelo
7   * @author       Rogelio
8   * @created      2013-10-01
9   */
10  class Permiso_model extends CI_Model {
11      /**
12      * @access private
13      * @var int
14      */
15      private $id;
16      /**
17      * @access private
18      * @var int
19      */
20      private $id_grupo;
21      /**
22      * @access private
23      * @var datetime
24      */
25      private $fecha;
26      /**
27      * @access private
28      * @var int
29      */
30      private $id_controlador_accion;
31
32      /*****
33      * Estas variables no pertenecen a la tabla *
34      *****/
35
36      /**
37      * @access private
38      * @var string
39      */
40      private $msg_error_usr;
41      /**
42      * @access private
43      * @var string
44      */
45      private $msg_error_log;
46
47      public function __construct()
48      {
49          $this-> load-> database();
50          if (!$this-> db-> conn_id)
51              throw new Exception("No se pudo conectar a la base de datos");
52      }
53
54      public function getId()
55      {
56          return $this-> id;
57      }
58
59      public function setId($value) {
60          $this-> id = $value;
61      }
62
63      public function getIdGrupo()
64      {
65          return $this-> id_grupo;
66      }
67  }
```

```

68     public function setIdGrupo($id_grupo)
69     {
70         $this-> id_grupo = $id_grupo;
71     }
72
73     public function getFecha()
74     {
75         return $this-> fecha;
76     }
77
78     public function setFecha($fecha)
79     {
80         $this-> fecha = $fecha;
81     }
82
83     public function getIdControladorAccion()
84     {
85         return $this-> id_controlador_accion;
86     }
87
88     public function setIdControladorAccion($id_controlador_accion)
89     {
90         $this-> id_controlador_accion = $id_controlador_accion;
91     }
92
93     /**
94      * Asigna el mensaje de error a visualizar: para usuario final (usr) o para bitácora (log)
95      *
96      * @access public
97      * @param $value tipo de error a visualizar: usr o log, default: usr
98      * @return boolean false si ocurrió algún error, true si se ejecutó correctamente
99      */
100    public function getMsgError($value = 'usr')
101    {
102        if ($value == 'log')
103            return $this-> msg_error_log;
104        return $this-> msg_error_usr;
105    }
106
107   /**
108    * Obtiene el permiso solicitado
109    *
110    * @access public
111    * @param int $id id del controlador_x_accion
112    * @return void/object false si ocurrió algún error, object si se ejecutó
correctamente
113    */
114    public function getPermission($id)
115    {
116        // obtiene permiso por controlador_x_accion e id_grupo
117        $query = $this-> db-> get_where('sis_permiso', array('id_controlador_accion' =>
$id, 'id_grupo' => $this-> session-> userdata(GROUP_ID)));
118        if (!$query){
119            $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
120            $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
121            throw new Exception(__CLASS__);
122        }
123        else
124            return $query-> row();
125        return;
126    }
127
128   /**
129    * Inserta en la base de datos el arreglo de permisos recibido
130    *
131    * @access public
132    * @param array object $data array con los permisos a insertar
133    * @return boolean false si ocurrió algún error, true si se
ejecutó correctamente
134    */
135    public function insertBatch($data)
136    {
137        // insert_batch hace inserciones en bloques de 100
138        // y siempre retorna true =
139        $result = $this-> db-> insert_batch('sis_permiso', $data);
140        if (!$result){
141            $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
142            $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();

```

```

143             throw new Exception(__CLASS__);
144         }
145     }
146 }
147 /**
148 * Elimina de la base de datos los permisos del entorno y grupo recibidos
149 *
150 * @access public
151 * @param int $entorno id de entorno
152 * @param int $grupo id del grupo
153 * @return boolean false si ocurrió algún error, true si se
ejecutó correctamente
154 */
155 public function deletePermissions($entorno, $grupo)
156 {
157     $this-> db-> select('sis_permiso.id');
158     $this-> db-> from('sis_permiso');
159     $this-> db-> join('sis_controlador_x_accion', 'sis_controlador_x_accion.id =
sis_permiso.id_controlador_accion');
160     $this-> db-> join('sis_controlador', 'sis_controlador.id =
sis_controlador_x_accion.id_controlador');
161     $this-> db-> where('sis_permiso.id_grupo', $grupo);
162     $this-> db-> where('sis_controlador.id_entorno', $entorno);
163     $query = $this-> db-> get();
164     if (!$query){
165         $this-> msg_error_usr = "Servicio temporalmente no disponible.";
166         $this-> msg_error_log = "(" . __METHOD__ . ") => "
; $this-
> db-> _error_number().': '.$this-> db-> _error_message();
167         throw new Exception(__CLASS__);
168     }
169     else
170     {
171         $i = 0;
172         $aBorrar = array();
173         foreach ($query-> result() as $row)
174         {
175             $aBorrar[$i] = $row-> id;
176             $i++;
177         }
178         if (count($aBorrar) > 0)
179         {
180             $this-> db-> where_in('id', $aBorrar);
181             $result = $this-> db-> delete('sis_permiso');
182             if (!$result){
183                 $this-> msg_error_usr = "Servicio temporalmente no disponible.";
184                 $this-> msg_error_log = "(" . __METHOD__ . ") => "
; $this-> db-> _error_number().': '.$this-> db-> _error_message();
185                 throw new Exception(__CLASS__);
186             }
187             else
188                 return true;
189         }
190         else
191             return true;
192     }
193 }
194 return false;
195 }
196 }
197 }
198 ?>
```

# Archivo fuente para poblacion\_model.php

La documentación para este archivo está disponible en [poblacion\\_model.php](#)

```
1      <?php
2
3  /**
4   * Modelo Poblacion
5   *
6   * @package    SIIGS
7   * @subpackage Modelo
8   * @author     Pascual
9   * @created    2013-12-24
10  */
11 class Poblacion_model extends CI_Model
12 {
13     /*****
14     * Estas variables no pertenecen a la tabla *
15     * *****/
16
17     /**
18     * @access private
19     * @var boolean
20     */
21     private $error;
22
23     /**
24     * @access private
25     * @var string
26     */
27     private $msg_error_usr;
28
29     /**
30     * @access private
31     * @var string
32     */
33     private $msg_error_log;
34
35
36     public function __construct()
37     {
38         parent::__construct();
39         $this-> load-> database();
40         $this-> error = false;
41         $this-> msg_error_usr = '';
42         $this-> msg_error_log = '';
43
44         /*if( !$this->db->conn_id ) {
45             throw new Exception ('ERROR: No se puede conectar con la Base de Datos');
46         }*/
47     }
48
49     /**
50     * Devuelve el mensaje de error,
51     * en caso de existir un error despues de ejecutar un metodo,
52     * de lo contrario false
53     *
54     * @access public
55     * @param string $type usr si se quiere devolver el mensaje de error a mostrar en la vista,
56     *                      log obtiene el mensaje de error con mas detalles para depuración,
57     *                      valor por defecto usr
58     * @return boolean/string
59     */
60     public function getMsgError($type = 'usr')
61     {
62         if($this-> error) {
63             if($type == 'usr')
64                 return $this-> msg_error_usr;
65             else if($type == 'log')
66                 return $this-> msg_error_log;
67             else
68         }
```

```

68             return false;
69         }
70     }
71     return false;
72 }
73
74 /**
75 * Inserta los registros contenidos en la tabla cat_poblacion
76 * a la tablaasu_poblacion
77 *
78 * @access public
79 * @return boolean false Si no se ejecutó la inserción, true si se ejecutó la inserción
80 */
81 public function process()
82 {
83     $result = false;
84     $sql = 'REPLACE INTO
85             asu_poblacion (id_asu, id_grupo_etareo, ano, poblacion)
86             SELECT
87                 (SELECT b.id
88                  FROM cat_municipio a
89                  JOIN asu_arbol_segmentacion b
90                  ON a.id = b.id_tabla_original AND
91                      b.grado_segmentacion = 3 AND
92                          b.id_raiz = 1
93                  WHERE
94                      b.id_raiz=1 AND
95                      a.id_estado = cat_poblacion.id_estado AND
96                      a.id_jurisdiccion = cat_poblacion.id_jurisdiccion AND
97                      a.id_municipio = cat_poblacion.id_municipio) AS id_asu,
98
99                 (SELECT id
100                   FROM asu_grupo_etareo
101                   WHERE RTRIM(LTRIM(LOWER(descripcion))) =
102                         RTRIM(LTRIM(LOWER(cat_poblacion.grupo_etareo))) ) AS grupo_etareo,
103                         cat_poblacion.ano,
104                         cat_poblacion.poblacion
105                   FROM
106                     cat_poblacion';
107
108     $result = $this-> db-> query($sql);
109
110     if( $this-> db-> _error_number() ) {
111         $this-> error = true;
112         $this-> msg_error_usr = 'Error al procesar la tabla población';
113         $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
114 > _error_number().': '.$this-> db-> _error_message();
115         throw new Exception('('.__METHOD__.') => ' . $this-> db-> _error_number().':
116           '.$this-> db-> _error_message());
117     }
118
119     return $result;
120 }
121
122 }

```

# Archivo fuente para raiz\_model.php

La documentación para este archivo está disponible en [raiz\\_model.php](#)

```
1  <?php
2
3  /**
4   * Modelo Raiz
5   *
6   * @package    SIIGS
7   * @subpackage Modelo
8   * @author     Geovanni
9   * @created    2013-10-07
10  */
11 class Raiz_model extends CI_Model {
12
13     /**
14      * @access private
15      * @var int
16      */
17     private $id;
18
19     /**
20      * @access private
21      * @var string
22      */
23     private $descripcion;
24
25     /**
26      * @access private
27      * @var string
28      */
29     private $msg_error_log;
30
31     /**
32      * @access private
33      * @var string
34      */
35     private $msg_error_usr;
36
37     /*****
38     /*Getters and setters block*/
39     *****/
40     public function getId() {
41         return $this-> id;
42     }
43
44     public function setId($value) {
45         $this-> id = $value;
46     }
47
48     public function getDescripcion() {
49         return $this-> descripcion;
50     }
51
52     public function setDescripcion($value) {
53         $this-> descripcion = $value;
54     }
55
56     /*****
57     /*Getters and setters block END*/
58     *****/
59
60     /**
61      * Devuelve los mensajes de error en caso de ocurrir alguna excepción
62      * 'usr' devuelve el mensaje para la vista de usuario
63      * 'log' devuelve el mensaje para el log de errores
64      *
65      * @access public
66      * @return string|boolean
67      * @param string $value, default 'usr' (Tipo mensaje)
```

```

68      */
69      public function getMsgError($value = 'usr')
70      {
71          if (!empty($this->msg_error_usr))
72          {
73              if ($value == 'usr')
74                  return $this->msg_error_usr;
75              else if ($value == 'log')
76                  return $this->msg_error_log;
77          }
78          else
79          {
80              return false;
81          }
82      }
83
84      public function __construct()
85      {
86          $this->load->database();
87          if (!$this->db->conn_id)
88          {
89              throw new Exception("No se pudo conectar a la base de datos");
90          }
91      }
92
93      /**
94      * Devuelve todos los registros de la tabla raiz
95      *
96      * @access public
97      * @return ArrayObject
98      * @throws Exception En caso de algun error al consultar la base de datos
99      */
100     public function getAll()
101    {
102        $query = $this->db->query("select a.*, sum(case when ifnull(b.id,'') = '' then 0 else 1 end) as catalogos fromasu_raiz a left outer joinasu_raiz_x_catalogo b on a.id = b.id_raiz_arbol group by a.id");
103
104        if (!$query)
105        {
106            $this->msg_error_log = "(" . __METHOD__ . ".{$this->db->_error_message()}) => " . $this->_error_number().":'".$this->db->_error_message();
107            $this->msg_error_usr = "Ocurrió un error al obtener los datos de raices";
108            throw new Exception(__CLASS__);
109        }
110        else
111            return $query->result();
112    }
113
114    /**
115     * Revisa si la raiz pasada como parametro existe en el ASU
116     *
117     * @access public
118     * @param int $id
119     * @return Boolean
120     * @throws Exception En caso de algun error al consultar la base de datos
121     */
122     public function ExistInArbol($id)
123    {
124        $query = $this->db->query('select * fromasu_arbol_segmentacion where id_raiz=' . $id);
125
126        if (!$query)
127        {
128            $this->msg_error_log = "(" . __METHOD__ . ".{$this->db->_error_message()}) => " . $this->_error_number().":'".$this->db->_error_message();
129            $this->msg_error_usr = "Ocurrió un error al obtener los datos de raices";
130            throw new Exception(__CLASS__);
131        }
132        else
133            return ($query->num_rows() == 0 ? false : true);
134    }
135
136    /**
137     * Devuelve la informacion de una raiz por su ID
138     *
139     * @access public
140     * @return Object

```

```

141     * @param int $id ID (Llave primaria)
142     * @throws Exception En caso de algun error al consultar la base de datos
143     */
144    public function getById($id)
145    {
146        $query = $this-> db-> get_where('asu_raiz', array('id' => $id));
147
148        if (!$query)
149        {
150            $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
151 > db-> _error_number(). ': ' . $this-> db-> _error_message();
152            $this-> msg_error_usr = "Ocurrió un error al obtener la información de la
153 raiз";
154            throw new Exception(__CLASS__);
155        }
156        else
157            return $query-> row();
158    }
159
160 /**
161 * Inserta en la tabla raiz, la información contenida en el objeto
162 *
163 * @access public
164 * @return int (Id de la inserción si no hubo errores al actualizar)
165 * @throws Exception En caso de algun error al consultar la base de datos
166 */
167 public function insert()
168 {
169     $data = array(
170         'descripcion' => $this-> descripcion
171     );
172
173     $query = $this-> db-> insert('asu_raiz', $data);
174
175     if (!$query)
176     {
177         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
178 > db-> _error_number(). ': ' . $this-> db-> _error_message();
179         $this-> msg_error_usr = "Ocurrió un error al insertar la raiz";
180         throw new Exception(__CLASS__);
181     }
182
183 /**
184 * Actualiza el objeto actual en la base de datos
185 *
186 * @access public
187 * @return boolean (Si no hubo errores al actualizar)
188 * @throws Exception En caso de algun error al consultar la base de datos
189 */
190 public function update()
191 {
192     $data = array(
193         'descripcion' => $this-> descripcion
194     );
195
196     $this-> db-> where('id', $this-> getId());
197     $query = $this-> db-> update('asu_raiz', $data);
198
199     if (!$query)
200     {
201         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
202 > db-> _error_number(). ': ' . $this-> db-> _error_message();
203         $this-> msg_error_usr = "Ocurrió un error al actualizar los datos de la
204 raiз";
205         throw new Exception(__CLASS__);
206     }
207     else
208         return true;
209 }
210
211 /**
212 * Elimina el registro actual de la base de datos
213 *
214 * @access public
215 * @return boolean (Si no hubo errores al eliminar)
216 * @throws Exception En caso de algun error al consultar la base de datos
217 */

```

```

216     public function delete()
217     {
218
219         $asu = $this-> db-> delete('asu_arbol_segmentacion', array('id_raiz' =>
220 $this-> getId()));
221         $query = $this-> db-> delete('asu_raiz', array('id' => $this-> getId()));
222
223         if (!$query || !$asu)
224         {
225             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
226 > db-> _error_number(). ': ' . $this-> db-> _error_message();
227             $this-> msg_error_usr = "Ocurrió un error al eliminar la raiz";
228             throw new Exception(__CLASS__);
229         }
230         else
231             return true;
232     }
233 }
```

# Archivo fuente para reglavaduna\_model.php

*La documentación para este archivo está disponible en [reglavaduna\\_model.php](#)*

```
1  <?php
2
3  /**
4   * Modelo ReglaVacuna
5   *
6   * @package      SIIGS
7   * @subpackage   Modelo
8   * @author       Geovanni
9   * @created      2013-12-09
10  */
11 class ReglaVacuna_model extends CI_Model {
12
13     /**
14      * @access private
15      * @var    int
16      */
17     private $id;
18
19     /**
20      * @access private
21      * @var    int
22      */
23     private $id_vacuna;
24
25     /**
26      * @access private
27      * @var    int
28      */
29     private $id_vacuna_previa;
30
31     /**
32      * @access private
33      * @var    int
34      */
35     private $dia_inicio_nacido;
36
37     /**
38      * @access private
39      * @var    int
40      */
41     private $dia_fin_nacido;
42
43     /**
44      * @access private
45      * @var    int
46      */
47     private $dia_inicio_previa;
48
49     /**
50      * @access private
51      * @var    int
52      */
53     private $dia_fin_previa;
54
55     /**
56      * @access private
57      * @var    int
58      */
59     private $id_via_vacuna;
60
61         /**
62          * @access private
63          * @var    float
64          */
```

```

65     private $dosis;
66
67     /**
68      * @access private
69      * @var string
70      */
71     private $region;
72
73     /**
74      * @access private
75      * @var string
76      */
77     private $observacion_region;
78
79     /**
80      * @access private
81      * @var bool
82      */
83     private $esq_com;
84
85     /**
86      * @access private
87      * @var int
88      */
89     private $orden_esq_comp;
90
91     /**
92      * @access private
93      * @var array(int)
94      */
95     private $alergias;
96
97     /**
98      * @access private
99      * @var boolean
100     */
101    private $forzar_aplicacion;
102
103    /**
104      * @access private
105      * @var string
106      */
107    private $msg_error_log;
108
109    /**
110      * @access private
111      * @var string
112      */
113    private $msg_error_usr;
114
115    ****
116    /*Getters and setters block*/
117    ****
118    public function getId() {
119        return $this-> id;
120    }
121    public function setId($value) {
122        $this-> id = $value;
123    }
124
125    public function getIdVacuna() {
126        return $this-> id_vacuna;
127    }
128    public function setIdVacuna($value) {
129        $this-> id_vacuna = $value;
130    }
131
132    public function getIdVacunaPrevia() {
133        return $this-> id_vacuna_previa;
134    }
135    public function setIdVacunaPrevia($value) {
136        $this-> id_vacuna_previa = $value;
137    }
138
139    public function getDiaInicioNacido() {
140        return $this-> dia_inicio_nacido;
141    }
142    public function setDiaInicioNacido($value) {
143        $this-> dia_inicio_nacido = $value;
144    }

```

```

145     public function getDiaFinNacido() {
146         return $this-> dia_fin_nacido;
147     }
148
149     public function setDiaFinNacido($value) {
150         $this-> dia_fin_nacido = $value;
151     }
152
153     public function getDiaInicioPrevia() {
154         return $this-> dia_inicio_previa;
155     }
156
157     public function setDiaInicioPrevia($value) {
158         $this-> dia_inicio_previa = $value;
159     }
160
161     public function getDiaFinPrevia() {
162         return $this-> dia_fin_previa;
163     }
164
165     public function setDiaFinPrevia($value) {
166         $this-> dia_fin_previa = $value;
167     }
168
169     public function getIdViaVacuna() {
170         return $this-> id_via_vacuna;
171     }
172
173     public function setIdViaVacuna($value) {
174         $this-> id_via_vacuna = $value;
175     }
176
177     public function getDosis() {
178         return $this-> dosis;
179     }
180
181     public function setDosis($value) {
182         $this-> dosis = $value;
183     }
184
185     public function getRegion() {
186         return $this-> region;
187     }
188
189     public function setRegion($value) {
190         $this-> region = $value;
191     }
192
193     public function getObservacionRegion() {
194         return $this-> observacion_region;
195     }
196
197     public function setObservacionRegion($value) {
198         $this-> observacion_region = $value;
199     }
200
201     public function getEsqComp() {
202         return $this-> esq_com;
203     }
204
205     public function setEsqComp($value) {
206         $this-> esq_com = $value;
207     }
208
209     public function getOrdenEsqComp() {
210         return $this-> orden_esq_comp;
211     }
212
213     public function setOrdenEsqComp($value) {
214         $this-> orden_esq_comp = $value;
215     }
216
217     public function getForzarAplicacion() {
218         return $this-> forzar_aplicacion;
219     }
220
221     public function setForzarAplicacion($value) {
222         $this-> forzar_aplicacion = $value;
223
224     /**
225      * Getters and setters block END*/

```

```

225     ****
226
227     /**
228      * Devuelve los mensajes de error en caso de ocurrir alguna excepción
229      * 'usr' devuelve el mensaje para la vista de usuario
230      * 'log' devuelve el mensaje para el log de errores
231      *
232      * @access public
233      * @return string|boolean
234      * @param string $value, default 'usr' (Tipo mensaje)
235      */
236     public function getMsgError($value = 'usr')
237     {
238         if (!empty($this->msg_error_usr))
239         {
240             if ($value == 'usr')
241                 return $this->msg_error_usr;
242             else if ($value == 'log')
243                 return $this->msg_error_log;
244         }
245         else
246         {
247             return false;
248         }
249     }
250
251     public function __construct()
252     {
253         $this->load->database();
254         if (!$this->db->conn_id)
255         {
256             throw new Exception("No se pudo conectar a la base de datos");
257         }
258     }
259
260     /**
261      * Devuelve todos los registros de la tabla regla_vacuna
262      *
263      * @access public
264      * @return ArrayObject
265      * @throws Exception En caso de algun error al consultar la base de datos
266      */
267     public function getAll()
268     {
269         $query = $this->db->query("SELECT distinct a.id,b.descripcion as vacuna,a.id_vacuna_secuencial , CASE WHEN IFNULL(a.dia_inicio_aplicacion_nacido,'') = '' THEN 'Secuencial' ELSE 'Nacimiento' END AS aplicacion , case when ifnull(a.dia_inicio_aplicacion_nacido,'') = '' then a.dia_inicio_aplicacion_secuencial else a.dia_inicio_aplicacion_nacido end as desde , case when ifnull(a.dia_inicio_aplicacion_nacido,'') = '' then a.dia_fin_aplicacion_secuencial else a.dia_fin_aplicacion_nacido end as hasta , case when ifnull(a.id_vacuna_secuencial,'') = '' then 'Ninguna' else c.descripcion end as previa, case when ifnull(a.id_via_vacuna,'') = '' then '' else d.descripcion end as via_vacuna, a.dosis as dosis, a.region as region, a.esq_com, a.orden_esq_com, a.alergias,a.observacion_region FROM cns_regla_vacuna a join cns_vacuna b on a.id_vacuna = b.id and b.activo = 1 left outer join cns_vacuna c on a.id_vacuna_secuencial = c.id and c.activo = 1 left outer join cns_via_vacuna d on a.id_via_vacuna = d.id");
270         if (!$query)
271         {
272             $this->msg_error_log = "(" . __METHOD__ . ".{$this->db->error_message()}) => " . $this->_error_number().":'." . $this->db->_error_message();
273             $this->msg_error_usr = "Ocurrió un error al obtener los datos de regla";
274             throw new Exception(__CLASS__);
275         }
276         else
277             return $query->result();
278     }
279
280     /**
281      * Devuelve la información de una regla de vacuna por su ID
282      *
283      * @access public
284      * @return Object
285      * @param int $id ID (Llave primaria)
286      * @throws Exception En caso de algun error al consultar la base de datos
287      */
288     public function getById($id)
289     {
290         $query = $this->db->query("SELECT distinct a.id,a.id_vacuna,a.id_via_vacuna,a.id_vacuna_secuencial,b.descripcion as vacuna , CASE WHEN IFNULL(a.dia_inicio_aplicacion_nacido,'') = '' THEN 'Secuencial' ELSE 'Nacimiento' END AS aplicacion

```

```

, case when ifnull(a.dia_inicio_aplicacion_nacido,'') = '' then a.dia_inicio_aplicacion_secuencial
else a.dia_inicio_aplicacion_nacido end as desde , case when
ifnull(a.dia_inicio_aplicacion_nacido,'') = '' then a.dia_fin_aplicacion_secuencial else
a.dia_fin_aplicacion_nacido end as hasta , case when ifnull(a.id_vacuna_secuencial,'') = '' then
'Ninguna' else c.descripcion end as previa , a.dia_inicio_aplicacion_secuencial as desdese,
a.dia_fin_aplicacion_secuencial as hastase, case when ifnull(a.id_via_vacuna,'') = '' then '' else
d.descripcion end as via_vacuna, a.dosis as dosis, a.region as region, a.esq_com, a.orden_esq_com,
a.alergias as id_alergias,a.forzar_aplicacion, a.observacion_region FROM cns_regla_vacuna a join
cns_vacuna b on a.id_vacuna = b.id and b.activo = 1 left outer join cns_vacuna c on
a.id_vacuna_secuencial = c.id and c.activo = 1 left outer join cns_via_vacuna d on a.id_via_vacuna =
d.id where a.id=" .$id);
291
292         if (!$query)
293     {
294         $this-> msg_error_log = " ( " . __METHOD__ . " ) => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
295         $this-> msg_error_usr = "Ocurrió un error al obtener la información de la
regla" ;
296         throw new Exception(__CLASS__);
297     }
298     else
299     {
300         $info = $query-> row();
301         //var_dump($info);
302         $descalergias = '';
303         if(!empty($info-> id_alergias))
304             if (count($info-> id_alergias)> 0)
305             {
306                 $infoalergias = $this-> db-> query("select * from cns_alergia where id
in (" . $info-> id_alergias." )");
307                 if ($infoalergias)
308                     if (count($infoalergias-> result())> 0)
309                     {
310                         foreach($infoalergias-> result() as $infoalergia)
311                             $descalergias.= $infoalergia-> descripcion.", "
312                             $descalergias = substr($descalergias, 0, count($descalergias)-3);
313                     }
314                 }
315                 $info-> alergias = $descalergias;
316                 return $info;
317             }
318         }
319
320 /**
321 *Inserta en la tabla regla_vacuna la información contenida en el objeto
322 */
323 *@access public
324 *@return int (Id de la inserción si no hubo errores al actualizar)
325 * @throws Exception En caso de algun error al consultar la base de datos
326 */
327 public function insert()
328 {
329     $data = array(
330         'id_vacuna' => $this-> id_vacuna,
331         'id_vacuna_secuencial' => $this-> id_vacuna_previa,
332         'dia_inicio_aplicacion_nacido' => ( $this-> dia_inicio_nacido == 0) ? 0 :
$this-> dia_inicio_nacido,
333         'dia_fin_aplicacion_nacido' => ( $this-> dia_fin_nacido == 0) ? 0 : $this-
> dia_fin_nacido,
334         'dia_inicio_aplicacion_secuencial' => ( $this-
> dia_inicio_previa == 0) ? null : $this-> dia_inicio_previa,
335         'dia_fin_aplicacion_secuencial' => ( $this-> dia_fin_previa ==
0) ? null : $this-> dia_fin_previa,
336         'id_via_vacuna' => $this-> id_via_vacuna,
337         'dosis' => $this-> dosis,
338         'region' => $this-> region,
339         'alergias' => (!empty( $this-> alergias)) ? $this-
> alergias : null,
340         'esq_com' => $this-> esq_com,
341         'forzar_aplicacion' => $this-> forzar_aplicacion,
342         'observacion_region' => $this-> observacion_region,
343         'ultima_actualizacion' => date('Y-m-d H:i:s')
344     );
345
346     $data['orden_esq_com'] = ($this-> esq_com) ? $this-> orden_esq_comp : null;
347
348     $query = $this-> db-> insert('cns_regla_vacuna', $data);
349
350     if (!$query)
351     {

```

```

352             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
353             $this-> msg_error_usr = "Ocurrió un error al insertar la regla";
354             throw new Exception(__CLASS__);
355         }
356     }
357     {
358         $this-> db-> query("update cns_tabla_catalogo set
fecha_actualizacion = NOW() where descripcion='cns_regla_vacuna' ");
359         return $this-> db-> insert_id($query);
360     }
361 }
362 /**
363 *Actualiza el objeto actual en la base de datos
364 *
365 *@access public
366 *@return boolean (Si no hubo errores al actualizar)
367 * @throws Exception En caso de algun error al consultar la base de datos
368 */
369 public function update()
370 {
371     $data = array(
372         'id_vacuna' => $this-> id_vacuna,
373         'id_vacuna_secuencial' => $this-> id_vacuna_previa,
374         'dia_inicio_aplicacion_nacido' => ( $this-> dia_inicio_nacido == 0 ) ? 0 :
375 $this-> dia_inicio_nacido,
376         'dia_fin_aplicacion_nacido' => ( $this-> dia_fin_nacido == 0 ) ? 0 : $this-
> dia_fin_nacido,
377         'dia_inicio_aplicacion_secuencial' => ( $this-
> dia_inicio_previa == 0 ) ? null : $this-> dia_inicio_previa,
378         'dia_fin_aplicacion_secuencial' => ( $this-> dia_fin_previa ==
0 ) ? null : $this-> dia_fin_previa,
379         'id_via_vacuna' => $this-> id_via_vacuna,
380         'dosis' => $this-> dosis,
381         'region' => $this-> region,
382         'alergias' => (!empty( $this-> alergias)) ? $this-
> alergias : null,
383         'esq_com' => $this-> esq_com,
384         'forzar_aplicacion' => $this-> forzar_aplicacion,
385         'observacion_region' => $this-> observacion_region,
386         'ultima_actualizacion' => date('Y-m-d H:i:s')
387     );
388
389     $data['orden_esq_com'] = ($this-> esq_com) ? $this-> orden_esq_comp : null;
390
391     $this-> db-> where('id' , $this-> getId());
392     $query = $this-> db-> update('cns_regla_vacuna' , $data);
393
394     if (!$query)
395     {
396         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
397         $this-> msg_error_usr = "Ocurrió un error al actualizar los datos de la
regla";
398         throw new Exception(__CLASS__);
399     }
400     else
401     {
402         $this-> db-> query("update cns_tabla_catalogo set
fecha_actualizacion = NOW() where descripcion='cns_regla_vacuna' ");
403         return true;
404     }
405 }
406 /**
407 * Elimina el registro actual de la base de datos
408 *
409 * @access public
410 * @return boolean (Si no hubo errores al eliminar)
411 * @throws Exception En caso de algun error al consultar la base de datos
412 */
413 public function delete()
414 {
415
416     $query = $this-> db-> delete('cns_regla_vacuna' , array('id' => $this-
> getId()));
417
418     if (!$query)
419     {

```

```

421           $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number(). ': ' . $this-> db-> _error_message();
422           $this-> msg_error_usr = "Ocurrió un error al eliminar la regla";
423           throw new Exception(__CLASS__);
424       }
425   }
426   {
427       $this-> db-> query("update cns_tabla_catalogo set
fecha_actualizacion = NOW() where descripcion='cns_regla_vacuna' ");
428       return true;
429   }
430 }
431 }
```

# Archivo fuente para usuario\_model.php

La documentación para este archivo está disponible en [usuario\\_model.php](#)

```
1  <?php
2  /**
3   * Modelo Usuario
4   *
5   * @package    SIIGS
6   * @subpackage Modelo
7   * @author     Rogelio
8   * @created    2013-09-25
9   */
10  class Usuario_model extends CI_Model {
11      /**
12      * Guarda la instancia del objeto global CodeIgniter
13      * para utilizarlo en la función estática
14      *
15      * @access private
16      * @var    instance
17      */
18      private static $CI;
19
20      /**
21      * @access private
22      * @var    int
23      */
24      private $id;
25
26      /**
27      * @access private
28      * @var    string
29      */
30      private $nombre_usuario;
31
32      /**
33      * @access private
34      * @var    string
35      */
36      private $clave;
37
38      /**
39      * @access private
40      * @var    string
41      */
42      private $apellido_paterno;
43
44      /**
45      * @access private
46      * @var    string
47      */
48      private $apellido_materno;
49
50      /**
51      * @access private
52      * @var    string
53      */
54      private $correo;
55
56      /**
57      * @access private
58      * @var    boolean
59      */
60      private $activo;
61
62      /**
63      * @access private
64      * @var    int
65      */
66      private $id_grupo;
67
68      /**
69      * Estas variables no pertenecen a la tabla *
70      */
```

```

68 * ****
69 */
70 /**
71 * @access private
72 * @var string
73 */
74 private $msg_error_usr;
75 /**
76 * @access private
77 * @var string
78 */
79 private $msg_error_log;
80
81 public function __construct()
82 {
83     parent::__construct();
84
85     self::$CI = & get_instance();
86
87     $this-> load-> database();
88     if (!$this-> db-> conn_id)
89         throw new Exception("No se pudo conectar a la base de datos");
90 }
91
92 public function getId()
93 {
94     return $this-> id;
95 }
96
97 public function setId($value) {
98     $this-> id = $value;
99 }
100
101 public function getNombreUsuario()
102 {
103     return $this-> nombre_usuario;
104 }
105
106 public function setNombreUsuario($nombre_usuario)
107 {
108     $this-> nombre_usuario = $nombre_usuario;
109 }
110
111 public function getClave()
112 {
113     return $this-> clave;
114 }
115
116 public function setClave($clave)
117 {
118     $this-> clave = $clave;
119 }
120
121 public function getNombre()
122 {
123     return $this-> nombre;
124 }
125
126 public function setNombre($nombre)
127 {
128     $this-> nombre = $nombre;
129 }
130
131 public function getApellidoPaterno()
132 {
133     return $this-> apellido_paterno;
134 }
135
136 public function setApellidoPaterno($apellido_paterno)
137 {
138     $this-> apellido_paterno = $apellido_paterno;
139 }
140
141 public function getApellidoMaterno()
142 {
143     return $this-> apellido_materno;
144 }
145
146 public function setApellidoMaterno($apellido_materno)
147 {

```

```

148     $this-> apellido_materno = $apellido_materno;
149 }
150
151 public function getCorreo()
152 {
153     return $this-> correo;
154 }
155
156 public function setCorreo($correo)
157 {
158     $this-> correo = $correo;
159 }
160
161 public function getActivo()
162 {
163     return $this-> activo;
164 }
165
166 public function setActivo($activo)
167 {
168     $this-> activo = $activo;
169 }
170
171 public function getIdGrupo()
172 {
173     return $this-> id_grupo;
174 }
175
176 public function setIdGrupo($id_grupo)
177 {
178     $this-> id_grupo = $id_grupo;
179 }
180
181 /**
182 * Asigna el mensaje de error a visualizar: para usuario final (usr) o para bitácora (log)
183 *
184 * @access public
185 * @param string $value tipo de error a visualizar: usr o log,
186 * @return boolean false si ocurrió algún error, true si se ejecutó
187 * correctamente
188 */
189 public function getMsgError($value = 'usr')
190 {
191     if ($value == 'log')
192         return $this-> msg_error_log;
193     return $this-> msg_error_usr;
194 }
195 /**
196 * Obtiene todos los usuarios existentes, se puede filtrar por: texto a buscar o solo
197 * activos si se desea
198 *
199 * @access public
200 * @param boolean/string $keywords false no hay texto a buscar/string con
201 * texto a buscar
202 * @param int $onlyActives true obtiene solo usuarios activos,
203 * @param int $offset Establece el desplazamiento del
204 * primer registro a devolver,
205 * @param int $row_count Establece la cantidad de registros a
206 * devolver
207 * @return void/arrayobject false si ocurrió algún error, array
208 * object si se ejecutó correctamente
209 */
210 public function getOnlyActives($keywords = '', $onlyActives = TRUE, $offset = null,
211 $row_count = null)
212 {
213     if (!empty($offset) && !empty($row_count))
214         $this-> db-> limit($offset, $row_count);
215     else if (!empty($offset))
216         $this-> db-> limit($offset);
217     if ($onlyActives === TRUE)
218     {
219         $query = $this-> db-> get_where('sis_usuario', array('activo' =>
220 $onlyActives));
221     }

```

```

217     else
218     {
219         if (empty($keywords)){
220
221             $query = $this-> db-> get('sis_usuario');
222
223         else
224         {
225             $this-> db-> select('*');
226             $this-> db-> from('sis_usuario');
227             $this-> db-> like('nombre_usuario', $keywords);
228             $this-> db-> or_like('nombre', $keywords);
229             $this-> db-> or_like('apellido_paterno', $keywords);
230             $this-> db-> or_like('apellido_materno', $keywords);
231             $query = $this-> db-> get();
232         }
233
234         if (!$query){
235             $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
236             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
237             throw new Exception(__CLASS__);
238         }
239         else
240             return $query-> result();
241         return;
242     }
243
244 /**
245 * Obtiene el usuario solicitado, se puede obtener el registro normal o personalizado para
246 * visualización (descripciones
247 *
248 * @access public
249 * @param int $id id del usuario
250 * @param boolean $viewMode true obtiene el modo visualización, false o
251 * null obtiene el registro normal
252 * @return void/object false si ocurrió algún error, object si se ejecutó
253 * correctamente
254 */
255 public function getById($id, $viewMode = FALSE)
256 {
257     if (!$viewMode)
258         $query = $this-> db-> get_where('sis_usuario', array('id' => $id));
259     else
260     {
261         $this-> db-> select('sis_usuario.* , sis_grupo.nombre as Grupo');
262         $this-> db-> from('sis_usuario');
263         $this-> db-> join('sis_grupo', 'sis_grupo.id = sis_usuario.id_grupo');
264         $this-> db-> where('sis_usuario.id', $id);
265         $query = $this-> db-> get();
266
267         if (!$query){
268             $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
269             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
270             throw new Exception(__CLASS__);
271         }
272         else
273             return $query-> row();
274         return;
275     }
276
277 /**
278 * Obtiene los usuarios activos del grupo solicitado
279 *
280 * @access public
281 * @param int $id id del usuario
282 * @param boolean $viewMode true obtiene el modo visualización, false o
283 * null obtiene el registro normal
284 * @return void/object false si ocurrió algún error, object si se ejecutó
285 * correctamente
286 */
287 public function getActivesByGroup($group_id)
288 {
289     $query = $this-> db-> get_where('sis_usuario', array('id_grupo' => $group_id,
290 'activo' => 1));
291
292     if (!$query){
293         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
294         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-

```

```

>   db-> _error_number().' : '.$this-> db-> _error_message();
289     throw new Exception(__CLASS__);
290   }
291   else
292     return $query-> result();
293   return;
294 }
295 /**
296 * Obtiene el usuario solicitado
297 *
298 * @access public
299 * @param string $username nombre de usuario
300 * @return void/object false si ocurrió algún error, object si se ejecutó
correctamente
301 */
302 public function getByUsername($username)
303 {
  $query = $this-> db-> get_where('sis_usuario', array('nombre_usuario' =>
$username));
305   if (!$query){
306     $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
307     $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().' : '.$this-> db-> _error_message();
309     throw new Exception(__CLASS__);
310   }
311   else
312     return $query-> row();
313   return;
314 }
315 /**
316 * Obtiene el numero total de usuarios
317 *
318 * @access public
319 * @param boolean/string $keywords false no hay texto a buscar/string con
texto a buscar
320 * @return int
321 */
322 public function getNumRows($keywords = '')
323 {
  if (!$keywords)
    $query = $this-> db-> get('sis_usuario');
  else
  {
    $this-> db-> select('*');
    $this-> db-> from('sis_usuario');
    $this-> db-> like('nombre_usuario', $keywords);
    $this-> db-> or_like('nombre', $keywords);
    $this-> db-> or_like('apellido_paterno', $keywords);
    $this-> db-> or_like('apellido_materno', $keywords);
    $query = $this-> db-> get();
  }
336   if(!$query) {
338     $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
339     $this-> msg_error_log = '(' . __METHOD__ . ') => ' . $this-> db-
> _error_number().' : '.$this-> db-> _error_message();
340     throw new Exception(__CLASS__);
341   }
342   return $query-> num_rows;
343 }
344 /**
345 * Inserta en la base de datos los datos del usuario (datos en propiedades)
346 *
347 * @access public
348 * @return boolean false si ocurrió algún error, true si se
ejecutó correctamente
349 */
350 public function insert()
351 {
  $data = array(
    'nombre_usuario' => $this-> nombre_usuario,
    'clave' => $this-> clave,
    'nombre' => $this-> nombre,
    'apellido_paterno' => $this-> apellido_paterno,
    'apellido_materno' => $this-> apellido_materno,
    'correo' => $this-> correo,
    'activo' => $this-> activo,
    'id_grupo' => $this-> id_grupo
  )

```

```

362     );
363     $result = $this-> db-> insert('sis_usuario', $data);
364     if (!$result){
365         $this-> msg_error_usr = "Servicio temporalmente no disponible.";
366         $this-> msg_error_log = "(" . __METHOD__ . ") => ";
367     > db-> _error_number(): '$this-> db-> _error_message();';
368         throw new Exception(__CLASS__);
369     }
370     return $result;
371 }
372 /**
373 * Actualiza en la base de datos los datos del usuario (datos en propiedades)
374 *
375 * @access public
376 * @return boolean false si ocurrió algún error, true si se ejecutó correctamente
377 */
378 public function update()
379 {
380     $data = array(
381         'nombre' => $this-> nombre,
382         'apellido_paterno' => $this-> apellido_paterno,
383         'apellido_materno' => $this-> apellido_materno,
384         'correo' => $this-> correo,
385         'activo' => $this-> activo,
386         'id_grupo' => $this-> id_grupo
387     );
388     $this-> db-> where('id', $this-> id);
389     $result = $this-> db-> update('sis_usuario', $data);
390     if (!$result){
391         $this-> msg_error_usr = "Servicio temporalmente no disponible.";
392         $this-> msg_error_log = "(" . __METHOD__ . ") => ";
393     > db-> _error_number(): '$this-> db-> _error_message();';
394         throw new Exception(__CLASS__);
395     }
396     return $result;
397 }
398 /**
399 * Elimina de la base de datos al usuario (id en propiedades)
400 *
401 * @access public
402 * @return boolean false si ocurrió algún error, true si se ejecutó correctamente
403 */
404 public function delete()
405 {
406     $result = $this-> db-> delete('sis_usuario', array('id' => $this-> getId()));
407     if (!$result){
408         $this-> msg_error_usr = "Servicio temporalmente no disponible.";
409         $this-> msg_error_log = "(" . __METHOD__ . ") => ";
410     > db-> _error_number(): '$this-> db-> _error_message();';
411         throw new Exception(__CLASS__);
412     }
413     return $result;
414 }
415 /**
416 * Valida las credenciales recibidas
417 *
418 * @access public
419 * @param string $username nombre de usuario
420 * @param string $password clave
421 * @return null/object null si ocurrió algún error, object si se ejecutó correctamente
422 */
423 public function authenticate($username, $password)
424 {
425     $query = $this-> db-> get_where('sis_usuario', array('nombre_usuario' =>
426 $username, 'clave' => $password));
427     if (!$query){
428         $this-> msg_error_usr = "Servicio temporalmente no disponible.";
429         $this-> msg_error_log = "(" . __METHOD__ . ") => ";
430     > db-> _error_number(): '$this-> db-> _error_message();';
431         throw new Exception(__CLASS__);
432     }
433     else
434         return $query-> row();
435     return null;
436 }

```

```

435
436     /**
437      * Verifica que el usuario haya iniciado sesión y además tenga permiso en la acción recibida
438      *
439      * @access public
440      * @param string      $path          entorno::controlador::accion
441      * @param int         $group_id      id del grupo a validar permisos
442      * @return void
443      */
444     public static function checkCredentials($path, $pathURL)
445     {
446         self::$CI-> load-> helper('url');
447         if (!self::$CI-> session-> userdata(GROUP_ID)) // si no esta logueado lo debe
mandar al login
448         {
449             self::$CI-> session-> set_userdata(REDIRECT_TO, $pathURL);
450             redirect(DIR_SIIGS.'/usuario/login', 'refresh');
451         }
452         self::$CI-> load-> model(DIR_SIIGS.'/ControladorAccion_model');
453         try{
454             // Obtener el id_controlador_accion a partir del id_controlador y el id_accion
455             $id_controlador_accion = self::$CI-> ControladorAccion_model-
> getIdByPath($path);
456             if($id_controlador_accion == 0) {
457                 log_message('error', '(Usuario_model::checkCredentials) No se encuentra la
relación entre el controlador y la acción: '.$path.', Error '.self::$CI-> db-> _error_number().':'.
'.self::$CI-> db-> _error_message());
458                 return false;
459             }
460             self::$CI-> load-> model(DIR_SIIGS.'/Permiso_model');
461             // Obtener permisos sobre la acción recibida
462             $row = self::$CI-> Permiso_model-> getPermission($id_controlador_accion);
463             if ($row)
464                 return true;
465             Bitacora_model::insert($path, 'Acceso denegado a: '.self::$CI-> session-
> userdata(USERNAME));
466         }
467         catch(Exception $e) {
468             log_message('error', '(Usuario_model::checkCredentials) Error al obtener permisos:
'.self::$CI-> session-> userdata(USERNAME).', Path: '.$path.', Error '.self::$CI-> db-
> _error_number().': '.self::$CI-> db-> _error_message());
469         }
470         return false;
471     }
472
473
474
475     ////////// recuperar contraseña
476
477     public function check_data($usuario,$correo)
478     {
479         $query = $this-> db-> get_where('sis_usuario', array('nombre_usuario' =>
$usuario, 'correo' => $correo));
480         if (!$query)
481         {
482             $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
483             $this-> msg_error_log = "(" . __METHOD__ . ") => " .
.$this-
> db-> _error_number().': '.$this-> db-> _error_message();
484             throw new Exception(__CLASS__);
485         }
486         else
487             return $query-> row();
488         return null;
489     }
490     public function check_token($correo)
491     {
492         $query = $this-> db-> get_where('sis_usuario', array('correo' => $correo));
493         if (!$query)
494         {
495             $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
496             $this-> msg_error_log = "(" . __METHOD__ . ") => " .
.$this-
> db-> _error_number().': '.$this-> db-> _error_message();
497             throw new Exception(__CLASS__);
498         }
499         else
500             return $query-> row();
501         return null;
502     }
503     public function update_pass($pass,$id)
504     {

```

```

505     $data = array('clave' =>      $pass);
506     $this-> db-> where('id' , $id);
507     $result = $this-> db-> update('sis_usuario', $data);
508     if (!$result)
509     {
510         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
511         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
512         throw new Exception(__CLASS__);
513     }
514     return $result;
515 }
516
517 ////// modificar datos
518
519 public function getuser($id)
520 {
521     $query = $this-> db-> get_where('sis_usuario', array('id' => $id));
522     if (!$query)
523     {
524         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
525         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
526         throw new Exception(__CLASS__);
527     }
528     else
529         return $query-> row();
530     return null;
531 }
532 public function getgrupo($grupo)
533 {
534     $query = $this-> db-> get_where('sis_grupo', array('id' => $grupo));
535     if (!$query)
536     {
537         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
538         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
539         throw new Exception(__CLASS__);
540     }
541     else
542         return $query-> row();
543     return null;
544 }
545 public function update_user($id,$clave,$correo)
546 {
547     $data = array('clave' => $clave,'correo' => $correo);
548     $this-> db-> where('id' , $id);
549     $result = $this-> db-> update('sis_usuario', $data);
550     if (!$result)
551     {
552         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
553         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
554         throw new Exception(__CLASS__);
555     }
556     return $result;
557 }
558
559 // obtiene los permisos que pertenezcan a un entorno parametros nombre entorno
560 public function get_permiso_entorno($nombre)
561 {
562     $this-> db-> distinct();
563     $this-> db-> select('p.id, p.id_grupo, p.fecha, p.id_controlador_accion');
564     $this-> db-> from('sis_entorno e');
565     $this-> db-> join('sis_controlador c', 'c.id_entorno = e.id','left');
566     $this-> db-> join('sis_controlador_x_accion ca', 'ca.id_controlador = c.id','left');
567     $this-> db-> join('sis_permiso p', 'p.id_controlador_accion = ca.id','left');
568     $this-> db-> where('e.nombre' , $nombre);
569     $this-> db-> where('p.id !=' , '');
570     $query = $this-> db-> get();
571     if (!$query)
572     {
573         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
574         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
575         throw new Exception(__CLASS__);
576     }
577     else
578         return $query-> result();
579     return null;

```

```

580     }
581
582     // obtiene los grupos que pertenezcan a un entorno parametros nombre entorno
583     public function get_grupo_entorno($nombre)
584     {
585         $this-> db-> distinct();
586         $this-> db-> select('g.id, g.nombre, g.descripcion');
587         $this-> db-> from('sis_entorno e');
588         $this-> db-> join('sis_controlador c', 'c.id_entorno = e.id','left');
589         $this-> db-> join('sis_controlador_x_accion ca', 'ca.id_controlador = c.id','left');
590         $this-> db-> join('sis_permiso p', 'p.id_controlador_accion = ca.id','left');
591         $this-> db-> join('sis_grupo g', 'g.id = p.id_grupo','left');
592         $this-> db-> where('e.nombre' , $nombre);
593         $this-> db-> where('g.id !=' , '');
594         $query = $this-> db-> get();
595         if (!$query)
596         {
597             $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
598             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
599             throw new Exception(__CLASS__);
600         }
601         else
602             return $query-> result();
603         return null;
604     }
605
606     // obtiene los usuario que pertenezcan a un entorno parametros nombre entorno
607     public function get_usuario_entorno($nombre,$inusuario="")
608     {
609         $this-> db-> distinct();
610         $this-> db-> select(' u.id, u.nombre_usuario, u.clave, u.nombre,
u.apellido_paterno, u.apellido_materno, u.correo, u.activo, u.id_grupo');
611         $this-> db-> from('sis_entorno e');
612         $this-> db-> join('sis_controlador c', 'c.id_entorno = e.id','left');
613         $this-> db-> join('sis_controlador_x_accion ca', 'ca.id_controlador = c.id','left');
614         $this-> db-> join('sis_permiso p', 'p.id_controlador_accion = ca.id','left');
615         $this-> db-> join('sis_grupo g', 'g.id = p.id_grupo','left');
616         $this-> db-> join('sis_usuario u', 'u.id_grupo = g.id','left');
617         $this-> db-> where('e.nombre' , $nombre);
618         $this-> db-> where('u.activo' , 1);
619         $this-> db-> where('u.id !=' , '');
620         if($inusuario!="")
621             $this-> db-> where_in('u.id' , $inusuario);
622         $query = $this-> db-> get();
623         if (!$query)
624         {
625             $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
626             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
627             throw new Exception(__CLASS__);
628         }
629         else
630             return $query-> result();
631         return null;
632     }
633 }
634 ?>
```

## Paquete TES

# Archivo fuente para enrolamiento.php

La documentación para este archivo está disponible en [enrolamiento.php](#)

```
1  <?php      if ( ! defined('BASEPATH')) exit('No direct script access allowed');
2  /**
3   * Controlador de enrolamiento
4   *
5   * @package    TES
6   * @subpackage Controlador
7   * @author     Eliecer
8   * @created    2013-12-17
9   */
10  class Enrolamiento extends CI_Controller
11  {
12
13      public function __construct()
14      {
15          parent::__construct();
16          try
17          {
18              $this-> load-> helper('url');
19              $this-> load-> helper('date');
20              $this-> load-> helper('formatFecha');
21          }
22          catch(Exception $e)
23          {
24              $this-> template-> write("content" , $e-> getMessage());
25              $this-> template-> render();
26          }
27      }
28
29      /**
30      *
31      * Este es el metodo por default, obtiene el listado de las personas
32      * se recibe el parametro $pag de tipo int que representa la paginacion
33      * @param      string      $pag      numero de pagina para la posicion
34      * @param      string      $id      id de una persona
35      *
36      * @return      echo object
37      */
38      public function index($pag = 0, $id="" , $array="")
39      {
40          try{
41              if (empty($this-> Usuario_model))
42                  return false;
43              if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url()))
44                  show_error('', 403, 'Acceso denegado');
45              $this-> load-> model(DIR_TES.'/Enrolamiento_model');
46              $data['title'] = 'Lista Enrolados';
47              $this-> load-> helper('form');
48              $this-> load-> library('pagination');

49              $data['id'] = $id;
50              $data['pag'] = $pag;
51              if($array!="")
52              {
53                  $data['infoclass'] = $array['infoclass'];
54                  $data['msgResult'] = $array['msgResult'];
55              }

56              // Configuración para el Paginador
57              $configPag['base_url'] = '/'.DIR_TES.'/enrolamiento/index/';
58              $configPag['first_link'] = 'Primero';
59              $configPag['last_link'] = '&Uacute;ltim' ;
60              $configPag['uri_segment'] = '4';
61              $configPag['total_rows'] = $this-> Enrolamiento_model-> getNumRows($this-
62
63      > input-> post('busqueda'));
64              $configPag['per_page'] = 20;
65              $this-> pagination-> initialize($configPag);
66              if ($this-> input-> post('busqueda'))
```

```

67          $data['users'] = $this-> Enrolamiento_model-> getListEnrolamiento($this-
> input-> post('busqueda'), $configPag['per_page'], $pag);
68          else
69              $data['users'] = $this-> Enrolamiento_model-> getListEnrolamiento('', 
$configPag['per_page'], $pag);
70      }
71      catch(Exception $e){
72          $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
73      }
74      // $this->load->view('usuario/index', $data);
75      $this-> template-> write_view('content',DIR_TES.'/enrolamiento/enrolamiento_list',
$data);
76      $this-> template-> render();
77  }
78 /**
79 *
80 * Este metodo extrae la informacion del paciente que sera impreso en la tarjeta
81 *
82 * @param string $id identificador de la persona
83 *
84 * @return echo
85 */
86
87 public function print_card($id)
88 {
89     $this-> load-> model(DIR_SIIGS.'/ArbolSegmentacion_model');
90     $this-> load-> model(DIR_TES.'/Enrolamiento_model');
91     $persona=$this-> Enrolamiento_model-> getById($id);
92     $alergia=$this-> Enrolamiento_model-> getAlergia($id);
93     // $persona = (array) $persona;
94     // var_dump($persona);
95     $datos=array("nombre" => $persona-> apellido_paterno.".
" . $persona-> apellido_materno." .
" . $persona-> nombre,
96                 "sexo" => $persona-> sexo,
97                 "nombre_madre" => $persona-> nombreT." .
" . $persona-> paternoT." .
" . $persona-> maternot,
98                 "domicilio" => $persona-> calle_domicilio." .
" . $persona-> numero_domicilio.", " .
" . $persona-> colonia_domicilio
99             );
100    $dom=$this-> ArbolSegmentacion_model-> getDescripcionById(array($persona-
101 > id_asu_localidad_domicilio),3); // loca edo
102    $asu=$this-> ArbolSegmentacion_model-> getDescripcionById(array($persona-
103 > id_asu_um_tratante),3); // um juridiccion
104    if($dom)
105    {
106        $dom=explode(", ", $dom[0]-> descripcion);
107        $datos["localidad"] = trim($dom[0]);
108        $datos["municipio"] = trim($dom[1]);
109    }
110    if($asu)
111    {
112        $asu=explode(", ", $asu[0]-> descripcion);
113        $datos["um"] = trim($asu[0]);
114        $datos["juridiccion"] = trim($asu[3]);
115    }
116    $valor=array();
117    foreach($alergia as $x)
118    {
119        $valor[]=$x-> descripcion;
120    }
121    $datos["alergias"] = implode(", ", $valor);
122    $folio=$this-> Enrolamiento_model-> getfolio($persona-> id);
123    $datos["folio"] = $folio[0]-> folio;
124    echo implode("| ", $datos);
125  }
126 /**
127 *
128 * Crea la pagina para ver la infromacion de la persona
129 *
130 * @param string $id identificador de la persona
131 *
132 * @return echo
133 */
134 public function view($id)
135 {
136     try
137     {
138         $this-> load-> model(DIR_TES.'/Enrolamiento_model');

```

```

139         $this-> load-> model(DIR_TES.'/Reporte_sincronizacion_model');
140         if (empty($this-> Enrolamiento_model))
141             return false;
142         if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url()))
143             show_error('', 403, 'Acceso denegado');
144
145         $data['title'] = 'Ver Paciente';
146         $data['enrolado'] = $this-> Enrolamiento_model-> getById($id);
147         if(empty($data['enrolado']))
148         {
149             $data['infoclass'] = 'error';
150             $data['msgResult'] = "Registro no encontrado";
151
152             $this-> template-
153             > write_view('content',DIR_TES.'/enrolamiento/enrolamiento_view', $data);
154             $this-> template-> render();
155             return true;
156         }
157         $data['alergias'] = $this-> Enrolamiento_model-> getAlergia($id);
158         $data['afiliaciones'] = $this-> Enrolamiento_model-> getAfiliaciones($id);
159         $data['consultas']=$this-> Enrolamiento_model-> getControlConsultas($id);
160         $data['nutricionales']=$this-> Enrolamiento_model-
161             > get_catalog_view("accion_nutricional"
162             ,,$id);
163         $fecha=$data['enrolado']-> fecha_nacimiento;
164         $data['vacunacion']=$this-> Reporte_sincronizacion_model-
165             > getListado("      SELECT DISTINCT r.id_vacuna, cv.codigo_barra,
v.descripcion,r.dia_inicio_aplicacion_nacido, r.dia_fin_aplicacion_nacido, p.fecha_nacimiento, CASE
WHEN r.id_vacuna = cv.id_vacuna THEN 'X' ELSE '' END AS tiene, CASE WHEN cv.fecha IS NULL THEN
CONCAT('Desde:',r.dia_inicio_aplicacion_nacido,' Hasta:',r.dia_fin_aplicacion_nacido) ELSE
CONCAT('Fecha Aplicada: ',' ',DATE_FORMAT(cv.fecha, '%d-%m-%Y')) END AS
fecha,DATEDIFF(NOW(),'$fecha') AS dias,CASE WHEN
DATEDIFF(NOW(),'$fecha')>=r.dia_inicio_aplicacion_nacido AND
DATEDIFF(NOW(),'$fecha')<=r.dia_fin_aplicacion_nacido THEN '1' ELSE (CASE WHEN
DATEDIFF(NOW(),'$fecha')>r.dia_fin_aplicacion_nacido AND cv.fecha IS NULL THEN '2' ELSE (CASE WHEN
DATEDIFF(NOW(),'$fecha')<r.dia_inicio_aplicacion_nacido AND cv.fecha IS NULL THEN '3' END) END)
END AS prioridad FROM cns_regla_vacuna r LEFT JOIN cns_vacuna v ON v.id=r.id_vacuna LEFT JOIN
cns_control_vacuna cv ON cv.id_persona=' $id' AND cv.id_vacuna=r.id_vacuna LEFT JOIN cns_persona p ON
p.id=cv.id_persona GROUP BY v.descripcion ORDER BY r.id_vacuna,r.orden_esq_com ASC");
166         $data['estimulacion_temprana'] = $this-> Enrolamiento_model-
167             > get_estimulacion($id);
168         $data['sales'] = $this-> Enrolamiento_model-> get_sales($id);
169
170         // Obtiene los datos para las graficas
171         $data['peso_edad'] = json_encode($this-> Enrolamiento_model-
172             > get_datos_grafica('peso_edad', $data['enrolado']-> sexo, $data['enrolado']-> edad_meses,
173             $data['enrolado']-> id));
174         $data['peso_talla'] = json_encode($this-> Enrolamiento_model-
175             > get_datos_grafica('peso_talla', $data['enrolado']-> sexo, $data['enrolado']-> edad_meses,
176             $data['enrolado']-> id));
177         $data['talla_edad'] = json_encode($this-> Enrolamiento_model-
178             > get_datos_grafica('talla_edad', $data['enrolado']-> sexo, $data['enrolado']-> edad_meses,
179             $data['enrolado']-> id));
180         $data['imc'] = json_encode($this-> Enrolamiento_model-
181             > get_datos_grafica('imc', $data['enrolado']-> sexo, $data['enrolado']-> edad_meses,
182             $data['enrolado']-> id));
183         $data['peri_cefa'] = json_encode($this-> Enrolamiento_model-
184             > get_datos_grafica('peri_cefa', $data['enrolado']-> sexo, $data['enrolado']-> edad_meses,
185             $data['enrolado']-> id));
186         $data['con_hemo'] = json_encode($this-> Enrolamiento_model-
187             > get_datos_grafica('con_hemo', $data['enrolado']-> sexo, $data['enrolado']-> edad_meses,
188             $data['enrolado']-> id, $data['enrolado']-> id_asu_localidad_domicilio));
189
190         catch(Exception $e)
191         {
192             $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
193         }
194         $this-> template-> write('header','','true');
195         $this-> template-> write('footer','','true');
196         $this-> template-> write('menu','','true');
197         $this-> template-> write('sala_prensa','','true');
198         $this-> template-> write_view('content',DIR_TES.'/enrolamiento/enrolamiento_view',
199             $data);
200         $this-> template-> render();
201
202         /**
203         *
204         * Crea el formulario para editar la informacion de la persona
205         */

```

```

190 * @param string $id identificador de la persona
191 *
192 * @return echo
193 */
194 public function update($id)
195 {
196     try
197     {
198         if (empty($this-> Usuario_model))
199             return false;
200         if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url()))
201             show_error('', 403, 'Acceso denegado');
202         $this-> load-> model(DIR_TES.'/Enrolamiento_model');
203         $data['id'] = $id;
204         $data['title'] = 'Ver Paciente';
205         $data['enrolado'] = $this-> Enrolamiento_model-> getById($id);
206         if(empty($data['enrolado']))
207         {
208             $data['infoclass'] = 'error';
209             $data['msgResult'] = "Registro no encontrado";
210         }
211         $this-> template-
212 > write_view('content',DIR_TES.'/enrolamiento/enrolamiento_update', $data);
213         $this-> template-> render();
214         return true;
215     }
216     $data['alergias'] = $this-> Enrolamiento_model-> getAlergia($id);
217     $data['afiliaciones'] = $this-> Enrolamiento_model-> getAfiliaciones($id);
218
219     $data['vacunas']=$this-> Enrolamiento_model-
220 > get_catalog_view("vacuna" , $id, "id_vacuna" );
221
222     $data['consultas']=$this-> Enrolamiento_model-> getControlConsultas($id);
223     $data['nutricionales']=$this-> Enrolamiento_model-
224 > get_catalog_view("accion_nutricional" , $id);
225
226     $nutricion=$this-> Enrolamiento_model-> get_control_nutricional($id);
227     $data['nutriciones']=$nutricion;
228
229     $data['peri_cefa'] = $this-> Enrolamiento_model-> get_peri_cefa($id);
230     $data['estimulacion_temprana'] = $this-> Enrolamiento_model-
231 > get_estimulacion($id);
232     $data['sales'] = $this-> Enrolamiento_model-> get_sales($id);
233
234     catch(Exception $e)
235     {
236         $data['infoclass'] = 'error';
237         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
238     }
239     try
240     {
241         if (empty($this-> Enrolamiento_model))
242             return false;
243     /*
244     if (!Usuario_model::checkCredentials(DIR_SIIGS.'::'.__METHOD__, current_url()))
245         show_error('', 403, 'Acceso denegado');*/
246
247         if($this-> validarForm("update"))
248         {
249             try
250             {
251                 $this-> Enrolamiento_model-> setId($id);
252                 $this-> addForm();
253                 // actualizar si todo bien
254                 if(isset($_POST["id_cns_basico"])
255                     $id=$this-> Enrolamiento_model-> update_basico();
256
257                 if(isset($_POST["id_cns_beneficiario"])
258                     $id=$this-> Enrolamiento_model-> update_beneficiario();
259
260                 if(isset($_POST["id_cns_tutor"])
261                     $id=$this-> Enrolamiento_model-> update_tutor();
262
263                 if(isset($_POST["id_cns_umt"])
264                     $id=$this-> Enrolamiento_model-> update_umt();
265
266                 if(isset($_POST["id_cns_regcivil"])
267                     $id=$this-> Enrolamiento_model-> update_regcivil();
268
269                 if(isset($_POST["id_cns_direccion"])
270

```

```

266             $id=$this-> Enrolamiento_model-> update_direccion();
267
268             if(isset($_POST[ "id_cns_alergia"
269                         $id=$this-> Enrolamiento_model->           ]))
270                         update_alergia();
271
272             if(isset($_POST[ "id_cns_vacuna"
273                         $id=$this-> Enrolamiento_model->           ]))
274                         update_vacuna();
275
276             if(isset($_POST[ "id_cns_consulta"
277                         $id=$this-> Enrolamiento_model->           ]))
278                         update_consulta();
279
280             if(isset($_POST[ "id_cns_accion"
281                         $id=$this-> Enrolamiento_model->           ]))
282                         update_accion();
283
284             if(isset($_POST[ "id_cns_nutricion"
285                         $id=$this-> Enrolamiento_model->           ]))
286                         update_nutricion();
287
288             if(!empty($_POST[ "peri_cefa"
289                         $id=$this-> Enrolamiento_model->           ]))
290                         update_peri_cefa();
291
292             if(!empty($_POST[ "estimulacion_fecha"
293                         $id=$this-> Enrolamiento_model->           ]))
294                         update_estimulacion();
295
296             if(!empty($_POST[ "sales_fecha"
297                         $id=$this-> Enrolamiento_model->           ]))
298                         update_sales();
299
300             $data['id'] = $this-> Enrolamiento_model-> getId();
301             $midata['infoclass'] = 'success';
302             $midata['msgResult'] = 'Registro Actualizado Exitosamente';
303             Bitacora_model::insert(DIR_SIIGS.'::'.__METHOD__, 'Usuario Enrolado:
304             '.strtoupper($this-> input-> post('nombre')));
305
306             $this-> index(0,$data['id'],$midata);
307         }
308         catch (Exception $e)
309     {
310         $data['infoclass'] = 'error';
311         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
312         $data["title"] = "TES";
313         $data["titulo"] = "Enrolamiento";
314
315         // $this->template->write_view('header',DIR_TES.'/header.php');
316         // $this->template->write_view('menu',DIR_TES.'/menu.php');
317         $this-> template-
318         > write_view('content',DIR_TES.'/enrolamiento/enrolamiento_update', $data);
319         // $this->template->write_view('footer',DIR_TES.'/footer.php');
320         $this-> template-> render();
321     }
322     else
323     {
324         $this-> template-
325         > write_view('content',DIR_TES.'/enrolamiento/enrolamiento_update', $data);
326         $this-> template-> render();
327     }
328     catch(Exception $e)
329     {
330         $data['infoclass'] = 'error';
331         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
332     }
333
334 /**
335 *
336 * Genera los options de un campo tipo select
337 *
338 * @param      string      $catalog      tabla de donde se extrae la informacion
339 * @param      string      $sel          identifica si un valor ya esta seleccionado
340 * @param      string      $orden        columna para hacer el ordenamiento
341 * @param      string      $campo        campo de la tabla para hacer el where
342 * @param      string      $valor        valor a comparar en el where
343 *
344 * @return
345 */
346 public function
347 catalog_select($catalog,$sel="" , $orden="" , $campo="" , $valor="")
348 {
349     $opcion="";
350     $this-> load-> model(DIR_TES.'/Enrolamiento_model');

```

```

342         $datos=$this-> Enrolamiento_model-
343 >     get_catalog("cns_ " . $catalog,$campo,$valor,$orden);
344     {
345         $opcion.= "<option value='>Seleccione...</option>" ;
346         foreach($datos as $dato)
347         {
348             $id=$dato-> id;
349             $che= "";
350             if(strpos(".", $sel,$id))$che="selected" ;
351             $descripcion=$dato-> descripcion;
352             $opcion.= "      <option value=' $id' $che> $descripcion</option>" ;
353         }
354         echo $opcion;
355     }
356     else
357         echo "<option>No hay Datos</option>" ;
358 }
359 /**
360 *
361 * Genera los options de un campo tipo select para los tratamientos de consultas
362 *
363 * @param string $campo campo de la tabla para hacer el where
364 * @param string $valor valor a comparar en el where
365 * @param string $sel identifica si un valor ya esta seleccionado
366 * @param string $orden columna para hacer el ordenamiento
367 *
368 */
369 * @return echo
370 */
371 public function
tratamiento_select($campo="" , $valor="" , $sel="" , $orden="" )
372 {
373     $opcion="";
374     $valor=urldecode($valor);
375     $this-> load-> model(DIR_TES.'Enrolamiento_model');
376     $datos=$this-> Enrolamiento_model-
377 >     get_catalog_tratamiento("cns_tratamiento" , $campo,$valor,$orden);
378     if(sizeof($datos)!=0)
379     {
380         $opcion.= "<option value='>Seleccione...</option>" ;
381         foreach($datos as $dato)
382         {
383             $che= "";
384             if($orden=="tipo" || $orden=="cc" )
385             {
386                 $yd=$dato-> id;
387                 $id=$dato-> tipo;
388                 $descripcion=$dato-> tipo;
389                 $che= "";
390                 if(strpos(".", $sel,$yd))$che="selected" ;
391             }
392             else
393             {
394                 $id=$dato-> id;
395                 $descripcion=$dato-> descripcion;
396                 if(strpos(".", $sel,$id))$che="selected" ;
397             }
398             $opcion.= "      <option value=' $id' $che> $descripcion</option>" ;
399         }
400         echo $opcion;
401     }
402     else
403         echo "<option>No hay Datos</option>" ;
404 }
405 /**
406 *
407 * Crea un grupo de radio o check con la informacion de los catalogos
408 *
409 * @param string $catalog tabla de donde se extrae la informacion
410 * @param string $tipo tipo de control radio o check
411 * @param int $col numero de columnas en la tabla
412 * @param string $sel identifica si un valor ya esta seleccionado
413 * @param string $orden columna para hacer el ordenamiento
414 *
415 * @return echo
416 */
417 public function catalog_check($catalog,$tipo,$col=1,$sel="" , $orden="" )
418 {

```

```

419     $opcion= " ";
420     $this-> load-> model(DIR_TES.'Enrolamiento_model');
421     $datos=$this-> Enrolamiento_model-
422 > get_catalog("cns_".catalog,"",$orden);
423 {
424     $i=0;$a=0;$y=0;$temp="" ;$x=0;
425     $opcion='<table width="100%" ><tr>' ;
426     foreach($datos as $dato)
427 {
428         $id=$dato-> id;
429         $descripcion=$dato-> descripcion;
430         $che="";
431         if($catalog=="alergia")
432 {
433             if($temp!=$dato-> tipo)
434             {
435                 if($y> 0)
436                 {
437                     $x++;
438                     $opcion.= "</tr></table>" ;
439                     if($x==$col){$opcion.= "<tr>" ; $x=0;}
440                 }
441             $opcion.= "<td width='33%' valign='top'><table width='98%'" . $dato-
442 > tipo." </th></tr><tr>" ;
443             $y++;
444         }
445         else $opcion.= "</tr><tr>" ;
446         $temp=$dato-> tipo;
447     }
448     if(stripes(".".$sel,$id))$che="checked" ;
449
450     if($a==$col&& $catalog!="alergia" ){$opcion.= "</tr><tr>" ; $a=0;}
451     if($catalog=="afiliacion")
452     {$xi=$i+200; $xy=$xi+1;}
453     if($catalog=="alergia")
454     {$xi=$i+400; $xy=$xi+1;}
455     $opcion.= "<td width='33%' valign='top'><label><input
name='".$catalog."[]' id='".$catalog.$i' type='".$tipo' value='".$id' $che style='margin-top:-2px;' tabindex='".$xi' onkeydown='return enterTab(event,0)'>
$descripcion</label></td>" ;
456     $i++;$a++;
457 }
458 $opcion.= '</tr></table>' ;
459 echo $opcion;
460 }
461 else
462 echo "No hay Datos" ;
463 /**
464 *
465 * Crea el autocomplete para facilitar la busqueda de un tutor
466 *
467 * @return
468 * @param string $term
469 */
470 public function autocomplete()
471 {
472     $term=$_GET["term"] ;
473     $this-> load-> model(DIR_TES.'Enrolamiento_model');
474     $datos=$this-> Enrolamiento_model-> autocomplete_tutor($term);
475     $array = array();
476     $i=0;
477     foreach($datos as $data)
478 {
479         $array[$i] = trim((($data-> curp)." => ".$data-> nombre." .
480 ". $data-> apellido_paterno." ". $data-> apellido_materno));
481         $i++;
482     }
483     echo json_encode($array);
484 }
485 /**
486 * Obtiene informacion del tutor
487 *
488 * @param string $curp curp del tutor
489 *
490 * @return
491 echo

```

```

491      */
492     public function data_tutor($curp)
493     {
494         $this-> load-> model(DIR_TES.'/Enrolamiento_model');
495         $datos=$this-> Enrolamiento_model-> data_tutor($curp);
496         if(sizeof($datos)!=0)
497         {
498
499             foreach($datos as $dato)
500             {
501                 $m=FALSE;$f=FALSE;
502                 if($dato-> sexo=="M") $m=1;
503                 if($dato-> sexo=="F") $f=1;
504                 $array=array(
505                     array(
506                         "idtutor" => $dato-> id,
507                         "nombreT" => $dato-> nombre,
508                         "paternoT" => $dato-> apellido_paterno,
509                         "maternoT" => $dato-> apellido_materno,
510                         "celularT" => $dato-> celular,
511                         "curpT" => $dato-> curp,
512                         "telefonoT" => $dato-> telefono,
513                         "companiaT" => $dato-> id_operadora_celular,
514                         "sexot_1" => $m,
515                         "sexot_2" => $f,
516                         "error" => ""
517                     )
518                 );
519             }
520         }
521         else
522             $array=array(
523                 array(
524                     "error" => "No existe curp: " . $curp,
525                 )
526             );
527
528         echo json_encode($array);
529     }
530     /**
531     *
532     * crea un archivo descargable el cual se necesita para el envio por nfc a la tarjeta del
533     * paciente
534     * @param string $id identificador de la persona
535     * @return echo
536     */
537     public function file_to_card($id,$tipo="")
538     {
539         $archivo=date("YmdHis").".tesf";
540         $SEPARADOR_BLOQUE = "\r\n";
541
542         header("Pragma: public");
543         header("Expires: 0");
544         header("Cache-Control: must-revalidate, post-check=0, pre-check=0");
545         if($tipo=="")
546         {
547             header("Content-Type: application/force-download");
548             header("Content-Type: application/octet-stream");
549             header("Content-Type: application/download");
550             header("Content-Disposition: attachment;filename=" . $archivo);
551         }
552         header("Content-Transfer-Encoding: binary");
553         $this-> load-> model(DIR_TES.'/Enrolamiento_model');
554         $data="";
555
556         // Bloque Comun
557
558         $version = $this-> Enrolamiento_model-> get_version();
559         $data.= $version[0]-> version."~";
560
561         $enrolado =(array)$this-> Enrolamiento_model-> getById($id);
562         $data.=$enrolado["id"]."=";
563         $data.=$enrolado["curp"]."=";
564         $data.=$enrolado["nombre"]."=";
565         $data.=$enrolado["apellido_paterno"]."=";
566         $data.=$enrolado["apellido_materno"]."=";
567         $data.=$enrolado["sexo"]."=";
568         $data.=$enrolado["id_tipo_sanguineo"];"
```

```

if($enrolado["id_tipo_sanguíneo"] == "") $data.= "n=" ; $else
570     $data.=$enrolado["fecha_nacimiento"] . "n=" ;
571     $data.=$enrolado["id_asu_localidad_nacimiento"] . "n=" ); $data.= "n=" ; $else
if($enrolado["id_asu_localidad_nacimiento"] == "") $data.= "n=" ; $else
572     $data.=$enrolado["calle_domicilio"] . "n=" ;
if($enrolado["calle_domicilio"] == "") )$data.= "n=" ; $else
573     $data.=$enrolado["numero_domicilio"] . "n=" ;
if($enrolado["numero_domicilio"] == "") )$data.= "n=" ; $else
574     $data.=$enrolado["colonia_domicilio"] . "n=" ;
if($enrolado["colonia_domicilio"] == "") )$data.= "n=" ; $else
575     $data.=$enrolado["referencia_domicilio"] . "n=" ;
if($enrolado["referencia_domicilio"] == "") )$data.= "n=" ; $else
576     $data.= "n=" ;
577     $data.=$enrolado["ageb"] . "n=" ;
if($enrolado["ageb"] == "") )$data.= "n=" ; $else $data.= "n=" ;
578     $data.=$enrolado["manzana"] . "n=" ;
if($enrolado["manzana"] == "") )$data.= "n=" ; $else $data.= "n=" ;
579     $data.=$enrolado["sector"] . "n=" ;
if($enrolado["sector"] == "") )$data.= "n=" ; $else $data.= "n=" ;
580     $data.=$enrolado["id_asu_localidad_domicilio"] . "n=" ;
if($enrolado["id_asu_localidad_domicilio"] == "") )$data.= "n=" ; $else
581     $data.= "n=" ;
582     $data.=$enrolado["cp_domicilio"] . "n=" ;
if($enrolado["cp_domicilio"] == "") )$data.= "n=" ; $else $data.= "n=" ;
583     $data.=$enrolado["telefono_domicilio"] . "n=" ;
if($enrolado["telefono_domicilio"] == "") )$data.= "n=" ; $else
584     $data.= "n=" ;
585     $data.=$enrolado["id_asu_um_tratante"] . "n=" ;
if($enrolado["id_asu_um_tratante"] == "") )$data.= "n=" ; $else
586     $data.=$enrolado["celular"] . "n=" ;
if($enrolado["celular"] == "") )$data.= "n=" ; $else $data.= "n=" ;
587     $data.=$enrolado["ultima_actualizacion"] . "n=" ;
if($enrolado["ultima_actualizacion"] == "") )$data.= "n=" ; $else
588     $data.= "n=" ;
589     $data.=$enrolado["id_nacionalidad"] . "n=" ;
if($enrolado["id_nacionalidad"] == "") )$data.= "n=" ; $else
590     $data.=$enrolado["id_operadora_celular"] . "n=" ;
if($enrolado["id_operadora_celular"] == "") )$data.= "n=" ; $else
591     $data.= "n=" ;
592     $data.=$enrolado["idT"] . "n=" ;
593     $data.=$enrolado["curpT"] . "n=" ;
if($enrolado["curpT"] == "") )$data.= "n=" ; $else $data.= "n=" ;
594     $data.=$enrolado["nombreT"] . "n=" ;
if($enrolado["nombreT"] == "") )$data.= "n=" ; $else $data.= "n=" ;
595     $data.=$enrolado["paternoT"] . "n=" ;
if($enrolado["paternoT"] == "") )$data.= "n=" ; $else $data.= "n=" ;
596     $data.=$enrolado["maternoT"] . "n=" ;
if($enrolado["maternoT"] == "") )$data.= "n=" ; $else $data.= "n=" ;
597     $data.=$enrolado["sexoT"] . "n=" ;
if($enrolado["sexoT"] == "") )$data.= "n=" ; $else $data.= "n=" ;
598     $data.=$enrolado["telefonoT"] . "n=" ;
if($enrolado["telefonoT"] == "") )$data.= "n=" ; $else $data.= "n=" ;
599     $data.=$enrolado["celularT"] . "n=" ;
if($enrolado["celularT"] == "") )$data.= "n=" ; $else $data.= "n=" ;
600     $data.=$enrolado["operadoraTid"] . "n=" ;
if($enrolado["operadoraTid"] == "") )$data.= "n=" ; $else
601     $data.= "n=" ;
602     $data.= "n=" ;
603     $registro = (array)$this->Enrolamiento_model->getRegistro_civil($id);
604     if(count($registro) > 0)
605     {
606         $data.=$registro["id_localidad_registro_civil"] . "n=" ;
607         $data.=$registro["fecha_registro"] . "n=" ;
608     }
609 }
```

```

610
611     $data.= " ~" ; 
612     $alergias = $this-> Enrolamiento_model-> getAlergia($id,'ultima_actualizacion');
613     foreach($alergias as $x)
614     {
615         $data.= $x-> id." o" ;
616         if(empty($alergias))
617             $data.= " ~" ;
618         else
619             $data=substr($data,0,strlen($data)-2)." ~" ;
620             $afiliaciones = $this-> Enrolamiento_model-
621 > getAfiliaciones($id,'ultima_actualizacion');
622             foreach($afiliaciones as $x)
623             {
624                 $data.= $x-> id." o" ;
625                 if(empty($afiliaciones))
626                     $data.= " ~" ;
627                 else
628                     $data=substr($data,0,strlen($data)-2)." ~" ;
629                     $vacunas=$this-> Enrolamiento_model-
630 > get_catalog_view("vacuna" , $id, ' ', 'fecha');
631                     foreach($vacunas as $x)
632                     {
633                         $data.= $x-> id." = " ;
634                         $data.= $x-> fecha." = " ;
635                         $data.= $x-> id_asu_um." o" ;
636                     }
637                     if(empty($vacunas))
638                         $data.= " ~" ;
639                     else
640                         $data=substr($data,0,strlen($data)-2)." ~" ;
641 // Bloque exclusivo ISECH
642 $data .= $SEPARADOR_BLOQUE;
643
644 $consultas=$this-> Enrolamiento_model-
645 > get_catalog("cns_control_consulta" , 'id_persona' , $id, 'fecha');
646 foreach($consultas as $x)
647     {
648         $data.= $x-> clave_cielo." = " ;
649         $data.= $x-> fecha." = " ;
650         $data.= $x-> id_asu_um." = " ;
651         $data.= $x-> id_tratamiento; if($x-
652 > id_tratamiento==" ")
653             $data.= " ~" ; else $data.= " = "
654             $data.= $x-> grupo_fecha_secuencial; if($x-
655 > grupo_fecha_secuencial==" ")
656             $data.= " ~" ; else $data.= " o "
657         if(empty($consultas))
658             $data.= " ~" ;
659         else
660             $data=substr($data,0,strlen($data)-2)." ~" ;
661
662 $anutricional=$this-> Enrolamiento_model-
663 > get_catalog_view("accion_nutricional" , $id,'fecha',' ');
664 foreach($anutricional as $x)
665     {
666         $data.= $x-> id." = " ;
667         $data.= $x-> fecha." = " ;
668         $data.= $x-> id_asu_um." o " ;
669     }
670     if(empty($anutricional))
671         $data.= " ~" ;
672     else
673         $data=substr($data,0,strlen($data)-2)." ~" ;
674
675 $nutricion=$this-> Enrolamiento_model-> get_control_nutricional($id);
676 foreach($nutricion as $x)
677     {
678         $data.= $x-> peso." = " ;
679         $data.= $x-> altura." = " ;
680         $data.= $x-> talla." = " ;
681         $data.= $x-> hemoglobina." = " ;
682         $data.= $x-> fecha." = " ;
683         $data.= $x-> id_asu_um." o " ;
684     }
685     if(empty($nutricion))
686         $data.= " ~" ;
687     else
688         $data=substr($data,0,strlen($data)-2)." ~" ;

```

```

684
685     $peri_cefa=$this->  Enrolamiento_model->  get_peri_cefa($id);
686     foreach($peri_cefa as $x)
687     {
688         $data.= $x->  perimetro_cefalico . "=" ;
689         $data.= $x->  fecha . "=" ;
690         $data.= $x->  id_asu_um . " o " ;
691     }
692     if(empty($peri_cefa))
693         $data.= " ~ " ;
694     else
695         $data=substr($data,0,strlen($data)-2)." ~ " ;
696
697     $sales=$this->  Enrolamiento_model->  get_sales($id);
698     foreach($sales as $x)
699     {
700         $data.= $x->  cantidad . "=" ;
701         $data.= $x->  fecha . "=" ;
702         $data.= $x->  id_asu_um . " o " ;
703     }
704     if(empty($sales))
705         $data.= " ~ " ;
706     else
707         $data=substr($data,0,strlen($data)-2)." ~ " ;
708
709     $estimulacion=$this->  Enrolamiento_model->  get_estimulacion($id);
710     foreach($estimulacion as $x)
711     {
712         $data.= $x->  tutor_capacitado . "=" ;
713         $data.= $x->  fecha . "=" ;
714         $data.= $x->  id_asu_um . " o " ;
715     }
716     if(empty($estimulacion))
717         $data.= " ~ " ;
718     else
719         $data=substr($data,0,strlen($data)-2);
720
721     $this->  update_card($id,0,' ', $archivo,4);
722     echo $data;
723
724     // Bloque exclusivo ICSS
725     echo $SEPARADOR_BLOQUE;
726 }
727 /**
728 *
729 * Este metodo actualiza el estado del archivo descargado si fue escrito correctamente o no
730 * en la tarjeta
731 * @param      string      $persona      id de la persona
732 * @param      boolean     $impreso      identifica si el proceso de impresion fue
733 * correcto o no
734 * @param      int         $fecha        fecha del evento
735 * @param      string     $archivo       archivo generado
736 * @param      string     $entorno       tipo de entorno
737 * @return
738 */
739 public function
update_card($persona,$impreso,$fecha="" , $archivo="" , $entorno='4')
740 {
741     $this->  load->  model(DIR_TES.'Enrolamiento_model');
742     if($impreso==1)$fecha=date("Y-m-d H:i:s");
743     $this->  Enrolamiento_model-
>  entorno_x_persona($entorno,$persona,$fecha,$archivo,$impreso);
744 }
745 /**
746 *
747 * valida que un archivo sea valido para enviar a la tarjeta por nfc
748 *
749 * @param      string      $persona      id de la persona
750 * @param      string     $archivo       archivo generado
751 * @return
752 */
753 public function validate_card($persona,$archivo)
754 {
755     $this->  load->  model(DIR_TES.'Enrolamiento_model');
756     echo $this->  Enrolamiento_model->  valid_card($persona,$archivo);
757 }
758
759 }
```

```

760 /**
761 *
762 * prepara los datos para insertarlos
763 *
764 * @return echo
765 */
766 public function insert()
767 {
768     $this-> load-> model(DIR_TES.'Enrolamiento_model');
769     try
770     {
771         if (empty($this-> Enrolamiento_model))
772             return false;
773
774         if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url()))
775             show_error('', 403, 'Acceso denegado');
776
777         if($this-> validarForm())
778         {
779             try
780             {
781                 $this-> addForm();
782
783                 $id=$this-> Enrolamiento_model-> insert();
784                 $midata['infoclass'] = 'success';
785                 $midata['msgResult'] = 'Registro Agregado Exitosamente';
786                 Bitacora_model::insert(DIR_SIIGS.'::'.__METHOD__, 'Usuario Enrolado:
'.strtoupper($this-> input-> post('nombre')));
787                 $this-> session-> set_userdata( 'umt', $this-> Enrolamiento_model-
> getumt() );
788
789                 $this-> index(0,$id,$midata);
790             }
791             catch (Exception $e)
792             {
793                 $data['infoclass'] = 'error';
794                 $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
795                 $data["title"]           = "TES";
796                 $data["titulo"]          = "Enrolamiento";
797                 $data["session"]         = $this-> session-> userdata('umt');
798
799                 // $this->template->write_view('header',DIR_TES.'/header.php');
800                 // $this->template->write_view('menu',DIR_TES.'/menu.php');
801                 $this-> template-
802                     > write_view('content',DIR_TES.'/enrolamiento/enrolamiento',$data);
803                     // $this->template->write_view('footer',DIR_TES.'/footer.php');
804                     $this-> template-> render();
805             }
806             else
807             {
808                 $data["title"]           = "TES";
809                 $data["titulo"]          = "Enrolamiento";
810                 $data["session"]         = $this-> session-> userdata('umt');
811
812                 // $this->template->write_view('header',DIR_TES.'/header.php');
813                 // $this->template->write_view('menu',DIR_TES.'/menu.php');
814                 $this-> template-
815                     > write_view('content',DIR_TES.'/enrolamiento/enrolamiento',$data);
816                     // $this->template->write_view('footer',DIR_TES.'/footer.php');
817                     $this-> template-> render();
818             }
819             catch(Exception $e)
820             {
821                 $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
822             }
823         }
824
825 /**
826 *
827 * valida los datos de entrada en el formulario
828 *
829 * @param string $op bandera que identifica si una seccion entra en
830 * validacion
831 * @return echo
832 */
833 public function validarForm($op="")
834 {

```

```

835     $data['titulo'] = 'Nuevo Enrolamiento';
836     $this-> load-> helper(array('form', 'url'));
837     $this-> load-> library('form_validation');
838     if(isset($_POST["id_cns_basico"]           ))||$op=="")
839     {
840         $this-> form_validation->
841             trim|xss_clean|required|max_length[40];
842             $this-> form_validation->
843                 trim|xss_clean|required|max_length[25];
844             $this-> form_validation->
845                 trim|required';
846             $this-> form_validation->
847                 trim|required';
848             $this-> form_validation->
849                 trim|xss_clean|trim';
850             $this-> form_validation->
851         }
852
853         if(isset($_POST["id_cns_recivil"]           ))||$op=="")
854         {
855             $this-> form_validation->
856                 trim|xss_clean';
857             $this-> form_validation->
858         }
859
860         if(isset($_POST["id_cns_umt"]           ))||$op=="")
861         {
862             $this-> form_validation->
863                 trim|required';
864         }
865
866         if(isset($_POST["id_cns_direccion"]           ))||$op=="")
867         {
868             $this-> form_validation->
869             $this-> form_validation->
870             $this-> form_validation->
871             $this-> form_validation->
872             $this-> form_validation->
873             $this-> form_validation->
874             $this-> form_validation->
875             $this-> form_validation->
876             $this-> form_validation->
877             $this-> form_validation->
878             $this-> form_validation->
879             $this-> form_validation->
880             $this-> form_validation->
881         }
882
883         if(isset($_POST["id_cns_tutor"]           ))||$op=="")
884         {
885             $this-> form_validation->
886             $this-> form_validation->
887             $this-> form_validation->
888             $this-> form_validation->
889             $this-> form_validation->
890             $this-> form_validation->
891             $this-> form_validation->
892             trim|xss_clean');//callback_ifCurpTExists
893             $this-> form_validation->
894             $this-> form_validation->
895         }
896         if($op=="update")
897             $this-> form_validation->
898             return $this-> form_validation-> run();
899     }
900
901 /**
902 */

```

```

903     * Pase de parametros para la insercion o actualizacion
904     *
905     * @return      echo
906     */
907     public function addForm()
908     {
909         $this-> Enrolamiento_model-> setnacionalidad($this-> input-
910 > post('nacionalidad'));
911         $this-> Enrolamiento_model-> setnombre(trim(strtoupper($this-> input-
912 > post('nombre'))));
913         $this-> Enrolamiento_model-> setpaterno(trim(strtoupper($this-> input-
914 > post('paterno'))));
915         $this-> Enrolamiento_model-> setmaterno(trim(strtoupper($this-> input-
916 > post('materno'))));
917         $this-> Enrolamiento_model-> setlnacimiento($this-> input-
918 > post('lnacimiento'));
919         $this-> Enrolamiento_model-> setcurp($this-> input-> post('curp').$this-
920 > input-> post('curp2'));
921         $this-> Enrolamiento_model-> setsex($this-> input-> post('sexo'));
922         $this-> Enrolamiento_model-> setsangre($this-> input-> post('sangre'));
923         $this-> Enrolamiento_model-> setfnacimiento($this-> input-
924 > post('fnacimiento'));
925         $this-> Enrolamiento_model-> settbeneficiario($this-> input-
926 > post('tbeneficiario'));
927         $this-> Enrolamiento_model-> setparto($this-> input-> post('parto'));
928         $this-> Enrolamiento_model-> settamiz($this-> input-> post('tamiz'));
929         $this-> Enrolamiento_model-> setprecurp($this-> input-
930 > post('precurp'));
931
932         $this-> Enrolamiento_model-> setidtutor($this-> input-> post('idtutor'));
933
934         $this-> Enrolamiento_model-> setnombreT(trim(strtoupper($this-> input-
935 > post('nombreT'))));
936         $this-> Enrolamiento_model-> setpaternoT(trim(strtoupper($this-> input-
937 > post('paternoT'))));
938         $this-> Enrolamiento_model-> setmaternoT(trim(strtoupper($this-> input-
939 > post('maternoT'))));
940         $this-> Enrolamiento_model-> setcurpT($this-> input-> post('curpT'));
941         $this-> Enrolamiento_model-> setsexot($this-> input-> post('sexot'));
942         $this-> Enrolamiento_model-> settelefonoT($this-> input-> post('telefonoT'));
943         $this-> Enrolamiento_model-> setcompaniaT($this-> input-> post('companiaT'));
944         $this-> Enrolamiento_model-> setcelularT($this-> input-> post('celularT'));
945
946         $this-> Enrolamiento_model-> setfechacivil($this-> input-> post('fechacivil'));
947         $this-> Enrolamiento_model-> setlugarcivil($this-> input-> post('lugarcivil'));
948         $this-> Enrolamiento_model-> setumt($this-> input-> post('um'));
949
950         $this-> Enrolamiento_model-> setcalle(trim(strtoupper($this-> input-
951 > post('calle'))));
952         $this-> Enrolamiento_model-> setreferencia(trim(strtoupper($this-> input-
953 > post('referencia'))));
954         $this-> Enrolamiento_model-> setcolonia(trim(strtoupper($this-> input-
955 > post('colonia'))));
956         $this-> Enrolamiento_model-> setlocalidad($this-> input-> post('localidad'));
957         $this-> Enrolamiento_model-> settelefono($this-> input-> post('telefono'));
958         $this-> Enrolamiento_model-> setcompania($this-> input-> post('compania'));
959         $this-> Enrolamiento_model-> setcelular($this-> input-> post('celular'));
960         $this-> Enrolamiento_model-> setnumero($this-> input-> post('numero'));
961         $this-> Enrolamiento_model-> setcp($this-> input-> post('cp'));
962         $this-> Enrolamiento_model-> setageb(str_pad(trim(strtoupper($this-> input-
963 > post('ageb'))), 4, '0', STR_PAD_LEFT));
964         $this-> Enrolamiento_model-> setsector($this-> input-> post('sector'));
965         $this-> Enrolamiento_model-> setmanzana($this-> input-> post('manzana'));
966
967         $this-> Enrolamiento_model-> setafiliacion($this-> input-> post('afiliacion'));
968         $this-> Enrolamiento_model-> setalergias($this-> input-> post('alergia'));
969
970         $this-> Enrolamiento_model-> setvacuna($this-> input-> post('vacuna'));
971         $this-> Enrolamiento_model-> setfvacuna($this-> input-> post('fvacuna'));
972         $this-> Enrolamiento_model-> setcodigo_barras($this-> input-
973 > post('ffoliovacuna'));
974
975         $this-> Enrolamiento_model-> setconsulta($this-> input-
976 > post('id_enfermedad_consulta'));
977         $this-> Enrolamiento_model-> setfconsulta($this-> input-
978 > post('fecha_consulta'));
979         $this-> Enrolamiento_model-> settconsulta($this-> input-
980 > post('ids_tratamiento_consulta'));

```

```

961         $this-> Enrolamiento_model-> setaccion_nutricional($this-> input-
962 > post('accion_nutricional'));
963         $this-> Enrolamiento_model-> setfaccion_nutricional($this-> input-
964 > post('faccion_nutricional'));
965         $this-> Enrolamiento_model-> setpeso($this-> input-> post('cpeso'));
966         $this-> Enrolamiento_model-> setaltura($this-> input-> post('caltura'));
967         $this-> Enrolamiento_model-> settalla($this-> input-> post('ctalla'));
968         $this-> Enrolamiento_model-> sethemoglobina($this-> input-
969         $this-> Enrolamiento_model-> setfnutricion($this-> input-> post('fCNu'));
970         $this-> Enrolamiento_model-> setperi_cefa($this-> input-> post('peri_cefa'));
971         $this-> Enrolamiento_model-> setfecha_peri_cefa($this-> input-
972 > post('fecha_peri_cefa'));
973         $this-> Enrolamiento_model-> setestimulacion_fecha($this-> input-
974 > post('estimulacion_fecha'));
975         $this-> Enrolamiento_model-> setestimulacion_capacitado($this-> input-
976 > post('estimulacion_capacitado'));
977         $this-> Enrolamiento_model-> setsales_fecha($this-> input-
978 > post('sales_fecha'));
979         $this-> Enrolamiento_model-> setsales_cantidad($this-> input-
980     }
981
982 /**
983 *
984 * Valida que la curp del paciente no exista
985 *
986 * @param string $curp curp de la persona
987 * @return echo
988 */
989 public function ifCurpExists($curp)
990 {
991     $id=$this-> input-> post('id');
992     if($id!="")
993     {
994         $curp = $this-> input-> post('curp').$this-> input-> post('curp2');
995         if (empty($this-> Enrolamiento_model))
996             return false;
997         $is_exist = null;
998         try {
999             $is_exist = $this-> Enrolamiento_model-> getByCurp($curp,'cns_persona',$id);
1000         }
1001         catch(Exception $e){
1002             if ($is_exist)
1003             {
1004                 $this-> form_validation-> set_message(
1005                     'ifCurpExists', 'El curp del paciente ya existe.');
1006             }
1007             else
1008             {
1009                 if (!$this-> Enrolamiento_model-> getMsgError())
1010                     return true;
1011                 else
1012                 {
1013                     $this-> form_validation-> set_message(
1014                         'ifCurpExists', $this-> Enrolamiento_model-> getMsgError());
1015                 }
1016             }
1017         }
1018     }else return true;
1019 }
1020
1021 /**
1022 *
1023 * valida que la curp del tutor no exista
1024 *
1025 * @param string $curp curp de la persona
1026 * @return echo
1027 */
1028 public function ifCurpTExists($curp)
1029 {
1030     $id=$this-> input-> post('idtutor');
1031     if($id!="")
1032 }
```

```

1033     {
1034         if (empty($this-> Enrolamiento_model))
1035             return false;
1036         $is_exist = null;
1037         try {
1038             $is_exist = $this-> Enrolamiento_model-> getByCurp($curp, 'cns_tutor', $id);
1039         }
1040         catch(Exception $e){
1041         }
1042         if ($is_exist)
1043         {
1044             $this-> form_validation-> set_message(
1045                 'ifCurpTExists', 'El curp del tutor ya existe.');
1046             return false;
1047         }
1048         else
1049         {
1050             if (!$this-> Enrolamiento_model-> getMsgError())
1051                 return true;
1052             else{
1053                 $this-> form_validation-> set_message(
1054                     'ifCurpTExists', $this-> Enrolamiento_model-> getMsgError());
1055                 return false;
1056             }
1057         }
1058     }
1059 }
1060 /**
1061 *
1062 * Este metodo verifica si un paciente comparte un mismo tutor
1063 *
1064 * @param string $id id de la persona
1065 * @return echo
1066 */
1067 public function brother_found($id)
1068 {
1069     $this-> load-> model(DIR_TES.'/Reporte_sincronizacion_model');
1070     $result=$this-> Reporte_sincronizacion_model-> getListado(" SELECT p.id,
1071     UPPER(CONCAT(p.nombre, ' ', p.apellido_paterno, ' ', p.apellido_materno)) AS nombre,
1072     p.calle_domicilio, p.numero_domicilio,
1073     p.preferencia_domicilio, p.colonia_domicilio, p.cp_domicilio, p.ageb, p.sector, p.manzana,
1074     p.id_asu_localidad_domicilio, p.telefono_domicilio
1075   FROM cns_persona p WHERE p.id='".$id"' );
1076     echo json_encode($result);
1077 }
1078 /**
1079 *
1080 * Este metodo extrae la informacion de las personas con las que se comparte el mismo tutor
1081 * si se selecciona una de estas importa los datos para el apartado direccion
1082 *
1083 * @param string $tutor id del tutor compartido
1084 * @return echo
1085 */
1086 public function brothers_search($tutor)
1087 {
1088     $this-> load-> model(DIR_TES.'/Reporte_sincronizacion_model');
1089     $result=$this-> Reporte_sincronizacion_model-> getListado(" SELECT DISTINCT
1090     t.id_persona, UPPER(CONCAT(p.nombre, ' ', p.apellido_paterno, ' ', p.apellido_materno)) AS nombre,
1091     p.calle_domicilio, p.numero_domicilio,
1092     p.preferencia_domicilio, p.colonia_domicilio, p.cp_domicilio, p.ageb, p.sector, p.manzana,
1093     p.id_asu_localidad_domicilio, p.telefono_domicilio
1094   FROM cns_persona_x_tutor t
1095   LEFT JOIN cns_persona p ON p.id=t.id_persona
1096   WHERE t.id_tutor='".$tutor' and t.id_tutor!='ffec1916fae9ee3q3ala98f0a7b31400'" );
1097     echo json_encode($result);
1098 }
1099 /**
1100 *
1101 * valida que el nodo seleccionado en el arbol sea una unidad medica
1102 *
1103 * @param string $id id de arbol de segmentacion
1104 * @return echo

```

```

1106 */
1107 public function validarism($id)
1108 {
1109     $this-> load-> model(DIR_TES.'/Reporte_sincronizacion_model');
1110     $result=$this-> Reporte_sincronizacion_model-> getListado("SELECT
1111 grado_segmentacion FROMasu_arbol_segmentacion WHERE id=''" . $id. "'");
1112     if($result[0]-> grado_segmentacion!="5")
1113         echo "no";
1114 }
1115 /**
1116 *
1117 * Busca dentro del catalogoasu_ageb la unidad médica de acuerdo a la localidad y ageb
1118 * Solo se permite su acceso por medio de peticiones AJAX
1119 *
1120 * @param int $idasulocalidad Id de la localidad en el ASU
1121 * @param string $ageb Número de ageb
1122 *
1123 * @return Object
1124 * (key,value)
1125 * (-1 , Clues) = La clues existe en el catalogo AGEB pero no está registrada en
nuestro arbol ASU
1126 * (0,0) = No existe ninguna unidad medica con esa localidad y ageb
1127 * (1,Clues) = Si existe en el catalogo AGEB y en el ASU
1128 */
1129 public function searchum($idasulocalidad,$ageb)
1130 {
1131     if (!$this-> input-> is_ajax_request())
1132         show_error('', 403, 'Acceso denegado');
1133
1134     $this-> load-> model(DIR_SIIGS.'/Ageb_model');
1135     $result=$this-> Ageb_model-> searchUM($idasulocalidad,$ageb);
1136
1137     if ($result == -1)
1138         echo json_encode (array("clave" => 0,"valor" => 0));
1139     else if ($result == 0) {
1140         echo json_encode (array("clave" => -1,"valor" => $result));
1141     }
1142     else{
1143         echo json_encode (array("clave" => 1,"valor" => $result));
1144     }
1145 }
1146
1147 /**
1148 *
1149 * Regresa un objeto JSON con la lista de agebs disponibles en la localidad
1150 * Solo se permite su acceso por medio de peticiones AJAX
1151 *
1152 * @param int $idasulocalidad Id de la localidad en el ASU
1153 *
1154 * @return Object
1155 */
1156 public function searchageb($idasulocalidad="" ,$like="")
1157 {
1158     if (!$this-> input-> is_ajax_request())
1159         show_error('', 403, 'Acceso denegado');
1160
1161     $this-> load-> model(DIR_SIIGS.'/Ageb_model');
1162     $result=$this-> Ageb_model-> searchageb($idasulocalidad,$like);
1163     $dato = array();
1164     foreach ($result as $item)
1165         $dato[] = $item-> ageb;
1166     echo json_encode ($dato);
1167 }
1168
1169 /**
1170 *
1171 * Comprueba la similitud de un paciente que se este capturando con los que ya existe en la
base de datos,
1172 * esto con la finalidad de disminuir datos repetidos
1173 *
1174 * @param string $nombre nombre del paciente que se esta capturando
1175 * @param string $paterno apellido paterno del paciente
1176 * @param string $materno apellido materno del paciente
1177 * @param string $curp curp del paciente
1178 * @param string $nacimiento fecha de nacimiento del paciente
1179 * @param string $calle calle del domicilio del paciente
1180
1181
1182

```

```

1183     * @param      string      $referencia  referencia del domicilio del paciente
1184     * @param      string      $colonia    colonia del paciente
1185     * @param      string      $cp        cp de la colonia donde vive el paciente
1186     * @param      string      $numero    numero de la vivienda
1187   *
1188   * @return      json($porcentaje_similitud,$persona)
1189   */
1190  public function paciente_similar($nombre, $paterno, $materno, $curp, $nacimiento, $lugar,
1191  $calle="" , $referencia="" , $colonia="" , $curpT="")
1192  {
1193      $this-> load-> model(DIR_TES.'/Enrolamiento_model');
1194      $result=$this-> Enrolamiento_model-> get_pacientes();
1195
1196      $array=array();
1197      if($result)
1198      {
1199          foreach($result as $x)
1200          {
1201              $similar=0;
1202              similar_text(urldecode($nombre), $x-> nombre, $percent);
1203              $similar=$similar+$percent;
1204
1205              similar_text(urldecode($paterno), $x-> apellido_paterno, $percent);
1206              $similar=$similar+$percent;
1207
1208              similar_text(urldecode($materno), $x-> apellido_materno, $percent);
1209              $similar=$similar+$percent;
1210
1211              similar_text(urldecode($curp), $x-> curp, $percent);
1212              $similar=$similar+$percent;
1213
1214              similar_text(urldecode($calle), $x-> calle_domicilio, $percent);
1215              $similar=$similar+$percent;
1216
1217              similar_text(urldecode($colonia), $x-> colonia_domicilio, $percent);
1218              $similar=$similar+$percent;
1219
1220              similar_text(urldecode($lugar), $x-> lugar, $percent);
1221              $similar=$similar+$percent;
1222
1223              similar_text(urldecode($curpT), $x-> curpT, $percent);
1224              $similar=$similar+$percent;
1225
1226              $total=$similar/8;
1227              if($total> 50&& date('Y-m-d', strtotime($nacimiento)) == $x-> fecha_nacimiento) || ($total> 92)
1228                  $array[]=$array("nombre" => $x-> nombre.' '.$x-> apellido_paterno.' '.$x-> apellido_materno, "id" => $x-> id, "total" => round($total, 2));
1229
1230          }
1231      echo json_encode($array);
1232  }
1233 /**
1234 *
1235 * Crea la pagina para ver la infromacion de la persona y comprrarla con la persona
1236 * capturada
1237 * @param      string      $id        identificador de la persona
1238 *
1239 * @return      echo
1240 */
1241 public function
comparar_view($id,$prod1="" , $prod2="" , $prod3="")
1242 {
1243     try
1244     {
1245         $this-> load-> model(DIR_TES.'/Enrolamiento_model');
1246         $this-> load-> model(DIR_TES.'/Reporte_sincronizacion_model');
1247         if (empty($this-> Enrolamiento_model))
1248             return false;
1249
1250         $data['title'] = 'Ver Paciente';
1251         $data['enrolado'] = $this-> Enrolamiento_model-> getById($id);
1252         if(empty($data['enrolado']))
1253         {
1254             $data['infoclass'] = 'error';
1255             $data['msgResult'] = "Registro no encontrado";
1256         }

```

```

1257     $this-> template-
1258 > write_view('content',DIR_TES.'/enrolamiento/enrolamiento_compara', $data);
1259     $this-> template-> render();
1260     return true;
1261 }
1262 $prod1=urldecode($prod1);
1263 $data['prod1']=explode("°", $prod1);
1264
1265 $prod2=urldecode($prod2);
1266 $data['prod2']=explode("°", $prod2);
1267
1268 $prod3=urldecode($prod3);
1269 $data['prod3']=explode("°", $prod3);
1270 }
1271 catch(Exception $e)
1272 {
1273     $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
1274
1275     $this-> template-> write('header','');
1276     $this-> template-> write('footer','');
1277     $this-> template-> write('menu','');
1278     $this-> template-> write('sala_prensa','');
1279
1280 > write_view('content',DIR_TES.'/enrolamiento/enrolamiento_compara', $data);
1281     $this-> template-> render();
1282 }
1283 /**
1284 * Obtiene el historial de vacunacion de un paciente cuyo id es pasado por parametro
1285 *
1286 * @param      date      $fecha
1287 *             @param      string      $id
1288 *
1289 * @return     echo
1290 */
1291 public function vacunacion($fecha,$id="")
1292 {
1293     $fecha=date("Y-m-d", strtotime($fecha));
1294     $this-> load-> model(DIR_TES.'/Reporte_sincronizacion_model');
1295     //echo "SELECT DISTINCT r.id_vacuna,cv.codigo_barras,
1296 v.descripcion,r.dia_inicio_aplicacion_nacido, r.dia_fin_aplicacion_nacido, p.fecha_nacimiento, CASE
1297 WHEN r.id_vacuna = cv.id_vacuna THEN 'X' ELSE '' END AS tiene, CASE WHEN cv.fecha IS NULL THEN
1298 CONCAT('Desde:',r.dia_inicio_aplicacion_nacido,' Hasta:',r.dia_fin_aplicacion_nacido) ELSE
1299 CONCAT('Fecha Aplicada: ',' ',DATE_FORMAT(cv.fecha, '%d-%m-%Y')) END AS
1300 fecha,DATEDIFF(NOW(),'$fecha') AS dias,CASE WHEN
1301 DATEDIFF(NOW(),'$fecha')>=r.dia_inicio_aplicacion_nacido AND
1302 DATEDIFF(NOW(),'$fecha')<=r.dia_fin_aplicacion_nacido THEN '1' ELSE (CASE WHEN
1303 DATEDIFF(NOW(),'$fecha')>r.dia_fin_aplicacion_nacido AND cv.fecha IS NULL THEN '2' ELSE (CASE WHEN
1304 DATEDIFF(NOW(),'$fecha')< r.dia_inicio_aplicacion_nacido AND cv.fecha IS NULL THEN '3' END) END)
1305 END AS prioridad FROM cns_regla_vacuna r LEFT JOIN cns_vacuna v ON v.id=r.id_vacuna LEFT JOIN
1306 cns_control_vacuna cv ON cv.id_persona='$id' AND cv.id_vacuna=r.id_vacuna LEFT JOIN cns_persona p ON
1307 p.id=cv.id_persona GROUP BY v.descripcion ORDER BY r.id_vacuna,r.orden_esq_com ASC";
1308     /die();
1309     $data['vacunacion']=$this-> Reporte_sincronizacion_model-> getListado(
1310     SELECT DISTINCT r.id_vacuna,cv.codigo_barras, v.descripcion,r.dia_inicio_aplicacion_nacido,
1311 r.dia_fin_aplicacion_nacido, p.fecha_nacimiento, CASE WHEN r.id_vacuna = cv.id_vacuna THEN 'X' ELSE
1312 '' END AS tiene, CASE WHEN cv.fecha IS NULL THEN CONCAT('Desde:',r.dia_inicio_aplicacion_nacido,' Hasta:',r.dia_fin_aplicacion_nacido) ELSE CONCAT('Fecha Aplicada: ',' ',DATE_FORMAT(cv.fecha, '%d-%m-%Y')) END AS fecha,DATEDIFF(NOW(),'$fecha') AS dias,CASE WHEN
1313 DATEDIFF(NOW(),'$fecha')>=r.dia_inicio_aplicacion_nacido AND
1314 DATEDIFF(NOW(),'$fecha')<=r.dia_fin_aplicacion_nacido THEN '1' ELSE (CASE WHEN
1315 DATEDIFF(NOW(),'$fecha')>r.dia_fin_aplicacion_nacido AND cv.fecha IS NULL THEN '2' ELSE (CASE WHEN
1316 DATEDIFF(NOW(),'$fecha')< r.dia_inicio_aplicacion_nacido AND cv.fecha IS NULL THEN '3' END) END)
1317 END AS prioridad FROM cns_regla_vacuna r LEFT JOIN cns_vacuna v ON v.id=r.id_vacuna LEFT JOIN
1318 cns_control_vacuna cv ON cv.id_persona='$id' AND cv.id_vacuna=r.id_vacuna LEFT JOIN cns_persona p ON
1319 p.id=cv.id_persona GROUP BY v.descripcion ORDER BY r.id_vacuna,r.orden_esq_com ASC" );
1320     $data["fecha"]           = $fecha;
1321     $data['id_x']=$id;
1322     $this-> load-> view(DIR_TES.'/enrolamiento/vacuna', $data);
1323 }
1324 /**
1325 *
1326 * Revisa si la sesion està activa
1327 *
1328 * @return bool
1329 */
1330 public function checar_session()
1331 {

```

```

1312     $this-> load-> library('session');
1313     if ($this-> session-> userdata(GROUP_ID)=="") ;
1314         echo "no" ;
1315     else
1316         echo "si" ;
1317     }
1318
1319     /*
1320     public function accion()
1321     {
1322         $data['prefix']='hola';
1323         $this->load->view(DIR_TES.'/TESNFC/Web/index',$data);
1324     }
1325     */
1326
1327 /**
1328 *
1329 * Genera los options de un campo tipo select para las categorías de CIE10
1330 *
1331 * @return
1332 */
1333 public function categoriacie10_select()
1334 {
    $this-> load-> model(DIR_TES.'/Enrolamiento_model');
1335
    $opcion = "";
    $datos = $this-> Enrolamiento_model-> getCategoriaCIE10();
1339
    if(sizeof($datos) != 0)
    {
        $opcion .= "<option value='>Seleccione...</option>";
        foreach($datos as $dato)
        {
            $opcion .= "      <option value='".$dato-> id' >    $dato-
> descripcion</option>      ";
        }
        echo $opcion;
    }
    else
        echo "<option>No hay Datos</option>";
}
1352
1353 /**
1354 *
1355 * Genera los options de un campo tipo select para los CIE10 correspondientes a una categoría
1356 *
1357 * @param      string      $categoria  Categoría de la CIE10
1358 * @return
1359 */
1360 public function cie10_select($categoria)
1361 {
    $this-> load-> model(DIR_TES.'/Enrolamiento_model');
1363
    $opcion = "";
    $datos = $this-> Enrolamiento_model-> getCIE10($categoria);
1366
    if(sizeof($datos) != 0)
    {
        $opcion .= "<option value='>Seleccione...</option>";
        foreach($datos as $dato)
        {
            $opcion .= "      <option value='".$dato-> id_cielo' >    $dato-
> descripcion</option>      ";
        }
        echo $opcion;
    }
    else
        echo "<option>No hay Datos</option>";
}
1378
1379 }
1380 }
```

# Archivo fuente para notificacion.php

La documentación para este archivo está disponible en [notificacion.php](#)

```
1  <?php
2  /**
3   * Controlador Notificación
4   *
5   * @package      TES
6   * @subpackage   Controlador
7   * @author       Rogelio
8   * @created      2013-11-26
9   */
10  class Notificacion extends CI_Controller {
11
12      public function __construct()
13      {
14          parent::__construct();
15          try{
16              $this-> load-> helper('url');
17              $this-> load-> model(DIR_TES.'/Notificacion_model');
18          }
19          catch(Exception $e)
20          {
21              $this-> template-> write("content" , $e-> getMessage());
22              $this-> template-> render();
23          }
24      }
25
26      /**
27       * 1) Visualiza las notificaciones existentes para su interacción CRUD
28       * 2) En caso de detectar un texto a buscar se filtran las notificaciones existentes acorde
a la búsqueda
29       *
30       * @access      public
31       * @param       int      $pag      número de página a visualizar (paginación)
32       * @return      void
33       */
34      public function index($pag = 0)
35      {
36          try{
37              if (empty($this-> Notificacion_model))
38                  return false;
39              if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url()))
40                  show_error('', 403, 'Acceso denegado');
41              $data['title'] = 'Catálogo de Notificaciones';
42              $this-> load-> helper('form');
43              $this-> load-> library('pagination');
44
45              $data['pag'] = $pag;
46              $data['msgResult'] = $this-> session-> flashdata('msgResult');
47              $data['clsResult'] = $this-> session-> flashdata('clsResult');
48
49              if($this-> input-> post('filtrar')) {
50                  // Eliminar el campo hidden y el botón
51                  unset($_POST['filtrar'], $_POST['btnFiltrar']);
52                  $filtros = array_filter($this-> input-> post());
53
54                  if(!empty($filtros)) {
55                      foreach ($filtros as $campo => $valor) {
56                          switch ($campo) {
57                              case 'fechaIni':
58                                  $this-> Notificacion_model-> addFilter('fecha_inicio',
'>=' , date('Y-m-d', strtotime($valor)));
59                                  break;
60                              case 'fechaFin':
61                                  $this-> Notificacion_model-> addFilter('fecha_inicio',
'<=' , date('Y-m-d', strtotime($valor)).' 23:59:59');
62                                  break;
63                          }
64                      }
65                  }
66              }
67          }
68      }
69
70      public function filtrar()
71      {
72          $data['title'] = 'Filtrar Notificaciones';
73
74          $this-> load-> helper('form');
75          $this-> load-> library('pagination');
76
77          $data['filtros'] = $this-> input-> post();
78
79          if(!empty($data['filtros'])) {
80              foreach ($data['filtros'] as $campo => $valor) {
81                  switch ($campo) {
82                      case 'fechaIni':
83                          $this-> Notificacion_model-> addFilter('fecha_inicio',
'>=' , date('Y-m-d', strtotime($valor)));
84                          break;
85                      case 'fechaFin':
86                          $this-> Notificacion_model-> addFilter('fecha_inicio',
'<=' , date('Y-m-d', strtotime($valor)).' 23:59:59');
87                          break;
88                  }
89              }
90          }
91
92          $this-> template-> write("content" , $this-> Notificacion_model-> getNotificaciones());
93          $this-> template-> render();
94      }
95
96      public function insertar()
97      {
98          $data['title'] = 'Nuevo Notificación';
99
100         $this-> load-> helper('form');
101
102         $data['filtros'] = $this-> input-> post();
103
104         $this-> template-> write("content" , $this-> Notificacion_model-> insertar());
105         $this-> template-> render();
106     }
107
108     public function editar($id)
109     {
110         $data['title'] = 'Actualizar Notificación';
111
112         $this-> load-> helper('form');
113
114         $data['filtros'] = $this-> input-> post();
115
116         $this-> template-> write("content" , $this-> Notificacion_model-> editar($id));
117         $this-> template-> render();
118     }
119
120     public function eliminar($id)
121     {
122         $data['title'] = 'Borrar Notificación';
123
124         $this-> load-> helper('form');
125
126         $data['filtros'] = $this-> input-> post();
127
128         $this-> template-> write("content" , $this-> Notificacion_model-> eliminar($id));
129         $this-> template-> render();
130     }
131
132     public function detalle($id)
133     {
134         $data['title'] = 'Ver Notificación';
135
136         $this-> load-> helper('form');
137
138         $data['filtros'] = $this-> input-> post();
139
140         $this-> template-> write("content" , $this-> Notificacion_model-> detalle($id));
141         $this-> template-> render();
142     }
143
144     public function reporte()
145     {
146         $data['title'] = 'Reporte Notificación';
147
148         $this-> load-> helper('form');
149
150         $data['filtros'] = $this-> input-> post();
151
152         $this-> template-> write("content" , $this-> Notificacion_model-> reporte());
153         $this-> template-> render();
154     }
155
156     public function exportar()
157     {
158         $data['title'] = 'Exportar Notificación';
159
160         $this-> load-> helper('form');
161
162         $data['filtros'] = $this-> input-> post();
163
164         $this-> template-> write("content" , $this-> Notificacion_model-> exportar());
165         $this-> template-> render();
166     }
167
168     public function imprimir()
169     {
170         $data['title'] = 'Imprimir Notificación';
171
172         $this-> load-> helper('form');
173
174         $data['filtros'] = $this-> input-> post();
175
176         $this-> template-> write("content" , $this-> Notificacion_model-> imprimir());
177         $this-> template-> render();
178     }
179
180     public function pdf()
181     {
182         $data['title'] = 'PDF Notificación';
183
184         $this-> load-> helper('form');
185
186         $data['filtros'] = $this-> input-> post();
187
188         $this-> template-> write("content" , $this-> Notificacion_model-> pdf());
189         $this-> template-> render();
190     }
191
192     public function excel()
193     {
194         $data['title'] = 'Excel Notificación';
195
196         $this-> load-> helper('form');
197
198         $data['filtros'] = $this-> input-> post();
199
200         $this-> template-> write("content" , $this-> Notificacion_model-> excel());
201         $this-> template-> render();
202     }
203
204     public function csv()
205     {
206         $data['title'] = 'CSV Notificación';
207
208         $this-> load-> helper('form');
209
210         $data['filtros'] = $this-> input-> post();
211
212         $this-> template-> write("content" , $this-> Notificacion_model-> csv());
213         $this-> template-> render();
214     }
215
216     public function json()
217     {
218         $data['title'] = 'JSON Notificación';
219
220         $this-> load-> helper('form');
221
222         $data['filtros'] = $this-> input-> post();
223
224         $this-> template-> write("content" , $this-> Notificacion_model-> json());
225         $this-> template-> render();
226     }
227
228     public function xml()
229     {
230         $data['title'] = 'XML Notificación';
231
232         $this-> load-> helper('form');
233
234         $data['filtros'] = $this-> input-> post();
235
236         $this-> template-> write("content" , $this-> Notificacion_model-> xml());
237         $this-> template-> render();
238     }
239
240     public function rss()
241     {
242         $data['title'] = 'RSS Notificación';
243
244         $this-> load-> helper('form');
245
246         $data['filtros'] = $this-> input-> post();
247
248         $this-> template-> write("content" , $this-> Notificacion_model-> rss());
249         $this-> template-> render();
250     }
251
252     public function sitemap()
253     {
254         $data['title'] = 'Sitemap Notificación';
255
256         $this-> load-> helper('form');
257
258         $data['filtros'] = $this-> input-> post();
259
260         $this-> template-> write("content" , $this-> Notificacion_model-> sitemap());
261         $this-> template-> render();
262     }
263
264     public function robots()
265     {
266         $data['title'] = 'Robots Notificación';
267
268         $this-> load-> helper('form');
269
270         $data['filtros'] = $this-> input-> post();
271
272         $this-> template-> write("content" , $this-> Notificacion_model-> robots());
273         $this-> template-> render();
274     }
275
276     public function sitemap_index()
277     {
278         $data['title'] = 'Sitemap Index Notificación';
279
280         $this-> load-> helper('form');
281
282         $data['filtros'] = $this-> input-> post();
283
284         $this-> template-> write("content" , $this-> Notificacion_model-> sitemap_index());
285         $this-> template-> render();
286     }
287
288     public function robots_index()
289     {
290         $data['title'] = 'Robots Index Notificación';
291
292         $this-> load-> helper('form');
293
294         $data['filtros'] = $this-> input-> post();
295
296         $this-> template-> write("content" , $this-> Notificacion_model-> robots_index());
297         $this-> template-> render();
298     }
299
300     public function sitemap_change()
301     {
302         $data['title'] = 'Sitemap Change Notificación';
303
304         $this-> load-> helper('form');
305
306         $data['filtros'] = $this-> input-> post();
307
308         $this-> template-> write("content" , $this-> Notificacion_model-> sitemap_change());
309         $this-> template-> render();
310     }
311
312     public function robots_change()
313     {
314         $data['title'] = 'Robots Change Notificación';
315
316         $this-> load-> helper('form');
317
318         $data['filtros'] = $this-> input-> post();
319
320         $this-> template-> write("content" , $this-> Notificacion_model-> robots_change());
321         $this-> template-> render();
322     }
323
324     public function sitemap_error()
325     {
326         $data['title'] = 'Sitemap Error Notificación';
327
328         $this-> load-> helper('form');
329
330         $data['filtros'] = $this-> input-> post();
331
332         $this-> template-> write("content" , $this-> Notificacion_model-> sitemap_error());
333         $this-> template-> render();
334     }
335
336     public function robots_error()
337     {
338         $data['title'] = 'Robots Error Notificación';
339
340         $this-> load-> helper('form');
341
342         $data['filtros'] = $this-> input-> post();
343
344         $this-> template-> write("content" , $this-> Notificacion_model-> robots_error());
345         $this-> template-> render();
346     }
347
348     public function sitemap_lastmod()
349     {
350         $data['title'] = 'Sitemap Lastmod Notificación';
351
352         $this-> load-> helper('form');
353
354         $data['filtros'] = $this-> input-> post();
355
356         $this-> template-> write("content" , $this-> Notificacion_model-> sitemap_lastmod());
357         $this-> template-> render();
358     }
359
360     public function robots_lastmod()
361     {
362         $data['title'] = 'Robots Lastmod Notificación';
363
364         $this-> load-> helper('form');
365
366         $data['filtros'] = $this-> input-> post();
367
368         $this-> template-> write("content" , $this-> Notificacion_model-> robots_lastmod());
369         $this-> template-> render();
370     }
371
372     public function sitemap_gzip()
373     {
374         $data['title'] = 'Sitemap Gzip Notificación';
375
376         $this-> load-> helper('form');
377
378         $data['filtros'] = $this-> input-> post();
379
380         $this-> template-> write("content" , $this-> Notificacion_model-> sitemap_gzip());
381         $this-> template-> render();
382     }
383
384     public function robots_gzip()
385     {
386         $data['title'] = 'Robots Gzip Notificación';
387
388         $this-> load-> helper('form');
389
390         $data['filtros'] = $this-> input-> post();
391
392         $this-> template-> write("content" , $this-> Notificacion_model-> robots_gzip());
393         $this-> template-> render();
394     }
395
396     public function sitemap_gz()
397     {
398         $data['title'] = 'Sitemap Gz Notificación';
399
400         $this-> load-> helper('form');
401
402         $data['filtros'] = $this-> input-> post();
403
404         $this-> template-> write("content" , $this-> Notificacion_model-> sitemap_gz());
405         $this-> template-> render();
406     }
407
408     public function robots_gz()
409     {
410         $data['title'] = 'Robots Gz Notificación';
411
412         $this-> load-> helper('form');
413
414         $data['filtros'] = $this-> input-> post();
415
416         $this-> template-> write("content" , $this-> Notificacion_model-> robots_gz());
417         $this-> template-> render();
418     }
419
420     public function sitemap_xhtml()
421     {
422         $data['title'] = 'Sitemap Xhtml Notificación';
423
424         $this-> load-> helper('form');
425
426         $data['filtros'] = $this-> input-> post();
427
428         $this-> template-> write("content" , $this-> Notificacion_model-> sitemap_xhtml());
429         $this-> template-> render();
430     }
431
432     public function robots_xhtml()
433     {
434         $data['title'] = 'Robots Xhtml Notificación';
435
436         $this-> load-> helper('form');
437
438         $data['filtros'] = $this-> input-> post();
439
440         $this-> template-> write("content" , $this-> Notificacion_model-> robots_xhtml());
441         $this-> template-> render();
442     }
443
444     public function sitemap_xml()
445     {
446         $data['title'] = 'Sitemap Xml Notificación';
447
448         $this-> load-> helper('form');
449
450         $data['filtros'] = $this-> input-> post();
451
452         $this-> template-> write("content" , $this-> Notificacion_model-> sitemap_xml());
453         $this-> template-> render();
454     }
455
456     public function robots_xml()
457     {
458         $data['title'] = 'Robots Xml Notificación';
459
460         $this-> load-> helper('form');
461
462         $data['filtros'] = $this-> input-> post();
463
464         $this-> template-> write("content" , $this-> Notificacion_model-> robots_xml());
465         $this-> template-> render();
466     }
467
468     public function sitemap_rss()
469     {
470         $data['title'] = 'Sitemap Rss Notificación';
471
472         $this-> load-> helper('form');
473
474         $data['filtros'] = $this-> input-> post();
475
476         $this-> template-> write("content" , $this-> Notificacion_model-> sitemap_rss());
477         $this-> template-> render();
478     }
479
480     public function robots_rss()
481     {
482         $data['title'] = 'Robots Rss Notificación';
483
484         $this-> load-> helper('form');
485
486         $data['filtros'] = $this-> input-> post();
487
488         $this-> template-> write("content" , $this-> Notificacion_model-> robots_rss());
489         $this-> template-> render();
490     }
491
492     public function sitemap_atom()
493     {
494         $data['title'] = 'Sitemap Atom Notificación';
495
496         $this-> load-> helper('form');
497
498         $data['filtros'] = $this-> input-> post();
499
500         $this-> template-> write("content" , $this-> Notificacion_model-> sitemap_atom());
501         $this-> template-> render();
502     }
503
504     public function robots_atom()
505     {
506         $data['title'] = 'Robots Atom Notificación';
507
508         $this-> load-> helper('form');
509
510         $data['filtros'] = $this-> input-> post();
511
512         $this-> template-> write("content" , $this-> Notificacion_model-> robots_atom());
513         $this-> template-> render();
514     }
515
516     public function sitemap_json()
517     {
518         $data['title'] = 'Sitemap Json Notificación';
519
520         $this-> load-> helper('form');
521
522         $data['filtros'] = $this-> input-> post();
523
524         $this-> template-> write("content" , $this-> Notificacion_model-> sitemap_json());
525         $this-> template-> render();
526     }
527
528     public function robots_json()
529     {
530         $data['title'] = 'Robots Json Notificación';
531
532         $this-> load-> helper('form');
533
534         $data['filtros'] = $this-> input-> post();
535
536         $this-> template-> write("content" , $this-> Notificacion_model-> robots_json());
537         $this-> template-> render();
538     }
539
540     public function sitemap_xml_gz()
541     {
542         $data['title'] = 'Sitemap Xml Gz Notificación';
543
544         $this-> load-> helper('form');
545
546         $data['filtros'] = $this-> input-> post();
547
548         $this-> template-> write("content" , $this-> Notificacion_model-> sitemap_xml_gz());
549         $this-> template-> render();
550     }
551
552     public function robots_xml_gz()
553     {
554         $data['title'] = 'Robots Xml Gz Notificación';
555
556         $this-> load-> helper('form');
557
558         $data['filtros'] = $this-> input-> post();
559
560         $this-> template-> write("content" , $this-> Notificacion_model-> robots_xml_gz());
561         $this-> template-> render();
562     }
563
564     public function sitemap_xhtml_gz()
565     {
566         $data['title'] = 'Sitemap Xhtml Gz Notificación';
567
568         $this-> load-> helper('form');
569
570         $data['filtros'] = $this-> input-> post();
571
572         $this-> template-> write("content" , $this-> Notificacion_model-> sitemap_xhtml_gz());
573         $this-> template-> render();
574     }
575
576     public function robots_xhtml_gz()
577     {
578         $data['title'] = 'Robots Xhtml Gz Notificación';
579
580         $this-> load-> helper('form');
581
582         $data['filtros'] = $this-> input-> post();
583
584         $this-> template-> write("content" , $this-> Notificacion_model-> robots_xhtml_gz());
585         $this-> template-> render();
586     }
587
588     public function sitemap_xhtml_atom()
589     {
590         $data['title'] = 'Sitemap Xhtml Atom Notificación';
591
592         $this-> load-> helper('form');
593
594         $data['filtros'] = $this-> input-> post();
595
596         $this-> template-> write("content" , $this-> Notificacion_model-> sitemap_xhtml_atom());
597         $this-> template-> render();
598     }
599
600     public function robots_xhtml_atom()
601     {
602         $data['title'] = 'Robots Xhtml Atom Notificación';
603
604         $this-> load-> helper('form');
605
606         $data['filtros'] = $this-> input-> post();
607
608         $this-> template-> write("content" , $this-> Notificacion_model-> robots_xhtml_atom());
609         $this-> template-> render();
610     }
611
612     public function sitemap_xhtml_rss()
613     {
614         $data['title'] = 'Sitemap Xhtml Rss Notificación';
615
616         $this-> load-> helper('form');
617
618         $data['filtros'] = $this-> input-> post();
619
620         $this-> template-> write("content" , $this-> Notificacion_model-> sitemap_xhtml_rss());
621         $this-> template-> render();
622     }
623
624     public function robots_xhtml_rss()
625     {
626         $data['title'] = 'Robots Xhtml Rss Notificación';
627
628         $this-> load-> helper('form');
629
630         $data['filtros'] = $this-> input-> post();
631
632         $this-> template-> write("content" , $this-> Notificacion_model-> robots_xhtml_rss());
633         $this-> template-> render();
634     }
635
636     public function sitemap_xhtml_atom_gz()
637     {
638         $data['title'] = 'Sitemap Xhtml Atom Gz Notificación';
639
640         $this-> load-> helper('form');
641
642         $data['filtros'] = $this-> input-> post();
643
644         $this-> template-> write("content" , $this-> Notificacion_model-> sitemap_xhtml_atom_gz());
645         $this-> template-> render();
646     }
647
648     public function robots_xhtml_atom_gz()
649     {
650         $data['title'] = 'Robots Xhtml Atom Gz Notificación';
651
652         $this-> load-> helper('form');
653
654         $data['filtros'] = $this-> input-> post();
655
656         $this-> template-> write("content" , $this-> Notificacion_model-> robots_xhtml_atom_gz());
657         $this-> template-> render();
658     }
659
660     public function sitemap_xhtml_rss_gz()
661     {
662         $data['title'] = 'Sitemap Xhtml Rss Gz Notificación';
663
664         $this-> load-> helper('form');
665
666         $data['filtros'] = $this-> input-> post();
667
668         $this-> template-> write("content" , $this-> Notificacion_model-> sitemap_xhtml_rss_gz());
669         $this-> template-> render();
670     }
671
672     public function robots_xhtml_rss_gz()
673     {
674         $data['title'] = 'Robots Xhtml Rss Gz Notificación';
675
676         $this-> load-> helper('form');
677
678         $data['filtros'] = $this-> input-> post();
679
680         $this-> template-> write("content" , $this-> Notificacion_model-> robots_xhtml_rss_gz());
681         $this-> template-> render();
682     }
683
684     public function sitemap_xhtml_atom_gz_atom()
685     {
686         $data['title'] = 'Sitemap Xhtml Atom Gz Atom Notificación';
687
688         $this-> load-> helper('form');
689
690         $data['filtros'] = $this-> input-> post();
691
692         $this-> template-> write("content" , $this-> Notificacion_model-> sitemap_xhtml_atom_gz_atom());
693         $this-> template-> render();
694     }
695
696     public function robots_xhtml_atom_gz_atom()
697     {
698         $data['title'] = 'Robots Xhtml Atom Gz Atom Notificación';
699
700         $this-> load-> helper('form');
701
702         $data['filtros'] = $this-> input-> post();
703
704         $this-> template-> write("content" , $this-> Notificacion_model-> robots_xhtml_atom_gz_atom());
705         $this-> template-> render();
706     }
707
708     public function sitemap_xhtml_rss_gz_atom()
709     {
710         $data['title'] = 'Sitemap Xhtml Rss Gz Atom Notificación';
711
712         $this-> load-> helper('form');
713
714         $data['filtros'] = $this-> input-> post();
715
716         $this-> template-> write("content" , $this-> Notificacion_model-> sitemap_xhtml_rss_gz_atom());
717         $this-> template-> render();
718     }
719
720     public function robots_xhtml_rss_gz_atom()
721     {
722         $data['title'] = 'Robots Xhtml Rss Gz Atom Notificación';
723
724         $this-> load-> helper('form');
725
726         $data['filtros'] = $this-> input-> post();
727
728         $this-> template-> write("content" , $this-> Notificacion_model-> robots_xhtml_rss_gz_atom());
729         $this-> template-> render();
730     }
731
732     public function sitemap_xhtml_atom_gz_atom_gz()
733     {
734         $data['title'] = 'Sitemap Xhtml Atom Gz Atom Gz Notificación';
735
736         $this-> load-> helper('form');
737
738         $data['filtros'] = $this-> input-> post();
739
740         $this-> template-> write("content" , $this-> Notificacion_model-> sitemap_xhtml_atom_gz_atom_gz());
741         $this-> template-> render();
742     }
743
744     public function robots_xhtml_atom_gz_atom_gz()
745     {
746         $data['title'] = 'Robots Xhtml Atom Gz Atom Gz Notificación';
747
748         $this-> load-> helper('form');
749
750         $data['filtros'] = $this-> input-> post();
751
752         $this-> template-> write("content" , $this-> Notificacion_model-> robots_xhtml_atom_gz_atom_gz());
753         $this-> template-> render();
754     }
755
756     public function sitemap_xhtml_rss_gz_atom_gz()
757     {
758         $data['title'] = 'Sitemap Xhtml Rss Gz Atom Gz Notificación';
759
760         $this-> load-> helper('form');
761
762         $data['filtros'] = $this-> input-> post();
763
764         $this-> template-> write("content" , $this-> Notificacion_model-> sitemap_xhtml_rss_gz_atom_gz());
765         $this-> template-> render();
766     }
767
768     public function robots_xhtml_rss_gz_atom_gz()
769     {
770         $data['title'] = 'Robots Xhtml Rss Gz Atom Gz Notificación';
771
772         $this-> load-> helper('form');
773
774         $data['filtros'] = $this-> input-> post();
775
776         $this-> template-> write("content" , $this-> Notificacion_model-> robots_xhtml_rss_gz_atom_gz());
777         $this-> template-> render();
778     }
779
780     public function sitemap_xhtml_atom_gz_atom_gz_atom()
781     {
782         $data['title'] = 'Sitemap Xhtml Atom Gz Atom Gz Atom Notificación';
783
784         $this-> load-> helper('form');
785
786         $data['filtros'] = $this-> input-> post();
787
788         $this-> template-> write("content" , $this-> Notificacion_model-> sitemap_xhtml_atom_gz_atom_gz_atom());
789         $this-> template-> render();
790     }
791
792     public function robots_xhtml_atom_gz_atom_gz_atom()
793     {
794         $data['title'] = 'Robots Xhtml Atom Gz Atom Gz Atom Notificación';
795
796         $this-> load-> helper('form');
797
798         $data['filtros'] = $this-> input-> post();
799
800         $this-> template-> write("content" , $this-> Notificacion_model-> robots_xhtml_atom_gz_atom_gz_atom());
801         $this-> template-> render();
802     }
803
804     public function sitemap_xhtml_rss_gz_atom_gz_atom()
805     {
806         $data['title'] = 'Sitemap Xhtml Rss Gz Atom Gz Atom Notificación';
807
808         $this-> load-> helper('form');
809
810         $data['filtros'] = $this-> input-> post();
811
812         $this-> template-> write("content" , $this-> Notificacion_model-> sitemap_xhtml_rss_gz_atom_gz_atom());
813         $this-> template-> render();
814     }
815
816     public function robots_xhtml_rss_gz_atom_gz_atom()
817     {
818         $data['title'] = 'Robots Xhtml Rss Gz Atom Gz Atom Notificación';
819
820         $this-> load-> helper('form');
821
822         $data['filtros'] = $this-> input-> post();
823
824         $this-> template-> write("content" , $this-> Notificacion_model-> robots_xhtml_rss_gz_atom_gz_atom());
825         $this-> template-> render();
826     }
827
828     public function sitemap_xhtml_atom_gz_atom_gz_atom_atom()
829     {
830         $data['title'] = 'Sitemap Xhtml Atom Gz Atom Gz Atom Atom Notificación';
831
832         $this-> load-> helper('form');
833
834         $data['filtros'] = $this-> input-> post();
835
836         $this-> template-> write("content" , $this-> Notificacion_model-> sitemap_xhtml_atom_gz_atom_gz_atom_atom());
837         $this-> template-> render();
838     }
839
840     public function robots_xhtml_atom_gz_atom_gz_atom_atom()
841     {
842         $data['title'] = 'Robots Xhtml Atom Gz Atom Gz Atom Atom Notificación';
843
844         $this-> load-> helper('form');
845
846         $data['filtros'] = $this-> input-> post();
847
848         $this-> template-> write("content" , $this-> Notificacion_model-> robots_xhtml_atom_gz_atom_gz_atom_atom());
849         $this-> template-> render();
850     }
851
852     public function sitemap_xhtml_rss_gz_atom_gz_atom_atom()
853     {
854         $data['title'] = 'Sitemap Xhtml Rss Gz Atom Gz Atom Atom Notificación';
855
856         $this-> load-> helper('form');
857
858         $data['filtros'] = $this-> input-> post();
859
860         $this-> template-> write("content" , $this-> Notificacion_model-> sitemap_xhtml_rss_gz_atom_gz_atom_atom());
861         $this-> template-> render();
862     }
863
864     public function robots_xhtml_rss_gz_atom_gz_atom_atom()
865     {
866         $data['title'] = 'Robots Xhtml Rss Gz Atom Gz Atom Atom Notificación';
867
868         $this-> load-> helper('form');
869
870         $data['filtros'] = $this-> input-> post();
871
872         $this-> template-> write("content" , $this-> Notificacion_model-> robots_xhtml_rss_gz_atom_gz_atom_atom());
873         $this-> template-> render();
874     }
875
876     public function sitemap_xhtml_atom_gz_atom_gz_atom_atom_atom()
877     {
878         $data['title'] = 'Sitemap Xhtml Atom Gz Atom Gz Atom Atom Atom Notificación';
879
880         $this-> load-> helper('form');
881
882         $data['filtros'] = $this-> input-> post();
883
884         $this-> template-> write("content" , $this-> Notificacion_model-> sitemap_xhtml_atom_gz_atom_gz_atom_atom_atom());
885         $this-> template-> render();
886     }
887
888     public function robots_xhtml_atom_gz_atom_gz_atom_atom_atom()
889     {
890         $data['title'] = 'Robots Xhtml Atom Gz Atom Gz Atom Atom Atom Notificación';
891
892         $this-> load-> helper('form');
893
894         $data['filtros'] = $this-> input-> post();
895
896         $this-> template-> write("content" , $this-> Notificacion_model-> robots_xhtml_atom_gz_atom_gz_atom_atom_atom());
897         $this-> template-> render();
898     }
899
900     public function sitemap_xhtml_rss_gz_atom_gz_atom_atom_atom()
901     {
902         $data['title'] = 'Sitemap Xhtml Rss Gz Atom Gz Atom Atom Atom Notificación';
903
904         $this-> load-> helper('form');
905
906         $data['filtros'] = $this-> input-> post();
907
908         $this-> template-> write("content" , $this-> Notificacion_model-> sitemap_xhtml_rss_gz_atom_gz_atom_atom_atom());
909         $this-> template-> render();
910     }
911
912     public function robots_xhtml_rss_gz_atom_gz_atom_atom_atom()
913     {
914         $data['title'] = 'Robots Xhtml Rss Gz Atom Gz Atom Atom Atom Notificación';
915
916         $this-> load-> helper('form');
917
918         $data['filtros'] = $this-> input-> post();
919
920         $this-> template-> write("content" , $this-> Notificacion_model-> robots_xhtml_rss_gz_atom_gz_atom_atom_atom());
921         $this-> template-> render();
922     }
923
924     public function sitemap_xhtml_atom_gz_atom_gz_atom_atom_atom_atom()
92
```

```

65
66
67     $data = array_merge($data, $filtros);
68 }
69
70     // Configuración para el Paginador
71     $configPag['base_url'] = '/' . DIR_TES . '/notificacion/index/';
72     $configPag['first_link'] = 'Primero';
73     $configPag['last_link'] = '&Uacute;ltimo' ;
74     $configPag['uri_segment'] = '4';
75     $configPag['total_rows'] = $this-> Notificacion_model-> getNumRows($this-
> input-> post('busqueda'));
76     $configPag['per_page'] = 20;
77     $this-> pagination-> initialize($configPag);
78     $this-> load-> model(DIR_SIIGS . '/ArbolSegmentacion_model');
79     $notifications = $this-> Notificacion_model-> getAll($this-> input-
> post('busqueda'), $configPag['per_page'], $pag);
80     foreach($notifications as $notif){
81         $descripciones = $this-> ArbolSegmentacion_model-
> getDescripcionById(explode(',', $notif-> id_arr_asu), 0);
82         for($i = 0; $i < count($descripciones); $i++){
83             $notif-> tabletas[$i] = $descripciones[$i]-> descripcion;
84         }
85         $data['notifications'] = $notifications;
86     }
87     catch(Exception $e){
88         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
89         $data['clsResult'] = 'error';
90     }
91     $this-> template-> write_view('content', DIR_TES . '/notificacion/index', $data);
92     $this-> template-> render();
93 }
94
95 /**
96 * Visualiza los datos de la notificación recibida
97 *
98 * @access public
99 * @param int $id id de notificación a visualizar
100 * @return void
101 */
102 public function view($id)
103 {
104     try {
105         if (empty($this-> Notificacion_model))
106             return false;
107         if (!Usuario_model::checkCredentials(DIR_TES .::__METHOD__, current_url()))
108             show_error('', 403, 'Acceso denegado');
109         $data['title'] = 'Ver detalles de la notificación';
110         $notification = $this-> Notificacion_model-> getById($id, true);
111         if (count($notification) > 0)
112         {
113             $notification = $notification[0];
114             $this-> load-> model(DIR_SIIGS . '/ArbolSegmentacion_model');
115             $descripciones = $this-> ArbolSegmentacion_model-
> getDescripcionById(explode(',', $notification-> id_arr_asu), 0);
116             for($i = 0; $i < count($descripciones); $i++)
117                 $notification-> tabletas[$i] = $descripciones[$i]-> descripcion;
118             }
119             $data['notificacion_item'] = $notification;
120         }
121         catch(Exception $e){
122             $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
123             $data['clsResult'] = 'error';
124         }
125         $this-> template-> write_view('content', DIR_TES . '/notificacion/view', $data);
126         $this-> template-> write('menu', '', true);
127         $this-> template-> write('sala_prensa', '', true);
128         $this-> template-> render();
129     }
130 }
131
132 /**
133 * 1) Prepara el formulario para la inserción de una notificación nueva
134 * 2) Realiza las validaciones necesarias sobre cada campo del registro
135 *
136 * @access public
137 * @return void
138 */
139 public function insert()
140 {

```

```

141     if (empty($this-> Notificacion_model))
142         return false;
143     if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url()))
144         show_error('', 403, 'Acceso denegado');
145     $data['title'] = 'Crear una nueva notificación';
146     $this-> load-> model(DIR_TES.'/notificacion_model');
147     $this-> load-> helper('form');
148     $this-> load-> library('form_validation');
149     $this-> form_validation-> set_rules('titulo', 'Titulo',
150                                         'trim|xss_clean|required|max_length[60]');
151     $this-> form_validation-> set_rules('contenido', 'Contenido',
152                                         'trim|xss_clean|required|max_length[300]');
153     $this-> form_validation-> set_rules('fecha_inicio', 'Fecha Inicio',
154                                         'trim|required');
155     $this-> form_validation-> set_rules('fecha_fin', 'Fecha Fin', 'trim|required');
156     $this-> form_validation-> set_rules('id_arr_asu', 'Reportar a tabletas',
157                                         'required');
158     // $arrItems = $this->grupo_model->getAll();
159     // $data['items'][0] = '-- Seleccione una opción --';
160     //
161     foreach ($arrGrupos as $grupo)
162     {
163         $data['grupos'][$grupo->id] = $grupo->nombree;
164     }
165
166     if ($this-> form_validation-> run() === FALSE)
167     {
168         $this-> template-> write_view('content',DIR_TES.'/notificacion/insert', $data);
169     }
170     else
171     {
172         try {
173             $this-> Notificacion_model-> setTitulo(strtoupper($this-> input-
174 > post('titulo')));
175             $this-> Notificacion_model-> setContenido($this-> input-
176 > post('contenido'));
177             $this-> Notificacion_model-> setFechaInicio(date('Y-m-d', strtotime($this-
178 > input-> post('fecha_inicio'))));
179             $this-> Notificacion_model-> setFechaFin(date('Y-m-d', strtotime($this-
180 > input-> post('fecha_fin'))));
181             $this-> Notificacion_model-> setIdsTabletas($this-> input-
182 > post('id_arr_asu'));
183             $this-> Notificacion_model-> insert();
184             $this-> session-> set_flashdata('msgResult', 'Registro agregado
exitosamente');
185             $this-> session-> set_flashdata('clsResult', 'success');
186             Bitacora_model::insert(DIR_TES.'::'.__METHOD__, 'Notificación agregada:
'.strtoupper($this->
187 .stroupper($this-> input-> post('titulo'))));
188             redirect(DIR_TES.'/notificacion','refresh');
189         }
190         catch (Exception $e){
191             $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
192             $data['clsResult'] = 'error';
193             $this-> template-> write_view('content',DIR_TES.'/notificacion/insert',
194 $data);
195         }
196     }
197     $this-> template-> render();
198 }
199 /**
200 * 1) Prepara el formulario para la modificación de una notificación existente
201 * 2) Realiza las validaciones necesarias sobre cada campo del registro
202 *
203 * @access public
204 * @param int $id id de la notificación a modificar
205 * @return void
206 */
207 public function update($id)
208 {
209     if (empty($this-> Notificacion_model))
210         return false;
211     if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url()))
212         show_error('', 403, 'Acceso denegado');
213     $data['title'] = 'Modificar notificación';
214     $this-> load-> helper('form');
215     $this-> load-> library('form_validation');
216     $this-> form_validation-> set_rules('titulo', 'Titulo',
217                                         'trim|xss_clean|required|max_length[60]');
218     $this-> form_validation-> set_rules('contenido', 'Contenido',

```

```

'trim|xss_clean|required|max_length[300]');
208     $this-> form_validation-> set_rules('fecha_inicio', 'Fecha Inicio',
'trim|required');
209     $this-> form_validation-> set_rules('fecha_fin', 'Fecha Fin', 'trim|required');
210     $this-> form_validation-> set_rules('id_arr_asu', 'Reportar a tabletas',
'required');
211
212     $notification = $this-> Notificacion_model-> getById($id, true)[0];
213     $this-> load-> model(DIR_SIIGS.'/ArbolSegmentacion_model');
214     $descripciones = $this-> ArbolSegmentacion_model-
> getDescripcionById(explode(',', $notification-> id_arr_asu), 0);
215     for($i = 0; $i < count($descripciones); $i++)
216         $notification-> tabletas[$i] = $descripciones[$i]-> descripcion;
217     $data['notificacion_item'] = $notification;
218
219     if ($this-> form_validation-> run() === FALSE)
220     {
221         $this-> template-> write_view('content',DIR_TES.'/notificacion/update', $data);
222         $this-> template-> render();
223     }
224     else
225     {
226         try {
227             $this-> Notificacion_model-> setId($id);
228             $this-> Notificacion_model-> setTitulo(strtoupper($this-> input-
> post('titulo')));
229             $this-> Notificacion_model-> setContenido($this-> input-
> post('contenido'));
230             $this-> Notificacion_model-> setFechaInicio(date('Y-m-d', strtotime($this-
> input-> post('fecha_inicio'))));
231             $this-> Notificacion_model-> setFechaFin(date('Y-m-d', strtotime($this-
> input-> post('fecha_fin'))));
232             $this-> Notificacion_model-> setIdsTabletas($this-> input-
> post('id_arr_asu'));
233             $this-> Notificacion_model-> update();
234             $this-> session-> set_flashdata('msgResult', 'Registro actualizado
exitosamente');
235             $this-> session-> set_flashdata('clsResult', 'success');
236             Bitacora_model::insert(DIR_TES.'::'.__METHOD__, 'Notificación actualizada:
'. $id);
237             redirect(DIR_TES.'/notificacion','refresh');
238         }
239         catch (Exception $e){
240             $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
241             $data['clsResult'] = 'error';
242             $this-> template-> write_view('content',DIR_TES.'/notificacion/update',
$data);
243             $this-> template-> render();
244         }
245     }
246 }
247
248 /**
249 * Solicita la eliminación de la notificación recibida
250 *
251 * @access public
252 * @param int $id id de notificación a eliminar
253 * @return void
254 */
255 public function delete($id)
256 {
257     try {
258         if (empty($this-> Notificacion_model))
259             return false;
260         if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url()))
261             show_error('', 403, 'Acceso denegado');
262         $this-> Notificacion_model-> setId($id);
263         $this-> Notificacion_model-> delete();
264         $this-> session-> set_flashdata('msgResult', 'Registro eliminado exitosamente');
265         $this-> session-> set_flashdata('clsResult', 'success');
266         Bitacora_model::insert(DIR_TES.'::'.__METHOD__, 'Notificación eliminada: !$id');
267     }
268     catch (Exception $e){
269         $this-> session-> set_flashdata('msgResult', Errorlog_model::save($e-
> getMessage(), __METHOD__));
270         $this-> session-> set_flashdata('clsResult', 'error');
271     }
272     redirect(DIR_TES.'/notificacion','refresh');
273 }
274

```

275 }

# Archivo fuente para reporteador.php

La documentación para este archivo está disponible en [reporteador.php](#)

```
1  <?php
2  /**
3   * Controlador Reporteador
4   *
5   * @package      TES
6   * @subpackage   Controlador
7   * @author       Rogelio
8   * @created      2013-12-20
9   */
10  class Reporteador extends CI_Controller {
11
12      public function __construct()
13  {
14          parent::__construct();
15          try{
16              $this-> load-> helper('url');
17              $this-> load-> model(DIR_TES.'/Reporteador_model');
18          }
19          catch(Exception $e)
20          {
21              $this-> template-> write("content" , $e-> getMessage());
22              $this-> template-> render();
23          }
24      }
25
26      /**
27       * Visualiza los reportes existentes
28       *
29       * @access      public
30       * @return     void
31       */
32      public function index()
33  {
34          try{
35              if (empty($this-> Reporteador_model))
36                  return false;
37              if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url()))
38                  show_error('', 403, 'Acceso denegado');
39              $data['title'] = 'Reportes';
40              $this-> load-> helper('form');
41              $this-> load-> model(DIR_SIIGS.'/ArbolSegmentacion_model');
42
43              $data['msgResult'] = $this-> session-> flashdata('msgResult');
44              $data['clsResult'] = $this-> session-> flashdata('clsResult');
45              $data['estados'] = (array)$this-> ArbolSegmentacion_model-> getDataKeyValue(1,
1);
46
47              $array[0] = (array("atributo" => "Cobertura y Concentrado de
Actividades por Tipo de
Biológico" , "valor" => "" , "lista" => "0" ));
48              // $array[1] = (array("atributo"=>"Concentrado de
Actividades" , "valor"=>"", "lista"=>"1"));
49              // $array[2] = (array("atributo"=>"Seguimiento RV-1 y RV-5 a
menores de 1 año" , "valor"=>"", "lista"=>"2"));
50              $array[3] = (array("atributo" => "Censo
Nominal" , "valor" => "" , "lista" => "3" ));
51              $array[4] = (array("atributo" => "Esquemas
Incompletos" , "valor" => "" , "lista" => "4" ));
52              $data['datos']=$array;
53          }
54          catch(Exception $e){
55              $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
56              $data['clsResult'] = 'error';
57          }
58          $this-> template-> write_view('content',DIR_TES.'/reporteador/index' , $data);
59          $this-> template-> render();
60      }
}
```

```

61
62     /**
63      * Renderiza la vista del reporte
64      *
65      * @access public
66      * @param int $op
67      * @param string $title
68      * @param int $nivel
69      * @param int $id
70      * @param string $fecha
71      * @return void
72     */
73     public function view($op, $title, $nivel, $id, $fecha = '')
74     {
75         try{
76             if (empty($this-> Reporteador_model))
77                 return false;
78
79             if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url()))
80                 show_error('', 403, 'Acceso denegado');
81
82             $array=array();
83             $data['title'] = $title;
84             $data['datos'] = $array;
85             $fechaFin = null;
86
87             if($op==0) {
88                 $this-> load-> helper('formatFecha');
89
90                 // Si es entero, significa que selecciono una semana nacional de salud
91                 if(!isDate($fecha, 'd-m-Y')) {
92                     $this-> load-> model(DIR_TES.'/Semana_nacional_model');
93                     // Se debe obtener la semana nacional
94                     $semana_nacional = $this-> Semana_nacional_model-> getById($fecha);
95
96                     $fecha = formatFecha($semana_nacional-> fecha_fin, 'Y-m-d');
97                 }
98
99             $array = $this-> Reporteador_model-> getCoberturaBiologicoListado($nivel,
$Id, $fecha);
100            $grupo = $this-> Reporteador_model-> getGrupoVacunas();
101
102            $filaTitulosGrupos = '';
103            $filaVacunas = '';
104            $contGrupos = 0;
105            $contVacunas = 0;
106
107            // Hace un recorrido de todos los grupos
108            foreach ($grupo as $gru) {
109                $filaTitulosGrupos .= '<th colspan="'.($gru-> grupo).'>' ;
110                $vacunas = '';
111
112                // Obtiene las vacunas de cada grupo
113                $vacunasGrupo = $this-> Reporteador_model-> getVacunasByGrupo($gru-> grupo);
114
115                // concatena todas las vacunas de cada grupo,
116                // para mostrar en los titulos la descripción corta
117                foreach ($vacunasGrupo as $vac) {
118                    $vacunas .= '<th>'. $vac-> descripcion_corta.'</th>' ;
119                    $contVacunas++;
120                }
121
122                $filaVacunas .= $vacunas . '<th>%Cob.</th>' ;
123                $contGrupos++;
124            }
125
126            $data['headTable'] = '<thead>
127                <tr>
128                    <th rowspan="3">Grupo de edad</th>
129                    <th colspan="3">Población</th>
130                    <th colspan="'.($contGrupos+$contVacunas).'">Biológico</th>
131                    <th rowspan="2" colspan="3">Esquemas completos</th>
132                </tr>
133                <tr>
134                    <th rowspan="2" colspan="2">Oficial</th>
135                    <th rowspan="2" colspan="2">Nominal</th>

```

```

136             <th rowspan="2">% Conc.</th>
137             '$filaTitulosGrupos.'
138         </tr>
139         <tr>
140             '$filaVacunas.'
141             <th>Total</th>
142             <th>%Of.</th>
143             <th>%Nom.</th>
144         </tr></thead> ;
145     }
146     if($op==1)
147         $array=$this-> Reporteador_model-> getConcentradoActividades($nivel, $id,
148 $fecha);
149     /*if($op==2)
150         $array=$this->Reporteador_model->getSeguimientoRV1RV5($nivel, $id,
151 $fecha);*/
152     if($op==3){
153         $this-> load-> model(DIR_TES.'/Reporte_censo_nominal');
154         $array = $this-> Reporteador_model-> getCensoNominal($nivel, $id, $th);
155         $data['headTable'] = $th;
156     }
157     if($op==4){
158         $this-> load-> model(DIR_TES.'/Reporte_censo_nominal');
159         $array = $this-> Reporteador_model-> getEsquemasIncompletos($nivel, $id,
160 $th);
161         $data['headTable'] = $th;
162     }
163     $data['datos'] = $array;
164     catch(Exception $e)
165     {
166         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
167         $data['infoclass'] = 'error';
168     }
169     $this-> template-> write('header','','true');
170     $this-> template-> write('footer','','true');
171     $this-> template-> write('menu','','true');
172     $this-> template-> write('sala_prensa','','true');
173     $this-> template-> write('ajustaAncho',1,true);
174     $this-> template-> write_view('content',DIR_TES.'/reporteador/reporter_view', $data);
175     $this-> template-> render();
176 }

```

# Archivo fuente para reporte\_sincronizacion.php

La documentación para este archivo está disponible en [reporte\\_sincronizacion.php](#)

```
1  <?php    if ( ! defined('BASEPATH')) exit('No direct script access allowed');
2  /**
3   * Controller Usuario
4   *
5   * @package      TES
6   * @subpackage   Controlador
7   * @author       Eliecer
8   * @created     2013-12-17
9   */
10  class Reporte_sincronizacion extends CI_Controller
11  {
12
13      public function __construct()
14      {
15          parent::__construct();
16          try
17          {
18              $this->load->helper('url');
19              $this->load->helper('date');
20          }
21          catch(Exception $e)
22          {
23              $this->template->write("content" , $e->getMessage());
24              $this->template->render();
25          }
26      }
27
28      /**
29      *
30      * Este es el metodo por default, crea el listado del reporte de sincronizacion
31      *
32      * @return      void
33      */
34
35      public function index()
36      {
37          try{
38              $this->load->model(DIR_TES.'/Reporte_sincronizacion_model');
39              if (empty($this->Reporte_sincronizacion_model))
40                  return false;
41              if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url()))
42                  show_error('', 403, 'Acceso denegado');
43
44              $data['title'] = 'Reporte Sincronización';
45
46              $ttr=$this->Reporte_sincronizacion_model->getCount("tes_tableta");
47              $array[0] = (array("atributo" => "Total de tabletas registradas" ,
48                                "valor" => $ttr,"lista" => "0" )); ;
49
50              $ttsa=$this->Reporte_sincronizacion_model->getCount(""
51 "select * from tes_tableta where id_tes_estado_tableta in(1)" );
52              $array[1] = (array("atributo" => "Total de tabletas sin asignar" ,
53                                "valor" => $ttsa,"lista" => "1" )); ;
54
55              $td=$this->Reporte_sincronizacion_model->getCount(""
56 "* from tes_tableta where id_tes_estado_tableta in(4)" );
57              $array[2] = (array("atributo" => "Porcentaje de tabletas desactualizadas" ,
58                                "valor" => number_format(($td*100)/$ttr,"2" ),"lista&quot;" );
59
60              $tir=$this->Reporte_sincronizacion_model->getCount(""
61 "* from tes_tableta where id_tes_estado_tableta in(5,6)" );
62              $array[3] = (array("atributo" => "Porcentaje de tabletas inactivas o en"
```

```

reparación"      , "valor"          => number_format(( $tir * 100 ) / $ttr, "2" ) , "lista"      =&
gt;"3"           );;
57
58         $tut=$this-> Reporte_sincronizacion_model-> getCount("") , "select
59         $array[4] = (array("atributo"          => "Total de UM con
tabletas"      , "valor"          => $tut,"lista"          => "4"           ));;
60
61         $us=$this-> Reporte_sincronizacion_model-> getCount("") , "select *
from tes_tableta where id_tes_estado_tableta in(2,3)" );
62         $array[5] = (array("atributo"          => "Porcentaje de tabletas
sincronizadas" , "valor"          => number_format(( $us * 100 ) / $ttr, "2" ),"lista";
;=> "5"           ));;
63
64         $array[6] = (array("atributo"          => "Total de tabletas
sincronizadas" , "valor"          => $us,"lista"          => "6"           ));;
65
66         $array[7] = (array("atributo"          => "Porcentaje de tabletas
desincronizadas" , "valor"          => number_format(100-
(( $us * 100 ) / $ttr), "2" ),"lista"          => "7"           ));;
67
68         $ttt=$this-> Reporte_sincronizacion_model-> getCount("") , "select
* from tes_tableta where id_tes_estado_tableta not in(2,3)" );
69         $array[8] = (array("atributo"          => "Total de tabletas
desincronizadas" , "valor"          => $ttt,"lista"          => "8"           ));;
70
71         $tpn=$this-> Reporte_sincronizacion_model-> getCount("") , "select
distinct(id_persona) from tes_pendientes_tarjeta" );
72         $array[9] = (array("atributo"          => "Total de pacientes que no llevan
su tes sincronizada con la
plataforma"      , "valor"          => $tpn,"lista"          => "9"           ));;
73
74         $tpt=$this-> Reporte_sincronizacion_model-
> getCount("tes_pendientes_tarjeta" );
75         $array[10] = (array("atributo"          => "Total de controles no
registrados en la tes" , "valor"          => $tpt,"lista"          => "10"           ));;
76
77         $version=$this-> Reporte_sincronizacion_model-> get_version();
78         $array[11] = (array("atributo"          => "Ultima version de la
app"      , "valor"          => $version[0]-> version,"lista"          => "11"           ));;
79         $array[12] = (array("atributo"          => $version[0]-
> fecha_liberacion,"lista"          => "12"           ));;
80
81         $data['datos']=$array;
82     }
83     catch(Exception $e)
84     {
85         $data['msgResult']= Errorlog_model::save($e-> getMessage(), __METHOD__);
86         $data['infoclass']="warning";
87     }
88     // $this->load->view('usuario/index', $data);
89     $this-> template-> write_view('content',DIR_TES.'/reporteador/sincronizacion',
90 $data);
91     $this-> template-> render();
92
93 /**
94 *
95 * Muestra detallada por cada list del reporte de sincronizacion
96 *
97 * @param string $op      Especifica el reporte a mostrar
98 * @param string $title   Especifica el titulo a mostrar en el reporte
99 *
100 * @return void
101 *
102 */
103 public function view($op,$title)
104 {
105     try{
106         $this-> load-> model(DIR_TES.'/Reporte_sincronizacion_model');
107         if (empty($this-> Reporte_sincronizacion_model))
108             return false;
109         if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url()))
110             show_error('', 403, 'Acceso denegado');
111
112         $data['title']= $title;
113         $array=array();
114         $campos="t.id as No,mac as Mac, tv.version+ -> '+tv.descripcion as
Version, et.descripcion as Estado, tc.descripcion as 'Tipo Censo', asu.descripcion as 'Unidad

```

```

Medica"      ;
115          $join = "left join tes_version tv on tv.id= t.id_version
116          left join sis_estado_tableta et on et.id=t.id_tes_estado_tableta
117          left join tes_tipo_censo tc on tc.id=t.id_tipo_censo
118          left join asu_arbol_segmentacion asu on asu.id=t.id_asu_um" ;
119          if($op==0)
120          $array=$this->  Reporte_sincronizacion_model->  getListado("    SELECT $campos
FROM tes_tableta t $join" );
121
122          if($op==1)
123          $array=$this->  Reporte_sincronizacion_model->  getListado("    SELECT $campos
FROM tes_tableta t $join WHERE id_tes_estado_tableta IN ('1')") );
124
125          if($op==2)
126          $array=$this->  Reporte_sincronizacion_model->  getListado("    SELECT $campos
FROM tes_tableta t $join WHERE id_tes_estado_tableta IN ('1')") );
127
128          if($op==3)
129          $array=$this->  Reporte_sincronizacion_model->  getListado("    SELECT $campos
FROM tes_tableta t $join WHERE id_tes_estado_tableta IN (5,6)") );
130
131          if($op==4)
132          $array=$this->  Reporte_sincronizacion_model->  getListado("    SELECT $campos
FROM tes_tableta t $join WHERE t.id_asu_um!=''" );
133
134          if($op==5 || $op==6)
135          $array=$this->  Reporte_sincronizacion_model->  getListado("    SELECT $campos
FROM tes_tableta t $join WHERE id_tes_estado_tableta IN (3,2)") );
136
137          if($op==7 || $op==8)
138          $array=$this->  Reporte_sincronizacion_model->  getListado("    SELECT $campos
FROM tes_tableta t $join WHERE id_tes_estado_tableta NOT IN (3,2)") );
139
140          if($op==9)
141          $array=$this->  Reporte_sincronizacion_model->  getListado("SELECT distinct
id_persona,fecha,tabla,resuelto,mac FROM tes_pendientes_tarjeta" );
142
143          if($op==10)
144          $array=$this->  Reporte_sincronizacion_model->  getListado("SELECT
id_persona,fecha,tabla,resuelto,mac FROM tes_pendientes_tarjeta" );
145
146          if($op==11)
147          $array=$this->  Reporte_sincronizacion_model->  getListado("SELECT * FROM
tes_version" );
148
149          if($op==12)
150          $array=$this->  Reporte_sincronizacion_model->  getListado("SELECT * FROM
tes_version order by fecha_liberacion DESC" );
151
152          $data['datos']=$array;
153      }
154      catch(Exception $e)
155      {
156          $data['msgResult']= Errorlog_model::save($e->  getMessage(), __METHOD__);
157          $data['infoclass']="warning" ;
158      }
159      // $this->load->view('usuario/index', $data);
160      $this->  template->  write('header','','true');
161      $this->  template->  write('footer','','true');
162      $this->  template->  write('menu','','true');
163      $this->  template->  write('sala_prensa','','true');
164      $this->  template->  write_view('content',DIR_TES.'/reporteador/reporte_view', $data);
165      $this->  template->  render();
166  }
167
168  /**
169  *
170  * Genera el item del reporte de lote de vacunacion
171  *
172  * @return      void
173  *
174  */
175  public function lote()
176  {
177      try{
178          $this->  load->  model(DIR_TES.'/Reporte_sincronizacion_model');
179          $this->  load->  model(DIR_SIIGS.'/ArbolSegmentacion_model');
180          if (empty($this->  Reporte_sincronizacion_model))
181              return false;
182          if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url()))

```

```

183     show_error('', 403, 'Acceso denegado');
184     $data['title'] = 'Seguimiento de lotes de vacunación';
185     $this-> load-> helper('form');
186     $unidad="";
187     $desde=date('Y-m-d H:i:s', strtotime($this-> input-> post('desde')));
188     if($this-> input-> post('hasta')=="")
189         $hasta=date("Y-m-d H:i:s");
190     else
191         $hasta=date('Y-m-d H:i:s', strtotime($this-> input-> post('hasta')));
192     $lotes=$this-> input-> post('lote');
193
194     $jurid=$this-> input-> post('juris');
195     $munic=$this-> input-> post('municipios');
196     $local=$this-> input-> post('localidades');
197     $ums = $this-> input-> post('ums');
198
199     if($ums!=""
200         &&      $ums!="0")
201         $unidad = " AND id_asu_um='$ums' ";
202
203     else if($local!=""
204         &&      $ums!="0")
205         $unidad = "AND id_asu_um IN (
206                         SELECT id FROMasu_arbol_segmentacion WHERE
207                         .$local.)"
208         ; // ums por loc
209     else if($munic!=""
209         &&      $ums!="0")
210         $unidad = "AND id_asu_um IN (
211                         SELECT id FROMasu_arbol_segmentacion WHERE id_padre IN (
212                             SELECT id FROMasu_arbol_segmentacion WHERE
213                             .$munic.) )"
214         ; // locs por mpio
215     else if($jurid!=""
215         &&      $ums!="0")
216         $unidad = "AND id_asu_um IN (
217                         SELECT id FROMasu_arbol_segmentacion WHERE id_padre IN (
218                             SELECT id FROMasu_arbol_segmentacion WHERE id_padre IN (
219                             SELECT id FROMasu_arbol_segmentacion WHERE
220                             .$jurid.) )"
221         ; // mpios por juris
222     if($lotes==""
222         )$mas="OR codigo_barras IS NULL" ;else
223     ;
224     $consulta=" select distinct(codigo_barras) from cns_control_vacuna where
225 (codigo_barras like '%$lotes%' $mas) $unidad and (fecha between '$desde' and '$hasta') ";
226
227     $count=$this-> Reporte_sincronizacion_model-> getCount("", $consulta);
228     $array=$this-> Reporte_sincronizacion_model-> getListado($consulta);
229
230     $i=0;$midato=array();
231     foreach($array as $x)
232     {
233         $consulta="select * from cns_control_vacuna where codigo_barras "
234         $consultb="select distinct(id_asu_um) from cns_control_vacuna where
235         codigo_barras " ;
236         $consultc="select distinct(cv.id_vacuna),v.descripcion from
237         cns_control_vacuna cv left join cns_vacuna v on v.id=cv.id_vacuna where cv.codigo_barras" ;
238         $consultd="select distinct(p.id_persona) from cns_control_vacuna cv left
239         join tes_pendientes_tarjeta p on p.id_persona=cv.id_persona where p.id_persona!="" and
240         cv.codigo_barras " ;
241         $in="";
242         if($x-> codigo_barras=="")
243         {
244             $tipoa="";
245             $midato[$i]["lote"] ="Sin lote" ;
246             $cantidad=$this-> Reporte_sincronizacion_model-
247             ,$consulta." IS NULL" );
248             $ums=$this-> Reporte_sincronizacion_model-
249             ,$consultb." IS NULL" );
250             $personas=$this-> Reporte_sincronizacion_model-
251             ,$consultd." IS NULL" );
252
253             $tipol=$this-> Reporte_sincronizacion_model-
254             ." IS NULL" );
255             foreach($tipol as $y)
256             {
257                 $tipoa.= $y-> descripcion." - ";
258             }
259             $sumsx=$this-> Reporte_sincronizacion_model-
260             ." IS NULL" );
261             foreach($sumsx as $u)
262             {
263                 $in.= $u-> id_asu_um.", ";
264             }
265             $localidades=$this-> Reporte_sincronizacion_model-
266             ." select distinct(id_padre) fromasu_arbol_segmentacion where id
267             in(" .substr($in,0,strlen($in)-1)." ) and id_padre!=0" );

```

```

247
248     $midato[$i][ "tipo"           ]=substr($tipoa,0,strlen($tipoa)-3);
249     $midato[$i][ "cantidad"       ]=$cantidad;
250     $midato[$i][ "ums"           ]=$ums;
251     $midato[$i][ "localidades"   ]=$localidades;
252     $midato[$i][ "personas"      ]=$personas;
253     $vacunas=$this-> Reporte_sincronizacion_model-> getListado("select
254 distinct(id_asu_um) from cns_control_vacuna where codigo_barras IS NULL");
255     $dom=$this-> ArbolSegmentacion_model-
256 > getDescripcionById(array($vacunas[0]-> id_asu_um),5);
257     $dom=(explode(",",$dom[0]-> descripcion));
258     $midato[$i][ "lugar"          ]=$dom[count($dom)-1];
259 }
260 else
261 {
262     $tipob="";
263     $midato[$i][ "lote"           ]=$x-> codigo_barras;
264     $cantidad=$this-> Reporte_sincronizacion_model-
265     ,$consulta.= " . $x-> codigo_barras." ;
266     $sums=$this-> Reporte_sincronizacion_model-
267     ,$consultb.= " . $x-> codigo_barras." ;
268     $personas=$this-> Reporte_sincronizacion_model-
269     ,$consultd.= " . $x-> codigo_barras." ;
270
271     $stipo2=$this-> Reporte_sincronizacion_model-
272 > getListado($consultc.= " . $x-> codigo_barras." );
273     foreach($stipo2 as $y)
274     {
275         $tipob.=$y-> descripcion." - ";
276     }
277
278     $sumsx=$this-> Reporte_sincronizacion_model-
279 > getListado($consultb.= " . $x-> codigo_barras." );
280
281     foreach($sumsx as $u)
282     {
283         $in.= $u-> id_asu_um." , ";
284     }
285     $localidades=$this-> Reporte_sincronizacion_model-
286     "select distinct(id_padre) from asu_arbol_segmentacion where id
287     in(" . substr($in,0,strlen($in)-1).") and id_padre!=0" );
288
289     $midato[$i][ "tipo"           ]=substr($tipob,0,strlen($tipob)-3);
290     $midato[$i][ "cantidad"       ]=$cantidad;
291     $midato[$i][ "ums"           ]=$ums;
292     $midato[$i][ "localidades"   ]=$localidades;
293     $midato[$i][ "personas"      ]=$personas;
294     $vacunas=$this-> Reporte_sincronizacion_model-> getListado("select
295 distinct(id_asu_um) from cns_control_vacuna where codigo_barras " . "=" .
296     $x-> codigo_barras." );
297     $dom=$this-> ArbolSegmentacion_model-
298 > getDescripcionById(array($vacunas[0]-> id_asu_um),5);
299     $dom=(explode(",",$dom[0]-> descripcion));
300     $midato[$i][ "lugar"          ]=$dom[count($dom)-1];
301 }
302 $i++;
303 }
304 $data[ "count"           ]=$count;
305 $data[ "datos"           ]=$midato;
306 $data[ 'msgResult' ] = $this-> session-> flashdata('msgResult');
307 $rsJuris = $this-> ArbolSegmentacion_model-> getDataKeyValue(1, 2);
308 $data[ 'juris'][0] = 'Seleccione una opción';
309 foreach ($rsJuris as $rsJ) {
310     $data[ 'juris'][$rsJ-> id] = $rsJ-> descripcion;
311
312     $data[ 'municipios'][0] = 'Seleccione una opción';
313     $data[ 'localidades'][0] = 'Seleccione una opción';
314     $data[ 'ums'][0] = 'Seleccione una opción';
315
316     if($this-> input-> post('juris')) {
317         $municipios = $this-> ArbolSegmentacion_model-
318 > getDataKeyValue(1, 3, $this-> input-> post('juris'));
319         foreach ($municipios as $mpio) {
320             $data[ 'municipios'][$mpio-> id] = $mpio-> descripcion;
321         }
322     }
323     if($this-> input-> post('municipios')) {
324         $localidades = $this-> ArbolSegmentacion_model-
325 > getDataKeyValue(1, 4, $this-> input-> post('municipios'));
326         foreach ($localidades as $loc) {
327             $data[ 'localidades'][$loc-> id] = $loc-> descripcion;
328         }
329     }
330 }
```

```

313                     $data['localidades'][$loc-> id] = $loc-> descripcion;
314                 }
315             }
316             if($this-> input-> post('localidades')) {
317                 $sums = $this-> ArbolSegmentacion_model-> getDataKeyValue(1, 5,
318             $this-> input-> post('localidades'));
319                 foreach ($sums as $sum) {
320                     $data['ums'][$sum-> id] = $sum-> descripcion;
321                 }
322             }
323         }
324     catch(Exception $e){
325         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
326         $data['infoclass']="warning";
327     }
328     $this-> template-> write_view('content',DIR_TES.'/reporteador/lote', $data);
329     $this-> template-> render();
330 }
331 /**
332 *
333 * Muestra detallada por cada list del reporte de lote de vacunacion
334 *
335 * @param string $lote Valor del lote del que se esta generando el
336 * @param string $title Especifica el titulo a mostrar en el reporte
337 * @param string $op Especifica el reporte a mostrar
338 * @param string $lugar Especifica el lugar donde se centrara el mapa
339 *
340 *
341 * @return void
342 *
343 */
344 public function lote_view($lote,$title,$op,$lugar="Chiapas")
345 {
346     try{
347         $this-> load-> model(DIR_TES.'/Reporte_sincronizacion_model');
348         $this-> load-> model(DIR_SIIGS.'/ArbolSegmentacion_model');
349         if (empty($this-> Reporte_sincronizacion_model))
350             return false;
351         if (!Usuario_model::checkCredentials(DIR_TES::__METHOD__, current_url()))
352             show_error('', 403, 'Acceso denegado');
353         $pagina="reporte_view";
354         $lote=urldecode($lote);
355         $data['title'] = $title." - ".$lote;
356         $array=array();
357         $consulta="select distinct(id_persona) from cns_control_vacuna where
358         codigo_barras ";
359         $consultb="select distinct(id_asu_um) from cns_control_vacuna where
360         codigo_barras ";
361         $consultc="select distinct(cv.id_vacuna),v.descripcion from cns_control_vacuna
362         cv left join cns_vacuna v on v.id=cv.id_vacuna where cv.codigo_barras ";
363         $consultd="select distinct(p.id_persona) from cns_control_vacuna cv left join
364         tes_pendientes_tarjeta p on p.id_persona=cv.id_persona where p.id_persona!='' and cv.codigo_barras
365         ";
366         if($lote=="Sin lote")
367         {
368             $consulta.=" IS NULL";
369             $consultb.=" IS NULL";
370             $consultc.=" IS NULL";
371             $consultd.=" IS NULL";
372         }
373         else
374         {
375             $consulta.= " = '$lote.'";
376             $consultb.= " = '$lote.'";
377             $consultc.= " = '$lote.'";
378             $consultd.= " = '$lote.'";
379         }
380         if($op==1)
381         {
382             $vacunas=$this-> Reporte_sincronizacion_model-> getListado($consulta);
383             foreach($vacunas as $x)
384             {
385                 $result=$this-> Reporte_sincronizacion_model-> getListado("SELECT
386                 distinct curp,nombre, apellido_paterno as paterno, apellido_materno as materno, sexo FROM cns_persona
387                 WHERE id=' '$x-> id_persona.''");
388                 foreach($result as $y)
389             }

```

```

384                     {
385                         $array[] = array( "Curnp"      => $y-> curp, "Ap."      => $y-> materno, "Sexo"    => $y-
386                         paterno, "Ap. Materno" > sexo);
387                     }
388                 }
389             }
390             if($op==2)
391             {
392                 $vacunas=$this-> Reporte_sincronizacion_model-> getListado($consultb);
393                 foreach($vacunas as $x)
394                 {
395                     $dom=$this-> ArbolSegmentacion_model-> getDescripcionById(array($x-
396 > id_asu_um),5);
397                     $dom[0]-> descripcion;
398                 }
399             }
400             if($op==3 || $op==4)
401             {
402                 $in="" ;$pagina="reporte_map";
403                 $umsx=$this-> Reporte_sincronizacion_model-> getListado($consultb);
404                 if($op==4)
405                 {
406                     $id_p="";
407                     $cns_p=(array)$this-> Reporte_sincronizacion_model-
408 > getListado($consultd);
409                     foreach($cns_p as $person)
410                         $id_p.= substr($id_p,0,strlen($id_p)-1);
411                     $umsx=$this-> Reporte_sincronizacion_model-> getListado(" SELECT
412 id_asu_um_tratante AS id_asu_um FROM cns_persona WHERE id IN($id_p) ");
413                     foreach($umsx as $u)
414                     {
415                         $in.=$u-> id_asu_um." ";
416                     }
417                     $tipol=$this-> Reporte_sincronizacion_model-> getListado($consultc);
418                     $tipoa="";
419                     foreach($tipol as $y)
420                     {
421                         $tipoa.=$y-> descripcion." - ";
422                     }
423
424                     $localidades=$this-> Reporte_sincronizacion_model-> getListado("select
425 distinct(id_padre) from asu_arbol_segmentacion where id in(" .substr($in,0,strlen($in)-1).")
426 and id_padre!=0");
427                     $arbol=array();
428                     $mapas=array();
429                     foreach($localidades as $x)
430                     {
431                         $arbol[]=$x-> id_padre;
432                     }
433                     $dom=$this-> ArbolSegmentacion_model-> getDescripcionById($arbol,1);
434                     $datos=array();
435                     if($dom)
436                     foreach($dom as $y)
437                     {
438                         $m=explode(",",$y-> descripcion);
439                         $datos[] = array("Id" => $y-> id,
440 "Localidad" => $m[0], "Municipio" => $m[1]);
441                         $latlon=$this-> Reporte_sincronizacion_model-> getListado("SELECT
442 * FROM asu_georeferencia WHERE id_asu='".$y-> id."' ");
443                         if($latlon)
444                         {
445                             $descripcion="";
446                             $consultx=$this-> Reporte_sincronizacion_model-
447 > getListado("SELECT DISTINCT(id_asu_um) FROM cns_control_vacuna WHERE id_asu_um IN(SELECT id
448 FROM asu_arbol_segmentacion WHERE id_padre='".$y-> id."' ");
449                             $consulty=array();$i=0;
450                             foreach($consultx as $cx)
451                             {
452                                 $consulty[$i]=$cx-> id_asu_um;
453                                 $i++;
454                             }
455                             $consultx=implode(",",$consulty);
456
457                         $mres=$this-> Reporte_sincronizacion_model-

```

```

>     getListado("      select id,descripcion fromasu_arbol_segmentacion where id in ($consultx)      ");
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
    } }

        select id_asu_um from cns_control_vacuna where id_asu_um= '$xy-' . $xy-
    {
        $ccc=$this-> Reporte_sincronizacion_model-
        , "select id_asu_um from cns_control_vacuna where id_asu_um= '$xy-' . $xy-
    );
        $descripcion.= "<tr><td>" . $xy-
        $descripcion.= "<td></td></tr>" . $xy-
    }
        $table= "<strong>Tipo:</strong>
</strong> $tipoa<br><table width='300'><tr><th align='left'>No</th><th align='left'>UM.</th><th align='left'>Cant.</th></tr>
$descripcion</table>" . $xy-
        $mapas[] =array(
            "localidad" => $m[0],
            "lat" => $latlon[0]-> lat_dec,
            "lon" => $latlon[0]-> lon_dec,
            "descripcion" => $table,
            "imagen" => "/resources/images/linfo.png",
            "icono" => "/resources/images/lsuccess.png"
        )
        $data[ "zoom" ]=8;
        $data[ "lugar" ]=$lugar;
        $data[ "array" ]=$mapas;
        $array=$datos;
    }
    $data[ 'datos' ]=$array;
}
catch(Exception $e)
{
    $data[ 'msgResult' ] = Errorlog_model::save($e-> getMessage(), __METHOD__);
    $data[ 'infoclass' ]="warning";
}
// $this->load->view('usuario/index', $data);
$this-> template-> write('header','','true');
$this-> template-> write('footer','','true');
$this-> template-> write('menu','','true');
$this-> template-> write('sala_prensa','','true');
$this-> template-> write_view('content',DIR_TES.'/reporteador/'.$pagina, $data);
$this-> template-> render();
}
}

```

# Archivo fuente para semana\_nacional.php

La documentación para este archivo está disponible en [semana\\_nacional.php](#)

```
1      <?php
2
3  /**
4   * Controlador Semana Nacional
5   *
6   * @package    TES
7   * @subpackage Controlador
8   * @author     Pascual
9   * @created    2014-03-07
10  */
11 class Semana_nacional extends CI_Controller {
12
13     public function __construct()
14     {
15         parent::__construct();
16
17         if(!$this-> db-> conn_id) {
18             $this-> template-> write('content', 'Error no se puede conectar a la Base de
19 Datos');
20             $this-> template-> render();
21
22         $this-> load-> helper('url');
23         $this-> load-> model(DIR_TES.'/Semana_nacional_model');
24     }
25
26     /**
27      * Lista todos los registros de semanas nacional, con su correspondiente paginación
28      * permite eliminar un elemento individual,
29      * muestra enlaces para actualizar y ver detalles de un elemento específico
30      *
31      * @access public
32      * @param int    $pag Establece el desplazamiento del primer registro a devolver
33      * @return void
34     */
35     public function index($pag = 0)
36     {
37         if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url())) {
38             show_error('', 403, 'Acceso denegado');
39             return false;
40         }
41
42         if(!isset($this-> Semana_nacional_model))
43             return false;
44
45         try {
46             $this-> load-> library('pagination');
47             $this-> load-> helper(array('form', 'formatFecha'));
48
49             $data = array();
50
51             $data['pag'] = $pag;
52             $data['msgResult'] = $this-> session-> flashdata('msgResult');
53             $data['clsResult'] = $this-> session-> flashdata('clsResult');
54             $data['title'] = 'Semana Nacional';
55
56             $registroEliminar = isset($_POST['registroEliminar']) ? $_POST['registroEliminar']
57 : null;
58
59             if( !empty($registroEliminar) ) {
60                 $this-> Semana_nacional_model-> delete($registroEliminar);
61                 $data['msgResult'] = 'Registros Eliminados exitosamente';
62                 $data['clsResult'] = 'success';
63             }
64
65             // Configuración para el Paginador
66             $configPag['base_url'] = site_url().DIR_TES.'/semana_nacional/index/';
```

```

66      $configPag['first_link'] = 'Primero';
67      $configPag['last_link'] = '&Uacute;ltimo' ;
68      $configPag['uri_segment'] = 4;
69      $configPag['total_rows'] = $this-> Semana_nacional_model-> getNumRows();
70      $configPag['per_page'] = 20;
71
72      $this-> pagination-> initialize($configPag);
73
74      $data['registros'] = $this-> Semana_nacional_model-
> getAll($configPag['per_page'], $spag);
75
76      } catch (Exception $e) {
77          $data['msgResult'] = Errorlog_model::save($this-> Semana_nacional_model-
> getMsgError(), __METHOD__);
78          $data['clsResult'] = 'error';
79      }
80
81      $this-> template-> write_view('content',DIR_TES.'/semana_nacional/index', $data);
82      $this-> template-> render();
83  }
84
85 /**
86 * Muestra el formulario para crear un nuevo registro en la semana_nacional,
87 * las variables se obtienen por el metodo POST
88 *
89 * @access public
90 * @return void
91 */
92 public function insert()
93 {
94     if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url())) {
95         show_error('', 403, 'Acceso denegado');
96         return false;
97     }
98
99     if(!isset($this-> Semana_nacional_model))
100        return false;
101
102    try {
103        $this-> load-> helper(array('form', 'formatFecha'));
104
105        $datos = $this-> input-> post();
106        $data['title'] = 'Crear un nuevo registro';
107        $data['msgResult'] = $this-> session-> flashdata('msgResult');
108        $data['clsResult'] = $this-> session-> flashdata('clsResult');
109
110        if(!empty($datos)) {
111            $this-> load-> library('form_validation');
112
113            $this-> form_validation-> set_rules('descripcion', 'Descripción',
114 'trim|xss_clean|max_length[45]|required');
115            $this-> form_validation-> set_rules('fecha_inicio', 'Fecha de inicio',
116 'required');
117            $this-> form_validation-> set_rules('fecha_fin', 'Fecha de fin',
118 'required');
119
120            if ($this-> form_validation-> run() === true) {
121                $this-> Semana_nacional_model-> setDescripcion($datos['descripcion']);
122                $this-> Semana_nacional_model-
123                > setFecha_inicio(formatFecha($datos['fecha_inicio'], 'Y-m-d'));
124                $this-> Semana_nacional_model-
125                > setFecha_fin(formatFecha($datos['fecha_fin'], 'Y-m-d'));
126
127                $this-> Semana_nacional_model-> insert();
128
129                $this-> session-> set_flashdata('msgResult', 'Registro guardado
exitosamente');
130                $this-> session-> set_flashdata('clsResult', 'success');
131
132                Bitacora_model::insert(DIR_TES.'::'.__METHOD__, 'Registro creado: '. $this-
> Semana_nacional_model-> getId());
133                redirect(DIR_TES.'/semana_nacional/', 'refresh');
134            } catch (Exception $e) {
135                $data['msgResult'] = Errorlog_model::save($this-> Semana_nacional_model-
> getMsgError(), __METHOD__);

```

```

136         $data['clsResult'] = 'error';
137     }
138
139     $this-> template-> write_view('content',DIR_TES.'/semana_nacional/insert', $data);
140     $this-> template-> render();
141 }
142
143 /**
144 * Muestra el formulario con los datos del registro especificado por el id,
145 * para actualizar sus datos
146 *
147 * @access public
148 * @param int    $id ID del elemento a actualizar
149 * @return void
150 */
151 public function update($id)
152 {
153     if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url())) {
154         show_error('', 403, 'Acceso denegado');
155         return false;
156     }
157
158     if(!isset($this-> Semana_nacional_model))
159         return false;
160
161     try {
162         $this-> load-> helper(array('form', 'formatFecha'));
163
164         $datos = $this-> input-> post();
165         $data['title'] = 'Actualizar datos del registro';
166         $data['registro'] = $this-> Semana_nacional_model-> getById($id);
167
168         if(!empty($datos)) {
169             $this-> load-> library('form_validation');
170
171             $this-> form_validation-> set_rules('descripcion', 'Descripción',
172 'trim|xss_clean|max_length[45]|required');
173             $this-> form_validation-> set_rules('fecha_inicio', 'Fecha de inicio',
174 'required');
175             $this-> form_validation-> set_rules('fecha_fin', 'Fecha de fin',
176 'required');
177
178             if ($this-> form_validation-> run() === true) {
179                 $this-> Semana_nacional_model-> setDescripcion($datos['descripcion']);
180                 $this-> Semana_nacional_model-
181 > setFecha_inicio(formatFecha($datos['fecha_inicio'], 'Y-m-d'));
182                 $this-> Semana_nacional_model-
183 > setFecha_fin(formatFecha($datos['fecha_fin'], 'Y-m-d'));
184
185                 $this-> Semana_nacional_model-> update($id);
186
187                 Bitacora_model::insert(DIR_TES.'::'.__METHOD__, 'Registro actualizado:
188 '. $id);
189                 $this-> session-> set_flashdata('msgResult', 'Registro guardado
190 exitosamente');
191                 $this-> session-> set_flashdata('clsResult', 'success');
192                 redirect(DIR_TES.'/semana_nacional/', 'refresh');
193                 die();
194             }
195         } catch (Exception $e) {
196             $data['msgResult'] = Errorlog_model::save($this-> Semana_nacional_model-
197 > errorMsg(), __METHOD__);
198             $data['clsResult'] = 'error';
199         }
200
201         $this-> template-> write_view('content',DIR_TES.'/semana_nacional/update', $data);
202         $this-> template-> render();
203 }
204
205 /**
206 * Muestra los datos del registro especificado por el id
207 *
208 * @access public
209 * @param int    $id ID del elemento a actualizar
210 * @return void
211 */
212 public function view($id)
213 {
214     $usuarios = array();

```

```

208
209     if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url())) {
210         show_error('', 403, 'Acceso denegado');
211         return false;
212     }
213
214     if(!isset($this-> Semana_nacional_model))
215         return false;
216
217     try {
218         $data['registro'] = $this-> Semana_nacional_model-> getById($id);
219         $data['title'] = 'Datos del registro';
220
221         $this-> load-> helper('formatFecha');
222
223         if( empty($data['registro']) ) {
224             $data['msgResult'] = 'ERROR: El registro solicitado no existe';
225             $data['clsResult'] = 'error';
226         }
227     } catch (Exception $e) {
228         $data['msgResult'] = Errorlog_model::save($this-> Semana_nacional_model-> getMsgError(), __METHOD__);
229         $data['clsResult'] = 'error';
230     }
231
232     $this-> template-> write_view('content',DIR_TES.'/semana_nacional/view', $data);
233     $this-> template-> write('menu','','true');
234     $this-> template-> write('sala_prensa','','true');
235     $this-> template-> render();
236 }
237
238 /**
239 * Eliminar el registro especificado por el id
240 *
241 * @access public
242 * @param int $id ID del elemento a eliminar
243 * @return void
244 */
245 public function delete($id)
246 {
247     if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url())) {
248         show_error('', 403, 'Acceso denegado');
249         return false;
250     }
251
252     if(!isset($this-> Semana_nacional_model))
253         return false;
254
255     try {
256         $this-> Semana_nacional_model-> delete($id);
257         $this-> session-> set_flashdata('msgResult', 'Registro eliminado exitosamente');
258         $this-> session-> set_flashdata('clsResult', 'success');
259     } catch (Exception $e) {
260         $this-> session-> set_flashdata('msgResult', Errorlog_model::save($this-> Semana_nacional_model-> getMsgError(), __METHOD__));
261         $this-> session-> set_flashdata('clsResult', 'error');
262     }
263
264     if (count($id) > 1)
265         Bitacora_model::insert(DIR_TES.'::'.__METHOD__, 'Registro eliminado: '.implode(',', $id));
266     else
267         Bitacora_model::insert(DIR_TES.'::'.__METHOD__, 'Registro eliminado: '.$id);
268     redirect(DIR_TES.'/semana_nacional/', 'refresh');
269     die();
270 }
271
272 /**
273 * Devuelve un json con todos los registros de semanas nacional
274 *
275 * @access public
276 * @return json
277 */
278 public function getAll()
279 {
280     $respuesta = array();
281
282     if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url())) {
283         $respuesta = array('error'=> true, 'msj_error'=> 'Acceso denegado');
284         echo json_encode($respuesta);

```

```

285     die();
286 }
287
288 if(!isset($this-> Semana_nacional_model)) {
289     $respuesta = array('error'=> true, 'msj_error'=> 'No se puede cargar el
modelo');
290     echo json_encode($respuesta);
291     die();
292 }
293
294 try {
295     $respuesta['registros'] = $this-> Semana_nacional_model-> getAll();
296
297     $respuesta['error'] = false;
298
299 } catch (Exception $e) {
300     $respuesta = array('error'=> true, 'msj_error'=> Errorlog_model::save($this-
> Semana_nacional_model-> getMsgError(), __METHOD__));
301     echo json_encode($respuesta);
302     die();
303 }
304
305 echo json_encode($respuesta);
306 }
307 ?>
308 }
```

# Archivo fuente para servicios.php

La documentación para este archivo está disponible en [servicios.php](#)

```
1  <?php
2  /**
3   * Controlador Servicios
4   *
5   * @package    TES
6   * @subpackage Controlador
7   * @author     Eliecer
8   * @created    2013-11-27
9   */
10  class Servicios extends CI_Controller {
11      public function __construct()
12      {
13          parent::__construct();
14
15          if(!$this-> db-> conn_id) {
16              $this-> template-> write('content', 'Error no se puede conectar a la Base de
Datos');
17              $this-> template-> render();
18
19              ini_set("buffering" , "0");
20              ob_start();
21
22              $this-> load-> helper('url');
23              $this-> load-> model(DIR_TES.'Tableta_model');
24              $this-> load-> model(DIR_TES.'Usuario_tableta_model');
25              $this-> load-> model(DIR_TES.'Enrolamiento_model');
26              $this-> load-> model(DIR_SIIGS.'Usuario_model');
27              $this-> load-> model(DIR_SIIGS.'ArbolSegmentacion_model');
28              $this-> load-> model(DIR_SIIGS.'ReglaVacuna_model');
29              $this-> load-> model(DIR_TES.'Reporte_sincronizacion_model');
30
31      /**
32      *
33      * Metodo principal al que se le hacen las peticiones y es el que se encarga de distribuir
la informacion
34      * recibe parametros por POST
35      * @param      int      $id_accion      Representa el tipo de accion que se ejecutara
36      * @param      int      $id_tab        Representa a la MAC de la tableta
37      * @param      int      $id_session    Representa la session activa para la peticion
38      * @param      int      $version_apk   Representa la version de la apk instalada en la
tableta
39      * @param      json     $datos        Representa el json que contiene el contenido
para la comunicacion tableta - servidor
40      *
41      * @return     void
42      */
43      public function Synchronization()
44      {
45          $id_accion=$this-> input-> post('id_accion');
46          $id_tab = $this-> input-> post('id_tab');
47          $id_sesion = $this-> input-> post('id_sesion');
48          $version_apk = $this-> input-> post('version_apk');
49          $datos = $this-> input-> post('msg');
50          if($datos=="")
51              $datos = $this-> input-> post('datos');
52
53          $this-> is_step_0(
54              json_encode(array("id_accion"      => $id_accion)),
55              json_encode(array("id_tab"        => $id_tab)),
56              json_encode(array("id_sesion"    => $id_sesion)),
57              $version_apk,$datos );
58      }
59
60 /**
61 *
62 * Paso 0 se procesa las peticiones segun la accion:
63 * Si la acci?n es 1: Valida la disponibilidad del dispositivo especificado y genera una
```

```

session que se mantiene activa en toda la sincronizacion
64      * Si la acci?n es 2: Regresa la informacion de todos los catalogos
65      * Si la acci?n es 3: Recibe un mensaje si es ok actualiza el estado de la tableta si es
error se crea un archivo log con la descripcion
66      * Si la acci?n es 4: Regresa la informacion de la persona que pertenezcan a la unidad
medica de la tableta
67      * Si la acci?n es 5: Recibe la informacion que envia la tableta y la almacena en sus
respectivas tablas
68      * Si la acci?n es 6: Regresa la informacion de los catalogos y personas que se
actualizaron o agregaron despues de la ultima sincronizacion de la tableta
69      *
70      * @param      int      $id_accion      Representa el tipo de accion que se ejecutara
71      * @param      int      $id_tab       Representa a la MAC de la tableta
72      * @param      int      $id_sesion    Representa la session activa para la peticion
73      * @param      int      $version_apk  Representa la version de la apk instalada en la
tableta
74      * @param      json     $datos       Representa el json que contiene el contenido
para la comunicacion tableta - servidor
75      *
76      * @return     void
77      */
78
79  public function is_step_0($id_accion, $id_tab = null, $id_sesion = null, $id_version =
null, $datos = null)
80  {
81      if(!isset($this->  Tableta_model))
82          echo json_encode(array("id_resultado"           =>      'No hay conexi?n'));
83      try
84      {
85          $this->  load->  library('session');
86          $id_accion=json_decode($id_accion);
87          $id_tab=json_decode($id_tab);
88          $id_sesion=json_decode($id_sesion);
89          $this->  session->  set_userdata( 'sinc', "1" );
90          switch($id_accion->  id_accion)
91          {
92              case 1: // debe existir la MAC
93                  $this->  is_step_1($id_tab->  id_tab, $id_version);
94                  break;
95              case 2: // debe existir el token y se regresa la info del dispositivo
96                  $this->  is_step_2($id_sesion->  id_sesion);
97                  break;
98              case 3: // debe existir el token y el dato de la descripcion del proceso
99                  $this->  is_step_3($id_sesion->  id_sesion, $datos);
100                 break;
101             case 4: // debe existir el token
102                 $this->  is_step_4($id_sesion->  id_sesion);
103                 break;
104             case 5: // debe existir el token y el dato de la descripcion del proceso
105                 $this->  ss_step_5($id_sesion->  id_sesion, $datos);
106                 break;
107             case 6: // debe existir el token
108                 $this->  ss_step_6($id_sesion->  id_sesion);
109                 break;
110             default:
111                 $this->  session->  sess_destroy();
112         }
113     }
114     catch (Exception $e)
115     {
116         Errorlog_model::save($e->  getMessage(), __METHOD__);
117     }
118 }
119 /**
120 *
121 * valida que la tableta este asignada a una unidad medica y que tenga un status valido
para la sincronizacion
122 * recibe parametros por POST
123 * @param      int      $id_tab       Representa a la MAC de la tableta
124 * @param      int      $version_apk  Representa la version de la apk instalada en la
tableta
125 *
126 * @return     session
127 */
128 public function is_step_1($id_tab,$id_version)
129 {
130     $tableta = $this->  Tableta_model->  getByMac($id_tab);
131     if ($count($tableta) == 1)
132     {
// debe tener usuarios asignados, el tipo de censo y la unidad m?dica

```

```

134         if ( $tableta-> usuarios_asignados == 1 )
135     {
136         if( $tableta-> id_tes_estado_tableta == 2 || $tableta-
137 > id_tes_estado_tableta == 3 || $tableta-> id_tes_estado_tableta == 4 )
138     {
139         if($tableta-> id_tes_estado_tableta != 4)
140         {
141             if( $tableta-> id_tipo_censo != null )
142             {
143                 if( $tableta-> id_asu_um != null )
144                 {
145                     // se crea el token temporal
146                     $token = md5(date("dmYHis" ) );
147                     $this-> session-> set_userdata( 'session' , $token );
148                     $this-> session-> set_userdata( 'mac' , $id_tab );
149                     $this-> session-> set_userdata( 'fecha' , $tableta-
150 > ultima_actualizacion );
151                     $this-> session-> set_userdata( 'pasos' , "1" );
152                     $this-> session-> set_userdata( 'id_version' , $id_version );
153                     $this-> session-> set_userdata( 'dias_extras' , $tableta-
154 > periodo_esq_inc );
155                     echo json_encode(array("id_sesion" => $this-
156 > session-> userdata('session')));
157                     ob_flush();
158                 }
159             }
160         }
161     }
162     {
163         echo json_encode(array("id_resultado" => 'Unidad
164 medica Nulo'));
165     }
166     else
167     {
168         echo json_encode(array("id_resultado" => 'Censo
169 Nulo'));
170     }
171     else
172     {
173         $mi_version = $this-> Enrolamiento_model-> get_version();
174         foreach($mi_version as $dato)
175         {
176             echo json_encode(array("id_resultado" =>
177             $dato-> host));
178             ob_flush();
179             die();
180         }
181     }
182     else
183     {
184         echo json_encode(array("id_resultado" => 'Estatus no valido'));
185     }
186     else
187     {
188         echo json_encode(array("id_resultado" => 'Tableta sin
189 configurar'));
190     }
191     else
192     {
193         echo json_encode(array("id_resultado" => 'Tableta desconocida'));
194     }
195 }
196 /**
197 *
198 * Valida que la session este activa, genera los catalogos a enviar en la sincronizacion
199 por primera vez
200 *
201 * @param int $id_session Representa la session activa para la peticion
202 * @param int $si Bandera que especifica si este paso es llamado
203 * @param json $sf Bandera que especifica el comportamiento del

```

```

armado del json
204      *
205      * @return          echo
206      */
207      public function is_step_2($id_sesion, $si="" , $sf="" )
208      {
209          ini_set("max_execution_time" , 999999999);
210          ini_set("memory_limit" , "-1" );
211          $micadena= "";
212          $misesion=$this-> session-> userdata('session');
213          $mac=$this-> session-> userdata('mac');
214
215          $mi_version = $this-> Enrolamiento_model-> get_version();
216          foreach($mi_version as $dato)
217          {
218              if($this-> session-> userdata('id_version') < $dato-> version)
219              {
220                  echo json_encode(array("id_resultado" => 'Desactualizado',
221 "url" => $dato-> host));
222                  ob_flush();
223                  die();
224              }
225              if ($id_sesion == $this-> session-> userdata('session')) // valida el token de
226              // entrada es el token que solicito el servicio
227              {
228                  // se obtiene el dispositivo por token
229                  $tableta = $this-> Tableta_model-> getByMac($this-> session-
230 > userdata('mac'));
231                  // se obtienen los usuarios asignados, el tipo de censo y la unidad m?dica
232                  if ($tableta-> usuarios_asignados == 1 && $tableta-> id_tipo_censo != null &&
233                  $tableta-> id_asu_um != null)
234                  {
235                      //***** inicio usuario *****
236                      // obtiene usuarios de las tabletas
237                      $cadena["id_tipo_censo"] = $tableta-> id_tipo_censo;
238                      $cadena["id_asu_um"] = $tableta-> id_asu_um;
239                      $cadena["id_nivel_atencion"] = 1;
240
241                      $micadena=json_encode($cadena);
242                      echo substr($micadena,0,strlen($micadena)-1)."";
243                      ob_flush();
244                      unset($cadena);
245                      $cadena=array();
246                      $inusuario=array(0);
247                      $usuariosXtableta = $this-> Usuario_tableta_model-
248 > getUsuariosByTableta($tableta-> id);
249
250                      if($usuariosXtableta)
251                      {
252                          foreach($usuariosXtableta as $dato)
253                          array_push($inusuario,$dato-> id_usuario);
254                      }
255                      $permisos = $this-> Usuario_model-> get_permiso_entorno("TES
256 Movil");
257
258                      if($permisos)
259                      {
260                          $cadena["sis_permiso"] = $permisos;
261                          $micadena=json_encode($cadena);
262                          echo substr($micadena,1,strlen($micadena)-2)."";
263                          $micadena="";
264                          ob_flush();
265                          unset($cadena);
266                          $cadena=array();
267                      }
268                      $grupos = $this-> Usuario_model-> get_grupo_entorno("TES Movil");
269                      if($grupos)
270                      {
271                          $cadena["sis_grupo"] = $grupos;
272                          $micadena=json_encode($cadena);
273                          echo substr($micadena,1,strlen($micadena)-2)."";
274                          $micadena="";
275                          ob_flush();
276                          unset($cadena);
277                          $cadena=array();
278                      }
279                      if(sizeof($inusuario)==0)array_push($inusuario,0);

```

```

277      $usuarios = $this->  Usuario_model->  get_usuario_entorno("TES
278      ,&$inusuario);
279      if($usuarios)
280      {
281          $cadena["sis_usuario"] = $usuarios;
282          $micadena=json_encode($cadena);
283          echo substr($micadena,1,strlen($micadena)-2)." ";
284          $micadena="";
285          ob_flush();
286          unset($cadena);
287          $cadena=array();
288      }
289      //***** fin usuario *****
290      //***** inicioasu *****
291      $inicio_asu=0;
292      if($si=="")
293      {
294          /*$count=$this->Enrolamiento_model-
295 >get_catalog_count("asu_arbol_segmentacion","id_raiz","1");
296          $mas=$count%8000;
297          $contador=$count/8000;
298          if($mas>0)(int)$contador++;
299          if($count>0)
300              $cadena["asu_arbol_segmentacion"] =array();
301              $micadena=json_encode($cadena);
302              echo substr($micadena,1,strlen($micadena)-3);
303              $micadena="";
304              ob_flush();
305              unset($cadena);
306              $cadena=array();
307
308          for($i=0;$i<$contador;$i++)
309          {
310              $fecha=$this->session->userdata('fecha');
311              if($sf=="")
312                  $asu=$this->Enrolamiento_model-
313 >get_catalog2("asu_arbol_segmentacion","id_raiz","1","","&qu
314 ot;",($i*8000),8000);
315              else
316                  $asu=$this->Enrolamiento_model-
317 >get_catalog2("asu_arbol_segmentacion","fecha_update
318 >=",$fecha,"id_raiz","1",($i*8000),8000);
319
320          if($asu)
321          {
322              $micadena=json_encode($asu);
323              echo substr($micadena,1,strlen($micadena)-2);
324
325          if($count>0)
326              echo "]";
327              ob_flush();*/
328
329
330          $fecha=$this->  session->  userdata('fecha');
331          if($sf=="")
332          {
333              $asu=$this->  Enrolamiento_model-
334 >  get_catalog2("asu_arbol_segmentacion" , "fecha_update >" , '2014-08-17
16:59:02' , "id_raiz" , "1" );
335              // $asu=$this->Enrolamiento_model-
336 >get_catalog2("asu_arbol_segmentacion","id_raiz","1");
337
338          else
339              $asu=$this->  Enrolamiento_model-
340 >  get_catalog2("asu_arbol_segmentacion" , "fecha_update
341 >=" , $fecha , "id_raiz" , "1" );
342
343          if($asu)
344          {
345              $cadena["asu_arbol_segmentacion"] = $asu;
346              $micadena=json_encode($cadena);
347              echo substr($micadena,1,strlen($micadena)-2);
348              $micadena="";
349              ob_flush();

```

```

346         unset($cadena);
347         $cadena=array();
348         $inicio_asu=1;
349     }
350     else
351     {
352
353         $cadena["asu_arbol_segmentacion"] = array(array(
354             "id"          => "0"
355         )));
356         $micadena=json_encode($cadena);
357         echo substr($micadena,1,strlen($micadena)-2);
358         $micadena="";
359         ob_flush();
360         unset($cadena);
361         $cadena=array();
362         $inicio_asu=1;
363     }
364     else
365     {
366         $cadena["asu_arbol_segmentacion"] = array(array(
367             "id"          => "0"
368         ));
369         $micadena=json_encode($cadena);
370         echo substr($micadena,1,strlen($micadena)-2);
371         $micadena="";
372         ob_flush();
373         unset($cadena);
374         $cadena=array();
375     }
376 //***** finasu *****
377
378 //***** inicio notificacion *****
379 $asu_um = $this->ArbolSegmentacion_model->getUMParentsById($tableta-
> id_asu_um);
380
381 $i=0;$array=array();$tem="";
382 if($asu_um)
383 foreach($asu_um as $id)
384 {
385     $result=$this->Enrolamiento_model->get_notificacion($id);
386     if($result)
387     {
388         if($tem!=$result[0]->id)
389         {
390             if($i==0)
391                 $array=$result;
392             else
393                 array_push($array,$result[0]);
394             $i++;
395         }
396         $tem=$result[0]->id;
397     }
398 //    $cadena["tes_notificacion"]='Error recuperando
399
400     tes_notificacion';
401 }
402 $noti=0;
403 if(count($array)>0)
404 {
405     if($inicio_asu==1)
406         echo ",";
407     $cadena["tes_notificacion"] = $array;
408     $micadena=json_encode($cadena);
409     echo substr($micadena,1,strlen($micadena)-2);
410     $micadena="";
411     ob_flush();
412     unset($cadena);
413     $cadena=array();
414     $noti=1;
415 }
416 //***** fin notificacion *****
417
418 //***** inicio catalogos *****
419 if($sf=="")
420 $this->catalogos_relevantes();
421 //***** fin catalogos *****
422

```

```

423                                     //***** inicio tes_pendientes_tarjeta *****
424                                     $pendiente=$this-> Enrolamiento_model-
425 > get_catalog2("tes_pendientes_tarjeta" , "resuelto" , "0" );
426                                     {
427                                         if($pendiente)
428                                         {
429                                             if($noti==1||$inicio_asu==1)
430                                             echo ",";
431                                             $cadena["tes_pendientes_tarjeta"] = $pendiente;
432                                             $micadena=json_encode($cadena);
433                                             echo substr($micadena,1,strlen($micadena)-2);
434                                             $micadena="";
435                                             ob_flush();
436                                             unset($cadena);
437                                             $cadena=array();
438                                         }
439                                         //***** fin tes_pendientes_tarjeta *****
440                                         // regresa el json con los datos necesarios
441                                         $this-> session-> set_userdata( 'paso' , "2" );
442                                         if($sf=="")
443                                         echo "}";
444                                         else
445                                         {
446                                             echo json_encode(array("id_resultado" => 'Tableta sin
configurar'));
447                                         ob_flush();
448                                         }
449                                         }
450                                         else
451                                         {
452                                             echo json_encode(array("id_resultado" => "Error de
procedimiento"));
453                                         ob_flush();
454                                         }
455                                         }
456                                         /**
457                                         *
458                                         * Guarda en la base de datos el estado de la sincronizacion
459                                         *
460                                         * @param int $id_session Representa la session activa para la peticion
461                                         * @param json $datos Representa el json que contiene el contenido
para la comunicacion tableta - servidor
462                                         *
463                                         * @return void
464                                         */
465                                         public function is_step_3($id_sesion,$datos)
466                                         {
467                                             $fp = fopen(APPPATH."logs/sincronizacionsecuencial.txt" , "a" );
468                                             fputs($fp, "FECHA: ".date("d/m/Y H:i:s") . " => MAC:" .
469                                             $this-> session-> userdata('mac')." => VERSION:" .
470                                             $this-> session-> userdata('id_version')." => PASO:" .
471                                             $this-> session-> userdata('paso')." JSON recibido:
" .($datos). "\r\n" );
472                                             $datos=(array)json_decode($datos);
473                                             $paso=$this-> session-> userdata('paso');
474                                             $sinc=$this-> session-> userdata('sinc');
475                                             if($datos["id_resultado"]=="error")
476                                             {
477                                                 fputs($fp, "ERROR: " . $datos[ "descripcion" ] . "\r\n" );
478                                                 ob_flush();
479                                             }
480                                             if($datos["id_resultado"]=="ok" && $this-> session-
481                                             userdata('paso')=="4")
482                                             {
483                                                 $this-> actualiza_estado_tableta($id_sesion,"3" ,
484                                             $this-> session-
485                                             userdata('id_version'));
486                                                 ob_flush();
487                                             if($datos["id_resultado"]=="ok" && $this-> session-
488                                             userdata('paso')=="6")
489                                             {
490                                                 $this-> actualiza_estado_tableta($id_sesion,"3" ,
491                                             $this-> session-
492                                             userdata('id_version'));
493                                         }
494                                         fclose($fp);
495                                         }

```

```

494     /**
495      *
496      * Prepara la informacion de personas con su catalogos transaccionales de cada una
497      *
498      * @param      int          $id_session      Representa la session activa para la peticion
499      *
500      * @return     echo
501      */
502     public function is_step_4($id_sesion)
503     {
504         ini_set("max_execution_time" , 999999999);
505         $tableta = $this-> Tableta_model-> getByMac($this-> session-> userdata('mac'));
506         if($tableta-> id_tipo_censo==1)
507             ini_set("memory_limit" , "300M" );
508         if($tableta-> id_tipo_censo==2)
509             ini_set("memory_limit" , "200M" );
510         if($tableta-> id_tipo_censo==3)
511             ini_set("memory_limit" , "100M" );
512         if ($id_sesion == $this-> session-> userdata('session')) // valida el token de
513         // entrada es el token que solicito el servicio
514         {
515             $cadena= "";
516             // se obtiene el dispositivo por token
517             $tableta = $this-> Tableta_model-> getByMac($this-> session-
518             > userdata('mac'));
519             // se obtienen los usuarios asignados, el tipo de censo y la unidad m?dica
520             if ($tableta-> usuarios_asignados == 1 && $tableta-> id_tipo_censo != null &&
521             $tableta-> id_asu_um != null)
522             {
523                 //***** inicio persona *****
524                 $asu_um = $this-> ArbolSegmentacion_model-> getUMParentsById($tableta-
525                 > id_asu_um);
526                 if($tableta-> id_tipo_censo!=5)
527                 {
528                     $asu_um = array_reverse($asu_um);
529                     $asu_um = $this-> ArbolSegmentacion_model-
530                     > getCluesFromId($asu_um[$tableta-> id_tipo_censo-1]);
531                 }
532                 else
533                     $asu_um[ 'children' ][0]=array( "key" => $tableta-> id_asu_um);
534                 $i=0;
535                 $miasu=array();
536                 foreach($asu_um[ "children" ] as $id)
537                 {
538                     $miasu[]=$id[ "key" ];
539                 }
540                 $personas=$this-> Enrolamiento_model-> get_cns_persona($miasu);
541                 if($personas)
542                 {
543                     $cadena[ "cns_persona" ]= $personas;
544                     $micadena=json_encode($cadena);
545                     echo "{" . substr($micadena,1,strlen($micadena)-2);
546                     $micadena="";
547                     ob_flush();
548                     unset($cadena);
549                     $cadena=array();
550                     //***** inicio control catalogos X persona *****
551                     $regla_vacuna=array();
552                     $mipersona=array();
553                     foreach($personas as $persona)
554                     {
555                         $vacunas=$array=$this-> Enrolamiento_model-
556                         > get_catalog2("cns_control_vacuna" , "id_persona" , $persona-> id);
557                         array_push($regla_vacuna,$this-> esquema_incompleto($persona-
558                         > id,$persona-> fecha_nacimiento,$vacunas));
559                         $mipersona[]=$persona-> id;
560                     }
561                     $catalog_relevante = $this-> Enrolamiento_model-
562                     > get_transaction_relevante();
563                     foreach($catalog_relevante as $catalog)
564                     {
565                         if($catalog-> descripcion!="cns_persona" && $catalog-
566                         > descripcion!="cns_tutor")
567                         {
568                             try
569                             {

```

```

565                                         if($catalog-> descripcion=="cns_persona_x_tutor" )          )
566                                         {
567                                         $array=$this-> Enrolamiento_model-
568                                         descripcion, $mipersona);
569                                         $mitutor=array();
570                                         foreach($array as $dato)
571                                         {
572                                         $mitutor[]=$dato-> id_tutor;
573                                         }
574                                         $array2=$this-> Enrolamiento_model-
575                                         echo " , ";
576                                         $cadena[ "cns_tutor" ]= $array2;
577                                         $micadena=json_encode($cadena);
578                                         echo substr($micadena,1,strlen($micadena)-2);
579                                         $micadena="";
580                                         ob_flush();
581                                         unset($cadena);
582                                         $cadena=array();
583                                         }
584                                         $count=$this-> Enrolamiento_model-
585                                         > get_cns_cat_persona_count($catalog-> descripcion, $mipersona);
586                                         $mas=$count%15000;
587                                         $contador=$count/15000;
588                                         if($mas> 0)(int)$contador++;
589                                         if($count> 0)
590                                         {
591                                         $cadena[$catalog-> descripcion]=array();
592                                         $micadena=json_encode($cadena);
593                                         echo " , ".substr($micadena,1,strlen($micadena)-3);
594                                         $micadena="";
595                                         ob_flush();
596                                         unset($cadena);
597                                         $cadena=array();
598                                         $array=array();
599                                         for($i=0;$i< $contador;$i++)
600                                         {
601                                         $array=$this-> Enrolamiento_model-
602                                         descripcion, $mipersona, ($i*15000),15000);
603                                         if($array)
604                                         {
605                                         $micadena=json_encode($array);
606                                         if($i> 0)echo " , ";
607                                         echo substr($micadena,1,strlen($micadena)-2);
608                                         $micadena="";
609                                         ob_flush();
610                                         }
611                                         if($count> 0)
612                                         {
613                                         echo " ] ";
614                                         ob_flush();
615                                         }
616                                         }
617                                         }
618                                         catch (Exception $e) {Errorlog_model::save($e-> getMessage(),
619                                         __METHOD__);}
620                                         }
621                                         }
622                                         }
623                                         $rv=array();
624                                         for($x=0;$x< count($regla_vacuna);$x++)
625                                         for($y=0;$y< count($regla_vacuna[$x]);$y++)
626                                         $rv[]=array("id_persona" => $regla_vacuna[$x][$y][ "id_persona" ],
627                                         "id_vacuna" =>
628                                         $regla_vacuna[$x][$y][ "id_vacuna" ],
629                                         "prioridad" =>
630                                         $regla_vacuna[$x][$y][ "prioridad" ],
631                                         $cadena[ "esquema_incompleto" ]=$rv,
632                                         //***** fin control catalogos X persona *****
633                                         }
634                                         else
635                                         {
636                                         echo json_encode(array("cns_persona" => array()));
637                                         $array=$this-> Reporte_sincronizacion_model-> getListado("SELECT id

```

```

FROM cns_persona WHERE activo=0" );
637     if($array)
638     {
639         $data=array();
640         foreach($array as $x)
641         {
642             $data[]=$x-> id;
643         }
644         echo ",";
645         $micadena["persona_x_borrar"] = $data;
646         $micadena=json_encode($micadena);
647         echo substr($micadena,1,strlen($micadena)-2);
648         $micadena="";
649         ob_flush();
650         unset($data);
651     }
652
653     // regresa el json con los datos necesarios
654     $this-> session-> set_userdata( 'paso' , "4" );
655     if($cadena!="")
656     {
657         echo ",";
658         $micadena=json_encode($cadena);
659         echo substr($micadena,1,strlen($micadena)-1);
660         $micadena="";
661         ob_flush();
662         unset($cadena);
663         $cadena=array();
664     }
665
666     //***** fin persona *****
667 }
668 else
669 {
670     echo json_encode(array("id_resultado" => 'Tableta sin
configurar'));
671 }
672 }
673 else
674 {
675     echo json_encode(array("id_resultado" => "Error de
procedimiento"));
676     ob_flush();
677 }
678 }
679 /**
680 *
681 * Recibe los datos que genero la tableta para ser almacenados en la base de datos del
682 servidor
683 *
684 * @param int $id_session Representa la session activa para la peticion
685 * @param json $datos Representa el json que contiene el contenido
para la comunicacion tableta - servidor
686 *
687 * @return echo
688 */
689 public function ss_step_5($id_sesion, $datos)
690 {
691     header('Content-Type: text/html; charset=UTF-8');
692     $bien=0;
693     $fp = fopen(APPPATH."logs/sincronizacionsecuencial.txt" , "a");
694     fputs($fp, "FECHA: ".date("d/m/Y H:i:s")." => MAC:" .
$this-> session-> userdata('mac')." => VERSION:" .
$this-> session-> userdata('id_version')." => PASO:" .
$this-> session-> userdata('paso')." JSON recibido:
");
695     ($datos)."\r\n";
696     $datos=(array)json_decode($datos);
697     try
698     {
699         if(array_key_exists("cns_persona" , $datos))
700             foreach($datos["cns_persona"] as $midato)
701             {
702                 if($this-> Enrolamiento_model-
> get_catalog2("cns_persona" , "id" , $midato-> id))
703                     $this-> Enrolamiento_model-
704                     > cns_update("cns_persona" , $midato,$midato-> id);
705                 else
706                     $this-> Enrolamiento_model-
707                     > cns_insert("cns_persona" , $midato);
708             }
709     }
710 }

```

```

708 }
709 if(array_key_exists("cns_visita" , $datos))
710 foreach($datos["cns_visita"] as $visita)
711 {
712     $this-> Enrolamiento_model-> cns_insert("cns_visita" , $visita);
713     if($visita-> id_estado_visita==1 || $visita-> id_estado_visita==4)
714         $this-> Enrolamiento_model-> cns_update_visita($visita-> id_persona);
715     if($visita-> id_estado_visita!=1 && $visita-> id_estado_visita!=3)
716         $this-> Enrolamiento_model-
717     > cns_update("cns_persona" , array("contador_visitadas" => '0'), $visita-
718     > id_persona);
719     $contador=$this-> Enrolamiento_model-
720 > get_catalog2("cns_persona" , "id" , $visita-> id_persona);
721     if($visita-> id_estado_visita==3 || $contador[0]-> contador_visitadas==3)
722         $this-> Enrolamiento_model-
723     > cns_update("cns_persona" , array("activo" => '0', 'ultima_actualizacion' =>
724     date("Y-m-d H:i:s") ), $visita-> id_persona);
725     if(array_key_exists("cns_tutor" , $datos))
726     foreach($datos["cns_tutor"] as $midato)
727     {
728         if($this-> Enrolamiento_model-
729     > get_catalog2("cns_tutor" , "id" , $midato-> id))
730         $this-> Enrolamiento_model-
731     > cns_update("cns_tutor" , $midato, $midato-> id);
732         else
733             $this-> Enrolamiento_model-> cns_insert("cns_tutor" , $midato);
734     }
735     catch (Exception $e) {Errorlog_model::save($e-> getMessage(), __METHOD__); $bien++;}
736     $catalog_relevante = $this-> Enrolamiento_model-> get_transaction_relevante();
737     foreach($catalog_relevante as $catalog)
738     {
739         if($catalog-> descripcion!="cns_persona" && $catalog-
740     > descripcion!="cns_tutor")
741         try
742         {
743             if(array_key_exists($catalog-> descripcion, $datos))
744             foreach($datos[$catalog-> descripcion] as $midato)
745             {
746                 $f_valor="";
747                 if(array_key_exists("id" , $midato))
748                 {
749                     $b_campo="id";
750                     $b_valor=$midato-> id;
751                 }
752                 else if(array_key_exists("id_persona" , $midato))
753                 {
754                     $b_campo="id_persona";
755                     $b_valor=$midato-> id_persona;
756                     if($catalog-
757     > descripcion=="cns_antiguo_domicilio" || $catalog-
758     > descripcion=="cns_antigua_um")
759                     {
760                         $f_campo='fecha_cambio';
761                         $f_valor=$midato-> fecha_cambio;
762                     }
763                     else if($catalog-> descripcion=="cns_persona_x_alergia")
764                     {
765                         $f_campo='ultima_actualizacion';
766                         $f_valor=$midato-> ultima_actualizacion;
767                     }
768                     else
769                         $f_campo='fecha';
770                         if($f_valor=="")
771                             $f_valor=$midato-> fecha;
772                     if(array_key_exists("id_invitado" , $midato))
773                         unset($midato-> id_invitado);
774                     if($this-> Enrolamiento_model-> get_catalog2($catalog-
775     > descripcion, $b_campo, $b_valor, $f_campo, $f_valor))
776                         $this-> Enrolamiento_model-> cns_update($catalog-

```

```

>     descripcion,$midato,""           ,$b_campo,$b_valor,$f_campo,$f_valor);
776     else                           $this-> Enrolamiento_model-> cns_insert($catalog-
777     descripcion,$midato);
778   }
779   }
780   }
781   catch (Exception $e) {Errorlog_model::save($e-> getMessage(), __METHOD__);}
782 }
783 if($bien==0)
784 {
785   if(array_key_exists("tes_pendientes_tarjeta" , $datos))
786     foreach($datos["tes_pendientes_tarjeta"]      ] as $midato)
787   {
788     if($this-> Enrolamiento_model-
789     get_catalog2("tes_pendientes_tarjeta" , "fecha"           , $midato)
790     fecha,"id_persona" , $midato-> id_persona))
791     $this-> Enrolamiento_model-
792     cns_update("tes_pendientes_tarjeta" , $midato,$midato-
793     fecha,"id_persona" , $midato-> id_persona);
794     else
795       $this-> Enrolamiento_model-
796     cns_insert("tes_pendientes_tarjeta" , $midato);
797   }
798   $this-> Enrolamiento_model-> tes_pendientes_tarjeta_delete();
799
800   if(array_key_exists("sis_bitacora" , $datos))
801     foreach($datos["sis_bitacora"]      ] as $midato)
802   {
803     $this-> Enrolamiento_model-> cns_insert("sis_bitacora" , $midato);
804   }
805   if(array_key_exists("sis_error" , $datos))
806     foreach($datos["sis_error"]      ] as $midato)
807   {
808     $this-> Enrolamiento_model-> cns_insert("sis_error" , $midato);
809   }
810   $this-> session-> set_userdata( 'paso' , "5");
811   $mi_version = $this-> Enrolamiento_model-> get_version();
812   foreach($mi_version as $dato)
813   {
814     if($this-> session-> userdata('id_version')< $dato-> version)
815     {
816       $this-> actualiza_estado_tableta($id_sesion, "4" , $this-
817       "url"      => $dato-> host );
818       ob_flush();
819       die();
820     }
821   }
822   else
823   {
824     echo json_encode(array("id_resultado"      => 'ok', "version"
825     ob_flush());
826   }
827 /**
828 *
829 * prepara los datos para la sincronizacion secuencia, envia unicamente aquellos datos
830 * modificados despues de la ultima sincronizacion de la tabla
831 * @param      int      $id_session      Representa la session activa para la peticion
832 *
833 * @return      void
834 */
835 public function ss_step_6($id_sesion)
836 {
837   ini_set("max_execution_time" , 999999999);
838   ini_set("memory_limit" , "200M" );
839   $fecha=$this-> session-> userdata('fecha');
840   $mi_version = $this-> Enrolamiento_model-> get_version();
841   foreach($mi_version as $dato)
842   {
843     if($this-> session-> userdata('id_version')< $dato-> version)
844     {

```

```

845             echo json_encode(array("id_resultado"           =>      'Desactualizado',
846             => $dato-> host));
847         }
848     }
849     if ($id_sesion == $this-> session-> userdata('session')) // valida el token de
850     // entrada es el token que solicito el servicio
851     {
852         // se obtiene el dispositivo por token
853         $tableta = $this-> Tableta_model-> getByMac($this-> session-
854         > userdata('mac'));
855         // se obtienen los usuarios asignados, el tipo de censo y la unidad m?dica
856         if ($tableta-> usuarios_asignados == 1 && $tableta-> id_tipo_censo != null &&
857             $tableta-> id_asu_um != null)
858         {
859             //***** inicio usu *****
860             $this-> is_step_2($id_sesion,"", "si");
861             //***** fin usu *****
862             //***** inicio catalogos *****
863             $this-> catalogos_relevantes("si");
864             //***** fin catalogos *****
865             //***** inicio persona *****
866             $asu_um = $this-> ArbolSegmentacion_model-> getUMParentsById($tableta-
867             > id_asu_um);
868             if($tableta-> id_tipo_censo!=5)
869             {
870                 $asu_um = array_reverse($asu_um);
871                 $asu_um = $this-> ArbolSegmentacion_model-
872                 > getCluesFromId($asu_um[$tableta-> id_tipo_censo-1]);
873             }
874             else
875                 $asu_um['children'][0]=array("key" => $tableta-> id_asu_um);
876             $miusu=array();
877             foreach($asu_um["children"] as $id)
878             {
879                 $miusu[]=$id["key"];
880             }
881             $personas=$this-> Enrolamiento_model-> get_cns_persona($miusu,$fecha);
882             if($personas)
883             {
884                 $cadena[ "cns_persona" ]= $personas;
885                 $micadena=json_encode($cadena);
886                 echo ", ". substr($micadena,1,strlen($micadena)-2);
887                 $micadena="";
888                 ob_flush();
889                 unset($cadena);
890                 $cadena=array();
891                 //***** inicio control catalogos X persona *****
892                 $mipersona=array();
893                 foreach($personas as $persona)
894                 {
895                     $mipersona[]=$persona-> id;
896                 }
897                 //***** inicio control catalogos X persona *****
898                 $catalog_relevante = $this-> Enrolamiento_model-
899                 > get_transaction_relevante();
900                 foreach($catalog_relevante as $catalog)
901                 {
902                     if($catalog-> descripcion!="cns_tutor" && $catalog-
903                     > descripcion!="cns_tutor")
904                     {
905                         try
906                         {
907                             $array=$this-> Enrolamiento_model-
908                             > get_cns_cat_persona($catalog-> descripcion, $mipersona);
909                             if($array)
910                             {
911                                 echo ", ";
912                                 $cadena[$catalog-> descripcion]=$array;
913                                 $micadena=json_encode($cadena);
914                                 echo substr($micadena,1,strlen($micadena)-2);
915                                 $micadena="";
916                                 ob_flush();
917                                 unset($cadena);
918                                 $cadena=array();
919                             }
920                         }
921                     }
922                 }
923             }
924         }
925     }

```

```

916
917 >     descripcion=="cns_persona_x_tutor"
918
919
920
921
922
923
924
925 >     get_persona_x_tutor($mitutor);
926
927
928
929
930
931
932
933
934
935
936
937     METHOD__);
938 }
939
940
941
942     $xy++;
943
944 //***** fin control catalogos X persona *****
945
946     $micadena=json_encode($cadena);
947     echo substr($micadena,1,strlen($micadena)-2);
948     $micadena="";
949     ob_flush();
950     unset($cadena);
951     $cadena=array();
952
953     $regla_vacuna=array();
954     $personas=$this->    Enrolamiento_model->    get_cns_persona($miasu);

955     if($personas)
956     {
957         $regla_vacuna=array();
958         foreach($personas as $persona)
959         {
960             $vacunas=$array=$this->    Enrolamiento_model-
961             >    get_catalog2("cns_control_vacuna" , "id_persona" , $persona->    id);
962             >    id,$persona->    fecha_nacimiento,$vacunas));
963         }
964
965         $rv=array();
966         for($x=0;$x<    count($regla_vacuna);$x++)
967             for($y=0;$y<    count($regla_vacuna[$x]);$y++)
968
969             $rv[]=[["id_persona"      =>    $regla_vacuna[$x][$y][ "id_persona"      =>
970                     "id_vacuna"          1,
971                     "prioridad"          =>
972                     $regla_vacuna[$x][$y][ "prioridad"          1];
973                     $cadena[ "esquema_incompleto" ]=$rv;
974
975                     $micadena=json_encode($cadena);
976                     echo (substr($micadena,1,strlen($micadena)-2));
977                     $micadena="";
978                     ob_flush();
979                     unset($cadena);
980                     $cadena=array();
981
982                     $array=$this->    Reporte_sincronizacion_model->    getListado("SELECT id
983 FROM cns_persona WHERE activo=0");
984                     if($array)
985                     {
986                         $data=array();
987                         foreach($array as $x)
988                         {

```

```

986                     $data[] = $x-> id;
987                 }
988                 echo ", ";
989                 $micadena[ "persona_x_borrar" ] = $data;
990                 $micadena = json_encode( $micadena );
991                 echo substr( $micadena, 1, strlen( $micadena ) - 2 );
992                 $micadena = "";
993                 ob_flush();
994                 unset( $data );
995             }
996             // regresa el json con los datos necesarios
997             $this-> session-> set_userdata( 'paso', "6" );
998         }
999         echo "}";
1000         ob_flush();
1001     }
1002 }
1003 else
1004 {
1005     echo json_encode( array( "id_resultado" => 'Tableta sin
configurar' ) );
1006 }
1007 }
1008 }
1009 else
1010 {
1011     echo json_encode( array( "id_resultado" => "Error de
procedimiento" ) );
1012     ob_flush();
1013 }
1014 /**
1015 *
1016 * Actualiza el estatus de la tableta
1017 *
1018 * @param int $id_session Representa la session activa para la
peticion
1019 * @param json $id_tes_estado_tableta Representa el status que tomara la
tableta
1020 * @param int $version Representa la version de la apk
instalada en la tableta
1021 *
1022 * @return echo
1023 */
1024 public function
actualiza_estado_tableta($id_sesion,$id_tes_estado_tableta="" ,$version="" )
1025 {
1026     if ($id_sesion == $this-> session-> userdata('session')) // valida el token de
entrada es el token que solicito el servicio
1027     {
1028         // se obtiene el dispositivo por token
1029         $tableta = $this-> Tableta_model-> getByMac($this-> session-
> userdata('mac'));
1030         // se obtienen los usuarios asignados, el tipo de censo y la unidad m?dica
1031         if ($tableta-> usuarios_asignados == 1 && $tableta-> id_tipo_censo != null &&
1032             $tableta-> id_asu_um != null)
1033         {
1034             // si todo la operacion ok actualiza estado de tableta (id_tes_estado_tableta),
version (version) y la fecha (ultima_actualizacion)
1035             $this-> Enrolamiento_model-> update_status_tableta($this-> session-
> userdata('mac'),$id_tes_estado_tableta,$version,date("Y-m-d H:i:s" ));
1036             $this-> session-> unset_userdata('session');
1037             $this-> session-> unset_userdata('mac');
1038             $this-> session-> unset_userdata('fecha');
1039             $this-> session-> unset_userdata('id_version');
1040         }
1041     }
1042 }
1043 /**
1044 *
1045 * Genera los esquemas incompletos de las personas que correspondan a la unidad medica de
la tableta
1046 *
1047 * @param int $id_persona Representa la persona a la que se le
calculara su esquema
1048 * @param string $fecha Representa la fecha de nacimiento de
la persona
1049 * @param array $vacunas Representa las vacunas aplicadas a la
persona
1050 */

```

```

1051     * @return          echo
1052     */
1053     public function esquema_incompleto($id_persona,$fecha,$vacunas)
1054     //agregar dias a la fecha si periodo de colchon ver tabla tabla_agregar_bit de prioridad
1 ya le toca 0 periodo de ventana "prioridad"=>1 ? 0
1055     $cadena= array();
1056     $regla=$this-> ReglaVacuna_model-> getAll();
1057
1058     $fecha      = date("Y-m-d" ,strtotime($fecha));
1059     $datetime1 = date_create($fecha);
1060     $datetime2 = date_create(date("Y-m-d" ));
1061     $interval  = date_diff($datetime1, $datetime2);
1062     $dias       = $interval-> format('%a');
1063     $dias_extra= $dias+$this-> session-> userdata('dias_extras');
1064     $mas=0;
1065     if($dias> 365&& $dias< 1461)$mas=365;
1066     if($dias> 1461)$mas=1461;
1067
1068     foreach($regla as $r)
1069     {
1070         $x=0;
1071         if($vacunas!="")
1072         {
1073             foreach($vacunas as $v)
1074             {
1075                 if($r-> id==$v-> id_vacuna)
1076                 {
1077                     $x++;
1078                 }
1079             }
1080             if($x==0)
1081             {
1082                 if($r-> hasta> $mas)
1083                 {
1084                     if(($dias>=( $r-> desde)&& $dias<=( $r-
1085 > hasta))||$dias>($ r->
1086 > hasta)))
1087                     array_push($cadena,array("id_persona" =>
1088 $id_persona,"id_vacuna" => $r-> id, "prioridad" => 1));
1089                     if(($dias_extra>=( $r-> desde)&& $dias_extra<=( $r-
1090 > hasta)))
1091                     array_push($cadena,array("id_persona" =>
1092 $id_persona,"id_vacuna" => $r-> id, "prioridad" => 0));
1093             }
1094         }
1095     }
1096     * Genera los catalogos relevantes por entorno
1097     *
1098     * @param string $sf bandera que activa el filtro de fechas segun
1099     * el tipo de sincronizacion
1100     * @return echo
1101     */
1102     public function catalogos_relevantes($sf="")
1103     {
1104         $fechis="";
1105         if($sf!="")$fechis=$this-> session-> userdata('fecha');
1106         $catalog_relevante = $this-> Enrolamiento_model-> get_catalog_relevante($fechis);
1107         foreach($catalog_relevante as $catalog)
1108         {
1109             $array=$this-> Enrolamiento_model-> get_catalog($catalog-> descripcion);
1110             $cadena[$catalog-> descripcion]= $array;
1111             $micadena=json_encode($cadena);
1112             echo "," .substr($micadena,1,strlen($micadena)-2);
1113             $micadena="";
1114             ob_flush();
1115             unset($cadena);
1116             $cadena=array();
1117         }
1118     }
1119     public function prueba2($id_accion,$id_tab=null,$id_session=null, $version=null)
1120     {
1121         $this-> is_step_0(
1122             json_encode(array("id_accion" => $id_accion)),
1123             json_encode(array("id_tab" => $id_tab)),
1124             json_encode(array("id_sesion" => $id_session)));

```

```

1125      $version,
1126      ' {"cns_antiguo_domicilio": [{"fecha_cambio": "2014-03-13
11:10:13", "id_asu_localidad_domicilio": "255", "calle_domicilio": &quo
t;2A AVENIDA ORIENTE
NORTE", "colonia_domicilio": null, "numero_domicilio": "SN", "cp_do
micio": "0", "id_persona": "2963091b0b95e0ceae236e1da027cc1f", "re
ferencia_domicilio": null}, {"fecha_cambio": "2014-03-13
11:16:23", "id_asu_localidad_domicilio": "255", "calle_domicilio": &quo
t;2A AVENIDA ORIENTE NORTE", "colonia_domicilio": "Los
manzanos", "numero_domicilio": "SN", "cp_domicilio": "0", &quo
t;id_persona": "2963091b0b95e0ceae236e1da027cc1f", "referencia_domicilio": &quot
;"}, {"fecha_cambio": "2014-03-13
11:16:39", "id_asu_localidad_domicilio": "255", "calle_domicilio": &quo
t;2A AVENIDA ORIENTE NORTE", "colonia_domicilio": "Los
manzanos", "numero_domicilio": "SN", "cp_domicilio": "18500",
"id_persona": "2963091b0b95e0ceae236e1da027cc1f", "referencia_domicilio": &quot
;"}, {"cns_persona_x_alergia": [{"id_persona": "2963091b0b95e0ceae236e1
da027cc1f", "id_alergia": "12", "ultima_actualizacion": "2014-03-
13
11:39:46"}, {"id_persona": "2963091b0b95e0ceae236e1da027cc1f", "id_alergia&
quot;: "3", "ultima_actualizacion": "2014-03-13
11:59:20"}], "sis_bitacora": [{"parametros": "paciente:2963091b0b95e0ceae23
6e1da027cc1f", "fecha_hora": "2014-03-13
11:10:13", "id_usuario": "17", "id_controlador_accion": "98"}]
, {"parametros": "paciente:2963091b0b95e0ceae236e1da027cc1f", "fecha_hora": "2014-03-13
11:16:23", "id_usuario": "17", "id_controlador_accion": "98"}]
, {"parametros": "paciente:2963091b0b95e0ceae236e1da027cc1f", "fecha_hora": "2014-03-13
11:16:39", "id_usuario": "17", "id_controlador_accion": "98"}]
, {"parametros": "paciente:2963091b0b95e0ceae236e1da027cc1f", "fecha_hora": "2014-03-13
11:32:24", "id_usuario": "17", "id_controlador_accion": "97"}]
, {"parametros": "paciente:2963091b0b95e0ceae236e1da027cc1f,
alergia:12", "fecha_hora": "2014-03-13
11:39:46", "id_usuario": "17", "id_controlador_accion": "99"}]
, {"parametros": "paciente:2963091b0b95e0ceae236e1da027cc1f,
alergia:3", "fecha_hora": "2014-03-13
11:59:20", "id_usuario": "17", "id_controlador_accion": "99"}]
, {"cns_persona": [{"sector": "", "id_nacionalidad": "142&quot
;, "calle_domicilio": "2A AVENIDA ORIENTE
NORTE", "colonia_domicilio": "Los
manzanos", "numero_domicilio": "SN", "telefono_domicilio": null, "c
p_domicilio": "18500", "fecha_registro": "2014-02-27
00:00:00", "ageb": "0155", "sexo": "M", "manzana": &q
uot;, "id_asu_localidad_nacimiento": "255", "referencia_domicilio": &q
uot;, "id_asu_um_tratante": "20140", "fecha_nacimiento": "2009-
11-03", "nombre": "HECTOR
ADRIAN", "id": "2963091b0b95e0ceae236e1da027cc1f", "id_asu_localidad_domi
cio": "255", "id_tipo_sanguineo": "9", "apellido_materno": &qu
ot;VAZQUEZ", "apellido_paterno": "PEREZ", "id_operadora_cellular": null,
"ultima_actualizacion": "2014-03-13
11:32:24", "curp": "PEVH091103HCSRZC05", "celular": null}], "cns_an
tigua_um": [{"fecha_cambio": "2014-03-13
11:32:24", "id_persona": "2963091b0b95e0ceae236e1da027cc1f", "id_asu_um_tra
tante": "20736"}], "cns_control_nutricional": [{"talla": "0",
"peso": "6.5", "id_persona": "0c417380cc549e24e63e48aa0cacde50", &
quot;altura": "61", "fecha": "2301-12-15
00:00:00", "id_asu_um": "28434"}, {"talla": "0", "peso&qu
ot;: "11", "id_persona": "11b3b8babbefdb8f70e90e8e44clea0f", "altura&qu
ot;: "80", "fecha": "2014-08-15
00:00:00", "id_asu_um": "20736"}, {"talla": "0", "peso&qu
ot;: "15.4", "id_persona": "9c7e78cd824d683be01a73469b6ea4b5", "altura&
quot;: "107", "fecha": "2014-04-26
00:00:00", "id_asu_um": "28434"}], "cns_control_vacuna": [{"id_per
sona": "0624f2e47590750afb51d1b1100c616a", "codigo_barras": "", "fe
cha": "2017-02-27
00:00:00", "id_asu_um": "20736", "id_vacuna": "19"}, {"id_
persona": "0969d0271222ac05ec556ad3179784b0", "codigo_barras": "", &qu
ot;fecha": "2014-12-23
00:00:00", "id_asu_um": "21354", "id_vacuna": "9"}, {"id_
persona": "0ea2856502442ad4a666d56f77f7dd08", "codigo_barras": "", &quo
t;fecha": "2231-02-26
00:00:00", "id_asu_um": "20736", "id_vacuna": "17"}, {"id_
persona": "10349cdd07c2f9d1d8a83dea47f03785", "codigo_barras": "", &qu
ot;fecha": "2019-02-27
00:00:00", "id_asu_um": "20736", "id_vacuna": "21"}, {"id_
persona": "1a1cdb25408000fb52368d1b15b9a12", "codigo_barras": "", &qu
ot;fecha": "2014-04-03
00:00:00", "id_asu_um": "20736", "id_vacuna": "16"}, {"id

```

```
_persona": "5054211bc85a686c17ee79419b6f9daf", "codigo_barras": "", &qu
ot;fecha": "2062-02-26
00:00:00", "id_asu_um": "20736", "id_vacuna": "11"}, {"id
_persona": "8ec3a57acf175a8a9c83e6e3d175212f", "codigo_barras": "", &qu
ot;fecha": "2014-05-27
00:00:00", "id_asu_um": "28434", "id_vacuna": "21"}, {"id
_persona": "b812dc891c0215e7de3e546c7a9183ab", "codigo_barras": "", &qu
ot;fecha": "2014-05-27
00:00:00", "id_asu_um": "20736", "id_vacuna": "21"}, {"id
_persona": "f76ec7746905417f63d5d05bc8994ee4", "codigo_barras": "", &qu
ot;fecha": "2014-04-27
00:00:00", "id_asu_um": "20736", "id_vacuna": "21"}], "si
s_error": [{"descripcion": "NFC:null:\njava.io.IOException", "fecha_hora&qu
ot;": "2014-03-13
12:00:33", "id_usuario": "17", "id_controlador_accion": "1000"
;}]}' );
1127    }
1128 }
1129 ?>
```

# Archivo fuente para tableta.php

La documentación para este archivo está disponible en [tableta.php](#)

```
1      <?php
2
3      /**
4       * Controlador Tableta
5       *
6       * @package    TES
7       * @subpackage Controlador
8       * @author     Pascual
9       * @created    2013-11-26
10      */
11     class Tableta extends CI_Controller {
12
13         public function __construct()
14     {
15             parent::__construct();
16
17             if(!$this-> db-> conn_id) {
18                 $this-> template-> write('content', 'Error no se puede conectar a la Base de
Datos');
19                 $this-> template-> render();
20             }
21
22             $this-> load-> helper('url');
23             $this-> load-> model(DIR_TES.'/Tableta_model');
24         }
25
26         /**
27          * Lista todos los registros de tabletas, con su correspondiente paginación
28          * permite eliminar un conjunto de registro o un elemento individual,
29          * muestra enlaces para actualizar y ver detalles de un elemento específico
30          *
31          * @access public
32          * @param int    $pag Establece el desplazamiento del primer registro a devolver
33          * @return void
34         */
35         public function index($pag = 0)
36     {
37             if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url())) {
38                 show_error('', 403, 'Acceso denegado');
39                 return false;
40             }
41
42             if(!isset($this-> Tableta_model))
43                 return false;
44
45             try {
46                 $this-> load-> library('pagination');
47                 $this-> load-> helper(array('form', 'formatFecha'));
48                 $this-> load-> model(array(DIR_TES.'/Tipo_censo_model',
DIR_SIIGS.'/ArbolSegmentacion_model'));
49
50                 $data = array();
51
52                 $data['pag'] = $pag;
53                 $data['msgResult'] = $this-> session-> flashdata('msgResult');
54                 $data['clsResult'] = $this-> session-> flashdata('clsResult');
55                 $data['estados'] = (array)$this-> ArbolSegmentacion_model-> getDataKeyValue(1,
1);
56                 $data['jurisdicciones'] = array(0=> 'Seleccione una opción');
57                 $data['municipios'] = array(0=> 'Seleccione una opción');
58                 $data['localidades'] = array(0=> 'Seleccione una opción');
59                 $data['unidades'] = array(0=> 'Seleccione una opción');
60                 $data['title'] = 'Tableta';
61                 $data['tipos_censo'] = $this-> Tipo_censo_model-> getAll();
62                 $data['unidades_medicas'] = array();
63
64                 $registroEliminar = isset($_POST['registroEliminar']) ? $_POST['registroEliminar'] : null;
            }
```

```

: null;
65
66     if( !empty($registroEliminar) ) {
67         $this-> Tableta_model-> delete($registroEliminar);
68         $data['msgResult'] = 'Registros Eliminados exitosamente';
69         $data['clsResult'] = 'success';
70     }
71
72     // Configuración para el Paginador
73     $configPag['base_url'] = site_url().DIR_TES.'/tableta/index/';
74     $configPag['first_link'] = 'Primero';
75     $configPag['last_link'] = '&Uacute;ltimo';
76     $configPag['uri_segment'] = 4;
77     $configPag['total_rows'] = $this-> Tableta_model-> getNumRows();
78     $configPag['per_page'] = 20;
79
80     $this-> pagination-> initialize($configPag);
81
82     $filtro = array(
83         'edo' => $this-> input-> post('estados'),
84         'juris' => $this-> input-> post('juris'),
85         'muni' => $this-> input-> post('municipios'),
86         'locali' => $this-> input-> post('localidades'),
87         'um' => $this-> input-> post('ums'),
88     );
89
90     if(!empty($filtro['edo'])) {
91         $query = $this-> db-> query('SELECT id,descripcion FROM
asu_arbol_segmentacion WHERE id_raiz=1 AND id_padre='.$filtro['edo'].' ORDER BY descripcion');
92         $result = $query-> result();
93
94         if ($result){
95             foreach ($result as $temp) {
96                 $data['jurisdicciones'][$temp-> id] = $temp-> descripcion;
97             }
98         }
99     }
100
101    if(!empty($filtro['juris'])) {
102        $query = $this-> db-> query('SELECT id,descripcion FROM
asu_arbol_segmentacion WHERE id_raiz=1 AND id_padre='.$filtro['juris'].' ORDER BY descripcion');
103        $result = $query-> result();
104
105        if ($result){
106            foreach ($result as $temp) {
107                $data['municipios'][$temp-> id] = $temp-> descripcion;
108            }
109        }
110    }
111
112    if(!empty($filtro['muni'])) {
113        $query = $this-> db-> query('SELECT id,descripcion FROM
asu_arbol_segmentacion WHERE id_raiz=1 AND id_padre='.$filtro['muni'].' ORDER BY descripcion');
114        $result = $query-> result();
115
116        if ($result){
117            foreach ($result as $temp) {
118                $data['localidades'][$temp-> id] = $temp-> descripcion;
119            }
120        }
121    }
122
123    if(!empty($filtro['locali'])) {
124        $query = $this-> db-> query('SELECT id,descripcion FROM
asu_arbol_segmentacion WHERE id_raiz=1 AND id_padre='.$filtro['locali'].' ORDER BY descripcion');
125        $result = $query-> result();
126
127        if ($result){
128            foreach ($result as $temp) {
129                $data['unidades'][$temp-> id] = $temp-> descripcion;
130            }
131        }
132    }
133
134    $data['registros'] = $this-> Tableta_model-> getAll($configPag['per_page'],
$pag, $filtro);
135
136    // Obtener la descripción de cada unidad medica
137    foreach ($data['registros'] as $registro) {
138        if(!empty($registro-> id_asu_um)) {

```

```

139             $unidad_medica = $this-> ArbolSegmentacion_model-
> getDescripcionById(array($registro-> id_asu_um), 2);
140
141             if($unidad_medica) {
142                 $data['unidades_medicas'][$unidad_medica[0]-> id] = $unidad_medica[0]-
> descripcion;
143             }
144         }
145     }
146 } catch (Exception $e) {
147     $data['msgResult'] = Errorlog_model::save($this-> Tableta_model-
> getMsgError(), __METHOD__);
148     $data['clsResult'] = 'error';
149 }
150
151     $this-> template-> write('ajustaAncho', 1, true);
152     $this-> template-> write_view('content',DIR_TES.'/tableta/index', $data);
153     $this-> template-> render();
154 }
155
156 /**
157 * Muestra el formulario para crear un nuevo registro en la tableta,
158 * las variables se obtienen por el metodo POST
159 *
160 * @access public
161 * @return void
162 */
163 public function insert()
164 {
165     if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url())) {
166         show_error('', 403, 'Acceso denegado');
167         return false;
168     }
169
170     if(!isset($this-> Tableta_model))
171         return false;
172
173     try {
174         $this-> load-> helper('form');
175
176         $datos = $this-> input-> post();
177         $data['title'] = 'Crear un nuevo registro';
178         $data['msgResult'] = $this-> session-> flashdata('msgResult');
179         $data['clsResult'] = $this-> session-> flashdata('clsResult');
180
181         if(!empty($datos)) {
182             $this-> load-> library('form_validation');
183
184             $this-> form_validation-> set_rules('mac', 'MAC',
'trim|xss_clean|max_length[20]|required|callback_validateMac');
185
186             if ($this-> form_validation-> run() === true) {
187                 $this-> Tableta_model-> setMac($datos['mac']);
188
189                 $this-> Tableta_model-> insert();
190
191                 $this-> session-> set_flashdata('msgResult', 'Registro guardado
exitosamente');
192                 $this-> session-> set_flashdata('clsResult', 'success');
193
194                 Bitacora_model::insert(DIR_TES.'::'.__METHOD__, 'Registro creado: '. $this-
> Tableta_model-> getId());
195                 //redirect(DIR_TES.'/tableta/', 'refresh');
196                 //die();
197             } else {
198                 $this-> session-> set_flashdata('msgResult', validation_errors());
199                 $this-> session-> set_flashdata('clsResult', 'error');
200             }
201         }
202     } catch (Exception $e) {
203         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
204         $data['clsResult'] = 'error';
205     }
206
207 // $this->template->write_view('content',DIR_TES.'/tableta/insert', $data);
208 // $this->template->render();
209 redirect(DIR_TES.'/tableta/', 'refresh');
210 }
211
212 /**

```

```

213 * Muestra el formulario con los datos del registro especificado por el id,
214 * para actualizar sus datos
215 *
216 * @access public
217 * @param int $id ID del elemento a actualizar
218 * @return void
219 */
220 public function update($id)
221 {
222     if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url())) {
223         show_error('', 403, 'Acceso denegado');
224         return false;
225     }
226
227     if(!isset($this-> Tableta_model))
228         return false;
229
230     try {
231         $this-> load-> helper('form');
232
233         $datos = $this-> input-> post();
234         $data['title'] = 'Actualizar datos del registro';
235         $data['registro'] = $this-> Tableta_model-> getById($id);
236
237         // Cargar los datos a mostrar en el select de status
238         $this-> load-> model( DIR_TES.'/Estado_tableta_model' );
239         $estados_tableta = $this-> Estado_tableta_model-> getAll();
240
241         $data['status'][0] = 'Elegir';
242         foreach ($estados_tableta as $status) {
243             $data['status'][$status-> id] = $status-> descripcion;
244         }
245
246         if(!empty($datos)) {
247             $this-> load-> library('form_validation');
248
249             $this-> form_validation-> set_rules('mac', 'MAC',
250 'trim|xss_clean|max_length[25]|required');
251
252             if ($this-> form_validation-> run() === true) {
253                 $this-> Tableta_model-> setMac($datos['mac']);
254
255                 if(isset($datos['status']))
256                     $this-> Tableta_model-> setId_tes_estado_tableta($datos['status']);
257
258                 if(isset($datos['periodo_esq_inc']))
259                     $this-> Tableta_model-
260 > setPeriodo_esq_inc($datos['periodo_esq_inc']);
261
262                 $this-> Tableta_model-> update($id);
263
264                 Bitacora_model::insert(DIR_TES.'::'.__METHOD__, 'Registro actualizado:
265 '. $id);
266
267                 $this-> session-> set_flashdata('msgResult', 'Registro guardado
268 exitosamente');
269                 $this-> session-> set_flashdata('clsResult', 'success');
270                 redirect(DIR_TES.'/tableta/', 'refresh');
271                 die();
272             }
273
274         } catch (Exception $e) {
275             $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
276             $data['clsResult'] = 'error';
277         }
278
279         $this-> template-> write_view('content',DIR_TES.'/tableta/update', $data);
280         $this-> template-> render();
281
282     /**
283     * Muestra los datos del registro especificado por el id
284     *
285     * @access public
286     * @param int $id ID del elemento a actualizar
287     * @return void
288     */
289     public function view($id)
290     {
291         $usuarios = array();

```

```

289     if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url())) {
290         show_error('', 403, 'Acceso denegado');
291         return false;
292     }
293
294     if(!isset($this-> Tableta_model))
295         return false;
296
297     try {
298         $this-> load-> model(DIR_TES.'/Usuario_tableta_model');
299
300         $data['registro'] = $this-> Tableta_model-> getById($id);
301         $data['title'] = 'Datos del registro';
302
303         $this-> load-> helper('formatFecha');
304
305         if( empty($data['registro']) ) {
306             $data['msgResult'] = 'ERROR: El registro solicitado no existe';
307             $data['clsResult'] = 'error';
308         } else {
309             $usuariosTableta = $this-> Usuario_tableta_model-
> getUsuariosByTableta($id);
310
311             if(!empty($usuariosTableta)) {
312                 $this-> load-> model(DIR_TES.'/Usuario_model');
313
314                 foreach ($usuariosTableta as $usuario) {
315                     $infoUsuario = $this-> Usuario_model-> getById($usuario-
> id_usuario, true);
316                     $usuarios[] = $infoUsuario-> nombre_usuario;
317                 }
318             }
319         }
320
321         $data['usuarios'] = $usuarios;
322     } catch (Exception $e) {
323         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
324         $data['clsResult'] = 'error';
325     }
326
327     $this-> template-> write_view('content',DIR_TES.'/tableta/view', $data);
328     $this-> template-> write('menu','','true');
329     $this-> template-> write('sala_prensa','','true');
330     $this-> template-> render();
331 }
332
333 /**
334 * Eliminar el registro especificado por el id
335 *
336 * @access public
337 * @param int $id ID del elemento a eliminar
338 * @return void
339 */
340 public function delete($id)
341 {
342     if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url())) {
343         show_error('', 403, 'Acceso denegado');
344         return false;
345     }
346
347     if(!isset($this-> Tableta_model))
348         return false;
349
350     try {
351         $this-> Tableta_model-> delete($id);
352         $this-> session-> set_flashdata('msgResult', 'Registro eliminado exitosamente');
353         $this-> session-> set_flashdata('clsResult', 'success');
354     } catch (Exception $e) {
355         $this-> session-> set_flashdata('msgResult', Errorlog_model::save($this-
> Tableta_model-> getMsgError(), __METHOD__));
356         $this-> session-> set_flashdata('clsResult', 'error');
357     }
358
359     if (count($id) > 1)
360         Bitacora_model::insert(DIR_TES.'::'.__METHOD__, 'Registro eliminado:
'.implode(',', $id));
361     else
362         Bitacora_model::insert(DIR_TES.'::'.__METHOD__, 'Registro eliminado: '.$id);
363     redirect(DIR_TES.'/tableta/', 'refresh');
364     die();
}

```

```

365     }
366
367     /**
368      * Valida si existe una MAC en la base de datos
369      *
370      * @param type $mac
371      * @return boolean
372      */
373     public function _validateMac($mac)
374     {
375         $result = $this-> Tableta_model-> getByMac($mac);
376
377         if(!empty($result))
378         {
379             $this-> form_validation-> set_message('_validateMac', 'La direccion MAC ya esta registrada.');
380             return FALSE;
381         }
382         else
383         {
384             return TRUE;
385         }
386     }
387
388     /**
389      * Registra tabletas desde un archivo csv
390      *
391      * @return redirect
392      */
393     public function uploadFile()
394     {
395         if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url())) {
396             show_error('', 403, 'Acceso denegado');
397             return false;
398         }
399
400         $config['upload_path'] = 'application/uploads';
401         $config['allowed_types'] = 'csv|txt|xls|xlsx';
402         $config['max_size'] = '5120'; //5MB
403         $config['overwrite'] = true;
404         $errores = array();
405         $msjErrores = '';
406
407         $this-> load-> library('upload', $config);
408
409         try {
410             if ( !$this-> upload-> do_upload('archivo') ) {
411                 $this-> session-> set_flashdata('msgResult', $this-> upload-> display_errors());
412                 $this-> session-> set_flashdata('clsResult', 'error');
413             } else {
414                 $archivo = $this-> upload-> data();
415                 $fichero = fopen($archivo['full_path'], "r");
416
417                 if($fichero) {
418                     while(($linea = fgets($fichero)) !== false) {
419                         $linea = trim($linea);
420                         $result = $this-> Tableta_model-> getByMac($linea);
421
422                         if(!empty($result)) {
423                             $errores[] = $linea;
424                         } else {
425                             $this-> Tableta_model-> setMac($linea);
426                             $this-> Tableta_model-> insert();
427                         }
428
429                         if(!feof($fichero)) {
430                             $this-> session-> set_flashdata('msgResult', 'Error al leer el archivo '. $archivo['file_name']);
431                             $this-> session-> set_flashdata('clsResult', 'error');
432                         } else {
433                             if(!empty($errores)) {
434                                 $msjErrores = 'Las siguientes direcciones MAC ya estan registradas en el sistema: '. implode(', ', $errores).';
435                             }
436                         }
437
438                         $this-> session-> set_flashdata('msgResult', 'Datos registrados correctamente. '.$msjErrores);
439                         $this-> session-> set_flashdata('clsResult', 'success');
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
999

```

```

440             }
441             fclose($fichero);
442         } else {
443             $this-> session-> set_flashdata('msgResult', 'Error al leer el archivo
444             .'$archivo['file_name']);
445         }
446     } catch (Exception $e) {
447         $this-> session-> set_flashdata('msgResult', Errorlog_model::save($e-
448 > getMessage(), __METHOD__));
449         $this-> session-> set_flashdata('clsResult', 'error');
450     }
451
452     redirect(DIR_TES.'/tableta/', 'refresh');
453 }
454
455 /**
456 * Asignar unidad medica y tipo de censo
457 *
458 * @param int $id
459 * @return redirect
460 */
461 public function setUM($id)
462 {
463     if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url())) {
464         show_error('', 403, 'Acceso denegado');
465         return false;
466     }
467
468     if(!isset($this-> Tableta_model))
469         return false;
470
471     try {
472         $datos = $this-> input-> post();
473         $this-> Tableta_model-> getById($id);
474
475         if(!empty($datos)) {
476             $this-> Tableta_model-> setId_tipo_censo($datos['id_tipo_censo']);
477             $this-> Tableta_model-> setId_asu_um($datos['id_unidad_medica']);
478             $this-> Tableta_model-> setPeriodo_esq_inc($datos['periodo_esq_inc']);
479
480             $this-> Tableta_model-> update($id);
481
482             Bitacora_model::insert(DIR_TES.'::'.__METHOD__, 'Registro actualizado: '.$id);
483             $this-> session-> set_flashdata('msgResult', 'Registro actualizado
484 exitosamente');
485             $this-> session-> set_flashdata('clsResult', 'success');
486         }
487     } catch (Exception $e) {
488         $this-> session-> set_flashdata('msgResult', Errorlog_model::save($e-
489 > getMessage(), __METHOD__));
490         $this-> session-> set_flashdata('clsResult', 'error');
491     }
492     redirect(DIR_TES.'/tableta/', 'refresh');
493 }
494 ?>

```

# Archivo fuente para usuario\_tableta.php

La documentación para este archivo está disponible en [usuario\\_tableta.php](#)

```
1      <?php
2
3  /**
4   * Controlador Usuario_tableta
5   *
6   * @package    TES
7   * @subpackage Controlador
8   * @author     Pascual
9   * @created    2013-11-27
10  */
11 class Usuario_tableta extends CI_Controller {
12
13     public function __construct()
14     {
15         parent::__construct();
16
17         if(!$this-> db-> conn_id) {
18             $this-> template-> write('content', 'Error no se puede conectar a la Base de
19 Datos');
20             $this-> template-> render();
21
22         $this-> load-> helper('url');
23         $this-> load-> model(DIR_TES.'/Tableta_model');
24         $this-> load-> model(DIR_TES.'/Usuario_tableta_model');
25     }
26
27     /**
28      * Lista todos los registros de usuarios correspondientes a una tableta en específico
29      * permite eliminar un conjunto de registro o un elemento individual,
30      * muestra enlaces para actualizar y ver detalles de un elemento específico
31      *
32      * @access public
33      * @param int    $idTableta ID de la Tableta
34      * @return void
35      */
36     public function index($idTableta)
37     {
38         if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url())) {
39             show_error('', 403, 'Acceso denegado');
40             return false;
41         }
42
43         if(!isset($this-> Tableta_model))
44             return false;
45
46         if(!isset($this-> Usuario_tableta_model))
47             return false;
48
49         try {
50             $this-> load-> helper(array('form', 'formatFecha'));
51             $this-> load-> model(DIR_SIIGS.'/Grupo_model');

52             $data = array();

53
54             $data['msgResult'] = $this-> session-> flashdata('msgResult');
55             $data['clsResult'] = $this-> session-> flashdata('clsResult');
56             $data['title'] = 'Usuarios asignados a la tableta';
57             $data['tableta'] = $this-> Tableta_model-> getById($idTableta);
58             $data['grupos'] = $this-> Grupo_model-> getAll();
59             $data['usuarios'] = array();

60             $registroEliminar = $this-> input-> post('registroEliminar');

61
62             if( !empty($registroEliminar) ) {
63                 $this-> Usuario_tableta_model-> delete($registroEliminar, $idTableta);
64                 $data['msgResult'] = 'Registros Eliminados exitosamente';
65             }
66         }
```

```

67             $data['clsResult'] = 'success';
68         }
69
70         $usuariosTableta = $this-> Usuario_tableta_model-
> getUsuariosByTableta($idTableta);
71
72         if(!empty($usuariosTableta)) {
73             $this-> load-> model(DIR_TES.'/Usuario_model');
74
75             foreach ($usuariosTableta as $usuario) {
76                 $infoUsuario = $this-> Usuario_model-> getById($usuario-> id_usuario,
true);
77                 $data['usuarios'][$usuario-> id_usuario] = array(
78                     'id' => $usuario-> id_usuario,
79                     'usuario' => $infoUsuario-> nombre_usuario,
80                     'nombre' => $infoUsuario-> nombre.' '.$infoUsuario-
> apellido_paterno.' '.$infoUsuario-> apellido_materno,
81                     'grupo' => $infoUsuario-> Grupo
82                 );
83             }
84         }
85
86     } catch (Exception $e) {
87         $data['msgResult'] = Errorlog_model::save($e-> getMessage(), __METHOD__);
88         $data['clsResult'] = 'success';
89     }
90
91     $this-> template-> write_view('content',DIR_TES.'/usuario_tableta/index', $data);
92     $this-> template-> render();
93 }
94
95 /**
96 * Muestra el formulario para crear un nuevo registro en la tableta,
97 * las variables se obtienen por el metodo POST
98 *
99 * @access public
100 * @return void
101 */
102 public function insert($id_tableta)
103 {
104     if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url())) {
105         show_error('', 403, 'Acceso denegado');
106         return false;
107     }
108
109     if(!isset($this-> Usuario_tableta_model))
110         return false;
111
112     try {
113         $id_usuario = $this-> input-> post('id_usuario');
114
115         if(!empty($id_usuario)) {
116             $this-> Usuario_tableta_model-> insert($id_usuario, $id_tableta);
117
118             $this-> session-> set_flashdata('msgResult', 'Registro guardado
exitosamente');
119             $this-> session-> set_flashdata('clsResult', 'success');
120
121             Bitacora_model::insert(DIR_TES.'::'.__METHOD__, 'Registro creado: Tableta =
'. $id_tableta.' - Usuario = '.$id_usuario);
122
123         } else {
124             $this-> session-> set_flashdata('msgResult', 'Error: debe proporcionar un
usuario');
125             $this-> session-> set_flashdata('clsResult', 'error');
126         }
127     } catch (Exception $e) {
128         $this-> session-> set_flashdata('msgResult', 'Error: Debe proporcionar un
usuario valido');
129         $this-> session-> set_flashdata('clsResult', 'error');
130     }
131
132     redirect(DIR_TES.'/usuario_tableta/index/'.$id_tableta, 'refresh');
133 }
134
135 /**
136 * Eliminar un usuario de una tableta especifica
137 *
138 * @access public
139 * @param int    $$id_usuario ID del usuario a eliminar

```

```

140     * @param int    $id_tableta  ID de la tableta que tiene asignada el usuario especificado
141     * @return void
142     */
143     public function delete($id_usuario, $id_tableta)
144     {
145         if (!Usuario_model::checkCredentials(DIR_TES.'::'.__METHOD__, current_url())) {
146             show_error('', 403, 'Acceso denegado');
147             return false;
148         }
149
150         if(!isset($this-> Tableta_model))
151             return false;
152
153         if(!isset($this-> Usuario_tableta_model))
154             return false;
155
156         try {
157             $this-> Usuario_tableta_model-> delete($id_usuario, $id_tableta);
158             $this-> session-> set_flashdata('msgResult', 'Registro eliminado exitosamente');
159             $this-> session-> set_flashdata('clsResult', 'success');
160         } catch (Exception $e) {
161             $this-> session-> set_flashdata('msgResult', Errorlog_model::save($e-
> getMessage(), __METHOD__));
162             $this-> session-> set_flashdata('clsResult', 'error');
163         }
164
165         Bitacora_model::insert(DIR_TES.'::'.__METHOD__, 'Registro eliminado: '.$id_usuario);
166
167         redirect(DIR_TES.'/usuario_tableta/index/'.$id_tableta, 'refresh');
168         die();
169     }
170
171 }
?>
```

# Archivo fuente para enrolamiento\_model.php

*La documentación para este archivo está disponible en [enrolamiento\\_model.php](#)*

```
1  <?php
2  /**
3   * Modelo Usuario
4   *
5   * @package      TES
6   * @subpackage   Modelo
7   * @author       Eliecer
8   * @created      2013-12-17
9   */
10  class Enrolamiento_model extends CI_Model
11  {
12      /**
13      * Guarda la instancia del objeto global CodeIgniter
14      * para utilizarlo en la función estática
15      *
16      * @access private
17      * @var    instance
18      */
19      private static $CI;
20
21      /**
22      * @variables
23      */
24      // Basico
25      private $id;
26      private $nacionalidad;
27      private $nombre;
28      private $paterno;
29      private $materno;
30      private $lnacimiento;
31      private $scurnp;
32      private $sexo;
33      private $sangre;
34      private $fnacimiento;
35      private $tbeneficiario;
36      private $parto;
37      private $stamiz_neonatal;
38      private $precurp;
39
40      // civil
41      private $fechacivil;
42      private $lugarcivil;
43      private $umt;
44
45      // direccion
46      private $calle;
47      private $referencia;
48      private $colonia;
49      private $localidad;
50      private $numero;
51      private $cp;
52      private $telefono;
53      private $compania;
54      private $celular;
55      private $ageb;
56      private $sector;
57      private $manzana;
58
59      // Tipo de beneficiario
60      private $afiliacion= array();
61
62      // historial de alergias
63      private $alergias= array();
64
```

```

65 // vacunacion
66 private $vacuna= array();
67 private $fvacuna= array();
68 private $codigo_barras= array();
69
70 // IRA
71 private $ira= array();
72 private $fira= array();
73 private $tira= array();
74
75 // EDA
76 private $eda= array();
77 private $fedaa= array();
78 private $tedaa= array();
79
80 // Consulta
81 private $consulta= array();
82 private $fcconsulta= array();
83 private $tconsulta= array();
84
85 // Accion nutricional
86 private $accion_nutricional= array();
87 private $faccion_nutricional= array();
88
89 // nutricion
90 private $peso= array();
91 private $altura= array();
92 private $talla= array();
93 private $hemoglobina= array();
94 private $fnutricion= array();
95 private $peri_cefa= array();
96 private $fecha_peri_cefa= array();
97
98 private $estimulacion_fecha= array();
99 private $estimulacion_capacitado= array();
100
101 private $sales_fecha= array();
102 private $sales_cantidad= array();
103
104 /*****
105 * Estas variables no pertenecen a la tabla *
106 *****/
107
108 /**
109 * @access private
110 * @var string
111 */
112 private $msg_error_usr;
113 /**
114 * @access private
115 * @var string
116 */
117 private $msg_error_log;
118 public function __construct()
119 {
120     parent::__construct();
121
122     self::$CI = & get_instance();
123
124     $this-> load-> database();
125     if (!$this-> db-> conn_id)
126         throw new Exception("No se pudo conectar a la base de datos");
127 }
128
129 public function getId()
130 {
131     return $this-> id;
132 }
133
134 public function setId($value)
135 {
136     $this-> id = $value;
137 }
138
139 public function getnombre()
140 {
141     return $this-> nombre;
142 }
143
144 public function setnombre($value)

```

```

145     {
146         $this-> nombre = $value;
147     }
148
149     public function getpaterno()
150     {
151         return $this-> paterno;
152     }
153
154     public function setpaterno($value)
155     {
156         $this-> paterno = $value;
157     }
158
159     public function getmaterno()
160     {
161         return $this-> materno;
162     }
163
164     public function setmaterno($value)
165     {
166         $this-> materno = $value;
167     }
168
169     public function getlnacimiento()
170     {
171         return $this-> lnacimiento;
172     }
173
174     public function setlnacimiento($value)
175     {
176         $this-> lnacimiento = $value;
177     }
178
179     public function getcurp()
180     {
181         return $this-> curp;
182     }
183
184     public function setcurp($value)
185     {
186         $this-> curp = $value;
187     }
188
189     public function getsexo()
190     {
191         return $this-> sexo;
192     }
193
194     public function setsexo($value)
195     {
196         $this-> sexo = $value;
197     }
198
199     public function getsangre()
200     {
201         return $this-> sangre;
202     }
203
204     public function setsangre($value)
205     {
206         $this-> sangre = $value;
207     }
208
209     public function getfnacimiento()
210     {
211         return $this-> fnacimiento;
212     }
213
214     public function setfnacimiento($value)
215     {
216         $this-> fnacimiento = $value;
217     }
218
219     public function gettbeneficiario()
220     {
221         return $this-> tbeneficiario;
222     }
223
224     public function settbeneficiario($value)

```

```

225     {
226         $this-> tbeneficiario = $value;
227     }
228
229     public function getparto()
230     {
231         return $this-> parto;
232     }
233
234     public function gettamiz()
235     {
236         return $this-> tamiz_neonatal;
237     }
238
239     public function getprecurp()
240     {
241         return $this-> precurp;
242     }
243
244     public function setprecurp($value)
245     {
246         $this-> precurp = $value;
247     }
248
249     public function setparto($value)
250     {
251         $this-> parto = $value;
252     }
253
254     public function settamiz($value)
255     {
256         $this-> tamiz_neonatal = $value;
257     }
258 //tutor
259     public function getidtutor()
260     {
261         return $this-> idtutor;
262     }
263
264     public function setidtutor($value)
265     {
266         $this-> idtutor = $value;
267     }
268
269     public function getnombreT()
270     {
271         return $this-> nombreT;
272     }
273
274     public function setnombreT($value)
275     {
276         $this-> nombreT = $value;
277     }
278
279     public function getpaternoT()
280     {
281         return $this-> paternoT;
282     }
283
284     public function setpaternoT($value)
285     {
286         $this-> paternoT = $value;
287     }
288
289     public function getmaternoT()
290     {
291         return $this-> maternoT;
292     }
293     public function setmaternoT($value)
294     {
295         $this-> maternoT = $value;
296     }
297
298     public function getcurpT()
299     {
300         return $this-> curpT;
301     }
302
303     public function setcurpT($value)
304     {

```

```

305     $this-> curpT = $value;
306 }
307
308 public function getsexoT()
309 {
310     return $this-> sexoT;
311 }
312
313 public function setsexoT($value)
314 {
315     $this-> sexoT = $value;
316 }
317 public function gettelefonoT()
318 {
319     return $this-> telefonoT;
320 }
321
322 public function settelefonoT($value)
323 {
324     $this-> telefonoT = $value;
325 }
326
327 public function getcompaniaT()
328 {
329     return $this-> companiaT;
330 }
331
332 public function setcompaniaT($value)
333 {
334     $this-> companiaT = $value;
335 }
336
337 public function getcelularT()
338 {
339     return $this-> celularT;
340 }
341
342 public function setcelularT($value)
343 {
344     $this-> celularT = $value;
345 }
346 /**
347 public function getfechacivil()
348 {
349     return $this-> fechacivil;
350 }
351
352 public function setfechacivil($value)
353 {
354     $this-> fechacivil = $value;
355 }
356
357 public function getlugarcivil()
358 {
359     return $this-> lugarcivil;
360 }
361
362 public function setlugarcivil($value)
363 {
364     $this-> lugarcivil = $value;
365 }
366
367 public function getumt()
368 {
369     return $this-> umt;
370 }
371
372 public function setumt($value)
373 {
374     $this-> umt = $value;
375 }
376
377 public function getreferencia()
378 {
379     return $this-> referencia;
380 }
381
382 public function setreferencia($value)
383 {
384     $this-> referencia = $value;

```

```

385 }
386
387 public function getcalle()
388 {
389     return $this-> calle;
390 }
391
392 public function setcalle($value)
393 {
394     $this-> calle = $value;
395 }
396
397 public function getcolonia()
398 {
399     return $this-> colonia;
400 }
401
402 public function setcolonia($value)
403 {
404     $this-> colonia = $value;
405 }
406
407 public function getlocalidad()
408 {
409     return $this-> localidad;
410 }
411
412 public function setlocalidad($value)
413 {
414     $this-> localidad = $value;
415 }
416
417 public function getnumero()
418 {
419     return $this-> numero;
420 }
421
422 public function setnumero($value)
423 {
424     $this-> numero = $value;
425 }
426
427 public function getcp()
428 {
429     return $this-> cp;
430 }
431
432 public function setcp($value)
433 {
434     $this-> cp = $value;
435 }
436
437 public function getageb()
438 {
439     return $this-> ageb;
440 }
441
442 public function setageb($value)
443 {
444     $this-> ageb = $value;
445 }
446
447 public function getsector()
448 {
449     return $this-> sector;
450 }
451
452 public function setsector($value)
453 {
454     $this-> sector = $value;
455 }
456
457 public function getmanzana()
458 {
459     return $this-> manzana;
460 }
461
462 public function setmanzana($value)
463 {
464     $this-> manzana = $value;

```

```

465 }
466 //*****
467
468 public function getafiliacion()
469 {
470     return $this-> afiliacion;
471 }
472
473 public function setafiliacion($value)
474 {
475     $this-> afiliacion = $value;
476 }
477
478 public function getalergias()
479 {
480     return $this-> alergias;
481 }
482
483 public function setalergias($value)
484 {
485     $this-> alergias = $value;
486 }
487
488 public function getvacuna()
489 {
490     return $this-> vacuna;
491 }
492
493 public function setvacuna($value)
494 {
495     $this-> vacuna = $value;
496 }
497
498 public function getfvacuna()
499 {
500     return $this-> fvacuna;
501 }
502
503 public function setfvacuna($value)
504 {
505     $this-> fvacuna = $value;
506 }
507
508 public function getcodigo_barras()
509 {
510     return $this-> codigo_barras;
511 }
512
513 public function setcodigo_barras($value)
514 {
515     $this-> codigo_barras = $value;
516 }
517
518 public function getconsulta()
519 {
520     return $this-> consulta;
521 }
522
523 public function setconsulta($value)
524 {
525     $this-> consulta = $value;
526 }
527
528 public function getfconsulta()
529 {
530     return $this-> fconsulta;
531 }
532
533 public function setfconsulta($value)
534 {
535     $this-> fconsulta = $value;
536 }
537
538 public function gettconsulta()
539 {
540     return $this-> tconsulta;
541 }
542
543 public function settconsulta($value)
544 {

```

```

545     $this-> tconsulta = $value;
546 }
547
548 public function getaccion_nutricional()
549 {
550     return $this-> accion_nutricional;
551 }
552
553 public function setaccion_nutricional($value)
554 {
555     $this-> accion_nutricional = $value;
556 }
557
558 public function getfaccion_nutricional()
559 {
560     return $this-> faccion_nutricional;
561 }
562
563 public function setfaccion_nutricional($value)
564 {
565     $this-> faccion_nutricional = $value;
566 }
567
568 public function getpeso()
569 {
570     return $this-> peso;
571 }
572
573 public function setpeso($value)
574 {
575     $this-> peso = $value;
576 }
577
578 public function getaltura()
579 {
580     return $this-> altura;
581 }
582
583 public function setaltura($value)
584 {
585     $this-> altura = $value;
586 }
587 public function gettalla()
588 {
589     return $this-> talla;
590 }
591
592 public function settalla($value)
593 {
594     $this-> talla = $value;
595 }
596
597 public function gethemoglobina()
598 {
599     return $this-> hemoglobina;
600 }
601
602 public function sethemoglobina($value)
603 {
604     $this-> hemoglobina = $value;
605 }
606
607 public function getfnutricion()
608 {
609     return $this-> fnutricion;
610 }
611
612 public function setfnutricion($value)
613 {
614     $this-> fnutricion = $value;
615 }
616
617 public function gettelefono()
618 {
619     return $this-> telefono;
620 }
621
622 public function settelefono($value)
623 {
624     $this-> telefono = $value;

```

```

625 }
626
627 public function getcompania()
628 {
629     return $this-> compania;
630 }
631
632 public function setcompania($value)
633 {
634     $this-> compania = $value;
635 }
636
637 public function getcelular()
638 {
639     return $this-> celular;
640 }
641
642 public function setcelular($value)
643 {
644     $this-> celular = $value;
645 }
646
647 public function getnacionalidad()
648 {
649     return $this-> nacionalidad;
650 }
651
652 public function setnacionalidad($value)
653 {
654     $this-> nacionalidad = $value;
655 }
656
657 public function getperi_cefa()
658 {
659     return $this-> peri_cefa;
660 }
661
662 public function setperi_cefa($value)
663 {
664     $this-> peri_cefa = $value;
665 }
666
667 public function getfecha_peri_cefa()
668 {
669     return $this-> fecha_peri_cefa;
670 }
671
672 public function setfecha_peri_cefa($value)
673 {
674     $this-> fecha_peri_cefa = $value;
675 }
676
677 public function getestimulacion_fecha()
678 {
679     return $this-> estimulacion_fecha;
680 }
681
682 public function setestimulacion_fecha($value)
683 {
684     $this-> estimulacion_fecha = $value;
685 }
686
687 public function getestimulacion_capacitado()
688 {
689     return $this-> estimulacion_capacitado;
690 }
691
692 public function setestimulacion_capacitado($value)
693 {
694     $this-> estimulacion_capacitado = $value;
695 }
696
697 public function getsales_cantidad()
698 {
699     return $this-> sales_cantidad;
700 }
701
702 public function setsales_cantidad($value)
703 {
704     $this-> sales_cantidad = $value;

```

```

705     }
706
707     public function getsales_fecha()
708     {
709         return $this-> sales_fecha;
710     }
711
712     public function setsales_fecha($value)
713     {
714         $this-> sales_fecha = $value;
715     }
716
717     /**
718     *
719     * Hace insert de las tablas cns_control_x que se reciben en la sincronizacion secuencial
720     *
721     * @param      string      $stabla      Nombre de la tabla a la que se afectara
722     * @param      array       $array       Datos que se guardaran en la tabla
723     *
724     * @return     result()
725     */
726
727     public function cns_insert($stabla,$array)
728     {
729         $result = $this-> db-> insert($stabla, $array);
730
731         if (!$result)
732         {
733             $this-> msg_error_usr = "Error $stabla." ;
734             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
735         }
736     }
737
738     /**
739     *
740     * Actualiza la tabla especificada
741     *
742     * @param      string      $stabla      Nombre de la tabla afectada
743     * @param      array       $array       Datos a actualizar
744     * @param      string      $id        identificador para el where
745     * @param      string      $campo     campo si se necesitaran un segundo where
746     * @param      string      $valor     valor del campo a comparar
747     *
748     * @return     result()
749     */
750
751     public function cns_update($stabla,$array,$id,$campo="" ,
$valor="" , $campo2="" , $valor2="")
752     {
753         if($id!="")
754             $this-> db-> where('id' , $id);
755         if($campo!="")
756             $this-> db-> where($campo , $valor);
757         if($campo2!="")
758             $this-> db-> where($campo2 , $valor2);
759         $result = $this-> db-> update($stabla, $array);
760
761         if (!$result)
762         {
763             $this-> msg_error_usr = "Error $stabla." ;
764             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
765         }
766     }
767     public function cns_update_visita($id)
768     {
769         $this-> db-> set('contador_visitas' , 'contador_visitas+1',false);
770         $this-> db-> where('id' , $id);
771         $result = $this-> db-> update("cns_persona" );
772
773         if (!$result)
774         {
775             $this-> msg_error_usr = "Error contador_visitas." ;
776             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
777         }
778     }
779
780     /**

```

```

781 *
782 * Guarda la persona capturada mediante el formulario web
783 *
784 * @return          result()
785 *
786 */
787 public function insert()
788 {
789     $unico_id=md5(uniqid());
790     $compania=$this-> compania;
791     if($compania=="") $compania=NULL;
792     $data = array(
793         // basico
794         'id' => $unico_id,
795         'id_nacionalidad' => $this-> nacionalidad,
796         'nombre' => $this-> nombre,
797         'apellido_paterno' => $this-> paterno,
798         'apellido_materno' => $this-> materno,
799         'id_asu_localidad_nacimiento' => $this-> lnacimiento,
800         'curp' => $this-> curp,
801         'sexo' => $this-> sexo,
802         'id_tipo_sanguineo' => $this-> sangre,
803         'fecha_nacimiento' => date('Y-m-d H:i:s', strtotime($this-> fnacimiento)),
804         'id_parto_multiple' => $this-> parte,
805         'tamiz_neonatal' => $this-> tamiz_neonatal,
806         'precurp' => $this-> precurp,
807
808         // civil
809         'fecha_registro' => date('Y-m-d H:i:s', strtotime($this-> fechacivil)),
810         'id_asu_um_tratante' => $this-> umt,
811
812         // direccion
813         'calle_domicilio' => $this-> calle,
814         'referencia_domicilio' => $this-> referencia,
815         'colonia_domicilio' => $this-> colonia,
816         'id_asu_localidad_domicilio' => $this-> localidad,
817         'numero_domicilio' => $this-> numero,
818         'cp_domicilio' => $this-> cp,
819         'ageb' => $this-> ageb,
820         'sector' => $this-> sector,
821         'manzana' => $this-> manzana,
822
823         'telefono_domicilio' => $this-> telefono,
824         'id_operadora_celular' => $compania,
825         'celular' => $this-> celular,
826         'ultima_actualizacion' => date("Y-m-d H:i:s")
827     );
828     $result = $this-> db-> insert('cns_persona', $data);
829     if (!$result)
830     {
831         $this-> msg_error_usr = "Enrolamiento Fallido." ;
832         $this-> msg_error_log = "("
> db-> _error_number()." : '$this-> db-> _error_message();"
833         throw new Exception("(" . __METHOD__ . ") => "
> _error_number()." : '$this-> db-> _error_message();"
834     }
835     else
836     {
837         $this-> setid($unico_id);
838         $dat = array(
839             'id_persona' => $this-> id,
840             'fecha_registro' => date('Y-m-d H:i:s', strtotime($this-> fechacivil)),
841             'id_localidad_registro_civil' => $this-> lugarcivil,
842         );
843         $res = $this-> db-> insert('cns_registro_civil', $dat);
844
845         $companiaT=$this-> companiat;
846         if($companiaT=="") $companiaT=NULL;
847
848         $unico_idtutor=md5(uniqid());
849         $data0 = array(
850             // tutor
851             //'id' => $unico_idtutor,
852             'nombre' => $this-> nombreT,
853             'apellido_paterno' => $this-> paternoT,
854             'apellido_materno' => $this-> maternoT,
855             'curp' => $this-> curpT,
856             'sexo' => $this-> sexoT,
857
858             'telefono' => $this-> telefonoT,

```

```

859             'id_operadora_celular' =>      $companiaT,
860             'celular' =>      $this->    celularT,
861             'ultima_actualizacion' =>      date("Y-m-d H:i:s")           )
862         );
863         if($this->    idtutor=="")
864         {
865             // Se le asigna ID al tutor en caso de que sea una nueva captura
866             $data0['id'] = $unico_idtutor;
867             $companiaT=$this->    companiaT;
868             if($companiaT=="")          $companiaT=NULL;
869
870             $result0 = $this->    db->    insert('cns_tutor', $data0);
871             if (! $result0)
872             {
873                 $this->    msg_error_usr = "No se guardo Tutor." ;
874                 $this->    msg_error_log = "(" . __METHOD__ . ") => "
875             . $this->    db->    _error_number().': ' . $this->    db->    _error_message();
876             > db->    _error_number().': ' . $this->    db->    _error_message());
877             }
878             else
879             {
880                 $this->    setidtutor($unico_idtutor);
881             }
882         }
883         else
884         {
885             $this->    db->    where('id' , $this->    idtutor);
886             $result0 = $this->    db->    update('cns_tutor', $data0);
887             if (! $result0)
888             {
889                 $this->    msg_error_usr = "No se actualizo Tutor." ;
890                 $this->    msg_error_log = "(" . __METHOD__ . ") => "
891             . $this->    db->    _error_number().': ' . $this->    db->    _error_message();
892             > db->    _error_number().': ' . $this->    db->    _error_message());
893             }
894             $data01 = array(
895                 'id_persona' =>      $this->    id,
896                 'id_tutor' =>      $this->    idtutor,
897                 'ultima_actualizacion' =>      date('Y-m-d H:i:s'),
898             );
899             $result01 = $this->    db->    insert('cns_persona_x_tutor', $data01);
900             if (! $result01)
901             {
902                 $this->    msg_error_usr = "No se relaciono Tutor." ;
903                 $this->    msg_error_log = "(" . __METHOD__ . ") => "
904             . $this->    db->    _error_number().': ' . $this->    db->    _error_message();
905             > db->    _error_number().': ' . $this->    db->    _error_message());
906             }
907             $id_asu_um=$this->    umt;
908             for($i=0;$i<    sizeof($this->    alergias);$i++)
909             {
910                 $data1 = array(
911                     // alergias
912                     'id_persona' =>      $this->    id,
913                     'id_alergia' =>      $this->    alergias[$i],
914                     'ultima_actualizacion' =>      date('Y-m-d H:i:s'),
915                 );
916                 if($this->    alergias[$i]!="")
917                 {
918                     $result1 = $this->    db->    insert('cns_persona_x_alergia', $data1);
919                     if (! $result1)
920                     {
921                         $this->    msg_error_usr = "Error Alergias." ;
922                         $this->    msg_error_log = "(" . __METHOD__ . ") => "
923                     . $this->    db->    _error_number().': ' . $this->    db->    _error_message();
924                     > $this->    db->    _error_number().': ' . $this->    db->    _error_message());
925                     }
926                 }
927             for($i=0;$i<    sizeof($this->    vacuna);$i++)
928             {
929                 $data2 = array(

```

```

931             // vacuna
932             'id_persona' => $this-> id,
933             'id_vacuna' => $this-> vacuna[$i],
934             'fecha' => date('Y-m-d H:i:s', strtotime($this-> fvacuna[$i])),
935             'id_asu_um' => $id_asu_um,
936             'codigo_barra' => $this-> codigo_barra[$i],
937         );
938         if($this-> fvacuna[$i]!="")
939     {
940             $result2 = $this-> db-> insert('cns_control_vacuna', $data2);
941             if (!$result2)
942             {
943                 $this-> msg_error_usr = "Error vacunas." ;
944                 $this-> msg_error_log = "(" . __METHOD__ . ")";
945             }
946             . $this-> db-> _error_number().':'. $this-> db-> _error_message();
947             throw new Exception("(" . __METHOD__ . ")");
948         }
949     }
950     for($i=0;$i< sizeof($this-> consulta);$i++)
951     {
952         $data5 = array(
953             // consulta
954             'id_persona' => $this-> id,
955             'clave_cielo' => $this-> consulta[$i],
956             'fecha' => date('Y-m-d H:i:s', strtotime($this-> fconsulta[$i])),
957             'id_asu_um' => $id_asu_um,
958             'id_tratamiento' => $this-> tconsulta[$i],
959             'grupo_fecha_secuencial' => date('Y-m-d H:i:s', strtotime($this-
> fconsulta[$i])),
960         );
961         if($this-> consulta[$i]!="")
962     {
963             $result5 = $this-> db-> insert('cns_control_consulta', $data5);
964             if (!$result5)
965             {
966                 $this-> msg_error_usr = "Error Consulta." ;
967                 $this-> msg_error_log = "(" . __METHOD__ . ")";
968             }
969             . $this-> db-> _error_number().':'. $this-> db-> _error_message();
970             throw new Exception("(" . __METHOD__ . ")");
971         }
972     }
973     for($i=0;$i< sizeof($this-> accion_nutricional);$i++)
974     {
975         $data6 = array(
976             // accion nutricional
977             'id_persona' => $this-> id,
978             'id_accion_nutricional' => $this-> accion_nutricional[$i],
979             'fecha' => date('Y-m-d H:i:s', strtotime($this-
> accion_nutricional[$i])),
980             'id_asu_um' => $id_asu_um,
981         );
982         if($this-> accion_nutricional[$i]!="")
983     {
984             $result6 = $this-> db-> insert('cns_control_accion_nutricional',
985             $data6);
986             if (!$result6)
987             {
988                 $this-> msg_error_usr = "Error Accion nutricional." ;
989                 $this-> msg_error_log = "(" . __METHOD__ . ")";
990             }
991             . $this-> db-> _error_number().':'. $this-> db-> _error_message();
992             throw new Exception("(" . __METHOD__ . ")");
993         }
994     }
995     for($i=0;$i< sizeof($this-> peso);$i++)
996     {
997         $data7 = array(
998             // nutricion
999             'id_persona' => $this-> id,
1000             'peso' => $this-> peso[$i],

```

```

1002             'altura' => $this-> altura[$i],
1003             'talla' => $this-> talla[$i],
1004             'hemoglobina' => $this-> hemoglobina[$i],
1005             'fecha' => date('Y-m-d H:i:s', strtotime($this-> fnutricion[$i])),
1006             'id_asu_um' => $id_asu_um,
1007         );
1008     );
1009     if($this-> peso[$i]!=""
1010        ||$this-> altura[$i]!=""
1011        ||$this-> hemoglobina[$i]!="")
1012     {
1013         $result7 = $this-> db-> insert('cns_control_nutricional', $data7);
1014         if (! $result7)
1015         {
1016             $this-> msg_error_usr = "Error Nutricion." ;
1017             $this-> msg_error_log = "(" . __METHOD__ . ") =>
1018             . $this-> db-> _error_number().':'. $this-> db-> _error_message();
1019             throw new Exception("(" . __METHOD__ . ") => "
1020             . $this-> db-> _error_number().':'. $this-> db-> _error_message());
1021         }
1022     }
1023     for($i=0;$i< sizeof($this-> afiliacion);$i++)
1024     {
1025         $data8 = array(
1026             // afiliacion
1027             'id_persona' => $this-> id,
1028             'id_afiliacion' => $this-> afiliacion[$i],
1029             'ultima_actualizacion' => date('Y-m-d H:i:s'),
1030         );
1031         if($this-> afiliacion[$i]!="")
1032         {
1033             $result8 = $this-> db-> insert('cns_persona_x_afiliacion', $data8);
1034             if (! $result8)
1035             {
1036                 $this-> msg_error_usr = "Error Afiliacion." ;
1037                 $this-> msg_error_log = "(" . __METHOD__ . ") =>
1038                 . $this-> db-> _error_number().':'. $this-> db-> _error_message();
1039             }
1040         }
1041         if(!empty($this-> peri_cefa)){
1042             for($index=0; $index< sizeof($this-> peri_cefa); $index++){
1043                 $datosPeriCefa = array(
1044                     'id_persona' => $this-> id,
1045                     'fecha' => date('Y-m-d H:i:s', strtotime($this-
1046 > fecha_peri_cefa[$index])),
1047                     'perimetro_cefalico' => $this-> peri_cefa[$index],
1048                     'id_asu_um' => $id_asu_um,
1049                 );
1050                 $resultPeriCefa = $this-> db-> insert('cns_control_peri_cefa',
1051 $datosPeriCefa);
1052                 if (! $resultPeriCefa)
1053                 {
1054                     $this-> msg_error_usr = "Error Perímetro Cefálico." ;
1055                     $this-> msg_error_log = "(" . __METHOD__ . ") =>
1056                     . $this-> db-> _error_number().':'. $this-> db-> _error_message();
1057                 }
1058             }
1059             if(!empty($this-> estimulacion_fecha)){
1060                 for($index=0; $index< sizeof($this-> estimulacion_fecha); $index++){
1061                     $datosEstimulacion = array(
1062                         'id_persona' => $this-> id,
1063                         'fecha' => date('Y-m-d H:i:s', strtotime($this-
1064 > estimulacion_fecha[$index])),
1065                         'tutor_capacitado' => $this-> estimulacion_capacitado[$index],
1066                         'id_asu_um' => $id_asu_um,
1067                     );
1068                     $resultEstimulacion = $this-> db-> insert('cns_estimulacion_temprana',
1069 $datosEstimulacion);
1070                     if (! $resultEstimulacion)
1071                     {
1072                         $this-> msg_error_usr = "Error Estimulación Temprana." ;

```

```

1071             $this-> msg_error_log = "(" . __METHOD__ . ") =>
1072     . $this-> db-> _error_number().': '.$this-> db-> _error_message();
1073     throw new Exception("(" . __METHOD__ . ") => "
1074     . $this-> db-> _error_number().': '.$this-> db-> _error_message());
1075 }
1076 }
1077 if(!empty($this-> sales_fecha)){
1078     for($index=0; $index< sizeof($this-> sales_fecha); $index++){
1079         $datosSRO = array(
1080             'id_persona' => $this-> id,
1081             'fecha' => date('Y-m-d H:i:s', strtotime($this-
1082 > sales_fecha[$index])),
1083             'cantidad' => $this-> sales_cantidad[$index],
1084             'id_asu_um' => $id_asu_um);
1085
1086         $resultSRO = $this-> db-> insert('cns_sales_rehidratacion', $datosSRO);
1087         if (!$resultSRO)
1088         {
1089             $this-> msg_error_usr = "Error Sales de Rehidratación
Oral." ;
1090             $this-> msg_error_log = "(" . __METHOD__ . ") =>
1091     . $this-> db-> _error_number().': '.$this-> db-> _error_message();
1092     throw new Exception("(" . __METHOD__ . ") => "
1093     . $this-> db-> _error_number().': '.$this-> db-> _error_message());
1094         }
1095     }
1096     return $this-> id;
1097 }
1098 /**
1099 * Este metodo actualiza o inserta los datos que permiten el envio de la informacion a la
tarjeta por nfc
1100 */
1101 * @param string $entorno id del entorno
1102 * @param strin $persona id de la persona a la que se le asigna una
tarjeta
1103 * @param string $fecha fecha en que se genera el evento
1104 * @param string $archivo nombre del archivo que se genero
1105 * @param booleana $impreso determina si el archivo fue escrita en la
tarjeta o no
1106 *
1107 * @return result()
1108 */
1109 */
1110 public function entorno_x_persona($entorno,$persona,$fecha,$archivo,$impreso)
1111 {
1112     $data = array(
1113         'id_entorno' => $entorno,
1114         'id_persona' => $persona,
1115         'fecha_entrega' => $fecha,
1116         'nombre_archivo' => $archivo,
1117         'impreso_tes' => $impreso,
1118     );
1119     $query = $this-> db-> get_where('tes_entorno_x_persona', array('id_persona' =>
$persona));
1120     if ($query-> num_rows() <= 0)
1121         $result = $this-> db-> insert('tes_entorno_x_persona', $data);
1122     else
1123     {
1124         $this-> db-> where('id_persona' , $persona);
1125         $result = $this-> db-> update('tes_entorno_x_persona', $data);
1126     }
1127     if (!$result)
1128     {
1129         $this-> msg_error_log = "(" . __METHOD__ . ") => "
1130     . $this-> db-> _error_number().': '.$this-> db-> _error_message();
1131     throw new Exception("(" . __METHOD__ . ") => "
1132     . $this-> db-> _error_number().': '.$this-> db-> _error_message());
1133 }
1134 /**
1135 *
1136 * Este metodo valida que exista un archivo para enviar a la tarjeta por nfc
1137 *
1138 * @param strin $persona id de la persona a la que se le asigna una

```

```

tarjeta
1139     * @param      string      $archivo      nombre del archivo que se genero
1140
1141     * @return      result()
1142
1143     */
1144     public function valid_card($persona,$archivo)
1145     {
1146         $query = $this-> db-> get_where('tes_entorno_x_persona', array('id_persona' =>
$persona, "nombre_archivo" => $archivo));
1147         return ($query-> num_rows() > 0);
1148     }
1149     /**
1150     *
1151     * Extrae el folio que se anexa en el envio para la tarjeta
1152     *
1153     * @param      strin       $persona      id de la persona a la que se le asigna una
tarjeta
1154     *
1155     * @return      result()
1156
1157     */
1158     public function getfolio($persona)
1159     {
1160         $query = $this-> db-> get_where('tes_entorno_x_persona', array('id_persona' =>
$persona));
1161         return $query-> result();
1162     }
1163
1164     /**
1165     *
1166     * Actualiza los datos basicos del paciente
1167     *
1168     * @return      result()
1169
1170     */
1171     public function update_basico()
1172     {
1173         //date_default_timezone_set('UTC');
1174         $data = array(
1175             // basico
1176             'id_nacionalidad' => $this-> nacionalidad,
1177             'nombre' => $this-> nombre,
1178             'apellido_paterno' => $this-> paterno,
1179             'apellido_materno' => $this-> materno,
1180             'id_asu_localidad_nacimiento' => $this-> lnacimiento,
1181             'curp' => $this-> curp,
1182             'sexo' => $this-> sexo,
1183             'id_tipo_sanguineo' => $this-> sangre,
1184             'id_parto_multiple' => $this-> parto,
1185             'tamiz_neonatal' => $this-> tamiz_neonatal,
1186             'precurrp' => $this-> precurrp,
1187             'fecha_nacimiento' => date('Y-m-d H:i:s', strtotime($this-> fnacimiento));
1188             $this-> db-> where('id', $this-> id);
1189             $result = $this-> db-> update('cns_persona', $data);
1190             if (!$result)
1191             {
1192                 $this-> msg_error_usr = "Actualizacion Fallida." ;
1193                 $this-> msg_error_log = "(" . __METHOD__ . ") => "
> $this-> _error_number()." : '$this-> db-> _error_message();"
1194                 throw new Exception("(" . __METHOD__ . ") => "
> $this-> db-> _error_number()." : '$this-> db-> _error_message();"
1195             }
1196         }
1197     /**
1198     *
1199     * Actualiza la unidad medica tratante del paciente
1200     *
1201     * @return      result()
1202
1203     */
1204     public function update_umt()
1205     {
1206         //date_default_timezone_set('UTC');
1207         $data = array(
1208             // civil
1209             'fecha_registro' => date('Y-m-d H:i:s', strtotime($this-> fechacivil)),
1210             'id_asu_um_tratante' => $this-> umt);
1211             $this-> db-> where('id', $this-> id);
1212             $result = $this-> db-> update('cns_persona', $data);

```

```

1213     if (!$result)
1214     {
1215         $this-> msg_error_usr = "Actualizacion Fallida." ;
1216         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message(); . $this->
1217         throw new Exception("(" . __METHOD__ . ") => " . $this-> db-
> _error_number().': '.$this-> db-> _error_message());
1218     }
1219 }
1220 /**
1221 *
1222 * actualiza la direccion del paciente
1223 *
1224 * @return      result()
1225 *
1226 */
1227 public function update_direccion()
1228 {
1229     //date_default_timezone_set('UTC');
1230     $compania=$this-> compania;
1231     if($compania=="") $compania=NULL;
1232     $data = array(
1233         // direccion
1234         'calle_domicilio' => $this-> calle,
1235         'referencia_domicilio' => $this-> referencia,
1236         'colonia_domicilio' => $this-> colonia,
1237         'id_asu_localidad_domicilio' => $this-> localidad,
1238         'numero_domicilio' => $this-> numero,
1239         'cp_domicilio' => $this-> cp,
1240         'ageb' => $this-> ageb,
1241         'sector' => $this-> sector,
1242         'manzana' => $this-> manzana,
1243
1244         'telefono_domicilio' => $this-> telefono,
1245         'id_operadora_cellular' => $compania,
1246         'celular' => $this-> celular,
1247         'ultima_actualizacion' => date("Y-m-d H:i:s")
1248     );
1249     $this-> db-> where('id' , $this-> id);
1250     $result = $this-> db-> update('cns_persona' , $data);
1251     if (!$result)
1252     {
1253         $this-> msg_error_usr = "Actualizacion Fallida." ;
1254         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message(); . $this->
1255         throw new Exception("(" . __METHOD__ . ") => " . $this-> db-
> _error_number().': '.$this-> db-> _error_message());
1256     }
1257 }
1258 /**
1259 *
1260 * actualiza el registro civil del paciente
1261 *
1262 * @return      result()
1263 *
1264 */
1265 public function update_regcivil()
1266 {
1267
1268     $data = array(
1269         'id_persona' => $this-> id,
1270         'fecha_registro' => date('Y-m-d H:i:s' , strtotime($this-> fechacivil)),
1271         'id_localidad_registro_civil' => $this-> lugarcivil,
1272     );
1273     $query = $this-> db-> get_where('cns_registro_civil' , array('id_persona' =>
$this-> id));
1274     if($query-> num_rows() > 0)
1275     {
1276         $this-> db-> where('id_persona' , $this-> id);
1277         $result = $this-> db-> update('cns_registro_civil' , $data);
1278     }
1279     else
1280         $res = $this-> db-> insert('cns_registro_civil' , $data);
1281 }
1282 /**
1283 *
1284 * Actualiza los datos del tutor del paciente
1285 *
1286 * @return      result()
1287 */

```

```

1288 */
1289 public function update_tutor()
1290 {
1291     $companiaT=$this->companiaT;
1292     if($companiaT=="") $companiaT=NULL;
1293     $data0 = array(
1294         // tutor
1295         'nombre' => $this->nombreT,
1296         'apellido_paterno' => $this->paternoT,
1297         'apellido_materno' => $this->maternoT,
1298         'curp' => $this->curpT,
1299         'sexo' => $this->sexoT,
1300
1301         'telefono' => $this->telefonoT,
1302         'id_operadora_cellular' => $companiaT,
1303         'celular' => $this->celularT,
1304
1305     );
1306
1307     // if($this->idtutor=="")
1308     {
1309         $unico_idtutor=md5(unigid());
1310         $data0['id']=$unico_idtutor;
1311         $result0 = $this->db->insert('cns_tutor', $data0);
1312         if (!$result0)
1313         {
1314             $this->msg_error_usr = "No se guardo Tutor." ;
1315             $this->msg_error_log = "(" . __METHOD__ . ") => " . $this-
>db->_error_number().':'. $this->db->_error_message();
1316         > db-> _error_number().':'. $this->db-> _error_message(); . $this-
1317         }
1318         else
1319         {
1320             $this->setidtutor($unico_idtutor);
1321             $this->idtutor=$unico_idtutor;
1322         }
1323     }
1324     else
1325     {
1326         $this->db->where('id', $this->idtutor);
1327         $result0 = $this->db->update('cns_tutor', $data0);
1328         if (!$result0)
1329         {
1330             $this->msg_error_usr = "No se actualizo Tutor." ;
1331             $this->msg_error_log = "(" . __METHOD__ . ") => " . $this-
>db->_error_number().':'. $this->db->_error_message();
1332         > db-> _error_number().':'. $this->db-> _error_message(); . $this-
1333         }
1334     }
1335
1336     // relacion tutor paciente
1337     $data01 = array(
1338         'id_persona' => $this->id,
1339         'id_tutor' => $this->idtutor,
1340         'ultima_actualizacion' => date('Y-m-d H:i:s'),
1341     );
1342     if ($this->db->delete('cns_persona_x_tutor', array('id_persona' => $this-
>id)))
1343     {
1344         $result01 = $this->db->insert('cns_persona_x_tutor', $data01);
1345         if (!$result01)
1346         {
1347             $this->msg_error_usr = "Error actualizando Tutor." ;
1348             $this->msg_error_log = "(" . __METHOD__ . ") => " . $this-
>db->_error_number().':'. $this->db->_error_message();
1349         > db-> _error_number().':'. $this->db->_error_message(); . $this-
1350         }
1351     }
1352 }
1353 /**
1354 *
1355 * Actualiza los datos de las alergias del paciente
1356 *
1357 * @return result()
1358 *
1359 */
1360 public function update_alergia()

```

```

1361    {
1362        $id_asu_um=$this-> umt;
1363        if ($this-> db-> delete('cns_persona_x_alergia', array('id_persona' => $this-
> id)))
1364            for($i=0;$i< sizeof($this-> alergias);$i++)
1365            {
1366                $data1 = array(
1367                    // alergias
1368                    'id_persona' => $this-> id,
1369                    'id_alergia' => $this-> alergias[$i],
1370                    'ultima_actualizacion' => date('Y-m-d H:i:s'),
1371                );
1372                if($this-> alergias[$i]!="")
1373                {
1374                    $result1 = $this-> db-> insert('cns_persona_x_alergia', $data1);
1375                    if (!$result1)
1376                    {
1377                        $this-> msg_error_usr = "Error actualizando Alergias." ;
1378                        $this-> msg_error_log = "(" . __METHOD__ . ") => "
1379 . $this-> db-> _error_number().': '.$this-> db-> _error_message();
1380                     throw new Exception("(" . __METHOD__ . ") => "
1381 . $this-> db-> _error_number().': '.$this-> db-> _error_message());
1382                 }
1383             }
1384         /**
1385         *
1386         * Actualiza las vacunas del paciente
1387         *
1388         * @return      result()
1389         */
1390     public function update_vacuna()
1391     {
1392         $id_asu_um=$this-> umt;
1393         if ($this-> db-> delete('cns_control_vacuna', array('id_persona' => $this-
> id)))
1394             for($i=0;$i< sizeof($this-> vacuna);$i++)
1395             {
1396                 $data2 = array(
1397                     // vacuna
1398                     'id_persona' => $this-> id,
1399                     'id_vacuna' => $this-> vacuna[$i],
1400                     'fecha' => date('Y-m-d H:i:s', strtotime($this-> fvacuna[$i])),
1401                     'id_asu_um' => $id_asu_um,
1402                     'codigo_barras' => $this-> codigo_barras[$i],
1403                 );
1404                 if($this-> fvacuna[$i]!="")
1405                 {
1406                     $result2 = $this-> db-> insert('cns_control_vacuna', $data2);
1407                     if (!$result2)
1408                     {
1409                         $this-> msg_error_usr = "Error actualizando vacunas." ;
1410                         $this-> msg_error_log = "(" . __METHOD__ . ") => "
1411 . $this-> db-> _error_number().': '.$this-> db-> _error_message();
1412                     throw new Exception("(" . __METHOD__ . ") => "
1413 . $this-> db-> _error_number().': '.$this-> db-> _error_message());
1414                 }
1415             }
1416         /**
1417         *
1418         * Actualiza el control consulta del paciente
1419         *
1420         * @return      result()
1421         */
1422     public function update_consulta()
1423     {
1424         $id_asu_um=$this-> umt;
1425         if ($this-> db-> delete('cns_control_consulta', array('id_persona' => $this-
> id)))
1426             for($i=0;$i< sizeof($this-> consulta);$i++)
1427             {
1428                 $data5 = array(
1429                     // consulta
1430                     'id_persona' => $this-> id,
1431                     'clave_cielo' => $this-> consulta[$i],
1432                 );
1433             }

```

```

1434             'fecha' => date('Y-m-d H:i:s', strtotime($this-> fconsulta[$i])),  

1435             'id_asu_um' => $id_asu_um,  

1436             'id_tratamiento' => $this-> tconsulta[$i],  

1437             'grupo_fecha_secuencial' => date('Y-m-d H:i:s', strtotime($this-  

> fconsulta[$i])),  

1438         );  

1439         if($this-> consulta[$i]!="")  

1440         {  

1441             $result5 = $this-> db-> insert('cns_control_consulta', $data5);  

1442             if (!$result5)  

1443             {  

1444                 $this-> msg_error_usr = "Error actualizando Consulta." ;  

1445                 $this-> msg_error_log = "(" . __METHOD__ . ") => "  

1446 . $this-> db-> _error_number().":'." . $this-> db-> _error_message();  

1447             throw new Exception("(" . __METHOD__ . ") => " . $this-  

> db-> _error_number().":'." . $this-> db-> _error_message());  

1448         }  

1449     }  

1450 }  

1451 /**
1452 *
1453 * Actualiza el control accion nutricional del paciente
1454 *
1455 * @return      result()
1456 *
1457 */
1458 public function update_accion()
1459 {
1460     $id_asu_um=$this-> umt;
1461     if ($this-> db-> delete('cns_control_accion_nutricional', array('id_persona' =>  

1462 $this-> id)))
1463     for($i=0;$i< sizeof($this-> accion_nutricional);$i++)
1464     {
1465         $data6 = array(
1466             // accion nutricional
1467             'id_persona' => $this-> id,
1468             'id_accion_nutricional' => $this-> accion_nutricional[$i],
1469             'fecha' => date('Y-m-d H:i:s', strtotime($this-> faccion_nutricional[$i])),
1470             'id_asu_um' => $id_asu_um,
1471         );
1472         if($this-> accion_nutricional[$i]!="")
1473         {
1474             $result6 = $this-> db-> insert('cns_control_accion_nutricional', $data6);
1475             if (!$result6)
1476             {
1477                 $this-> msg_error_usr = "Error actualizando Accion  

nutricional." ;
1478                 $this-> msg_error_log = "(" . __METHOD__ . ") => "  

1479 . $this-> db-> _error_number().":'." . $this-> db-> _error_message();
1480             throw new Exception("(" . __METHOD__ . ") => " . $this-  

> db-> _error_number().":'." . $this-> db-> _error_message());
1481         }
1482     }
1483 }
1484 /**
1485 *
1486 * Actualiza el control nutricional del paciente
1487 *
1488 * @return      result()
1489 *
1490 */
1491 public function update_nutricion()
1492 {
1493     $id_asu_um=$this-> umt;
1494     if ($this-> db-> delete('cns_control_nutricional', array('id_persona' => $this-  

> id)))
1495     for($i=0;$i< sizeof($this-> peso);$i++)
1496     {
1497         $data7 = array(
1498             // nutricion
1499             'id_persona' => $this-> id,
1500             'peso' => $this-> peso[$i],
1501             'altura' => $this-> altura[$i],
1502             'talla' => $this-> talla[$i],
1503             'hemoglobina' => $this-> hemoglobina[$i],
1504             'fecha' => date('Y-m-d H:i:s', strtotime($this-> fnutricion[$i])),  

1505

```

```

1506             'id_asu_um' => $id_asu_um,
1507
1508         );
1509         if($this-> peso[$i]!=""
1510 >     ||$this-> hemoglobina[$i]!=""
1511 >     ||$this-> talla[$i]!="")
1512     {
1513         $result7 = $this-> db-> insert('cns_control_nutricional', $data7);
1514         if (!$result7)
1515         {
1516             $this-> msg_error_usr = "Error actualizando Nutricion." ;
1517             $this-> msg_error_log = "(" . __METHOD__ . ") => "
1518             . $this-> db-> _error_number().':'. $this-> db-> _error_message();
1519             throw new Exception("(" . __METHOD__ . ") => "
1520             . $this-> db-> _error_number().':'. $this-> db-> _error_message());
1521         }
1522     }
1523 /**
1524 *
1525 * Actualiza el tipo de beneficiario del paciente
1526 *
1527 */
1528 public function update_beneficiario()
1529 {
1530     $id_asu_um=$this-> umt;
1531
1532     if ($this-> db-> delete('cns_persona_x_afiliacion', array('id_persona' => $this-
1533 > id)))
1534     {
1535         for($i=0;$i< sizeof($this-> afiliacion);$i++)
1536         {
1537             $data8 = array(
1538                 // afiliacion
1539                 'id_persona' => $this-> id,
1540                 'id_afiliacion' => $this-> afiliacion[$i],
1541                 'ultima_actualizacion' => date('Y-m-d H:i:s'),
1542             );
1543             if($this-> afiliacion[$i]!="")
1544             {
1545                 $result8 = $this-> db-> insert('cns_persona_x_afiliacion', $data8);
1546                 if (!$result8)
1547                 {
1548                     $this-> msg_error_usr = "Error actualizando Afiliacion." ;
1549                     $this-> msg_error_log = "(" . __METHOD__ . ") => "
1550                     . $this-> db-> _error_number().':'. $this-> db-> _error_message();
1551                 }
1552             }
1553         }
1554     /**
1555     *
1556     * Hace update de la tabla que este sincronizando dependiendo del resultado
1557     *
1558     * @param string $mac Mac de la tableta
1559     * @param strin $status nuevo status
1560     * @param string $version version de la apk de la tableta
1561     * @param string $fecha fecha del evento
1562     *
1563     * @return result()
1564     */
1565
1566 public function update_status_tableta($mac,$status,$version,$fecha)
1567 {
1568     $data = array
1569     (
1570         'id_tes_estado_tableta' => $status,
1571         'id_version' => $version,
1572         'ultima_actualizacion' => $fecha,
1573     );
1574     $this-> db-> where('mac' , $mac);
1575     $result0 = $this-> db-> update('tes_tableta' , $data);
1576     if (!$result0)
1577     {
1578         $this-> msg_error_usr = "No se actualizo Tutor." ;
1579         $this-> msg_error_log = "(" . __METHOD__ . ") => "
1580         . $this-> db-> _error_number().':'. $this-> db-> _error_message();
1581     }
1582 }

```

```

>     db-> _error_number()."': '$this-> db-> _error_message();
1580         throw new Exception("(" . __METHOD__ .") => " . $this-> db-
1581     }
1582 }
1583 /**
1584 *
1585 * Este metodo retorna el list de las personas enroladas
1586 *
1587 * @param      string      $keywords    palabras claves para hacer el filtro
1588 * @param      string      $offset      inicio del registro
1589 * @param      string      $row_count  numero de filas a mostrar por pagina
1590 *
1591 * @return     result()
1592 *
1593 */
1594 public function getListEnrolamiento($keywords = '', $offset = null, $row_count = null)
1595 {
1596     if(!empty($offset) && !empty($row_count))
1597         $this-> db-> limit($offset, $row_count);
1598     else if (!empty($offset))
1599         $this-> db-> limit($offset);
1600
1601     if (empty($keywords)){
1602
1603         $query = $this-> db-> get('cns_persona');
1604     }
1605     else
1606     {
1607         $this-> db-> select('*');
1608         $this-> db-> from('cns_persona');
1609         $cadena=explode(" ", $keywords);
1610         if(count($cadena)==1)
1611         {
1612             $this-> db-> like('curp', $keywords);
1613             $this-> db-> or_like('nombre', $keywords);
1614             $this-> db-> or_like('apellido_paterno', $keywords);
1615             $this-> db-> or_like('apellido_materno', $keywords);
1616         }
1617         else if(count($cadena)==2)
1618         {
1619             $this-> db-> where("(nombre like '%' . $cadena[0]. "%" and
1620 (nombre like '%' . $cadena[1]. "%" or apellido_paterno like '%' . $cadena[1]. "%")) or
1621 (apellido_paterno like '%' . $cadena[0]. "%" and apellido_materno like
1622 '%' . $cadena[1]. "%"));
1623         else if (count($cadena)==3)
1624         {
1625             $this-> db-> where("(nombre like '%' . $cadena[0]. "%" and
1626 nombre like '%' . $cadena[1]. "%" and nombre like '%' . $cadena[2]. "%") or
1627 (apellido_paterno like '%' . $cadena[0]. "%" and apellido_paterno like
1628 '%' . $cadena[1]. "% and apellido_paterno like '%' . $cadena[2]. "%") or
1629 (nombre like '%' . $cadena[0]. "%" and apellido_paterno like '%' . $cadena[1]. "% and
1630 apellido_paterno like '%' . $cadena[2]. "%") or
1631 (nombre like '%' . $cadena[0]. "%" and apellido_paterno like '%' . $cadena[1]. "% and
1632 apellido_paterno like '%' . $cadena[3]. "%") or
1633 (apellido_paterno like '%' . $cadena[0]. "%" and apellido_paterno like
1634 '%' . $cadena[1]. "% and apellido_paterno like '%' . $cadena[3]. "%") or
1635 (nombre like '%' . $cadena[0]. "%" and nombre like
1636 '%' . $cadena[1]. "% and apellido_paterno like '%' . $cadena[2]. "% and
1637 apellido_paterno like '%' . $cadena[3]. "%") or
1638 (apellido_paterno like '%' . $cadena[0]. "%" and apellido_paterno like
1639 '%' . $cadena[1]. "% and apellido_materno like '%' . $cadena[2]. "% and
1640 apellido_materno like '%' . $cadena[3]. "%"));
1641     }
1642 }

```

```

1639         else if (_count($cadena)==5)
1640     {
1641         $this-> db-> where("(nombre like '%' . $cadena[0]. "%" and
1642 nombre like '%' . $cadena[1]. "%" and nombre like '%' . $cadena[2]. "%" and
1643 apellido_paterno like '%' . $cadena[3]. "%" and apellido_materno like
1644 '%' . $cadena[4]. "%") or
1645         (nombre like '%' . $cadena[0]. "%" and nombre like
1646 '%' . $cadena[1]. "%" and nombre like '%' . $cadena[2]. "%" and apellido_paterno like
1647 '%' . $cadena[3]. "%" and apellido_materno like '%' . $cadena[4]. "%") or
1648         (nombre like '%' . $cadena[0]. "%" and nombre like
1649 '%' . $cadena[1]. "%" and apellido_paterno like '%' . $cadena[2]. "%" and apellido_paterno like
1650 '%' . $cadena[3]. "%" and apellido_paterno like '%' . $cadena[4]. "%") or
1651         (apellido_paterno like '%' . $cadena[0]. "%" and apellido_paterno like
1652 '%' . $cadena[1]. "%" and apellido_paterno like '%' . $cadena[2]. "%" and apellido_materno like
1653 '%' . $cadena[3]. "%" and apellido_materno like '%' . $cadena[4]. "%") or
1654         (apellido_paterno like '%' . $cadena[0]. "%" and apellido_paterno like
1655 '%' . $cadena[1]. "%" and apellido_materno like '%' . $cadena[2]. "%" and apellido_materno like
1656 '%' . $cadena[3]. "%" and apellido_materno like '%' . $cadena[4]. "%");
1657     }
1658     $query = $this-> db-> get();
1659 }
1660
1661     if (!$query){
1662         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
1663         $this-> msg_error_log = "("
1664 > _error_number(). : '$this-> db-> _error_message();' . $this->
1665 > _error_number(). : '$this-> db-> _error_message();' . $this-> db-
1666     }
1667     else
1668         return $query-> result();
1669     return;
1670 }
1671 /**
1672 *
1673 * Devuelve el numero de filas en la tabla cns_persona
1674 *
1675 * @param string $keywords palabras clave para hacer el filtro
1676 *
1677 * @return result()
1678 *
1679 */
1680 public function getNumRows($keywords = '')
1681 {
1682     if (!$keywords)
1683         $query = $this-> db-> get('cns_persona');
1684     else
1685     {
1686
1687         $this-> db-> select('*');
1688         $this-> db-> from('cns_persona');
1689         $cadena=explode(" ", $keywords);
1690         if(_count($cadena)==1)
1691         {
1692             $this-> db-> like('curp', $keywords);
1693             $this-> db-> or_like('nombre', $keywords);
1694             $this-> db-> or_like('apellido_paterno', $keywords);
1695             $this-> db-> or_like('apellido_materno', $keywords);
1696         }
1697         else if(_count($cadena)==2)
1698         {
1699             $this-> db-> where("(nombre like '%' . $cadena[0]. "%" and
1700 (nombre like '%' . $cadena[1]. "%" or apellido_paterno like '%' . $cadena[1]. "%")) or
1701 (apellido_paterno like '%' . $cadena[0]. "%" and apellido_materno like
1702 '%' . $cadena[1]. "%"));
1703         }
1704         else if (_count($cadena)==3)
1705         {

```

```

1694      $this-> db-> where("(nombre like '%' . $cadena[0].%" and
1695      nombre like '%' . $cadena[1].%" and nombre like '%' . $cadena[2].%"') or
1696      (apellido_paterno like '%' . $cadena[0].%" and apellido_paterno like
1697      '%' . $cadena[1].%" and apellido_paterno like '%' . $cadena[2].%"') or
1698      (nombre like '%' . $cadena[0].%" and apellido_paterno like
1699      '%' . $cadena[1].%" and apellido_paterno like '%' . $cadena[2].%"') or
1700      (apellido_paterno like '%' . $cadena[0].%" and apellido_paterno like
1701      '%' . $cadena[1].%" and apellido_paterno like '%' . $cadena[2].%"") );
1702      }
1703      else if (count($cadena)==4)
1704      {
1705          $this-> db-> where("(nombre like '%' . $cadena[0].%" and
1706          nombre like '%' . $cadena[1].%" and nombre like '%' . $cadena[2].%" and
1707          apellido_paterno like '%' . $cadena[3].%"') or
1708          (apellido_paterno like '%' . $cadena[0].%" and apellido_paterno like
1709          '%' . $cadena[1].%" and apellido_paterno like '%' . $cadena[2].%" and
1710          apellido_materno like '%' . $cadena[3].%"') or
1711          (nombre like '%' . $cadena[0].%" and nombre like
1712          '%' . $cadena[1].%" and apellido_paterno like '%' . $cadena[2].%" and
1713          apellido_paterno like '%' . $cadena[3].%"') or
1714          (nombre like '%' . $cadena[0].%" and nombre like
1715          '%' . $cadena[1].%" and apellido_paterno like '%' . $cadena[2].%" and
1716          apellido_paterno like '%' . $cadena[3].%"') or
1717          (apellido_paterno like '%' . $cadena[0].%" and apellido_paterno like
1718          '%' . $cadena[1].%" and apellido_paterno like '%' . $cadena[2].%" and
1719          apellido_materno like '%' . $cadena[3].%"') or
1720      }
1721      if(!$query)
1722      {
1723          $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
1724          $this-> msg_error_log = '(__METHOD__.') => ' . $this-> db-
> _error_number(). : __METHOD__.') => ' . $this-> db-
> _error_message();
1725          throw new Exception("(__METHOD__.') => " . $this-> db-
> _error_number(). : __METHOD__.') => " . $this-> db-
> _error_message());
1726      }
1727      return $query-> num_rows;
1728  }
1729 /**
1730 *
1731 * * Obtiene la informacion de la persona
1732 */

```

```

1734     * @param      string      $id      identificado de la persona
1735     *
1736     * @return      result()
1737     */
1738
1739     public function getById($id)
1740     {
1741
1742         $this-> db-> select('p.* , s.id as sangre, s.descripcion as tsangre, n.id as
1743         nacionalidadid, n.descripcion as nacionalidad, o.id as operadoraid, o.descripcion as operadora, t.id
1744         as idT, t.curp as curpT, t.nombre as nombreT, t.apellido_paterno as paternoT, t.apellido_materno as
1745         maternoT, t.sexo as sexoT, t.telefono as telefonoT, t.celular as celularT, ol.id as operadoraTid,
1746         ol.descripcion as operadoraT, rc.id_localidad_registro_civil, pm.descripcion as parto,
1747         TIMESTAMPDIFF(month, fecha_nacimiento, CURDATE()) AS edad_meses, tamiz_neonatal');
1748
1749         $this-> db-> from('cns_persona p');
1750         $this-> db-> join('cns_nacionalidad n', 'n.id = p.id_nacionalidad', 'left');
1751         $this-> db-> join('cns_tipo_sanguineo s', 's.id = p.id_tipo_sanguineo', 'left');
1752         $this-> db-> join('cns_operadora Celular o', 'o.id =
1753         p.id_operadora_celular', 'left');
1754         $this-> db-> join('cns_persona_x_tutor pt', 'pt.id_persona = p.id', 'left');
1755         $this-> db-> join('cns_tutor t', 't.id = pt.id_tutor', 'left');
1756         $this-> db-> join('cns_operadora Celular ol', 'ol.id =
1757         t.id_operadora_celular', 'left');
1758         $this-> db-> join('cns_registro_civil rc', 'rc.id_persona = p.id', 'left');
1759         $this-> db-> join('cns_parto_multiple pm', 'pm.id = p.id_parto_multiple', 'left');
1760         $this-> db-> where('p.id', $id);
1761         $query = $this-> db-> get();
1762         if (!$query){
1763             $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
1764             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
1765             > db-> _error_number().': '.$this-> db-> _error_message();
1766             throw new Exception("(" . __METHOD__ . ") => " . $this-> db-
1767             > _error_number().': '.$this-> db-> _error_message());
1768         }
1769         else
1770             return $query-> row();
1771         return;
1772     }
1773
1774 /**
1775 *
1776 * obtiene informacion del registro civil
1777 *
1778 * @param      string      $id      identificador de la persona
1779 *
1780 * @return      result()
1781 */
1782
1783     public function getRegistro_civil($id)
1784     {
1785
1786         $this-> db-> select('*');
1787         $this-> db-> from('cns_registro_civil');
1788         $this-> db-> where('id_persona', $id);
1789         $query = $this-> db-> get();
1790         if (!$query){
1791             $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
1792             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
1793             > db-> _error_number().': '.$this-> db-> _error_message();
1794             throw new Exception("(" . __METHOD__ . ") => " . $this-> db-
1795             > _error_number().': '.$this-> db-> _error_message());
1796         }
1797         else
1798             return $query-> row();
1799         return;
1800     }
1801
1802 /**
1803 *
1804 * Obtiene las alergias asociadas a una persona
1805 *
1806 * @param      string      $id      identificador de la persona
1807 * @param      strin      $order      nombre del campo para hacer el order by
1808 *
1809 * @return      result()
1810 */
1811
1812     public function getAlergia($id = '', $order = '')
1813     {
1814         $this-> db-> select('a.id, a.descripcion');
1815         $this-> db-> from('cns_persona_x_alergia p');

```

```

1803     $this-> db-> join('cns_alergia a', 'a.id = p.id_alergia','left');
1804     $this-> db-> where('p.id_persona', $id);
1805     if($order!="")
1806     $this-> db-> order_by($order, "asc");
1807     $query = $this-> db-> get();
1808
1809     if (! $query){
1810         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
1811         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
1812 > _error_number().': '.$this-> db-> _error_message(); . $this-> db-
1813 > _error_number().': '.$this-> db-> _error_message(); . $this-> db-
1814     }
1815     else
1816         return $query-> result();
1817     return;
1818 }
1819 /**
1820 *
1821 * Obtiene las afiliaciones asociadas a una persona
1822 *
1823 * @param string $id identificador de la persona
1824 * @param strin $order nombre del campo para hacer el order by
1825 *
1826 * @return result()
1827 *
1828 */
1829
1830 public function getAfiliaciones($id = '',$order="" )
1831 {
1832     $this-> db-> select('a.id, a.descripcion');
1833     $this-> db-> from('cns_persona_x_afiliacion p');
1834     $this-> db-> join('cns_afiliacion a', 'a.id = p.id_afiliacion','left');
1835     $this-> db-> where('p.id_persona', $id);
1836     if($order!="")
1837     $this-> db-> order_by($order, "asc");
1838     $query = $this-> db-> get();
1839
1840     if (! $query){
1841         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
1842         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
1843 > _error_number().': '.$this-> db-> _error_message(); . $this-> db-
1844 > _error_number().': '.$this-> db-> _error_message(); . $this-> db-
1845     }
1846     else
1847         return $query-> result();
1848     return;
1849 }
1850 /**
1851 * Hace select de los catalogos que tengan relacion con una persona para mostrarlos en el
view
1852 *
1853 * @param string $catalog Nombre de la tabla
1854 * @param string $id identificador de la persona
1855 * @param strin $order1 nombre del campo para hacer el order by
1856 * @param strin $order2 nombre del campo para hacer el order by
1857 *
1858 * @return result()
1859 *
1860 */
1861
1862 public function get_catalog_view($catalog,$id,$order1="" ,,$order2="" )
1863 {
1864     $this-> db-> select('*');
1865     $this-> db-> from('cns_control_'. $catalog .' p');
1866     $this-> db-> join('cns_'. $catalog .' a', 'a.id = p.id_'. $catalog , 'left');
1867     $this-> db-> where('p.id_persona', $id);
1868     if($order1!="")
1869     $this-> db-> order_by($order1, "asc");
1870     if($order2!="")
1871     $this-> db-> order_by($order2, "asc");
1872     $query = $this-> db-> get();
1873     if (! $query)
1874     {
1875         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
1876         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
1877 > _error_number().': '.$this-> db-> _error_message(); . $this-

```

```

1877             throw new Exception("(" . __METHOD__ . ") => " . $this-> db-
> _error_number().': '.$this-> db-> _error_message());
1878         }
1879     else
1880         return $query-> result();
1881     return null;
1882 }
1883 /**
1884 *
1885 * Obtiene los datos del control nutricional asociados a una persona
1886 *
1887 * @param      string      $id      identificador de la persona
1888 * @param      strin       $order    nombre del campo para hacer el order by
1889 *
1890 * @return     result()
1891 *
1892 */
1893 public function get_control_nutricional($id,$order="" )
1894 {
1895     $this-> db-> select('*');
1896     $this-> db-> from('cns_control_nutricional');
1897     $this-> db-> where('id_persona', $id);
1898     if($order!="")
1899         $this-> db-> order_by($order, "asc");
1900     else
1901         $this-> db-> order_by("fecha" , "ASC");
1902     $query = $this-> db-> get();
1903     if (!$query)
1904     {
1905         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
1906         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
1907         throw new Exception("(" . __METHOD__ . ") => " . $this-> db-
> _error_number().': '.$this-> db-> _error_message());
1908     }
1909     else
1910         return $query-> result();
1911     return null;
1912 }
1913 /**
1914 *
1915 * Hace select de las tablas cns_x que representa a los catalogos
1916 *
1917 * @param      string      $catalog  Nombre de la tabla
1918 * @param      strin       $campo    nombre del campo para hacer el where
1919 * @param      string      $id       valor del campo para el where
1920 * @param      strin       $order    nombre del campo para hacer el order by
1921 *
1922 * @return     result()
1923 *
1924 */
1925 public function
get_catalog($catalog,$campo="" , $id="" , $orden="" )
1926 {
1927     if($catalog=="cns_regla_vacuna" )
1928         $this-> db-> select('id, id_vacuna, dia_inicio_aplicacion_nacido,
dia_fin_aplicacion_nacido, id_vacuna_secuencial, dia_inicio_aplicacion_secuencial,
dia_fin_aplicacion_secuencial, ultima_actualizacion, activo, id_via_vacuna, dosis, region, esq_com,
orden_esq_com, alergias, forzar_aplicacion, observacion_region');
1929     else
1930         $this-> db-> select('*');
1931     $this-> db-> from($catalog);
1932     if($id!="")
1933         $this-> db-> where($campo, $id);
1934     if($orden!="")
1935         $this-> db-> order_by($orden, "asc");
1936     $query = $this-> db-> get();
1937     if (!$query)
1938     {
1939         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
1940         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
1941         throw new Exception("(" . __METHOD__ . ") => " . $this-> db-
> _error_number().': '.$this-> db-> _error_message());
1942     }
1943     else
1944         return $query-> result();
1945     return null;
1946 }
1947

```

```

1948
1949 /**
1950 *
1951 * Hace select de los tratamientos de las consultas
1952 *
1953 * @param      string      $catalog  Nombre de la tabla
1954 * @param      strin       $campo    nombre del campo para hacer el where
1955 * @param      string      $valor    valor del campo para el where
1956 * @param      strin       $order    nombre del campo para hacer el order by
1957 *
1958 * @return     result()
1959 *
1960 */
1961 public function get_catalog_tratamiento($catalog,$campo,$valor,$orden)
1962 {
1963     if($orden=="tipo" || $orden=="cc")
1964         $this-> db-> select('distinct(tipo),id');
1965     else
1966         $this-> db-> distinct('*');
1967     $this-> db-> from($catalog);
1968     if($campo!="")
1969         $this-> db-> where($campo, $valor);
1970     if($orden!="cc" && $campo!="tipo")
1971         $this-> db-> group_by("tipo");
1972     if($orden!=""
1973         && $orden!="cc")
1974         $this-> db-> order_by($orden, "asc");
1975     $query = $this-> db-> get();
1976     if (!$query)
1977     {
1978         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
1979         $this-> msg_error_log = "(" . __METHOD__ . ") => "
1980 > db-> _error_number().':'. $this-> db-> _error_message();
1981         throw new Exception("(" . __METHOD__ . ") => "
1982 > $this-> db-> _error_message());
1983     }
1984     else
1985         return $query-> result();
1986     return null;
1987 }
1988 /**
1989 * Obtiene el numero de resultados de una tabla
1990 *
1991 * @param      string      $catalog  Nombre de la tabla
1992 *
1993 * @return     count
1994 */
1995 public function get_catalog_count($catalog,$campo="" ,$valor="" )
1996 {
1997     if($campo!="")
1998     {
1999         $this-> db-> where($campo, $valor);
2000         $this-> db-> from($catalog);
2001         return $this-> db-> count_all_results();
2002     }
2003     else
2004         return $this-> db-> count_all($catalog);
2005 }
2006
2007 /**
2008 *
2009 * Obtiene los datos de una tabla
2010 *
2011 * @param      string      $catalog  Nombre de la tabla
2012 * @param      strin       $campo1   nombre del campo para hacer el where
2013 * @param      string      $id1     valor del campo para el where
2014 * @param      strin       $campo2   nombre del campo para hacer el where
2015 * @param      string      $id2     valor del campo para el where
2016 * @param      strin       $l1      nombre del campo para hacer el limit offset
2017 * @param      strin       $l2      nombre del campo para hacer el limit count
2018 *
2019 * @return     result()
2020 *
2021 */
2022 public function
get_catalog2($catalog,$campo1="" ,$id1="" ,$campo2="" ,$id2="" ,$l1
1="" ,$l2="" )
2023 {

```

```

2024         ini_set("memory_limit", "-1");
2025
2026     if($catalog=="tes_notificacion")
2027         $this-> db-> select('id,titulo,contenido,fecha_inicio,fecha_fin');
2028     else if($catalog=="asu_arbol_segmentacion")
2029         $this-> db-> select('id,grado_segmentacion,id_padre,orden, visible,
2030 descripción');
2031     else if($catalog=="tes_pendientes_tarjeta")
2032         $this-> db-> select('fecha, id_persona, tabla, registro_json, ');
2033     else
2034         $this-> db-> select('*');
2035     $this-> db-> from($catalog);
2036     if($id1!="")
2037         $this-> db-> where($campo1, $id1);
2038     if($id2!="")
2039         $this-> db-> where($campo2, $id2);
2040     if($l2!="")
2041         $this-> db-> limit($l2, $l1);
2042     $query = $this-> db-> get();
2043     if (!$query)
2044     {
2045         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
2046         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
2047 > db-> _error_number().':'. $this-> db-> _error_message();
2048         throw new Exception("(" . __METHOD__ . ") => " . $this-> db-
2049 > _error_number().':'. $this-> db-> _error_message());
2050     }
2051     else
2052     {
2053         *
2054         * Este metodo obtiene las notificaciones que se enviaran en la sincronizacion
2055         *
2056         * @param string $id id del arbol de segmentacion
2057         *
2058         *
2059         * @return result()
2060         *
2061     */
2062     public function get_notificacion($id)
2063     {
2064         $this-> db-> select('id,titulo,contenido,fecha_inicio,fecha_fin');
2065         $this-> db-> from("tes_notificacion");
2066         $this-> db-> like("id_arr_asu", $id);
2067         $this-> db-> where("fecha_fin >=", date("Y-m-d"));
2068         $query = $this-> db-> get();
2069         if (!$query)
2070         {
2071             $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
2072             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
2073 > db-> _error_number().':'. $this-> db-> _error_message();
2074             throw new Exception("(" . __METHOD__ . ") => " . $this-> db-
2075 > _error_number().':'. $this-> db-> _error_message());
2076         }
2077         else
2078             return $query-> result();
2079     }
2080     /*
2081     * Este metodo obtiene las personas que seran enviadas en la sincronizacion
2082     *
2083     * @param string $array arreglo con los ids de las personas que cumplen
2084     * con los requisitos del envio
2085     * @param string $fecha fecha que determina si se envia o no una persona
2086     *
2087     */
2088     public function get_cns_persona($array,$fecha="")
2089     {
2090         $this-> db-> select('id, curp, nombre, apellido_paterno, apellido_materno, sexo,
2091 id_tipo_sanguineo, fecha_nacimiento, id_asu_localidad_nacimiento, calle_domicilio, numero_domicilio,
2092 colonia_domicilio, referencia_domicilio, ageb, manzana, sector, id_asu_localidad_domicilio,
2093 cp_domicilio, telefono_domicilio, fecha_registro, id_asu_um_tratante, celular, ultima_actualizacion,
2094 id_nacionalidad, id_operadora_cellular, ultima_sincronizacion, id_parto_multiple, tamiz_neonatal');
2095         $this-> db-> from("cns_persona");
2096         if($fecha!="")
2097     }

```

```

2094     $this-> db-> where("ultima_actualizacion >=" , $fecha);
2095     $this-> db-> where_in("id_asu_um_tratante" , $array);
2096     $this-> db-> where("activo" , "1" );
2097     $this-> db-> where("(DATEDIFF(NOW(),fecha_nacimiento)/365.25)<" , 5);
2098     $query = $this-> db-> get();
2099     if (!$query)
2100     {
2101         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
2102         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
2103         > _error_number().': '.$this-> db-> _error_message();
2104         throw new Exception("(" . __METHOD__ . ") => " . $this-> db-
2105         > _error_number().': '.$this-> db-> _error_message());
2106     }
2107     else
2108         return $query-> result();
2109     return null;
2110 }
2111 /**
2112 * Este metodo obtiene los controles que le corresponde a cada persona y que seran
2113 incluidas en la sincronizacion
2114 *
2115 * @param string $catalog Nombre de la tabla
2116 * @param strin $array ids de personas
2117 * @param string $l1 offset para el limit
2118 * @param strin $l2 count para el limit
2119 *
2120 * @return result()
2121 */
2122 public function get_cns_cat_persona($catalog, $array, $l1="" , $l2="")
2123 {
2124     if($catalog=="tes_notificacion")
2125         $this-> db-> select('id,titulo,contenido,fecha_inicio,fecha_fin');
2126     else if($catalog=="asu_arbol_segmentacion")
2127         $this-> db-> select('id,grado_segmentacion,id_padre,orden, visible,
2128 descripcion');
2129     else if($catalog=="tes_pendientes_tarjeta")
2130         $this-> db-> select('fecha, id_persona, tabla, registro_json,');
2131     else
2132         $this-> db-> select('*');
2133     $this-> db-> from($catalog);
2134     $this-> db-> where_in("id_persona" , $array);
2135     if($l1!=" " || $l2!=" ")
2136         $this-> db-> limit($l2, $l1);
2137     $query = $this-> db-> get();
2138     if (!$query)
2139     {
2140         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
2141         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
2142         > _error_number().': '.$this-> db-> _error_message();
2143         throw new Exception("(" . __METHOD__ . ") => " . $this-> db-
2144         > _error_number().': '.$this-> db-> _error_message());
2145     }
2146     else
2147         return $query-> result();
2148     return null;
2149 }
2150 /**
2151 * Hace el count de personas que se envian en la sincronizacion
2152 *
2153 * @param string $catalog Nombre de la tabla
2154 * @param strin $personas personas que cumplen el requisito
2155 *
2156 * @return result()
2157 */
2158 public function get_cns_cat_persona_count($catalog,$personas)
2159 {
2160     $this-> db-> select ( 'COUNT(*) AS numrows' );
2161     $this-> db-> from($catalog);
2162     $this-> db-> where_in("id_persona" , $personas);
2163     $query = $this-> db-> get();
2164     return $query-> row()-> numrows;
2165 }
2166 /**

```

```

2168 *
2169 * obtiene los tutores de las personas que se envian en la sincronizacion
2170 *
2171 * @param string $array tutores que tienen asignado un paciente
2172 *
2173 * @return result()
2174 *
2175 */
2176 public function get_persona_x_tutor($array)
2177 {
2178     $this-> db-> distinct('*');
2179     $this-> db-> from("cns_tutor" );
2180     $this-> db-> where_in("id" , $array);
2181     $query = $this-> db-> get();
2182     if (!$query)
2183     {
2184         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
2185         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number(). ': ' . $this-> db-> _error_message();
2186         throw new Exception("(" . __METHOD__ . ") => " . $this-> db-
> _error_number(). ': ' . $this-> db-> _error_message());
2187     }
2188     else
2189         return $query-> result();
2190     return null;
2191 }
2192 /**
2193 *
2194 * obtiene los catalogos relevante x entorno para la sincronizacion
2195 *
2196 * @return result()
2197 *
2198 */
2199 public function get_catalog_relevante($fecha= "") )
2200 {
2201     $this-> db-> select('*');
2202     $this-> db-> from('cns_catalogo_relevante_x_entorno r');
2203     $this-> db-> join('cns_tabla_catalogo c' , 'c.id = r.id_tabla_catalogo','left');
2204     $this-> db-> where('fecha_actualizacion >=' , $fecha);
2205     $query = $this-> db-> get();
2206     if (!$query)
2207     {
2208         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
2209         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number(). ': ' . $this-> db-> _error_message();
2210         throw new Exception("(" . __METHOD__ . ") => " . $this-> db-
> _error_number(). ': ' . $this-> db-> _error_message());
2211     }
2212     else
2213         return $query-> result();
2214     return null;
2215 }
2216 /**
2217 *
2218 * Obtiene las transacciones relevante para la sincronizacion
2219 *
2220 *
2221 *
2222 * @return result()
2223 *
2224 */
2225 public function get_transaction_relevante()
2226 {
2227     $this-> db-> select('*');
2228     $this-> db-> from('cns_transaccion_relevante_x_entorno r');
2229     $this-> db-> join('cns_tabla_transaccion c' , 'c.id =
r.id_tabla_transaccion','left');
2230     $query = $this-> db-> get();
2231     if (!$query)
2232     {
2233         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
2234         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number(). ': ' . $this-> db-> _error_message();
2235         throw new Exception("(" . __METHOD__ . ") => " . $this-> db-
> _error_number(). ': ' . $this-> db-> _error_message());
2236     }
2237     else
2238         return $query-> result();
2239     return null;
2240 }

```

```

2241
2242 	/**
2243 	*
2244 	* obtiene cual es la ultima version de apk de la tableta
2245 	*
2246 	* @return          result()
2247 	*
2248 	*/
2249 public function get_version()
2250 {
2251     $this-> db-> select('host');
2252     $this-> db-> select_max('version');
2253     $this-> db-> from('tes_version');
2254     $query = $this-> db-> get();
2255     if (!$query)
2256     {
2257         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
2258         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().':'. $this-> db-> _error_message();
2259         throw new Exception("(" . __METHOD__ . ") => " . $this-> db-
> _error_number().':'. $this-> db-> _error_message());
2260     }
2261     else
2262         return $query-> result();
2263     return null;
2264 }
2265 /**
2266 *
2267 * obtiene informacion del tutor
2268 *
2269 * @param      string      $curp      Curp del tutor
2270 *
2271 * @return     result()
2272 *
2273 */
2274 public function data_tutor($curp)
2275 {
2276     $query = $this-> db-> get_where('cns_tutor', array('curp' => $curp));
2277     if (!$query)
2278     {
2279         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
2280         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().':'. $this-> db-> _error_message();
2281         throw new Exception("(" . __METHOD__ . ") => " . $this-> db-
> _error_number().':'. $this-> db-> _error_message());
2282     }
2283     else
2284         return $query-> result();
2285     return null;
2286 }
2287 /**
2288 *
2289 * obtiene informacion del tutor para generar el autocomplete
2290 *
2291 * @param      string      $keywords  palabras claves para hacer el filtro
2292 *
2293 * @return     result()
2294 *
2295 */
2296 public function autocomplete_tutor($keywords)
2297 {
2298     $this-> db-> select('*');
2299     $this-> db-> from('cns_tutor');
2300     $this-> db-> like('curp', $keywords);
2301     $this-> db-> or_like('nombre', $keywords);
2302     $this-> db-> or_like('apellido_paterno', $keywords);
2303     $this-> db-> or_like('apellido_materno', $keywords);
2304     $this-> db-> or_like('CONCAT(nombre, ",apellido_paterno, "
2305     ",apellido_materno)', $keywords);
2306     $query = $this-> db-> get();
2307     if (!$query)
2308     {
2309         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
2310         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().':'. $this-> db-> _error_message();
2311         throw new Exception("(" . __METHOD__ . ") => " . $this-> db-
> _error_number().':'. $this-> db-> _error_message());
2312     }
2313     else

```

```

2314         return $query-> result();
2315     return null;
2316 }
2317 /**
2318 *
2319 * valida que no se repita la curp en personas y tutor
2320 *
2321 * @param      string      $curp      Curp a validar
2322 * @param      strin       $tabla     Tabla en la que se debe hacer la validacion
2323 * @param      string      $id        id de la persona o tutor para excluirse
2324 *
2325 * @return     result()
2326 *
2327 */
2328 public function getByCurp($curp,$tabla,$id)
2329 {
2330     if($id!="")
2331         $query = $this-> db-> get_where($tabla, array('curp' => $curp,"id" => $id));
2332     else
2333         $query = $this-> db-> get_where($tabla, array('curp' => $curp));
2334     if (!$query){
2335         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
2336         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
2337 > db-> _error_number().':'. $this-> db-> _error_message(); . $this-> db-
2338 > _error_number().':'. $this-> db-> _error_message(); . $this-> db-
2339     }
2340     else
2341         return $query-> row();
2342     return;
2343 }
2344 /**
2345 * Elimina los pendientes de las personas que no tengan asignado una tarjeta
2346 *
2347 * @return     result()
2348 *
2349 */
2350 public function tes_pendientes_tarjeta_delete()
2351 {
2352     $query = $this-> db-> query("DELETE FROM tes_pendientes_tarjeta WHERE
2353 id_persona NOT IN(SELECT id_persona FROM tes_entorno_x_persona)" );
2354     if (!$query){
2355         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
2356         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
2357 > db-> _error_number().':'. $this-> db-> _error_message(); . $this-> db-
2358 > _error_number().':'. $this-> db-> _error_message(); . $this-> db-
2359     }
2360 }
2361 /**
2362 *
2363 * devuelve todos los pacientes de la base de datos
2364 *
2365 * @return     result()
2366 *
2367 */
2368 public function get_pacientes()
2369 {
2370     $query = $this-> db-> query("SELECT p.id, p.curp, p.nombre,
2371 p.apellido_paterno, p.apellido_materno, p.fecha_nacimiento, p.calle_domicilio, p.numero_domicilio,
2372 p.colonia_domicilio, p.referencia_domicilio, p.cp_domicilio,
2373 CONCAT(t.nombre,' ',t.apellido_paterno,' ',t.apellido_materno) AS nombreT, t.curp AS curpT,
2374 a.descripcion AS lugar
2375 FROM cns_persona p
2376 LEFT JOIN cns_persona_x_tutor pt ON pt.id_persona=p.id
2377 LEFT JOIN cns_tutor t ON t.id=pt.id_tutor
2378 LEFT JOIN asu_arbol_segmentacion a ON a.id=p.id_asu_localidad_nacimiento" );
2379     if (!$query){
2380         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
2381         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
2382 > db-> _error_number().':'. $this-> db-> _error_message(); . $this-> db-
2383 > _error_number().':'. $this-> db-> _error_message(); . $this-> db-
2384     }
2385     else

```

```

2383     return $query-> result();
2384 }
2385
2386 public function getMsgError($value = 'usr')
2387 {
2388     if ($value == 'log')
2389         return $this-> msg_error_log;
2390     return $this-> msg_error_usr;
2391 }
2392
2393 /**
2394 * Obtiene los datos específicos de un catálogo para ser visualizados en una gráfica
2395 *
2396 * @access public
2397 * @param int $catalogo Determina el catalogo a consultar
2398 * @param int $sexo El sexo del paciente (F, M)
2399 * @param int $edad_meses Edad del paciente en meses
2400 * @param int $id_persona Identificador del paciente
2401 * @param int $asu_locali Identificador del asu de la localidad del domicilio
2402 * @return void
2403 */
2404 public function get_datos_grafica($catalogo, $sexo, $edad_meses, $id_persona,
$asu_locali='')
2405 {
2406     $datos = array('series' => NULL,
2407                   'labels' => NULL
2408                   );
2409     $stalla = 45; // Talla ideal al nacimiento
2410     $series = NULL;
2411     $puntos = NULL;
2412
2413     // Obtiene datos del paciente
2414     $queryPaciente = 'SELECT
2415                     TIMESTAMPDIFF(MONTH, fecha_nacimiento, fecha) AS edad_meses,
2416                     peso, altura, hemoglobina,
2417                     ROUND((peso/POW((altura/100),2)), 1) AS imc
2418                     FROM cns_control_nutricional
2419                     INNER JOIN cns_persona ON cns_persona.id = cns_control_nutricional.id_persona
2420                     WHERE cns_control_nutricional.id_persona=' . $id_persona . '
2421                     ORDER BY fecha ASC';
2422
2423     $objQueryPac = $this-> db-> query($queryPaciente);
2424     $objResultPac = $objQueryPac-> result();
2425
2426     if($this-> db-> _error_number()) {
2427         $this-> error = true;
2428         $this-> msg_error_usr = 'Error al obtener los datos para las gráficas';
2429         $this-> msg_error_log = '(' . __METHOD__ . ') => ' . $this-> db-
2430 > _error_number() . ': ' . $this-> db-> _error_message();
2431         throw new Exception($this-> msg_error_log);
2432     }
2433
2434     switch($catalogo){
2435         case 'peso_edad':
2436             foreach ($objResultPac as $pac) {
2437                 $puntos[] = array($pac-> edad_meses, $pac-> peso);
2438             }
2439             break;
2440         case 'peso_talla':
2441             foreach ($objResultPac as $pac) {
2442                 $puntos[] = array($pac-> altura, $pac-> peso);
2443
2444                 if($pac-> altura > $stalla){
2445                     $stalla = $pac-> altura;
2446                 }
2447             }
2448             break;
2449         case 'talla_edad':
2450             foreach ($objResultPac as $pac) {
2451                 $puntos[] = array($pac-> edad_meses, $pac-> altura);
2452             }
2453             break;
2454         case 'imc':
2455             foreach ($objResultPac as $pac) {
2456                 $puntos[] = array($pac-> edad_meses, $pac-> imc);
2457             }
2458             break;
2459         case 'peri_cefa':
2460             $queryPaciente = 'SELECT
TIMESTAMPDIFF(MONTH, fecha_nacimiento, fecha) AS edad_meses,

```

```

2461             perimetro_cefalico
2462             FROM cns_control_peri_cefa
2463             INNER JOIN cns_persona ON cns_persona.id = cns_control_peri_cefa.id_persona
2464             WHERE cns_control_peri_cefa.id_persona= '".$id_persona."'"
2465             ORDER BY fecha ASC';
2466
2467             $objQueryPac = $this-> db-> query($queryPaciente);
2468             $objResultPac = $objQueryPac-> result();
2469
2470             if($this-> db-> _error_number()) {
2471                 $this-> error = true;
2472                 $this-> msg_error_usr = 'Error al obtener los datos para las gráficas';
2473                 $this-> msg_error_log = ('.'.__METHOD__.') => ' . $this-> db-
2474 > _error_number().' : '.$this-> db-> _error_message();
2475                 throw new Exception($this-> msg_error_log);
2476             }
2477
2478             foreach ($objResultPac as $pac) {
2479                 $puntos[] = array($pac-> edad_meses, $pac-> perimetro_cefalico);
2480             }
2481             break;
2482         case 'con_hemo':
2483             foreach ($objResultPac as $pac) {
2484                 if($pac-> hemoglobina) {
2485                     $puntos[] = array($pac-> edad_meses, $pac-> hemoglobina);
2486                 }
2487             }
2488
2489             $series[] = array(
2490                 'color' => 'black',
2491                 'label' => ' &nbsp; Paciente',
2492                 'data' => $puntos,
2493             );
2494
2495             $puntos = NULL;
2496
2497             // Obtiene datos de los catálogos
2498             if ($catalogo == 'con_hemo') {
2499                 $objQueryDat = $this-> db-> query('SELECT mujer_embarazada_ninio_6_59_meses AS
hb FROM asu_hemoglobina_altitud WHERE id_localidad_asu='.$asu_locali);
2500
2501                 if ($objQueryDat-> num_rows() > 0) {
2502                     $objResultDat = $objQueryDat-> row();
2503
2504                     if($this-> db-> _error_number()) {
2505                         $this-> error = true;
2506                         $this-> msg_error_usr = 'Error al obtener los datos para las gráficas';
2507                         $this-> msg_error_log = ('.'.__METHOD__.') => ' . $this-> db-
2508 > _error_number().' : '.$this-> db-> _error_message();
2509                         throw new Exception($this-> msg_error_log);
2510                     }
2511
2512                     $puntos[] = array(0, $objResultDat-> hb);
2513                     $puntos[] = array((($edad_meses+3)), $objResultDat-> hb);
2514
2515                     $series[] = array(
2516                         'color' => 'blue',
2517                         'label' => ' &nbsp; Concentración de Hemoglobina',
2518                         'data' => $puntos,
2519                     );
2520
2521                     $datos['labels'] = array('xaxes'=> 'Edad (meses)', 'yaxes'=> 'Hb (g/dL)');
2522                     $datos['series'] = $series;
2523                 } else {
2524                     switch($catalogo){
2525                         case 'peso_edad':
2526                             $queryCatalogo = 'SELECT id, descripcion, color FROM
cns_estado_nutricion_peso';
2527                             $queryDatos = 'SELECT edad_meses AS x, peso AS y FROM
cns_edo_nutri_peso_x_edad WHERE sexo="'.$sexo.'" AND edad_meses<=' .($edad_meses+3).' AND
id_estado_nutricion_peso=';
2528                             $datos['labels'] = array('xaxes'=> 'Edad (meses)', 'yaxes'=> 'Peso
(Kg)');
2529                             break;
2530                         case 'peso_talla':
2531                             $queryCatalogo = 'SELECT id, descripcion, color FROM
cns_estado_nutricion_peso';
2532                             $queryDatos = 'SELECT altura AS x, peso AS y FROM

```

```

cns_edo_nutri_peso_x_altura WHERE sexo=" " . $sexo . " AND altura<=" . ($talla+10) . ' AND
id_estado_nutricion_peso=' ;
2533                                     $datos[ 'labels' ] = array('xaxes'=> 'Talla (cm)', 'yaxes'=> 'Peso (Kg)');
2534                                     break;
2535                                     case 'talla_edad':
2536                                         $queryCatalogo = 'SELECT id, descripcion, color FROM
cns_estado_nutricion_altura';
2537                                         $queryDatos = 'SELECT edad_meses AS x, altura AS y FROM
cns_edo_nutri_altura_x_edad WHERE sexo=" " . $sexo . " AND edad_meses<=" . ($edad_meses+3) . '
AND id_estado_nutricion_altura=' ;
2538                                         $datos[ 'labels' ] = array('xaxes'=> 'Edad (meses)', 'yaxes'=> 'Talla
(cm)');
2539                                         break;
2540                                         case 'imc':
2541                                             $queryCatalogo = 'SELECT id, descripcion, color FROM cns_estado_imc';
2542                                             $queryDatos = 'SELECT edad_meses AS x, imc AS y FROM cns_imc_x_edad WHERE
sexo=" " . $sexo . " AND edad_meses<=" . ($edad_meses+3) . ' AND id_estado_imc=' ;
2543                                             $datos[ 'labels' ] = array('xaxes'=> 'Edad (meses)', 'yaxes'=> 'IMC
(Kg/m2)');
2544                                             break;
2545                                         case 'peri_cefa':
2546                                             $queryCatalogo = 'SELECT id, descripcion, color FROM cns_estado_peri_cefa';
2547                                             $queryDatos = 'SELECT edad_meses AS x, perimetro AS y FROM
cns_perimetro_cefalico WHERE sexo=" " . $sexo . " AND edad_meses<=" . ($edad_meses+3) . ' AND
id_estado_peri_cefa=' ;
2548                                             $datos[ 'labels' ] = array('xaxes'=> 'Edad (meses)', 'yaxes'=> 'Perímetro
cefálico (cm)');
2549                                         }
2550
2551             $objQueryCat = $this-> db-> query($queryCatalogo);
2552             $objResultCat = $objQueryCat-> result();
2553
2554             if($this-> db-> _error_number()) {
2555                 $this-> error = true;
2556                 $this-> msg_error_usr = 'Error al obtener los datos para las gráficas';
2557                 $this-> msg_error_log = '('. __METHOD__ . ') => ' . $this-> db-
> _error_number(). ': ' . $this-> db-> _error_message();
2558                 throw new Exception($this-> msg_error_log);
2559             }
2560
2561             foreach ($objResultCat as $cat) {
2562                 $objQueryDat = $this-> db-> query($queryDatos.$cat-> id);
2563                 $objResultDat = $objQueryDat-> result();
2564
2565                 if($this-> db-> _error_number()) {
2566                     $this-> error = true;
2567                     $this-> msg_error_usr = 'Error al obtener los datos para las gráficas';
2568                     $this-> msg_error_log = '('. __METHOD__ . ') => ' . $this-> db-
> _error_number(). ': ' . $this-> db-> _error_message();
2569                     throw new Exception($this-> msg_error_log);
2570                 }
2571
2572                 $puntos = NULL;
2573
2574                 foreach ($objResultDat as $dat) {
2575                     $puntos[] = array($dat-> x, $dat-> y);
2576                 }
2577
2578                 $series[] = array(
2579                     'color' => $cat-> color,
2580                     'label' => ' ' . $cat-> descripcion,
2581                     'data' => $puntos,
2582                 );
2583             }
2584
2585             $datos[ 'series' ] = $series;
2586         }
2587
2588         return $datos;
2589     }
2590
2591 /**
2592 *
2593 * Obtiene los registros de perimetro cefalico asociados a una persona
2594 *
2595 * @param string $id identificador de la persona
2596 * @param strin $order nombre del campo para hacer el order by
2597 *
2598 * @return result()
2599 */

```

```

2600 */
2601 public function get_peri_cefa($id,$order="" )
2602 {
2603     $this-> db-> select('*');
2604     $this-> db-> from('cns_control_peri_cefa');
2605     $this-> db-> where('id_persona', $id);
2606     if($order!="")
2607         $this-> db-> order_by($order, "asc" );
2608     else
2609         $this-> db-> order_by("fecha" , "ASC" );
2610     $query = $this-> db-> get();
2611     if (!$query)
2612     {
2613         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
2614         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
2615         throw new Exception("(" . __METHOD__ . ") => " . $this-> db-
> _error_number().': '.$this-> db-> _error_message());
2616     }
2617     else
2618         return $query-> result();
2619     return null;
2620 }
2621 /**
2622 *
2623 * Actualiza los registros de perimetro cefalico
2624 *
2625 * @return      result()
2626 *
2627 */
2628 public function update_peri_cefa()
2629 {
2630     $id_asu_um = $this-> umt;
2631     if ($this-> db-> delete('cns_control_peri_cefa', array('id_persona' => $this-
> id))) {
2632         for($index=0; $index< sizeof($this-> peri_cefa); $index++){
2633             $datosPeriCefa = array(
2634                 'id_persona' => $this-> id,
2635                 'fecha' => date('Y-m-d H:i:s', strtotime($this-
> fecha_peri_cefa[$index])),
2636                 'perimetro_cefalico' => $this-> peri_cefa[$index],
2637                 'id_asu_um' => $id_asu_um,
2638             );
2639             $resultPeriCefa = $this-> db-> insert('cns_control_peri_cefa',
2640 $datosPeriCefa);
2641             if (!$resultPeriCefa)
2642             {
2643                 $this-> msg_error_usr = "Error Perímetro Cefálico." ;
2644                 $this-> msg_error_log = "(" . __METHOD__ . ") => " .
$this-> db-> _error_number().': '.$this-> db-> _error_message();
2645                 throw new Exception("(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message());
2646             }
2647         }
2648     }
2649 }
2650 /**
2651 *
2652 * Obtiene los registros de estimulacion temprana asociados a una persona
2653 *
2654 * @param      string      $id      identificador de la persona
2655 * @param      strin       $order   nombre del campo para hacer el order by
2656 *
2657 * @return      result()
2658 *
2659 */
2660 public function get_estimulacion($id,$order="" )
2661 {
2662     $this-> db-> select('*');
2663     $this-> db-> from('cns_estimulacion_temprana');
2664     $this-> db-> where('id_persona', $id);
2665     if($order!="")
2666         $this-> db-> order_by($order, "asc" );
2667     else
2668         $this-> db-> order_by("fecha" , "ASC" );
2669     $query = $this-> db-> get();
2670     if (!$query)
2671     {

```

```

2673         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
2674         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
2675         throw new Exception("(" . __METHOD__ . ") => " . $this-> db-
> _error_number().': '.$this-> db-> _error_message());
2676     }
2677     else
2678         return $query-> result();
2679     return null;
2680 }
2681 /**
2682 *
2683 * Actualiza los registros de estimulacion temprana
2684 *
2685 * @return      result()
2686 *
2687 */
2688 public function update_estimulacion()
2689 {
2690     $id_asu_um = $this-> umt;
2691     if ($this-> db-> delete('cns_estimulacion_temprana', array('id_persona' => $this-
> id))) {
2692         for($index=0; $index< sizeof($this-> estimulacion_fecha); $index++){
2693             $datosEstimulacion = array(
2694                 'id_persona' => $this-> id,
2695                 'fecha' => date('Y-m-d H:i:s', strtotime($this-
> estimulacion_fecha[$index])),
2696                 'tutor_capacitado' => $this-> estimulacion_capacitado[$index],
2697                 'id_asu_um' => $id_asu_um);
2698             $resultEstimulacion = $this-> db-> insert('cns_estimulacion_temprana',
2699 $datosEstimulacion);
2700             if (!$resultEstimulacion)
2701             {
2702                 $this-> msg_error_usr = "Error Estimulación Temprana." ;
2703                 $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
2704                 throw new Exception("(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message());
2705             }
2706         }
2707     }
2708 }
2709 /**
2710 * Obtiene el listado de categorias de CIE10
2711 *
2712 * @access public
2713 * @return object/booleanDevuelve el objeto con sus datos correspondientes, de lo
2714 contrario, false Si no se encontró el registro
2715 */
2716 public function getCategoriacIE10()
2717 {
2718     $result = false;
2719     $this-> db-> select('*');
2720     $this-> db-> from('cns_categoria_ciel0');
2721     $this-> db-> order_by('descripcion', 'ASC');
2722     $query = $this-> db-> get();
2723     $result = $query-> result();
2724
2725     if($this-> db-> _error_number()) {
2726         $this-> error = true;
2727         $this-> msg_error_usr = 'No se encontraron registros en la busqueda';
2728         $this-> msg_error_log = '(' . __METHOD__ . ') => ' . $this-> db-
> _error_number().': '.$this-> db-> _error_message();
2729         throw new Exception();
2730     } else if(empty($result)) {
2731         $this-> msg_error_usr = 'No se encontraron registros en la busqueda';
2732     }
2733
2734     return $result;
2735 }
2736 /**
2737 * Obtiene el listado de CIE10 correspondientes a una CIE10
2738 *
2739 * @access public

```

```

2744     * @return object/booleanDevuelve el objeto con sus datos correspondientes, de lo
2745     * contrario, false Si no se encontró el registro
2746     */
2747     public function getCIE10($categoria)
2748     {
2749         $result = false;
2750
2751         $this-> db-> select('*');
2752         $this-> db-> from('cns_ciel0');
2753         $this-> db-> where('id_categoria' , urldecode($categoria));
2754         $this-> db-> order_by('descripcion' , 'ASC');
2755
2756         $query = $this-> db-> get();
2757         $result = $query-> result();
2758
2759         if($this-> db-> _error_number()) {
2760             $this-> error = true;
2761             $this-> msg_error_usr = 'No se encontraron registros en la búsqueda';
2762             $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
2763 > _error_number().': '.$this-> db-> _error_message();
2764             throw new Exception();
2765         } else if(empty($result)) {
2766             $this-> msg_error_usr = 'No se encontraron registros en la búsqueda';
2767         }
2768
2769         return $result;
2770     }
2771
2772 /**
2773 * Obtiene todas las consultas asociadas a un paciente
2774 *
2775 * @access public
2776 * @return object/booleanDevuelve el objeto con sus datos correspondientes, de lo
2777 * contrario, false Si no se encontró el registro
2778 */
2779 public function getControlConsultas($idPersona)
2780 {
2781     $result = false;
2782
2783     $this-> db-> select('*');
2784     $this-> db-> from('cns_control_consulta');
2785     $this-> db-> where('id_persona' , $idPersona);
2786     $this-> db-> order_by('fecha' , 'ASC');
2787
2788     $query = $this-> db-> get();
2789     $result = $query-> result();
2790
2791     if($this-> db-> _error_number()) {
2792         $this-> error = true;
2793         $this-> msg_error_usr = 'No se encontraron registros en la búsqueda';
2794         $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
2795 > _error_number().': '.$this-> db-> _error_message();
2796         throw new Exception();
2797     } else if(empty($result)) {
2798         $this-> msg_error_usr = 'No se encontraron registros en la búsqueda';
2799     } else {
2800         foreach ($result as $idxConsulta => $consulta) {
2801             $this-> db-> select('descripcion');
2802             $this-> db-> from('cns_ciel0');
2803             $this-> db-> where('id_ciel0' , $consulta-> clave_ciel0);
2804
2805             $query = $this-> db-> get();
2806             $cie10 = $query-> row();
2807
2808             $result[$idxConsulta]-> descripCIE10 = $cie10-> descripcion;
2809
2810             $medicamentos = '';
2811             $idsMedicamentos = explode(' ' , $result[$idxConsulta]-> id_tratamiento);
2812
2813             foreach ($idsMedicamentos as $idMed) {
2814                 $this-> db-> select('descripcion');
2815                 $this-> db-> from('cns_tratamiento');
2816                 $this-> db-> where('id' , $idMed);
2817
2818                 $query = $this-> db-> get();
2819                 $med = $query-> row();
2820
2821                 $medicamentos .= $med-> descripcion.' ';
2822             }
2823         }
2824     }
2825 }

```

```

2820             $result[$idxConsulta]-> descripTratamiento = substr($medicamentos, 0, -2);
2821         }
2822     }
2823 }
2824 return $result;
2825 }
2826 /**
2827 *
2828 * Actualiza los registros de sales de rehidratacion oral
2829 *
2830 * @return      result()
2831 *
2832 */
2833 public function update_sales()
2834 {
2835     $id_asu_um = $this-> umt;
2836     if ($this-> db-> delete('cns_sales_rehidratacion', array('id_persona' => $this-
2837 > id))) {
2838         for($index=0; $index< sizeof($this-> sales_fecha); $index++){
2839             $datosSRO = array(
2840                 'id_persona' => $this-> id,
2841                 'fecha' => date('Y-m-d H:i:s', strtotime($this-> sales_fecha[$index])),
2842                 'cantidad' => $this-> sales_cantidad[$index],
2843                 'id_asu_um' => $id_asu_um);
2844
2845             $resultSRO = $this-> db-> insert('cns_sales_rehidratacion', $datosSRO);
2846             if (!$resultSRO)
2847             {
2848                 $this-> msg_error_usr = "Error Sales de Rehidratación Oral." ;
2849                 $this-> msg_error_log = "(" . __METHOD__ . ") => "
2850 . $this-> db-> _error_number().':'. $this-> db-> _error_message();
2851             throw new Exception("(" . __METHOD__ . ") => " . $this-
2852 > db-> _error_number().':'. $this-> db-> _error_message());
2853         }
2854     }
2855 }
2856 /**
2857 *
2858 * Obtiene los registros de sales de rehidratacion oral asociados a una persona
2859 *
2860 * @param      string      $id      identificador de la persona
2861 * @param      strin       $order   nombre del campo para hacer el order by
2862 *
2863 * @return      result()
2864 *
2865 */
2866 public function get_sales($id,$order="")
2867 {
2868     $this-> db-> select('*');
2869     $this-> db-> from('cns_sales_rehidratacion');
2870     $this-> db-> where('id_persona', $id);
2871     if($order!="")
2872     $this-> db-> order_by($order, "asc" );
2873     else
2874     $this-> db-> order_by("fecha" , "ASC" );
2875     $query = $this-> db-> get();
2876     if (!$query)
2877     {
2878         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
2879         $this-> msg_error_log = "(" . __METHOD__ . ") => "
2880 . $this-> db-> _error_number().':'. $this-> db-> _error_message();
2881     throw new Exception("(" . __METHOD__ . ") => " . $this-> db-
2882 > _error_number().':'. $this-> db-> _error_message());
2883     }
2884     else
2885     return $query-> result();
2886     return null;
2887 }
2888 ?>

```

# Archivo fuente para estado\_tableta\_model.php

*La documentación para este archivo está disponible en [estado\\_tableta\\_model.php](#)*

```
1  <?php
2
3  /**
4   * Modelo Estado_tableta
5   *
6   * @package    TES
7   * @subpackage Modelo
8   * @author     Pascual
9   * @created    2013-11-26
10  */
11 class Estado_tableta_model extends CI_Model
12 {
13     /**
14      * @access private
15      * @var int(11)
16      */
17     private $id;
18
19     /**
20      * @access private
21      * @var varchar(20)
22      */
23     private $descripcion;
24
25
26     ****
27     * Estas variables no pertenecen a la tabla *
28     ****
29
30     /**
31      * @access private
32      * @var boolean
33      */
34     private $error;
35
36     /**
37      * @access private
38      * @var string
39      */
40     private $msg_error_usr;
41
42     /**
43      * @access private
44      * @var string
45      */
46     private $msg_error_log;
47
48
49     public function __construct()
50     {
51         parent::__construct();
52         $this->load->database();
53         $this->error = false;
54         $this->msg_error_usr = '';
55         $this->msg_error_log = '';
56
57         if( !$this->db->conn_id ) {
58             throw new Exception ('ERROR: No se puede conectar con la Base de Datos');
59         }
60     }
61
62     public function getId()
63     {
64         return $this->id;
```

```

65
66
67     public function getDescripcion() {
68         return $this-> descripcion;
69     }
70
71     public function setId($id)
72     {
73         return $this-> id = $id;
74     }
75
76     public function setDescripcion($descripcion) {
77         $this-> mac = $descripcion;
78     }
79
80     /**
81      * Devuelve el mensaje de error,
82      * en caso de existir un error despues de ejecutar un metodo,
83      * de lo contrario false
84      *
85      * @access public
86      * @param string $type usr si se quiere devolver el mensaje de error a mostrar en la vista,
87      *                      log obtiene el mensaje de error con mas detalles para depuración,
88      *                      valor por defecto usr
89      * @return boolean/string
90      */
91     public function getMsgError($type = 'usr')
92     {
93         if($this-> error) {
94             if($type == 'usr')
95                 return $this-> msg_error_usr;
96             else if($type == 'log')
97                 return $this-> msg_error_log;
98             else
99                 return false;
100        }
101
102        return false;
103    }
104
105    /**
106     * Obtiene los datos del registro que tiene el ID especificado
107     *
108     * @access public
109     * @param int $id           Si no se establece el valor de ID, se toma el valor del objeto
actual
110     * @return object/booleanDevuelve el objeto con sus datos correspondientes, de lo
contrario, false Si no se encontró el registro
111     */
112    public function getById($id)
113    {
114        $result = false;
115
116        $query = $this-> db-> get_where('sis_estado_tableta', array('id' => $id));
117        $result = $query-> row();
118
119        if($this-> db-> _error_number()) {
120            $this-> error = true;
121            $this-> msg_error_usr = 'No se encontraron registros en la búsqueda';
122            $this-> msg_error_log = '(' . __METHOD__ . ') => ' . $this-> db-
> _error_number() . ': ' . $this-> db-> _error_message();
123            throw new Exception();
124        }
125
126        if(!empty($result)) {
127            $this-> id = $id;
128            $this-> descripcion = $result-> descripcion;
129        }
130
131        return $result;
132    }
133
134    /**
135     * Obtiene todos los registros de la tabla
136     *
137     * @access public
138     * @return array object   Devuelve un arreglo de objetos obtenidos de la base de datos
139     */
140    public function getAll()
141    {

```

```

142     $result = 0;
143
144     $this-> db-> where('id >', 4);
145     $query = $this-> db-> get('sis_estado_tableta');
146     $result = $query-> result();
147
148     if($this-> db-> _error_number()) {
149         $this-> error = true;
150         $this-> msg_error_usr = 'No se encontraron registros en la busqueda';
151         $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
> _error_number(). ': ' . $this-> db-> _error_message();
152         throw new Exception();
153     } else if(empty ($result)) {
154         $this-> msg_error_usr = 'No se encontraron registros en la busqueda';
155     }
156
157     return $result;
158 }
159
160 ?>

```

# Archivo fuente para notificacion\_model.php

La documentación para este archivo está disponible en [notificacion\\_model.php](#)

```
1  <?php
2  /**
3   * Modelo Usuario
4   *
5   * @package      TES
6   * @subpackage   Modelo
7   * @author       Rogelio
8   * @created      2013-11-26
9   */
10  class Notificacion_model extends CI_Model {
11      /**
12      * @access private
13      * @var int
14      */
15      private $id;
16      /**
17      * @access private
18      * @var string
19      */
20      private $titulo;
21      /**
22      * @access private
23      * @var string
24      */
25      private $contenido;
26      /**
27      * @access private
28      * @var datetime
29      */
30      private $fecha_inicio;
31      /**
32      * @access private
33      * @var datetime
34      */
35      private $fecha_fin;
36      /**
37      * @access private
38      * @var string
39      */
40      private $id_arr_asu;
41
42      /**
43      * Estas variables no pertenecen a la tabla
44      */
45
46      /**
47      * @access private
48      * @var string
49      */
50      private $msg_error_usr;
51      /**
52      * @access private
53      * @var string
54      */
55      private $msg_error_log;
56      /**
57      * @access private
58      * @var array
59      */
60      private $tabletas;
61
62      /**
63      * @access private
64      * @var array
```

```

65      */
66      private $filters;
67
68      /**
69      * @access private
70      * @var array
71      */
72      private $filtersOr;
73
74      public function __construct()
75      {
76          parent::__construct();
77
78          $this-> filters = array();
79          $this-> filtersOr = array();
80
81          $this-> load-> database();
82          if (!$this-> db-> conn_id)
83              throw new Exception("No se pudo conectar a la base de datos");
84      }
85
86      public function getId()
87      {
88          return $this-> id;
89      }
90
91      public function setId($value) {
92          $this-> id = $value;
93      }
94
95      public function getTitulo()
96      {
97          return $this-> titulo;
98      }
99
100     public function setTitulo($titulo)
101     {
102         $this-> titulo = $titulo;
103     }
104
105     public function getContenido()
106     {
107         return $this-> contenido;
108     }
109
110     public function setContenido($contenido)
111     {
112         $this-> contenido = $contenido;
113     }
114
115     public function getFechaInicio()
116     {
117         return $this-> fecha_inicio;
118     }
119
120     public function setFechaInicio($fecha_inicio)
121     {
122         $this-> fecha_inicio = $fecha_inicio;
123     }
124
125     public function getFechaFin()
126     {
127         return $this-> fecha_fin;
128     }
129
130     public function setFechaFin($fecha_fin)
131     {
132         $this-> fecha_fin = $fecha_fin;
133     }
134
135     public function getIdsTabletas()
136     {
137         return $this-> id_arr_asu;
138     }
139
140     public function setIdsTabletas($id_arr_asu)
141     {
142         $this-> id_arr_asu = $id_arr_asu;
143     }
144

```

```

145     /**
146      * Asigna el mensaje de error a visualizar: para usuario final (usr) o para bitácora (log)
147      *
148      * @access      public
149      * @param       string      $value      tipo de error a visualizar: usr o log,
150      * @return      boolean      false      si ocurrió algún error, true si se ejecutó
151      * correctamente
152      */
153     public function getMsgError($value = 'usr')
154     {
155         if ($value == 'log')
156             return $this-> msg_error_log;
157         return $this-> msg_error_usr;
158     }
159     /**
160      * Obtiene todas las notificaciones existentes, se puede filtrar por: texto a buscar si se
161      * desea
162      * @access      public
163      * @param       boolean/string    $keywords      false no hay texto a buscar/string con
164      * texto a buscar
165      * @param       int            $offset      Establece el desplazamiento del
166      * primer registro a devolver,
167      * @param       int            $row_count    Establece la cantidad de registros a
168      * devolver
169      * @return      void/arrayobject
170      * object si se ejecutó correctamente
171     */
172     public function getAll($keywords = '', $offset = null, $row_count = null)
173     {
174         if(!empty($offset) && !empty($row_count))
175             $this-> db-> limit($offset, $row_count);
176         else if (!empty($offset))
177             $this-> db-> limit($offset);
178         if (empty($keywords) && empty($this-> filters)){
179             $query = $this-> db-> get('tes_notificacion');
180             return $query-> result();
181         }
182         else
183         {
184             $this-> db-> select('*');
185             $this-> db-> from('tes_notificacion');
186             if (!empty($keywords)){
187                 $this-> db-> where(" (titulo LIKE '%$keywords%' OR contenido LIKE
188 '%$keywords%')");
189             }
190             if( !empty($this-> filters) ){
191                 $this-> db-> where($this-> filters);
192             }
193             $query = $this-> db-> get();
194             if (!$query){
195                 $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
196                 $this-> msg_error_log = "(" . __METHOD__ . ". " . $this->
197                 db-> _error_number() . ': ' . $this-> db-> _error_message());
198                 throw new Exception(__CLASS__);
199             }
200         }
201     /**
202      * Obtiene la notificación solicitada
203      *
204      * @access      public
205      * @param       int            $id          id de notificación
206      * @return      void/object
207      * false si ocurrió algún error, object si se ejecutó
208      * correctamente
209      */
210     public function getById($id)
211     {
212         $query = $this-> db-> get_where('tes_notificacion', array('id' => $id));
213         if (!$query){
214             $this-> msg_error_usr = "Servicio temporalmente no disponible." ;

```

```

214             $this-> msg_error_log = "(" . __METHOD__ . ")" => "
215 > db-> _error_number().': '.$this-> db-> _error_message();
216         }
217     }
218     else
219         return $query-> result();
220     return;
221 }
222 /**
223 * Obtiene el numero total de notificaciones
224 *
225 * @access public
226 * @param boolean/string      $keywords      false no hay texto a buscar/string con
227 texto a buscar
228 * @return int
229 */
230 public function getNumRows($keywords = '')
231 {
232     if (!$keywords)
233         $query = $this-> db-> get('tes_notificacion');
234     else
235     {
236         $this-> db-> select('*');
237         $this-> db-> from('tes_notificacion');
238         $this-> db-> like('titulo', $keywords);
239         $this-> db-> or_like('contenido', $keywords);
240         $query = $this-> db-> get();
241     }
242     if(!$query) {
243         $this-> msg_error_usr = "Servicio temporalmente no disponible.";
244         $this-> msg_error_log = '(' . __METHOD__ . ')' => '$this-> db-
> _error_number().': '.$this-> db-> _error_message();
245         throw new Exception(__CLASS__);
246     }
247     return $query-> num_rows;
248 }
249 /**
250 * Inserta en la base de datos los datos de la notificación (datos en propiedades)
251 *
252 * @access public
253 * @return boolean      false si ocurrió algún error, true si se
254 ejecutó correctamente
255 */
256 public function insert()
257 {
258     $data = array(
259         'titulo' => $this-> titulo,
260         'contenido' => $this-> contenido,
261         'fecha_inicio' => $this-> fecha_inicio,
262         'fecha_fin' => $this-> fecha_fin,
263         'id_arr_asu' => $this-> id_arr_asu
264     );
265     $result = $this-> db-> insert('tes_notificacion', $data);
266     if (!$result){
267         $this-> msg_error_usr = "Servicio temporalmente no disponible.";
268         $this-> msg_error_log = "(" . __METHOD__ . ")" => "
> db-> _error_number().': '.$this-> db-> _error_message();
269         throw new Exception(__CLASS__);
270     }
271     return $result;
272 }
273 /**
274 * Actualiza en la base de datos los datos de la notificación (datos en propiedades)
275 *
276 * @access public
277 * @return boolean      false si ocurrió algún error, true si se
278 ejecutó correctamente
279 */
280 public function update()
281 {
282     $data = array(
283         'titulo' => $this-> titulo,
284         'contenido' => $this-> contenido,
285         'fecha_inicio' => $this-> fecha_inicio,
286         'fecha_fin' => $this-> fecha_fin,
287         'id_arr_asu' => $this-> id_arr_asu
288     );

```

```

288     $this-> db-> where('id' , $this-> id);
289     $result = $this-> db-> update('tes_notificacion' , $data);
290     if (!$result){
291         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
292         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
293 > db-> _error_number()." : '$this-> db-> _error_message()';
294         throw new Exception(__CLASS__);
295     }
296     return $result;
297 }
298 /**
299 * Elimina de la base de datos la notificación (id en propiedades)
300 *
301 * @access public
302 * @return boolean false si ocurrió algún error, true si se
303 ejecutó correctamente
304 */
305 public function delete()
306 {
307     $result = $this-> db-> delete('tes_notificacion' , array('id' => $this-
> getId()));
308     if (!$result){
309         $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
310         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number()." : '$this-> db-> _error_message()';
311         throw new Exception(__CLASS__);
312     }
313     return $result;
314 }
315 /**
316 * Agrega una nueva regla de filtrado al arreglo de filtros
317 *
318 * @access public
319 * @param string $columna Puede ser cualquier campo del objeto (id, id_usuario,
fecha_hora, parametros, id_controlador_accion)
320 * @param string $condicion Establece la condición a evaluar, entre los valores permitidos
están: =, !=, >, <, >=, <=
321 * @param string $valor Valor contra el cual se realizará la evaluación del campo
322 * @return void/boolean Devuelve falso en caso de no poder establecer el filtro
323 */
324 public function addFilter($columna, $condicion, $valor)
325 {
326     $columnasPermitidas = array(
327         'titulo',
328         'contenido',
329         'fecha_inicio'
330     );
331     $condicionesPermitidas = array('=' , '>' , '<' , '!=', '>=' , '<=' , 'like');
332     if(!in_array($columna, $columnasPermitidas)) {
333         $this-> error = true;
334         $this-> msg_error_usr = 'ERROR: Columna no permitida en el filtro ('.$columna.')';
335         $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> msg_error_usr;
336         throw new Exception(__CLASS__);
337     }
338     if(!in_array($condicion, $condicionesPermitidas)) {
339         $this-> error = true;
340         $this-> msg_error_usr = 'ERROR: Condición no permitida en el filtro
('.$condicion.')';
341         $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> msg_error_usr;
342         throw new Exception(__CLASS__);
343     }
344     if(empty($valor)) {
345         $this-> error = true;
346         $this-> msg_error_usr = 'ERROR: Debe definir un valor para el filtro';
347         $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> msg_error_usr;
348         throw new Exception(__CLASS__);
349     }
350     $this-> filters[$columna . '.' . $condicion] = $valor;
351 }
352 }
353 }
354 }
355 }
356 }
357 }
358 }
359 }
360 }

```

361 ?>

# Archivo fuente para reporteador\_model.php

La documentación para este archivo está disponible en [reporteador\\_model.php](#)

```
1  <?php
2  /**
3   * Modelo Reporteador
4   *
5   * @package      TES
6   * @subpackage   Modelo
7   * @author       Rogelio
8   * @created      2013-12-20
9   */
10  class Reporteador_model extends CI_Model {
11
12      /**
13      * Estas variables no pertenecen a la tabla *
14      * ****
15
16      /**
17      * @access private
18      * @var    string
19      */
20      private $msg_error_usr;
21
22      /**
23      * @access private
24      * @var    string
25      */
26      private $msg_error_log;
27
28      public function __construct()
29      {
30          parent::__construct();
31
32          $this-> load-> database();
33          if (!$this-> db-> conn_id)
34              throw new Exception("No se pudo conectar a la base de datos");
35
36      /**
37      * Asigna el mensaje de error a visualizar: para usuario final (usr) o para bitácora (log)
38      *
39      * @access    public
40      * @param     string      $value      tipo de error a visualizar: usr o log,
41      * @return    boolean     false       si ocurrió algún error, true si se ejecutó
42      * correctamente
43      */
44      public function getMsgError($value = 'usr')
45      {
46          if ($value == 'log')
47              return $this-> msg_error_log;
48          return $this-> msg_error_usr;
49      }
50
51      /**
52      * Obtiene el reporte de cobertura por tipo de biológico
53      *
54      * @access  public
55      * @param   int      $nivel  Nivel del elemento del árbol ASU
56      * @param   int      $id     Identificador del elemento ASU
57      * @param   date    $fecha  Fecha de corte de elemento
58      * @return  void
59      */
60      public function get CoberturaBiologicoListado($nivel, $id, $fecha, $fechaFin = null)
61      {
62          $result = array();
63          $idsAsu = array();
```

```

63     $idsUMs = array();
64     $isLocalidad = false;
65     $isUnidadMedica = false;
66     // NOTA: Excluir el grupo etareo menor de ocho
67     $sqlGrupoEtareo = "SELECT * FROM asu_grupo_etareo WHERE id!=9 ORDER BY
dia_fin";
68 ;
69     $queryGrupoEtareo = $this-> db-> query($sqlGrupoEtareo);
70     $resultGrupoEtareo = $queryGrupoEtareo-> result();
71 ;
72     if (!$resultGrupoEtareo){
73         $this-> msg_error_usr = "Error al obtener los grupos etareos." ;
74         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
75         throw new Exception("(" . __METHOD__ . ") => " . $this-> db-
> _error_number().': '.$this-> db-> _error_message());
76     }
77 ;
78     // Se obtiene datos del asu
79     $queryAsu = $this-> db-> query('SELECT * FROM asu_arbol_segmentacion WHERE
id='.$id);
80     $resultAsu = $queryAsu-> row();
81 ;
82     if (!$resultAsu){
83         $this-> msg_error_usr = "Error al obtener los datos del ASU." ;
84         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
85         throw new Exception("No se encuentra el Identificador de ASU para el nivel de
filtro seleccionado" );
86     }
87 ;
88     switch ($resultAsu-> grado_segmentacion) {
89         case 1: // Estado
90             // Obtiene todos los municipios del estado
91             $queryIdsAsu = $this-> db-> query('SELECT id FROM asu_arbol_segmentacion
WHERE id_padre IN
92                                     ( SELECT id FROM asu_arbol_segmentacion WHERE id_raiz=1 AND
id_padre='.$id.' )');
93             $resultIdsAsu = $queryIdsAsu-> result();
94 ;
95             if (!$resultIdsAsu){
96                 $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
97                 $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
98                 throw new Exception("(" . __METHOD__ . ") => " . $this-> db-
> _error_number().': '.$this-> db-> _error_message());
99             }
100 ;
101             foreach ($resultIdsAsu as $tempAsu) {
102                 $idsAsu[] = $tempAsu-> id;
103             }
104 ;
105             break;
106         case 2: // Jurisdicción
107             // Obtiene todos los municipios de la jurisdicción
108             $queryIdsAsu = $this-> db-> query('SELECT id FROM asu_arbol_segmentacion
WHERE id_raiz=1 AND id_padre = '.$id);
109             $resultIdsAsu = $queryIdsAsu-> result();
110 ;
111             if (!$resultIdsAsu){
112                 $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
113                 $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
114                 throw new Exception("(" . __METHOD__ . ") => " . $this-> db-
> _error_number().': '.$this-> db-> _error_message());
115             }
116 ;
117             foreach ($resultIdsAsu as $tempAsu) {
118                 $idsAsu[] = $tempAsu-> id;
119             }
120 ;
121             break;
122         case 3: // Municipio
123             $idsAsu = array($id);
124             break;
125         case 4: // Localidad
126             $isLocalidad = true;
127             $idsAsu = array($id);
128             break;
129         case 5: // Unidad Medica

```

```

130         $isUnidadMedica = true;
131         $idsAsu = array($id);
132         break;
133     default:
134         $this-> msg_error_usr = "El grado de segmentación específico no es valido
para este reporte. Solo se puede emitir a nivel Estatal, Jurisdiccional y Municipal" ;
135         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
136         throw new Exception("El grado de segmentación específico no es valido para
este reporte. Solo se puede emitir a nivel Estatal, Jurisdiccional y Municipal" );
137         break;
138     }
139
140     $gruposVacuna = $this-> Reporteador_model-> getGrupoVacunas();
141
142     // Validar que exista poblacion para el año especificado
143     $anio = (int)formatFecha($fecha, 'Y');
144
145     $queryPob = $this-> db-> query('SELECT
146             SUM(poblacion) AS poblacion
147             FROM
148                 asu_poblacion
149             WHERE
150                 id_asu IN ('.implode(',',$idsAsu').') AND
151                 ano = '.$anio);
152
153     $resultPob = $queryPob-> row();
154
155     // Si no existe poblacion en el año, tomar la población del año anterior
156     if(!$resultPob-> poblacion) {
157         $anio--;
158     }
159
160     // Obtiene el id asu de las UMs
161     if($isLocalidad) {
162         $queryIdsUMS = $this-> db-> query('SELECT id FROM asu_arbol_segmentacion WHERE
id_raiz=1 AND id_padre IN ('.implode(',',$idsAsu').')');
163         $resultIdsUMS = $queryIdsUMS-> result();
164     }
165     if($isUnidadMedica) {
166         $queryIdsUMS = $this-> db-> query('SELECT id FROM asu_arbol_segmentacion WHERE
id_raiz=1 AND id IN ('.implode(',',$idsAsu').')');
167         $resultIdsUMS = $queryIdsUMS-> result();
168     }
169
170     if (!$resultIdsUMS){
171         $this-> msg_error_usr = "No se encontraron unidades médicas con los
parametros especificados." ;
172         $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().': '.$this-> db-> _error_message();
173         throw new Exception("(" . __METHOD__ . ") => " . $this-> db-
> _error_number().': '.$this-> db-> _error_message());
174     }
175
176     foreach ($resultIdsUMS as $tempAsu) {
177         $idsUMS[] = $tempAsu-> id;
178     }
179
180     foreach ($resultGrupoEtareo as $grupoEtareo) {
181         // Se crea el objeto reporte del tipo clase generica
182         // dado que el reporte es dinamico no se puede conocer
183         // con anticipación las columnas que contendrá
184         $objReporte = new stdClass();
185         $idGrupoEtareo = $grupoEtareo-> id;
186
187         // Corrige el grupo etareo menor de uno
188         // se toma la poblacion de menores de uno para todos grupos de meses
189         if($idGrupoEtareo>= 10 && $idGrupoEtareo<= 21) {
190             $idGrupoEtareo = 1;
191         }
192
193         $queryPob = $this-> db-> query('SELECT
194             SUM(poblacion) AS poblacion
195             FROM
196                 asu_poblacion
197             WHERE
198                 id_asu IN ('.implode(',',$idsAsu').') AND
199                 id_grupo_etareo = '.$idGrupoEtareo.' AND
200                 ano = '.$anio);
201

```

```

202         $resultPob = $queryPob->    row();
203
204         if (!$resultPob) {
205             $this->  msg_error_usr = "No se pudo obtener los datos de la
206 poblaci n" ;
207             $this->  msg_error_log = "No se pudo obtener los datos de la
208 poblaci n" ;
209             //throw new Exception("No se pudo obtener los datos de la poblaci n");
210             $resultPob = new stdClass();
211             $resultPob->  poblacion = 0;
212         }
213
214         $queryNom = $this->  db->  query('SELECT
215             COUNT(id) AS nominal
216             FROM
217             cns_persona
218             WHERE
219                 activo = 1 AND
220                 id_asu_um_tratante IN ('.$_implode(',',$idsUMS).') AND
221                 TIMESTAMPDIFF(DAY, fecha_nacimiento, "'.$formatFecha($fecha, 'Y-m-
d').'")'
222                     BETWEEN '.$grupoEtareo->  dia_inicio.' AND '.$grupoEtareo->  dia_fin);
223
224         $resultNom = $queryNom->  row();
225
226         $objReporte->  grupo_etareo = $grupoEtareo->  descripcion;
227         $objReporte->  pob_oficial = (int)$resultPob->  poblacion;
228         $objReporte->  pob_nominal = (int)$resultNom->  nominal;
229         $objReporte->  concordancia = $objReporte->  pob_oficial ? round((($objReporte-
>  pob_nominal/$objReporte->  pob_oficial)*100, 2) : 0;
230
231         foreach ($gruposVacuna as $grupo) {
232             $ultimaDosisTotal = 0;
233             $ultimaDosisDescrip = '';
234
235             // Obtiene las vacunas de cada grupo
236             $vacunas = $this->  Reporteador_model->  getVacunasByGrupo($grupo->  grupo);
237
238             foreach ($vacunas as $vac) {
239                 if( $grupoEtareo->  dia_inicio >= $vac->  dia_inicio_aplicacion_nacido
&& $vac->  dia_fin_aplicacion_nacido <= ( $grupoEtareo->  dia_fin+1 ) ){
240                     $objReporte->{  $vac->  descripcion_corta} = $vac-
241 >  descripcion_corta;
242                     $queryCob = $this->  db->  query('SELECT
243                         COUNT(cns_persona.id) AS total
244                         FROM
245                             cns_persona
246                             INNER JOIN
247                             cns_control_vacuna
248                             ON cns_persona.id = cns_control_vacuna.id_persona
249                             WHERE
250                                 activo = 1 AND
251                                 cns_control_vacuna.fecha<="'.$formatFecha($fecha, 'Y-m-
d').'" AND
252                                 id_vacuna = '.$vac->  id.' AND
253                                 id_asu_um_tratante IN ('.$_implode(',',$idsUMS).') AND
254                                 TIMESTAMPDIFF(DAY, fecha_nacimiento,
255                                 .formatFecha($fecha, 'Y-m-d'))."')
256                                 BETWEEN '.$grupoEtareo->  dia_inicio.' AND '.$grupoEtareo-
257 >  dia_fin);
258                     $resultCob = $queryCob->  row();
259
260                     $objReporte->{  $vac->  descripcion_corta} = $resultCob->  total;
261                     $ultimaDosisTotal = $resultCob->  total;
262                 } else {
263                     $objReporte->{  $vac->  descripcion_corta} = '-';
264                     $ultimaDosisTotal = 0;
265                 }
266                 $ultimaDosisDescrip = $vac->  descripcion_corta;
267             }
268             $objReporte->{  $ultimaDosisDescrip.'_cob'} = ($objReporte->  pob_oficial ?
269             round($ultimaDosisTotal/$objReporte->  pob_oficial, 2)*100 : 0).'';
270
271             $queryEsqComp = $this->  db->  query('SELECT
272                 COUNT(evalEsquema(id, TIMESTAMPDIFF(DAY, fecha_nacimiento, CURDATE()))) AS
273 esquema_completo
274                 FROM

```

```

271             cns_persona
272             WHERE activo = 1 AND
273             id_asu_um_tratante IN ('.implode(',',$idsUMs)') AND
274             TIMESTAMPDIFF(DAY, fecha_nacimiento, '') .formatFecha($fecha, 'Y-m-
d'). '')
275             BETWEEN '.$grupoEtareo-> dia_inicio.' AND '.$grupoEtareo-> dia_fin)';
276
277             $resultEsqComp = $queryEsqComp-> row();
278
279             $objReporte-> esq_comp_tot = $resultEsqComp-> esquema_completo;
280             $objReporte-> esq_comp_oficial = $objReporte-> pob_oficial ? round($objReporte-
> esq_comp_tot/$objReporte-> pob_oficial, 2) : 0;
281             $objReporte-> esq_comp_nominal = $objReporte-> pob_nominal ? round($objReporte-
> esq_comp_tot/$objReporte-> pob_nominal, 2) : 0;
282
283             $result[] = $objReporte;
284         }
285
286         return $result;
287     }
288
289     public function getConcentradoActividades($nivel, $id, $fecha)
290     {
291         $sql = "";
292         $query = $this-> db-> query($sql); //echo $this->db->last_query();
293
294         if (!$query){
295             $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
296             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().':'. $this-> db-> _error_message(); ;
297             throw new Exception(__CLASS__);
298         }
299         else
300             return $query-> result();
301         return;
302     }
303
304     public function getSeguimientoRV1RV5($nivel, $id, $fecha)
305     {
306         $sql = "";
307         $query = $this-> db-> query($sql); //echo $this->db->last_query();
308
309         if (!$query){
310             $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
311             $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number().':'. $this-> db-> _error_message(); ;
312             throw new Exception(__CLASS__);
313         }
314         else
315             return $query-> result();
316         return;
317     }
318
319     public function getCensoNominal($nivel, $id, & $th)
320     {
321         $result = array();
322         $sqlIdsConTutor = "SELECT
p.id,p.apellido_paterno,p.apellido_materno,p.nombre,p.calle_domicilio AS
domicilio,p.curp,p.fecha_nacimiento,p.sexo,'' AS edadEmb,
'' AS esquema,t.apellido_paterno AS apellido_paterno_tutor,t.apellido_materno AS
apellido_materno_tutor,
t.nombre AS nombre_tutor,t.curp AS curp_tutor,t.sexo AS sexo_tutor, m.descripcion
AS parto_multiple
FROM cns_persona p
INNER JOIN cns_persona_x_tutor pt ON p.id=pt.id_persona
INNER JOIN cns_tutor t ON t.id=pt.id_tutor INNER JOIN cns_parto_multiple m ON
p.id_parto_multiple=m.id WHERE p.activo=1" ;
323         switch($nivel){
324             case 5:
325                 $sqlIdsConTutor .= " AND p.id_asu_um_tratante=" . $id;
326                 break;
327             case 4:
328                 $sqlIdsConTutor .= " AND p.id_asu_um_tratante IN (
SELECT id FROM asu_arbol_segmentacion WHERE
id_padre=" . $id . " ) " ; // ums por loc
329                 break;
330             case 3:
331                 $sqlIdsConTutor .= " AND p.id_asu_um_tratante IN (
SELECT id FROM asu_arbol_segmentacion WHERE id_padre IN (
SELECT id FROM asu_arbol_segmentacion WHERE
"
332
333
334
335
336
337
338
339

```

```

id_padre="      .$id.") )"
340          break;
341      case 2:
342          $sqlIdsConTutor .= " AND p.id_asu_um_tratante IN (
343              SELECT id FROM asu_arbol_segmentacion WHERE id_padre IN (
344                  SELECT id FROM asu_arbol_segmentacion WHERE id_padre IN (
345                      SELECT id FROM asu_arbol_segmentacion WHERE
346              id_padre="      .$id.") ) )
347          break;
348      case 1:
349          $sqlIdsConTutor .= " AND p.id_asu_um_tratante IN (
350              SELECT id FROM asu_arbol_segmentacion WHERE id_padre IN (
351                  SELECT id FROM asu_arbol_segmentacion WHERE id_padre IN (
352                      SELECT id FROM asu_arbol_segmentacion WHERE id_padre="      .$id.")
353      ) ) "
354          ; // juris por estado
355          break;
356      }
357
358      $queryIdsConTutor = $this-> db-> query($sqlIdsConTutor);
359      $resultIdsConTutor = $queryIdsConTutor-> result();
360
361      if (!$resultIdsConTutor){
362          if ($this-> db-> _error_number() != 0){
363              $this-> msg_error_usr = "Servicio temporalmente no disponible."
364              $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
365          > db-> _error_number().': ' . $this-> db-> _error_message();
366          throw new Exception("(" . __METHOD__ . ") => " . $this-> db-
367          > _error_number().': ' . $this-> db-> _error_message());
368      }
369
370      $sqlVacunasEsquemaCompleto = "SELECT v.id,v.descripcion_corta as descripcion FROM
371      cns_regla_vacuna rv INNER JOIN cns_vacuna v ON rv.id_vacuna=v.id
372      WHERE rv.esq_com=1 ORDER BY rv.orden_esq_com";
373
374      $queryVacunasEsquemaCompleto = $this-> db-> query($sqlVacunasEsquemaCompleto);
375      $resultVacunasEsquemaCompleto = $queryVacunasEsquemaCompleto-> result();
376      $th = '<tr><th>Apellido Paterno</th><th>Apellido
Materno</th><th>Nombre</th>
377          <th>Domicilio</th><th>CURP</th><th>Fecha
Nac</th><th>Parto Múltiple</th><th>Sexo</th>' ;
378      foreach($resultVacunasEsquemaCompleto as $vacuna){
379          $th.= '<th><div><span>' . $vacuna-
380      > descripcion.'</span></div></th>' ;
381          $th.='</tr>' ;
382
383          foreach ($resultIdsConTutor as $IdConTutor) {
384              $sqlVacunasAplicadas = "SELECT id_vacuna FROM cns_control_vacuna WHERE
385              id_persona=" . $IdConTutor-> id . ";
386              $queryVacunasAplicadas = $this-> db-> query($sqlVacunasAplicadas);
387              $resultVacunasAplicadas = $this-> object_to_array($queryVacunasAplicadas-
388          > result(), 'id_vacuna');
389
390              // se inserta el registro del infante
391              $objReporte = new Reporte_censo_nominal();
392              $objReporte-> apellido_paterno = $IdConTutor-> apellido_paterno;
393              $objReporte-> apellido_materno = $IdConTutor-> apellido_materno;
394              $objReporte-> nombre = $IdConTutor-> nombre;
395              $objReporte-> domicilio = $IdConTutor-> domicilio;
396              $objReporte-> curp = $IdConTutor-> curp;
397              $objReporte-> fecha_nacimiento = $IdConTutor-> fecha_nacimiento;
398              $objReporte-> parto_multiple = $IdConTutor-> parto_multiple;
399              $objReporte-> sexo = $IdConTutor-> sexo;
400              foreach ($resultVacunasEsquemaCompleto as $vacuna){
401                  $objReporte-> vacunas[$vacuna-> id] = in_array($vacuna-> id,
402              $resultVacunasAplicadas) ? VACUNA_APPLICADA : VACUNA_NOAPPLICADA;
403              }
404              $result[] = $objReporte;
405
406              // se inserta el registro del tutor
407              $objReporte = new Reporte_censo_nominal();
408              $objReporte-> apellido_paterno = $IdConTutor-> apellido_paterno_tutor;
409              $objReporte-> apellido_materno = $IdConTutor-> apellido_materno_tutor;
410              $objReporte-> nombre = $IdConTutor-> nombre_tutor;
411              $objReporte-> domicilio = '';
412              $objReporte-> curp = $IdConTutor-> curp_tutor;
413              $objReporte-> fecha_nacimiento = '';
414              $objReporte-> parto_multiple = '';
415              $objReporte-> sexo = $IdConTutor-> sexo_tutor;
416              foreach ($resultVacunasEsquemaCompleto as $vacuna){
417                  $objReporte-> vacunas[$vacuna-> id] = VACUNA_NOAPPLICADA;

```

```

408         }
409         $result[] = $objReporte;
410     }
411     return $result;
412 }
413
414     public function getEsquemasIncompletos($nivel, $id, & $sth)
415     {
416         $result = array();
417         $sqlIdsConTutor = "SELECT
p.id,p.apellido_paterno,p.apellido_materno,p.nombre,p.calle_domicilio AS
domicilio,p.curp,p.fecha_nacimiento,p.sexo,'' AS edadEmb,
        '' AS esquema,t.apellido_paterno AS apellido_paterno_tutor,t.apellido_materno AS
apellido_materno_tutor,
        t.nombre AS nombre_tutor,t.curp AS curp_tutor,t.sexo AS sexo_tutor,
TIMESTAMPDIFF(DAY, fecha_nacimiento, CURDATE()) AS edad_dias
        FROM cns_persona p
        INNER JOIN cns_persona_x_tutor pt ON p.id=pt.id_persona
        INNER JOIN cns_tutor t ON t.id=pt.id_tutor WHERE p.activo=1" ;
418
419         switch($nivel){
420             case 5:
421                 $sqlIdsConTutor .= " AND p.id_asu_um_tratante=" . $id;
422                 break;
423             case 4:
424                 $sqlIdsConTutor .= " AND p.id_asu_um_tratante IN (
425                     SELECT id FROMasu_arbol_segmentacion WHERE
426                     id_padre=" . $id . ")" ; // ums por loc
427                 break;
428             case 3:
429                 $sqlIdsConTutor .= " AND p.id_asu_um_tratante IN (
430                     SELECT id FROMasu_arbol_segmentacion WHERE id_padre IN (
431                         SELECT id FROMasu_arbol_segmentacion WHERE
432                         id_padre=" . $id . ")" ) ; // locs por mpio
433                 break;
434             case 2:
435                 $sqlIdsConTutor .= " AND p.id_asu_um_tratante IN (
436                     SELECT id FROMasu_arbol_segmentacion WHERE id_padre IN (
437                         SELECT id FROMasu_arbol_segmentacion WHERE id_padre IN (
438                             SELECT id FROMasu_arbol_segmentacion WHERE id_padre IN (
439                                 SELECT id FROMasu_arbol_segmentacion WHERE
440                                 id_padre=" . $id . ")" ) ) ; // mpios por juris
441                 break;
442             case 1:
443                 $sqlIdsConTutor .= " AND p.id_asu_um_tratante IN (
444                     SELECT id FROMasu_arbol_segmentacion WHERE id_padre IN (
445                         SELECT id FROMasu_arbol_segmentacion WHERE id_padre IN (
446                             SELECT id FROMasu_arbol_segmentacion WHERE id_padre IN (
447                                 SELECT id FROMasu_arbol_segmentacion WHERE id_padre=" . $id . ")
448                         ) ) " ; // juris por estado
449                 break;
450         }
451         $queryIdsConTutor = $this-> db-> query($sqlIdsConTutor);
452         $resultIdsConTutor = $queryIdsConTutor-> result();
453
454         if (!$resultIdsConTutor){
455             if ($this-> db-> _error_number() != 0){
456                 $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
457                 $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
458 > db-> _error_number().":'." . $this-> db-> _error_message();
459                 throw new Exception("(" . __METHOD__ . ") => " . $this-> db-
460 > _error_number().":'." . $this-> db-> _error_message());
461             }
462         }
463         $sqlVacunasEsquemaCompleto = "SELECT v.id,v.descripcion_corta AS descripcion FROM
cns_reglas_vacuna rv INNER JOIN cns_vacuna v ON rv.id_vacuna=v.id
        WHERE rv.esq_com=1 ORDER BY rv.orden_esq_com" ;
464         $queryVacunasEsquemaCompleto = $this-> db-> query($sqlVacunasEsquemaCompleto);
465         $resultVacunasEsquemaCompleto = $queryVacunasEsquemaCompleto-> result();
466         $th = '<tr><th>Apellido Paterno</th><th>Apellido
Materno</th><th>Nombre</th>
467             <th>Sexo</th>' ;
468             foreach($resultVacunasEsquemaCompleto as $vacuna){
469                 $th.= '<th>' . $vacuna-> descripcion.'</th>' ;
470             }
471             $th.= '</tr>' ;
472             foreach ($resultIdsConTutor as $IdConTutor) {
473                 $sqlVacunasAplicadas = "SELECT id_vacuna FROM cns_control_vacuna WHERE
474                     id_persona=" . $IdConTutor-> id . " " ;

```

```

474         $queryVacunasAplicadas = $this-> db-> query($sqlVacunasAplicadas);
475         $resultVacunasAplicadas = $this-> object_to_array($queryVacunasAplicadas-
> result(), 'id_vacuna');
476
477         $sqlVacunasCorresponden = "SELECT      id,      id_vacuna,
478          dia_inicio_aplicacion_nacido,      dia_fin_aplicacion_nacido
479          FROM cns_reglea_vacuna WHERE
480          ("      . $IdConTutor-> edad_dias." >= dia_inicio_aplicacion_nacido
481          AND
482          "      . $IdConTutor-> edad_dias." <= dia_fin_aplicacion_nacido) OR
483          (dia_fin_aplicacion_nacido<="      . $IdConTutor-
484          > edad_dias.")";
485         $queryVacunasCorresponden = $this-> db-> query($sqlVacunasCorresponden);
486         $resultVacunasCorresponden = $this-> object_to_array($queryVacunasCorresponden-
> result(), 'id_vacuna');
487
488         // echo $IdConTutor->id.".".$IdConTutor-
489         // >edad_dias."<br>";
490         // var_dump($resultVacunasAplicadas);
491         // echo "<br>";
492         // var_dump($resultVacunasCorresponden);
493         // echo "<br>";
494
495         // el infante debe aparecer si no tiene todas las vacunas que le correspondan
496         $puestas = 0;
497         foreach ($resultVacunasAplicadas as $vacunaPuesta){
498             foreach ($resultVacunasCorresponden as $vacunaCorresponde){
499                 if ($vacunaPuesta == $vacunaCorresponde)
500                     $puestas++;
501             }
502         }
503         if ($puestas != count($resultVacunasCorresponden)){
504             // se inserta el registro del infante
505             $objReporte = new Reporte_censo_nominal();
506             $objReporte-> apellido_paterno = $IdConTutor-> apellido_paterno;
507             $objReporte-> apellido_materno = $IdConTutor-> apellido_materno;
508             $objReporte-> nombre = $IdConTutor-> nombre;
509             $objReporte-> domicilio = $IdConTutor-> domicilio;
510             $objReporte-> curp = $IdConTutor-> curp;
511             $objReporte-> fecha_nacimiento = $IdConTutor-> fecha_nacimiento;
512             $objReporte-> sexo = $IdConTutor-> sexo;
513             foreach ($resultVacunasEsquemaCompleto as $vacuna){
514                 $objReporte-> vacunas[$vacuna-> id] = in_array($vacuna-> id,
515 $resultVacunasAplicadas) ? VACUNA_APPLICADA : VACUNA_NOAPPLICADA;
516             }
517             $result[] = $objReporte;
518             // se inserta el registro del tutor
519             $objReporte = new Reporte_censo_nominal();
520             $objReporte-> apellido_paterno = $IdConTutor-> apellido_paterno_tutor;
521             $objReporte-> apellido_materno = $IdConTutor-> apellido_materno_tutor;
522             $objReporte-> nombre = $IdConTutor-> nombre_tutor;
523             $objReporte-> domicilio = '';
524             $objReporte-> curp = $IdConTutor-> curp_tutor;
525             $objReporte-> fecha_nacimiento = '';
526             $objReporte-> sexo = $IdConTutor-> sexo_tutor;
527             foreach ($resultVacunasEsquemaCompleto as $vacuna){
528                 $objReporte-> vacunas[$vacuna-> id] = VACUNA_NOAPPLICADA;
529             }
530             $result[] = $objReporte;
531         }
532         return $result;
533     }
534
535     // convierte array de objetos en array simple de un valor
536     function object_to_array($data, $campo)
537     {
538         $result = array();
539         $i = 0;
540         foreach ($data as $key => $value)
541         {
542             $result[$i] = $value-> $campo;
543             $i++;
544         }
545         return $result;
546     }
547
548     /*
549      * Devuelve el listado de vacunas correspondientes al esquema completo
550     */

```

```

547     function getVacunas() {
548         $queryVacunas = $this-> db-> query('SELECT
549             cns_vacuna.id,
550             descripcion,
551             descripcion_corta,
552             dia_inicio_aplicacion_nacido,
553             dia_fin_aplicacion_nacido,
554             grupo
555             FROM cns_vacuna
556             INNER JOIN cns_regla_vacuna
557                 ON cns_vacuna.id = cns_regla_vacuna.id_vacuna
558             WHERE
559                 cns_vacuna.activo = 1 AND
560                 esq_com = 1
561             ORDER BY orden_esq_com');
562
563         $vacunas = $queryVacunas-> result();
564
565         return $vacunas;
566     }
567
568     /*
569      * Devuelve el listado de vacunas correspondientes a un grupo especificado
570      *
571      * @param string $grupo
572      */
573     function getVacunasByGrupo($grupo) {
574         $queryVacunas = $this-> db-> query('SELECT
575             cns_vacuna.id,
576             descripcion,
577             descripcion_corta,
578             dia_inicio_aplicacion_nacido,
579             dia_fin_aplicacion_nacido,
580             grupo
581             FROM cns_vacuna
582             INNER JOIN cns_regla_vacuna
583                 ON cns_vacuna.id = cns_regla_vacuna.id_vacuna
584             WHERE
585                 cns_vacuna.activo = 1 AND
586                 esq_com = 1 AND
587                 grupo = "' . $grupo . '"'
588             ORDER BY orden_esq_com');
589
590         $vacunas = $queryVacunas-> result();
591
592         return $vacunas;
593     }
594
595     /*
596      * Devuelve los grupos de vacunas con su respectiva cantidad de dosis
597      */
598     function getGrupoVacunas() {
599         $queryGrupoVacunas = $this-> db-> query('SELECT
600             grupo,
601             COUNT(cns_vacuna.id) AS total
602             FROM cns_vacuna
603             INNER JOIN cns_regla_vacuna
604                 ON cns_vacuna.id = cns_regla_vacuna.id_vacuna
605             WHERE
606                 cns_vacuna.activo = 1 AND
607                 esq_com = 1
608             GROUP BY grupo
609             ORDER BY orden_esq_com');
610
611         $grupos = $queryGrupoVacunas-> result();
612
613         return $grupos;
614     }
615
616 }
617 ?>
```

# Archivo fuente para reporte\_censo\_nominal.php

*La documentación para este archivo está disponible en [reporte\\_censo\\_nominal.php](#)*

```
1  <?php
2
3  /**
4   * Modelo Reporte_censo_nominal
5   *
6   * @package      TES
7   * @subpackage   Modelo
8   * @author       Rogelio
9   * @created      2013-01-08
10  */
11 class Reporte_censo_nominal extends CI_Model
12 {
13     /**
14      * @var      varchar
15      */
16     public $apellido_paterno;
17
18     /**
19      * @var      varchar
20      */
21     public $apellido_materno;
22
23     /**
24      * @var      varchar
25      */
26     public $nombre;
27
28     /**
29      * @var      varchar
30      */
31     public $domicilio;
32
33     /**
34      * @var      varchar
35      */
36     public $curp;
37
38     /**
39      * @var      varchar
40      */
41     public $fecha_nacimiento;
42
43     /**
44      * @var      varchar
45      */
46     public $parto_multiple;
47
48     /**
49      * @var      varchar
50      */
51     public $sexo;
52
53     /**
54      * @var      array
55      */
56     public $vacunas;
57
58     public function __construct()
59     {
60         parent::__construct();
61     }
62
63 }
64 ?>
```

# Archivo fuente para reporte\_sincronizacion\_model.php

La documentación para este archivo está disponible en [reporte\\_sincronizacion\\_model.php](#)

```
1  <?php
2  /**
3   * Modelo Usuario
4   *
5   * @package      TES
6   * @subpackage   Modelo
7   * @author       Eliecer
8   * @created      2013-12-17
9   */
10  class Reporte_sincronizacion_model extends CI_Model
11  {
12      /**
13      * Guarda la instancia del objeto global CodeIgniter
14      * para utilizarlo en la función estática
15      *
16      * @access private
17      * @var    instance
18      */
19      private static $CI;
20
21
22      /**
23      *
24      * Obtiene el resultado de una consulta
25      *
26      * @param      string      $sql      consulta slq a ejecutar
27      *
28      * @return     result()
29      *
30      */
31      public function getListado($sql)
32      {
33          $query = $this-> db-> query($sql); //echo $this->db->last_query();
34
35          if (!$query){
36              $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
37              $this-> msg_error_log = "(" . __METHOD__ . ") => " . $this-
> db-> _error_number(). ': ' . $this-> db-> _error_message();
38              throw new Exception(__CLASS__);
39          }
40          else
41              return $query-> result();
42          return;
43      }
44
45      /**
46      *
47      * obtiene el numero de registros de una tabla o consulta
48      *
49      * @param      string      $tabla      nombre de una tabla en la base de datos
50      * @param      string      $sentencia  consulta sql
51      *
52      * @return     num_rows()
53      *
54      */
55      public function getCount($tabla,$sentencia= "")           )
56      {
57          if($sentencia!="")
58              $query=$this-> db-> query($sentencia);
59          else
60              $query=$this-> db-> get($tabla);
61          return $query-> num_rows();
62      }
63  }
```

```

64     /**
65      *
66      * obtiene la ultima version de la apk de las tabletas
67      *
68      * @return          result()
69      */
70
71     public function get_version()
72     {
73         $this-> db-> select('host,fecha_liberacion');
74         $this-> db-> select_max('version');
75         $this-> db-> from('tes_version');
76         $query = $this-> db-> get(); //echo $this->db->last_query();
77         if (!$query)
78         {
79             $this-> msg_error_usr = "Servicio temporalmente no disponible." ;
80             $this-> msg_error_log = "(" . __METHOD__ . ") => "
> db-> _error_number() . ': ' . $this-> db-> _error_message();
81             throw new Exception(__CLASS__);
82         }
83         else
84             return $query-> result();
85         return null;
86     }
87
88     public function getMsgError($value = 'usr')
89     {
90         if ($value == 'log')
91             return $this-> msg_error_log;
92         return $this-> msg_error_usr;
93     }
94 }
```

# Archivo fuente para semana\_nacional\_model.php

*La documentación para este archivo está disponible en [semana\\_nacional\\_model.php](#)*

```
1  <?php
2
3  /**
4   * Modelo Tableta
5   *
6   * @package    TES
7   * @subpackage Modelo
8   * @author     Pascual
9   * @created    2013-11-26
10  */
11 class Semana_nacional_model extends CI_Model
12 {
13     /**
14      * @access private
15      * @var int(11)
16      */
17     private $id;
18
19     /**
20      * @access private
21      * @var varchar(45)
22      */
23     private $descripcion;
24
25     /**
26      * @access private
27      * @var date
28      */
29     private $fecha_inicio;
30
31     /**
32      * @access private
33      * @var date
34      */
35     private $fecha_fin;
36
37
38     ****
39     * Estas variables no pertenecen a la tabla *
40     ****
41
42     /**
43      * @access private
44      * @var boolean
45      */
46     private $error;
47
48     /**
49      * @access private
50      * @var string
51      */
52     private $msg_error_usr;
53
54     /**
55      * @access private
56      * @var string
57      */
58     private $msg_error_log;
59
60
61     public function __construct()
62     {
63         parent::__construct();
64         $this-> load-> database();
```

```

65     $this-> error = false;
66     $this-> msg_error_usr = '';
67     $this-> msg_error_log = '';
68
69     if( !$this-> db-> conn_id ) {
70         throw new Exception ('ERROR: No se puede conectar con la Base de Datos');
71     }
72 }
73
74 public function getId()
75 {
76     return $this-> id;
77 }
78
79 public function getDescripcion()
80 {
81     return $this-> descripcion;
82 }
83
84 public function getFecha_inicio()
85 {
86     return $this-> fecha_inicio;
87 }
88
89 public function getFecha_fin()
90 {
91     return $this-> fecha_fin;
92 }
93
94 public function setDescripcion($descripcion)
95 {
96     $this-> descripcion = $descripcion;
97 }
98
99 public function setFecha_inicio($fecha_inicio)
100 {
101     $this-> fecha_inicio = $fecha_inicio;
102 }
103
104 public function setFecha_fin($fecha_fin)
105 {
106     $this-> fecha_fin = $fecha_fin;
107 }
108
109 /**
110 * Devuelve el mensaje de error,
111 * en caso de existir un error despues de ejecutar un metodo,
112 * de lo contrario false
113 *
114 * @access public
115 * @param string $type usr si se quiere devolver el mensaje de error a mostrar en la vista,
116 *                      log obtiene el mensaje de error con mas detalles para depuración,
117 *                      valor por defecto usr
118 * @return boolean/string
119 */
120 public function getMsgError($type = 'usr')
121 {
122     if($this-> error) {
123         if($type == 'usr')
124             return $this-> msg_error_usr;
125         else if($type == 'log')
126             return $this-> msg_error_log;
127         else
128             return false;
129     }
130
131     return false;
132 }
133
134 /**
135 * Inserta en la base de datos, la informacion contenida en el objeto
136 *
137 * @access public
138 * @return boolean false Si no se ejecutó la inserción, true si se ejecutó la inserción
139 */
140 public function insert()
141 {
142     $result = false;
143     $data = array();
144

```

```

145     $data['descripcion'] = $this-> descripcion;
146     $data['fecha_inicio'] = $this-> fecha_inicio;
147     $data['fecha_fin'] = $this-> fecha_fin;
148
149     $result = $this-> db-> insert('cns_semana_vacunacion', $data);
150
151     if( $this-> db-> _error_number() ) {
152         $this-> error = true;
153         $this-> msg_error_usr = 'No se puede insertar el registro';
154         $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
> _error_number().': '.$this-> db-> _error_message();
155         throw new Exception($this-> msg_error_log);
156     } else {
157         // Obtiene el id asignado a la ultima inserción
158         $this-> id = $this-> db-> insert_id();
159     }
160
161     return $result;
162 }
163
164 /**
165 * Actualiza los datos del objeto actual
166 *
167 * @access public
168 * @param int $id Si no se establece el valor de ID, se toma el valor del objeto actual
169 * @return boolean false Si no se ejecutó la actualización, true si se ejecutó la actualización
170 */
171 public function update($id = null)
172 {
173     $result = false;
174     $data = array();
175
176     $data['descripcion'] = $this-> descripcion;
177     $data['fecha_inicio'] = $this-> fecha_inicio;
178     $data['fecha_fin'] = $this-> fecha_fin;
179
180     $id = is_null($id) ? $this-> id : $id;
181     $result = $this-> db-> update('cns_semana_vacunacion', $data, array('id' =>
$id));
182
183     if( $this-> db-> _error_number() ) {
184         $this-> error = true;
185         $this-> msg_error_usr = 'No se puede actualizar el registro';
186         $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
> _error_number().': '.$this-> db-> _error_message();
187         throw new Exception();
188     } else {
189         // Obtiene el id asignado a la ultima inserción
190         $this-> id = $id;
191     }
192
193     return $result;
194 }
195
196 /**
197 * Elimina el registro actual de la base de datos
198 *
199 * @access public
200 * @param int $id Si no se establece el valor de ID, se toma el valor del objeto actual
201 * @return int    false Si no se eliminó el registro, true si se ejecutó la eliminación
202 */
203 public function delete($id = null)
204 {
205     $result = false;
206
207     $id = is_null($id) ? $this-> id : $id;
208
209     if(is_array($id)) {
210         // Eliminar un conjunto de registros
211         foreach ($id as $idx) {
212             $result = $this-> db-> delete('cns_semana_vacunacion', array('id' =>
$idx));
213
214             if(empty($result)) {
215                 $this-> error = true;
216                 $this-> msg_error_usr = 'No se puede eliminar el registro';
217                 $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
> _error_number().': '.$this-> db-> _error_message();
218                 throw new Exception();
219             }

```

```

220     }
221     } else {
222         // Eliminar un solo registro
223         $result = $this-> db-> delete('cns_semana_vacunacion', array('id' => $id));
224
225         if(empty($result)) {
226             $this-> error = true;
227             $this-> msg_error_usr = 'No se puede eliminar el registro';
228             $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
229 > _error_number().': '.$this-> db-> _error_message();
230             throw new Exception();
231         }
232
233         return $result;
234     }
235
236     /**
237      * Obtiene los datos del registro que tiene el ID especificado
238      *
239      * @access public
240      * @param int $id      Si no se establece el valor de ID, se toma el valor del objeto
241      *                      contrario, false Si no se encontró el registro
242      */
243     public function getById($id)
244     {
245         $result = false;
246
247         $this-> db-> select('*');
248         $this-> db-> from('cns_semana_vacunacion');
249         $this-> db-> where('id', $id);
250
251         $query = $this-> db-> get();
252         $result = $query-> row();
253
254         if($this-> db-> _error_number()) {
255             $this-> error = true;
256             $this-> msg_error_usr = 'No se encontraron registros en la búsqueda';
257             $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
258 > _error_number().': '.$this-> db-> _error_message();
259             throw new Exception();
260         }
261
262         if(!empty($result)) {
263             $this-> id = $id;
264             $this-> descripcion = $result-> descripcion;
265             $this-> fecha_inicio = $result-> fecha_inicio;
266             $this-> fecha_fin = $result-> fecha_fin;
267         }
268
269         return $result;
270     }
271
272     /**
273      * Obtiene todos los registros de la tabla
274      *
275      * @access public
276      * @param int $offset    Establece el desplazamiento del primer registro a devolver,
277      *                      si se define solo el valor de offset
278      *                      el valor especifica el número de registros a retornar desde el
279      *                      comienzo del conjunto de resultados.
280      * @param int $row_count Establece la cantidad de registros a devolver
281      * @return array object  Devuelve un arreglo de objetos obtenidos de la base de datos
282      */
283     public function getAll($offset=null, $row_count=null)
284     {
285         $result = 0;
286
287         $this-> db-> select('id, descripcion, fecha_inicio, fecha_fin');
288         $this-> db-> from('cns_semana_vacunacion');
289
290         if(!empty($offset) && !empty($row_count))
291             $this-> db-> limit($offset, $row_count);
292         else if (!empty($offset))
293             $this-> db-> limit($offset);
294
295         $query = $this-> db-> get();

```

```

295     $result = $query->    result();
296
297     if($this->  db->  _error_number()) {
298         $this->  error = true;
299         $this->  msg_error_usr = 'No se encontraron registros en la busqueda';
300         $this->  msg_error_log = '('.__METHOD__.') => ' . $this->  db-
> _error_number().': '.$this->  db->  _error_message();
301         throw new Exception();
302     } else if(empty ($result)) {
303         $this->  msg_error_usr = 'No se encontraron registros en la busqueda';
304     }
305
306     return $result;
307 }
308
309 /**
310 * Obtiene el numero total de registros en la tabla
311 *
312 * @access public
313 * @return int
314 */
315 public function getNumRows()
316 {
317     $result = 0;
318
319     $result = $this->  db->  count_all_results('cns_semana_vacunacion');
320
321     if($this->  db->  _error_number()) {
322         $this->  error = true;
323         $this->  msg_error_usr = 'No se encontraron registros en la busqueda';
324         $this->  msg_error_log = '('.__METHOD__.') => ' . $this->  db-
> _error_number().': '.$this->  db->  _error_message();
325         throw new Exception();
326     } else if(empty ($result)) {
327         $this->  msg_error_usr = 'No se encontraron registros en la busqueda';
328     }
329
330     return $result;
331 }
332 }
333 ?>

```

# Archivo fuente para tableta\_model.php

La documentación para este archivo está disponible en [tableta\\_model.php](#)

```
1  <?php
2
3  /**
4   * Modelo Tableta
5   *
6   * @package    TES
7   * @subpackage Modelo
8   * @author     Pascual
9   * @created    2013-11-26
10  */
11 class Tableta_model extends CI_Model
12 {
13     /**
14      * @access private
15      * @var int(11)
16      */
17     private $id;
18
19     /**
20      * @access private
21      * @var varchar(20)
22      */
23     private $mac;
24
25     /**
26      * @access varchar(10)
27      * @var int
28      */
29     private $id_version;
30
31     /**
32      * @access private
33      * @var datetime
34      */
35     private $usuarios_asignados;
36
37     /**
38      * @access private
39      * @var bit(1)
40      */
41     private $ultima_actualizacion;
42
43     /**
44      * @access private
45      * @var int(11)
46      */
47     private $id_tes_estado_tableta;
48
49     /**
50      * @access private
51      * @var int(11)
52      */
53     private $id_tipo_censo;
54
55     /**
56      * @access private
57      * @var int(11)
58      */
59     private $id_asu_um;
60
61     /**
62      * @access private
63      * @var int(11)
64      */
65     private $periodo_esq_inc;
66
67     *****/
```

```

68     * Estas variables no pertenecen a la tabla *
69     * ****
70
71     /**
72      * @access private
73      * @var    boolean
74     */
75     private $error;
76
77     /**
78      * @access private
79      * @var    string
80     */
81     private $msg_error_usr;
82
83     /**
84      * @access private
85      * @var    string
86     */
87     private $msg_error_log;
88
89
90     public function __construct()
91     {
92         parent::__construct();
93         $this-> load-> database();
94         $this-> error = false;
95         $this-> msg_error_usr = '';
96         $this-> msg_error_log = '';
97
98         if( !$this-> db-> conn_id ) {
99             throw new Exception ('ERROR: No se puede conectar con la Base de Datos');
100        }
101    }
102
103    public function getId()
104    {
105        return $this-> id;
106    }
107
108    public function getMac() {
109        return $this-> mac;
110    }
111
112    public function getIdVersion() {
113        return $this-> id_version;
114    }
115
116    public function getLastUpdate() {
117        return $this-> ultima_actualizacion;
118    }
119
120    public function getUsersAssigned() {
121        return $this-> usuarios_asignados;
122    }
123
124    public function getIdTabletStatus() {
125        return $this-> id_tablet_status;
126    }
127
128    public function getIdCensoType() {
129        return $this-> id_censo_type;
130    }
131
132    public function getIdAssumption() {
133        return $this-> id_assumption;
134    }
135
136    public function getLastPeriod() {
137        return $this-> periodo_esq_inc;
138    }
139
140    public function setId($id)
141    {
142        return $this-> id = $id;
143    }
144
145    public function setMac($mac) {
146        $this-> mac = $mac;
147    }

```

```

148
149     public function setIdVersion($id_version) {
150         $this-> id_version = $id_version;
151     }
152
153     public function setUltima_actualizacion($ultima_actualizacion) {
154         $this-> ultima_actualizacion = $ultima_actualizacion;
155     }
156
157     public function setUsuarios_asignados($usuarios_asignados) {
158         $this-> usuarios_asignados = $usuarios_asignados;
159     }
160
161     public function setId_tes_estado_tableta($id_tes_estado_tableta) {
162         $this-> id_tes_estado_tableta = $id_tes_estado_tableta;
163     }
164
165     public function setId_tipo_censo($id_tipo_censo) {
166         $this-> id_tipo_censo = $id_tipo_censo;
167     }
168
169     public function setId_asu_um($id_asu_um) {
170         $this-> id_asu_um = $id_asu_um;
171     }
172
173     public function setPeriodo_esq_inc($periodo_esq_inc) {
174         return $this-> periodo_esq_inc = $periodo_esq_inc;
175     }
176
177 /**
178 * Devuelve el mensaje de error,
179 * en caso de existir un error despues de ejecutar un metodo,
180 * de lo contrario false
181 *
182 * @access public
183 * @param string $type usr si se quiere devolver el mensaje de error a mostrar en la vista,
184 *                      log obtiene el mensaje de error con mas detalles para depuración,
185 *                      valor por defecto usr
186 * @return boolean/string
187 */
188     public function getMsgError($type = 'usr')
189     {
190         if($this-> error) {
191             if($type == 'usr')
192                 return $this-> msg_error_usr;
193             else if($type == 'log')
194                 return $this-> msg_error_log;
195             else
196                 return false;
197         }
198         return false;
199     }
200
201 /**
202 * Inserta en la base de datos, la informacion contenida en el objeto
203 *
204 * @access public
205 * @return boolean false Si no se ejecutó la inserción, true si se ejecutó la inserción
206 */
207     public function insert()
208     {
209         $result = false;
210         $data = array();
211
212         $data['mac'] = trim($this-> mac);
213         $data['usuarios_asignados'] = 0;
214         $data['id_tes_estado_tableta'] = 1;
215
216         $result = $this-> db-> insert('tes_tableta', $data);
217
218         if( $this-> db-> _error_number() ) {
219             $this-> error = true;
220             $this-> msg_error_usr = 'No se puede insertar el registro';
221             $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
222 > _error_number().': '.$this-> db-> _error_message();
223             throw new Exception();
224         } else {
225             // Obtiene el id asignado a la ultima inserción
226             $this-> id = $this-> db-> insert_id();

```

```

227     }
228
229     return $result;
230 }
231
232 /**
233 * Actualiza los datos del objeto actual
234 *
235 * @access public
236 * @param int $id Si no se establece el valor de ID, se toma el valor del objeto actual
237 * @return boolean false Si no se ejecutó la actualización, true si se ejecutó la actualización
238 */
239 public function update($id = null)
{
    $result = false;
    $data = array();
243
    $data['mac'] = $this-> mac;
245
    if( !empty($this-> id_tes_estado_tableta) )
        $data['id_tes_estado_tableta'] = $this-> id_tes_estado_tableta;
248
    if( !empty($this-> id_version) )
        $data['id_version'] = $this-> id_version;
251
    if( !empty($this-> ultima_actualizacion) )
        $data['ultima_actualizacion'] = $this-> ultima_actualizacion;
254
    if( !empty($this-> usuarios_asignados) )
        $data['usuarios_asignados'] = $this-> usuarios_asignados;
257
    if( !empty($this-> id_tipo_censo) )
        $data['id_tipo_censo'] = $this-> id_tipo_censo;
260
    if( !empty($this-> id_asu_um) ) {
        $data['id_asu_um'] = $this-> id_asu_um;
        $data['id_tes_estado_tableta'] = 2;
    }
265
    if( !empty($this-> periodo_esq_inc) )
        $data['periodo_esq_inc'] = $this-> periodo_esq_inc;
268
    $id = is_null($id) ? $this-> id : $id;
    $result = $this-> db-> update('tes_tableta', $data, array('id' => $id));
271
    if( $this-> db-> _error_number() ) {
        $this-> error = true;
        $this-> msg_error_usr = 'No se puede actualizar el registro';
        $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
> _error_number().': '.$this-> db-> _error_message();
        throw new Exception();
    } else {
        // Obtiene el id asignado a la ultima inserción
        $this-> id = $id;
    }
282
    return $result;
}
283
284 /**
285 * Elimina el registro actual de la base de datos
286 *
287 * @access public
288 * @param int $id Si no se establece el valor de ID, se toma el valor del objeto actual
289 * @return int false Si no se eliminó el registro, true si se ejecutó la eliminación
290 */
291 public function delete($id = null)
{
    $result = false;
294
    $id = is_null($id) ? $this-> id : $id;
297
    if(is_array($id)) {
        // Eliminar un conjunto de registros
        foreach ($id as $idx) {
            $result = $this-> db-> delete('tes_tableta', array('id' => $idx));
302
            if(empty($result)) {
                $this-> error = true;
                $this-> msg_error_usr = 'No se puede eliminar el registro';
            }
        }
    }
}

```

```

306         $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
> _error_number().': '.$this-> db-> _error_message();
307         throw new Exception();
308     }
309 }
310 } else {
311     // Eliminar un solo registro
312     $result = $this-> db-> delete('tes_tableta', array('id' => $id));
313
314     if(empty($result)) {
315         $this-> error = true;
316         $this-> msg_error_usr = 'No se puede eliminar el registro';
317         $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
> _error_number().': '.$this-> db-> _error_message();
318         throw new Exception();
319     }
320 }
321
322 return $result;
323 }
324
325 /**
326 * Obtiene los datos del registro que tiene el ID especificado
327 *
328 * @access public
329 * @param int $id           Si no se establece el valor de ID, se toma el valor del objeto
actual
330 * @return object/booleanDevuelve el objeto con sus datos correspondientes, de lo
contrario, false Si no se encontró el registro
331 */
332 public function getById($id)
333 {
334     $result = false;
335
336     $this-> db-> select('tes_tableta.*', sis_estado_tableta.descripcion AS status,
tes_tipo_censo.descripcion AS tipo_censo');
337     $this-> db-> from('tes_tableta');
338     $this-> db-> join('sis_estado_tableta', 'tes_tableta.id_tes_estado_tableta =
sis_estado_tableta.id', 'left');
339     $this-> db-> join('tes_tipo_censo', 'tes_tableta.id_tipo_censo =
tes_tipo_censo.id', 'left');
340     $this-> db-> where('tes_tableta.id', $id);
341
342     $query = $this-> db-> get();
343     $result = $query-> row();
344
345     if($this-> db-> _error_number()) {
346         $this-> error = true;
347         $this-> msg_error_usr = 'No se encontraron registros en la búsqueda';
348         $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
> _error_number().': '.$this-> db-> _error_message();
349         throw new Exception();
350     }
351
352     if(!empty($result)) {
353         $this-> id = $id;
354         $this-> mac = $result-> mac;
355         $this-> id_version = $result-> id_version;
356         $this-> ultima_actualizacion = $result-> ultima_actualizacion;
357         $this-> usuarios_asignados = $result-> usuarios_asignados;
358         $this-> id_tes_estado_tableta = $result-> id_tes_estado_tableta;
359         $this-> id_tipo_censo = $result-> id_tipo_censo;
360         $this-> id_asu_um = $result-> id_asu_um;
361         $this-> periodo_esq_inc = $result-> periodo_esq_inc;
362     }
363
364     return $result;
365 }
366
367 /**
368 * Obtiene los datos del registro la MAC especificada
369 *
370 * @access public
371 * @param int $mac           Dirección MAC
372 * @return object/booleanDevuelve el objeto con sus datos correspondientes, de lo
contrario, false Si no se encontró el registro
373 */
374 public function getByMac($mac)
375 {
376     $result = false;

```

```

377
378     $this-> db-> select('tes_tableta.*', sis_estado_tableta.descripcion AS status,
379     tes_tipo_censo.descripcion AS tipo_censo');
380     $this-> db-> from('tes_tableta');
381     $this-> db-> join('sis_estado_tableta', 'tes_tableta.id_tes_estado_tableta =
382     sis_estado_tableta.id', 'left');
383     $this-> db-> join('tes_tipo_censo', 'tes_tableta.id_tipo_censo =
384     tes_tipo_censo.id', 'left');
385     $this-> db-> where('tes_tableta.mac', $mac);
386
387     $query = $this-> db-> get();
388     $result = $query-> row();
389
390     if($this-> db-> _error_number()) {
391         $this-> error = true;
392         $this-> msg_error_usr = 'No se encontraron registros en la busqueda';
393         $this-> msg_error_log = '('.__METHOD__.' => ' . $this-> db-
394     > _error_number().': ' . $this-> db-> _error_message());
395         throw new Exception();
396     }
397
398     if(!empty($result)) {
399         $this-> id = $result-> id;
400         $this-> mac = $result-> mac;
401         $this-> id_version = $result-> id_version;
402         $this-> ultima_actualizacion = $result-> ultima_actualizacion;
403         $this-> usuarios_asignados = $result-> usuarios_asignados;
404         $this-> id_tes_estado_tableta = $result-> id_tes_estado_tableta;
405         $this-> id_tipo_censo = $result-> id_tipo_censo;
406         $this-> id_asu_um = $result-> id_asu_um;
407         $this-> periodo_esq_inc = $result-> periodo_esq_inc;
408     }
409
410     return $result;
411 }
412 /**
413 * Obtiene todos los registros de la tabla
414 *
415 * @access public
416 * @param int $offset Establece el desplazamiento del primer registro a devolver,
417 *                     si se define solo el valor de offset
418 *                     el valor especifica el n mero de registros a retornar desde el
419 *                     comienzo del conjunto de resultados.
420 * @param int $row_count Establece la cantidad de registros a devolver
421 * @return array object Devuelve un arreglo de objetos obtenidos de la base de datos
422 */
423 public function getAll($offset=null, $row_count=null, $filtro)
424 {
425     $result = 0;
426     $idsAsuUM = array();
427
428     /* *****/
429     if(!empty($filtro['um'])) {
430         $idsAsuUM = array($filtro['um']);
431     } else if(!empty($filtro['local'])) {
432         $queryIdsAsu = $this-> db-> query('SELECT id FROM asu_arbol_segmentacion WHERE
433         id_raiz=1 AND id_padre=' . $filtro['local']);
434         $resultIdsAsu = $queryIdsAsu-> result();
435
436         if (!$resultIdsAsu){
437             return null;
438         }
439
440         foreach ($resultIdsAsu as $tempAsu) {
441             $idsAsuUM[] = $tempAsu-> id;
442         }
443     } else if(!empty($filtro['muni'])) {
444         $queryIdsAsu = $this-> db-> query('SELECT id FROM asu_arbol_segmentacion WHERE
445         id_padre IN
446         (SELECT id FROM asu_arbol_segmentacion WHERE
447         id_raiz=1 AND id_padre=' . $filtro['muni']. ')');
448         $resultIdsAsu = $queryIdsAsu-> result();
449
450         if (!$resultIdsAsu){
451             return null;
452         }
453     }

```

```

449         foreach ($resultIdsAsu as $tempAsu) {
450             $idsAsuUM[] = $tempAsu-> id;
451         }
452     /* **** */
453 } else if(!empty($filtro['juris'])) {
454     $queryIdsAsu = $this-> db-> query('SELECT id FROM asu_arbol_segmentacion WHERE
id_padre IN
455                                         (SELECT id FROM asu_arbol_segmentacion WHERE
id_padre IN
456                                         (SELECT id FROM asu_arbol_segmentacion
WHERE id_raiz=1 AND id_padre='.$filtro['juris'].'));
457     $resultIdsAsu = $queryIdsAsu-> result();
458
459     if (!$resultIdsAsu){
460         return null;
461     }
462
463     foreach ($resultIdsAsu as $tempAsu) {
464         $idsAsuUM[] = $tempAsu-> id;
465     }
466     /* **** */
467 } else if(!empty($filtro['edo'])) {
468     $queryIdsAsu = $this-> db-> query('SELECT id FROM asu_arbol_segmentacion WHERE
id_padre IN
469                                         (SELECT id FROM asu_arbol_segmentacion WHERE
id_padre IN
470                                         (SELECT id FROM asu_arbol_segmentacion
WHERE id_raiz=1 AND id_padre='.$filtro['edo'].'));
471     $resultIdsAsu = $queryIdsAsu-> result();
472
473     if (!$resultIdsAsu){
474         return null;
475     }
476
477     foreach ($resultIdsAsu as $tempAsu) {
478         $idsAsuUM[] = $tempAsu-> id;
479     }
480 }
481
482
483     $this-> db-> select('tes_tableta.*', sis_estado_tableta.descripcion AS status,
tes_tipo_censo.descripcion AS tipo_censo');
484     $this-> db-> from('tes_tableta');
485     $this-> db-> join('sis_estado_tableta', 'tes_tableta.id_tes_estado_tableta =
sis_estado_tableta.id', 'left');
486     $this-> db-> join('tes_tipo_censo', 'tes_tableta.id_tipo_censo =
tes_tipo_censo.id', 'left');
487
488     if(!empty($idsAsuUM))
489         $this-> db-> where('tes_tableta.id_asu_um IN ('.$implode(',',$idsAsuUM).')');
490
491     if(!empty($offset) && !empty(
492         $this-> db-> limit($offset, $row_count));
493     else if (!empty($offset))
494         $this-> db-> limit($offset);
495
496     $query = $this-> db-> get();
497     $result = $query-> result();
498
499     if($this-> db-> _error_number()) {
500         $this-> error = true;
501         $this-> msg_error_usr = 'No se encontraron registros en la busqueda';
502         $this-> msg_error_log = '('._METHOD.') => ' . $this-> db-
> _error_number().': '.$this-> db-> _error_message();
503         throw new Exception();
504     } else if(empty ($result)) {
505         $this-> msg_error_usr = 'No se encontraron registros en la busqueda';
506     }
507
508     return $result;
509 }
510
511 /**
512 * Obtiene el numero total de registros en la tabla
513 * en caso de existir filtros, estos son aplicados a la consulta
514 *
515 * @access public
516 * @return int
517 */

```

```
518     public function getNumRows()
519     {
520         $result = 0;
521
522         $result = $this-> db-> count_all_results('tes_tableta');
523
524         if($this-> db-> _error_number()) {
525             $this-> error = true;
526             $this-> msg_error_usr = 'No se encontraron registros en la busqueda';
527             $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
528 > _error_number().': '.$this-> db-> _error_message();
529             throw new Exception();
530         } else if(empty ($result)) {
531             $this-> msg_error_usr = 'No se encontraron registros en la busqueda';
532         }
533
534         return $result;
535     }
536 ?>
```

# Archivo fuente para tipo\_censo\_model.php

*La documentación para este archivo está disponible en [tipo\\_censo\\_model.php](#)*

```
1  <?php
2
3  /**
4   * Modelo Tipo_censo
5   *
6   * @package    TES
7   * @subpackage Modelo
8   * @author     Pascual
9   * @created    2013-12-02
10  */
11 class Tipo_censo_model extends CI_Model
12 {
13     /**
14      * @access private
15      * @var int(11)
16      */
17     private $id;
18
19     /**
20      * @access private
21      * @var varchar(20)
22      */
23     private $descripcion;
24
25
26     /**
27      * Estas variables no pertenecen a la tabla *
28      * ****
29     */
30     /**
31      * @access private
32      * @var boolean
33      */
34     private $error;
35
36     /**
37      * @access private
38      * @var string
39      */
40     private $msg_error_usr;
41
42     /**
43      * @access private
44      * @var string
45      */
46     private $msg_error_log;
47
48
49     public function __construct()
50     {
51         parent::__construct();
52         $this-> load-> database();
53         $this-> error = false;
54         $this-> msg_error_usr = '';
55         $this-> msg_error_log = '';
56
57         if( !$this-> db-> conn_id ) {
58             throw new Exception ('ERROR: No se puede conectar con la Base de Datos');
59         }
60     }
61
62     public function getId()
63     {
64         return $this-> id;
```

```

65     }
66
67     public function getDescripcion()
68     {
69         return $this-> descripcion;
70     }
71
72     public function setId($id)
73     {
74         return $this-> id = $id;
75     }
76
77     public function setDescripcion($descripcion) {
78         $this-> mac = $descripcion;
79     }
80
81     /**
82      * Devuelve el mensaje de error,
83      * en caso de existir un error despues de ejecutar un metodo,
84      * de lo contrario false
85      *
86      * @access public
87      * @param string $type usr si se quiere devolver el mensaje de error a mostrar en la vista,
88      *                      log obtiene el mensaje de error con mas detalles para depuración,
89      *                      valor por defecto usr
90      * @return boolean/string
91      */
92     public function getMsgError($type = 'usr')
93     {
94         if($this-> error) {
95             if($type == 'usr')
96                 return $this-> msg_error_usr;
97             else if($type == 'log')
98                 return $this-> msg_error_log;
99             else
100                return false;
101        }
102
103        return false;
104    }
105
106    /**
107     * Obtiene los datos del registro que tiene el ID especificado
108     *
109     * @access public
110     * @param int $id           Si no se establece el valor de ID, se toma el valor del objeto
actual
111     * @return object/booleanDevuelve el objeto con sus datos correspondientes, de lo
contrario, false Si no se encontró el registro
112     */
113    public function getById($id)
114    {
115        $result = false;
116
117        $query = $this-> db-> get_where('tes_tipo_censo', array('id' => $id));
118        $result = $query-> row();
119
120        if($this-> db-> _error_number()) {
121            $this-> error = true;
122            $this-> msg_error_usr = 'No se encontraron registros en la búsqueda';
123            $this-> msg_error_log = '(' . __METHOD__ . ') => ' . $this-> db-
> _error_number() . ':' . $this-> db-> _error_message();
124            throw new Exception();
125        }
126
127        if(!empty($result)) {
128            $this-> id = $id;
129            $this-> descripcion = $result-> descripcion;
130        }
131
132        return $result;
133    }
134
135    /**
136     * Obtiene todos los registros de la tabla
137     *
138     * @access public
139     * @return array object   Devuelve un arreglo de objetos obtenidos de la base de datos
140     */
141    public function getAll()

```

```

142     {
143         $result = 0;
144
145         $query = $this-> db-> get('tes_tipo_censo');
146         $result = $query-> result();
147
148         if($this-> db-> _error_number()) {
149             $this-> error = true;
150             $this-> msg_error_usr = 'No se encontraron registros en la busqueda';
151             $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
152 > _error_number().': '.$this-> db-> _error_message();
153             throw new Exception();
154         } else if(empty ($result)) {
155             $this-> msg_error_usr = 'No se encontraron registros en la busqueda';
156         }
157
158         return $result;
159     }
160 }
161 ?>

```

# Archivo fuente para usuario\_tableta\_model.php

*La documentación para este archivo está disponible en [usuario\\_tableta\\_model.php](#)*

```
1  <?php
2
3  /**
4   * Modelo Estado_tableta
5   *
6   * @package    TES
7   * @subpackage Modelo
8   * @author     Pascual
9   * @created    2013-11-27
10  */
11 class Usuario_tableta_model extends CI_Model
12 {
13     /**
14      * @access private
15      * @var int(11)
16      */
17     private $id_tes_tableta;
18
19     /**
20      * @access private
21      * @var int(11)
22      */
23     private $id_usuario;
24
25
26     /**
27      * Estas variables no pertenecen a la tabla *
28      * ****
29     */
30     /**
31      * @access private
32      * @var boolean
33      */
34     private $error;
35
36     /**
37      * @access private
38      * @var string
39      */
40     private $msg_error_usr;
41
42     /**
43      * @access private
44      * @var string
45      */
46     private $msg_error_log;
47
48
49     public function __construct()
50     {
51         parent::__construct();
52         $this-> load-> database();
53         $this-> error = false;
54         $this-> msg_error_usr = '';
55         $this-> msg_error_log = '';
56
57         if( !$this-> db-> conn_id ) {
58             throw new Exception ('ERROR: No se puede conectar con la Base de Datos');
59         }
60     }
61
62     public function getTableta()
63     {
64         return $this-> id_tes_tableta;
```

```

65
66
67     public function getUsuario() {
68         return $this-> id_usuario;
69     }
70
71     public function setTableta($id_tes_tableta)
72     {
73         return $this-> id_tes_tableta = $id_tes_tableta;
74     }
75
76     public function setUsuario($id_usuario) {
77         $this-> id_usuario = $id_usuario;
78     }
79
80     /**
81      * Devuelve el mensaje de error,
82      * en caso de existir un error despues de ejecutar un metodo,
83      * de lo contrario false
84      *
85      * @access public
86      * @param string $type usr si se quiere devolver el mensaje de error a mostrar en la vista,
87      *                      log obtiene el mensaje de error con mas detalles para depuración,
88      *                      valor por defecto usr
89      * @return boolean/string
90      */
91     public function getMsgError($type = 'usr')
92     {
93         if($this-> error) {
94             if($type == 'usr')
95                 return $this-> msg_error_usr;
96             else if($type == 'log')
97                 return $this-> msg_error_log;
98             else
99                 return false;
100        }
101
102        return false;
103    }
104
105    /**
106     * Obtiene el id de todos los usuarios asignados a la tableta especificada
107     *
108     * @access public
109     * @param int $id_tes_tableta Si no se establece el valor de ID, se toma el valor del
110     * objeto actual
111     * @return object/boolean Devuelve el objeto con sus datos correspondientes, de lo
112     * contrario, false Si no se encontró el registro
113     */
114     public function getUsuariosByTableta($id_tes_tableta = null)
115     {
116         $tableta = is_null($id_tes_tableta) ? $this-> id_tes_tableta : $id_tes_tableta;
117
118         $query = $this-> db-> get_where('tes_usuario_x_tableta', array('id_tes_tableta' => $tableta));
119         $result = $query-> result();
120
121         if($this-> db-> _error_number()) {
122             $this-> error = true;
123             $this-> msg_error_usr = 'No se encontraron registros en la búsqueda';
124             $this-> msg_error_log = '(' . __METHOD__ . ') => ' . $this-> db-> _error_number(). ': ' . $this-> db-> _error_message();
125             throw new Exception();
126         }
127
128         return $result;
129     }
130
131     /**
132      * Obtiene el id de todas las tabletas relacionadas con el usuario especificado
133      *
134      * @access public
135      * @param int $id_usuario Si no se establece el valor de ID, se toma el valor del objeto
136      * actual
137      * @return object/boolean Devuelve el objeto con sus datos correspondientes, de lo
138      * contrario, false Si no se encontró el registro
139      */
140     public function getTabletasByUsuario($id_usuario = null)
141     {
142         $usuario = is_null($id_usuario) ? $this-> id_usuario : $id_usuario;

```

```

139
140     $query = $this-> db-> get_where('tes_usuario_x_tableta', array('id_usuario' =>
141 $usuario));
142     $result = $query-> result();
143
144     if($this-> db-> _error_number()) {
145         $this-> error = true;
146         $this-> msg_error_usr = 'No se encontraron registros en la busqueda';
147         $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
> _error_number().': '.$this-> db-> _error_message();
148         throw new Exception();
149     }
150
151     return $result;
152 }
153 /**
154 * Elimina la relacion entre usuario y tableta
155 *
156 * @access public
157 * @param int $id_usuario
158 * @param int $id_tableta Si no se proporciona el parametro, se toma el id del objeto
159 * @return boolean Devuelve true Si se elimino el registro exitosamente, false
160 en caso contrario
161 */
162 public function delete($id_usuario, $id_tableta=null)
163 {
164     $result = false;
165
166     $id_tableta = is_null($id_tableta) ? $this-> id_tes_tableta : $id_tableta;
167
168     if(is_array($id_usuario)) {
169         // Eliminar un conjunto de registros
170         foreach ($id_usuario as $idx) {
171             $this-> db-> where('id_tes_tableta', $id_tableta);
172             $this-> db-> where('id_usuario', $idx);
173             $result = $this-> db-> delete('tes_usuario_x_tableta');
174
175             if(empty($result)) {
176                 $this-> error = true;
177                 $this-> msg_error_usr = 'No se puede eliminar el registro';
178                 $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
> _error_number().': '.$this-> db-> _error_message();
179                 throw new Exception();
180             }
181         }
182     } else {
183         // Eliminar un solo registro
184         $this-> db-> where('id_tes_tableta', $id_tableta);
185         $this-> db-> where('id_usuario', $id_usuario);
186         $result = $this-> db-> delete('tes_usuario_x_tableta');
187
188         if(empty($result)) {
189             $this-> error = true;
190             $this-> msg_error_usr = 'No se puede eliminar el registro';
191             $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
> _error_number().': '.$this-> db-> _error_message();
192             throw new Exception();
193         }
194     }
195
196     return $result;
197 }
198 /**
199 * Inserta en la base de datos, la informacion contenida en el objeto
200 *
201 * @access public
202 * @return boolean false Si no se ejecutó la inserción, true si se ejecutó la inserción
203 */
204 public function insert($id_usuario=null, $id_tableta=null)
205 {
206     $result = false;
207     $data = array();
208
209     $data['id_usuario'] = $this-> id_usuario ? $this-> id_usuario : $id_usuario;
210     $data['id_tes_tableta'] = $this-> id_tes_tableta ? $this-> id_tes_tableta :
211 $id_tableta;

```

```

212     $result = $this-> db-> insert('tes_usuario_x_tableta', $data);
213
214     if( $this-> db-> _error_number() ) {
215         $this-> error = true;
216         $this-> msg_error_usr = 'No se puede insertar el registro';
217         $this-> msg_error_log = '('.__METHOD__.') => ' . $this-> db-
218 > _error_number().': '.$this-> db-> _error_message();
219         throw new Exception();
220     } else {
221         $this-> id_usuario = $data['id_usuario'];
222         $this-> id_tes_tableta = $data['id_tes_tableta'];
223     }
224
225     $this-> db-> update('tes_tableta', array('usuarios_asignados'=> 1), 'id =
226 .' . $data['id_tes_tableta']);
227
228     return $result;
229 }
230 ?>

```

# Indice

## A

<a href="#">ArbolSegmentacion_model::getById()</a>	146
Obtener el elemento del ASU por medio de su ID	
<a href="#">ArbolSegmentacion_model::convertType()</a>	146
Convierte el tipo de arreglo para enviarlo como se debe recibir en el cliente de Javascript	
<a href="#">ArbolSegmentacion_model::getChildrenFromId()</a>	147
Accion para devolver los hijos de un elemento en el ASU a partir de su ID	
<a href="#">ArbolSegmentacion_model::getChildrenFromLevel()</a>	147
Accion para devolver el esquema completo del ASU a partir de un nivel especificado, niveles omitidos y elementos preseleccionados	
<a href="#">ArbolSegmentacion_model::getDataKeyValue()</a>	148
Accion para obtener los registros de un ASU determinado en cierto nivel y con un ID de filtro	
<a href="#">ArbolSegmentacion_model::getCluesFromId()</a>	148
*	
<a href="#">ArbolSegmentacion_model</a>	146
Modelo ArbolSegmentacion	
<a href="#">Ageb_model::searchUM()</a>	145
Devuelve la informació n de una UM de acuerdo a su localidad y ageb	
<a href="#">Ageb_model</a>	144
Modelo Ageb	
<a href="#">Accion_model::update()</a>	143
Actualiza el objeto actual en la base de datos	
<a href="#">Ageb_model::getMsgError()</a>	144
Devuelve el mensaje de error, en caso de existir un error despues de ejecutar un metodo, de lo contrario false	
<a href="#">Ageb_model::process()</a>	144
Inserta los registros contenidos en la tabla cat_poblacion a la tablaasu_poblacion	
<a href="#">Ageb_model::searchageb()</a>	145
Devuelve una lista de AGEBS de la localidad pasada como parámetro	
<a href="#">ArbolSegmentacion_model::getDescripcionById()</a>	148
Accion para obtener la descripcion e informació n adicional del elemento en el ASU	
<a href="#">ArbolSegmentacion_model::getListChildrenLevel()</a>	149
<a href="#">accion.php</a>	439
Código fuente	
<a href="#">ayuda.php</a>	438
Código fuente	
<a href="#">accion_model.php</a>	540
Código fuente	
<a href="#">ageb_model.php</a>	544
Código fuente	
<a href="#">arbolegmentacion_model.php</a>	547
Código fuente	

<a href="#">ArbolSegmentacion_model::addSelectedItems()</a>	152
<a href="#">ArbolSegmentacion_model::addSelectedItems()</a>	151
<a href="#">ArbolSegmentacion_model::getTree()</a>	150
<i>Regresa el objeto del arbol de segmentacion</i>	
<a href="#">ArbolSegmentacion_model::getMsgError()</a>	149
<i>Devuelve los mensajes de error en caso de ocurrir alguna excepción</i>	
<i>'usr' devuelve el mensaje para la vista de usuario</i>	
<i>'log' devuelve el mensaje para el log de errores</i>	
<a href="#">ArbolSegmentacion_model::getTreeBlock()</a>	150
<i>Accion para devolver un bloque del ASU a partir de un nivel especificado o una clave seleccionada, niveles omitidos y elementos preseleccionados</i>	
<a href="#">ArbolSegmentacion_model::getTreeBlockData()</a>	150
<i>Regresa el objeto del arbol de segmentacion por nivel o hijos de un elemento seleccionado</i>	
<a href="#">ArbolSegmentacion_model::getUMParentsById()</a>	151
<i>Regresa la informacion de los padres de una unidad medica en el ASU</i>	
<a href="#">Accion_model::setRows()</a>	143
<a href="#">Accion_model::setOffset()</a>	143
<a href="#">Ayuda::index()</a>	76
<i>Función que renderiza el contenido de la ayuda dependiendo de la sección donde se encuentre</i>	
<a href="#">Ayuda</a>	76
<i>Controlador Ayuda</i>	
<a href="#">accion_model.php</a>	119
<a href="#">ageb_model.php</a>	120
<a href="#">Accion_model</a>	139
<i>Modelo Accion</i>	
<a href="#">arbolsegmentacion_model.php</a>	121
<a href="#">Accion::view()</a>	76
<i>Acción para visualizar información de una acción específica, obtiene el objeto acción por medio del id proporcionado.</i>	
<a href="#">Accion::update()</a>	75
<i>Acción para preparar la actualización de una acción ya existente,</i>	
<a href="#">Accion</a>	74
<i>Controlador Accion</i>	
<a href="#">accion.php</a>	59
<a href="#">Accion::delete()</a>	74
<i>Acción para eliminar una acción, recibe el id de la acción a eliminar</i>	
<a href="#">Accion::index()</a>	75
<i>Acción por default del controlador, carga la lista</i>	
<a href="#">Accion::insert()</a>	75
<i>Acción para preparar la inserción de nuevas acciones , realiza la validación del formulario del lado cliente</i>	
<a href="#">Accion_model::delete()</a>	139
<i>Elimina el registro actual de la base de datos</i>	
<a href="#">Accion_model::getAll()</a>	139
<i>Devuelve todos los registros de la tabla acciones</i>	
<a href="#">Accion_model::setDescripcion()</a>	142
<a href="#">Accion_model::insert()</a>	141
<i>Inserta en la tabla accion la información contenida en el objeto</i>	
<a href="#">Accion_model::setId()</a>	142
<a href="#">Accion_model::setMetodo()</a>	142
<a href="#">Accion_model::setNombre()</a>	142

<a href="#">Accion_model::getNumRows()</a>	141
<i>Devuelve el numero de registros</i>	
<a href="#">Accion_model::getNombre()</a>	141
<a href="#">Accion_model::getDescripcion()</a>	140
<a href="#">Accion_model::getById()</a>	140
<i>Devuelve la informacion de una accion por su ID</i>	
<a href="#">Accion_model::getId()</a>	140
*****	
<a href="#">Accion_model::getMetodo()</a>	140
<a href="#">Accion_model::getMsgError()</a>	141
<i>Devuelve los mensajes de error en caso de ocurrir alguna excepcion</i>	
' <i>usr</i> ' devuelve el mensaje para la vista de usuario	
' <i>log</i> ' devuelve el mensaje para el log de errores	
<a href="#">ayuda.php</a>	58

## B

<a href="#">Bitacora_model::setFecha_hora()</a>	156
<a href="#">Bitacora_model::setId()</a>	157
<a href="#">Bitacora_model::resetFilter()</a>	156
<i>Elimina todos los filtros registrados</i>	
<a href="#">Bitacora_model::insert()</a>	156
<i>Inserta a la bitacora la informacion porporcionada</i>	
<a href="#">Bitacora_model::getNumRows()</a>	155
<i>Obtiene el numero total de registros en la tabla Bitacora en caso de existir filtros, estos son aplicados a la consulta</i>	
<a href="#">Bitacora_model::getParametros()</a>	155
<a href="#">Bitacora_model::setId_accion()</a>	157
<a href="#">Bitacora_model::setId_controlador()</a>	157
<a href="#">bitacora.php</a>	443
<i>Código fuente</i>	
<a href="#">bitacora_model.php</a>	563
<i>Código fuente</i>	
<a href="#">Bitacora_model::update()</a>	158
<i>Actualiza los datos del objeto actual</i>	
<a href="#">Bitacora_model::setParametros()</a>	158
<a href="#">Bitacora_model::setId_controlador_accion()</a>	157
<a href="#">Bitacora_model::setId_usuario()</a>	158
<a href="#">Bitacora_model::getMsgError()</a>	155
<i>Devuelve el mensaje de error, en caso de existir un error despues de ejecutar un metodo, de lo contrario false</i>	
<a href="#">Bitacora_model::getId_usuario()</a>	155
<a href="#">bitacora_model.php</a>	122
<a href="#">Bitacora_model</a>	152
<i>Modelo Bitacora</i>	
<a href="#">Bitacora::view()</a>	78
<i>Muestra los datos del registro especificado por el id</i>	
<a href="#">Bitacora::validateExistUsuario()</a>	78
<i>callback utilizado por las acciones create y update para validar la existencia de un usuario</i>	
<a href="#">Bitacora</a>	77
<i>Controlador Bitacora</i>	

<a href="#">Bitacora::index()</a>	77
<i>Lista todos los registros de la bitacora, con su correspondiente paginación</i>	
<a href="#">Bitacora_model::addFilter()</a>	152
<i>Agrega una nueva regla de filtrado al arreglo de filtros</i>	
<a href="#">Bitacora_model::delete()</a>	153
<i>Elimina el registro actual de la base de datos</i>	
<a href="#">Bitacora_model::getId()</a>	154
<a href="#">Bitacora_model::getId_controlador_accion()</a>	155
<a href="#">Bitacora_model::getFecha_hora()</a>	154
<a href="#">Bitacora_model::getById()</a>	154
<i>Obtiene los datos del registro de la bitacora que tiene el ID especificado</i>	
<a href="#">Bitacora_model::deleteByFilter()</a>	153
<i>Elimina el conjunto de registros que cumplen con el o los criterios de filtrado</i>	
<a href="#">Bitacora_model::getAll()</a>	153
<i>Obtiene todos los registros de la tabla Bitacora en caso de existir filtros, estos son aplicados a la consulta</i>	
<a href="#">bitacora.php</a>	60

## C

<a href="#">Catalogo_model::updateComentario()</a>	170
<i>Cambia el comentario de la tabla indicada</i>	
<a href="#">Catalogo_model::setRows()</a>	170
<a href="#">Catalogo_model::setOffset()</a>	169
<a href="#">Catalogo_model::setNombre()</a>	169
<a href="#">Catalogo_x_raiz_model</a>	170
<i>Modelo Raiz_x_Catalogo</i>	
<a href="#">constructor Catalogo_x_raiz_model::__construct()</a>	171
<a href="#">Catalogo_x_raiz_model::getId()</a>	172
<i>Devuelve la información de un catalogo x accion por su ID</i>	
<a href="#">Catalogo_x_raiz_model::getByArbol()</a>	171
<i>Devuelve todos los registros de la tabla raiz_x_catalogo de una raiz determinada</i>	
<a href="#">Catalogo_x_raiz_model::delete()</a>	171
<i>Elimina el registro actual de la base de datos</i>	
<a href="#">Catalogo_x_raiz_model::check()</a>	171
<i>Revisa inconsistencias en los datos de un catalogo x raiz con respecto a su catalogo padre</i>	
<a href="#">Catalogo_model::setLlave()</a>	169
<a href="#">Catalogo_model::setId()</a>	169
<a href="#">Catalogo_model::getLlave()</a>	167
<a href="#">Catalogo_model::getId()</a>	167
*****	
<a href="#">Catalogo_model::getCampos()</a>	166
<a href="#">Catalogo_model::getByName()</a>	166
<i>Devuelve la informacion de un catalogo por su nombre</i>	
<a href="#">Catalogo_model::getMsgError()</a>	167
<i>Devuelve los mensajes de error en caso de ocurrir alguna excepción 'usr' devuelve el mensaje para la vista de usuario 'log' devuelve el mensaje para el log de errores</i>	
<a href="#">Catalogo_model::getNombre()</a>	167
<a href="#">Catalogo_model::setCampos()</a>	168
<a href="#">Catalogo_model::insert()</a>	168

<i>Inserta en la base datos el catálogo y obtiene los datos de la tabla temporal</i>	
<a href="#">Catalogo_model::getNumRows()</a>	168
<i>Devuelve el numero de registros</i>	
<a href="#">Catalogo_x_raiz_model::getByNivel()</a>	172
<i>Devuelve el catalogo padre de un elemento raiz_x_catalogo</i>	
<a href="#">Catalogo_x_raiz_model::getColumnaDescripcion()</a>	173
<a href="#">Catalogo_x_raiz_model::setId()</a>	176
<a href="#">Catalogo_x_raiz_model::setGrado()</a>	176
<a href="#">Catalogo_x_raiz_model::setColumnaLlave()</a>	175
<a href="#">Catalogo_x_raiz_model::setColumnaDescripcion()</a>	175
<a href="#">Catalogo_x_raiz_model::setIdRaiz()</a>	176
<a href="#">Catalogo_x_raiz_model::setRelacionHijo()</a>	177
<a href="#">constructor Cie10_model:: construct()</a>	178
<a href="#">Cie10_model</a>	178
<i>Modelo Cie10</i>	
<a href="#">Catalogo_x_raiz_model::setTablaCatalogo()</a>	177
<a href="#">Catalogo_x_raiz_model::setRelacionPadre()</a>	177
<a href="#">Catalogo_x_raiz_model::insert()</a>	175
<i>Inserta en la base datos la información del objeto actual</i>	
<a href="#">Catalogo_x_raiz_model::getTablaCatalogo()</a>	175
<a href="#">Catalogo_x_raiz_model::getIdRaiz()</a>	173
<a href="#">Catalogo_x_raiz_model::getId()</a>	173
*****	
<a href="#">Catalogo_x_raiz_model::getGrado()</a>	173
<a href="#">Catalogo_x_raiz_model::getColumnaLlave()</a>	173
<a href="#">Catalogo_x_raiz_model::getMsgError()</a>	173
<i>Devuelve los mensajes de error en caso de ocurrir alguna excepción</i>	
<i>'usr' devuelve el mensaje para la vista de usuario</i>	
<i>'log' devuelve el mensaje para el log de errores</i>	
<a href="#">Catalogo_x_raiz_model::getNivel()</a>	174
<i>Devuelve el nivel siguiente para la tabla raiz_x_catalogo de un arbol determinado</i>	
<a href="#">Catalogo_x_raiz_model::getRelations()</a>	174
<i>Devuelve las relaciones de una raiz x catalogo</i>	
<a href="#">Catalogo_x_raiz_model::getRelacionPadre()</a>	174
<a href="#">Catalogo_x_raiz_model::getRelacionHijo()</a>	174
<a href="#">Catalogo_model::getAllData()</a>	166
<i>Devuelve los datos de un catalogo pasado como parametro</i>	
<a href="#">Catalogo_model::getAll()</a>	165
<i>Devuelve una lista con los catalogos existentes en la DB</i>	
<a href="#">controlador_model.php</a>	128
<a href="#">controladoraccion_model.php</a>	127
<a href="#">cie10_model.php</a>	126
<a href="#">catalogo_x_raiz_model.php</a>	125
<a href="#">constructor Accion_model:: construct()</a>	139
<a href="#">constructor Ageb_model:: construct()</a>	144
<a href="#">constructor CatalogoCsv_model:: construct()</a>	159
<a href="#">CatalogoCsv_model</a>	159
<i>Modelo CatalogoCsv</i>	
<a href="#">constructor Bitacora_model:: construct()</a>	152
<a href="#">constructor ArbolSegmentacion_model:: construct()</a>	146
<a href="#">catalogo_model.php</a>	124
<a href="#">catalogocsv_model.php</a>	123

<a href="#">constructor Grupo:: construct()</a>	100
<a href="#">constructor Errorlog:: construct()</a>	99
<a href="#">constructor Entorno:: construct()</a>	96
<a href="#">Controlador::view()</a>	96
<i>Acción para visualizar información de un controlador específico, obtiene el objeto controlador por medio del id proporcionado.</i>	
<a href="#">constructor Menu:: construct()</a>	103
<a href="#">constructor Permiso:: construct()</a>	105
<a href="#">constructor Usuario:: construct()</a>	114
<a href="#">constructor ReglaVacuna:: construct()</a>	111
<a href="#">constructor Raiz:: construct()</a>	106
<a href="#">CatalogoCsv_model::activaEnCatalogo()</a>	159
<i>Acción para activar o desactivar registros de catalogos como el de EDA, IRA y Consultas</i>	
<a href="#">CatalogoCsv_model::checkPk()</a>	159
<i>Revisa en la base de datos por registros duplicados en los campos pasados por parametro</i>	
<a href="#">CatalogoCsv_model::setRows()</a>	164
<a href="#">CatalogoCsv_model::setOffset()</a>	163
<a href="#">CatalogoCsv_model::setNombre()</a>	163
<a href="#">CatalogoCsv_model::setLlave()</a>	163
<a href="#">Catalogo_model</a>	164
<i>Modelo Catalogo</i>	
<a href="#">constructor Catalogo_model:: construct()</a>	164
<a href="#">Catalogo_model::delete()</a>	165
<i>Elimina el registro actual de la base de datos</i>	
<a href="#">Catalogo_model::checkTypeData()</a>	165
<i>Revisa en la base de datos por registros que no coincidan con el tipo de dato pasado como parametro en el campo indicado</i>	
<a href="#">Catalogo_model::checkPk()</a>	164
<i>Revisa en la base de datos por registros duplicados en los campos pasados por parametro</i>	
<a href="#">CatalogoCsv_model::setId()</a>	163
<a href="#">CatalogoCsv_model::setCampos()</a>	162
<a href="#">CatalogoCsv_model::getCampos()</a>	161
<a href="#">CatalogoCsv_model::getByName()</a>	160
<i>Devuelve la informacion de un catalogo por su nombre</i>	
<a href="#">CatalogoCsv_model::getAllData()</a>	160
<i>Devuelve los datos de un catalogo pasado como parametro</i>	
<a href="#">CatalogoCsv_model::getAll()</a>	160
<i>Devuelve una lista con los catalogos existentes en la DB</i>	
<a href="#">CatalogoCsv_model::getId()</a>	161
*****	
<a href="#">CatalogoCsv_model::getLlave()</a>	161
<a href="#">CatalogoCsv_model::getNumRows()</a>	162
<i>Devuelve el numero de registros</i>	
<a href="#">CatalogoCsv_model::getNombre()</a>	162
<a href="#">CatalogoCsv_model::getMsgError()</a>	161
<i>Devuelve los mensajes de error en caso de ocurrir alguna excepción</i>	
<i>'usr' devuelve el mensaje para la vista de usuario</i>	
<i>'log' devuelve el mensaje para el log de errores</i>	
<a href="#">Cie10_model::activaEnCatalogo()</a>	178
<i>Acción para activar o desactivar registros de catalogos como el de EDA, IRA y Consultas</i>	
<a href="#">Cie10_model::agregaEnCatalogo()</a>	178

*Accion para agregar registros del CIE10 a otros catalogos como el de EDA, IRA y Consultas*

<u>constructor Usuario_model:: construct()</u>	236
<u>constructor ReglaVacuna_model:: construct()</u>	227
<u>constructor Raiz_model:: construct()</u>	224
<u>constructor Poblacion_model:: construct()</u>	222
<u>constructor Enrolamiento:: construct()</u>	256
<u>constructor Notificacion:: construct()</u>	267
<u>constructor Servicios:: construct()</u>	274
<u>constructor Semana_nacional:: construct()</u>	272
<u>constructor Reporte_sincronizacion:: construct()</u>	270
<u>constructor Reporteador:: construct()</u>	269
<u>constructor Permiso_model:: construct()</u>	219
<u>constructor Menu_model:: construct()</u>	211
<u>Controlador_model::update()</u>	192
<i>Actualiza el objeto actual en la base de datos</i>	
<u>Controlador_model::setRows()</u>	192
<u>Controlador_model::setOffset()</u>	192
<u>Controlador_model::setNombre()</u>	192
<u>constructor Entorno_model:: construct()</u>	193
<u>constructor Errorlog_model:: construct()</u>	199
<u>constructor Hemoglobina_model:: construct()</u>	210
<u>constructor Grupo_model:: construct()</u>	206
<u>constructor Georeferencia_model:: construct()</u>	204
<u>constructor Tableta:: construct()</u>	280
<u>constructor Usuario_tableta:: construct()</u>	283
<u>catalogocsv_model.php</u>	571
<i>Código fuente</i>	
<u>controlador.php</u>	483
<i>Código fuente</i>	
<u>cie10.php</u>	474
<i>Código fuente</i>	
<u>catalogo_x_raiz.php</u>	470
<i>Código fuente</i>	
<u>catalogo_model.php</u>	576
<i>Código fuente</i>	
<u>catalogo_x_raiz_model.php</u>	582
<i>Código fuente</i>	
<u>controlador_model.php</u>	597
<i>Código fuente</i>	
<u>controladoraccion_model.php</u>	594
<i>Código fuente</i>	
<u>cie10_model.php</u>	588
<i>Código fuente</i>	
<u>catalogocsv.php</u>	461
<i>Código fuente</i>	
<u>catalogo.php</u>	449
<i>Código fuente</i>	
<u>constructor Reporteador_model:: construct()</u>	346
<u>constructor Notificacion_model:: construct()</u>	340
<u>constructor Estado_tableta_model:: construct()</u>	338
<u>constructor Enrolamiento_model:: construct()</u>	294
<u>constructor Reporte_censo_nominal:: construct()</u>	350

<u>constructor Semana nacional model:: construct()</u>	353
<u>constructor Usuario_tableta model:: construct()</u>	366
<u>constructor Tipo censo model:: construct()</u>	364
<u>constructor Tableta model:: construct()</u>	357
<u>Controlador_model::setIdEntorno()</u>	191
<u>Controlador_model::setId()</u>	191
<u>Cie10_model::update()</u>	183
<i>Actualiza el objeto actual en la base de datos</i>	
<u>Cie10_model::setRows()</u>	183
<u>Cie10_model::setOffset()</u>	182
<u>Cie10_model::setId()</u>	182
<u>ControladorAccion_model</u>	183
<i>Modelo ControladorAccion</i>	
<u>constructor ControladorAccion_model:: construct()</u>	183
<u>ControladorAccion_model::getMsgError()</u>	185
<i>Devuelve los mensajes de error en caso de ocurrir alguna excepción</i>	
<i>'usr' devuelve el mensaje para la vista de usuario</i>	
<i>'log' devuelve el mensaje para el log de errores</i>	
<u>ControladorAccion_model::getIdByPath()</u>	184
<i>Devuelve el id de una accion por controlador</i>	
<i>de acuerdo al path</i>	
<u>ControladorAccion_model::getId()</u>	184
<i>Devuelve el Id de una accion por controlador de una accion y controlador determinados</i>	
<u>ControladorAccion_model::getById()</u>	184
<i>Devuelve la informacion de una accion por controlador de acuerdo a su Id</i>	
<u>Cie10_model::setDescripcion()</u>	182
<u>Cie10_model::getNumRows()</u>	181
<i>Devuelve el numero de registros</i>	
<u>Cie10_model::getById()</u>	180
<i>Devuelve la informacion de un registro del catalogo cie10 por su ID</i>	
<u>Cie10_model::getAllData()</u>	179
<i>Devuelve los datos de un catalogo pasado como parametro</i>	
<u>Cie10_model::getAll()</u>	179
<i>Devuelve una lista con los registros existentes en el catalogo cie10</i>	
<u>Cie10_model::checkPk()</u>	179
<i>Revisa en la base de datos por registros duplicados en los campos pasados por parametro</i>	
<u>Cie10_model::getCatalogoByName()</u>	180
<i>Devuelve una lista con los registros existentes en el catalogo requerido</i>	
<u>Cie10_model::getData()</u>	180
<i>Devuelve una lista con los registros existentes en el catalogo cie10 omitiendo los ID</i>	
<u>Cie10_model::getMsgError()</u>	181
<i>Devuelve los mensajes de error en caso de ocurrir alguna excepción</i>	
<i>'usr' devuelve el mensaje para la vista de usuario</i>	
<i>'log' devuelve el mensaje para el log de errores</i>	
<u>Cie10_model::getId()</u>	181
*****	
<u>Cie10_model::getDescripcion()</u>	181
<u>ControladorAccion_model::setHelp()</u>	185
<i>Establece el mensaje de ayuda</i>	
<u>Controlador_model</u>	186
<i>Modelo Controlador</i>	
<u>Controlador_model::getNumRows()</u>	189

<i>Devuelve el numero de registros</i>	
<a href="#">Controlador_model::getNombre()</a>	189
<a href="#">Controlador_model::getMsgError()</a>	189
<i>Devuelve los mensajes de error en caso de ocurrir alguna excepción</i>	
'usr' devuelve el mensaje para la vista de usuario	
'log' devuelve el mensaje para el log de errores	
<a href="#">Controlador_model::getIdEntorno()</a>	188
<a href="#">Controlador_model::getPermisos()</a>	190
<i>Devuelve los permisos asignados a un grupo sobre un entorno determinado (Mapea la información de las acciones asignadas a un controlador y los une con los permisos de un grupo sobre esas acciones)</i>	
<a href="#">Controlador_model::insert()</a>	190
<i>Inserta en la tabla controlador, la información contenida en el objeto</i>	
<a href="#">Controlador_model::setDescripcion()</a>	191
<a href="#">Controlador_model::setClase()</a>	190
<a href="#">Controlador_model::setAccion()</a>	190
<a href="#">Controlador_model::getId()</a>	188
*****	
<a href="#">Controlador_model::getDescripcion()</a>	188
<a href="#">Controlador_model::getAccion()</a>	187
<a href="#">Controlador_model::delete()</a>	186
<i>Elimina el registro actual de la base de datos</i>	
<a href="#">Controlador_model::accionesUpdate()</a>	186
<i>Actualiza las acciones asignadas a un controlador</i>	
<a href="#">constructor Controlador_model:: construct()</a>	186
<a href="#">Controlador_model::getAcciones()</a>	187
<i>Devuelve todas las acciones asignadas al controlador por su Id</i>	
<a href="#">Controlador_model::getAll()</a>	187
<i>Devuelve todos los registros de la tabla controlador</i>	
<a href="#">Controlador_model::getClase()</a>	188
<a href="#">Controlador_model::getById()</a>	188
<i>Devuelve la información de un controlador por su ID</i>	
<a href="#">Controlador_model::getByEntorno()</a>	187
<i>Devuelve todos los controladores que pertenecen a un entorno por su Id</i>	
<a href="#">Controlador::update()</a>	95
<i>Acción para preparar la actualización de un controlador ya existente,</i>	
<a href="#">Controlador::insert()</a>	95
<i>Acción para preparar la insercion de nuevos controladores , realiza la validacion del formulario del lado cliente</i>	
<a href="#">CI_Pagination::\$first_link</a>	27
<a href="#">CI_Pagination::\$display_pages</a>	27
<a href="#">CI_Pagination::\$cur_tag_open</a>	27
<a href="#">CI_Pagination::\$cur_tag_close</a>	27
<a href="#">CI_Pagination::\$first_tag_close</a>	27
<a href="#">CI_Pagination::\$first_tag_open</a>	27
<a href="#">CI_Pagination::\$last_link</a>	28
<a href="#">CI_Pagination::\$full_tag_open</a>	28
<a href="#">CI_Pagination::\$full_tag_close</a>	28
<a href="#">CI_Pagination::\$first_url</a>	27
<a href="#">CI_Pagination::\$cur_page</a>	27
<a href="#">CI_Pagination::\$base_url</a>	27
<a href="#">CI_Form_validation:: execute()</a>	25
<i>Executes the Validation routines</i>	

<a href="#">CI_Form_validation::xss_clean()</a>	25
XSS Clean	
<a href="#">CI_Form_validation::valid_ip()</a>	24
Validate IP Address	
<a href="#">CI_Form_validation::valid_emails()</a>	24
Valid Emails	
<a href="#">CI_Form_validation::reduce_array()</a>	25
Traverse a multidimensional \$_POST array index until the data is found	
<a href="#">CI_Form_validation::reset_post_array()</a>	26
Re-populate the _POST array with our finalized and processed data	
<a href="#">CI_Pagination::\$anchor_class</a>	27
<a href="#">CI_Pagination</a>	26
Pagination Class	
<a href="#">CI_Form_validation::translate_fieldname()</a>	26
Translate a field name	
<a href="#">CI_Pagination::\$last_tag_close</a>	28
<a href="#">CI_Pagination::\$last_tag_open</a>	28
<a href="#">CI_Pagination::\$total_rows</a>	29
<a href="#">CI_Pagination::\$suffix</a>	29
<a href="#">CI_Pagination::\$query_string_segment</a>	29
<a href="#">CI_Pagination::\$prev_tag_open</a>	29
<a href="#">CI_Pagination::\$uri_segment</a>	29
<a href="#">CI_Pagination::\$use_page_numbers</a>	29
<a href="#">CI_Pagination::initialize()</a>	30
Initialize Preferences	
<a href="#">CI_Pagination::create_links()</a>	29
Generate the pagination links	
<a href="#">constructor CI_Pagination::__construct()</a>	29
Constructor	
<a href="#">CI_Pagination::\$prev_tag_close</a>	29
<a href="#">CI_Pagination::\$prev_link</a>	29
<a href="#">CI_Pagination::\$num_links</a>	28
<a href="#">CI_Pagination::\$next_tag_open</a>	28
<a href="#">CI_Pagination::\$next_tag_close</a>	28
<a href="#">CI_Pagination::\$next_link</a>	28
<a href="#">CI_Pagination::\$num_tag_close</a>	28
<a href="#">CI_Pagination::\$num_tag_open</a>	28
<a href="#">CI_Pagination::\$prefix</a>	28
<a href="#">CI_Pagination::\$per_page</a>	28
<a href="#">CI_Pagination::\$page_query_string</a>	28
<a href="#">CI_Form_validation::valid_email()</a>	24
Valid Email	
<a href="#">CI_Form_validation::valid_base64()</a>	23
Valid Base64	
<a href="#">CI_Form_validation::decimal()</a>	12
Decimal number	
<a href="#">CI_Form_validation::alpha_numeric()</a>	12
Alpha-numeric	
<a href="#">CI_Form_validation::alpha_dash()</a>	12
Alpha-numeric with underscores and dashes	
<a href="#">CI_Form_validation::alpha()</a>	11
Alpha	
<a href="#">CI_Form_validation::encode_php_tags()</a>	13

<i>Convert PHP tags to entities</i>	
<a href="#">CI_Form_validation::error()</a>	13
<i>Get Error Message</i>	
<a href="#">CI_Form_validation::integer()</a>	15
<i>Integer</i>	
<a href="#">CI_Form_validation::greater_than()</a>	14
<i>Greater than</i>	
<a href="#">CI_Form_validation::exact_length()</a>	14
<i>Exact Length</i>	
<a href="#">CI_Form_validation::error_string()</a>	13
<i>Error String</i>	
<a href="#">constructor CI_Form_validation:: construct()</a>	11
<i>Constructor</i>	
<a href="#">CI_Form_validation::\$ safe_form_data</a>	11
<a href="#">CI_Form_validation::\$ config_rules</a>	10
<a href="#">CI_Form_validation::\$error_string</a>	9
<a href="#">CI_Form_validation::\$CI</a>	9
<a href="#">CI_Form_validation</a>	9
<i>Form Validation Class</i>	
<a href="#">CI_Form_validation::\$ error_array</a>	10
<a href="#">CI_Form_validation::\$ error_messages</a>	10
<a href="#">CI_Form_validation::\$ field_data</a>	11
<a href="#">CI_Form_validation::\$ error_suffix</a>	10
<a href="#">CI_Form_validation::\$ error_prefix</a>	10
<a href="#">CI_Form_validation::is_natural()</a>	15
<i>Is a Natural number (0,1,2,3, etc.)</i>	
<a href="#">CI_Form_validation::is_natural_no_zero()</a>	15
<i>Is a Natural number, but not a zero (1,2,3, etc.)</i>	
<a href="#">CI_Form_validation::set_message()</a>	21
<i>Set Error Message</i>	
<a href="#">CI_Form_validation::set_error_delimiters()</a>	20
<i>Set The Error Delimiter</i>	
<a href="#">CI_Form_validation::set_checkbox()</a>	20
<i>Set Checkbox</i>	
<a href="#">CI_Form_validation::run()</a>	20
<i>Run the Validator</i>	
<a href="#">CI_Form_validation::set_radio()</a>	21
<i>Set Radio</i>	
<a href="#">CI_Form_validation::set_rules()</a>	22
<i>Set Rules</i>	
<a href="#">CI_Form_validation::strip_image_tags()</a>	23
<i>Strip Image Tags</i>	
<a href="#">CI_Form_validation::set_value()</a>	22
<i>Get the value from a form</i>	
<a href="#">CI_Form_validation::set_select()</a>	22
<i>Set Select</i>	
<a href="#">CI_Form_validation::required()</a>	19
<i>Required</i>	
<a href="#">CI_Form_validation::regex_match()</a>	19
<i>Performs a Regular Expression match test.</i>	
<a href="#">CI_Form_validation::matches()</a>	17
<i>Match one field to another</i>	
<a href="#">CI_Form_validation::less_than()</a>	16

<i>Less than</i>	
<a href="#">CI_Form::validation::is_unique()</a>	16
<i>Match one field to another</i>	
<a href="#">CI_Form::validation::is_numeric()</a>	16
<i>Is Numeric</i>	
<a href="#">CI_Form::validation::max_length()</a>	17
<i>Max Length</i>	
<a href="#">CI_Form::validation::min_length()</a>	17
<i>Minimum Length</i>	
<a href="#">CI_Form::validation::prep_url()</a>	18
<i>Prep URL</i>	
<a href="#">CI_Form::validation::prep_for_form()</a>	18
<i>Prep data for form</i>	
<a href="#">CI_Form::validation::numeric()</a>	18
<i>Numeric</i>	
<a href="#">CI_Template</a>	30
<i>CodeIgniter Template Class</i>	
<a href="#">CI_Template::\$CI</a>	31
<a href="#">CatalogoCsv::createTablePob()</a>	85
<i>Acción para ejecutar la creación de la tabla poblacional</i>	
<i>No recibe parámetros</i>	
<a href="#">CatalogoCsv::createTableHemoGlobina()</a>	84
<i>Acción para ejecutar la creación de la tabla Asu Ageb</i>	
<i>No recibe parámetros</i>	
<a href="#">CatalogoCsv::createTableGeo()</a>	84
<i>Acción para ejecutar la creación de la tabla poblacional</i>	
<i>No recibe parámetros</i>	
<a href="#">CatalogoCsv::createTableAgeb()</a>	84
<i>Acción para ejecutar la creación de la tabla Asu Ageb</i>	
<i>No recibe parámetros</i>	
<a href="#">CatalogoCsv::index()</a>	85
<i>Acción por default del controlador, carga la lista de catálogoscsv disponibles y una lista de opciones</i>	
<i>No recibe parámetros</i>	
<a href="#">CatalogoCsv::loadupdate()</a>	85
<i>Acción para cargar datos desde un archivo CSV, recibe el stream desde las variables PHP</i>	
<a href="#">Catalogo_x_raiz</a>	87
<i>Controlador Raiz_x_Catalogo</i>	
<a href="#">CatalogoCsv::array_unique_recursive()</a>	86
<i>_array_unique_recursive</i>	
<i>Revisa valores duplicados en arreglos que contienen arreglos</i>	
<a href="#">CatalogoCsv::view()</a>	86
<i>Acción para visualizar información de un catálogo específico, obtiene el objeto catalogocsv por medio del nombre proporcionado</i>	
<a href="#">CatalogoCsv::update()</a>	85
<i>Acción para preparar la actualización de un catálogo ya existente,</i>	
<a href="#">CatalogoCsv::checkpk()</a>	84
<i>Acción para revisar registros repetidos en las columnas designadas como primary key</i>	
<a href="#">CatalogoCsv::ActivaEnCatalogo()</a>	83
<i>*</i>	
<i>Acción para activar o desactivar elementos en los catálogos indicados en el parámetro Solo se permite su acceso por medio de peticiones AJAX</i>	
<a href="#">Catalogo::loadupdate()</a>	81

<i>Acción para cargar datos desde un archivo CSV, recibe el stream desde las variables PHP compara los datos recibidos con los datos que contiene actualmente el catálogo, regresa como resultado las filas nuevas y las filas a modificar.</i>	
<a href="#"><u>Catalogo::load()</u></a>	81
<i>Acción para cargar datos desde un archivo CSV, recibe el stream desde las variables PHP Guarda en la tabla tmp_catalogos toda la estructura del CSV e imprime las columnas del archivo. Solo se permite su acceso por medio de peticiones AJAX</i>	
<a href="#"><u>Catalogo::insert()</u></a>	80
<i>Acción para preparar la inserción de nuevos catálogos , realiza la validación del formulario del lado del servidor y crea la estructura para el catálogo, crea la tabla y obtiene los datos a partir de la tabla tmp_catalogo</i>	
<a href="#"><u>Catalogo::index()</u></a>	80
<i>Acción por default del controlador, carga la lista de catálogos disponibles y una lista de opciones No recibe parámetros</i>	
<a href="#"><u>Catalogo::update()</u></a>	81
<i>Acción para preparar la actualización de un catálogo ya existente,</i>	
<a href="#"><u>Catalogo::view()</u></a>	82
<i>Acción para visualizar información de un catálogo específico, obtiene el objeto catálogo por medio del nombre proporcionado</i>	
<a href="#"><u>constructor CatalogoCsv:: construct()</u></a>	83
<a href="#"><u>CatalogoCsv</u></a>	83
<i>Controlador CatalogoCsv</i>	
<a href="#"><u>Catalogo:: array_unique_recursive()</u></a>	82
<i>_array_unique_recursive Revisa valores duplicados en arreglos multidimensionales</i>	
<a href="#"><u>constructor Catalogo_x_raiz:: construct()</u></a>	87
<a href="#"><u>Catalogo_x_raiz::check()</u></a>	87
<i>Acción que sirve para revisar inconsistencias en el arbol de segmentacion Recibe como parámetro el catálogo x raiz y revisa que todos los registros tengan un correspondiente en el catálogo padre.</i>	
<a href="#"><u>constructor Controlador:: construct()</u></a>	93
<a href="#"><u>Controlador</u></a>	93
<i>Controlador Controlador</i>	
<a href="#"><u>Cie10:: array_unique_recursive()</u></a>	92
<i>_array_unique_recursive Revisa valores duplicados en arreglos que contienen arreglos</i>	
<a href="#"><u>Cie10::view()</u></a>	92
<i>*</i>	
<i>Accion para mostrar información de los catalogos IDE , ERA y Consultas</i>	
<a href="#"><u>Controlador::accion()</u></a>	93
<i>Acción para preparar la actualización de acciones asignadas a un controlador, recibe un ID para obtener las acciones asignadas a ese controlador y mostrarlos en la vista update</i>	
<a href="#"><u>Controlador::delete()</u></a>	93
<i>Acción para eliminar un controlador, recibe el id del controlador a eliminar</i>	
<a href="#"><u>Controlador::index()</u></a>	94
<i>Acción por default del controlador, carga la lista de controladores disponibles y una lista de opciones Recibe un parametro en caso de filtrado por entornos</i>	
<a href="#"><u>Controlador::help()</u></a>	94
<i>Establece el texto de ayuda para el controlador accion</i>	

<a href="#"><u>Controlador::getGroupPermissions()</u></a>	94
Acción para servir un array de objetos con los permisos asignados a	
<a href="#"><u>Cie10::update()</u></a>	91
Acción para preparar la actualización de un registro del CIE10,	
<a href="#"><u>Cie10::load()</u></a>	91
Acción para cargar datos desde un archivo CSV, recibe el stream desde las variables PHP	
<a href="#"><u>Cie10</u></a>	89
Controlador Cie10	
<a href="#"><u>Catalogo_x_raiz::view()</u></a>	88
Acción para visualizar información de una raiz_x_catalogo específica, obtiene el objeto raiz_x_catalogo por medio del id proporcionado.	
<a href="#"><u>Catalogo_x_raiz::insert()</u></a>	88
Acción para preparar la inserción de nuevas raíces para catálogos , realiza la validación del formulario del lado cliente	
<a href="#"><u>Catalogo_x_raiz::delete()</u></a>	88
Acción para eliminar un catálogo en el arbol, recibe el id del catálogo en la raíz a eliminar	
<a href="#"><u>constructor Cie10:: construct()</u></a>	89
<a href="#"><u>Cie10::ActivaEnCatalogo()</u></a>	89
*	
Accion para activar o desactivar elementos en los catalogos IRA EDA Consultas	
Solo se permite su acceso por medio de peticiones AJAX	
<a href="#"><u>Cie10::insert()</u></a>	90
Acción para cargar datos desde un archivo CSV, recibe el stream desde las variables PHP	
compara los datos recibidos con los datos que contiene actualmente el catálogo, regresa como	
resultado las filas nuevas y las filas a modificar	
<a href="#"><u>Cie10::index()</u></a>	90
Acción por default del controlador, carga la lista	
de datos disponibles en el cie10 y una lista de opciones	
<a href="#"><u>Cie10::AgregaEnCatalogo()</u></a>	90
*	
Accion para agregar elementos del catalogo cie10 a los catalogos de EDA, IRA y	
Consultas dependiendo de los	
Solo se permite su acceso por medio de peticiones AJAX	
<a href="#"><u>Catalogo::delete()</u></a>	80
Acción para eliminar un catálogo, recibe el nombre del catalogo a eliminar	
<a href="#"><u>Catalogo::checkTypeData()</u></a>	79
Acción para revisar si los tipos de datos coinciden con los datos contenidos en	
la tabla temporal que fueron tomados del CSV	
<a href="#"><u>CI_Template::add_template()</u></a>	33
Dynamically add a template and optionally switch to it	
<a href="#"><u>CI_Template::add_region()</u></a>	32
Dynamically add region to the currently set template	
<a href="#"><u>CI_Template::add_js()</u></a>	32
Dynamically include javascript in the template	
<a href="#"><u>CI_Template::add_css()</u></a>	32
Dynamically include CSS in the template	
<a href="#"><u>CI_Template::empty_region()</u></a>	33
Empty a region's content	
<a href="#"><u>CI_Template::initialize()</u></a>	33
Initialize class settings using config settings	
<a href="#"><u>CI_Template::set_master_template()</u></a>	35
Set master template	

<a href="#">CI_Template::render()</a>	<i>Render the master template or a single region</i>	35
<a href="#">CI_Template::parse_view()</a>	<i>Parse content from a View to a region with the Parser Class</i>	34
<a href="#">CI_Template::load()</a>	<i>Load the master template or a single region</i>	34
<a href="#">constructor CI_Template::CI_Template()</a>	<i>Constructor</i>	31
<a href="#">CI_Template::\$template</a>		31
<a href="#">CI_Template::\$master</a>		31
<a href="#">CI_Template::\$js</a>		31
<a href="#">CI_Template::\$css</a>		31
<a href="#">CI_Template::\$config</a>		31
<a href="#">CI_Template::\$output</a>		31
<a href="#">CI_Template::\$parser</a>		31
<a href="#">CI_Template::\$regions</a>		31
<a href="#">CI_Template::\$parse_template</a>		31
<a href="#">CI_Template::\$parser_method</a>		31
<a href="#">CI_Template::set_parser()</a>	<i>Set parser</i>	35
<a href="#">CI_Template::set_parser_method()</a>	<i>Set parser method</i>	36
<a href="#">constructor Accion:: construct()</a>		74
<a href="#">controlador.php</a>		65
<a href="#">cie10.php</a>		64
<a href="#">catalogo_x_raiz.php</a>		63
<a href="#">constructor Ayuda:: construct()</a>		76
<a href="#">constructor Bitacora:: construct()</a>		77
<a href="#">Catalogo::checkpk()</a>	<i>Acción para revisar registros repetidos en las columnas designadas como primary key</i>	79
<a href="#">constructor Catalogo:: construct()</a>		79
<a href="#">Catalogo</a>	<i>Controlador Catalogo</i>	79
<a href="#">catalogocsv.php</a>		62
<a href="#">catalogo.php</a>		61
<a href="#">CI_Template::write_view()</a>	<i>Write content from a View to a region. 'Views within views'</i>	37
<a href="#">CI_Template::write()</a>	<i>Write contents to a region</i>	37
<a href="#">CI_Template::set_template()</a>	<i>Use given template settings</i>	36
<a href="#">CI_Template::set_regions()</a>	<i>Set regions for writing to</i>	36
<a href="#">constructor Menubuilder:: construct()</a>		40
<a href="#">constructor Graph:: construct()</a>		44
<a href="#">constructor Session:: construct()</a>		53
<a href="#">Constructor</a>		
<a href="#">constructor Tree:: construct()</a>		48
<a href="#">constructor Obtenercurp:: construct()</a>		46
<a href="#">constructor Index:: construct()</a>		4

## E

<a href="#"><u>Enrolamiento_model::setaccion_nutricional()</u></a>	317
<a href="#"><u>Enrolamiento_model::insert()</u></a>	317
<i>Guarda la persona capturada mediante el formulario web</i>	
<a href="#"><u>Enrolamiento_model::get_version()</u></a>	316
<i>obtiene cual es la ultima version de apk de la tableta</i>	
<a href="#"><u>Enrolamiento_model::setafiliacion()</u></a>	317
<a href="#"><u>Enrolamiento_model::setageb()</u></a>	317
<a href="#"><u>Enrolamiento_model::setaltura()</u></a>	318
<a href="#"><u>Enrolamiento_model::setalergias()</u></a>	318
<a href="#"><u>Enrolamiento_model::get_transaction_relevante()</u></a>	316
<i>Obtiene las transacciones relevante para la sincronizacion</i>	
<a href="#"><u>Enrolamiento_model::get_sales()</u></a>	316
<i>Obtiene los registros de sales de rehidratacion oral asociados a una persona</i>	
<a href="#"><u>Enrolamiento_model::get_estimulacion()</u></a>	314
<i>Obtiene los registros de estimulacion temprana asociados a una persona</i>	
<a href="#"><u>Enrolamiento_model::get_datos_grafica()</u></a>	314
<i>Obtiene los datos especificos de un catalogo para ser visualizados en una grafica</i>	
<a href="#"><u>Enrolamiento_model::get_control_nutricional()</u></a>	313
<i>Obtiene los datos del control nutricional asociados a una persona</i>	
<a href="#"><u>Enrolamiento_model::get_notificacion()</u></a>	314
<i>Este metodo obtiene las notificaciones que se enviaran en la sincronizacion</i>	
<a href="#"><u>Enrolamiento_model::get_pacientes()</u></a>	315
<i>devuelve todos los pacientes de la base de datos</i>	
<a href="#"><u>Enrolamiento_model::get_persona_x_tutor()</u></a>	315
<i>obtiene los tutores de las personas que se envian en la sincronizacion</i>	
<a href="#"><u>Enrolamiento_model::get_peri_cefa()</u></a>	315
<i>Obtiene los registros de perimetro cefalico asociados a una persona</i>	
<a href="#"><u>Enrolamiento_model::setcalle()</u></a>	318
<a href="#"><u>Enrolamiento_model::setcelular()</u></a>	318
<a href="#"><u>Enrolamiento_model::setestimulacion_fecha()</u></a>	322
<a href="#"><u>Enrolamiento_model::setestimulacion_capacitado()</u></a>	321
<a href="#"><u>Enrolamiento_model::setcurpT()</u></a>	321
<a href="#"><u>Enrolamiento_model::setfaccion_nutricional()</u></a>	322
<a href="#"><u>Enrolamiento_model::setfconsulta()</u></a>	322
<a href="#"><u>Enrolamiento_model::setfecha_peri_cefa()</u></a>	323
<a href="#"><u>Enrolamiento_model::setfechacivil()</u></a>	322
<a href="#"><u>Enrolamiento_model::setcurp()</u></a>	321
<a href="#"><u>Enrolamiento_model::setcp()</u></a>	320
<a href="#"><u>Enrolamiento_model::setcodigo_barras()</u></a>	319
<a href="#"><u>Enrolamiento_model::setcelularT()</u></a>	319
<a href="#"><u>Enrolamiento_model::setcolonia()</u></a>	319
<a href="#"><u>Enrolamiento_model::setcompania()</u></a>	320
<a href="#"><u>Enrolamiento_model::setconsulta()</u></a>	320
<a href="#"><u>Enrolamiento_model::setcompaniaT()</u></a>	320
<a href="#"><u>Enrolamiento_model::get_cns_persona()</u></a>	313
<i>Este metodo obtiene las personas que seran enviadas en la sincronizacion</i>	
<a href="#"><u>Enrolamiento_model::get_cns_cat_persona_count()</u></a>	313
<i>Hace el count de personas que se envian en la sincronizacion</i>	
<a href="#"><u>Enrolamiento_model::getRegistro_civil()</u></a>	307
<i>obtiene informacion del registro civil</i>	
<a href="#"><u>Enrolamiento_model::getreferencia()</u></a>	307

<a href="#">Enrolamiento_model::getprecurp()</a>	307
<a href="#">Enrolamiento_model::getsales_cantidad()</a>	307
<a href="#">Enrolamiento_model::getsales_fecha()</a>	308
<a href="#">Enrolamiento_model::getsector()</a>	308
<a href="#">Enrolamiento_model::getsangre()</a>	308
<a href="#">Enrolamiento_model::getpeso()</a>	307
<a href="#">Enrolamiento_model::getperi_cefa()</a>	307
<a href="#">Enrolamiento_model::getnumero()</a>	306
<a href="#">Enrolamiento_model::getnombreT()</a>	305
<a href="#">Enrolamiento_model::getnombre()</a>	305
<a href="#">Enrolamiento_model::getNumRows()</a>	306
<i>Devuelve el numero de filas en la tabla cns_persona</i>	
<a href="#">Enrolamiento_model::getparto()</a>	306
<a href="#">Enrolamiento_model::getpaternoT()</a>	306
<a href="#">Enrolamiento_model::getpaterno()</a>	306
<a href="#">Enrolamiento_model::getsexo()</a>	308
<a href="#">Enrolamiento_model::getsexoT()</a>	308
<a href="#">Enrolamiento_model::get_catalog_count()</a>	311
<i>Obtiene el numero de resultados de una tabla</i>	
<a href="#">Enrolamiento_model::get_catalog2()</a>	310
<i>Obtiene los datos de una tabla</i>	
<a href="#">Enrolamiento_model::get_catalog()</a>	310
<i>Hace select de las tablas cns_x que representa a los catalogos</i>	
<a href="#">Enrolamiento_model::get_catalog_relevante()</a>	311
<i>obtiene los catalogos relevantes x entorno para la sincronizacion</i>	
<a href="#">Enrolamiento_model::get_catalog_tratamiento()</a>	311
<i>Hace select de los tratamientos de las consultas</i>	
<a href="#">Enrolamiento_model::get_cns_cat_persona()</a>	312
<i>Este metodo obtiene los controles que le corresponde a cada persona y que seran incluidas en la sincronizacion</i>	
<a href="#">Enrolamiento_model::get_catalog_view()</a>	312
<i>Hace select de los catalogos que tengan relacion con una persona para mostrarlos en el view</i>	
<a href="#">Enrolamiento_model::getvacuna()</a>	309
<a href="#">Enrolamiento_model::getumt()</a>	309
<a href="#">Enrolamiento_model::gettamiz()</a>	309
<a href="#">Enrolamiento_model::gettalla()</a>	308
<a href="#">Enrolamiento_model::getbeneficiario()</a>	309
<a href="#">Enrolamiento_model::gettconsulta()</a>	309
<a href="#">Enrolamiento_model::gettelefonoT()</a>	309
<a href="#">Enrolamiento_model::gettelefono()</a>	309
<a href="#">Enrolamiento_model::setfnacimiento()</a>	323
<a href="#">Enrolamiento_model::setfnutricion()</a>	323
<a href="#">Enrolamiento_model::update_regcivil()</a>	335
<i>actualiza el registro civil del paciente</i>	
<a href="#">Enrolamiento_model::update_peri_cefa()</a>	335
<i>Actualiza los registros de perimetro cefalico</i>	
<a href="#">Enrolamiento_model::update_nutricion()</a>	335
<i>Actualiza el control nutricional del paciente</i>	
<a href="#">Enrolamiento_model::update_sales()</a>	336
<i>Actualiza los registros de sales de rehidratacion oral</i>	
<a href="#">Enrolamiento_model::update_status_tableta()</a>	336
<i>Hace update de la tableta que este sincronizando dependiendo del resultado</i>	

<a href="#"><u>Enrolamiento_model::update_umt()</u></a>	Actualiza la unidad medica tratante del paciente	. . . . . 336
<a href="#"><u>Enrolamiento_model::update_tutor()</u></a>	Actualiza los datos del tutor del paciente	. . . . . 336
<a href="#"><u>Enrolamiento_model::update_estimulacion()</u></a>	Actualiza los registros de estimulacion temprana	. . . . . 335
<a href="#"><u>Enrolamiento_model::update_direccion()</u></a>	actualiza la direccion del paciente	. . . . . 335
<a href="#"><u>Enrolamiento_model::update_accion()</u></a>	Actualiza el control accion nutricional del paciente	. . . . . 334
<a href="#"><u>Enrolamiento_model::tes_pendientes_tarjeta_delete()</u></a>	Elimina los pendientes de las personas que no tengan asignado una tarjeta	. . . . . 333
<a href="#"><u>Enrolamiento_model::setvacuna()</u></a>	. . . . . 333	
<a href="#"><u>Enrolamiento_model::update_alergia()</u></a>	Actualiza los datos de las alergias del paciente	. . . . . 334
<a href="#"><u>Enrolamiento_model::update_basico()</u></a>	Actualiza los datos basicos del paciente	. . . . . 334
<a href="#"><u>Enrolamiento_model::update_consulta()</u></a>	Actualiza el control consulta del paciente	. . . . . 334
<a href="#"><u>Enrolamiento_model::updateBeneficiario()</u></a>	Actualiza el tipo de beneficiario del paciente	. . . . . 334
<a href="#"><u>Enrolamiento_model::update_vacuna()</u></a>	Actualiza las vacunas del paciente	. . . . . 337
<a href="#"><u>Enrolamiento_model::valid_card()</u></a>	Este metodo valida que exista un archivo para enviar a la tarjeta por nfc	. . . . . 337
<a href="#"><u>entorno_model.php</u></a>	Código fuente	. . . . . 604
<a href="#"><u>errorlog.php</u></a>	Código fuente	. . . . . 495
<a href="#"><u>entorno.php</u></a>	Código fuente	. . . . . 490
<a href="#"><u>errorlog_model.php</u></a>	Código fuente	. . . . . 609
<a href="#"><u>enrolamiento.php</u></a>	Código fuente	. . . . . 657
<a href="#"><u>estado_tableta_model.php</u></a>	Código fuente	. . . . . 766
<a href="#"><u>enrolamiento_model.php</u></a>	Código fuente	. . . . . 726
<a href="#"><u>Estado_tableta_model::setId()</u></a>	. . . . . 339	
<a href="#"><u>Estado_tableta_model::setDescripcion()</u></a>	. . . . . 339	
<a href="#"><u>Estado_tableta_model::getAll()</u></a>	Obtiene todos los registros de la tabla	. . . . . 338
<a href="#"><u>Estado_tableta_model</u></a>	Modelo Estado_tableta	. . . . . 337
<a href="#"><u>Estado_tableta_model::getById()</u></a>	Obtiene los datos del registro que tiene el ID especificado	. . . . . 338
<a href="#"><u>Estado_tableta_model::getDescripcion()</u></a>	. . . . . 338	
<a href="#"><u>Estado_tableta_model::getMsgError()</u></a>	Devuelve el mensaje de error, en caso de existir un error despues de ejecutar un metodo, de lo contrario false	. . . . . 339
<a href="#"><u>Estado_tableta_model::getId()</u></a>	. . . . . 339	

<a href="#">Enrolamiento_model::setumt()</a>	333
<a href="#">Enrolamiento_model::settelefonoT()</a>	332
<a href="#">Enrolamiento_model::setnacionalidad()</a>	326
<a href="#">Enrolamiento_model::setmaternoT()</a>	326
<a href="#">Enrolamiento_model::setmaterno()</a>	326
<a href="#">Enrolamiento_model::setnombre()</a>	327
<a href="#">Enrolamiento_model::setnombreT()</a>	327
<a href="#">Enrolamiento_model::setparto()</a>	327
<a href="#">Enrolamiento_model::setnumero()</a>	327
<a href="#">Enrolamiento_model::setmanzana()</a>	325
<a href="#">Enrolamiento_model::setlugarcivil()</a>	325
<a href="#">Enrolamiento_model::sethemoglobina()</a>	324
<a href="#">Enrolamiento_model::setfvacuna()</a>	323
<a href="#">Enrolamiento_model::setId()</a>	324
<a href="#">Enrolamiento_model::setidtutor()</a>	324
<a href="#">Enrolamiento_model::setlocalidad()</a>	325
<a href="#">Enrolamiento_model::setlnacimiento()</a>	325
<a href="#">Enrolamiento_model::setpaterno()</a>	328
<a href="#">Enrolamiento_model::setpaternoT()</a>	328
<a href="#">Enrolamiento_model::settalla()</a>	331
<a href="#">Enrolamiento_model::setsexoT()</a>	331
<a href="#">Enrolamiento_model::setsexo()</a>	330
<a href="#">Enrolamiento_model::settamiz()</a>	331
<a href="#">Enrolamiento_model::settbeneficiario()</a>	332
<a href="#">Enrolamiento_model::settelefono()</a>	332
<a href="#">Enrolamiento_model::setconsulta()</a>	332
<a href="#">Enrolamiento_model::setsector()</a>	330
<a href="#">Enrolamiento_model::setsangre()</a>	330
<a href="#">Enrolamiento_model::setpeso()</a>	329
<a href="#">Enrolamiento_model::setperi_cefa()</a>	328
<a href="#">Enrolamiento_model::setprecup()</a>	329
<a href="#">Enrolamiento_model::setreferencia()</a>	329
<a href="#">Enrolamiento_model::setsales_fecha()</a>	330
<a href="#">Enrolamiento_model::setsales_cantidad()</a>	329
<a href="#">Enrolamiento_model::getnacionalidad()</a>	305
<a href="#">Enrolamiento_model::getMsgError()</a>	305
<a href="#"><b>Errorlog_model::insert()</b></a>	201
<i>Inserta en la base de datos la informacion del error</i>	
<a href="#"><b>Errorlog_model::getNumRows()</b></a>	201
<i>Obtiene el numero total de registros en la tabla Error en caso de existir filtros, estos son aplicados a la consulta</i>	
<a href="#">Errorlog_model::getId_usuario()</a>	201
<a href="#">Errorlog_model::resetFilter()</a>	202
<i>Elimina todos los filtros registrados</i>	
<a href="#">Errorlog_model::save()</a>	202
<i>Guardar el mensaje de error descriptivo en la base de datos,</i>	
<a href="#">Errorlog_model::setId()</a>	203
<a href="#">Errorlog_model::setDescripcion()</a>	202
<a href="#">Errorlog_model::getId_controlador_accion()</a>	201
<a href="#">Errorlog_model::getId()</a>	201
<a href="#">Errorlog_model::addFilter()</a>	199
<i>Agrega una nueva regla de filtrado al arreglo de filtros</i>	
<a href="#"><b>Errorlog_model</b></a>	199

<i>Modelo Errorlog</i>	
<a href="#">Entorno_model::update()</a>	198 <i>Actualiza el objeto actual en la base de datos</i>
<a href="#">Errorlog_model::getAll()</a>	199 <i>Obtiene todos los registros de la tabla Error en caso de existir filtros, estos son aplicados a la consulta</i>
<a href="#">Errorlog_model::getById()</a>	200 <i>Obtiene los datos del registro del Error que tiene el ID especificado</i>
<a href="#">Errorlog_model::getFecha_hora()</a>	200
<a href="#">Errorlog_model::getDescripcion()</a>	200
<a href="#">Errorlog_model::setId_accion()</a>	203
<a href="#">Errorlog_model::setId_controlador()</a>	203
<a href="#">Enrolamiento::categoriacie10_select()</a>	258 <i>Genera los options de un campo tipo select para las categorías de CIE10</i>
<a href="#">Enrolamiento::catalog_select()</a>	258 <i>Genera los options de un campo tipo select</i>
<a href="#">Enrolamiento::catalog_check()</a>	257 <i>Crea un grupo de radio o check con la informacion de los catalogos</i>
<a href="#">Enrolamiento::checkar_session()</a>	258 <i>Revisa si la sesión está activa</i>
<a href="#">Enrolamiento::cie10_select()</a>	259 <i>Genera los options de un campo tipo select para los CIE10 correspondientes a una categoría</i>
<a href="#">Enrolamiento::data_tutor()</a>	259 <i>Obtiene informacion del tutor</i>
<a href="#">Enrolamiento::comparar_view()</a>	259 <i>Crea la pagina para ver la infromacion de la persona y compararla con la persona capturada</i>
<a href="#">Enrolamiento::brother_found()</a>	257 <i>Este metodo verifica si un paciente comparte un mismo tutor</i>
<a href="#">Enrolamiento::brothers_search()</a>	257 <i>Este metodo extrae la informacion de las personas con las que se comparte el mismo tutor si se selecciona una de estas importa los datos para el apartado direccion</i>
<a href="#">Errorlog_model::setId_usuario()</a>	204
<a href="#">Errorlog_model::setId_controlador_accion()</a>	203
<a href="#">enrolamiento.php</a>	248
<a href="#">Enrolamiento</a>	256 <i>Controlador de enrolamiento</i>
<a href="#">Enrolamiento::autocomplete()</a>	256 <i>Crea el autocomplete para facilitar la busqueda de un tutor</i>
<a href="#">Enrolamiento::addForm()</a>	256 <i>Pase de parametros para la insercion o actualizacion</i>
<a href="#">Entorno_model::setNombre()</a>	198
<a href="#">Entorno_model::setIp()</a>	198
<a href="#">Errorlog::view()</a>	100 <i>Muestra los datos del registro especificado por el id</i>
<a href="#">Errorlog::index()</a>	99 <i>Lista todos los registros de la tabla error, con su correspondiente paginación permite hacer filtrados por Usuario, Entorno, Controlador, Acción y Fecha, muestra enlaces para ver detalles de un elemento específico</i>
<a href="#">Errorlog</a>	99 <i>Controlador Errorlog</i>
<a href="#">entorno_model.php</a>	129

<a href="#">errorlog_model.php</a>	130
<a href="#">Entorno_model::delete()</a>	193
<i>Elimina el registro actual de la base de datos</i>	
<a href="#">Entorno_model</a>	193
<i>Modelo Entorno</i>	
<a href="#">Entorno:: ExistEntornoUpdate()</a>	98
<i>Acción para validar que no exista previamente el entorno a actualizar</i>	
<a href="#">Entorno:: ExistEntorno()</a>	98
<i>Acción para validar que no exista previamente el entorno a insertar</i>	
<i>(Esta acción no puede ser accedida desde el navegador)</i>	
<a href="#">Entorno::delete()</a>	96
<i>Acción para eliminar un entorno, recibe el id del entorno a eliminar</i>	
<a href="#">Entorno</a>	96
<i>Controlador Entorno</i>	
<a href="#">errorlog.php</a>	67
<a href="#">Entorno::index()</a>	97
<i>Acción por default del controlador, carga la lista</i>	
<i>de entornos disponibles y una lista de opciones</i>	
<i>No recibe parámetros</i>	
<a href="#">Entorno::insert()</a>	97
<i>Acción para preparar la inserción de nuevos entornos , realiza la validación</i>	
<i>del formulario del lado cliente y del lado servidor para evitar entornos duplicados</i>	
<a href="#">Entorno::view()</a>	98
<i>Acción para visualizar de un entorno específico, obtiene el objeto</i>	
<i>entorno por medio del id proporcionado.</i>	
<a href="#">Entorno::update()</a>	97
<i>Acción para preparar la actualización de un entorno ya existente,</i>	
<a href="#">Entorno_model::getAII()</a>	193
<i>Devuelve todos los registros de la tabla entorno</i>	
<a href="#">Entorno_model::getById()</a>	194
<i>Devuelve la información de un entorno por su ID</i>	
<a href="#">Entorno_model::insert()</a>	196
<i>Inserta en la tabla entorno la información contenida en el objeto</i>	
<a href="#">Entorno_model::getPermissionsByGroup()</a>	196
<i>Obtiene los permisos asignados al grupo</i>	
<a href="#">Entorno_model::getNombre()</a>	196
<a href="#">Entorno_model::setDescripcion()</a>	197
<a href="#">Entorno_model::setDirectorio()</a>	197
<a href="#">Entorno_model::setId()</a>	197
<a href="#">Entorno_model::setHostname()</a>	197
<a href="#">Entorno_model::getMsgError()</a>	195
<i>Devuelve los mensajes de error en caso de ocurrir alguna excepción</i>	
<i>'usr' devuelve el mensaje para la vista de usuario</i>	
<i>'log' devuelve el mensaje para el log de errores</i>	
<a href="#">Entorno_model::getIp()</a>	195
<a href="#">Entorno_model::getDescripcion()</a>	194
<a href="#">Entorno_model::getByName()</a>	194
<i>Devuelve la información de un entorno por su nombre</i>	
<a href="#">Entorno_model::getDirectorio()</a>	195
<a href="#">Entorno_model::getHostname()</a>	195
<a href="#">Entorno_model::getInfo()</a>	195
<i>Devuelve la información del objeto en forma de string</i>	
<a href="#">Entorno_model::getId()</a>	195

\*\*\*\*\*

<a href="#">Enrolamiento::file_to_card()</a>	260
<i>crea un archivo descargable el cual se necesita para el envio por nfc a la tarjeta del paciente</i>	
<a href="#">Enrolamiento::ifCurpExists()</a>	260
<i>Valida que la curp del paciente no exista</i>	
<a href="#">Enrolamiento_model::getcurp()</a>	301
<a href="#">Enrolamiento_model::getcp()</a>	301
<a href="#">Enrolamiento_model::getControlConsultas()</a>	301
<i>Obtiene todas las consultas asociadas a un paciente</i>	
<a href="#">Enrolamiento_model::getcurpT()</a>	301
<a href="#">Enrolamiento_model::getestimulacion_capacitado()</a>	301
<a href="#">Enrolamiento_model::getfaccion_nutricional()</a>	302
<a href="#">Enrolamiento_model::getestimulacion_fecha()</a>	301
<a href="#">Enrolamiento_model::getconsulta()</a>	300
<a href="#">Enrolamiento_model::getcompaniaT()</a>	300
<a href="#">Enrolamiento_model::getcelularT()</a>	299
<a href="#">Enrolamiento_model::getcelular()</a>	299
<a href="#">Enrolamiento_model::getCategoriaCIE10()</a>	299
<i>Obtiene el listado de categorias de CIE10</i>	
<a href="#">Enrolamiento_model::getCIE10()</a>	299
<i>Obtiene el listado de CIE10 correspondientes a una CIE10</i>	
<a href="#">Enrolamiento_model::getcodigo_barras()</a>	300
<a href="#">Enrolamiento_model::getcompania()</a>	300
<a href="#">Enrolamiento_model::getcolonia()</a>	300
<a href="#">Enrolamiento_model::getconsulta()</a>	302
<a href="#">Enrolamiento_model::getfechacivil()</a>	302
<a href="#">Enrolamiento_model::getlocalidad()</a>	304
<a href="#">Enrolamiento_model::getlnacimiento()</a>	304
<a href="#">Enrolamiento_model::getListEnrolamiento()</a>	304
<i>Este metodo retorna el list de las personas enroladas</i>	
<a href="#">Enrolamiento_model::getlugarcivil()</a>	304
<a href="#">Enrolamiento_model::getmanzana()</a>	304
<a href="#">Enrolamiento_model::getmaternoT()</a>	305
<a href="#">Enrolamiento_model::getmaterno()</a>	305
<a href="#">Enrolamiento_model::getidtutor()</a>	303
<a href="#">Enrolamiento_model::getId()</a>	303
<a href="#">Enrolamiento_model::getfnacimiento()</a>	302
<a href="#">Enrolamiento_model::getfecha_peri_cefa()</a>	302
<a href="#">Enrolamiento_model::getfnutricion()</a>	302
<a href="#">Enrolamiento_model::getfolio()</a>	303
<i>Extrae el folio que se anexa en el envio para la tarjeta</i>	
<a href="#">Enrolamiento_model::gethemoglobina()</a>	303
<a href="#">Enrolamiento_model::getfvacuna()</a>	303
<a href="#">Enrolamiento_model::getcalle()</a>	299
<a href="#">Enrolamiento_model::getById()</a>	298
<i>Obtiene la informacion de la persona</i>	
<a href="#">Enrolamiento::vacunacion()</a>	264
<i>Obtiene el historial de vacunacion de un paciente cuyo id es pasado por parametro</i>	
<a href="#">Enrolamiento::update_card()</a>	264
<i>Este metodo actualiza el estado del archivo descargado si fue escrito correctamente o no en la tarjeta</i>	
<a href="#">Enrolamiento::update()</a>	264

<i>Crea el formulario para editar la informacion de la persona</i>	
<a href="#">Enrolamiento::validarForm()</a>	265
<i>valida los datos de entrada en el formulario</i>	
<a href="#">Enrolamiento::validarism()</a>	265
<i>valida que el nodo seleccionado en el arbol sea una unidad medica</i>	
<a href="#">Enrolamiento::view()</a>	266
<i>Crea la pagina para ver la infromacion de la persona</i>	
<a href="#">Enrolamiento::validate_card()</a>	266
<i>valida que un archivo sea valido para enviar a la tarjeta por nfc</i>	
<a href="#">Enrolamiento::tratamiento_select()</a>	263
<i>Genera los options de un campo tipo select para los tratamientos de consultas</i>	
<a href="#">Enrolamiento::searchum()</a>	263
<i>Busca dentro del catalogo asu_ageb la unidad médica de acuerdo a la localidad y ageb</i>	
<i>Solo se permite su acceso por medio de peticiones AJAX</i>	
<a href="#">Enrolamiento::index()</a>	261
<i>Este es el metodo por default, obtiene el listado de las personas</i>	
<a href="#">Enrolamiento::ifCurpTExists()</a>	261
<i>valida que la curp del tutor no exista</i>	
<a href="#">Enrolamiento::insert()</a>	261
<i>prepara los datos para insertarlos</i>	
<a href="#">Enrolamiento::paciente_similar()</a>	262
<i>Comprueba la similitud de un paciente que se este capturando con los que ya existe en la base de datos,</i>	
<i>esto con la finalidad de disminuir datos repetidos</i>	
<a href="#">Enrolamiento::searchageb()</a>	262
<i>Regresa un objeto JSON con la lista de agebs disponibles en la localidad</i>	
<i>Solo se permite su acceso por medio de peticiones AJAX</i>	
<a href="#">Enrolamiento::print_card()</a>	262
<i>Este metodo extrae la informacion del paciente que sera impreso en la tarjeta</i>	
<a href="#">enrolamiento_model.php</a>	285
<a href="#">estado_tableta_model.php</a>	286
<a href="#">Enrolamiento_model::getageb()</a>	297
<a href="#">Enrolamiento_model::getAfiliaciones()</a>	297
<i>Obtiene las afiliaciones asociadas a una persona</i>	
<a href="#">Enrolamiento_model::getafiliacion()</a>	297
<a href="#">Enrolamiento_model::getAlergia()</a>	297
<i>Obtiene las alergias asociadas a una persona</i>	
<a href="#">Enrolamiento_model::getalergias()</a>	298
<a href="#">Enrolamiento_model:: getByCurp()</a>	298
<i>valida que no se repita la curp en personas y tutor</i>	
<a href="#">Enrolamiento_model::getaltura()</a>	298
<a href="#">Enrolamiento_model::getaccion_nutricional()</a>	297
<a href="#">Enrolamiento_model::entorno_x_persona()</a>	296
<i>Este metodo actualiza o inserta los datos que permiten el envio de la informacion a la tarjeta por nfc</i>	
<a href="#">Enrolamiento_model::autocomplete_tutor()</a>	294
<i>obtiene informacion del tutor para genberar el autocomplete</i>	
<a href="#">Enrolamiento_model</a>	294
<i>Modelo Usuario</i>	
<a href="#">Enrolamiento_model::cns_insert()</a>	295
<i>Hace insert de las tablas cns_control_x que se reciben en la sincronizacion secuencial</i>	
<a href="#">Enrolamiento_model::cns_update()</a>	295
<i>Actualiza la tabla especificada</i>	

<a href="#">Enrolamiento_model::data_tutor()</a>	296
<i>obtiene informacion del tutor</i>	
<a href="#">Enrolamiento_model::cns_update_visita()</a>	295
<a href="#">entorno.php</a>	66

## F

<a href="#">Form_validation.php</a>	390
<i>Código fuente</i>	
<a href="#">Form_validation.php</a>	6
<i>CodeIgniter</i>	

## G

<a href="#">Grupo_model::getMsgError()</a>	208
<i>Asigna el mensaje de error a visualizar: para usuario final (usr) o para bitácora (log)</i>	
<a href="#">Grupo_model::getNombre()</a>	208
<a href="#">Grupo_model::getNumRows()</a>	208
<i>Obtiene el numero total de grupos</i>	
<a href="#">Grupo_model::getId()</a>	208
<a href="#">Grupo_model::getEntornosById()</a>	207
<i>Obtiene el grupo solicitado con sus entornos vinculados</i>	
<a href="#">Grupo_model::getByld()</a>	206
<i>Obtiene el grupo solicitado</i>	
<a href="#">Grupo_model::getByName()</a>	207
<i>Obtiene el grupo solicitado</i>	
<a href="#">Grupo_model::getDescripcion()</a>	207
<a href="#">Grupo_model::insert()</a>	209
<i>Inserta en la base de datos los datos del grupo (datos en propiedades)</i>	
<a href="#">Grupo_model::setDescripcion()</a>	209
<a href="#">grupo.php</a>	498
<i>Código fuente</i>	
<a href="#">georeferencia_model.php</a>	615
<i>Código fuente</i>	
<a href="#">grupo_model.php</a>	617
<i>Código fuente</i>	
<a href="#">graph.php</a>	424
<i>Código fuente</i>	
<a href="#">Grupo_model::update()</a>	210
<i>Actualiza en la base de datos los datos del grupo (datos en propiedades)</i>	
<a href="#">Grupo_model::setId()</a>	209
<a href="#">Grupo_model::setNombre()</a>	209
<a href="#">Grupo_model::getAll()</a>	206
<i>Obtiene todos los grupos existentes</i>	
<a href="#">Grupo_model::delete()</a>	206
<i>Elimina de la base de datos al grupo (id en propiedades)</i>	
<a href="#">Grupo::delete()</a>	101
<i>Solicita la eliminación del grupo recibido</i>	
<a href="#">Grupo::index()</a>	101
1) <i>Visualiza los grupos existentes para su interacción CRUD</i>	
2) <i>En caso de detectar un texto a buscar se filtran los grupos existentes acorde a la</i>	

<i>búsqueda</i>	
<a href="#"><u>Grupo::insert()</u></a>	101
1) Prepara el formulario para la inserción de un grupo nuevo	
2) Realiza las validaciones necesarias sobre cada campo del registro	
<a href="#"><u>Grupo</u></a>	100
Controlador Grupo	
<a href="#"><u>grupo.php</u></a>	68
<a href="#"><u>Graph</u></a>	44
Controlador Objetos	
<a href="#"><u>Graph::graph_init()</u></a>	44
Crea una grafica en el lugar que se llame	
<a href="#"><u>Graph::map()</u></a>	45
crea un objeto mapa con la ayuda de la api de google	
<a href="#"><u>Grupo::update()</u></a>	102
1) Prepara el formulario para la modificación de un grupo existente	
2) Realiza las validaciones necesarias sobre cada campo del registro	
<a href="#"><u>Grupo::view()</u></a>	102
Visualiza los datos del grupo recibido	
<a href="#"><u>Georeferencia_model::getMsgError()</u></a>	205
Devuelve el mensaje de error, en caso de existir un error despues de ejecutar un metodo, de lo contrario false	
<a href="#"><u>Georeferencia_model::process()</u></a>	205
Inserta los registros contenidos en la tabla cat_georeferencia a la tablaasu_georeferencia	
<a href="#"><u>Grupo_model</u></a>	205
Modelo Grupo	
<a href="#"><u>Georeferencia_model</u></a>	204
Modelo Georeferencia	
<a href="#"><u>grupo_model.php</u></a>	132
<a href="#"><u>Grupo:: ifGroupExists()</u></a>	102
Callback para validar que un nombre de grupo no se duplique	
<a href="#"><u>georeferencia_model.php</u></a>	131
<a href="#"><u>graph.php</u></a>	42

## H

<a href="#"><u>hemoglobina_model.php</u></a>	621
Código fuente	
<a href="#"><u>Hemoglobina_model::process()</u></a>	211
Inserta los registros contenidos en la tabla cat_georeferencia a la tablaasu_georeferencia	
<a href="#"><u>Hemoglobina_model::getMsgError()</u></a>	210
Devuelve el mensaje de error, en caso de existir un error despues de ejecutar un metodo, de lo contrario false	
<a href="#"><u>Hemoglobina_model</u></a>	210
Modelo Hemoglobina	
<a href="#"><u>hemoglobina_model.php</u></a>	133

<b>I</b>	
<a href="#">index.php</a>	388
Código fuente	
<a href="#">index.php</a>	389
Código fuente	
<a href="#">Index::index()</a>	4
<a href="#">Index</a>	4
<a href="#">index.php</a>	3
<a href="#">index.php</a>	2

<b>M</b>	
<a href="#">Menu_model::getRuta()</a>	215
<a href="#">Menu_model::hasChild()</a>	215
<i>Verifica si el nodo actual tiene hijos</i>	
<a href="#">Menu_model::insert()</a>	216
<i>Inserta en la base de datos, la informacion contenida en el objeto</i>	
<a href="#">Menu_model::resetFilter()</a>	216
<i>Elimina todos los filtros registrados</i>	
<a href="#">Menu_model::getNumRows()</a>	215
<i>Obtiene el numero total de registros en la tabla Menu en caso de existir filtros, estos son aplicados a la consulta</i>	
<a href="#">Menu_model::getNombre()</a>	215
<a href="#">Menu_model::getId_padre()</a>	214
<a href="#">Menu_model::getId_raiz()</a>	214
<a href="#">Menu_model::getMsgError()</a>	214
<i>Devuelve el mensaje de error, en caso de existir un error despues de ejecutar un metodo, de lo contrario false</i>	
<a href="#">Menu_model::setAtributo()</a>	216
<a href="#">Menu_model::setId()</a>	216
<a href="#">Menu_model::update()</a>	218
<i>Actualiza los datos del objeto actual</i>	
<a href="#">menubuilder.php</a>	422
Código fuente	
<a href="#">menu.php</a>	502
Código fuente	
<a href="#">menu_model.php</a>	623
Código fuente	
<a href="#">Menu_model::setRuta()</a>	218
<a href="#">Menu_model::setNombre()</a>	217
<a href="#">Menu_model::setId_controlador()</a>	217
<a href="#">Menu_model::setId_padre()</a>	217
<a href="#">Menu_model::setId_raiz()</a>	217
<a href="#">Menu_model::getId_controlador()</a>	214
<a href="#">Menu_model::getId()</a>	214
<a href="#">Menu</a>	103
<i>Controlador Menu</i>	
<a href="#">Menu::delete()</a>	103
<i>Eliminar el registro especificado por el id</i>	
<a href="#">Menu::index()</a>	103

<i>Lista todos los registros de la menu, con su correspondiente paginación</i>	
<a href="#">Menu::insert()</a>	104
<i>Muestra el formulario para crear un nuevo registro en la menu,     las variables se obtienen por el metodo POST</i>	
<a href="#">menu.php</a>	69
<a href="#">Menubuilder::isGranted()</a>	41
<i>Función que valida si se visualiza o no la acción especificada (usada en views)</i>	
<a href="#">Menubuilder</a>	40
<i>Menu Builder</i>	
<a href="#">Menubuilder::build()</a>	40
<i>Construye el menu basandose en los permisos del usuario logeado</i>	
<a href="#">Menubuilder::crearMenu()</a>	41
<i>Función recursiva que recorre todo el árbol de elementos del menu</i>	
<a href="#">Menu::update()</a>	104
<i>Muestra el formulario con los datos del registro especificado por el id,     para actualizar sus datos</i>	
<a href="#">Menu::view()</a>	105
<i>Muestra los datos del registro especificado por el id</i>	
<a href="#">Menu_model::getAll()</a>	212
<i>Obtiene todos los registros de la tabla Menu     en caso de existir filtros, estos son aplicados a la consulta</i>	
<a href="#">Menu_model::getAtributo()</a>	213
<a href="#">Menu_model::getById()</a>	213
<i>Obtiene los datos del registro de la menu que tiene el ID especificado</i>	
<a href="#">Menu_model::getByPadre()</a>	213
<i>Obtiene todos los nodos hijos de un padre</i>	
<a href="#">Menu_model::deleteByFilter()</a>	212
<i>Elimina el conjunto de registros que cumplen con el o los criterios de filtrado</i>	
<a href="#">Menu_model::delete()</a>	212
<i>Elimina el registro actual de la base de datos</i>	
<a href="#">menu_model.php</a>	134
<a href="#">Menu_model</a>	211
<i>Modelo Menu</i>	
<a href="#">Menu_model::addFilter()</a>	212
<i>Agrega una nueva regla de filtrado al arreglo de filtros</i>	
<a href="#">menubuilder.php</a>	39

## N

<a href="#">Notificacion_model::insert()</a>	343
<i>Inserta en la base de datos los datos de la notificación (datos en propiedades)</i>	
<a href="#">Notificacion_model::setContenido()</a>	343
<a href="#">Notificacion_model::getTitulo()</a>	343
<a href="#">Notificacion_model::getNumRows()</a>	343
<i>Obtiene el numero total de notificaciones</i>	
<a href="#">Notificacion_model::getIdsTabletas()</a>	342
<a href="#">Notificacion_model::getMsgError()</a>	342
<i>Asigna el mensaje de error a visualizar: para usuario final (usr) o para bitácora (log)</i>	
<a href="#">Notificacion_model::setFechaFin()</a>	344
<a href="#">Notificacion_model::setFechalnicio()</a>	344
<a href="#">notificacion.php</a>	677
<i>Código fuente</i>	

<a href="#">notificacion_model.php</a>	769
Código fuente	
<a href="#">Notificacion_model::update()</a>	345
Actualiza en la base de datos los datos de la notificación (datos en propiedades)	
<a href="#">Notificacion_model::setTitulo()</a>	345
<a href="#">Notificacion_model::setId()</a>	344
<a href="#">Notificacion_model::setIdsTabletas()</a>	345
<a href="#">Notificacion_model::getId()</a>	342
<a href="#">Notificacion_model::getFechaInicio()</a>	342
<a href="#">Notificacion::update()</a>	268
1) Prepara el formulario para la modificación de una notificación existente	
2) Realiza las validaciones necesarias sobre cada campo del registro	
<a href="#">Notificacion::view()</a>	268
Visualiza los datos de la notificación recibida	
<a href="#">Notificacion::insert()</a>	267
1) Prepara el formulario para la inserción de una notificación nueva	
2) Realiza las validaciones necesarias sobre cada campo del registro	
<a href="#">Notificacion::index()</a>	267
1) Visualiza las notificaciones existentes para su interacción CRUD	
2) En caso de detectar un texto a buscar se filtran las notificaciones existentes acorde a la búsqueda	
<a href="#">Notificacion</a>	266
Controlador Notificación	
<a href="#">Notificacion::delete()</a>	267
Solicita la eliminación de la notificación recibida	
<a href="#">notificacion_model.php</a>	287
<a href="#">Notificacion_model</a>	340
Modelo Usuario	
<a href="#">Notificacion_model::getContenido()</a>	342
<a href="#">Notificacion_model::getFechaFin()</a>	342
<a href="#">Notificacion_model::getById()</a>	341
Obtiene la notificación solicitada	
<a href="#">Notificacion_model::getAll()</a>	341
Obtiene todas las notificaciones existentes, se puede filtrar por: texto a buscar si se desea	
<a href="#">Notificacion_model::addFilter()</a>	340
Agrega una nueva regla de filtrado al arreglo de filtros	
<a href="#">Notificacion_model::delete()</a>	341
Elimina de la base de datos la notificación (id en propiedades)	
<a href="#">notificacion.php</a>	249

## O

<a href="#">Obtenercurp::curp()</a>	48
Consulta si la curp existe en la base de datos de la condusef	
<a href="#">obtenercurp.php</a>	426
Código fuente	
<a href="#">Obtenercurp::calcular_curp()</a>	47
Calcula la curp y el rfc con los datos proporcionados	
<a href="#">Obtenercurp::calculacurp()</a>	47
Calcula la curp y el rfc con los datos proporcionados	
<a href="#">Obtenercurp</a>	45
Controlador Objeto	

<a href="#">Obtenercurp::\$estados</a>	46
Arreglo con los estados y sus abreviaturas, sirven para el cálculo de la CURP	
<a href="#">obtenercurp.php</a>	43

## P

<a href="#">Permiso_model::setIdControladorAccion()</a>	221
<a href="#">Permiso_model::setIdGrupo()</a>	222
<a href="#">Permiso_model::setId()</a>	221
<a href="#">Permiso_model::setFecha()</a>	221
<a href="#">Permiso_model::insertBatch()</a>	221
Inserta en la base de datos el arreglo de permisos recibido	
<a href="#">Poblacion_model</a>	222
Modelo Poblacion	
<a href="#">Poblacion_model::getMsgError()</a>	223
Devuelve el mensaje de error, en caso de existir un error despues de ejecutar un metodo, de lo contrario false	
<a href="#">permiso_model.php</a>	632
Código fuente	
<a href="#">poblacion_model.php</a>	635
Código fuente	
<a href="#">permiso.php</a>	508
Código fuente	
<a href="#">Pagination.php</a>	408
Código fuente	
<a href="#">Poblacion_model::process()</a>	223
Inserta los registros contenidos en la tabla cat_poblacion a la tablaasu_poblacion	
<a href="#">Permiso_model::getPermission()</a>	220
Obtiene el permiso solicitado	
<a href="#">Permiso_model::getMsgError()</a>	220
Asigna el mensaje de error a visualizar: para usuario final (usr) o para bitácora (log)	
<a href="#">permiso_model.php</a>	135
<a href="#">poblacion_model.php</a>	136
<a href="#">Permiso::index()</a>	106
1) Visualiza los entornos existentes para su selección	
<a href="#">Permiso</a>	105
Controlador Permiso	
<a href="#">permiso.php</a>	70
<a href="#">Permiso_model</a>	219
Modelo Permiso	
<a href="#">Permiso_model::deletePermissions()</a>	219
Elimina de la base de datos los permisos del entorno y grupo recibidos	
<a href="#">Permiso_model::getIdGrupo()</a>	220
<a href="#">Permiso_model::getIdControladorAccion()</a>	220
<a href="#">Permiso_model::getId()</a>	219
<a href="#">Permiso_model::getFecha()</a>	219
<a href="#">Pagination.php</a>	7
CodeIgniter	

# R

<a href="#">Reporte_sincronizacion::lote_view()</a>	271
<i>Muestra detallada por cada list del reporte de lote de vacunacion</i>	
<a href="#">Reporte_sincronizacion::view()</a>	271
<i>Muestra detallada por cada list del reporte de sincronizacion</i>	
<a href="#">Reporte_sincronizacion::lote()</a>	270
<i>Genera el item del reporte de lote de vacunacion</i>	
<a href="#">Reporte_sincronizacion::index()</a>	270
<i>Este es el metodo por default, crea el listado del reporte de sincronizacion</i>	
<a href="#">Reporteador::view()</a>	269
<i>Renderiza la vista del reporte</i>	
<a href="#">Reporte_sincronizacion</a>	270
<i>Controller Usuario</i>	
<a href="#">reporteador_model.php</a>	288
<a href="#">reporte_censo_nominal.php</a>	289
<a href="#">Reporteador_model::getConcentradoActividades()</a>	347
<a href="#">Reporteador_model::getEsquemasIncompletos()</a>	347
<a href="#">Reporteador_model::getCoberturaBiologicoListado()</a>	346
<i>Obtiene el reporte de cobertura por tipo de biológico</i>	
<a href="#">Reporteador_model::getCensoNominal()</a>	346
<a href="#">reporte_sincronizacion_model.php</a>	290
<a href="#">Reporteador_model</a>	345
<i>Modelo Reporteador</i>	
<a href="#">Reporteador::index()</a>	269
<i>Visualiza los reportes existentes</i>	
<a href="#">Reporteador</a>	269
<i>Controlador Reporteador</i>	
<a href="#">ReglaVacuna_model::setIdVacuna()</a>	234
<a href="#">ReglaVacuna_model::setIdVacunaPrevia()</a>	234
<a href="#">ReglaVacuna_model::setId()</a>	233
<a href="#">ReglaVacuna_model::setForzarAplicacion()</a>	233
<a href="#">ReglaVacuna_model::setDosis()</a>	232
<a href="#">ReglaVacuna_model::setEsqComp()</a>	233
<a href="#">ReglaVacuna_model::setIdViaVacuna()</a>	234
<a href="#">ReglaVacuna_model::setObservacionRegion()</a>	234
<a href="#">reporteador.php</a>	250
<a href="#">reporte_sincronizacion.php</a>	251
<a href="#">ReglaVacuna_model::update()</a>	235
<i>Actualiza el objeto actual en la base de datos</i>	
<a href="#">ReglaVacuna_model::setRegion()</a>	235
<a href="#">ReglaVacuna_model::setOrdenEsqComp()</a>	235
<a href="#">Reporteador_model::getGrupoVacunas()</a>	347
<a href="#">Reporteador_model::getMsgError()</a>	347
<i>Asigna el mensaje de error a visualizar: para usuario final (usr) o para bitácora (log)</i>	
<a href="#">raiz.php</a>	510
<i>Código fuente</i>	
<a href="#">reglavacuna.php</a>	521
<i>Código fuente</i>	
<a href="#">Reporte_sincronizacion_model::get_version()</a>	352
<i>obtiene la ultima version de la apk de las tabletas</i>	
<a href="#">Reporte_sincronizacion_model::getMsgError()</a>	352
<a href="#">Reporte_sincronizacion_model::getCount()</a>	351

<i>obtiene el numero de registros de una tabla o consulta</i>	
<a href="#">Reporte_sincronizacion_model::getListado()</a>	351
<i>Obtiene el resultado de una consulta</i>	
<a href="#">raiz_model.php</a>	637
<i>Código fuente</i>	
<a href="#">reglavacuna_model.php</a>	641
<i>Código fuente</i>	
<a href="#">reporte_censo_nominal.php</a>	784
<i>Código fuente</i>	
<a href="#">reporte_sincronizacion_model.php</a>	785
<i>Código fuente</i>	
<a href="#">reporteador_model.php</a>	775
<i>Código fuente</i>	
<a href="#">reporte_sincronizacion.php</a>	685
<i>Código fuente</i>	
<a href="#">reporteador.php</a>	682
<i>Código fuente</i>	
<a href="#">Reporte_sincronizacion_model</a>	351
<i>Modelo Usuario</i>	
<a href="#">Reporte_censo_nominal::\$vacunas</a>	350
<a href="#">Reporte_censo_nominal</a>	348
<i>Modelo Reporte_censo_nominal</i>	
<a href="#">Reporte_censo_nominal::\$apellido_materno</a>	349
<a href="#">Reporteador_model::object_to_array()</a>	348
<a href="#">Reporteador_model::getVacunasByGrupo()</a>	348
<a href="#">Reporteador_model::getSeguimientoRV1RV5()</a>	348
<a href="#">Reporteador_model::getVacunas()</a>	348
<a href="#">Reporte_censo_nominal::\$apellido_paterno</a>	349
<a href="#">Reporte_censo_nominal::\$curp</a>	349
<a href="#">Reporte_censo_nominal::\$parto_multiple</a>	350
<a href="#">Reporte_censo_nominal::\$sexo</a>	350
<a href="#">Reporte_censo_nominal::\$nombre</a>	350
<a href="#">Reporte_censo_nominal::\$fecha_nacimiento</a>	349
<a href="#">Reporte_censo_nominal::\$domicilio</a>	349
<a href="#">ReglaVacuna_model::setDialnicioPrevia()</a>	232
<a href="#">ReglaVacuna_model::setDialnicioNacido()</a>	232
<a href="#">ReglaVacuna::view()</a>	113
<i>Acción para visualizar información de una regla específica, obtiene el objeto regla_vacuna por medio del id proporcionado.</i>	
<a href="#">raiz_model.php</a>	137
<a href="#">ReglaVacuna::update()</a>	112
<i>Acción para preparar la actualización de una regla ya existente,</i>	
<a href="#">ReglaVacuna::insert()</a>	112
<i>Acción para preparar la inserción de nuevas reglas , realiza la validación del formulario del lado cliente</i>	
<a href="#">ReglaVacuna::delete()</a>	111
<i>Acción para eliminar una regla, recibe el id de la regla a eliminar</i>	
<a href="#">ReglaVacuna::index()</a>	112
<i>Acción por default del controlador, carga la lista de reglas de vacunas disponibles y una lista de opciones No recibe parámetros</i>	
<a href="#">reglavacuna_model.php</a>	138
<a href="#">Raiz_model</a>	223

<i>Modelo Raiz</i>	
<a href="#"><u>Raiz_model::getById()</u></a>	225 <i>Devuelve la información de una raiz por su ID</i>
<a href="#"><u>Raiz_model::getDescripcion()</u></a>	225
<a href="#"><u>Raiz_model::getAll()</u></a>	224 <i>Devuelve todos los registros de la tabla raiz</i>
<a href="#"><u>Raiz_model::ExistInArbol()</u></a>	224 <i>Revisa si la raiz pasada como parametro existe en el ASU</i>
<a href="#"><u>Raiz_model::delete()</u></a>	224 <i>Elimina el registro actual de la base de datos</i>
<a href="#"><u>ReglaVacuna</u></a>	111 <i>Controlador ReglaVacuna</i>
<a href="#"><u>Raiz::view()</u></a>	110 <i>Acción para visualizar información de una raiz específica, obtiene el objeto raiz por medio del id proporcionado.</i>
<a href="#"><u>Raiz::getChildrenFromLevel()</u></a>	107 <i>Acción para regresar el arbol de segmentacion determinado, el objeto regresado contiene estructura de arbol y es consumida solamente por peticiones AJAX</i>
<a href="#"><u>Raiz::getDataKeyValue()</u></a>	108 <i>Acción para obtener los registros de un ASU determinado en cierto nivel y con un ID de filtro</i>
<a href="#"><u>Raiz::delete()</u></a>	107 <i>Acción para eliminar una raiz, recibe el id de la raiz a eliminar</i>
<a href="#"><u>Raiz::createasu()</u></a>	106 <i>Acción para crear el ASU a partir de una raiz Solo se permite su acceso por medio de peticiones AJAX</i>
<a href="#"><u>reglavacuna.php</u></a>	72
<a href="#"><u>Raiz</u></a>	106 <i>Controlador Raiz</i>
<a href="#"><u>Raiz::getDataTreeFromId()</u></a>	108 <i>Acción para regresar la descripción e informacion adicional de un arreglo de ID's desde el arbol de segmentacion</i>
<a href="#"><u>Raiz::getTreeBlock()</u></a>	108 <i>Sirve para obtener bloques del arbol de segmentación única ASU</i>
<a href="#"><u>Raiz::update()</u></a>	110 <i>Acción para preparar la actualización de una raiz ya existente,</i>
<a href="#"><u>Raiz::updateasu()</u></a>	110 <i>Acción para actualizar el ASU a partir de una raiz Solo se permite su acceso por medio de peticiones AJAX</i>
<a href="#"><u>Raiz::insert()</u></a>	109 <i>Acción para preparar la inserción de nuevas acciones , realiza la validación del formulario del lado cliente</i>
<a href="#"><u>Raiz::iniciarasu()</u></a>	109 <i>Crea los archivos JSON necesarios para iniciar el ASU en caché y agilizar su carga</i>
<a href="#"><u>Raiz::index()</u></a>	109 <i>Acción por default del controlador, carga la lista de Raices disponibles y una lista de opciones No recibe parámetros</i>
<a href="#"><u>Raiz_model::getId()</u></a>	225 *****
<a href="#"><u>Raiz_model::getMsgError()</u></a>	225 <i>Devuelve los mensajes de error en caso de ocurrir alguna excepción 'usr' devuelve el mensaje para la vista de usuario</i>

'log' devuelve el mensaje para el log de errores	
<a href="#">ReglaVacuna_model::getIdViaVacuna()</a>	230
<a href="#">ReglaVacuna_model::getMsgError()</a>	230
Devuelve los mensajes de error en caso de ocurrir alguna excepción	
'usr' devuelve el mensaje para la vista de usuario	
'log' devuelve el mensaje para el log de errores	
<a href="#">ReglaVacuna_model::getIdVacunaPrevia()</a>	230
<a href="#">ReglaVacuna_model::getIdVacuna()</a>	229
<a href="#">ReglaVacuna_model::getForzarAplicacion()</a>	229
<a href="#">ReglaVacuna_model::getId()</a>	229
*****	
<a href="#">ReglaVacuna_model::getObservacionRegion()</a>	230
<a href="#">ReglaVacuna_model::getOrdenEsqComp()</a>	231
<a href="#">ReglaVacuna_model::setDiaFinNacido()</a>	231
<a href="#">ReglaVacuna_model::setDiaFinPrevia()</a>	232
<a href="#">ReglaVacuna_model::setAlergias()</a>	231
<a href="#">ReglaVacuna_model::insert()</a>	231
Inserta en la tabla regla_vacuna la información contenida en el objeto	
<a href="#">ReglaVacuna_model::getRegion()</a>	231
<a href="#">ReglaVacuna_model::getEsqComp()</a>	229
<a href="#">ReglaVacuna_model::getDosis()</a>	229
<a href="#">ReglaVacuna_model</a>	227
Modelo ReglaVacuna	
<a href="#">ReglaVacuna_model::delete()</a>	227
Elimina el registro actual de la base de datos	
<a href="#">Raiz_model::update()</a>	226
Actualiza el objeto actual en la base de datos	
<a href="#">Raiz_model::setId()</a>	226
<a href="#">Raiz_model::insert()</a>	226
Inserta en la tabla raiz, la información contenida en el objeto	
<a href="#">Raiz_model::setDescripcion()</a>	226
<a href="#">ReglaVacuna_model::getAlergias()</a>	227
<a href="#">ReglaVacuna_model::getAll()</a>	228
Devuelve todos los registros de la tabla regla_vacuna	
<a href="#">ReglaVacuna_model::getDiaInicioNacido()</a>	229
<a href="#">ReglaVacuna_model::getDiaInicioPrevia()</a>	229
<a href="#">ReglaVacuna_model::getDiaFinPrevia()</a>	228
<a href="#">ReglaVacuna_model::getDiaFinNacido()</a>	228
<a href="#">ReglaVacuna_model::getById()</a>	228
Devuelve la información de una regla de vacuna por su ID	
<a href="#">raiz.php</a>	71

## S

<a href="#">semana_nacional_model.php</a>	291
<a href="#">Servicios::Synchronization()</a>	279
Metodo principal al que se le hacen las peticiones y es el que se encarga de distribuir la información	
<a href="#">Semana_nacional_model</a>	352
Modelo Tableta	
<a href="#">Semana_nacional_model::delete()</a>	353
Elimina el registro actual de la base de datos	

<a href="#">Semana nacional model::getById()</a>	353
Obtiene los datos del registro que tiene el ID especificado	
<a href="#">Semana nacional model::getAll()</a>	353
Obtiene todos los registros de la tabla	
<a href="#">Servicios::ss_step_6()</a>	278
prepara los datos para la sincronizacion secuencia, envia unicamente aquellos datos modificados despues de la ultima sincronizacion de la tableta	
<a href="#">Servicios::ss_step_5()</a>	278
Recibe los datos que genero la tableta para ser almacenados en la base de datos del servidor	
<a href="#">Servicios::is_step_2()</a>	276
Valida que la session este activa, genera los catalogos a enviar en la sincronizacion por primera vez	
<a href="#">Servicios::is_step_10()</a>	276
valida que la tableta este asignada a una unidad medica y que tenga un status valido para la sincronizacion	
<a href="#">Servicios::is_step_3()</a>	277
Guarda en la base de datos el estado de la sincronizacion	
<a href="#">Servicios::is_step_4()</a>	277
Prepara la informacion de personas con su catalogos transaccionales de cada una	
<a href="#">Servicios::prueba2()</a>	278
<a href="#">Semana nacional model::getDescripcion()</a>	354
<a href="#">Semana nacional model::getFecha_fin()</a>	354
<a href="#">Semana nacional model::update()</a>	356
Actualiza los datos del objeto actual	
<a href="#">Semana nacional model::setFecha_inicio()</a>	356
<a href="#">Session.php</a>	434
Código fuente	
<a href="#">semana_nacional.php</a>	693
Código fuente	
<a href="#">semana_nacional_model.php</a>	787
Código fuente	
<a href="#">servicios.php</a>	698
Código fuente	
<a href="#">Semana nacional model::setFecha_fin()</a>	356
<a href="#">Semana nacional model::setDescripcion()</a>	355
<a href="#">Semana nacional model::getId()</a>	354
<a href="#">Semana nacional model::getFecha_inicio()</a>	354
<a href="#">Semana nacional model::getMsgError()</a>	354
Devuelve el mensaje de error, en caso de existir un error despues de ejecutar un metodo, de lo contrario false	
<a href="#">Semana nacional model::getNumRows()</a>	355
Obtiene el numero total de registros en la tabla	
<a href="#">Semana nacional model::insert()</a>	355
Inserta en la base de datos, la informacion contenida en el objeto	
<a href="#">Servicios::is_step_0()</a>	275
Paso 0 se procesa las peticiones segun la accion:	
<a href="#">Servicios::esquema_incompleto()</a>	275
Genera los esquemas incompletos de las personas que correspondan a la unidad medica de la tableta	
<a href="#">Session::is_expired()</a>	54
Check if session is expired	

<a href="#"><u>Session::flashdata()</u></a>	<i>Fetch a specific flashdata item from the session array</i>	. . . . . 54
<a href="#"><u>Session::keep_flashdata()</u></a>	<i>Keeps existing flashdata available to next request.</i>	. . . . . 54
<a href="#"><u>Session::sess_create()</u></a>	<i>Create Session</i>	. . . . . 55
<a href="#"><u>Session::set_flashdata()</u></a>	<i>Add or change flashdata, only available until the next request</i>	. . . . . 55
<a href="#"><u>Session::sess_destroy()</u></a>	<i>Destroy session</i>	. . . . . 55
<a href="#"><u>Session::all_userdata()</u></a>	<i>Fetch all session data</i>	. . . . . 53
<a href="#"><u>Session::\$store</u></a>		. . . . . 53
<a href="#"><u>Session::\$ci</u></a>		. . . . . 52
<a href="#"><u>Session</u></a>	<i>CodeIgniter Native Session Library</i>	. . . . . 52
<a href="#"><u>Session::\$flashdata_key</u></a>		. . . . . 52
<a href="#"><u>Session::\$sess_expiration</u></a>		. . . . . 53
<a href="#"><u>Session::\$sess_namespace</u></a>		. . . . . 53
<a href="#"><u>Session::set_userdata()</u></a>	<i>Set value for specific user data element</i>	. . . . . 55
<a href="#"><u>Session::unset_userdata()</u></a>	<i>remove array value for specific user data element</i>	. . . . . 56
<a href="#"><u>Semana_nacional::update()</u></a>	<i>Muestra el formulario con los datos del registro especificado por el id, para actualizar sus datos</i>	. . . . . 273
<a href="#"><u>Semana_nacional::insert()</u></a>	<i>Muestra el formulario para crear un nuevo registro en la semana_nacional, las variables se obtienen por el metodo POST</i>	. . . . . 273
<a href="#"><u>Semana_nacional::view()</u></a>	<i>Muestra los datos del registro especificado por el id</i>	. . . . . 273
<a href="#"><u>Servicios</u></a>	<i>Controlador Servicios</i>	. . . . . 274
<a href="#"><u>Servicios::catalogos_relevantes()</u></a>	<i>Genera los catalogos relevantes por entorno</i>	. . . . . 275
<a href="#"><u>Servicios::actualiza_estado_tableta()</u></a>	<i>Actualiza el estatus de la tabla</i>	. . . . . 274
<a href="#"><u>Semana_nacional::index()</u></a>	<i>Lista todos los registros de semanas nacional, con su correspondiente paginación permite eliminar un elemento individual, muestra enlaces para actualizar y ver detalles de un elemento específico</i>	. . . . . 272
<a href="#"><u>Semana_nacional::getAll()</u></a>	<i>Devuelve un json con todos los registros de semanas nacional</i>	. . . . . 272
<a href="#"><u>semana_nacional.php</u></a>		. . . . . 252
<a href="#"><u>Session::userdata()</u></a>	<i>Get specific user data element</i>	. . . . . 56
<a href="#"><u>servicios.php</u></a>		. . . . . 253
<a href="#"><u>Semana_nacional</u></a>	<i>Controlador Semana Nacional</i>	. . . . . 271
<a href="#"><u>Semana_nacional::delete()</u></a>	<i>Eliminar el registro especificado por el id</i>	. . . . . 272
<a href="#"><u>Session.php</u></a>		. . . . . 51

# T

<a href="#">Tableta_model::setId_tipo_censo()</a>	362
<a href="#">Tableta_model::setId_tes_estado_tableta()</a>	361
<a href="#">Tableta_model::setMac()</a>	362
<a href="#">Tableta_model::setPeriodo_esq_inc()</a>	362
<a href="#">Tableta_model::setUsuarios_asignados()</a>	363
<a href="#">Tableta_model::setUltima_actualizacion()</a>	363
<a href="#">Tableta_model::setId_asu_um()</a>	361
<a href="#">Tableta_model::setIdVersion()</a>	361
<a href="#">Tableta_model::getUltima_actualizacion()</a>	360
<a href="#">Tableta_model::getPeriodo_esq_inc()</a>	360
<a href="#">Tableta_model::getUsuarios_asignados()</a>	360
<a href="#">Tableta_model::insert()</a>	360
<i>Inserta en la base de datos, la informacion contenida en el objeto</i>	
<a href="#">Tableta_model::setId()</a>	361
<a href="#">Tableta_model::update()</a>	363
<i>Actualiza los datos del objeto actual</i>	
<a href="#">Tipo_censo_model</a>	364
Modelo Tipo_censo	
<a href="#">tree.php</a>	432
Código fuente	
<a href="#">template.php</a>	413
Código fuente	
<a href="#">tableta.php</a>	716
Código fuente	
<a href="#">tableta_model.php</a>	792
Código fuente	
<a href="#">tipo_censo_model.php</a>	800
Código fuente	
<a href="#">Tipo_censo_model::setId()</a>	366
<a href="#">Tipo_censo_model::setDescripcion()</a>	365
<a href="#">Tipo_censo_model::getById()</a>	364
<i>Obtiene los datos del registro que tiene el ID especificado</i>	
<a href="#">Tipo_censo_model::getAll()</a>	364
<i>Obtiene todos los registros de la tabla</i>	
<a href="#">Tipo_censo_model::getDescripcion()</a>	365
<a href="#">Tipo_censo_model::getId()</a>	365
<a href="#">Tipo_censo_model::getMsgError()</a>	365
<i>Devuelve el mensaje de error, en caso de existir un error despues de ejecutar un metodo, de lo contrario false</i>	
<a href="#">Tableta_model::getNumRows()</a>	360
<i>Obtiene el numero total de registros en la tabla en caso de existir filtros, estos son aplicados a la consulta</i>	
<a href="#">Tableta_model::getMsgError()</a>	359
<i>Devuelve el mensaje de error, en caso de existir un error despues de ejecutar un metodo, de lo contrario false</i>	
<a href="#">Tableta::setUM()</a>	281
<i>Asignar unidad medica y tipo de censo</i>	
<a href="#">Tableta::insert()</a>	281
<i>Muestra el formulario para crear un nuevo registro en la tableta,</i>	

*las variables se obtienen por el metodo POST*

<a href="#"><u>Tableta::update()</u></a>	281
<i>Muestra el formulario con los datos del registro especificado por el id, para actualizar sus datos</i>	
<a href="#"><u>Tableta::uploadFile()</u></a>	282
<i>Registra tabletas desde un archivo csv</i>	
<a href="#"><u>Tableta::view()</u></a>	282
<i>Muestra los datos del registro especificado por el id</i>	
<a href="#"><u>Tableta::index()</u></a>	280
<i>Lista todos los registros de tabletas, con su correspondiente paginación, permite eliminar un conjunto de registro o un elemento individual, muestra enlaces para actualizar y ver detalles de un elemento específico</i>	
<a href="#"><u>Tableta::delete()</u></a>	280
<i>Eliminar el registro especificado por el id</i>	
<a href="#"><u>Tree</u></a>	48
<i>Controlador Objeto</i>	
<a href="#"><u>tree.php</u></a>	44
<a href="#"><u>Tree::create()</u></a>	49
<i>Crea el arbol y lo muestra en la view</i>	
<a href="#"><u>tableta.php</u></a>	254
<a href="#"><u>Tableta</u></a>	279
<i>Controlador Tableta</i>	
<a href="#"><u>Tableta:: validateMac()</u></a>	282
<i>Valida si existe una MAC en la base de datos</i>	
<a href="#"><u>tableta_model.php</u></a>	292
<a href="#"><u>Tableta_model::getId_asu_um()</u></a>	359
<a href="#"><u>Tableta_model::getIdVersion()</u></a>	359
<a href="#"><u>Tableta_model::getId_tes_estado_tableta()</u></a>	359
<a href="#"><u>Tableta_model::getId_tipo_censo()</u></a>	359
<a href="#"><u>Tableta_model::getMac()</u></a>	359
<a href="#"><u>Tableta_model::getId()</u></a>	358
<a href="#"><u>Tableta_model::getByMac()</u></a>	358
<i>Obtiene los datos del registro la MAC especificada</i>	
<a href="#"><u>Tableta_model</u></a>	357
<i>Modelo Tableta</i>	
<a href="#"><u>tipo_censo_model.php</u></a>	293
<a href="#"><u>Tableta_model::delete()</u></a>	357
<i>Elimina el registro actual de la base de datos</i>	
<a href="#"><u>Tableta_model::getAll()</u></a>	357
<i>Obtiene todos los registros de la tabla</i>	
<a href="#"><u>Tableta_model::getById()</u></a>	358
<i>Obtiene los datos del registro que tiene el ID especificado</i>	
<a href="#"><u>template.php</u></a>	8
<i>CodeIgniter</i>	

## U

<a href="#"><u>Usuario_model::setId()</u></a>	244
<a href="#"><u>Usuario_model::setCorreo()</u></a>	244
<a href="#"><u>Usuario_model::setClave()</u></a>	244
<a href="#"><u>Usuario_model::setApellidoPaterno()</u></a>	243
<a href="#"><u>Usuario_model::setIdGrupo()</u></a>	245

<a href="#">Usuario_model::setNombre()</a>	245
<a href="#">Usuario_model::update_pass()</a>	246
<a href="#">Usuario_model::update()</a>	245
Actualiza en la base de datos los datos del usuario (datos en propiedades)	
<a href="#">Usuario_model::setNombreUsuario()</a>	245
<a href="#">Usuario_model::setApellidoMaterno()</a>	243
<a href="#">Usuario_model::setActivo()</a>	243
<a href="#">Usuario_model::getOnlyActives()</a>	241
Obtiene todos los usuarios existentes, se puede filtrar por: texto a buscar o solo activos si se desea	
<a href="#">Usuario_model::getNumRows()</a>	241
Obtiene el numero total de usuarios	
<a href="#">Usuario_model::getNombreUsuario()</a>	241
<a href="#">Usuario_model::getUser()</a>	242
<a href="#">Usuario_model::get_grupo_entorno()</a>	242
<a href="#">Usuario_model::insert()</a>	243
Inserta en la base de datos los datos del usuario (datos en propiedades)	
<a href="#">Usuario_model::get_usuario_entorno()</a>	242
<a href="#">Usuario_model::get_permiso_entorno()</a>	242
<a href="#">Usuario_model::update_user()</a>	246
<a href="#">usuario_tableta.php</a>	255
<a href="#">Usuario_tableta_model::setTableta()</a>	368
<a href="#">Usuario_tableta_model::insert()</a>	368
Inserta en la base de datos, la informacion contenida en el objeto	
<a href="#">Usuario_tableta_model::getUsuariosByTableta()</a>	368
Obtiene el id de todos los usuarios asignados a la tableta especificada	
<a href="#">Usuario_tableta_model::getUsuario()</a>	368
<a href="#">Usuario_tableta_model::setUsuario()</a>	369
<a href="#">usuario.php</a>	527
Código fuente	
<a href="#">usuario_tableta_model.php</a>	803
Código fuente	
<a href="#">usuario_tableta.php</a>	723
Código fuente	
<a href="#">usuario_model.php</a>	648
Código fuente	
<a href="#">Usuario_tableta_model::getTabletasByUsuario()</a>	367
Obtiene el id de todas las tabletas relacionadas con el usuario especificado	
<a href="#">Usuario_tableta_model::getTableta()</a>	367
<a href="#">Usuario_tableta::index()</a>	283
Lista todos los registros de usuarios correspondientes a una tableta en especifico permite eliminar un conjunto de registro o un elemento individual, muestra enlaces para actualizar y ver detalles de un elemento especifico	
<a href="#">Usuario_tableta::delete()</a>	283
Eliminar un usuario de una tableta especifica	
<a href="#">Usuario_tableta</a>	283
Controlador Usuario_tableta	
<a href="#">Usuario_tableta::insert()</a>	284
Muestra el formulario para crear un nuevo registro en la tableta, las variables se obtienen por el metodo POST	
<a href="#">usuario_tableta_model.php</a>	294
<a href="#">Usuario_tableta_model::getMsgError()</a>	367
Devuelve el mensaje de error,	

<i>en caso de existir un error despues de ejecutar un metodo, de lo contrario false</i>	
<a href="#"><u>Usuario_tableta_model::delete()</u></a>	366
<i>Elimina la relacion entre usuario y tabla</i>	
<a href="#"><u>Usuario_tableta_model</u></a>	366
<i>Modelo Estado_tableta</i>	
<a href="#"><u>Usuario_model::getNombre()</u></a>	240
<a href="#"><u>Usuario_model::getMsgError()</u></a>	240
<i>Asigna el mensaje de error a visualizar: para usuario final (usr) o para bitácora (log)</i>	
<a href="#"><u>Usuario::logout()</u></a>	116
<i>Termina la sesión</i>	
<a href="#"><u>Usuario::login()</u></a>	116
<i>Ofrece el inicio de sesión</i>	
<a href="#"><u>Usuario::load_update()</u></a>	116
<a href="#"><u>Usuario::insert()</u></a>	115
1) <i>Prepara el formulario para la inserción de un usuario nuevo</i>	
2) <i>Realiza las validaciones necesarias sobre cada campo del registro</i>	
<a href="#"><u>Usuario::remember()</u></a>	116
<a href="#"><u>Usuario::reset()</u></a>	116
<a href="#"><u>Usuario::update()</u></a>	117
1) <i>Prepara el formulario para la modificación de un usuario existente</i>	
2) <i>Realiza las validaciones necesarias sobre cada campo del registro</i>	
<a href="#"><u>Usuario::token()</u></a>	117
<a href="#"><u>Usuario::send_mail()</u></a>	117
<a href="#"><u>Usuario::index()</u></a>	115
1) <i>Visualiza los usuarios existentes para su interacción CRUD</i>	
2) <i>En caso de detectar un texto a buscar se filtran los usuarios existentes acorde a la búsqueda</i>	
<a href="#"><u>Usuario::get_token()</u></a>	115
<i>Acción para obtener el token oculto en el login</i>	
<a href="#"><u>Usuario::automatic_access()</u></a>	114
<a href="#"><u>Usuario::\$mas</u></a>	113
<i>Acción para hacer autologin en otro sistema</i>	
<a href="#"><u>Usuario</u></a>	113
<i>Controlador Usuario</i>	
<a href="#"><u>Usuario::cerrar_etab()</u></a>	114
<i>Acción para cerrar el login en otro sistema</i>	
<a href="#"><u>Usuario::delete()</u></a>	114
<i>Solicita la eliminación del usuario recibido</i>	
<a href="#"><u>Usuario::get_galleta()</u></a>	115
<i>Obtiene las cookies que se generan en la session</i>	
<a href="#"><u>Usuario::getActivesByGroup()</u></a>	114
<i>Acción para servir un array de objetos con los usuarios activos por grupo AJAX y devuelve un objeto JSON</i>	
<a href="#"><u>Usuario::form_init()</u></a>	114
<a href="#"><u>Usuario::update_info()</u></a>	118
<a href="#"><u>Usuario::view()</u></a>	118
<i>Visualiza los datos del usuario recibido</i>	
<a href="#"><u>Usuario_model:: getByUsername()</u></a>	239
<i>Obtiene el usuario solicitado</i>	
<a href="#"><u>Usuario_model::getById()</u></a>	238
<i>Obtiene el usuario solicitado, se puede obtener el registro normal o personalizado para visualización (descripciones)</i>	

*en tablas vinculadas)*

<a href="#"><u>Usuario_model::getApellidoPaterno()</u></a>	238
<a href="#"><u>Usuario_model::getApellidoMaterno()</u></a>	238
<a href="#"><u>Usuario_model::getClave()</u></a>	239
<a href="#"><u>Usuario_model::getCorreo()</u></a>	239
<a href="#"><u>Usuario_model::getIdGrupo()</u></a>	240
<a href="#"><u>Usuario_model::getId()</u></a>	240
<a href="#"><u>Usuario_model::getgrupo()</u></a>	239
<a href="#"><u>Usuario_model::getActivo()</u></a>	238
<a href="#"><u>Usuario_model::getActivesByGroup()</u></a>	238
<i>Obtiene los usuarios activos del grupo solicitado</i>	
<a href="#"><u>Usuario_model</u></a>	236
<i>Modelo Usuario</i>	
<a href="#"><u>usuario_model.php</u></a>	139
<a href="#"><u>Usuario:: ifUserExists()</u></a>	118
<i>Callback para validar que un nombre de usuario no se duplique</i>	
<a href="#"><u>Usuario_model::authenticate()</u></a>	236
<i>Valida las credenciales recibidas</i>	
<a href="#"><u>Usuario_model::checkCredentials()</u></a>	236
<i>Verifica que el usuario haya iniciado sesión y además tenga permiso en la acción recibida</i>	
<a href="#"><u>Usuario_model::delete()</u></a>	237
<i>Elimina de la base de datos al usuario (id en propiedades)</i>	
<a href="#"><u>Usuario_model::check_token()</u></a>	237
<a href="#"><u>Usuario_model::check_data()</u></a>	237
<a href="#"><u>usuario.php</u></a>	73