



Facultad de Ingeniería
Escuela de Ingeniería Civil en Informática

DESARROLLO DE UNA HERRAMIENTA CASE PARA DESCRIBIR ARQUITECTURAS ORIENTADAS A SERVICIOS UTILIZANDO I*

Por

Cristian Rivera Pérez

Trabajo realizado para optar al Título de
INGENIERO CIVIL EN INFORMÁTICA

Prof. Guía: Carlos Becerra Castro

Abril 2012

Certifico que he leído este documento y que, en mi opinión, es adecuado en ámbito y calidad como trabajo para optar al título de Ingeniero Civil en Informática.

Carlos Becerra Castro Profesor Guía

Certifico que he leído este documento y que, en mi opinión, es adecuado en ámbito y calidad como trabajo para optar al título de Ingeniero Civil en Informática.

Marta Barría Martínez Profesor Informante

Certifico que he leído este documento y que, en mi opinión, es adecuado en ámbito y calidad como trabajo para optar al título de Ingeniero Civil en Informática.

Roberto Muñoz Soto Profesor Informante

Aprobado por la Escuela de Ingeniería Civil en Informática, UNIVERSIDAD DE VALPARAÍSO.

Resumen

Siempre el horizonte del desarrollo de software ha sido la automatización y sistematización de procesos, que ayudan a realizar tareas de forma más simple, rápida y eficiente. Específicamente en el área de las arquitecturas Orientadas a Servicios (SOA) se busca la sistematización en la generación de arquitecturas candidatas utilizando herramientas que den soporte al proceso. Este trabajo de título apunta a esto, a desarrollar una herramienta CASE para sistematizar el proceso de generación de arquitecturas orientadas a servicios a partir de una especificación de requerimientos en i*. Donde i* es un framework que se utiliza en la orientación a metas, cuya estructura facilita la asociación a arquitecturas SOA. Como principal resultado se obtuvo la herramienta CASE que permite describir requerimientos utilizando i* y así realizar un mapeo a arquitecturas SOA.

Agradecimientos

En estas pocas líneas aprovecho de agradecer a todas las personas que hicieron posible que pudiera finalizar estos seis años de forma exitosa.

En primer lugar agradecer a mis padres y hermano que me ayudaron en cuanto pudieron teniéndome a mí y a mi familia mientras terminaba mis estudios, siempre brindando una mano de apoyo cuando lo necesité y por lo que recuerdo fueron muchas.

También quiero agradecer a Daniel Bravo, Juan López y Marcelo Vega quienes me enseñaron que en la vida tenemos que tomarnos las cosas con humor y verle siempre el lado positivo de las cosas. También le agradezco a estos grandes amigos su amistad y compresión en muchos momentos complicados de mi vida donde siempre estuvieron ahí apoyándome y tirándome para arriba durante todos estos años.

Agradezco de manera especial a Cynthia Manríquez quién ha sido mi compañera y parte fundamental de las decisiones en mi vida y quién siempre me animó a seguir adelante no importando que tan complicadas o malas fueron las etapas que nos tocaron vivir juntos.

Agradezco y a mi hijo quién es la razón de mi existir y mi energía en la vida, quién motiva todo lo que hago y haré durante el resto de mis días.

También quiero agradecer en general a todas las personas que trabajaron a lo largo de mi formación profesional. Como académicos, funcionarios y amigos quienes hicieron posible que las cosas que algún día se vieron imposibles se concreten el día de hoy.

Índice general

Resumen	III
Agradecimientos	IV
1. Introducción	1
2. Marco Teórico	3
2.1. Arquitectura Orientada a Servicio (SOA)	3
2.1.1. Que es SOA en la actualidad	5
2.1.2. Web Services (Servicios Web)	6
2.1.3. WSDL	7
2.1.4. Herramienta para generar Arquitectura SOA	11
2.2. i*(i-star)	14
2.2.1. Elementos del framework i*	15
2.2.2. Tipos de Modelos de i*	18
2.2.3. Herramientas que soportan i*	20
2.3. Trabajos de i* y SOA	23
2.3.1. Representación de Arquitecturas de Software Orientado a servicios	25
3. Definición del problema y Análisis	27
3.1. El problema	27
3.2. Arquitecturas Orientadas a Servicios	27
3.3. Orientación a metas	28
3.4. Solución al Problema	28
3.5. Objetivo General	28
3.6. Objetivos Específicos	28
4. Especificación de Requerimientos	30
4.1. Requerimientos Funcionales	30
4.2. Requerimientos no Funcionales	31
4.3. Modelo Conceptual	32

4.4. Descripción Usuarios	35
4.5. Casos de Uso	35
4.5.1. Acceso al Sistema:	35
4.5.2. Caso de Uso Gestión de Usuarios	37
4.5.3. Selección de Diagramas:	38
4.5.4. Pasar a Diagrama Siguiente o Posterior:	39
4.5.5. Exportar Modelo:	40
4.5.6. Configurar Datos:	41
4.5.7. Caso de Uso General:	41
5. Diseño	43
5.1. Diseño arquitectónico	43
5.1.1. Arquitectura de 3 capas	44
5.2. Modelo Navegacional	51
5.2.1. Secciones	52
5.3. Diseño lógico	53
5.3.1. Diagrama de clases	53
5.3.2. Clase Usuario	54
5.3.3. Clase Extendida Administrador	55
5.3.4. Clase Extendida Usuario registrado	55
5.3.5. Clase Herramienta de Modelado	56
5.3.6. Clase Conector	56
5.3.7. Clase Elemento	57
5.3.8. Clase Patrones	58
5.3.9. Clase Modelo	58
5.3.10. Interface GenerarWSDL	59
5.4. Diseño de datos	59
5.5. Diseño de pruebas	62
5.5.1. Pruebas Unitarias	62
5.5.2. Pruebas de Integración	64
5.5.3. Prueba de Aceptación	66
6. Implementación	69
6.1. Plataforma de Desarrollo	69
6.2. Herramientas de Software	69
6.3. Herramientas de Hardware	70
6.4. Lenguajes de Programación y frameworks	71
6.5. Estrategias de Implementación	73

7. Pruebas	80
7.1. Pruebas Unitarias	80
7.1.1. Análisis de Resultados	82
7.2. Pruebas de Integración	82
7.2.1. Resultados de Pruebas de Integración	83
7.3. Pruebas de Aceptación	83
7.3.1. Encuesta usuarios	84
7.3.2. Encuesta Administrador	85
8. Implementación	87
9. Conclusiones	89
Bibliografía	91
A. Casos de Uso extendidos, Diagramas de secuencia y Diagramas de estados	95
A.1. Caso de Uso extendido de Acceso al Sistema	95
A.1.1. Diagrama de Flujo Acceso al Sistema	97
A.1.2. Diagrama de Estado Acceso al Sistema	98
A.2. Caso de Uso Extendido Crear Usuarios	99
A.2.1. Diagrama de Flujo Crear Usuario	99
A.2.2. Diagrama de Estado Crear Usuario	101
A.3. Caso de Uso extendido Modificar Usuario	102
A.3.1. Diagrama de Flujo Modificar Usuario	103
A.3.2. Diagrama de Estado Modificar Usuario	104
A.4. Caso de Uso extendido Eliminar Usuario	105
A.4.1. Diagrama de Flujo Eliminar Usuario	105
A.4.2. Diagrama de Estado Modificar Usuario	107
A.5. Caso de uso Extendido Selección de diagrama	108
A.5.1. Diagrama de Flujo Selección Diagrama	108
A.5.2. Diagrama de Estado Selección Diagrama	110
A.6. Caso de Uso Extendido Pasar a siguiente/posterior Diagrama	111
A.6.1. Diagrama de Flujo Pasar Siguiente/Posterior	111
A.6.2. Diagrama de Estado Pasar Siguiente/Posterior	112
A.7. Caso de Uso Extendido Exportación	113
A.7.1. Diagrama de Flujo Exportación	113
A.7.2. Diagrama de Estado Exportación	115
A.8. Caso de Uso Extendido Configurar Datos	116
A.8.1. Diagrama de Flujo Configurar Datos	117
A.8.2. Diagrama de Estado Configurar Datos	118

B. Casos de Uso Reales	119
C. Pruebas Unitarias	152
C.1. Modulos a probar	152
C.2. Acceder Al Sistema - Interfaz	155
C.3. Acceder Al Sistema - Lógica	156
C.4. Acceder Al Sistema - Persistencia	157
C.5. Crear Usuario - Interfaz	158
C.6. Crear Usuario - Lógica	159
C.7. Crear Usuario - Persistencia	160
C.8. Modificar Datos Usuario - Interfaz	161
C.9. Modificar Datos Usuario - Lógica	162
C.10. Modificar Datos Usuario - Persistencia	163
C.11. Eliminar Usuario - Interfaz	164
C.12. Eliminar Usuario - Lógica	165
C.13. Eliminar Usuario - Persistencia	166
C.14. Pasar Siguiente/Anterior - Interfaz	167
C.15. Pasar Siguiente/Anterior - Lógica	168
C.16. Exportar Modelo - Interfaz	169
C.17. Exportar Modelo - Lógica	170
C.18. Configurar Datos - Interfaz	171
C.19. Configurar Datos - Lógica	172
D. Pruebas de Integración	173
E. Pruebas de Aceptación	192
E.1. Encuesta Usuario	192
E.2. Encuesta Administrador	198

Índice de tablas

2.1. Tabla comparativa herramientas para modelar SOA	14
2.2. Tabla comparativa de las diferentes herramientas para modelar i*	23
5.1. Modelo Prueba Unitaria	61
5.2. Encuesta Usuario	67
5.3. Encuesta Administrador	68
A.1. Caso de Uso extendido de Acceso al Aistema	96
A.2. Caso de Uso extendido de Crear Usuario	99
A.3. Caso de Uso extendido Modificar Usuario	102
A.4. Caso de uso extendido Eliminar Usuario	105
A.5. Caso de Uso extendido de Selección de diagramas	108
A.6. Caso de Uso extendido para avanzar y retroceder	111
A.7. Caso de Uso extendido de Exportación	113
A.8. Caso de Uso extendido de Configuración de datos	116
B.1. Caso de Uso Real de Acceso al Sistema	120
B.2. Caso de Uso Real de Crear Usuario	125
B.3. Caso de Uso Real Modificar Usuario	129
B.4. Caso de uso Real Eliminar Usuario	133
B.5. Caso de Uso Real de Selección de diagramas	136
B.6. Caso de Uso Real para avanzar y retroceder	143
B.7. Caso de Uso Real de Exportación	145
B.8. Caso de Uso Real de Configuración de datos	148

Índice de figuras

2.1. Primer enfoque organización SOA	4
2.2. Segundo enfoque SOA	5
2.3. Arquitectura SOA	6
2.4. Ejemplo WSDL	8
2.5. Representación Estructura Básica de WSDL	9
2.6. Representación gráfica de los Objetivos o Metas	15
2.7. Representación gráfica de los Objetivos, o Metas, suaves	15
2.8. Representación gráfica de los Recursos	16
2.9. Representación gráfica de las Tareas	16
2.10. Representación gráfica de los Actores	16
2.11. Representación gráfica del Límite del actor	17
2.12. Representación gráfica de las Asociaciones	18
2.13. Representación gráfica de los Enlaces	18
2.14. Representación modelos SD ejemplo en un E-Commerce	19
2.15. Representación modelos SR ejemplo en un E-Commerce	20
2.16. Acercamiento a SOA con i*	25
4.1. Modelo Conceptual	33
4.2. Relación de usuarios y sistema	35
4.3. Acceso del usuario al sistema	36
4.4. Acceso del Administrador al sistema	37
4.5. Gestión de Usuarios	38
4.6. Selección de un modelo por el usuario	39
4.7. Caso de Uso para Avanzar y Retroceder en los diagramas	40
4.8. Caso de Uso para Exportar Modelo	40
4.9. Caso de Uso para Configurar Datos	41
4.10. Casos de Uso para Configurar Datos	42
5.1. Primer Enfoque Arquitectura de tres capas	44
5.2. Subsistema de Login	45
5.3. Subsistema de Perfil	46

5.4.	Subsistema de Perfil	47
5.5.	Capa de presentación GOSOA	47
5.6.	Subsistema de Autentificación	48
5.7.	Subsistema Lógica de perfil	49
5.8.	Capa Lógica	49
5.9.	Capa de persistencia	50
5.10.	Capa de persistencia	51
5.11.	Modelo navegacional GOSOA	52
5.12.	Diagrama de Clases	54
5.13.	Diagrama relacional de GOSOA	59
5.14.	Modelo Prueba Unitaria	63
5.15.	Diagrama de pruebas de integración	64
5.16.	Formato pruebas de integración	65
5.17.	Instrucciones para la encuesta	66
6.1.	Diagrama de Integración de la implementación	73
6.2.	Imagen accediendo al sistema	74
6.3.	Imagen creando usuario	75
6.4.	Imagen de modificación de usuarios	76
6.5.	Imagen de eliminación de usuarios	77
6.6.	Imagen de Selección de diagramas	78
6.7.	Imagen de exportación del modelo	79
6.8.	Imagen de Configuración de datos	79
7.1.	Resumen de las pruebas unitarias	82
7.2.	Resultados de Integración	83
7.3.	Resumen de Preguntasde los usuarios	84
7.4.	Resumen de Preguntas de La sección Administrador	85
A.1.	Diagrama de secuencia acceso del usuario al sistema	97
A.2.	Diagrama de Estado acceso del usuario al sistema	98
A.3.	Diagrama de Secuencia Crear Usuario	100
A.4.	Diagrama de Estado Crear Usuario	101
A.5.	Diagrama de secuencia Modificar Usuario	103
A.6.	Diagrama de Estado Modificar Usuario	104
A.7.	Diagrama de Secuencia Eliminar Usuario	106
A.8.	Diagrama de Estado Eliminar Usuario	107
A.9.	Diagrama de Secuencia selección de un modelo por el usuario	109
A.10.	Diagrama de Estado selección de un modelo por el usuario	110
A.11.	Diagrama de Secuencia para Avanzar y Retroceder en los diagramas	112
A.12.	Diagrama de Estado para Avanzar y Retroceder en los diagramas	112

A.13. Diagrama de Secuencia para Exportar Modelo	114
A.14. Diagrama de Estado para Exportar Modelo	115
A.15. Diagrama de Secuencia para Configurar Datos	117
A.16. Diagrama de Estado para Configurar Datos	118
 B.1. Interfaz de Login	121
B.2. Interfaz Perfil de Usuario	122
B.3. Interfaz Perfil de administrador	123
B.4. Interfaz de Login	124
B.5. Interfaz Perfil de administrador	126
B.6. Interfaz de creación de usuarios	127
B.7. Interfaz de creación de usuarios	128
B.8. Interfaz Perfil de administrador	130
B.9. Interfaz Modificación datos de usuario	131
B.10. Interfaz de problema conexión al buscar usuario	132
B.11. Interfaz de eliminación de usuarios	134
B.12. Interfaz de problema conexión al buscar usuario	135
B.13. Interfaz de selección de diagramas	137
B.14. Interfaz de Diagrama Global Model	138
B.15. Interfaz de Diagrama Process Model	139
B.16. Interfaz de Diagrama Protocol Model	140
B.17. Interfaz de Diagrama Service Desing Pattern View Model	141
B.18. Interfaz de Diagrama Service Desing Pattern View Model	142
B.19. Interfaz de para cambiar de Modelo	144
B.20. Interfaz Para empezar la exportación	146
B.21. Interfaz de imagen de exportación	147
B.22. Interfaz volver al diagrama	149
B.23. Interfaz de opciones de elemento	150
B.24. Interfaz de opciones de elemento avanzadas	151
 C.1. Pruebas Unitarias a nivel de interfaz	152
C.2. Pruebas Unitarias a nivel de Lógica	153
C.3. Pruebas Unitarias a nivel de Persistencia	153
C.4. Pruebas Unitarias del modelador a nivel de Interfaz	154
C.5. Pruebas Unitarias del modelador a nivel de Lógica	154
C.6. Acceder al Sistema - Interfaz	155
C.7. Acceder al Sistema - Lógica	156
C.8. Acceder al Sistema - Persistencia	157
C.9. Crear Usuario - Interfaz	158
C.10. Crear Usuario - Lógica	159
C.11. Crear Usuario - Persistencia	160

C.12. Modificar Datos Usuario - Interfaz	161
C.13. Modificar Datos Usuario - Lógica	162
C.14. Modificar Datos Usuario - Persistencia	163
C.15. Eliminar Usuario - Interfaz	164
C.16. Eliminar Usuario - Lógica	165
C.17. Eliminar Usuario - Persistencia	166
C.18. Pasar Siguiente/Anterior - Interfaz	167
C.19. Pasar Siguiente/Anterior - Lógica	168
C.20. Exportar Modelo - Interfaz	169
C.21. Exportar Modelo - Lógica	170
C.22. Configurar Datos - Interfaz	171
C.23. Configurar Datos - Lógica	172
D.1. Prueba Integración 1	174
D.2. Prueba Integración 2	175
D.3. Prueba Integración 3	176
D.4. Prueba Integración 4	177
D.5. Prueba Integración 5	178
D.6. Prueba Integración 6	179
D.7. Prueba Integración 7	180
D.8. Prueba Integración 8	181
D.9. Prueba Integración 9	182
D.10. Prueba Integración 10	183
D.11. Prueba Integración 11	184
D.12. Prueba Integración 12	185
D.13. Prueba Integración 13	186
D.14. Prueba Integración 14	187
D.15. Prueba Integración 15	188
D.16. Prueba Integración 16	189
D.17. Prueba Integración 17	190
D.18. Prueba Integración 18	191
E.1. Pregunta 1	192
E.2. Pregunta 2	193
E.3. Pregunta 3	193
E.4. Pregunta 4	194
E.5. Pregunta 5	194
E.6. Pregunta 6	195
E.7. Pregunta 7	195
E.8. Pregunta 8	196
E.9. Pregunta 9	196

E.10. Pregunta 10	197
E.11. Pregunta 11	197
E.12. Resumen de Preguntas de los usuarios	198
E.13. Pregunta 1 Administrador	199
E.14. Pregunta 2 Administrador	199
E.15. Pregunta 3 Administrador	200
E.16. Pregunta 4 Administrador	200
E.17. Pregunta 5 Administrador	201
E.18. Pregunta 6 Administrador	201
E.19. Pregunta 7 Administrador	202
E.20. Pregunta 8 Administrador	202
E.21. Pregunta 9 Administrador	203
E.22. Pregunta 10 Administrador	203
E.23. Pregunta 11 Administrador	204
E.24. Resumen de Preguntas de La sección Administrador	204

Capítulo 1

Introducción

En el área de SOA (Arquitecturas Orientadas a Servicios) se busca la sistematización en la generación de arquitecturas candidatas utilizando herramientas que den soporte al proceso.

Existen herramientas que permiten sistemáticamente describir un sistema desde los requerimientos al diseño de la arquitectura¹, pero no una que permita crear arquitecturas orientadas a servicios. En la construcción de este proceso sistemático se propone utilizar la orientación a metas, que se enfoca en describir los objetivos que tiene un actor, asociado al sistema bajo desarrollo, y las tareas que debe hacer éste para cumplir esos objetivos, descrito por medio de lenguaje natural. Esta manera de representar o modelar un sistema es muy amplia, ya que permite incorporar factores que al realizar modelos en forma convencional no tomamos en cuenta, como por ejemplo: actores junto a metas; tareas que se deben hacer para realizar estas metas; y los recursos necesarios. Además la orientación a metas tiene un framework llamado i*, que entrega reglas y normas estándares para crear representaciones utilizando este paradigma, que permite fácilmente mapear a arquitecturas SOA.

En general las arquitecturas orientadas a servicios utilizan un conjunto de servicios distribuidos (por ejemplo servicios disponibles en la web), lo que permite realizar cambios a bajo costo o reemplazarlos si es necesario, lo que no se puede realizar con arquitecturas tradicionales. Tampoco es necesario preocuparse por la tecnología fundamental, ya que lo importante son sus interfaces que están diseñadas en un lenguaje web y XML que son universales [1].

Es por esto que se desarrollará una herramienta CASE que de soporte al proceso de generación de arquitecturas SOA utilizando el framework i*, según la metodología desarrollada

¹por ejemplo: JFrameBuilder, DreamWeaver, VisualWebDeveloper, VisualBasic.

por Franch [2]. Que describe 5 fases de especificación de requerimientos hasta arquitecturas SOA a nivel de componentes. Las cuales son: Global model, Process Model, Protocol Model, Service Design Pattern View y Service Composition Design Pattern View.

El documento se estructura de la siguiente forma:

En el capítulo 2 se genera una descripción del marco referencial tomando en cuenta temas como: Arquitecturas Orientadas a Servicios (SOA), i estrella (i^*) y trabajos relacionados entre i^* y SOA. En el capítulo 3 se realiza una definición del problema y la propuesta de una solución a éste. En el capítulo 4 se generan las especificaciones de requerimientos. Luego el diseño en el capítulo 5 , la implementación de presenta en el capítulo 6 donde se especifica la plataforma de desarrollo, herramientas de software que se utilizarón, herramientas de hardware, los lenguajes de programación junto con los frameworks y las estrategias de implementación usadas. Las pruebas del sistema se encuentran en el capítulo 7 donde encontramos pruebas unitarias, de integración y de aceptación. En el capítulo 8 de Implantación se explica en detalle como se implantó el sistema. Finalmente en el capítulo 9 se encuentran las conclusiones donde se presentan los resultados y conclusiones obtenidas del desarrollo del sistema.

Capítulo 2

Marco Teórico

Para contextualizar este trabajo de título se presentan las Arquitecturas Orientadas a Servicios, así como el framework i* y trabajos relacionados entre ambos.

Además existen comparaciones entre herramientas específicas de cada dominio.

2.1. Arquitectura Orientada a Servicio (SOA).

SOA es un conjunto de los principales diseños utilizados durante la fase de desarrollo y de integración en los sistemas de información. Un sistema basado en SOA tendrá un paquete de funcionalidades como un conjunto de servicios interoperables, que pueden ser utilizados en múltiples sistemas independientes del dominio del negocio.

También podemos decir que proporciona un camino para los consumidores de servicios, como las aplicaciones web, para estar al tanto de los servicios disponibles basados en SOA. Por ejemplo, varios departamentos diferentes dentro de una empresa pueden implementar SOA en diferentes lenguajes de programación. Sus clientes se beneficiarán de una bien definida interfaz para acceder a estos servicios. Generalmente se utiliza XML para definir la interfaz de los servicios SOA, aunque no es necesario[3].

En la Figura 2.1 se muestra como SOA es el intermediario entre los servicios que entrega un proveedor y los servicios que son necesitados por un consumidor.

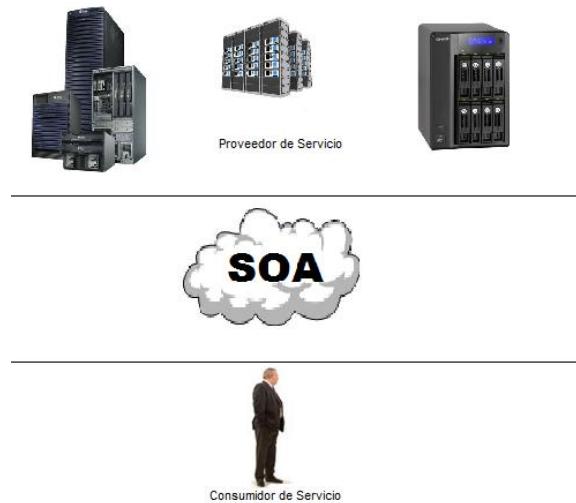


Figura 2.1: Primer enfoque organización SOA

SOA define como integrar aplicaciones diferentes para un ambiente en general web con múltiples plataformas de ejecución. En vez de definir una API¹ SOA define una interfaz en término de protocolos y funcionalidades. Un punto final de una aplicación es el punto de entrada para una implementación SOA, es decir, que SOA se implementa al final de una aplicación y sirve como intermediarios con otras.

Para generar una arquitectura SOA se requiere un bajo nivel de acoplamiento de los servicios con los sistemas operativos y otras tecnologías que son base de aplicaciones. Así SOA separa funciones en distintas unidades o servicios que los desarrolladores hacen accesible a través de la web con el fin de combinarlos y rehusarlos en la producción de nuevas aplicaciones. Estos servicios y sus consumidores se comunican entre sí pasando información bien definida, en formato en común, o coordinando una actividad entre dos o más servicios[4].

En la Figura 2.2 se encuentra representado SOA como ente que hace posible la comunicación entre el consumidor y el proveedor de servicios.

¹(Application Programming Interface). Grupo de rutinas que provee un sistema operativo, una aplicación o una biblioteca, que definen cómo invocar desde un programa un servicio que éstos prestan. En otras palabras, una API representa un interfaz de comunicación entre componentes software.



Figura 2.2: Segundo enfoque SOA

2.1.1. Que es SOA en la actualidad

Las arquitecturas SOA se basan en una malla de servicios de software. Estos servicios son unidades débilmente acopladas de funcionalidades que no tienen llamadas entre sí incorporadas en ellas. Cada servicio implementa una acción: cómo llenar una solicitud en línea para una cuenta, ver el resumen de una cuenta en línea, realizar una reserva en línea, o para reservar un boleto de avión. En lugar de integrar los servicios realizando llamadas internas en el código, ellos usan protocolos ya definidos para describir como estos servicios pasan o parcelan los mensajes usando metadata² para su descripción.

Los desarrolladores de SOA asocian objetos individuales de SOA usando la orquestación³. En este proceso de orquestación los desarrolladores asocian funcionalidades de software (los servicios). Para la eficiencia de SOA, la arquitectura debe cumplir los siguientes requerimientos:

- **Interoperabilidad:** la relación entre diferentes sistemas y lenguajes de programación proporciona la base para la integración entre aplicaciones en diferentes plataformas a través de un protocolo de comunicación.

²Datos estructurados y codificados que describen características de instancias conteniendo informaciones para ayudar a identificar, descubrir, valorar y administrar las instancias descritas.

³Orquestación describe la disposición automatizada, la coordinación y gestión de sistemas informáticos complejos, middleware y servicios.

- **Crear un conjunto de recursos accesibles:** se debe crear un conjunto de recursos que nos permitan desarrollar nuevas funcionalidades en base de los recursos que se encuentren disponibles.

Es así como la Figura 2.3 muestra al usuario, quien encuentra y luego utiliza el servicio (en el directorio de servicios y el proveedor de servicios respectivamente); al proveedor de servicios quien publica los servicios en un directorio de servicios.

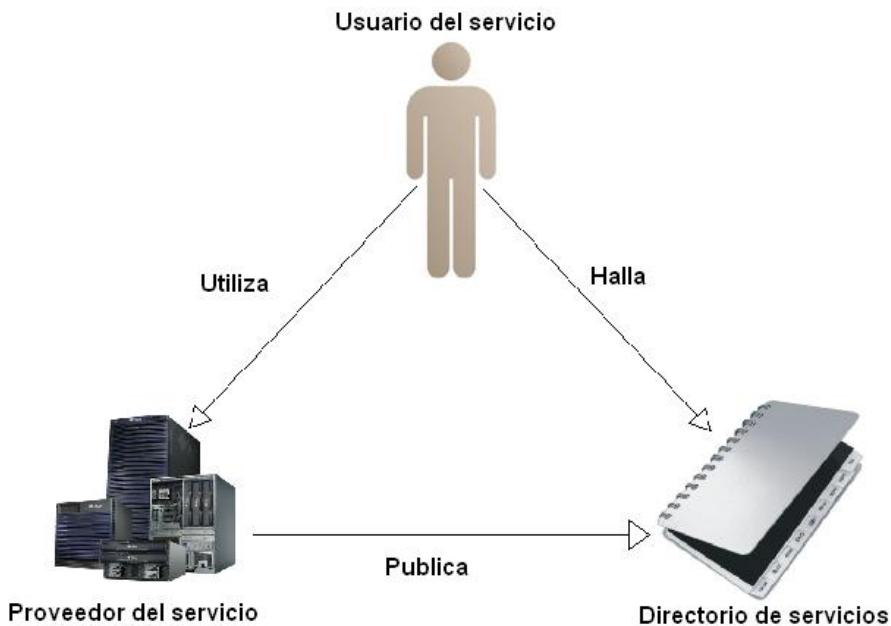


Figura 2.3: Arquitectura SOA

2.1.2. Web Services (Servicios Web)

Los web services pueden ser implementados en las arquitecturas orientadas a servicios (SOA). Los Web Services crean bloques funcionales accesibles a través de protocolos estándares de internet independientes de la plataforma y del lenguaje de programación. Estos servicios pueden ser nuevas aplicaciones o simplemente puede representar el envoltorio alrededor de sistemas existentes para ser accedidos a través de la red. Existen 2 grandes organizaciones que se encargan de la arquitectura y reglamentación de los web services, que son OASIS [5] y W3C [6].

Así podemos decir que un servicio web es un conjunto de protocolos o estándares que sirven para intercambiar datos entre las aplicaciones. Diferentes aplicaciones desarrolladas

en distintos lenguajes de programación y ejecutadas sobre distintas plataformas pueden intercambiar datos a través de los Web Services [7].

2.1.3. WSDL

WSDL son las siglas de Web Service Description Language, que es una estructura en formato XML⁴ utilizado para describir web services. Actualmente se utiliza la versión 2.0 instaurada por la W3C [8].

Así los WSDL definen la interfaz pública de los Web Services. Describe, en formato XML, los requisitos del protocolo y el formato de los mensajes para interactuar con los servicios disponibles en su catálogo.

De esta manera un programa que necesita usar un servicio puede leer el WSDL para ver el conjunto de servicios que se encuentran disponibles en formato XML Schema⁵, luego el cliente puede hacer una llamada al servicio específico a través de SOAP en la lista de servicios disponibles. pero pueden ser utilizadas otras formas de unión. Estas otras formas podrían ser CORBA⁶, Internet Inter-ORB Protocol (IIOP)⁷, .NET⁸.

Podemos definir una estructura de datos de los WSDL para la Interfaz de un servicio:

- **Tipos de datos:** *<types>* este campo hace referencia al tipo de dato que se utiliza en el mensaje.
- **Mensaje:** *<message>* que van los elementos del mensaje.

⁴Extensible Markup Language (XML) es un conjunto de reglas para la codificación de documentos de una forma leible por la máquina.

⁵XML Schema es un lenguaje de esquema utilizado para describir la estructura y las restricciones de los contenidos de los documentos XML de una forma muy precisa, más allá de las normas sintácticas impuestas por el propio lenguaje XML.

⁶En computación, CORBA (Common Object Request Broker Architecture - arquitectura común de intermediarios en peticiones a objetos). es un estándar que establece una plataforma de desarrollo de sistemas distribuidos facilitando la invocación de métodos remotos bajo un paradigma orientado a objetos.

⁷En computación distribuida, el General Inter-ORB Protocol (GIOP) es el protocolo abstracto por el cual los object request brokers (ORB) se comunican. Normas asociadas con el protocolo son mantenidos por el Object Management Group (OMG).

⁸.NET es un framework de Microsoft que hace un énfasis en la transparencia de redes, con independencia de plataforma de hardware y que permita un rápido desarrollo de aplicaciones. Basado en ella.

- **Tipos de puertos:** `<portType>` aquí definimos las operaciones que nos son permitidas y los mensajes intercambiados en los servicios.
- **Bindings:** `<binding>` se especifican los protocolos de información que se usarán.
- **Servicios:** `<service>` es el conjunto de puertos y direcciones de estos.

En la Figura 2.4 se muestra un ejemplo simple de un WSDL.

```

<?xml version="1.0" encoding="UTF-8">
<definition> es el elemento raíz de un WSDL
    <type> indica que tipo de datos serán transmitidos</type>
    <message> indica que mensaje será transmitido</message>
    <portType> indica que operaciones o funciones se soportan</portType>
    <binding> indica:
        ... como se transmitirán los mensajes por la red
    </binding>
    <service> indica donde está localizado el servicio</service>
</definition>

```

Figura 2.4: Ejemplo WSDL

Existen cuatro tipos de operaciones al momento de querer implementar el mensaje en un Web Services, estas son:

1. Sólo ida: Los mensajes son enviados, pero no es necesaria una respuesta.
2. Petición/Respuesta: El emisor envía un mensaje y espera la respuesta.
3. Solicitar respuesta: El emisor envía un mensaje y solicita que le respondan.
4. Notificación: Los mensajes son enviados a múltiples receptores, y los que reciben envían una notificación.

La Figura 2.5 muestra la estructura básica de un WSDL y se explica como para cada enlace de servicio pueden existir varias operaciones y para cada operación debe existir una definición del tipo de dato y una definición del mensaje.

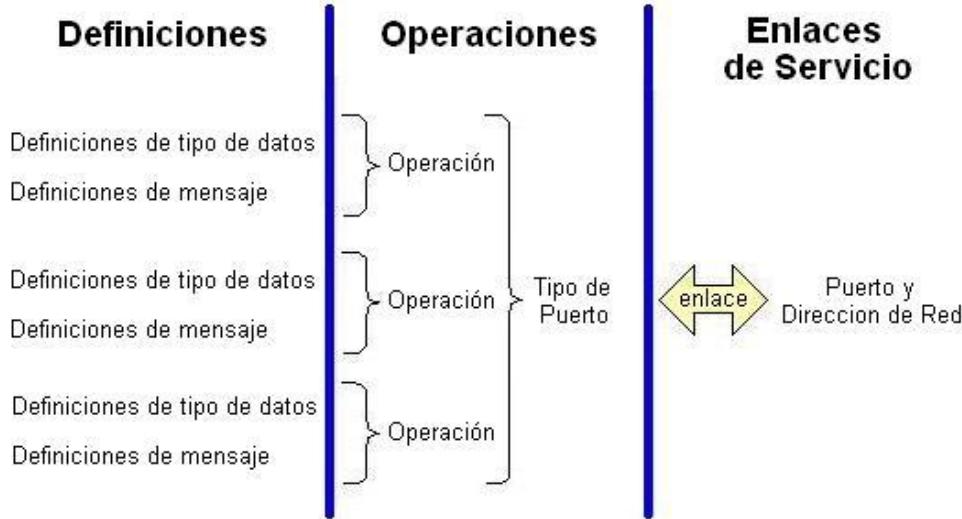


Figura 2.5: Representación Estructura Básica de WSDL

Tipos de WSDL

- Extensible Markup Language (XML)**: Es un conjunto de normas de codificación de documentos en formato electrónico. El diseño de los objetivos de XML hace hincapié en la simplicidad, generalidad y usabilidad en Internet. Se trata de un formato de datos de texto con un fuerte apoyo a través de Unicode para los idiomas del mundo. Aunque el diseño de XML se centra en los documentos, es ampliamente utilizado para la representación de las estructuras de datos arbitrarios, por ejemplo, en Web Services [9].
- Web Services Dynamic Discovery**: Define un protocolo multicast para localizar servicios. De forma predeterminada, las sondas se envían a un grupo multicast, y los servicios de destino que coinciden, devuelven una respuesta directa al solicitante. Para escalar a un gran número de puntos finales, el protocolo define el comportamiento de la supresión de multidifusión si un proxy está disponible en la red. Para reducir al mínimo la necesidad de votación, los servicios de destino que desea ser descubierta enviar un aviso cuando entran y salen de la red [10].
- Web Services Endpoint Language**: Es un formato XML para el descripción de las características no operativas de servicios extremos, como la calidad del servicio, el costo o la seguridad entre otras propiedades [11].
- Web Services Policy Framework**: Proporciona un modelo general del propósito y

la sintaxis correspondiente para describir y comunicar las políticas de un Web Services. WS-Policy define un conjunto básico de construcciones que se pueden utilizar y ampliar por otros Web Services para describir una amplia gama de requisitos de servicios, preferencias y capacidades [12].

- **Web Services Policy Assertions:** especifica un conjunto de políticas, mensajes y afirmaciones que se pueden especificar dentro de una política.
 - **Web Services Policy Attachment:** especifica tres mecanismos específicos de utilización de las expresiones con las políticas de XML. En concreto, se definen como expresiones que asocian la política con definiciones WSDL y entidades UDDI.
5. **Web Services Metadata Exchange:** Define tres pares de mensajes de petición-respuesta para recuperar tres tipos de metadata: uno recupera la WS-Policy asociadas con el extremo receptor o con un namespace de destino determinado, otro recupera el WSDL asociado a el receptor o con un namespace dado, y el tercero recupera un XML-schema con un namespace determinado. En conjunto, estos mensajes permiten la recuperación gradual de la metadata de un Web Services. También definen la forma de asociar la política específica de la implementación de la totalidad o parte de un portType WSDL cuando se expone a partir de una específica implementación [13].
 6. **Web Services Addressing:** Esta especificación provee de un mecanismo por el cual se pueden identificar Web Services y mensajes de Web Services independientemente del protocolo de transporte utilizado. WS-Addressing define un espacio de nombres que se utiliza para identificar Web Services. Gracias a esta especificación es posible hacer que las peticiones a Web Services puedan ser transmitidas a través de redes compuestas por nodos que realicen algún procesamiento sobre el mensaje, como firewalls, gateways entre otras. De manera que la información de transporte sea independiente del protocolo utilizado para la transmisión del mensaje [14].
 7. **Simple Object Access Protocol (SOAP):** Protocolo de especificación para el intercambio de información estructurada en la aplicación de Web Services en redes informáticas . Se basa en Lenguaje de marcado extensible (XML) para su formato de mensaje, y por lo general cuenta con otra de capa de aplicación protocolos, en particular llamada a procedimiento remoto (RPC) y de transferencia de hipertexto (HTTP), para la negociación y la transmisión de mensajes. SOAP puede formar la capa base de un protocolo de Web Services, proporcionando un marco de mensajería básico sobre los Web Services [15].

2.1.4. Herramienta para generar Arquitectura SOA

En la actualidad no existen herramientas que modelen específicamente arquitecturas orientadas a servicios y menos obtener los WSDL a partir de un modelo de arquitectura SOA.

Los mejores acercamientos que se pueden obtener en este momento es el modelado de esta arquitectura a través de herramientas de modelado para UML, que aunque son específicas del dominio del problema nos dan la posibilidad realizar estas abstracciones.

- **Sybase PowerDesigner:** Según su sitio oficial [16] Sybase PowerDesigner es una herramienta estándar de modelamiento para administración de metadata a nivel empresarial. Ofrece innovaciones en Modelamiento de Procesos Empresariales, incluyendo soporte de simulación y procesamiento ejecutable de procesos de negocio.

Permite a las empresas, visualizar, analizar y manipular metadata, logrando una efectiva arquitectura empresarial de información.

Para Arquitecturas Empresariales brinda un enfoque basado en modelos, el cual permite alinear al negocio con la tecnología de información, facilitando la implementación de arquitecturas efectivas de información empresarial. Brinda potentes técnicas de análisis, diseño y gestión de metadata.

PowerDesigner ofrece análisis de impacto de gran alcance, la gestión del diseño en tiempo de cambio y las técnicas de gestión de metadatos para su empresa.

La gestión de metadata ayuda a que las áreas de negocio estén alineadas y por lo tanto sean coherentes, además de permitir a las empresas tener capacidad de respuesta frente a cambios. Como resultado de estas dos cuestiones se obtiene agilidad empresarial, lo que mejorará a nivel operacional y estratégico el funcionamiento y rendimiento de cualquier entidad.

Ofrece a arquitectos, ejecutivos y profesionales la capacidad de alinear sus procesos con el área de las TI permitiendo a las organizaciones navegar por la metadata y relacionar los procesos de negocio, los modelos de datos y los requerimientos, de forma que si hay algún cambio en los procesos de negocio las empresas puedan conocer la repercusión de éste en los modelos de datos y así sucesivamente.

- **Rational Software Architect:** Según su sitio oficial [17] Rational Software Architect es un software que proporciona un soporte de desarrollo y diseño integrado para el desarrollo dirigido por el modelo con UML.

Rational Software Architect es una herramienta de diseño y desarrollo integrados que potencia el desarrollo orientado al modelado con UML para la creación de aplicaciones y servicios con buena arquitectura.

Con Rational Software Architect se puede:

- Unificar todos los aspectos del diseño y desarrollo de software.
 - Desarrollar aplicaciones de manera más productiva.
 - Sacar provecho de la tecnología más avanzada en lenguaje de modelado.
 - Revisar y controlar la estructura de las aplicaciones Java.
 - Potenciar con una plataforma de modelado abierta y extensible.
 - Simplificar con una solución para diseño y desarrollo en una herramienta.
 - Integrar otras facetas del ciclo de vida.
- **Registry and Repository:** Según su sitio oficial [18] Registry and Repository es una Herramienta que se utiliza en el mercado para aumentar la visibilidad y control de una arquitectura SOA para poder conocerla y manejarla mejor.

Profundiza en los servicios de SOA, en sus políticas de consumo, y en la metadata asociada a la empresa. Este registro a nivel empresarial proporciona capacidades automatizadas para ayudar a las organizaciones optimizar la productividad y los recursos en un entorno SOA.

Asegura el conocimiento de aplicaciones, servicios y sus consumidores con metadata asociada y documentos en toda la organización.

Ayuda a implementar las prácticas recomendadas y aumenta la agilidad del tiempo de ejecución permitiendo garantizar:

- La aplicación coherente de las políticas operativas.
 - Ejecución de las políticas de ciclo de vida del gobierno.
- **Rational RequisitePro:** según su sitio oficial [19] Rational RequisitePro ayuda a los equipos de proyecto a manejar sus requerimientos, para diseñar buenos casos de usos, para mejorar la trazabilidad, para fortalecer la colaboración, para reducir el rework⁹ del proyecto, y para aumentar la calidad.
 - **WebSphere Business Modeler:** según su sitio oficial [20] es la principal herramienta de modelado y análisis de IBM enfocada en los negocios del usuario.

Ofrece modelado de procesos, simulación y capacidades de análisis para ayudar a los usuarios de negocios a entender, documentar e implementar procesos de negocio para la mejora continua de éstos.

A continuación presentamos una tabla 2.1 que se estructura de la siguiente forma:

- Existen 5 columnas y cada una de ellas representa a una herramienta de las mencionadas anteriormente.
- Existen 12 preguntas concretas que pueden ser contestadas con un si o un no. Si la respuesta es negativa se pone una cruz. Si es afirmativa, un victo bueno.
- Las preguntas están enfocadas a funcionalidades que la herramienta puede o no tener.

⁹rework: significa reorganizar o hacer cambios a una idea, un escrito o un problema entre otros, con el propósito de mejorarlo o actualizarlo.

		Sybase Power Designer	Rational Software Architect	Registry and Repository	Rational RequisitePro	WebSphere Business Modeler
1	¿La herramienta permite navegar por la metadata?	✓	✓	✓	✓	✓
2	¿La herramienta soporta técnicas de modelamiento de procesos empresariales?	✓	✓	✓	✓	✓
3	¿La herramienta cuenta con soporte para simulación?	✓	✗	✗	✗	✓
4	¿La herramienta permite procesamiento ejecutable de procesos de negocio?	✓	✗	✗	✗	✗
5	¿La herramienta permite crear modelos basados en SOA?	✓	✓	✓	✓	✓
6	¿La herramienta permite modelar requerimientos?	✓	✓	✗	✓	✓
7	¿La herramienta permite mapear requerimientos a una arquitectura de software?	✓	✓	✗	✓	✓
8	¿La herramienta soporta el framework i*?	✗	✗	✗	✗	✗
9	¿La herramienta permite crear modelos SR?	✗	✗	✗	✗	✗
10	¿La herramienta permite crear modelos SD?	✗	✗	✗	✗	✗
11	¿La herramienta permite mapear requerimientos a SOA?	✗	✗	✗	✗	✗
12	¿La herramienta permite mapear i* a SOA?	✗	✗	✗	✗	✗

Tabla 2.1: Tabla comparativa herramientas para modelar SOA

En conclusión podemos determinar que las herramientas, en general, permiten generar modelos empresariales de SOA, pero ninguna de ellas puede seguir algún tipo de metodología para generar este modelo a partir de requerimientos. Estas Conclusiones se basan en el análisis de la Tabla 2.1.

2.2. i*(i-star)

i* [21] es un lenguaje de modelado adecuado para fases tempranas de representación de sistemas con el fin de comprender el dominio del problema. El lenguaje de modelado i* permite modelar ambas situaciones: el “como son” y el “como debe” ser las cosas.

El nombre i* hace alusión a la “internacionalidad distribuida”, la cual recalca este framework. Es un enfoque original desarrollado para modelar y razonar sobre el ambiente de las organizaciones y sus sistemas de información compuesto de actores heterogéneos con necesidades diferentes. Este lenguaje involucra ambos tipos de orientación: la orientación al actor y la orientación a metas. Así, i* da respuesta al cómo y al por qué y no al qué.

2.2.1. Elementos del framework i*

El modelo describe dependencias entre los actores. Hay 4 elementos para describir estas dependencias:

- Los objetivos o metas
- Los objetivos, o metas, suaves
- Las tareas, y
- Los recursos

Detallando cada elemento tenemos:

- Objetivos: También se conocido como metas. son lo que los actores desean lograr. Su representación es lo que se ve en la Figura 2.6.



Figura 2.6: Representación gráfica de los Objetivos o Metas

- Objetivos, o metas, suaves: Son objetivos que se desean alcanzar pero no necesariamente se alcanzan. Su representación es lo que se ve en la Figura 2.7.



Figura 2.7: Representación gráfica de los Objetivos, o Metas, suaves

- Recursos: Son los recursos que los actores necesitan para cumplir sus objetivos. Su representación es lo que se ve en la Figura 2.8.



Figura 2.8: Representación gráfica de los Recursos

- Tareas: Las tareas son las acciones que necesitan realizar los actores para llegar a cumplir un objetivo. Su representación es lo que se ve en la Figura 2.9.



Figura 2.9: Representación gráfica de las Tareas

También existen los actores que son los que realizan estas acciones. Se pueden diferenciar distintos tipos de actores, pero generalmente solo se usa el término de actor. Los cuales se representan como se muestra en la Figura 2.10.

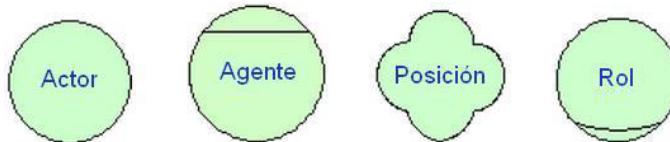


Figura 2.10: Representación gráfica de los Actores

El concepto central es el de actor. Las organizaciones de actores son vistos como poseedores de propiedades intencionales como objetivos, creencias, habilidades, etc. Los actores dependen de cada otro para los objetivos que deben alcanzarse, tareas son desarrolladas y recursos son suministrados. Al depender de otro, un actor puede alcanzar objetivos que son difíciles o imposibles de alcanzar por sí mismos. Por otro lado un actor se vuelve vulnerable si depende de actores que no entregan cosas. Entonces los actores son puntos estratégicos, ya que están preocupados de las oportunidades y las vulnerabilidades, y busca el reordenamiento de su entorno que mejor sirva a sus intereses para la estructuración de estas relaciones intencionales.

Además existen límites de actores indican las fronteras intencionales de un agente en particular. Todos los elementos dentro de un límite para un actor son explícitamente deseados por el actor. Para lograr esto, muchas veces el actor debe depender de las intenciones de otros actores, representado esta dependencia con vínculos a través de fronteras actor. A su vez, un actor puede depender de algunos elementos, representados por un vínculo de dependencia en dirección contraria. A continuación se muestra un ejemplo en la Figura 2.11.

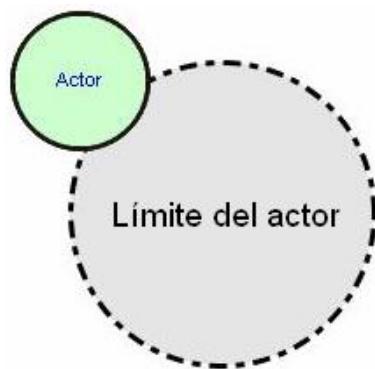


Figura 2.11: Representación gráfica del Límite del actor

También podemos crear relaciones entre estos elementos que se han descrito. Estas representaciones las podemos dividir en dos grupos:

- Las Asociaciones.
- Los Enlaces.

Así podemos explicar cada uno de ellos por separado a continuación.

- **Asociaciones:** Las relaciones entre actores son descritas por vínculos de asociación gráficos entre estos actores. Estas representaciones se describen en la Figura 2.12.

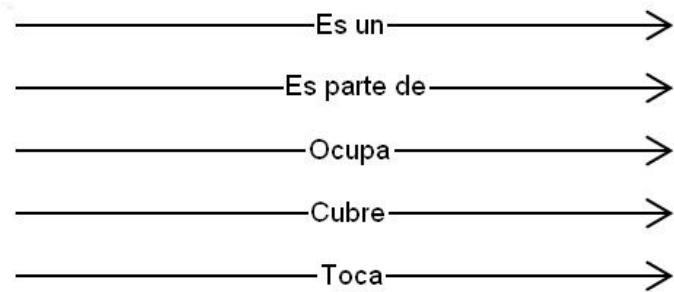


Figura 2.12: Representación gráfica de las Asociaciones

- **Enlaces:** Estos enlaces indican una relación entre un extremo y un medio para alcanzarlo. Los “medios” se expresa en forma de una tarea, puesto que la noción de la tarea encarna cómo hacer algo, con el “fín” se expresa como una meta. En la notación gráfica, los puntos de la punta de flecha de los medios hasta el final. Estas representaciones se describen en la Figura 2.13.

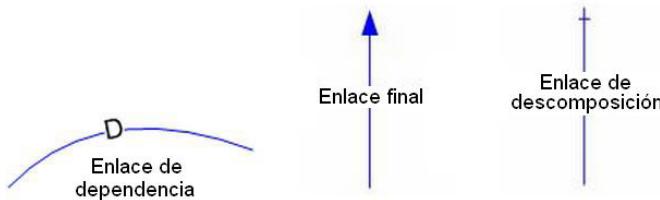


Figura 2.13: Representación gráfica de los Enlaces

2.2.2. Tipos de Modelos de i*

El framework i* consiste en dos principales componentes de modelado:

1. Strategic Dependency Model (SD)

Un modelo SD describe una red de relaciones de dependencia entre varios actores en un contexto organizacional. El actor es usualmente identificado en el contexto del modelo. Este modelo muestra como es un actor y como depende del trabajo de otros actores [22].

Un modelo SD consiste de un conjunto de nodos y vínculos conectando a los actores. Los nodos representan actores y cada vínculo representa una dependencia entre dos actores. El actor que está dependiendo de otro se llama “Depender” y el actor del que se dependía se llama “Dependee” [22].

La Figura 2.14 muestra un ejemplo pequeño de un modelo SD.

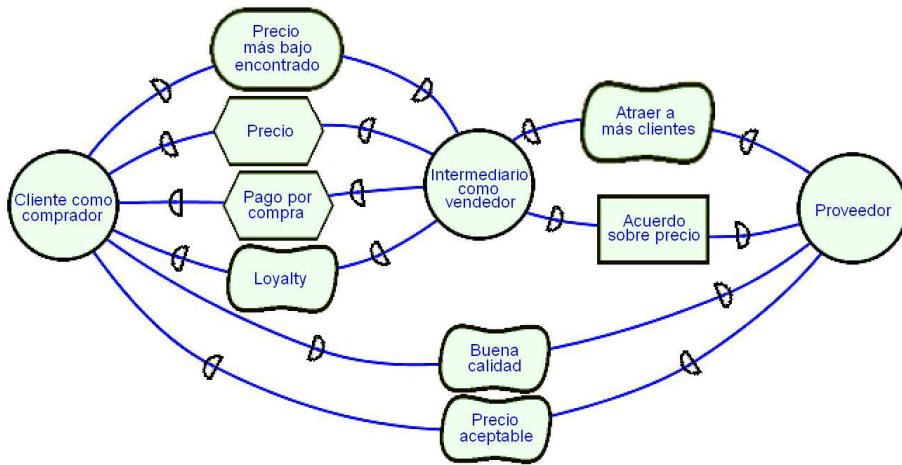


Figura 2.14: Representación modelos SD ejemplo en un E-Commerce

2. Strategic Rational Model (SR)

Un modelo SR permite el modelado de las razones asociadas a cada actor y sus dependencias, y entrega información de cómo los actores logran sus objetivos y objetivos suaves. Este modelo solo incluye elementos considerados como suficientemente importantes para afectar los resultados de un objetivo.

El modelo SR muestra las dependencias de los actores mediante la inclusión del modelo SD. El modelo SR especifica objetivos, objetivos suaves, tareas y recursos. Comparado con el modelo SD, el modelo SR nos da un mayor nivel de detalle de modelado, observando a los actores en la parte interna del modelo, con estas relaciones intencionales. Estos elementos de intencionalidad aparecen no solo como dependencias externas, sino también como elementos internos vinculados por relaciones de medios-fines y descomposición de tareas. Los vínculos medios-fines proporcionan un conocimiento de cómo un actor llevaría a cabo algunas tareas, persigue un objetivo, necesita un recurso, o quiere un objetivo suave; el vínculo de descomposición

de tareas, proporciona una descomposición jerárquica de los elementos de intencionalidad, que crean una rutina. Este modelo se utiliza para describir los intereses y preocupaciones de los stakeholders, y como deberían ser tratados diferentes configuraciones y ambientes [22].

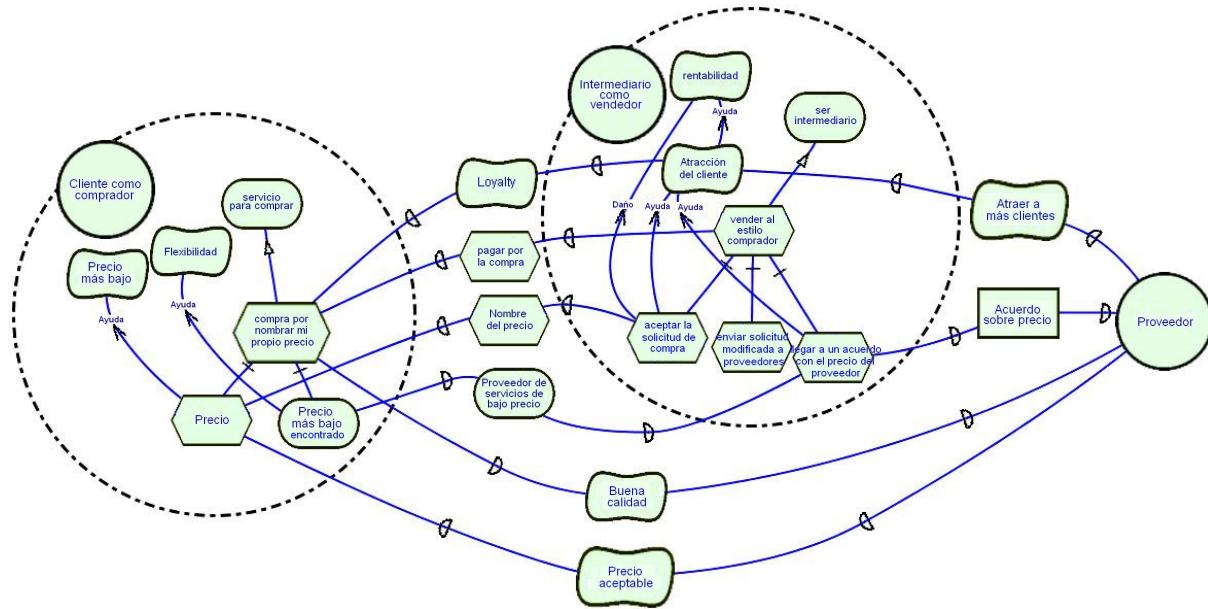


Figura 2.15: Representación modelos SR ejemplo en un E-Commerce

2.2.3. Herramientas que soportan i*

En el sitio oficial de i* [23], se encuentran una variedad de herramientas para la utilización de i*.

- **OpenOme:** se encuentra como aplicación independiente y como plugin para otras herramientas populares como Eclipse¹⁰ y protégé¹¹. OpenOme está diseñado para ser una herramienta de modelado y análisis para la orientación a metas y la orientación a agentes [24].
- **Ome:** es un editor gráfico que soporta orientación a metas y orientación a agentes [25].

¹⁰Eclipse es un entorno de desarrollo integrado de código abierto multiplataforma para desarrollar lo que el proyecto llama “Aplicaciones de Cliente Enriquecido”.

¹¹Protégé es un editor libre de código abierto y un sistema de adquisición de conocimiento. Al igual que Eclipse, Protégé es un framework para el cual otros proyectos sugieren plugins. La aplicación está escrita en Java y usa fuertemente Swing para crear su compleja interfaz.

- **REDEPEND-REACT-BCN:** es una herramienta que soporta modelado en i* y el análisis de estos modelos. Esta versión es una ampliación de otra herramienta llamada i* REDEPEND. Esta herramienta tiene su foco en la representación de los sistemas de información usando el framework i* y da funcionalidades específicas para la generación y evaluación de arquitecturas alternativas [26].
- **TAOM4E:** apoya un modelo impulsado por el desarrollo de software orientado a agentes y, en particular, la metodología de *Tropos*¹². Se ha diseñado teniendo en cuenta las recomendaciones de Model Driven Architecture (MDA) [28].
- **GR-Tool:** *forward and backward reasoning* es soportado en Tropos por una herramienta de razonamiento objetivo (GR-Tool). Básicamente, el GR-Tool es una herramienta gráfica en la que es posible establecer los modelos de objetivo y ejecutar los algoritmos y herramientas para el *forward and backward reasoning*. Los algoritmos para el *forward and backward reasoning* han sido completamente desarrollados en Java y se almacenan en el GR-Tool [29].
- **T-Tool:** T-Tool proporciona un marco para la utilización eficaz de los métodos formales en la fase temprana de requisitos. El marco permite el análisis formal y la mecanización en las fases tempranas de especificaciones de requerimientos expresados en un lenguaje de modelado formal [30].
- **ST-Tool:** Es definida como *Secure Tropos tool*, es una herramienta gráfica donde es posible establecer modelos de secure Tropos y para realizar el análisis formales de especificaciones Secure Tropos. La herramienta está escrita en Java con los componentes Swing, y utiliza XML como formato de documento. El análisis formal se basa en la programación lógica. ST-Tool permite a diferentes sistemas basados en el registro de datos para analizar las especificaciones de Secure Tropos [31].
- **J-PRIM:** JPRIM es una herramienta en Java que soporta PRIM, una metodología que se ocupa de modelos i* y el análisis desde el punto de vista de Reingeniería de Procesos. J-Prim permite analizar un sistema de información existente y para representarla como una jerarquía de elementos i*. Una vez modelada, varias alternativas para el sistema se pueden explorar, cada uno de un modelo como un modelo diferente de i*. Todas las alternativas generadas pueden ser evaluadas mediante la definición y aplicación de indicadores sobre los modelos i* con el fin de establecer cuál es la más apropiada para el sistema de cómo debe ser [32].

¹²Tropos es una metodología para el diseño de Sistemas Multiajente (SMA), liderada por Universidades Italianas y con apoyo de algunas otras a nivel mundial. En su sitio web [27] se pueden encontrar documentos y descargar las herramientas desarrolladas por este grupo.

- **jUCMNav:** jUCMNav es un editor gráfico de notación de requerimientos de usuarios UIT-T (Z.150). URN se compone de dos anotaciones complementarias: la notación de escenario de Mapa de Caso de Uso (UCM) y el lenguaje de requerimientos de la orientación a objetivos (GRL). GRL se basa en i* y los frameworks de NFR¹³. jUCMNav es un plug-in para Eclipse que proporciona a los editores las capacidades, entre ambos puntos de vista, de análisis (incluidas las evaluaciones del modelo GRL), e importar diversos formatos de exportación [33].
- **DesCARTES:** Está diseñado para apoyar la edición de varios modelos: los modelos i* (SD y SR) 2.2.2, modelos NFR, modelos UML, modelos UML en el contexto de Tropos y la evolución-Tropos. Descartes es un plug-in para Eclipse. La originalidad de la herramienta es que permite el desarrollo de la metodología de análisis y modelos de diseño. Integra también capacidades de ingeniería y un módulo de proyectos de software de gestión integrada [34].

También tenemos herramientas de modelado general que nos ayudan a diseñar representaciones en i*.

- **DIA:** Permite múltiples representación de esquemas, por lo que se pueden generar los elementos de i* de forma manual, pero no es específico del sistema [35].
- **Visio:** Existe un Plugin que permite generar los elementos de i* como un diagrama [36].

A continuación presentamos una tabla 2.2 que se estructura de la siguiente forma:

- Existen 8 columnas y cada una de ellas representa a una herramienta de las mencionadas anteriormente.
- Existen 12 preguntas concretas que pueden ser contestadas con un si o un no. Si la respuesta es negativa se pone una cruz. Si es afirmativa, un victo bueno.
- Las preguntas están enfocadas a funcionalidades que la herramienta puede o no tener.

¹³Requerimientos No Funcionales.

		Open OME	OME	REDEPEND REACT BCN	TAOM4E	ST Tool	J PRIM	jUCM Nav	Des CARTES
1	¿La herramienta permite navegar por la metadata?	X	X	X	X	X	X	X	X
2	¿La herramienta soporta técnicas de modelamiento de procesos empresariales?	✓	✓	✓	✓	✓	✓	✓	✓
3	¿La herramienta cuenta con soporte para simulación?	X	X	X	X	X	X	X	X
4	¿La herramienta permite procesamiento ejecutable de procesos de negocio?	X	X	X	X	X	X	X	X
5	¿La herramienta permite crear modelos basados en SOA?	X	X	X	X	X	X	X	X
6	¿La herramienta permite modelar requerimientos?	✓	✓	✓	✓	✓	✓	✓	✓
7	¿La herramienta permite mapear requerimientos a una arquitectura de software?	X	X	✓	X	X	X	X	X
8	¿La herramienta soporta el framework i*?	✓	✓	✓	✓	✓	✓	✓	✓
9	¿La herramienta permite crear modelos SR?	✓	✓	✓	✓	✓	✓	✓	✓
10	¿La herramienta permite crear modelos SD?	✓	✓	✓	✓	✓	✓	✓	✓
11	¿La herramienta permite mapear requerimientos a SOA?	X	X	X	X	X	X	X	X
12	¿La herramienta permite mapear i* a SOA?	X	X	X	X	X	X	X	X

Tabla 2.2: Tabla comparativa de las diferentes herramientas para modelar i*

Revisando las herramientas existentes se puede visualizae que ninguna de ellas genera una modelo arquitectural a partir de la representación en i*.

En conclusión las herramientas pueden expresar los requerimientos pero no pueden ser mapeados a alguna arquitectura como muestra la siguiente tabla comparativa de la Figura 2.2.

2.3. Trabajos de i* y SOA

En la metodología que se basa este trabajo de título [2], comenta el punto de vista de Estrada, quién se enfoca en la creación de modelos SOA implementando i*. En su trabajo se distinguen 3 niveles de abstracción:

- Servicios
- Modelos, y
- Protocolos

El enfoque incluye:

1. Un lenguaje de modelado conceptual, basado en i*, el cual define los conceptos del modelado como sus relaciones correspondientes.
2. Una arquitectura orientada a servicios específica para un modelo de i*, que define los componentes de servicio y el modelo de los diagramas.
3. Un método de modelado de negocios para representar los servicios nivel de organización.

Las arquitecturas de servicios son descritas por tres modelos complementarios que ofrecen una visión de lo que una empresa ofrece y lo que tiene a cambio. Estos modelos son:

Global model: el proceso de modelado de la organización comienza con la definición de alto nivel de los servicios ofrecidos y utilizados por la empresa. El Global Model permite la representación de los servicios del negocio y del actor que juega el papel de solicitante y proveedor. En este modelo se definen los servicios básicos y compuestos.

Process Model: Una vez que los procesos empresariales se han identificado. Ellos deben ser descompuestos en un conjunto de procesos que deben ser realizados. Esto se hace con el Process Model que representa la abstracción funcional de los procesos de negocio para servicios específicos. Este modelo proporciona los mecanismos necesarios para describir el flujo de múltiples procesos.

Protocol Model: finalmente, la semántica de los protocolos y las transacciones de cada proceso de negocio es representado en un sólo diagrama usando los constructores principales de i*. Este modelo nos da una descripción de un conjunto de actividades estructuradas y asociadas que producen un resultado específico o producto para un servicio del negocio.

La Figura 2.16 muestra el proceso por el cual se pasa de un modelo al siguiente.

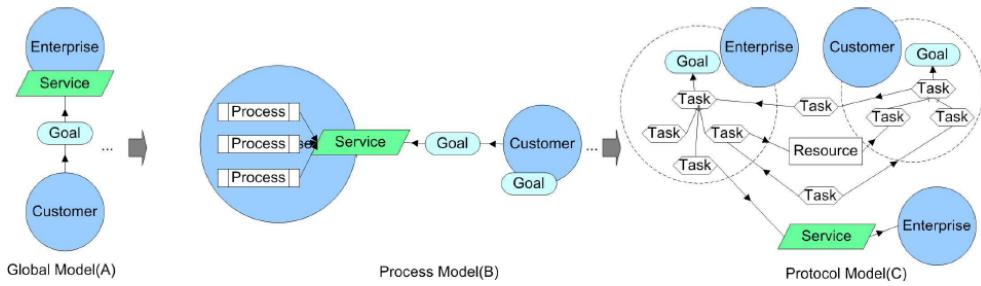


Figura 2.16: Acercamiento a SOA con i*

Este enfoque permite analizar una definición de un modelo de protocolos (Protocol Model), realizando la descripción de los procesos en modelos separados.

2.3.1. Representación de Arquitecturas de Software Orientado a servicios

Para poder mapear requerimientos a diseños arquitecturales debemos formalizar el modelo arquitectural como un objetivo, que debe incluir nociones de servicios, componentes e interfaces de distintos modelos de abstracción.

El proceso de conversión de i* a SOA, está basado en el trabajo de Grau y Franch [37], que definen importantes niveles de abstracción para componentes intencionales.

- **Service:** es un conjunto de funcionalidades de software relacionadas y políticas que controlan su uso. Un servicio es accesible por un protocolo de comunicación estándar independiente de la plataforma o del lenguaje de programación.
- **Capacidades del Servicio:** Es un conjunto de operaciones definidas para cada servicio independiente de su implementación, por lo tanto, esta noción es especialmente usada durante la etapa de modelado de servicios cuando el diseño físico de un servicio no se ha determinado.
- **Componentes de Servicio:** Representa los componentes específicos que pueden ser integrados dentro de un servicio, para implementar una capacidad.

Los conectores son descritos de acuerdo con su nivel de abstracción. Se proponen los siguientes tipos

- **Relaciones Intencionales:** involucran humanos o actores organizacionales y son representados en el modelo de requerimientos. Estos representan la necesaria intención del actor sobre el sistema.

- Dependencias entre Objetivos: Requerimientos funcionales sobre el sistema.
- Dependencias de Recursos: Flujo de conceptos, o concepto relevantes del dominio que no existen físicamente.
- Relaciones Arquitecturales: Ocurren entre los componentes de servicios o servicios, como los siguientes:
 - Interfaces de servicios: describe la relación entre los servicios. Las definiciones de dependencia se encuentran *ocultas* de las propiedades de implementación, haciéndolas independientes del lenguaje de programación o la tecnología. Las interfaces de los servicios son descritas por WSDL
 - Interfaces de componentes de servicios: Describen la relación de los componentes con un servicio, ellos son descritos por la notación propuesta por Han [38]

Algunos conceptos de patrones de servicios son requeridos para aplicar esto en diferentes niveles de abstracción.

El conjunto de patrones de Thomas Erl son usados [39]:

- **Vista de patrones de Servicios:** en este modelo nosotros usaremos los patrones de diseño de la orientación a servicios para describir la arquitectura del sistema. Hay dos subvistas del modelo:
 - **Vista de Patrones de diseño de Servicios:** Muestra la estructura de los componentes para cada servicio, basado en los patrones de servicios¹⁴.
 - **Vista de patrones de servicios de Composición:** Muestra la estructura y dependencias de servicios que forman el sistema en desarrollo¹⁵.

No existen más trabajos que aborden el enfoque de este tema.

¹⁴implementación de redundancia, servicio de replicación de información, etc.

¹⁵servicios de agentes, consultas asíncronas, etc.

Capítulo 3

Definición del problema y Análisis

Para encontrar solución a la problemática encontrada en el capítulo anterior, primero se debe estructurar el problema en forma detallada con los análisis respectivos.

Se podrá ver en detalle el problema, análisis sobre SOA, análisis de i*, junto a la solución propuesta con sus objetivos generales y específicos.

3.1. El problema

Existen muchas herramientas de visualización que nos permiten generar diagramas complementarios al desarrollo de software. También si buscamos un poco más encontramos software que a partir de una especificación generan código o arquitecturas lista para utilizar. El problema específico que se plantea en este trabajo es que no existen herramientas que generen sistemáticamente arquitecturas orientadas a servicios a partir requerimientos orientados a metas.

3.2. Arquitecturas Orientadas a Servicios

SOA o Arquitectura Orientada a Servicios, es una plataforma que se destaca por tener servicios débilmente acoplados y altamente interoperables, lo que es de gran ayuda al momento de la implementación en cualquier tipo de sistema que se quiera desarrollar. Además otro factor importante, es que SOA se basa en definiciones formales que son independientes de la plataforma de desarrollo o del lenguaje de programación, es así como podemos crear servicios en C y que puedan ser utilizados por una aplicación en JAVA[5]. La orientación a servicios da la posibilidad reemplazar un servicios sin preocuparse por la tecnología ya que

sus interfaces son las importantes y funcionan según un especificación web y XML que es universal. Con esto se puede asegurar activos de software de una compañía como son bases de datos, aplicaciones y hacerlo parte de una solución global [1].

Así SOA es una excelente alternativa para escoger al momento de hacer desarrollo de software que concuerde con las características de la orientación a servicios.

3.3. Orientación a metas

Existen paradigmas que permiten modelar eficientemente representaciones más complejas de lo acostumbrado. Esto es la orientación a metas (Oriented Goal - OG) que hace fácil visualizar metas, tareas y servicios para ser alcanzadas por actores[5]. Por lo que la utilización de este paradigma ayuda a obtener una representación altamente compleja de la vida real y no tan acotada como a la que estamos acostumbrados. Además existe el framework i*(i star) que nos da una serie de herramientas que nos permiten estandarizar nuestro proceso de modelado para llevarlo y transformarlo a otro tipo de estructuras.

3.4. Solución al Problema

La solución a este problema es poder desarrollar una herramienta CASE que nos permita diseñar arquitecturas orientadas a servicios a partir de especificaciones de requerimientos en i*. SOA puede ser fácilmente asociado a i* [21], debido a que incorporan factores en común como son los servicios, recursos, y actores que necesitan de estos servicios. La metodología para realizar esta conversión ya existe y soluciona también el problema de la generación de patrones de arquitectura SOA desde i* [2].

3.5. Objetivo General

El objetivo principal del trabajo de título fue crear una herramienta CASE que permitió sistematizar la descripción de arquitecturas orientadas a servicios utilizando para ello una especificación de requerimientos en i*.

3.6. Objetivos Específicos

Los objetivos específicos consistieron en lo siguiente:

- Creación de una herramienta que permitió generar modelos según el standar i*(Global Model,Service Process Model,Service Design Pattern View,Service Composition Design Pattern View) [2].
- realización de conversiones automáticas entre modelos a través de patrones SOA de estructuración de componentes.
- Generación de descripciones de arquitecturas SOA a nivel de descripción de componentes.

Capítulo 4

Especificación de Requerimientos

A continuación se describirán los requerimientos funcionales(RF) y requerimientos no funcionales(RNF) que indican lo necesario para desarrollar nuestra herramienta. También se realizan los diagramas conceptuales necesarios para el entendimiento del dominio del problema junto con las descripciones de los distintos usuarios de GOSOA¹. y sus casos de uso.

4.1. Requerimientos Funcionales

- Debe existir perfiles usuarios para el manejo de la herramienta (Usuarios Registrados y administrador).
- Debe existir un administrador, quien tiene acceso a la edición de los usuarios.
- Debe existir un contenedor principal en la aplicación donde se puedan escoger algunos de estos modelos, soportados en la metodología:
 - Global Model.
 - Process Model.
 - Protocol Model.
 - Service Design Pattern View.
 - Service Composition Design Pattern View.
 - Service Components View.
- Debe poder navegar entre los modelos especificados anteriormente.

¹GOSOA es el nombre de la aplicación que se divide en dos partes: GO por Goal Oriented, y SOA, por Service Oriented Arquitecture.

- Deben existir elementos de i* y los soportados en la metodología, como:
 - Actores.
 - Recursos.
 - Tareas.
 - Procesos.
 - Objetivos.
 - Objetivos suaves.
- Debe haber consistencia de los operadores descritos en la metodología, como:
 - Enlace de dependencia.
 - Enlace final.
 - Enlace de descomposición.
 - Enlaces soportados por los patrones de Thomas Erl [39].
- Los elementos mencionados anteriormente deben poder ser arrastrados al contenedor principal de la aplicación.
- Debe poder generar relaciones entre los modelos descritos anteriormente (Global, Process, Protocol, Service Design Pattern, Service Composition Design Pattern View, Service Components View).
- Debe poder especificar campos que utilizarán los componentes para su posterior mapeo a SOA y a la obtención de los WSDL
- Debe poder generar WSDL a partir de los modelos

4.2. Requerimientos no Funcionales

- La herramienta debe tener todas las ventajas de un sistema web.
 - Poder acceder desde cualquier lugar a la utilización de la herramienta
 - Poder ser indexada² por los navegadores.
 - Es independiente del sistema operativo del usuario.

²En informática, tiene como propósito ejecutar la elaboración de un índice que contenga de forma ordenada la información, esto con la finalidad de obtener resultados de forma sustancialmente más rápida y relevante al momento de realizar una búsqueda.

- La mantención y las modificaciones se ven reflejadas de manera instantánea y no debemos descargar ni una actualización o algo por es estilo (Service Pack).
- Debe ejecutarse distintos navegadores. por lo menos en:
 - Chrome Versión 11.0.696.60. o superior.
 - Mozilla Firefox Versión 3.6. o superior.
 - Internet Explorer Versión 9.0. o superior.
- Debe alojarse en un servidor que pueda ejecutar contenido web.
- Debe existir manejo de bases de datos para la gestión de usuarios.
- El tiempo máximo de respuesta debe ser de 8 segundos entre cambios de modelos.
- La concurrencia debe ser manejada por el alojamiento web.

4.3. Modelo Conceptual

Para poder entender el sistema GOSOA, debemos tener en cuenta los conceptos fundamentales del dominio de nuestro problema, por ello a continuación se muestra el modelo de la conceptualización del proceso del sistema GOSOA.

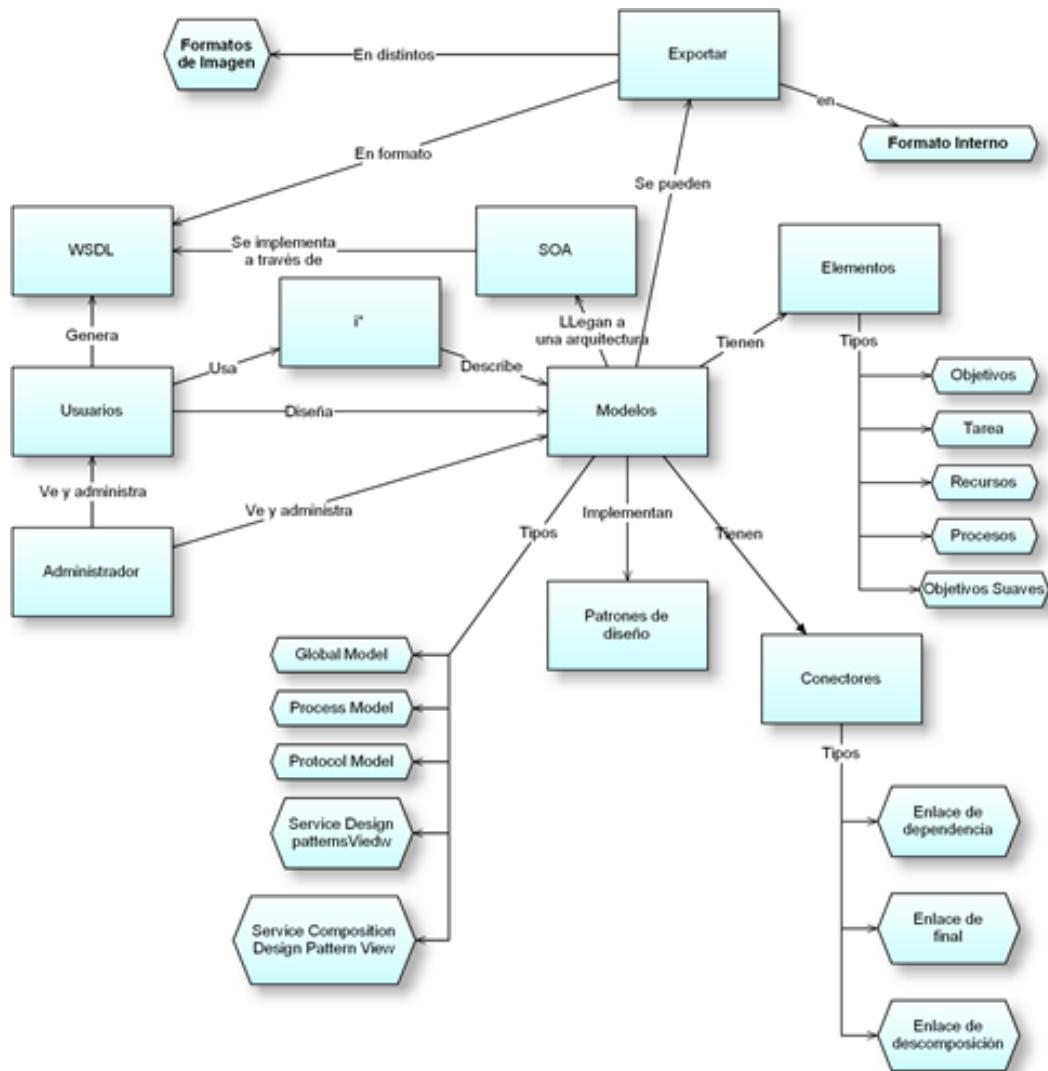


Figura 4.1: Modelo Conceptual

Podemos detallar en parte el diagrama conceptual a continuación separando y explicando cada concepto:

- Administrador:
 - Ve y administrad modelos.
 - Ve y administrad usuarios.
- Usuarios:

- Diseña modelos.
 - usa i * para el diseño de los modelos.
 - genera WSDL.
- WSDL: son descripciones de webservices en XML.
 - i*: sirve para describir los modelos.
- Modelos:
 - Implementan patrones de diseño.
 - Existen diferentes tipos de modelos.
 - Global Model.
 - Process model.
 - Protocol model.
 - Service Design pattern View.
 - Service Composition Design Pattern View
 - Tienen conectores.
 - Enlace de dependencia.
 - Enlace de final.
 - Enlace de descomposición.
 - Elementos:
 - Objetivos.
 - Tareas.
 - Recursos.
 - Procesos.
 - Objetivos Suaves.
 - Pueden Exportar.
 - Llegan a una arquitectura SOA.
- SOA:
 - Se implementa a través de los WSDL.
 - Exportar:
 - En formato de imagen.
 - En formato interno.
 - WSDL.

4.4. Descripción Usuarios

El sistema debe poder manejar dos niveles de usuarios. Estos niveles los denominaremos administrador y usuario registrado. A continuación se explica en detalle cada uno:

- Administrador: Usuario con la posibilidad de administrar los usuarios que existen en el sistema. También tiene acceso a toda la información que los usuarios crean en el sitio (puede ver lo modelos creados).
- Usuario Registrado: Es la persona que utiliza el sistema y puede guardar información propia de los modelos que ha realizado.

A continuación en la Figura 4.2 se describe la relación de los usuarios con el sistema.

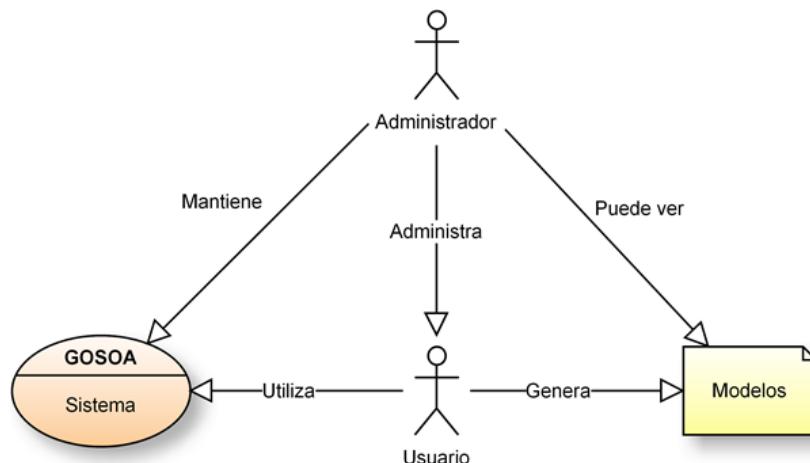


Figura 4.2: Relación de usuarios y sistema

4.5. Casos de Uso

Los casos de uso son una especificación del comportamiento de distintos escenarios del sistema. Y estos serán divididos en funcionalidades y luego se realizará una representación global de estos.

4.5.1. Acceso al Sistema:

El acceso al sistema se divide en dos partes. En el acceso del usuario registrado y el acceso del administrador. Ambos deben realizar el mismo acceso pero las funcionalidades que tienen ambos no son las mismas.

El usuario registrado al ingresar al sistema debe iniciar una sesión para tener acceso a las funcionalidades de este. El usuario tendrá un perfil con datos personales y modelos que ha creado, y en donde podrá administrarlos. Esto se representa en la Figura 4.3.

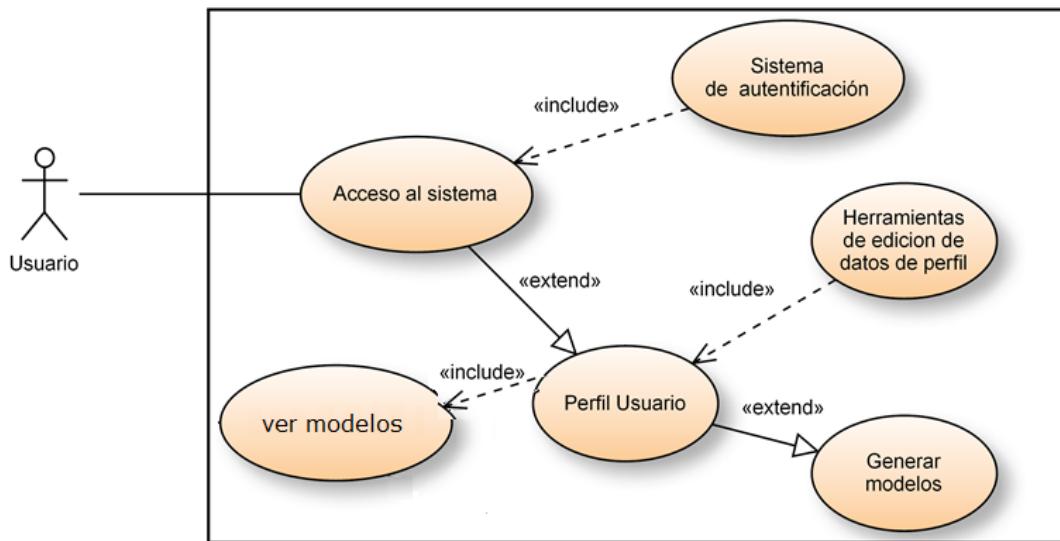


Figura 4.3: Acceso del usuario al sistema

El administrador también debe acceder al sitio. Este acceso se hace por el mismo medio, pero se cargan funcionalidades distintas. El administrador puede cambiar aspectos y funcionalidades del sitio y también puede modificar, eliminar o agregar usuarios registrados. Estas funcionalidades se incluyen en “Administra Modelos” y la “Herramienta de edición de datos de perfil”. la Figura 4.4 muestra aquello.

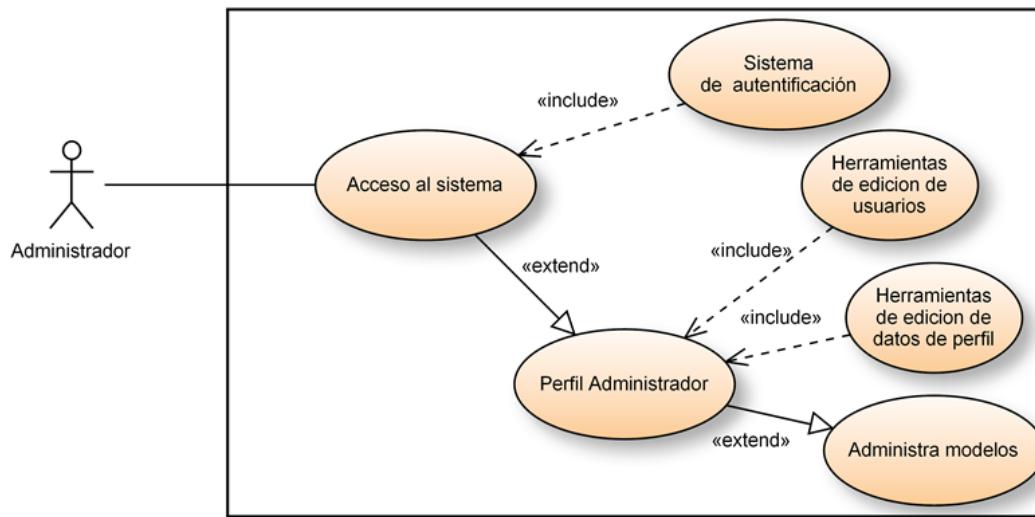


Figura 4.4: Acceso del Administrador al sistema

4.5.2. Caso de Uso Gestión de Usuarios

El Administrador debe tener la posibilidad de gestionar a los usuarios registrados que utilizan el sistema, esto es poder buscar, crear, eliminar y modificar estos. Esto se puede visualizar en la Figura 4.5.

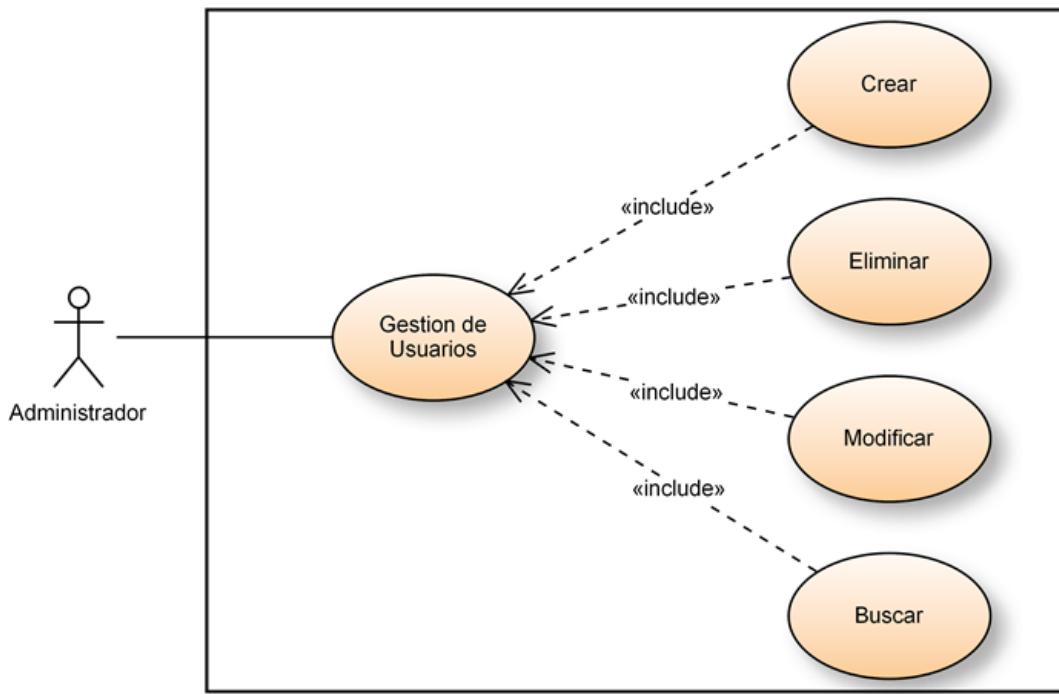


Figura 4.5: Gestión de Usuarios

4.5.3. Selección de Diagramas:

El usuario registrado puede escoger por cual diagrama comenzar a modelar, por esto, puede elegir cualquiera de los modelos especificados en la metodología.

Al seleccionar cualquier modelo se deben cargar las herramientas adecuadas para la generación de este. Esto se puede visualizar en la Figura 4.6.

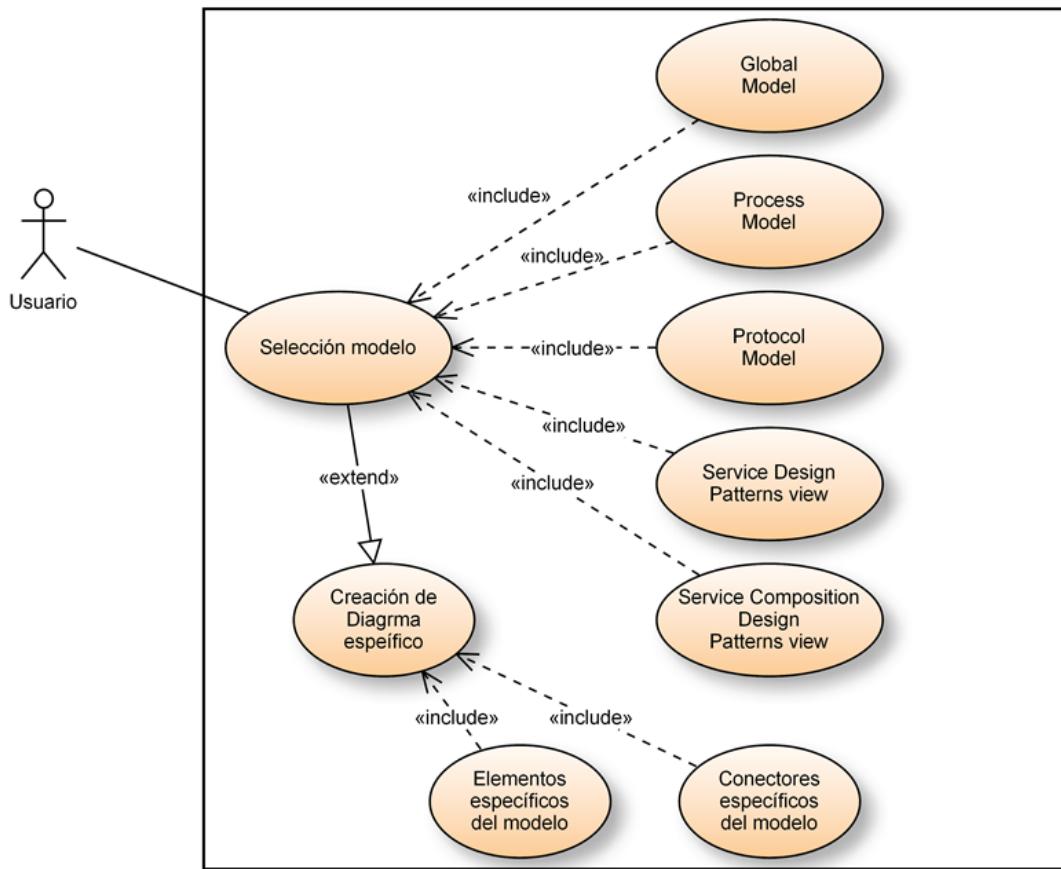


Figura 4.6: Selección de un modelo por el usuario

4.5.4. Pasar a Diagrama Siguiente o Posterior:

Al pasar de un diagrama a otro se pueden incorporar herramientas que no eran necesarios en el modelo anterior. Así cuando realizamos estos pasos se van incorporando elementos, que llamaremos “Se cargan Elementos Modelo Siguiente/Posterior”, y conectores “Se cargan Conectores Modelo Siguiente/Posterior”. Esto se puede visualizar en la Figura 4.7.

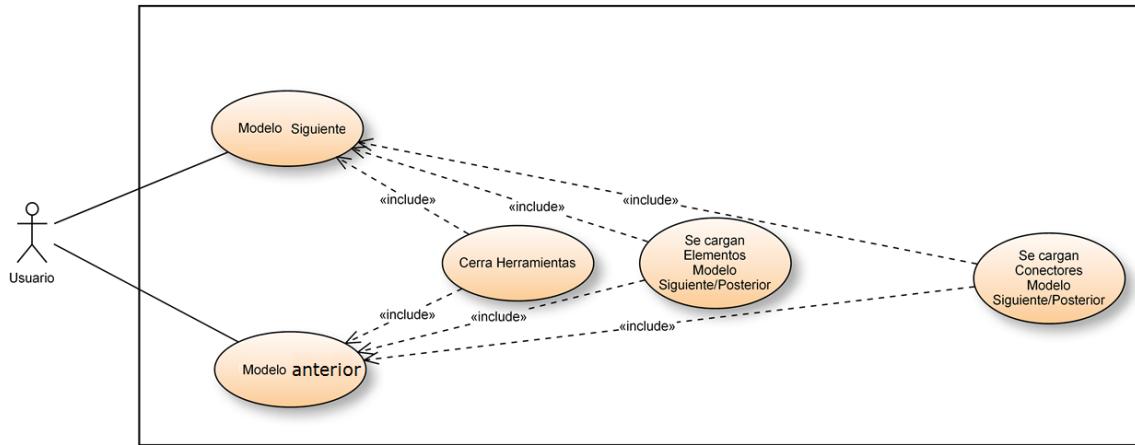


Figura 4.7: Caso de Uso para Avanzar y Retroceder en los diagramas

4.5.5. Exportar Modelo:

Debe existir la posibilidad de exportar el modelo a una imagen o un WSDL. Al utilizar la funcionalidad llamada “Exportar Modelo” se ejecutan dos módulos llamados “Generar Imagen”, para generar la imágenes del sistema, y “Generar WSDL”, para generar los WSDL del sistema. Esto se puede visualizar en la Figura 4.8.

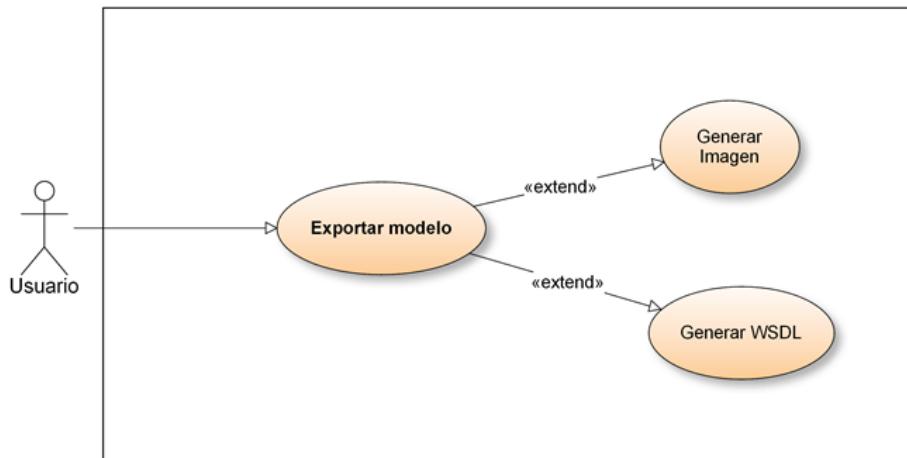


Figura 4.8: Caso de Uso para Exportar Modelo

4.5.6. Configurar Datos:

Al configurar los datos podemos editar dos tipos de datos: los datos básicos; los datos Avanzados. Los datos básicos son los que describen cosas básicas como el nombre del elemento o el conector. y en la configuración avanzada se pueden cambiar los datos que pertenecen a la configuración final de SOA (que permitirá la comunicación en la implementación posterior de SOA a través del WSDL).

La funcionalidad que permite cambiar los datos básico de un elemento o conector se llamará “Cambiar Datos Básicos”. En la funcionalidad principal “Configurar datos” puede moverse a una funcionalidad para “Configurar Datos SOA”, la cual implementa la funcionalidad “Cambiar Campos de Comunicación”. Esto se puede visualizar en la Figura 4.9.

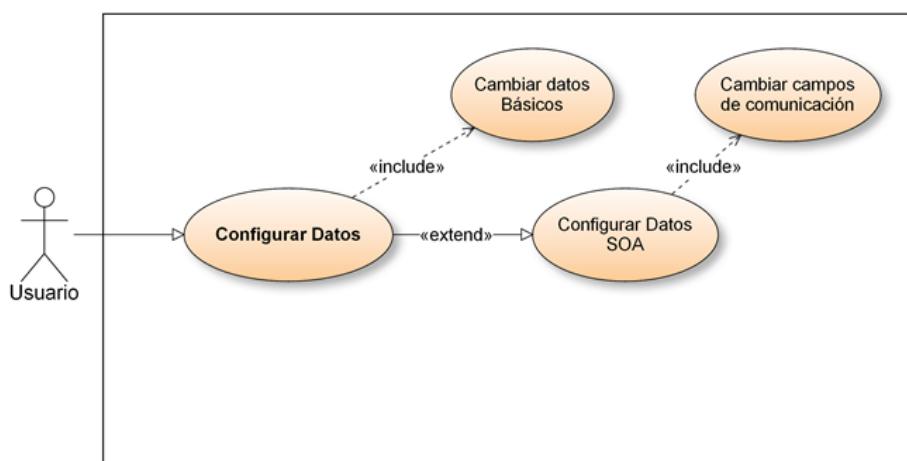


Figura 4.9: Caso de Uso para Configurar Datos

4.5.7. Caso de Uso General:

Finalmente se genera un caso de uso general incluyendo todas las funcionalidades que se indicaron en los puntos anteriores, que al juntarlas se obtiene el diagrama mostrado en la Figura 4.10.

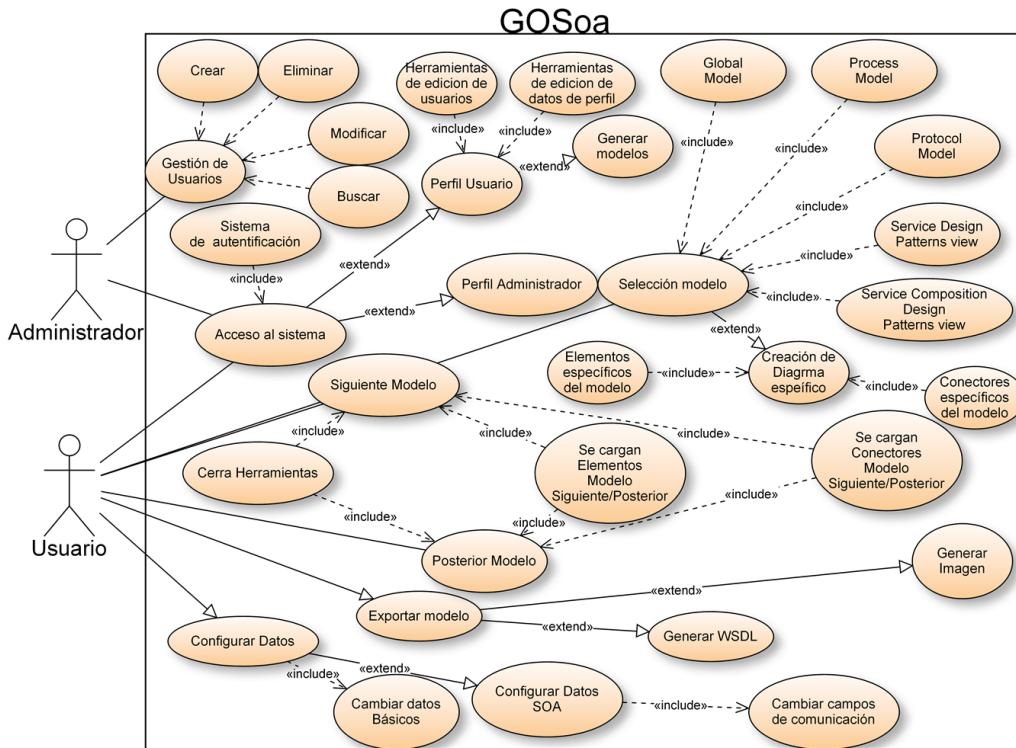


Figura 4.10: Casos de Uso para Configurar Datos

En la Figura 4.10 se muestra la conjunción de todos los casos de usos y cual es su representación general en el sistema. Los diagramas que se incluyen son los mencionados en las secciones anteriores:

- Acceso al sistema.
- Caso de uso Gestión de usuarios.
- Selección de Diagramas.
- Pasar a Diagrama siguiente o posterior.
- Exportar Modelo.
- Configurar Datos.

Se pueden encontrar los casos de uso extendidos, diagramas de secuencia y diagramas de estado en el Anexo A.1.

Capítulo 5

Diseño

Para el diseño de la aplicación es necesario definir lo siguiente:

- Diseño Arquitectónico
- Modelo Navegacional
- Diseño Lógico
- Diseño de Datos
- Diseño de Pruebas

5.1. Diseño arquitectónico

En la construcción de GOSOA, se utiliza un modelo de 3 capas, que se divide en: capa de presentación, capa lógica y capa de persistencia.

Como muestra la Figura 5.1.

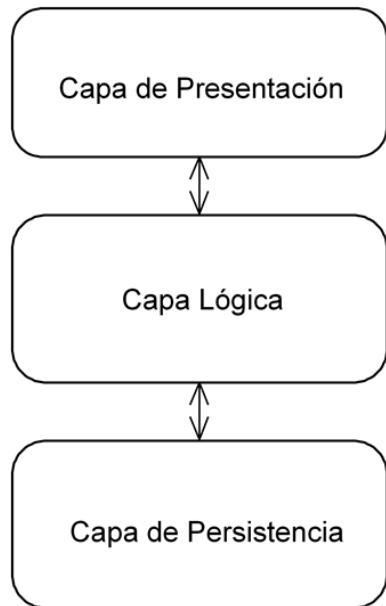


Figura 5.1: Primer Enfoque Arquitectura de tres capas

En la Figura 5.1 se muestra la separación que hay entre las capas y las uniones entre las mismas. Es así como la capa de presentación solo se une a la capa lógica. La lógica envía y recibe datos de la capa de presentación y de la capa de persistencia, y finalmente, la capa de persistencia envía y recibe datos de la capa lógica.

5.1.1. Arquitectura de 3 capas

La arquitectura consistirá en 3 capas bien definidas, e independientes entre ellas. Las cuales se detallan a continuación con sus correspondientes subsistemas.

Capa de Presentación

La capa de presentación es la interfaz que une al usuario con el sistema GOSOA. Así esta capa debe presentar los siguientes subsistemas:

- Subsistema de Login.
- Subsistema de Perfil.
- Subsistema de modelado.

cada uno de los cuales se explica a continuación.

- Subsistema de Login: el subsistema de login se divide en dos módulos, el de login y el de registro. El módulo de login redirecciona al módulo de registro si el usuario no se encuentra registrado. Esto es representado en la Figura 5.2.
 - Módulo Login: Es el módulo que nos permite ingresar los datos (solo la interfaz), para poder autentificarnos en el sistema
 - Módulo Registro: Es el módulo que le permite, a un nuevo usuario, ingresar sus datos para que el administrador posteriormente lo valide como un usuario del sistema.

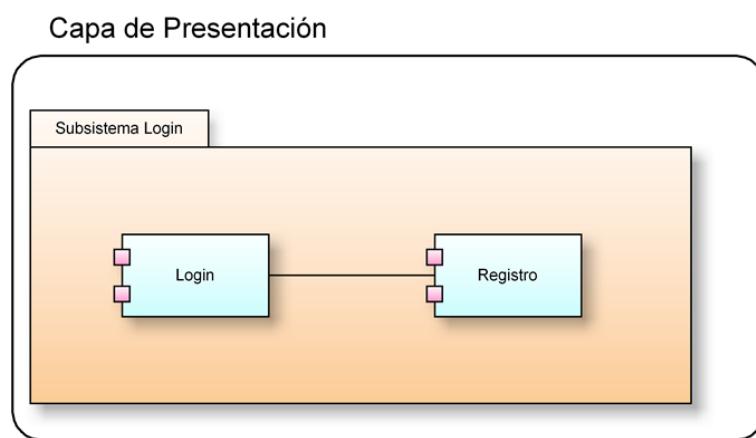


Figura 5.2: Subsistema de Login

- Subsistema de perfil: El Subsistema de perfil se divide en dos módulos, el Módulo Usuario y el Módulo Administrador como se puede visualizar en la Figura 5.3.
 - Módulo Usuario: es el módulo que muestra la interfaz del perfil del usuario registrado.
 - Módulo Administrador: es el módulo que muestra la interfaz del perfil del administrador.

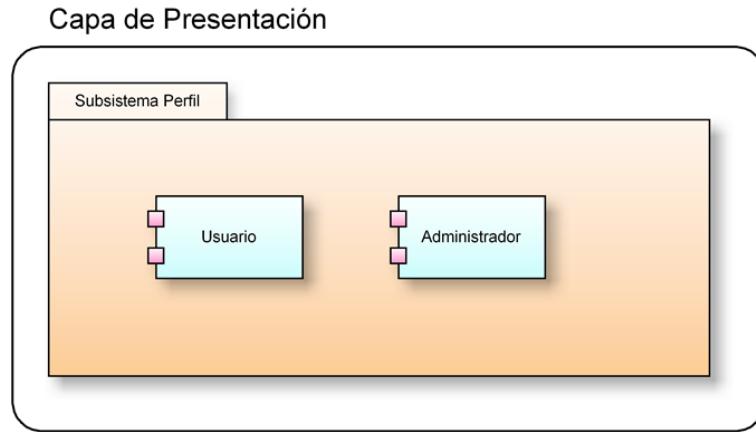


Figura 5.3: Subsistema de Perfil

- Subsistema de Modelado: El subsistema de modelado consta de 5 módulos: Módulo de herramienta de modelado, módulo de conectores, módulo de elementos, módulo de exportación y módulo de patrones. Estos módulos están en la herramienta que es un subsistema externo al módulo de 3 capas y se tiene que mostrar dentro de un contenedor que se llama módulo contenedor.

El módulo contenedor debe cargar el subsistema de la herramienta de modelado. dentro de esta herramienta el módulo principal es la de herramienta de modelado que depende de 4 módulos que le brindan funcionalidades como se presentado en la Figura 5.4.

- Módulo contenedor: es el contenedor que soporta la herramienta que hace posible mostrarla en un entorno web.
- Módulo de herramienta de modelado: es el módulo que contiene la ventana principal de la aplicación donde se generan los modelos por el usuario. Éste utiliza de los otros módulos.
- Módulo de conectores: El módulo de conectores nos permite cargar los conectores pertinentes a cada modelo que estamos generando.
- Módulo de elementos: El módulo de elementos nos permite cargar los elementos pertinentes a cada modelo que estamos creando.
- Módulo de exportación: El módulo de exportación nos permite exportar nuestro modelo creado en tres tipos de formatos.
 1. Formato Imagen.

2. Formato Interno del sistema.
3. Formato WSDL.

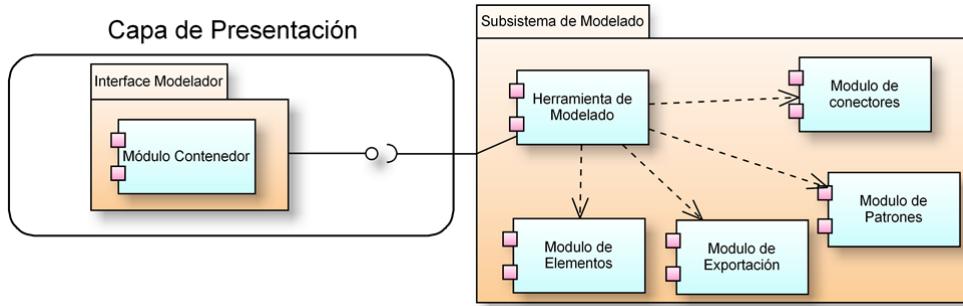


Figura 5.4: Subsistema de Perfil

Así podemos definir la vista general de la capa de presentación. La Figura 5.5 presenta esta representación.

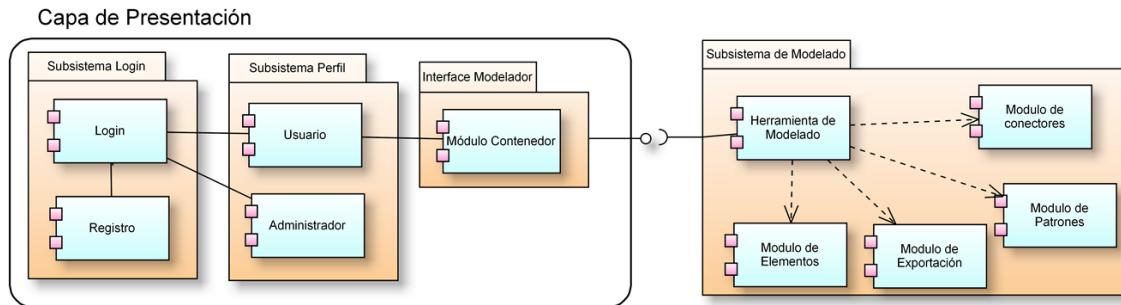


Figura 5.5: Capa de presentación GOSOA

Capa Lógica

La capa lógica es donde residen los programas que se ejecutan, se reciben las peticiones del usuario registrado y se envían las respuestas tras el proceso. Esta capa brinda funcionalidades a la capa superior (capa de presentación) y comunica con las bases de datos (capa de persistencia). Esta capa sólo contiene dos subsistemas:

- Subsistema de autenticación: es el subsistema que brinda funcionalidades al subsistema de login. Donde sus componentes son:

- Módulo Validador de usuario: es el módulo que permite saber si un usuario registrado ha introducido nombre y contraseñas válidas en el sistema. también es el encargado de discriminar si el usuario que inicia sesión es un administrador o un usuario registrado.
- Módulo manejador de registro: Es el módulo encarado de realizar las inscripciones de los usuarios registrados que desean registrarse en GOSOA.

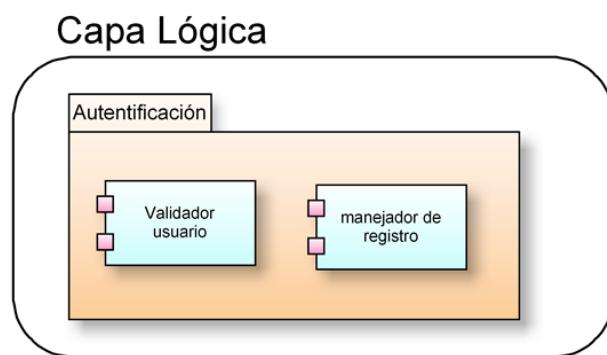


Figura 5.6: Subsistema de Autentificación

- Subsistema Lógica de Perfil: este subsistema alimenta las funcionalidades del subsistema de perfil en la capa de presentación. los módulos que contiene son los siguientes:
 - Módulo Eliminar Usuario: Facultad del administrador para Eliminar usuarios.
 - Módulo Buscar Usuario: Facultad del administrador para Buscar usuarios.
 - Módulo Editar Usuario: Facultad del administrador para Editar datos de los usuarios.
 - Módulo Crear Usuario: Facultad del administrador para Crear usuarios.
 - Módulo Aceptar Usuario: Facultad del administrador para Aceptar usuarios que hayan realizado el registro en el sistema.
 - Módulo Editar datos personales: Facultad que tiene el usuario que se encuentra registrado para poder editar su información personal

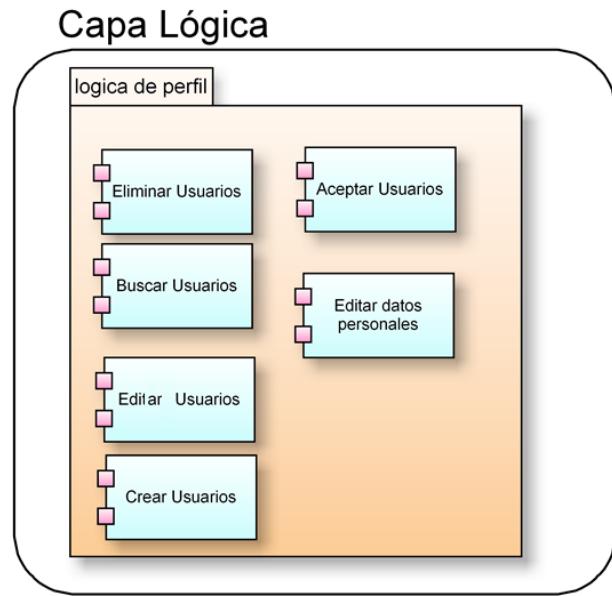


Figura 5.7: Subsistema Lógica de perfil

Generalizando la capa lógica lo que se obtiene es lo que se muestra en la Figura.

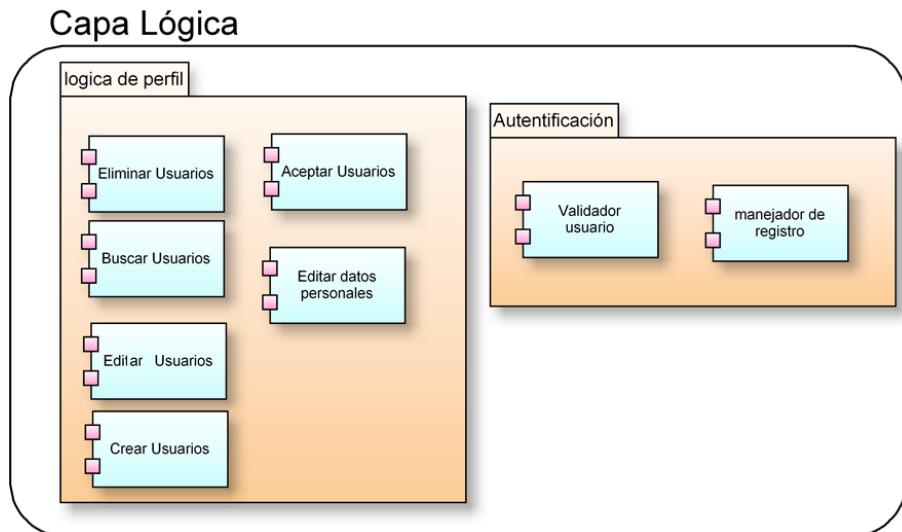


Figura 5.8: Capa Lógica

Capa de Persistencia

Es el sistema de almacenamiento de datos y se accede a ellos a través de MySQL. Los módulos que tienen acceso son todos aquellos que se encuentran en la capa lógica.

La representación de este se puede apreciar en la Figura 5.9.



Figura 5.9: Capa de persistencia

Modelo arquitectónico General

Se han representado las capas del sistema GOSOA por separado, pero están conviven en un sólo sistema que está representado por la Figura 5.10. La que presenta un sistema completo formado por la capa de presentación, capa lógica y capa de persistencia.

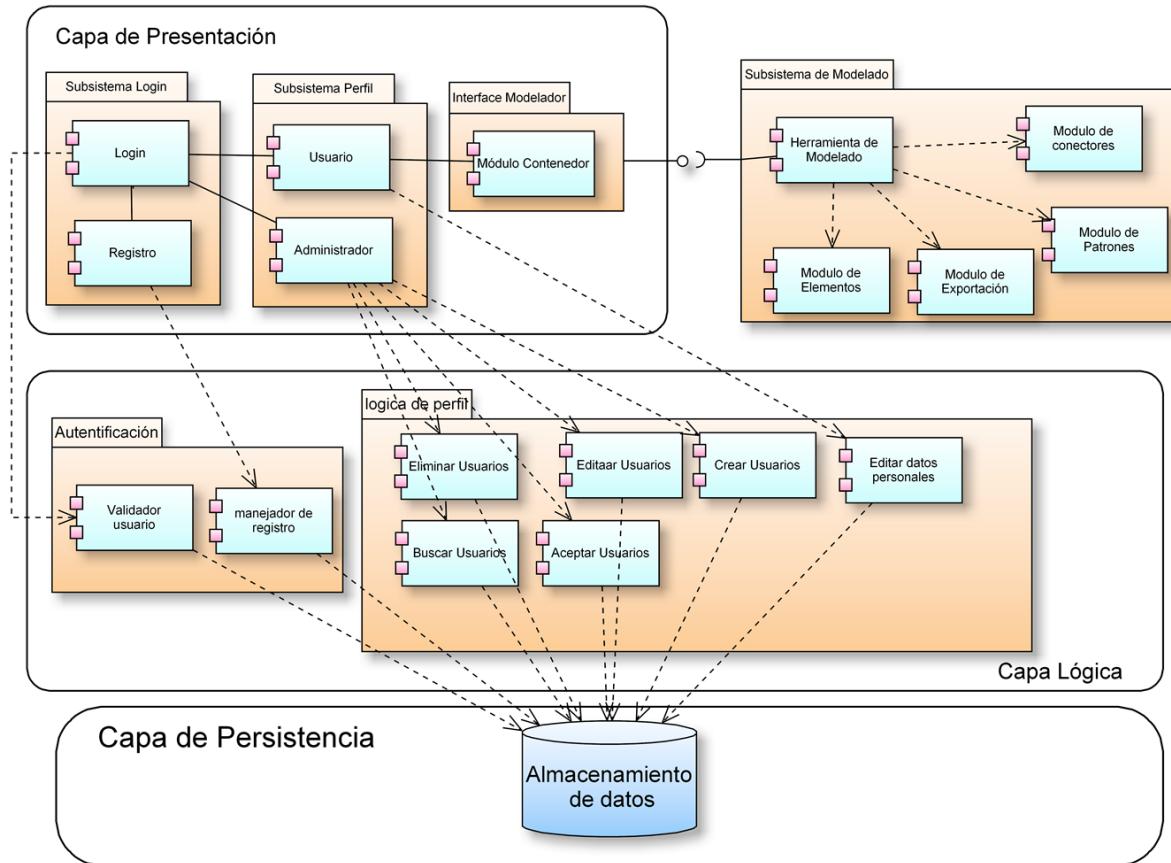


Figura 5.10: Capa de persistencia

5.2. Modelo Navegacional

GOSOA consta con un sistema de navegación que se divide en dos partes:

El Home, que es el sistema de navegación entre las distintas páginas del sitio; Y el sistema de navegación entre los modelos de la aplicación (Global, Process, Protocol, Design Pattern View, Composition Design Pattern View).

La Figura 5.11 muestra el modelo navegacional.

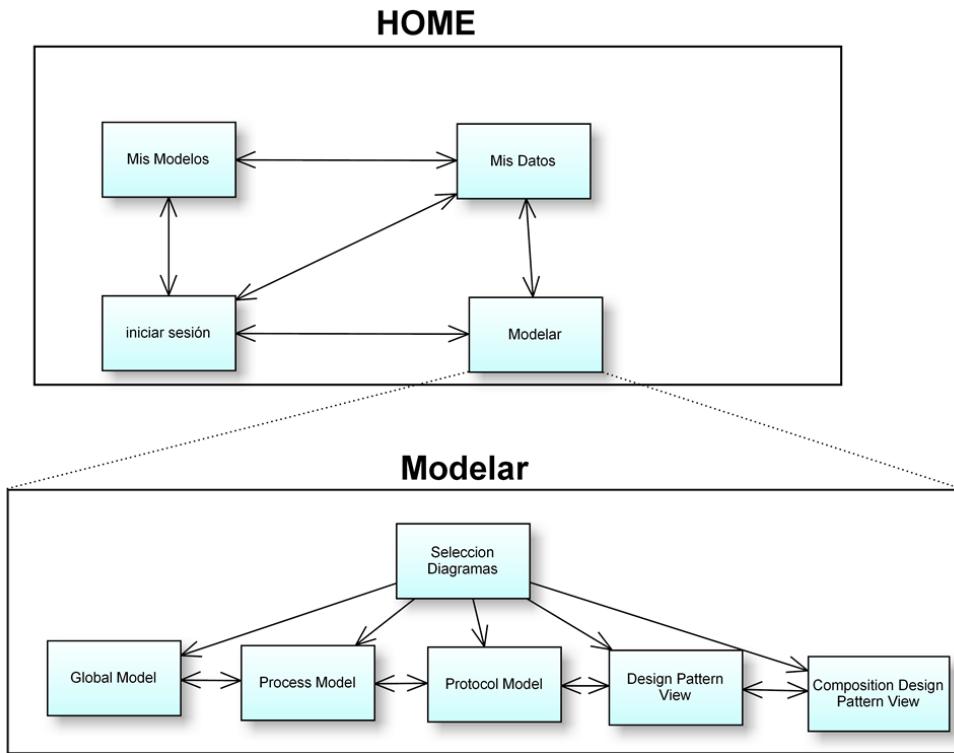


Figura 5.11: Modelo navegacional GOSOA

La navegación del sistema es a través de 4 secciones principales, las cuales son:

- Mis Modelos.
- Mis datos.
- Iniciar Sesión.
- Modelador

5.2.1. Secciones

“Mis modelos” es una sección en la que los usuarios pueden encontrar todos los modelos que han realizado y se han guardado en el sistema.

“Mis Datos” es la sección en que el usuario registrado puede modificar sus datos personales. Debemos también explicar que si el usuario es del tipo Administrador, éste podrá editar los datos de los otros usuarios registrados.

“Iniciar Sesión” es la sección en que se inicia sesión en el sistema.

“Modelador” Es una sección donde se encuentra el modelador de diagramas. En este modelador se puede navegar por los distintos modelos que permite la metodología:

- Global Model.
- Process Model.
- Protocol Model.
- Design Pattern View.
- Composition Design Pattern View.

5.3. Diseño lógico

En esta sección se detalla el diagrama de clases general del sistema y además se detallan las Clases en las subsecciones siguientes.

5.3.1. Diagrama de clases

En el desarrollo de la plataforma GOSOA, como se utiliza la orientación a objetos para su desarrollo, se debe especificar el diagrama de clases respectivo.

En la Figura 5.12 se puede visualizar este diagrama.

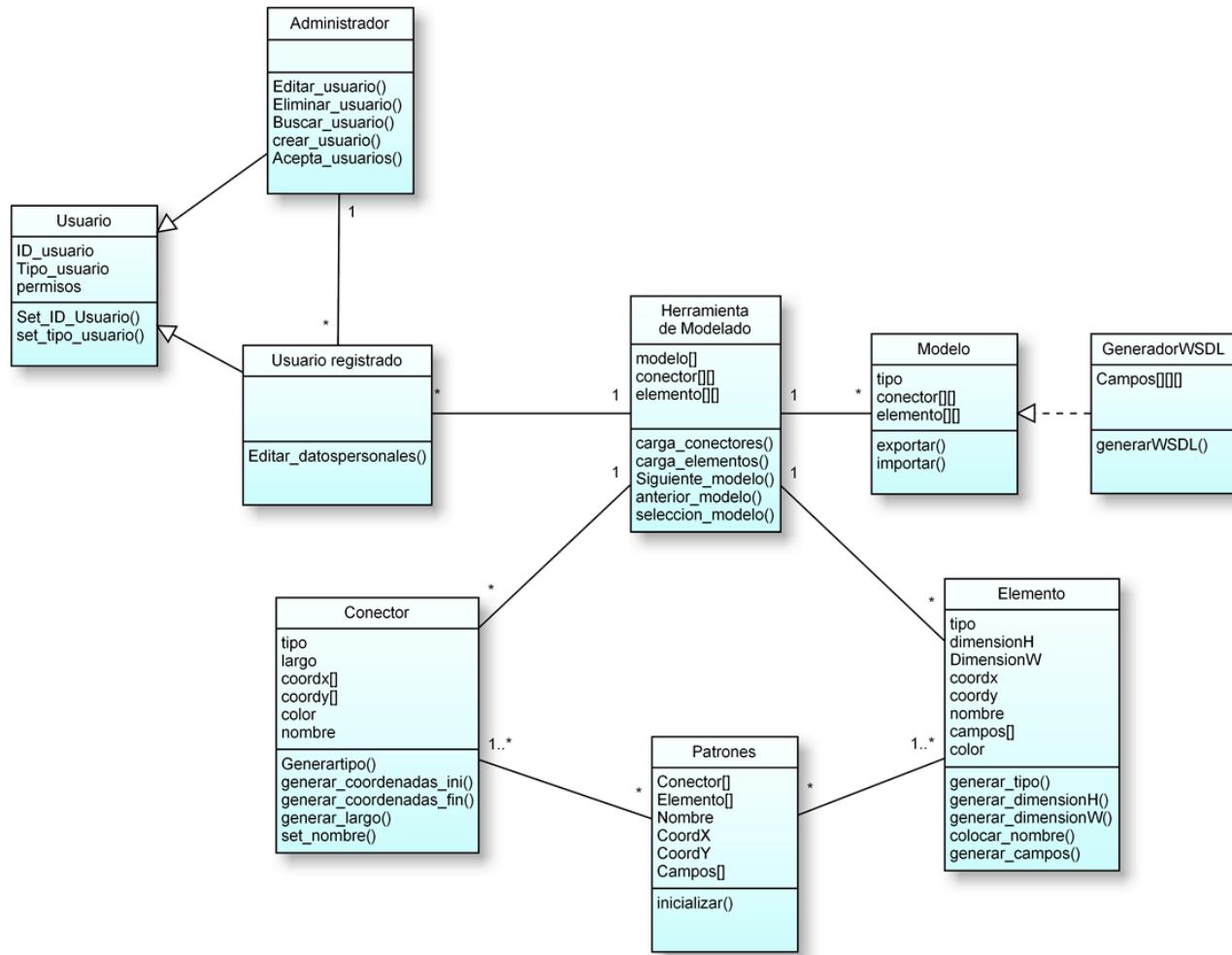


Figura 5.12: Diagrama de Clases

A continuación se muestra el detalle de las clases generadas para la plataforma GO-SOA, junto con sus representaciones pertinentes.

5.3.2. Clase Usuario

Usuario: es la clase que representa a los usuarios registrados que están en el sistema. De esta clase heredan dos clases más (**Administrador**, **Usuario Registrado**). los atributos y métodos de las clases son los siguientes:

- Atributos:

- ID_usuairo: es un identificador único de cada usuario registrado.
 - Tipo_usuario: nos dice el tipo de usuario que ha ingresado al sistema.
 - permisos: nos dice cuales son los atributos y limitaciones de un usuario registrado (enfocado a la posible evolución de la plataforma).
- Métodos
 - Set_ID_Usuario(): permite asignar el ID al usuario registrado correspondiente.
 - Set_tipo_usuario(): permite designar que tipo de usuario registrado ha ingresado al sistema.

5.3.3. Clase Extendida Administrador

Administrador: la clase Administrador es una clase que extiende de Usuario ya que presenta todos los atributos de esta, pero además implementa métodos nuevos.

La clase extendida Administrador se detalla a continuación.

- Atributos: Son los mismos que la clase usuario ya que están heredados.
- Métodos:
 - Editar_usuario(): permite realizar la edición de la información personal de los usuarios.
 - Eliminar_usuario(): permite la eliminación de usuarios registrados del sistema.
 - Buscar_usuario(): permite encontrar usuarios registrados en el sistema.
 - Crear_usuario(): permite crear usuarios registrados nuevos en el sistema sin necesidad de que se registren
 - Acepta_usuarios(): permite aceptar usuarios registrados que estén pendientes a confirmación previo registro de estos.

5.3.4. Clase Extendida Usuario registrado

La clase Usuario Registrado es una clase que extiende de la clase Usuario, por lo cual tiene todos los atributos que tiene esta. A continuación se detalla la clase extendida usuario registrado.

- Atributos: esta clase contiene todos los atributos de la clase Usuario.
- Métodos:

- `Editar_datospersonales()`: permite la edición de los datos personales de un usuario registrado.

5.3.5. Clase Herramienta de Modelado

La clase herramienta de modelado es un contenedor el cual tiene los modelos que se van generando.

A continuación se detalla la clase.

■ Atributos:

- `modelo`: contiene un arreglo de los modelos que se van generando.
- `conector`: contiene los conectores de cada modelo.
- `elemento`: contiene los elementos de cada modelo.

■ Métodos:

- `carga_conectores()`: carga los conectores que se necesitan en el modelo.
- `carga_elementos()`: carga los elementos que se necesitan en el modelo.
- `Siguiente_modelo()`: carga todas las herramientas necesarias del siguiente modelo.
- `anterior_modelo()`: carga todas las herramientas necesarias del anterior modelo.
- `seleccion_modelo()`: se permite seleccionar alguno de los modelos.

5.3.6. Clase Conector

La clase Conector es la que se utiliza para generar todas las flechas y conectores que unen elementos en los diagramas o modelos.

A continuación se detalla la clase.

■ Atributos:

- `tipo`: determina el tipo de conector.
- `largo`: determina el largo del conector.
- `coordx`: muestra la información de las coordenadas x del conector.
- `coordy`: muestra la información de las coordenadas y del conector.

- color: determina el color del conector.
 - nombre: determina si va un nombre en el conector.
- Métodos:
- Generar_tipo(): crea el conector de un tipo determinado.
 - generar_coordenadas_ini(): entrega las coordenadas del inicio del conector.
 - generar_coordenadas_fin(): entrega las coordenadas del fin del conector.
 - generar_largo(): genera el largo del conector.
 - set_nombre(): permite setear el nombre del conector.

5.3.7. Clase Elemento

La clase Elemento es la que se utiliza para generar todas los elementos o figuras en los diagramas o modelos.

A continuación se detalla la clase.

- Atributos:
- tipo: determina el tipo de elemento.
 - dimensionH: determina la altura del elemento.
 - DimensionW: determina el ancho del elemento.
 - coordx: determina la coordenada x de la esquina superior izquierda.
 - coordy: determina la coordenada y de la esquina superior izquierda.
 - color: determina el color del conector.
 - campos: determina si existen más campos de nombres en el elemento.
 - nombre: determina si va un nombre en el conector.
- Métodos:
- generar_tipo(): crea el conector de un tipo determinado.
 - generar_dimensionH(): genera al altura del elemento.
 - generar_dimensionW(): genera el ancho del elemento.
 - generar_campos(): genera los campos extras en el elemento.
 - set_nombre(): permite setear el nombre del elemento.

5.3.8. Clase Patrones

La clase Patrones es la que se utiliza para generar todos los patrones en los diagramas o modelos.

A continuación se detalla la clase.

- Atributos:

- Tipo: determina el tipo de elemento.
- Conector: array de conectores que lo componen.
- Elemento: array de elementos que lo componen.
- Coordx: determina la coordenada x de la esquina superior izquierda.
- Coordy: determina la coordenada y de la esquina superior izquierda.
- Color: determina el color del elemento.
- Campos: determina si existen más campos de nombres en el patrón.
- Nombre: determina el nombre del elemento.

- Métodos:

- inicializar(): inicializa el patrón con los elementos y conectores necesarios.

5.3.9. Clase Modelo

La clase Modelo es la que se utiliza para al modelo que se está haciendo.

A continuación se detalla la clase.

- Atributos:

- tipo: determina el tipo de modelo.
- conector: array de conectores que lo componen.
- elemento: array de elementos que lo componen.

- Métodos:

- exportar(): exporta el modelo en el formato que el usuario escoja.
- importar(): se carga el modelo en el sistema.

5.3.10. Interface GenerarWSDL

La Interface GenerarWSDL es implementada por la Clase Modelo y permite la generación de WSDL.

A continuación se detalla la clase.

- Atributos:
 - Campos: array de todos los campos para generar el WSDL.
- Métodos:
 - generarWSDL(): nos permite generar el WSDL.

5.4. Diseño de datos

A continuación se muestra el diagrama de entidad relación de la plataforma GOSOA.



Figura 5.13: Diagrama relacional de GOSOA

En la Figura 5.13 se ve que existen 3 tablas en el sitio. Las cuales se explican a continuación con sus respectivos atributos.

- **ELEMENTOS**: los elementos son todos los objetos que tienen los modelos, sean estos conectores, elementos o patrones. Y sus atributos son los siguientes:

- Identificador: identificador único de cada elemento que existe en el sistema.
 - Tipo: indica el tipo de elemento que es. Esto es Conektor, Elemento o patrón.
 - Nombre: indica el nombre del elemento.
 - Imagen: muestra una dirección de la imagen en miniatura del elemento.
 - URL: contiene la dirección con las especificaciones de cada elemento.
- MODELOS: son los modelos que se encuentran guardados en el sistema como imágenes, que pueden ser privados o públicos según ha decidido el usuario. Y sus atributos son los siguientes:
- Identificador: identificador único de cada modelo que existe en el sistema.
 - Nombre: nombre del modelo que el usuario ha escogido.
 - Tipo: indica el tipo de modelo al que corresponde (Global Model, Process Model, Protocol Model, Service Design Pattern View, Service Composition Design Pattern View).
 - Identificador_usuario: indica a qué usuario pertenece el identificador almacenado.
 - Público: variable que indica si el modelo es público o no.
- USUARIOS:
- Identificador: identificador único de cada usuario registrado en el sistema.
 - Nombre: indica el nombre de usuario registrado.
 - Email: indica el email del usuario registrado.
 - Contraseña: indica la contraseña del usuario registrado.
 - Tipo: indica el tipo de usuario registrado.
 - Permisos: indica permisos del usuario registrado.

También podemos especificar un diccionario de datos que contiene las características lógicas y puntuales de los datos que fueron utilizados en el sistema GOSOA.

Primero se especifican los datos utilizados en las clases de GOSOA.

Diccionario de Clases			
Variable	Significado	Tipo de Dato	Rango
Usuario.ID_usuario	Identificador único del usuario	Numérico	1-N
Usuario.Tipo_usuario	Indica tipo de usuario	alphanumérico	[admin usuario invitado]
Usuario.Permisos	Indica permisos del usuario	Binario	0000-1111
Herramienta.modelo[]	Array con los modelos almacenados	Modelo	1-N
Herramienta.conector[]	Array con los conectores que se utilizarán en algún modelo	Conector	1-N
Herramienta.elemento[]	Array con los elementos que se utilizarán en algún modelo	Elemento	1-N
Modelo.tipo	Indica el tipo de modelo	Alfanumérico	[global process protocol design composition]
Modelo.Conector[]	Array de conectores en el modelo	Conector	1-N
Modelo.Elemento[]	Array de elementos en el modelo	Elemento	1-N
Modelo.Campos[][][]	Array de campos para generar el WSDL	alphanumérico	*
Conector.tipo	Indica el tipo de conector	alphanumérico	*
Conector.largo	Indica largo del conector	Numérico	1-N
Conector.coordx[]	Coordenadas x de cada esquina del conector	Numérico	1.00-2000.00
Conector.coordy[]	Coordenadas y de cada esquina del conector	Numérico	0.00-2000.00
Conector.color	Indica el color del conector	hexadecimal	#000000-#ffffff
Conector.nombre	Indica el nombre del conector	Alfanumérico	*
Elemento.tipo	Indica el tipo de elemento	alphanumérico	*
Elemento.DimensionH	Indica altura del elemento	Numérico	0.00-2000.00
Elemento.DimensionW	Indica ancho del elemento	Numérico	0.00-2000.00
Elemento.coordx	Coordenada x de la esquina superior izquierda	Numérico	1.00-2000.00
Elemento.coordy	Coordenada y de la esquina superior izquierda	Numérico	1.00-2000.00
Elemento.nombre	Indica el nombre del conector	Alfanumérico	*
Elemento.campos[]	Array de campos para generar el WSDL	alphanumérico	*
Elemento.color	Indica el color del elemento	hexadecimal	#000000-#ffffff

Tabla 5.1: Modelo Prueba Unitaria

5.5. Diseño de pruebas

Primero se definió las pruebas unitarias de cada caso de uso, luego las pruebas de integración y finalmente las pruebas de aceptación.

5.5.1. Pruebas Unitarias

El principal objetivo que tuvieron las pruebas unitarias es el de encontrar errores en la estructura de nuestro sistema. Esto ayuda a asegurar que cada módulo funcione eficazmente por separado.

Primero se definió el formato para realizar las pruebas unitarias al momento de evaluarlas.

El formato es el que se muestra en la Figura 5.14.

Prueba Unitaria								
Fecha		Nº						
Nombre		Módulo						
Información Caso de Prueba								
Descripción:		Enfoque:						
Datos de Entrada	Datos De Salida (Esperados)							
Procedimiento								
1. 2. 3. ... n.								
Precondiciones:								
Resultados								
Resultados Obtenidos:	Estado							
	Aprobado							
	No Aprobado							
	De no aprobar especificar falla:							
	Grave							
Observaciones								

Figura 5.14: Modelo Prueba Unitaria

Los módulos a probar se explican en el Anexo C de este documento.

5.5.2. Pruebas de Integración

Las pruebas de integración se realizan una vez aprobadas las pruebas unitarias, ya que lo que permiten es comprobar la relación al unir todos los módulos ya probados de las pruebas unitarias. A continuación se presenta el esquema de integración en la Figura 5.15 y finalmente el formato de la prueba en la Figura 5.16.

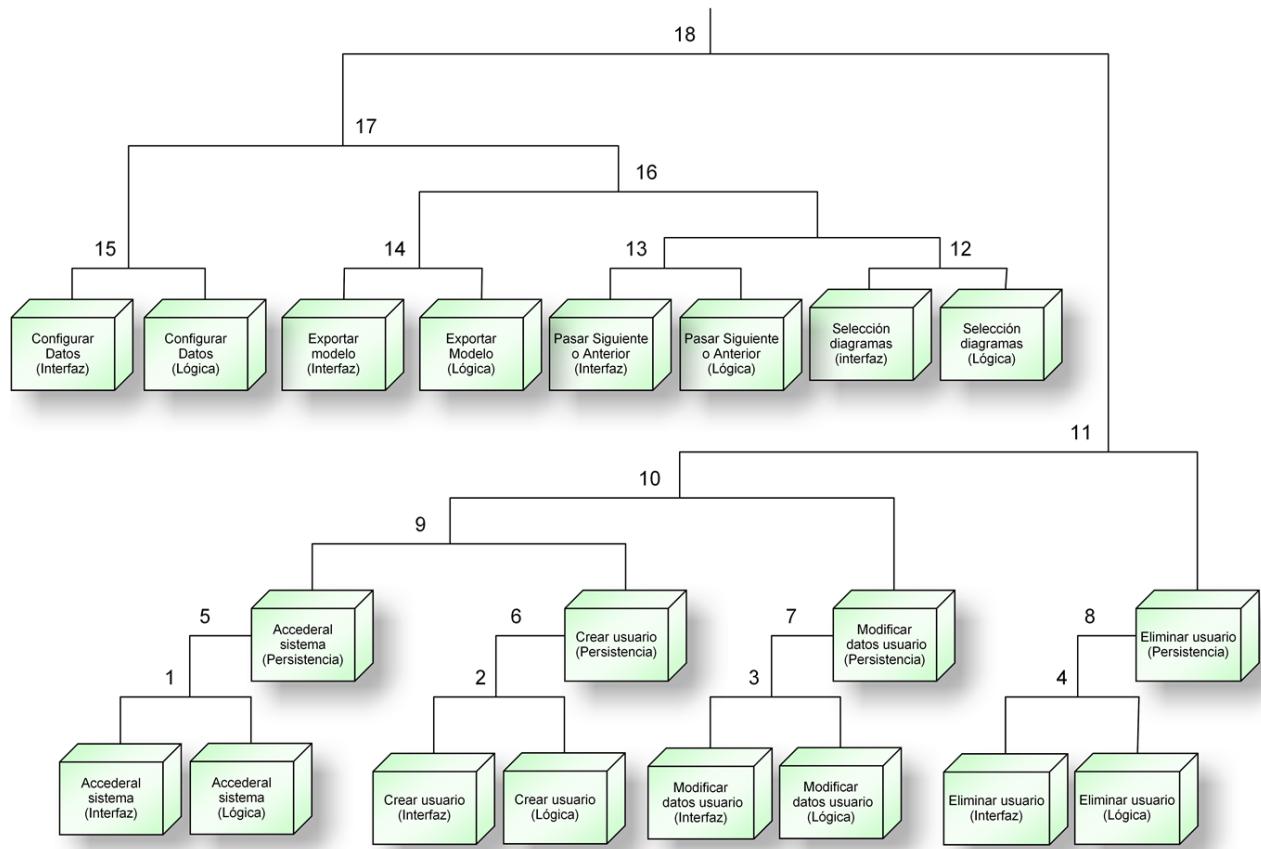


Figura 5.15: Diagrama de pruebas de integración

La Figura 5.15 muestra la integración de los subsistemas de GOSOA. Esta integración se da por los números que se van asignando a cada prueba, cada uno de los números significa la realización de las pruebas de integración.

Entre Las pruebas 1 a la 10, se integran los módulos de subsistemas entre sí. O sea, la interfaz se integra con la parte lógica y luego con la parte de persistencia.

Entre 11 y 18, se integran los módulos del modelador entre sus partes lógica y de interfaz.

El formato de las pruebas de integración se presenta en la Figura 5.16.

Prueba de Integración					
Fecha	Nº				
Nombre	Módulos				
Información Caso de Prueba de Integración					
Descripción:		Enfoque:			
Datos de Entrada		Datos De Salida (Esperados)			
Procedimiento					
1. 2. 3. ... n.					
Precondiciones:					
Resultados					
Resultados Obtenidos:	Estado				
	Aprobado				
	No Aprobado				
	De no aprobar especificar falla:				
	Grave				
	Menor				
Observaciones					

Figura 5.16: Formato pruebas de integración

5.5.3. Prueba de Aceptación

El objetivo de la prueba de aceptación es que la persona a la que está dirigida el sistema lo apruebe a través de un test de su facilidad de uso. Para esto se realiza una encuesta que ayuda a lograr este objetivo. La encuesta debe estar orientada a dos tipos de personas. El administrador del sitio y a los usuarios de éste.

Instrucciones:
Contesta la encuesta con:
N No aplica
1 Malo
2 Insuficiente
3 Regular
4 Bueno
5 Excelente

Figura 5.17: Instrucciones para la encuesta

A continuación, en las Tablas 5.2 y 5.3 se muestran las dos encuestas.

Encuesta de aceptación						
Preguntas Administrador	N	1	2	3	4	5
¿Se entienden las 3 secciones de la página?						
¿Cómo considera la búsqueda de usuarios?						
¿Encuentra que es intuitivo crear un diagrama?						
¿Qué le parece la ubicación de los elementos para modelar?						
¿Cree que es fácil poner nombres a los elementos y conectores?						
¿Qué le parece la manera de pasar entre diagramas?						
¿Le parece de fácil entendimiento como se redimensionan los elementos?						
¿El inicio de sesión es de fácil entendimiento?						
¿Sabe que significa que los modelos sean públicos?						
¿Es fácil realizar el registro?						
¿Es fácil generar los WSDL?						
Tomando en cuenta las características que usted considere necesaria póngale nota al sistema						

Tabla 5.2: Encuesta Usuario

Encuesta de aceptación						
Preguntas Usuario	N	1	2	3	4	5
¿Se entienden las 3 secciones de la página?						
¿Está bien ubicada la búsqueda de usuarios?						
¿Se ve bien los iconos de eliminar?						
¿Sabe cómo eliminar un usuario?						
¿Sabe como editar un usuario?						
¿Cree que está bien diseñada la manera de editar usuarios?						
¿Le parecen suficientes los campos que se piden en el registro?						
¿Es sencillo crear nuevos usuarios?						
¿Encuentra que el sitio es seguro?						
¿Se muestran de buena manera todos los diagramas de los usuarios?						
¿Es complicado ocultar diagramas públicos?						
Tomando en cuenta las características que usted considere necesaria póngale nota al sistema						

Tabla 5.3: Encuesta Administrador

Capítulo 6

Implementación

En este Capítulo se presenta la plataforma de desarrollo a utilizar, las herramientas necesarias tanto en software como en hardware, los lenguajes de programación y las estrategias para utilizarlos.

6.1. Plataforma de Desarrollo

El sistema GOSOA es esencialmente un sistema web, ya que permite a diferentes usuarios utilizar la aplicación desde cualquier dispositivo que tenga conexión a internet y que soporte flash.

6.2. Herramientas de Software

Adobe Flash CS4 Profesional.

El software Adobe Flash Professional CS4 es el estándar del sector para la creación y entrega interactivas de experiencias virtuales y envolventes presentadas de manera uniforme en ordenadores personales, dispositivos móviles y pantallas de prácticamente cualquier tamaño y resolución [40].

También da soporte para los distintos tipos de actionscript existentes (actionscript 1,2 y 3) y otros lenguajes como abode Air ¹.

¹Adobe AIR es una plataforma que permite crear aplicaciones que funcionen a la vez en múltiples Sistemas Operativos. Piensa en ello como una página web: se verá en Windows, Mac, Linux, etc.

Adobe Flash se utilizó para la creación del modelador con el lenguaje ActionScript 3.

NotePad++

Herramienta de edición de texto que puede trabajar con diferentes lenguajes de programación, como lo son por ejemplo:

- Ada, Asp, Assembly, Autoit, Batch, C, C#, C++, Caml, Cmake, COBOL, CSS, D, Diff, Flash actionscript, fortran, Gui4cli, hakel, HTML, INNO, Java, Javascript, Jsp, KIXtart, LISP, Lua, Makefile, matlab, MS INI file, MS-DOS style, Normal Text, NSIS, Objective-C, Pascal, Perl, PHP, Postscript, PowerShell, Properties, Python, R, Resource file, Ruby, Shell, Scheme, Smalltalk, SQL, TCL, Tex, VB, VHDL, Verilog, XML, YAML.

Además permite el uso de definiciones propias por parte del usuario. Pero los lenguajes que se utilizaron con este editor, fueron XML, HTML, PHP, javascript, SQL. [41].

MySQL 5.1

MySQL es un Administrador de Base de Datos (DBMS), el cual no otorga muchas prestaciones versus sus competidores más cercanos como Postgres y Oracle, pero cumple con las características necesarias para ser electo como DBMS de el sistema GOSOA [42].

Apache 2.1 con PHP 5.1 support

Servidor de aplicaciones web con soporte para Web Services, que permite la ejecución de archivos HTML, PHP, y Javascript que son utilizados para el desarrollo de las secciones que no pertenecen al modelador. Que son el Login, perfil de usuario, Web Services, entre otros [43].

6.3. Herramientas de Hardware

Las Herramientas de Hardware utilizadas son las siguientes:

- Servidor: El servidor utilizado es un Intel Xeon CPU 2GHz, con una memoria RAM de 4GB 300GB de disco duro y Sistema Operativo Windows Server 2008 de 32 Bits.

6.4. Lenguajes de Programación y frameworks

Los lenguajes de programación utilizados son los siguiente:

HTML

Es el lenguaje de marcado predominante para la elaboración de páginas web. Es usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes [44].

PHP

Es un lenguaje de programación interpretado, diseñado para la creación de páginas web dinámicas. Se usa principalmente para la interpretación del lado del servidor y para realizar la conexión con la base de datos [45].

javascript

JavaScript es un lenguaje de scripting basado en objetos no tipeado y liviano, utilizado para acceder a objetos en aplicaciones. Principalmente, se utiliza integrado en un navegador web permitiendo el desarrollo de interfaces de usuario mejoradas y páginas web dinámicas. Se utiliza para las validaciones en el sitio aprovechando que se ejecuta en el lado del cliente [46].

SQL

el lenguaje de consulta estructurado o SQL es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones en éstas. Lo utilizamos para acceder a la informaciones en las bases de datos del sistema GOSOA [42].

Actionscript 3

Adobe ActionScript es el lenguaje de programación de la Plataforma Adobe Flash. La programación con ActionScript permite mucha mas eficiencia en las aplicaciones de la plataforma Flash para construir animaciones de todo tipo, desde simples a complejas, ricas en datos e interfaces interactivas. Es por ello que lo utilizamos en el modelador [47].

Y los frameworks utilizados son:

JQuery

Es un framework de javascript que permite simplificar el manejo de documentos html. Puede manejar eventos, objetos etc [48].

amfphp

Es un framework que nos permite comunicar diferentes lenguajes de programación con php, a través de Web Services [49].

6.5. Estrategias de Implementación

La estrategia de implementación adoptada es incremental, desarrollando cada subsistema descrito por la Figura 6.1, donde se observa el esquema de integración de estos.

La implementación se genera en el orden que describen los números en la Figura 6.1.

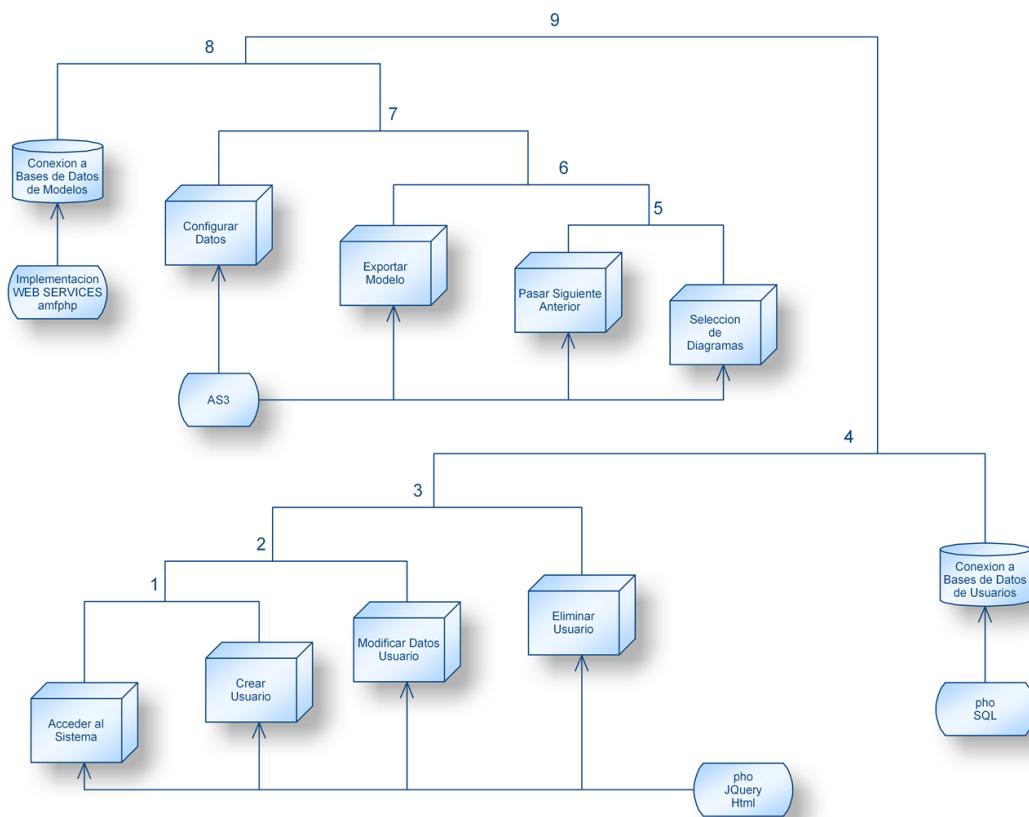


Figura 6.1: Diagrama de Integración de la implementación

A continuación se explica el diagrama y se muestran pantallas del sistema. Para más información sobre las pantallas del sistema revisar Anexo B

1. Se implementan los subsistemas Acceder al Sistema y Crear Usuario y luego se integran entre sí. Acceder al sistema se muestra en la Figura 6.2 y crear usuario en la Figura 6.3.

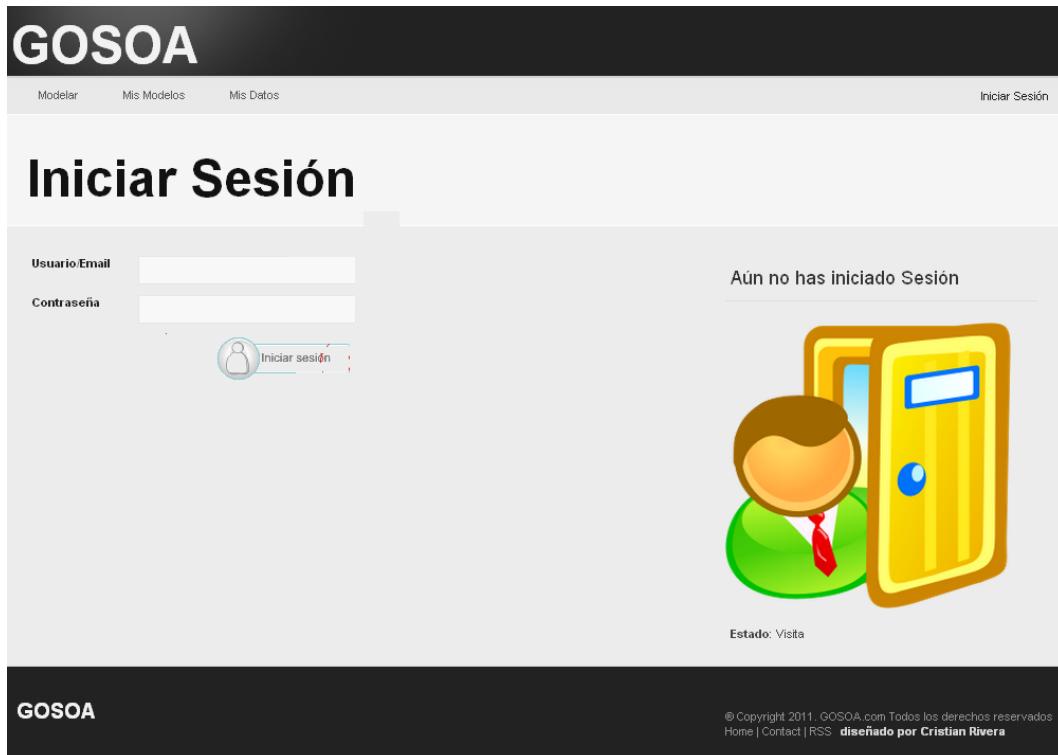


Figura 6.2: Imagen accediendo al sistema



Figura 6.3: Imagen creando usuario

2. Se implementa el sistema modificar usuario y se integra al sistema que existe hasta el momento. La pantalla Modificar Usuario se muestra en la Figura 6.4

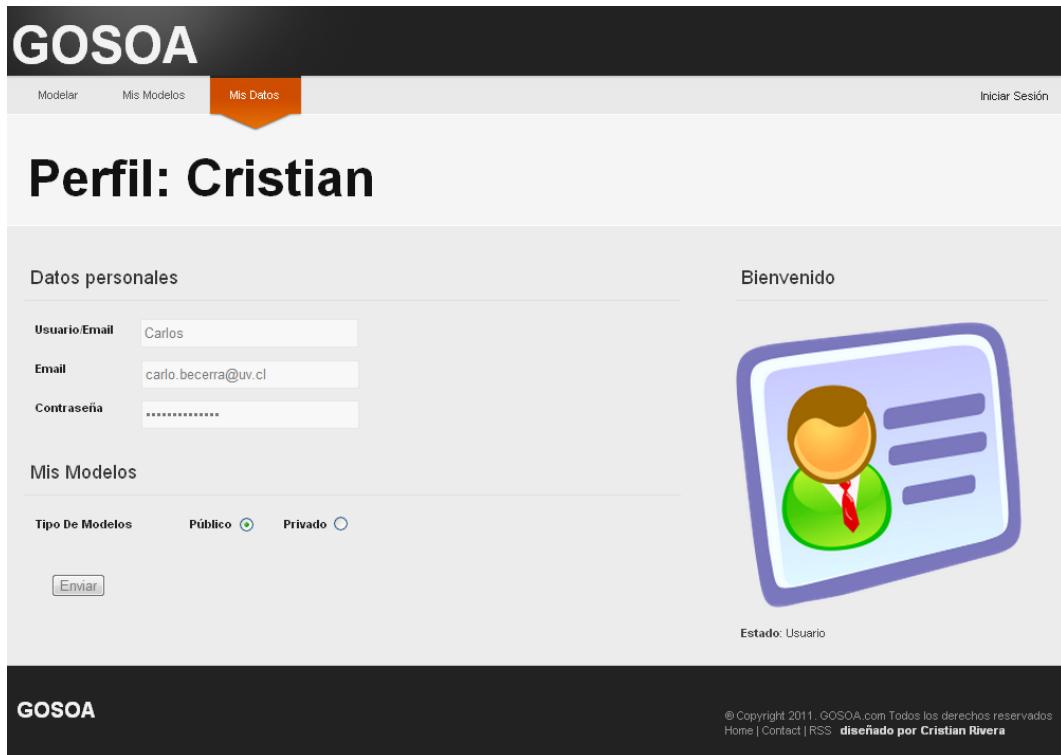


Figura 6.4: Imagen de modificación de usuarios

3. Se implementa el sistema de eliminación de usuarios y se integra al sistema que existe hasta el momento. La eliminación se de usuario se muestra en la Figura 6.5 en la Marca “C”.

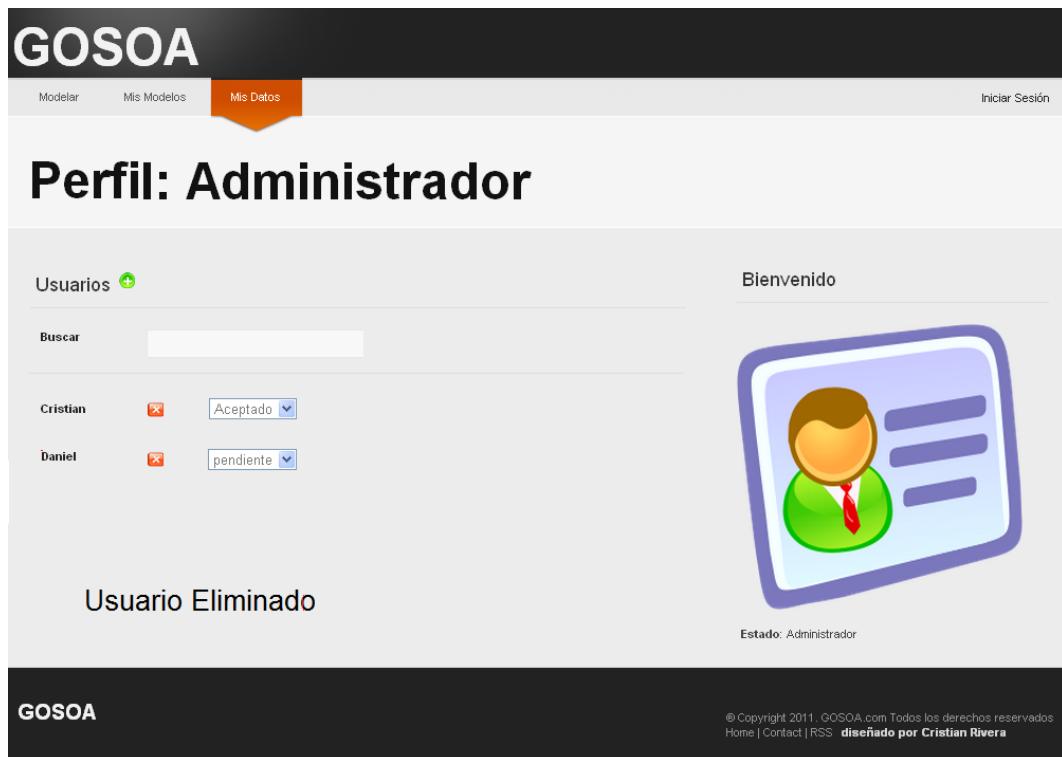


Figura 6.5: Imagen de eliminación de usuarios

4. Finalmente se integra estos sistemas con la utilizacion de las bases de datos
5. Se crean los sistemas de y selección de diagramas y de pasar al siguiente diagrama. Luego se integran entre sí. El sistema de navegacióin entre diagramas se muestra en la Figura 6.6, En la Marca “A” se retrocede en el diagrama y en la Marca “B” se avanza en el diagrama.

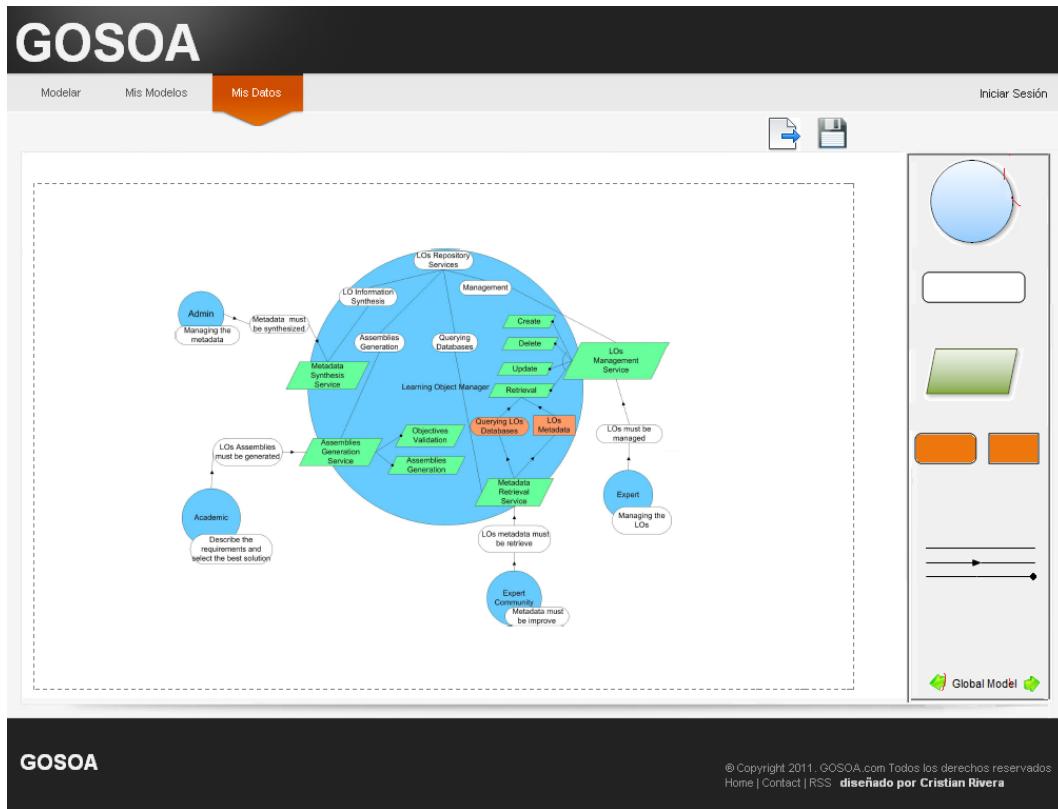


Figura 6.6: Imagen de Selección de diagramas

6. Se crea el sistema de exportar modelo y se integra a los sistemas anteriores. La exportación de modelos se muestra en la Figura 6.7 Marca “B”.

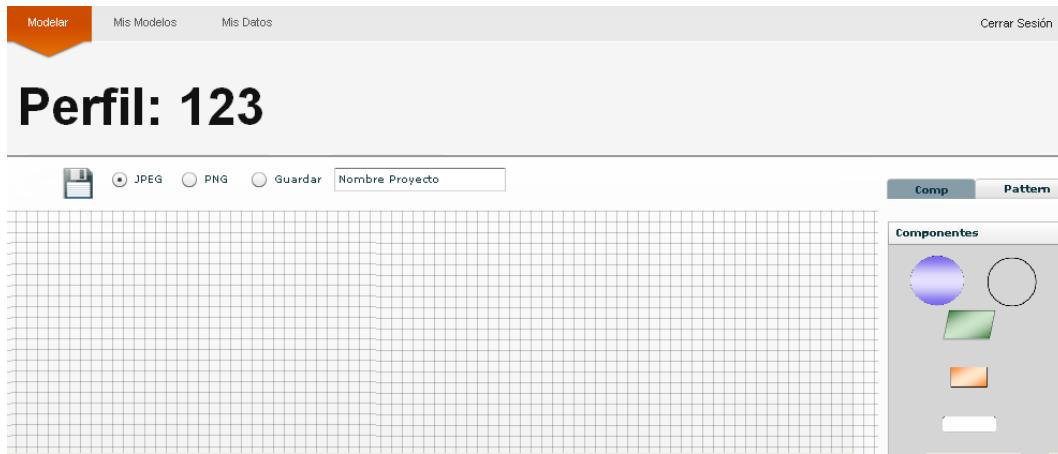


Figura 6.7: Imagen de exportación del modelo

7. Se crea el sistema de configurar datos y se integra a los sistemas anteriores. El sistema de configuración de datos se muestra en la Figura 6.8.

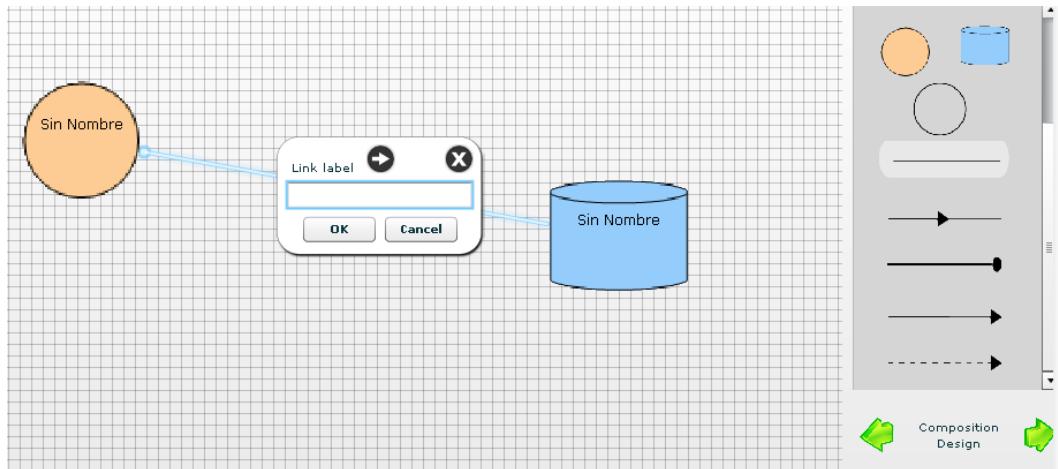


Figura 6.8: Imagen de Configuración de datos

8. Se Conectan estos sistemas a la base de datos a través de Web Services.
9. Se integran el sistema del modelador junto con el sistema de navegación de páginas y se crea el sistema final.

Capítulo 7

Pruebas

Las pruebas en este capítulo están enfocadas en la detección de errores y aceptación del sistema GOSOA.

Las pruebas se dividen en:

- Pruebas Unitarios
- Pruebas de Integración
- Pruebas de Aceptación

Ya en la fase de implementación se comienza con las pruebas unitarias del sistema junta a sus correspondientes pruebas de integración.

7.1. Pruebas Unitarias

Las pruebas unitarias que se realizaron a GOSOA se enfocan a encontrar errores en el código utilizando caja negra, donde miramos el sistema como una caja en donde no conocemos su contenido, solo las entradas y las salidas esperadas.

Los sistemas a probar son los siguientes.

- Interfaz GOSOA
 - Acceder al Sistema Interfaz
 - Crear Usuario Interfaz
 - Modificar datos de Usuario Interfaz

- Eliminar Usuario Interfaz
- Lógica GOSOA
 - Acceder al Sistema Lógica
 - Crear Usuario Lógica
 - Modificar datos de Usuario Lógica
 - Eliminar Usuario Lógica
- Persistencia GOSOA
 - Acceder al Sistema Persistencia
 - Crear Usuario Persistencia
 - Modificar datos de Usuario Persistencia
 - Eliminar Usuario Persistencia

Por otro lado tenemos las pruebas unitarias del modelador.

- Interfaz Modelador
 - Pasar Siguiente/anterior Interfaz
 - Exportar Modelo Interfaz
 - Configurar Datos Interfaz
- Lógica Modelador
 - Pasar Siguiente/anterior Lógica
 - Exportar Modelo Lógica
 - Configurar Datos Lógica

El detalle de las pruebas realizadas se puede visualizar en el enexo C.

A continuación se muestra un resumen del resultado de las pruebas unitarias

Nº	Tipo de Prueba	Nombre Módulo	Resultado
1	Interfaz	Acceder al sistema	APROBADO
2	Interfaz	Crear usuario	APROBADO
3	Interfaz	Modificar datos usuarios	APROBADO
4	Interfaz	Eliminar usuario	APROBADO
5	Lógica	Acceder al sistema	APROBADO
6	Lógica	Crear usuario	APROBADO
7	Lógica	Modificar datos usuarios	APROBADO
8	Lógica	Eliminar usuario	APROBADO
9	Persistencia	Acceder al sistema	APROBADO
10	Persistencia	Crear usuario	APROBADO
11	Persistencia	Modificar datos usuarios	APROBADO
12	Persistencia	Eliminar usuario	APROBADO
13	Interfaz	Pasar siguiente/anterior	APROBADO
14	Lógica	Pasar siguiente/anterior	APROBADO
15	Interfaz	Exportar modelo	APROBADO
16	Lógica	Exportar modelo	APROBADO
17	Interfaz	Configurar Datos	APROBADO
18	Lógica	Configurar Datos	APROBADO

Figura 7.1: Resumen de las pruebas unitarias

7.1.1. Análisis de Resultados

El detalle de los resultados obtenidos se puede visualizar en el Anexo C. Aquí podemos ver que todos los resultados son exitosos , ya que aquellos que presentaron problemas en esta fase fueron solucionados y corregidos en el momento que se detecto el defecto.

7.2. Pruebas de Integración

Las pruebas de integración se desarrollan según el modelo presentado en la Figura 5.15 de la sección 5.5.2. Los resultados de cada prueba se puede visualizar en el Anexo D.

A continuación se detalla el resumen de las pruebas de integración con la Figura 7.2.

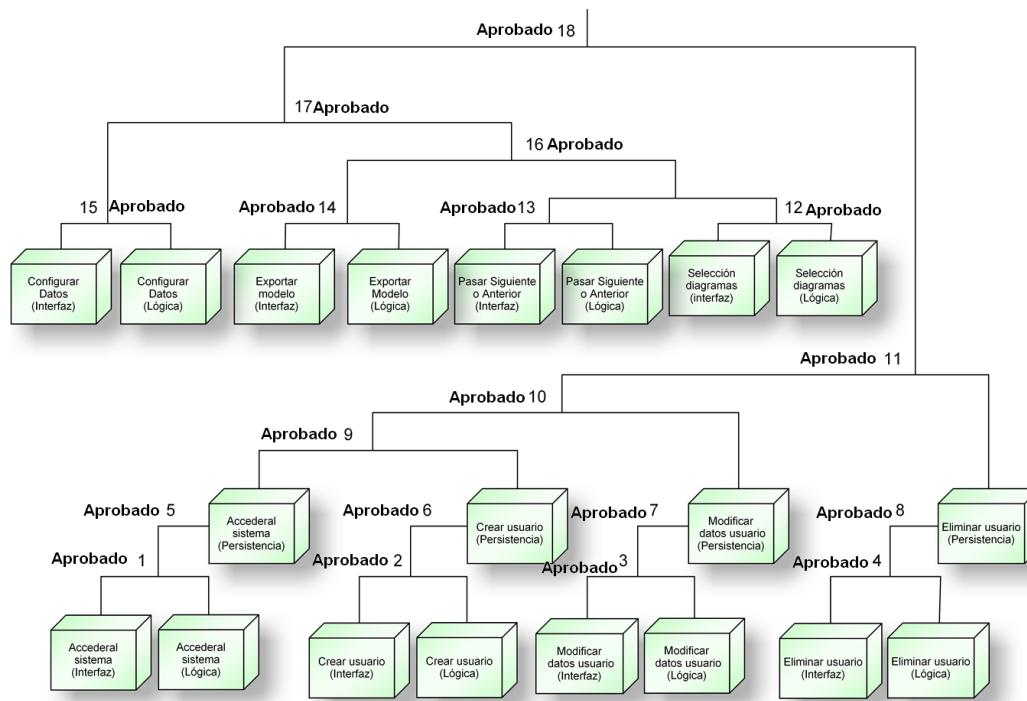


Figura 7.2: Resultados de Integración

7.2.1. Resultados de Pruebas de Integración

Los resultados de las pruebas de integración se pueden ver en el Anexo D. Los errores fueron solucionados y los resultados que ahí aparecen son los finales.

El principal conflicto en la integración se presentó en la conexión entre la aplicación en flash y php, ya que los mensajes estaban en distinto formato.

7.3. Pruebas de Aceptación

Para las pruebas de aceptación se realizaron las encuestas que se habían propuesto en la sección de diseño, capítulo 5. Cabe destacar que las pruebas fueron realizadas a través de un sistema Online Ubicado en la misma página web junto a un tutorial para que las personas lo pudiesen seguir y así después contestar las preguntas.

Se realizaron 11 preguntas por cada encuesta (usuario y administrador), a un total de 13 usuarios los cuales contestaron cada una de las preguntas en una escala de 0 a 5, donde 0 es el peor puntaje y 5 el mejor. Donde el resultado mínimo esperado es un 3.5 que corresponde al 70 % de aprobación.

Las pruebas se enfocaron en el sistema analizado desde el punto de vista del usuario y el administrador.

7.3.1. Encuesta usuarios

A continuación se muestra el resumen final de las preguntas realizadas para el perfil de usuario. El detalle de las respuesta se encuentra en el Anexo E. Para la representación de los gráficos elegimos la siguiente nomenclatura:

- Color Azul: Es el puntaje que cada usuario dio a la pregunta.

El resumen de las respuestas de los usuarios se muestra en la Figura 7.3.

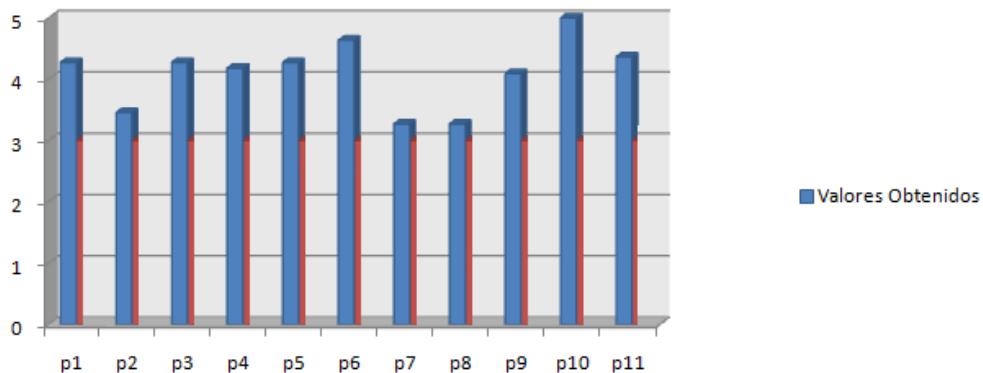


Figura 7.3: Resumen de Preguntas de los usuarios

Podemos concluir que los resultados obtenidos fueron exitosos ya que estuvieron sobre el resultado mínimo esperado de 70 % en su mayoría. Además se puede decir que el porcentaje final de aprobación estuvo sobre el 82 %.

También podemos determinar la mejor y peor pregunta, que fueron:

- Peor Pregunta:

- ¿Se entiende que significa que los modelos sean públicos o privados?
 - Tuvo una valoración promedio de 3.5 lo que equivale a un 70 % de aprobación.
 - Los usuarios no entendieron que era un modelo público y privado hasta que se les explico de que se trataba.
- Mejor Pregunta:
- ¿Evalué la Simplicidad de la realización del registro?
 - Tuvo una valoración promedio de 4.8 lo que equivale a un 96 % de aprobación.

7.3.2. Encuesta Administrador

A continuación se muestra el resumen final de las preguntas realizadas para el perfil de Administrador. El detalle de las respuesta se encuentra en el Anexo E. Para la representación de los gráficos elegimos la siguiente nomenclatura:

- Color Azul: Es el puntaje que cada usuario dio a la pregunta.

El resumen de las respuestas de los administradores se en la Figura E.24.

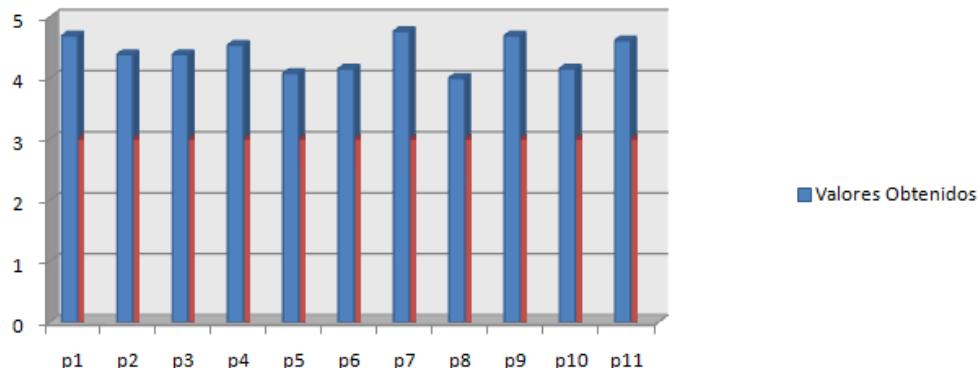


Figura 7.4: Resumen de Preguntas de La sección Administrador

Podemos concluir que los resultados obtenidos fueron exitosos ya que estuvieron sobre el resultado esperado de cada pregunta. También podemos determinar la mejor y peor pregunta, que fueron:

- Peor Pregunta:

- ¿Le parece bien los campos que se le exigen al usuario para el registro (enfoque seguridad)?
 - Tuvo una valoración promedio de 4.0 lo que equivale a un 80 % de aprobación.
 - Los usuarios pensaron que los campos pedidos eran insuficientes y que se debiesen agregar otros como Nombre, Apellido, fecha de nacimiento, etc.
- Mejor Pregunta:
- ¿Evalué la gestión de usuarios (buscar, crear, eliminar, editar)?
 - Tuvo una valoración promedio de 4.8 lo que equivale a un 96 % de aprobación.

Capítulo 8

Implantación

En el capítulo de implantación se detalla el procedimiento por el cual es sistema queda alojado y funcionando en cristian.my-place.us o cristianrivera.com/tesis.

En primer lugar se deben instalar (o deben estar instalados) los siguientes servicios en el servidor:

- php versión 5.0 ó superior
- Mysql version 5.0 ó superior

Y el cliente debe tener instalado Filezilla para poder subir a través de ftp los archivos. Lo que se debe hacer es:

- Se necesitan las claves de acceso de los respectivos hosting.
 - Claves de Usuario y pass del gestor de archivos ftp.
 - Claves de Usuario y pass del panel de control del Hosting
 - Claves de Usuario y pass del gestor de bases de datos (en este caso MySQL)
- Se crea el espacio, que no debe ser menor a las 50 MegaBytes, y las bases de datos correspondientes desde el panel de control del hosting.
- Los archivos se encuentran alojados en www.cristianrivera.com/tesis/archivos.zip
- Las bases de datos se encuentran en (*Dump*) www.cristianrivera.com/tesis/GOSOA.sql
- A través del gestor ftp se suben los archivos al hosting

- Se escriben las claves de acceso a la base de datos en todos los archivos que lo necesiten.

Al subir los archivos y al hacer el dump a la base de datos el sistema queda listo para ser accedido desde cualquier lugar.

Capítulo 9

Conclusiones

El sistema GOSOA cumple con los objetivos propuestos en este trabajo de título. Por una parte permite la creación de diagramas que dan soporte a la metodología que pasa requerimientos en i* a Arquitecturas Orientadas a Servicios y por otra parte permite el fácil acceso a la herramienta ya que esta se encuentra alojada en un sitio web, lo que permite que puede ser accedida desde cualquier lugar y en cualquier momento sólo dependiendo de una conexión a internet. También el Sitio provee una gestión de usuarios lo cual permite que cada uno de ellos almacenar y compartir los diagramas que generen.

Se han implementado una serie de algoritmos para poder modelar objetos en AS3 (como realizar líneas en la pantalla, mover objetos, insertar texto, entre otros), que están disponibles en el código fuente de este trabajo de título, como también nuevas técnicas de dibujo de estos objetos.

La herramienta en si permite a usuarios generar especificaciones en i*, para así a través de una serie de modelos llegar a un sistema Orientado a Servicios, lo cual es muy útil hoy en día ya que la mayoría de las empresas conocen los beneficios de este tipo de arquitecturas y la adoptan por su simplicidad y gran interoperabilidad.

El sistema fue implementado con AS3 para poder utilizar FLASH y así poder ejecutarlo en cualquier navegador existente en este momento. Además se utiliza el mismo principio de la Arquitectura orientada a servicio y se usan web services para la comunicación entre los distintos lenguajes implementados en el sistema. Para esta implementación se uso AMFPHP que permite la comunicación entre Web Services de flex o aplicaciones flash a php, el cual lo utilizamos para guardar los modelos de los usuarios en la base de datos y recuperarlos posteriormente.

Finalmente realizados las pruebas para identificar el correcto funcionamiento del sitio. Además las encuesta dieron como resultados una aprobación para el perfil de usuario de un 82 % y para el perfil de administrador de un 80 %.

Finalmente el sistema ha sido liberado y funcionando.

Los trabajos futuros son variados ya que abre una gama de posibilidades nueva de trabajos. Un trabajo fundamental sería la conversión del sitio a html5 por el incierto futuro de flash como interprete Web. Otro trabajo sería la conversión automatizada entre los modelos, porque en la actualidad el usuario redibuja parte de los modelos anteriores en los modelos siguientes, por lo que sería bueno que el usuario solo eligiese los elementos a replicar y el sistema lo hiciera en forma automática. Otro trabajo futuro que se puede esperar de la aplicación es la creación de una aplicación que no se encuentre en línea, en caso de que los usuarios lo necesiten. Además al crear esta versión fuera de escritorio se debe crear una manera de guardar los modelos con algún formato xml o similar. Porque en este momento se guardan directamente en la base de datos y no existe algún formato específico para guardar estos en forma local.

Bibliografía

- [1] Magister Magazine. Último acceso 26 de marzo 2011.
<http://www.mastermagazine.info/articulo/3391.php>.
- [2] Becerra C y Franch X y Astudillo H, *From i* Models to Service Oriented Architecture models*. Architectures, Concepts and Technologies for Service Oriented Computing, 2010.
- [3] Estandares SOA según la W3C, “Último acceso 05 de abril 2012.
<http://www.w3c.es/eventos/2007/diaw3c/presentaciones/soayestandares.pdf>.”
- [4] Bell M, *Introduction to Service-Oriented Modeling*. Wiley & Sons, 2008. Service-Oriented Modeling: Service Analysis, Design, and Architecture.
- [5] Organización internacional OASIS, “Último acceso 05 de abril 2012.
http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=soa-rm.”
- [6] W3C SOAP specification, “Último acceso 18 de abril 2011.
http://www.w3.org/tr/overview#tr_soap.”
- [7] Que es web services según la W3C, “Último acceso 05 de abril 2012.
<http://www.w3.org/2002/ws-desc>.”
- [8] Sitio Especificacion W3C para JavaScript, “Último acceso 17 de diciembre 2011.
<http://www.w3.org/standards/techs/js>.”
- [9] XML, “Último acceso 05 de abril 2012. <http://www.xml.com>.”
- [10] WSDL, “Último acceso 05 de abril 2012. <http://www.w3.org/2002/ws-desc>.”
- [11] WSEL, “Último acceso 05 de abril 2012. http://www.w3.org/2001/04/ws-ws-proceedings/rod_smith/img13.htm.”
- [12] WSPF, “Último acceso 05 de abril 2012. <http://msdn.microsoft.com/es-es/netframework/aa663324.aspx>.”

- [13] WSME, “Último acceso 24 de abril 2011. http://msdn.microsoft.com/eses/webservices/understanding/spec_metadataexchange.asp.”
- [14] WSA, “Último acceso 05 de abril 2012. <http://www.w3.org/submission/ws-addressing/>.”
- [15] SOAP, “Último acceso 05 de abril 2012. <http://www.w3.org/tr/2000/note-soap-20000508>.”
- [16] Sitio Oficial Sybase PowerDesigner, “Último acceso 05 de abril 2012. <http://www.sybase.com/products/modelingdevelopment/powerdesigner>.”
- [17] Sitio Oficial Rational Software Architect, “Último acceso 05 de abril 2012. <http://www.rational.com.ar>.”
- [18] Sitio Oficial Registry and Repository, “Último acceso 05 de abril 2012. <http://www-01.ibm.com/software/integration/wsrr/>.”
- [19] Sitio Oficial Rational RequisitePro, “Último acceso 05 de abril 2012. <http://www-01.ibm.com/software/awdtools/reqpro/>.”
- [20] Sitio Oficial WebSphere Business Modeler, “Último acceso 05 de abril 2012. <http://www-01.ibm.com/software/integration/wbimodeler/advanced/features/>.”
- [21] Yu E, *Social modeling and i**. Faculty of Information, University of Toronto, Toronto, Canada M5S 3G6, 2000.
- [22] Yu E, *Strategic actor relationships modelling with i**. Faculty of Information, University of Toronto, Toronto, Canada M5S 3G6, 2001.
- [23] Página oficial de i* con sus herramientas, “Último acceso 05 de abril 2012. http://istar.rwth-aachen.de/tiki-index.php?page_ref_id=21.”
- [24] Página oficial OpenOme, “Último acceso 05 de abril 2012. <https://se.cs.toronto.edu/trac/ome/>.”
- [25] Página oficial Ome, “Último acceso 22 de abril 2011. <http://istar.rwth-aachen.de/tiki-index.php?page=ome>.”
- [26] Página oficial REDEPEND.REACT-BCN, “Último acceso 22 de abril 2011. http://www.lsi.upc.edu/sim_ggrau/redepend-react/.”
- [27] Página proyecto Tropos, “Último acceso 05 de abril 2012. <http://www.troposproject.org/>.”

- [28] Página herramienta taom4e, “Último acceso 22 de abril 2011. <http://sra.itc.it/tools/taom4e/>.”
- [29] Página herramienta GR-Tool, “Último acceso 22 de abril 2011. <http://sesa.dit.unitn.it/goaleditor/>.”
- [30] Página herramienta T-Tool, “Último acceso 22 de abril 2011. http://dit.unitn.it/sim_ft/ft_tool.html.”
- [31] Página herramienta ST-Tool, “Último acceso 22 de abril 2011. <http://sesa.dit.unitn.it/sttool/index.php>.”
- [32] Página herramienta J-PRIM, “Último acceso 22 de abril 2011. http://www.lsi.upc.edu/sim_ggrau/jprim/.”
- [33] Página herramienta jUCMNav, “Último acceso 05 de abril 2012. <http://jucmnav.softwareengineering.ca/jucmnav/>.”
- [34] Página herramienta DesCARTES, “Último acceso 05 de abril 2012. <http://www.isys.ucl.ac.be/descartes>.”
- [35] Sitio Oficial Dia, “Último acceso 05 de abril 2012. <http://projects.gnome.org/dia/>.”
- [36] Sitio Oficial Visio, “Último acceso 05 de abril 2012. <http://office.microsoft.com/en-us/visio/>.”
- [37] Grau G. y Franch X., *On the Adequacy of i* Models for Representing and Analyzing Software Architectures*. Wiley & Sons, 2007, pages 296-305. Advances in Conceptual Modeling Foundations and Applications.
- [38] Han J, A *Comprehensive Interface Definition Framework for Software Components*. WPrentice Hall/PearsonPTR. APSEC 98: Proceedings of the Fifth Asia Pacific Software Engineering Conference 1998, IEEE Computer Society.
- [39] Thomas Erl, *SOA Design Patterns*. WPrentice Hall/PearsonPTR,, 2009. Upper Saddle River, NJ, USA.
- [40] Adobe Flash CS4. Último acceso 05 de abril 2012. <http://www.adobe.com>.
- [41] Sitio Oficial Notepadpp, “Último acceso 05 de abril 2012. <http://notepad-plus-plus.org/>.”
- [42] Sitio Especificacion MySql, “Último acceso 05 de abril 2012. <http://www.mysql.com/>.”

- [43] Sitio Oficial Apache, “Último acceso 05 de abril 2012. [http://www.apache.org/”](http://www.apache.org/)
- [44] Sitio Oficial w3 para Html, “Último acceso 05 de abril 2012. [http://www.w3.org/markup/”](http://www.w3.org/markup/)
- [45] Sitio Oficial PHP, “Último acceso 17 de diciembre 2011. [http://www.php.net/”](http://www.php.net/)
- [46] Sitio Especificacion W3C para JavaScript, “Último acceso 05 de abril 2012. [http://www.w3.org/standards/techs/js/”](http://www.w3.org/standards/techs/js/)
- [47] Sitio AS3, “Último acceso 17 de diciembre 2011. [http://as3.es/”](http://as3.es/)
- [48] Sitio Oficial JQuery, “Último acceso 05 de abril 2012. [http://jquery.com/”](http://jquery.com/)
- [49] Sitio Oficial Proyecto AMFPHP, “Último acceso 05 de abril 2012. [http://www.silexlabs.org/amfphp/”](http://www.silexlabs.org/amfphp/)

Apéndice A

Casos de Uso extendidos, Diagramas de secuencia y Diagramas de estados

A.1. Caso de Uso extendido de Acceso al Sistema

Vemos el caso de uso extendido.

APÉNDICE A. CASOS DE USO EXTENDIDOS, DIAGRAMAS DE SECUENCIA Y DIAGRAMAS DE ESTADO

Nombre Caso de Uso	Acceso al Sistema
Actores	Usuario
Propósito	Acceder al sistema
Resumen	El usuario debe iniciar una sesión para poder acceder al sistema
Tipo	Opcional
Referencias Cruzadas	
Curso Normal (Usuario)	Curso Normal (Sistema)
El usuario ingresa al sistema	El sistema pide los datos de inicio de sesión
El usuario ingresa los datos del sistema	El sistema Valida el usuario
El usuario accede al sistema	El sistema muestra el perfil del usuario o el administrador según corresponda
Curso Alternativo (Usuario)	Curso Alternativo (Sistema)
El usuario ingresa entra la sitio	El sistema le pide los datos de acceso al sistema
El usuario ingresa datos	el sistema no puede validar los datos ya que estos son incorrectos y muestra una pantalla de inicio de sesión inválido

Tabla A.1: Caso de Uso extendido de Acceso al Sistema

A.1.1. Diagrama de Flujo Acceso al Sistema

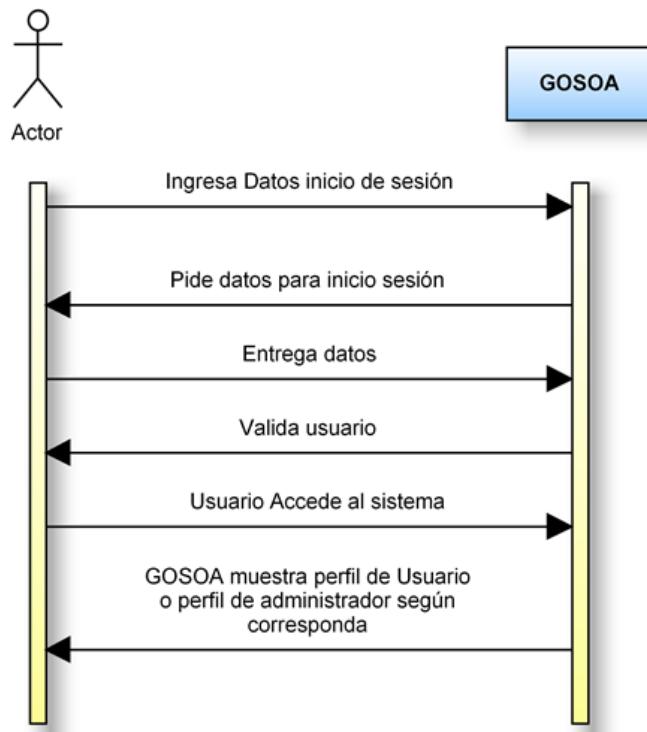


Figura A.1: Diagrama de secuencia acceso del usuario al sistema

APÉNDICE A. CASOS DE USO EXTENDIDOS, DIAGRAMAS DE SECUENCIA Y DIAGRAMAS DE ESTADO

A.1.2. Diagrama de Estado Acceso al Sistema

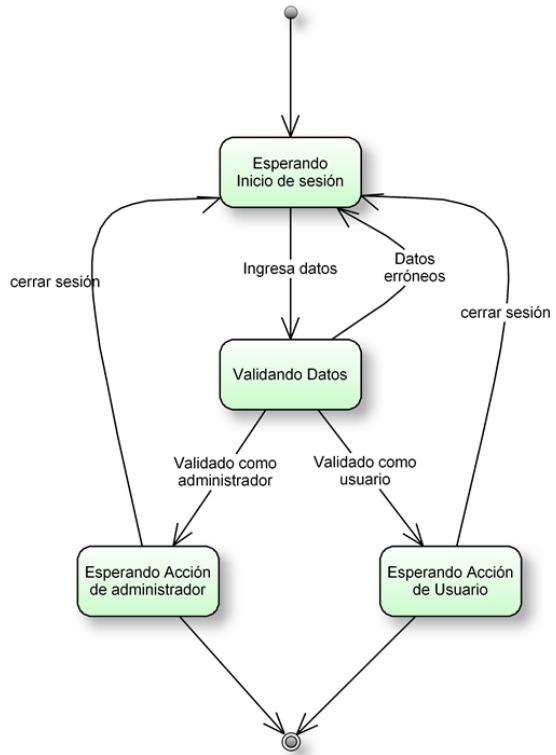


Figura A.2: Diagrama de Estado acceso del usuario al sistema

APÉNDICE A. CASOS DE USO EXTENDIDOS, DIAGRAMAS DE SECUENCIA Y DIAGRAMAS DE ESTADO

A.2. Caso de Uso Extendido Crear Usuarios

Vemos el caso de uso extendido.

Nombre Caso de Uso	Gestión de Usuarios
Actores	Usuario
Propósito	Crear Usuario
Resumen	El administrador debe poder crear, eliminar, modificar y buscar a los usuarios
Tipo	Opcional
Referencias Cruzadas	
Precondiciones	El Administrador debe estar autenticado en el sistema
Curso Normal (Usuario)	Curso Normal (Sistema)
El usuario accede a la opción crear usuario	El sistema despliega un formulario para que se ingresen los datos del nuevo usuario
El usuario ingresa los datos en el sistema y presiona la confirmación de éstos	El sistema muestra un mensaje de operación realizada exitosamente
Curso Alternativo (Usuario)	Curso Alternativo (Sistema)
El usuario accede a la opción crear usuario	El sistema despliega un formulario para que se ingresen los datos del nuevo usuario
El usuario ingresa los datos en el sistema y presiona la confirmación de éstos	El sistema muestra un error por problema de conexión a la base de datos

Tabla A.2: Caso de Uso extendido de Crear Usuario

A.2.1. Diagrama de Flujo Crear Usuario

Podemos ver la en el diagrama de secuencia el flujo de la información.

APÉNDICE A. CASOS DE USO EXTENDIDOS, DIAGRAMAS DE SECUENCIA Y DIAGRAMAS DE ESTADO

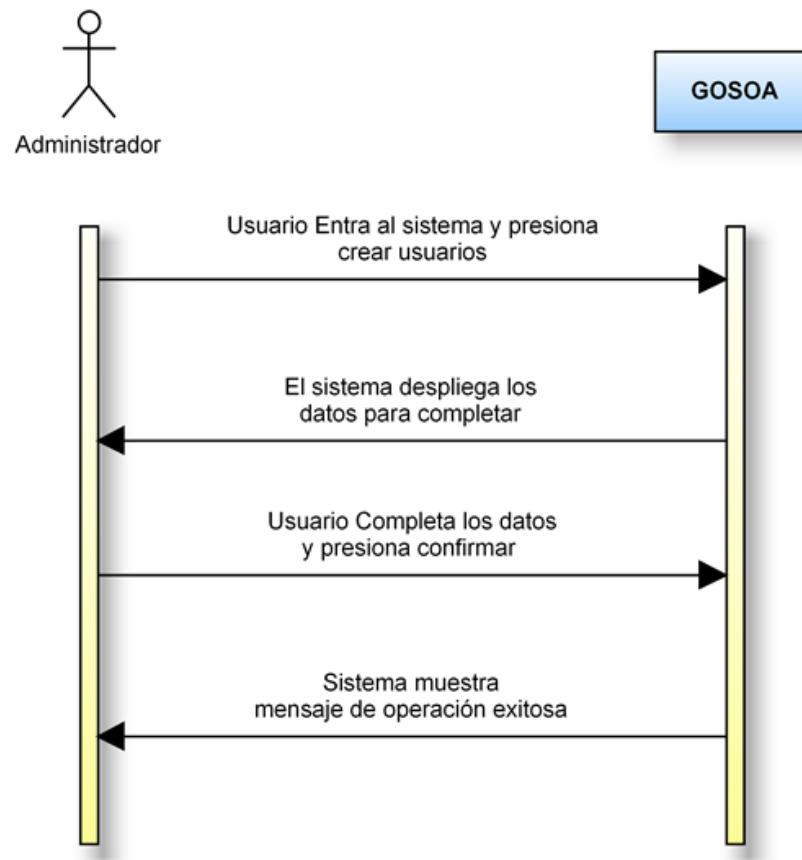


Figura A.3: Diagrama de Secuencia Crear Usuario

A.2.2. Diagrama de Estado Crear Usuario

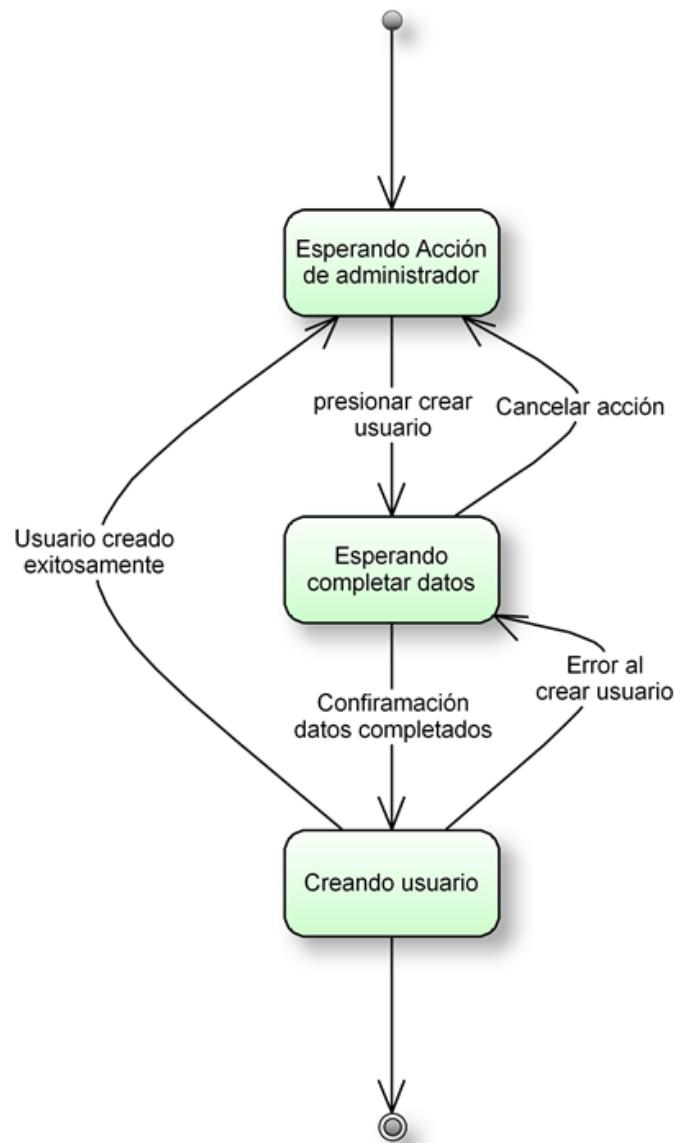


Figura A.4: Diagrama de Estado Crear Usuario

APÉNDICE A. CASOS DE USO EXTENDIDOS, DIAGRAMAS DE SECUENCIA Y DIAGRAMAS DE ESTADO

A.3. Caso de Uso extendido Modificar Usuario

Nombre Caso de Uso	Gestión de Usuarios
Actores	Usuario
Propósito	Modificar datos de usuario
Resumen	El administrador debe poder crear, eliminar, modificar y buscar a los usuarios
Tipo	Opcional
Referencias Cruzadas	
Precondiciones	El Administrador debe estar autenticado en el sistema
Curso Normal (Usuario)	Curso Normal (Sistema)
El usuario accede a la opción buscar usuario	
El usuario escribe el nombre en el campo o lo busca entre la lista de usuarios	El sistema despliega un campo para buscar y el listado de los usuarios
El usuario selecciona modificar	El sistema despliega los datos del usuario seleccionado junto a las opciones de modificar y eliminar
El usuario cambia los datos necesarios y confirma la edición	El sistema despliega los datos del usuario seleccionado en campos editables
	El sistema muestra un mensaje de operación exitosa
Curso Alternativo (Usuario)	Curso Alternativo (Sistema)
El usuario accede a la opción buscar usuario	El sistema muestra un error de conexión a la base de datos

Tabla A.3: Caso de Uso extendido Modificar Usuario

APÉNDICE A. CASOS DE USO EXTENDIDOS, DIAGRAMAS DE SECUENCIA Y DIAGRAMAS DE ESTADO

A.3.1. Diagrama de Flujo Modificar Usuario

Podemos ver la en el diagrama de secuencia el flujo de la información.

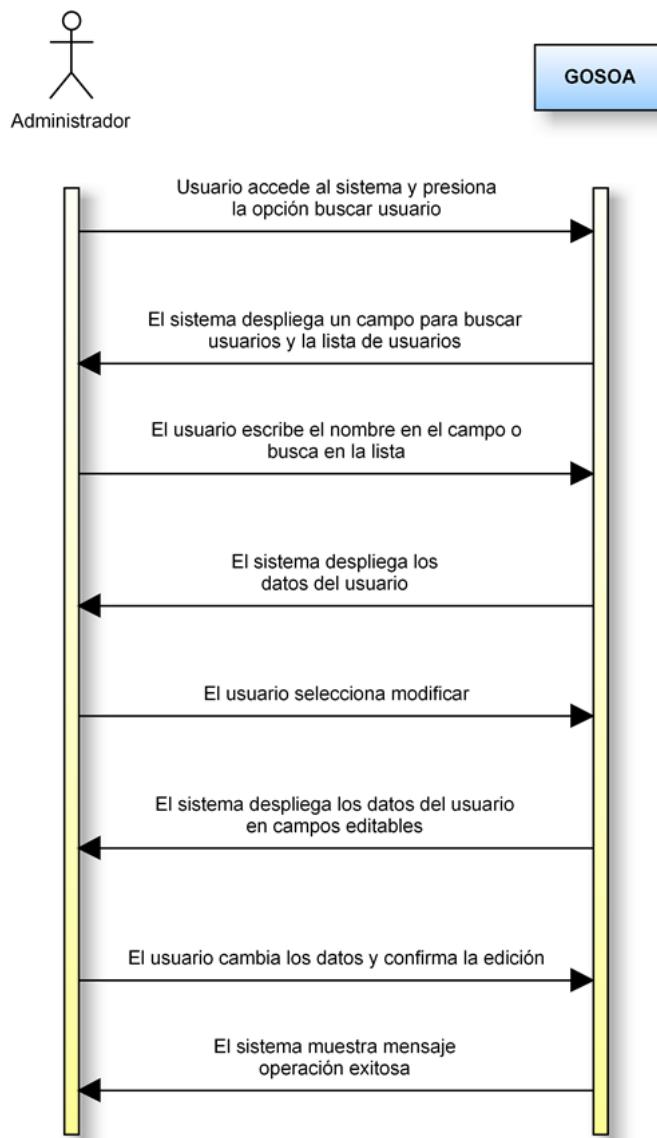


Figura A.5: Diagrama de secuencia Modificar Usuario

A.3.2. Diagrama de Estado Modificar Usuario

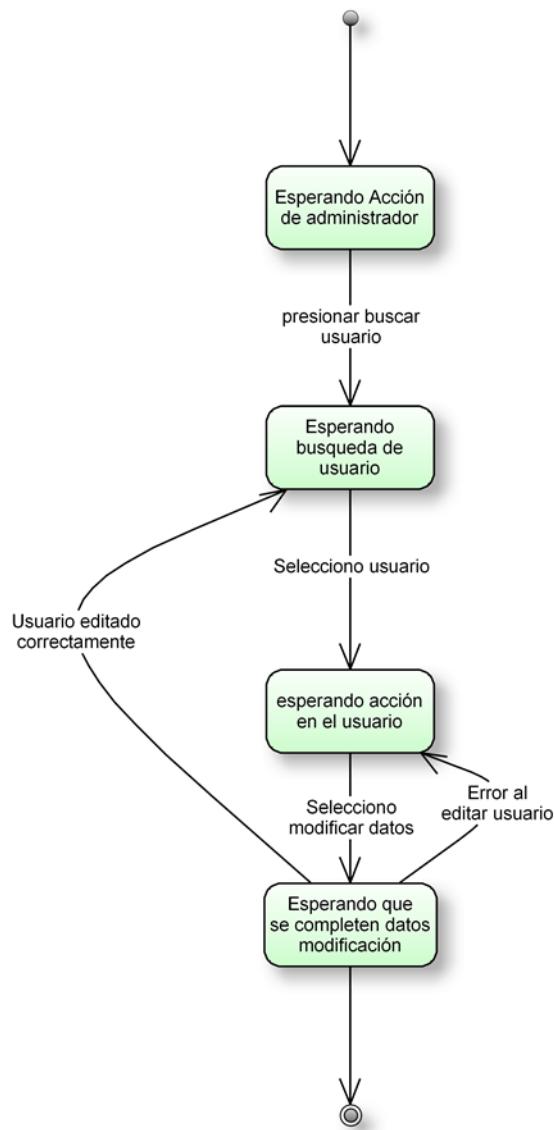


Figura A.6: Diagrama de Estado Modificar Usuario

A.4. Caso de Uso extendido Eliminar Usuario

Nombre Caso de Uso	Gestión de Usuarios
Actores	Usuario
Propósito	Eliminar usuario
Resumen	El administrador debe poder crear, eliminar, modificar y buscar a los usuarios
Tipo	Opcional
Referencias Cruzadas	
Precondiciones	El Administrador debe estar autenticado en el sistema
Curso Normal (Usuario)	<p>El usuario accede a la opción buscar usuario</p> <p>El usuario escribe el nombre en el campo o lo busca entre la lista de usuarios</p> <p>El usuario selecciona eliminar</p>
	<p>El sistema despliega un campo para buscar y el listado de los usuarios</p> <p>El sistema despliega los datos del usuario seleccionado junto a las opciones de modificar y eliminar</p> <p>El sistema muestra un mensaje de operación exitosa</p>
Curso Alternativo (Usuario)	Curso Alternativo (Sistema)
El usuario accede a la opción buscar usuario	El sistema muestra un error de conexión a la base de datos

Tabla A.4: Caso de uso extendido Eliminar Usuario

A.4.1. Diagrama de Flujo Eliminar Usuario

Podemos ver la en el diagrama de secuencia el flujo de la información.

APÉNDICE A. CASOS DE USO EXTENDIDOS, DIAGRAMAS DE SECUENCIA Y DIAGRAMAS DE ESTADO

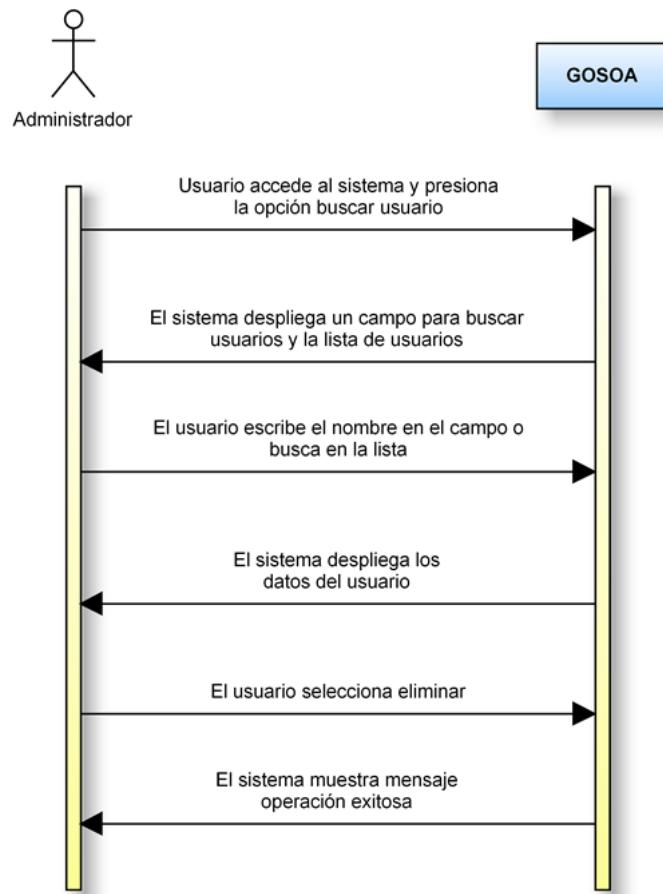


Figura A.7: Diagrama de Secuencia Eliminar Usuario

A.4.2. Diagrama de Estado Modificar Usuario

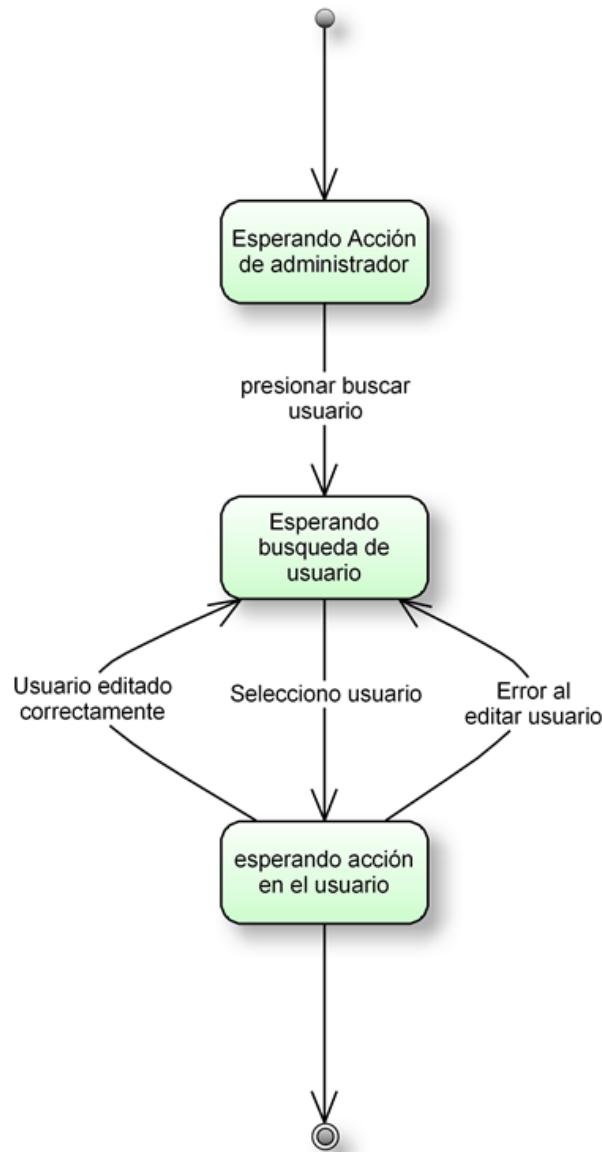


Figura A.8: Diagrama de Estado Eliminar Usuario

APÉNDICE A. CASOS DE USO EXTENDIDOS, DIAGRAMAS DE SECUENCIA Y DIAGRAMAS DE ESTADO

A.5. Caso de uso Extendido Selección de diagrama

Nombre Caso de Uso	Selección de Diagramas
Actores	Usuario
Propósito	Seleccionar el diagrama a realizar
Resumen	El usuario puede escoger entre los distintos diagramas especificados en la metodología
Tipo	Obligatorio
Referencias Cruzadas	
Precondiciones	El Usuario debe estar autenticado en el sistema
Curso Normal (Usuario)	Curso Normal (Sistema)
El usuario ingresa al sistema	El sistema muestra una pantalla con los modelos para que el usuario escoja uno de ellos
El usuario escoge unos de los diagramas	El sistema muestra una ventana para generar los diagramas y las herramientas respectivas para cada uno de ellos
Curso Alternativo (Usuario)	Curso Alternativo (Sistema)

Tabla A.5: Caso de Uso extendido de Selección de diagramas

A.5.1. Diagrama de Flujo Selección Diagrama

Podemos ver el flujo de la información en el siguiente diagrama de secuencia:

APÉNDICE A. CASOS DE USO EXTENDIDOS, DIAGRAMAS DE SECUENCIA Y DIAGRAMAS DE ESTADO

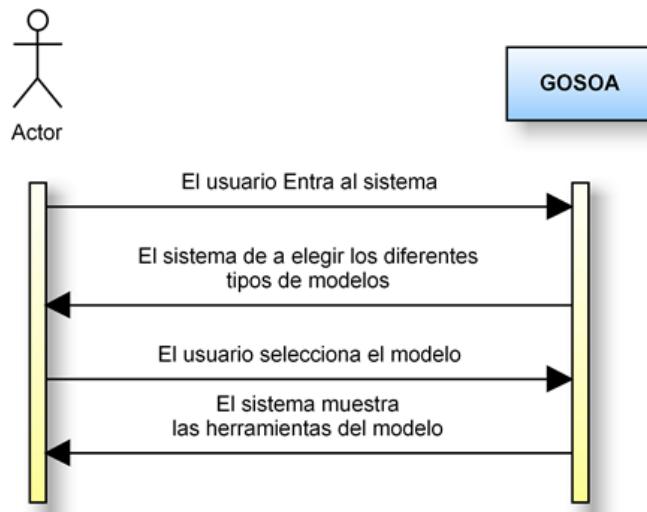


Figura A.9: Diagrama de Secuencia selección de un modelo por el usuario

A.5.2. Diagrama de Estado Selección Diagrama

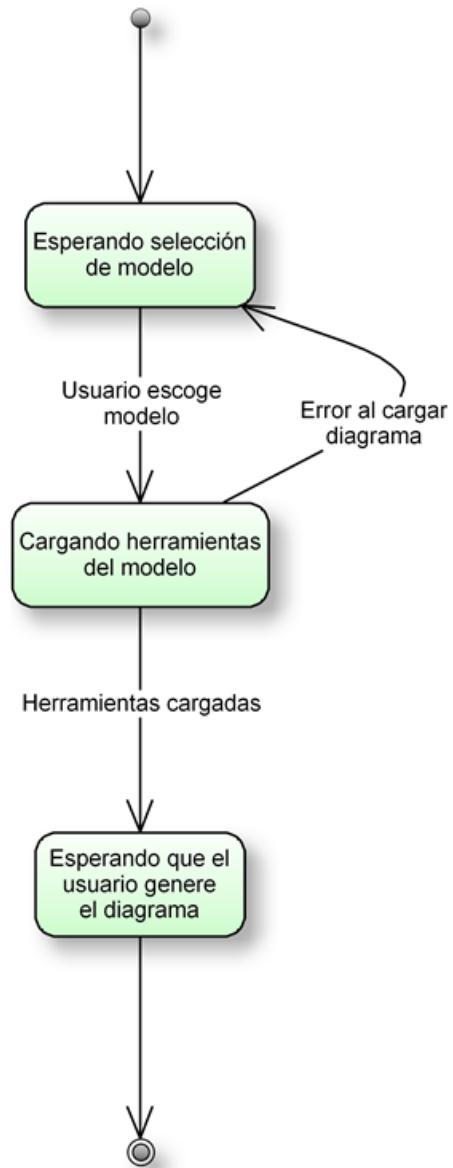


Figura A.10: Diagrama de Estado selección de un modelo por el usuario

APÉNDICE A. CASOS DE USO EXTENDIDOS, DIAGRAMAS DE SECUENCIA Y DIAGRAMAS DE ESTADO

A.6. Caso de Uso Extendido Pasar a siguiente/posterior Diagrama

Nombre Caso de Uso	Pasar a diagrama siguiente o anterior
Actores	Usuario
Propósito	Navegar entre diagramas
Resumen	El usuario puede navegar entre los diagramas posteriores y anteriores según la metodología que se ha descrito
Tipo	Opcional
Referencias Cruzadas	
Precondiciones	El Usuario debe estar autenticado en el sistema
Curso Normal (Usuario)	Curso Normal (Sistema)
El usuario se encuentra en un Modelo y presiona el botón siguiente	El sistema carga las herramientas del siguiente modelo
Curso Alternativo (Usuario)	Curso Alternativo (Sistema)
El usuario presiona el botón atrás	El sistema carga las herramientas del modelo anterior

Tabla A.6: Caso de Uso extendido para avanzar y retroceder

A.6.1. Diagrama de Flujo Pasar Siguiente/Posterior

Muestra el diagrama de secuencia de el proceso de avanzar y retroceder entre diagramas.

APÉNDICE A. CASOS DE USO EXTENDIDOS, DIAGRAMAS DE SECUENCIA Y DIAGRAMAS DE ESTADO

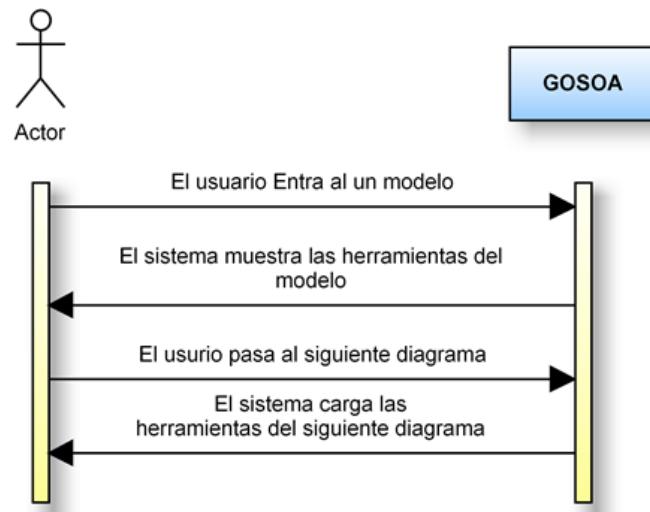


Figura A.11: Diagrama de Secuencia para Avanzar y Retroceder en los diagramas

A.6.2. Diagrama de Estado Pasar Siguiente/Posterior

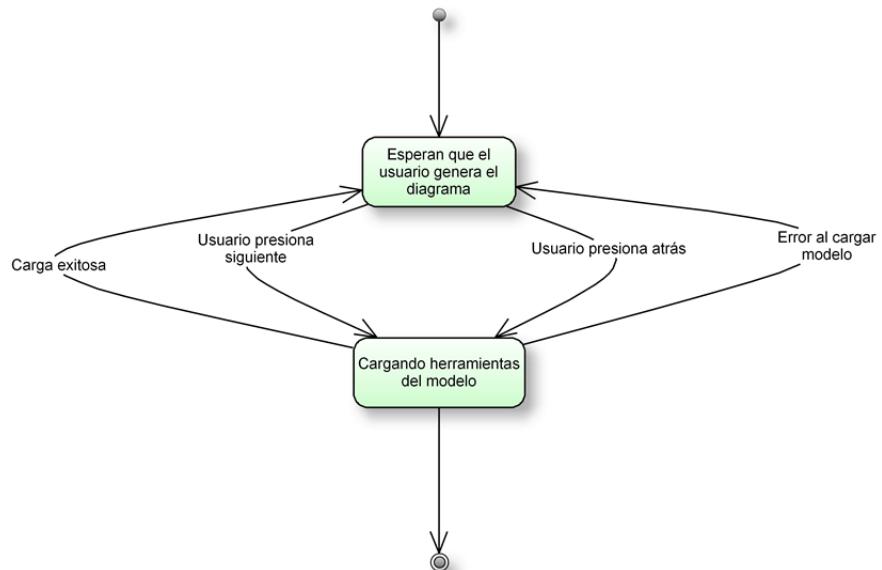


Figura A.12: Diagrama de Estado para Avanzar y Retroceder en los diagramas

APÉNDICE A. CASOS DE USO EXTENDIDOS, DIAGRAMAS DE SECUENCIA Y DIAGRAMAS DE ESTADO

A.7. Caso de Uso Extendido Exportación

Nombre Caso de Uso	Exportar modelo
Actores	Usuario
Propósito	Exportar diagramas de modelos determinados
Resumen	El usuario puede exportar los modelos según los formatos que se predefinan
Tipo	Opcional
Referencias Cruzadas	
Precondiciones	El Usuario debe estar autentificado en el sistema
Curso Normal (Usuario)	Curso Normal (Sistema) El usuario presiona el botón de exportación El usuario Escoge algunos de los formatos para la exportación del modelo El sistema muestra las opciones de exportación de los modelos El sistema exporta el modelo
Curso Alternativo (Usuario)	Curso Alternativo (Sistema)
El usuario cancela la exportación	El sistema muestra un mensaje de exportación cancelada

Tabla A.7: Caso de Uso extendido de Exportación

A.7.1. Diagrama de Flujo Exportación

Se muestra el diagrama de secuencia de la exportación.

APÉNDICE A. CASOS DE USO EXTENDIDOS, DIAGRAMAS DE SECUENCIA Y DIAGRAMAS DE ESTADO

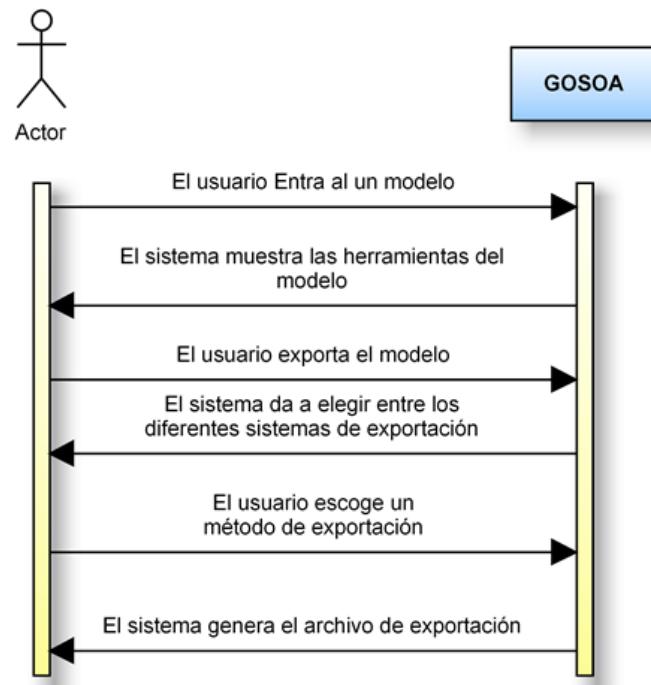


Figura A.13: Diagrama de Secuencia para Exportar Modelo

A.7.2. Diagrama de Estado Exportación

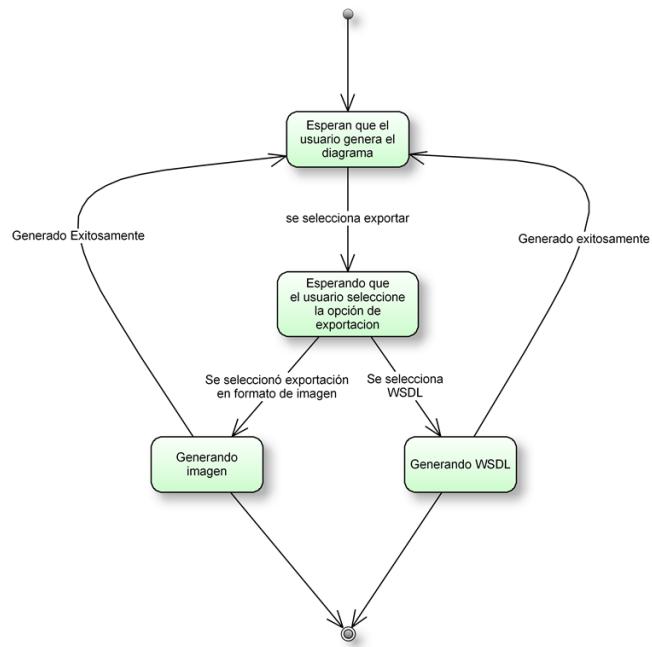


Figura A.14: Diagrama de Estado para Exportar Modelo

A.8. Caso de Uso Extendido Configurar Datos

Nombre Caso de Uso	Configurar Datos
Actores	Usuario
Propósito	Configurar las componentes del sistema
Resumen	En el sistema existen varios componentes que tienen campos como nombres, y estos deben ser configurados. También existen valores generar el WSDL que debe ser configurados previamente
Tipo	Obligatorio
Referencias Cruzadas	
Precondiciones	El Usuario debe estar autenticado en el sistema
Curso Normal (Usuario)	<p>El usuario hace doble click en un elemento del modelo</p> <p>El usuario configura los datos de los elementos y presiona la configuración avanzada</p> <p>El usuario configura los datos avanzados de los elementos</p>
	<p>El sistema genera una ventana para configurar los datos del elemento</p> <p>El sistema muestra una ventana para la configuración avanzada</p> <p>El sistema muestra una ventana de cambio de configuración exitosa</p>
Curso Alternativo (Usuario)	Curso Alternativo (Sistema)
El usuario cancela la operación	El sistema muestra un mensaje de cambio de configuración cancelada

Tabla A.8: Caso de Uso extendido de Configuración de datos

APÉNDICE A. CASOS DE USO EXTENDIDOS, DIAGRAMAS DE SECUENCIA Y DIAGRAMAS DE ESTADO

A.8.1. Diagrama de Flujo Configurar Datos

A continuación se muestra el diagrama de secuencia para configurar los datos.

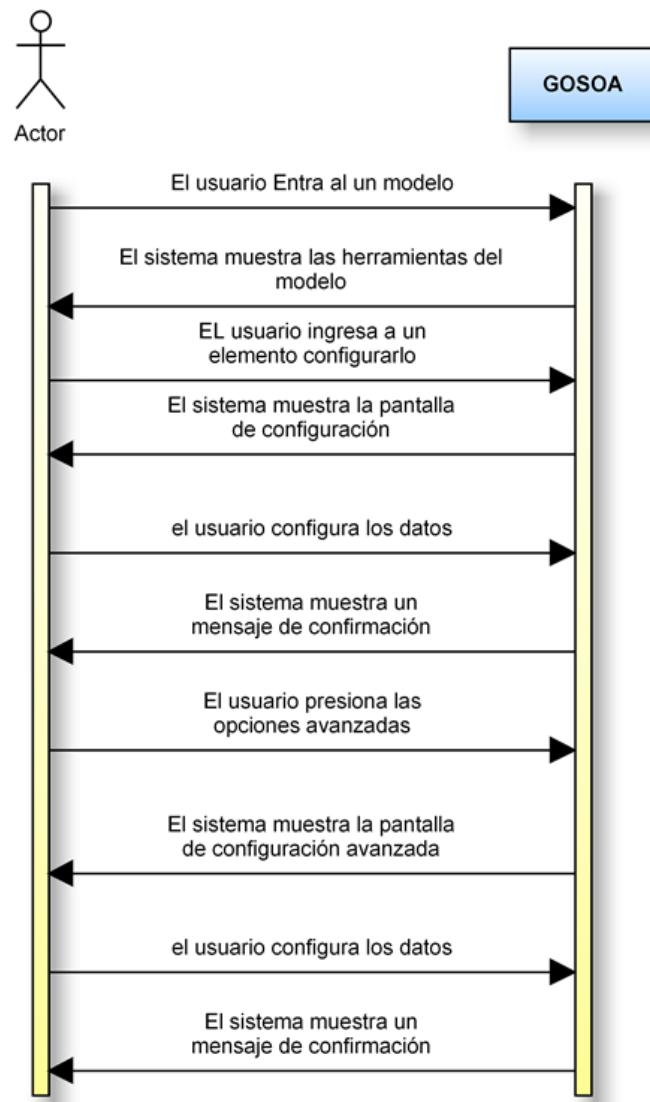


Figura A.15: Diagrama de Secuencia para Configurar Datos

A.8.2. Diagrama de Estado Configurar Datos

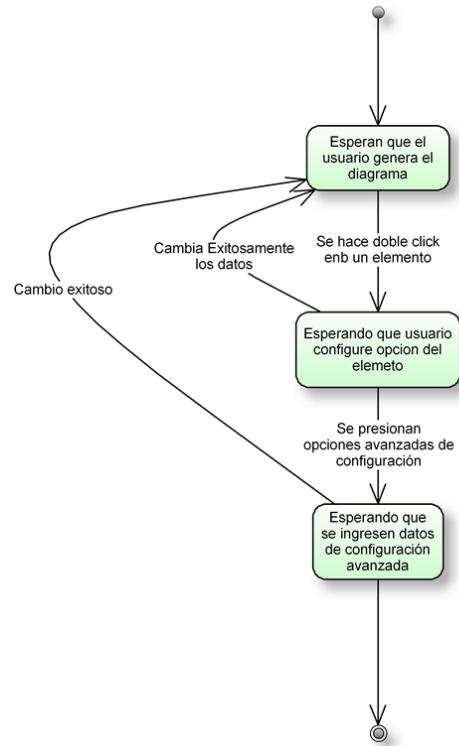


Figura A.16: Diagrama de Estado para Configurar Datos

Apéndice B

Casos de Uso Reales

Nombre Caso de Uso	Acceso al Sistema
Actores	Usuario
Propósito	Acceder al sistema
Resumen	El usuario debe iniciar una sesión para poder acceder al sistema
Tipo	Opcional
Referencias Cruzadas	
Curso Normal (Usuario)	Curso Normal (GOSOA)
El usuario ingresa al sistema	El sistema despliega el formulario con los campos para el nombre de usuario o mail (campo a Figura B.1) junto con la contraseña (campo b Figura B.1)
El usuario ingresa su nombre de usuario y su contraseña y presiona el botón iniciar sesión (campo c Figura B.1)	El sistema Valida al usuario
El usuario accede al sistema	El sistema muestra el perfil del usuario (Figura B.2) o el administrador (Figura B.3) según corresponda
Curso Alternativo (Usuario)	Curso Alternativo (Sistema)
El usuario ingresa entra la sitio	El sistema despliega el formulario con los campos para el nombre de usuario o mail (campo a Figura B.1) junto con la contraseña (campo b Figura B.1)
El usuario ingresa su nombre de usuario y su contraseña y presiona el botón iniciar sesión (campo c Figura B.1)	el sistema no puede validar los datos ya que estos son incorrectos y muestra una pantalla de inicio de sesión inválido (Figura B.4)

Tabla B.1: Caso de Uso Real de Acceso al Sistema

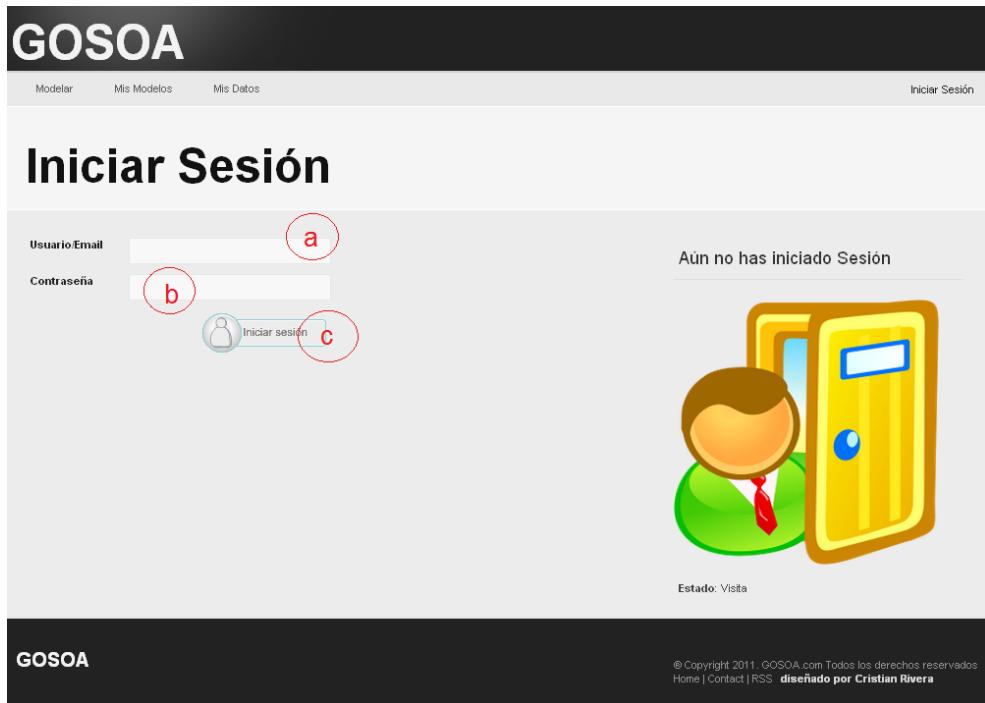


Figura B.1: Interfaz de Login



Figura B.2: Interfaz Perfil de Usuario

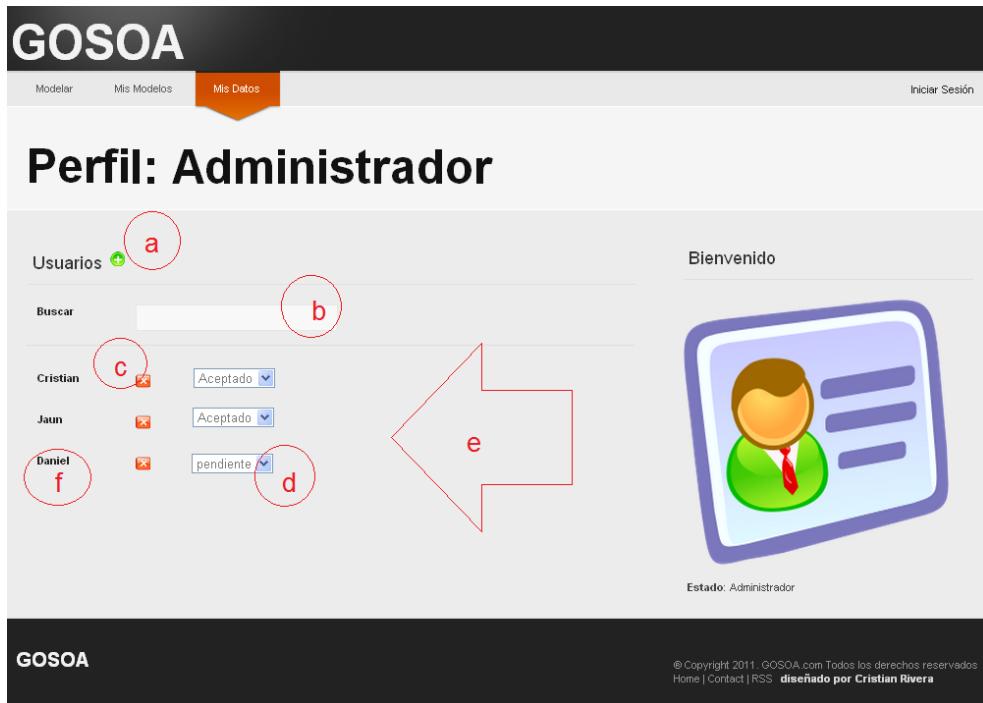


Figura B.3: Interfaz Perfil de administrador

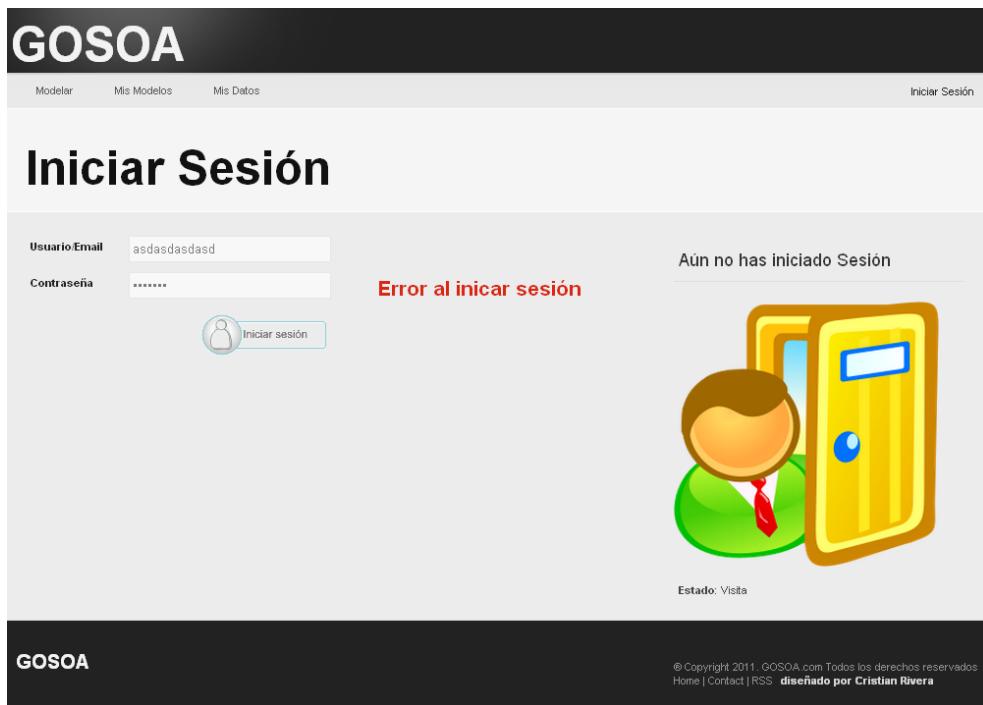


Figura B.4: Interfaz de Login

Nombre Caso de Uso	Gestión de Usuarios
Actores	Usuario
Propósito	Crear Usuario
Resumen	El administrador debe poder crear, eliminar, modificar y buscar a los usuarios
Tipo	Opcional
Referencias Cruzadas	
Precondiciones	El Administrador debe estar autenticado en el sistema
Curso Normal (Usuario)	<p>El Administrador presiona el botón “+” al lado de las palabras Usuarios (campo a Figura B.5)</p> <p>El usuario ingresa los datos en el sistema y presiona el botón de registrar (campo e Figura B.6)</p>
	<p>El sistema despliega un formulario para que se ingresen los datos del nuevo usuario, que son el nombre de usuario(campo a Figura B.6), email (campo b Figura B.6), contraseña (campo c Figura B.6), modelo privado o público (campo d Figura B.6)</p> <p>El sistema muestra un mensaje de operación realizada exitosamente (campo f Figura B.6)</p>
Curso Alternativo (Usuario)	Curso Alternativo (Sistema)
<p>El Administrador presiona el botón “+” al lado de las palabras Usuarios (campo a Figura B.5)</p> <p>El usuario ingresa los datos en el sistema y presiona el botón de registrar (campo f Figura B.6)</p>	<p>El sistema despliega un formulario para que se ingresen los datos del nuevo usuario, que son el nombre de usuario(campo a Figura B.6), email (campo b Figura B.6), contraseña (campo c Figura B.6), modelo privado o público (campo d Figura B.6)</p> <p>El sistema muestra un error por problema de conexión a la base de datos (Figura B.7)</p>

Tabla B.2: Caso de Uso Real de Crear Usuario

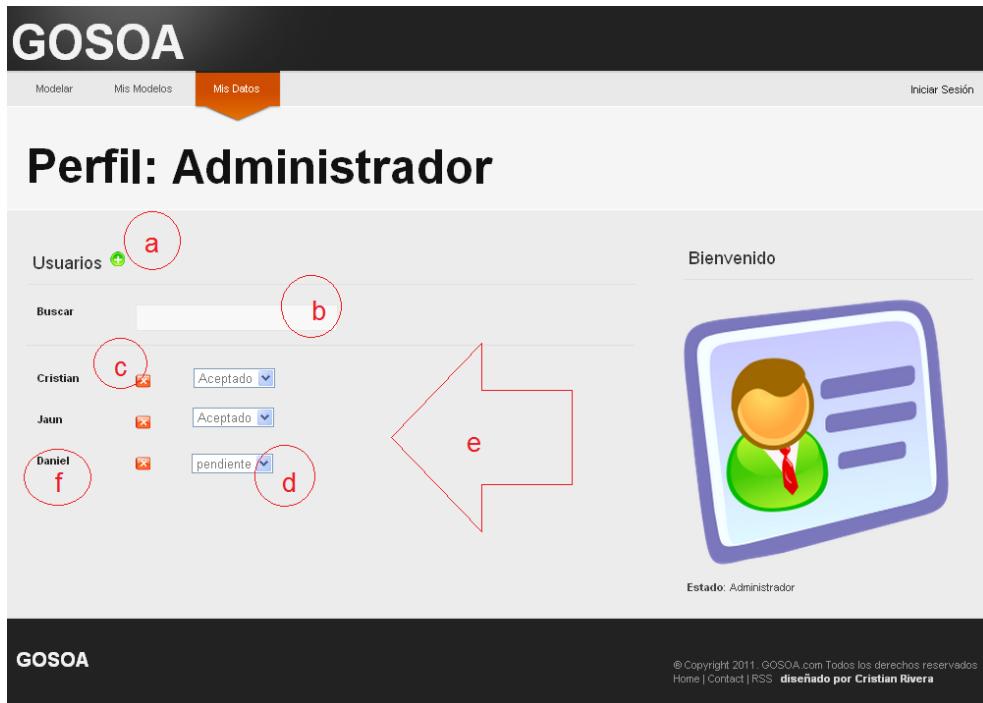


Figura B.5: Interfaz Perfil de administrador

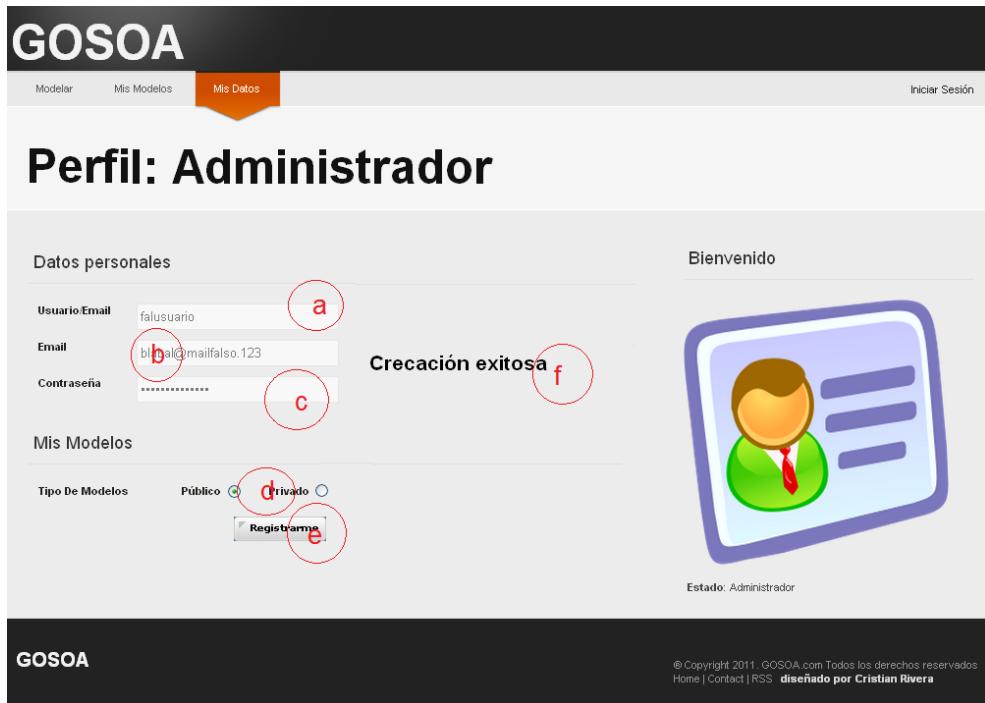


Figura B.6: Interfaz de creación de usuarios



Figura B.7: Interfaz de creación de usuarios

Nombre Caso de Uso	Gestión de Usuarios
Actores	Usuario
Propósito	Modificar datos de usuario
Resumen	El administrador debe poder crear, eliminar, modificar y buscar a los usuarios
Tipo	Opcional
Referencias Cruzadas	
Precondiciones	El Administrador debe estar autenticado en el sistema
Curso Normal (Usuario)	<p>El usuario accede a la opción buscar usuario (campo b Figura B.8)</p> <p>El usuario escribe el nombre en el campo y presiona enter o lo busca entre la lista de usuarios (campo e Figura B.8)</p> <p>El usuario selecciona modificar (campo f Figura B.8)</p> <p>El usuario cambia los datos necesarios y confirma la edición presionando el botón registrar (campo e Figura B.9)</p>
	<p>El sistema queda a la espera del ingreso de caracteres</p> <p>El sistema despliega los datos del usuario seleccionado junto a las opciones de modificar (campo f Figura B.8) y eliminar (campo c Figura B.8)</p> <p>El sistema despliega los datos del usuario seleccionado en campos editables (campo f Figura B.9)</p> <p>El sistema muestra un mensaje de operación exitosa (campo f Figura B.9)</p>
Curso Alternativo (Usuario)	Curso Alternativo (Sistema)
El usuario accede a la opción buscar usuario (campo b Figura B.10)	El sistema muestra un error de conexión a la base de datos (campo c Figura B.10)

Tabla B.3: Caso de Uso Real Modificar Usuario

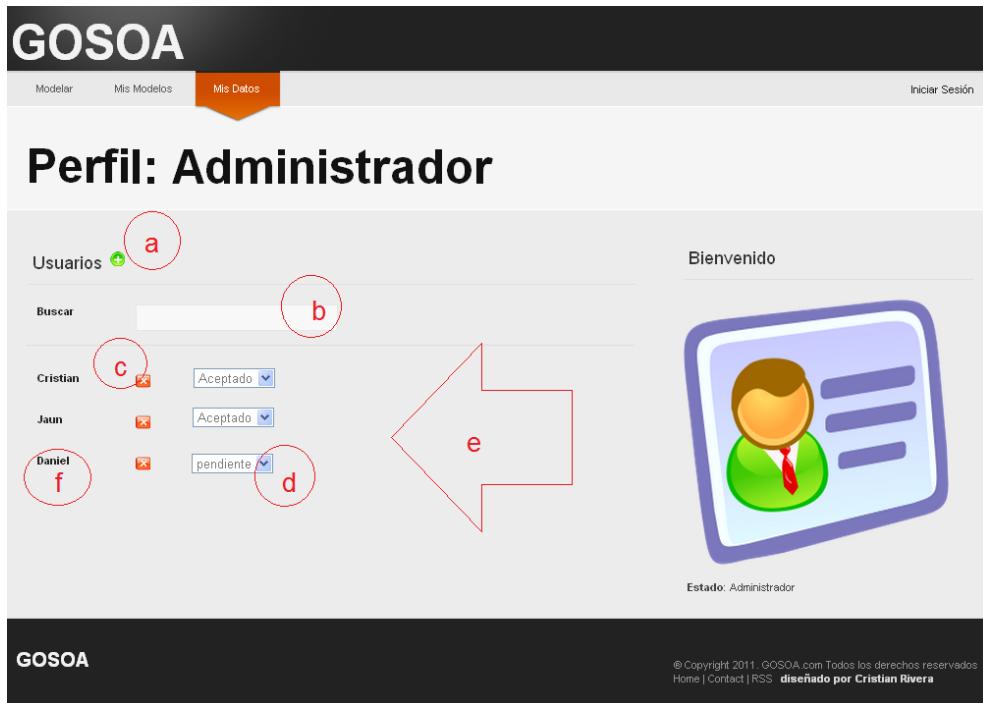


Figura B.8: Interfaz Perfil de administrador

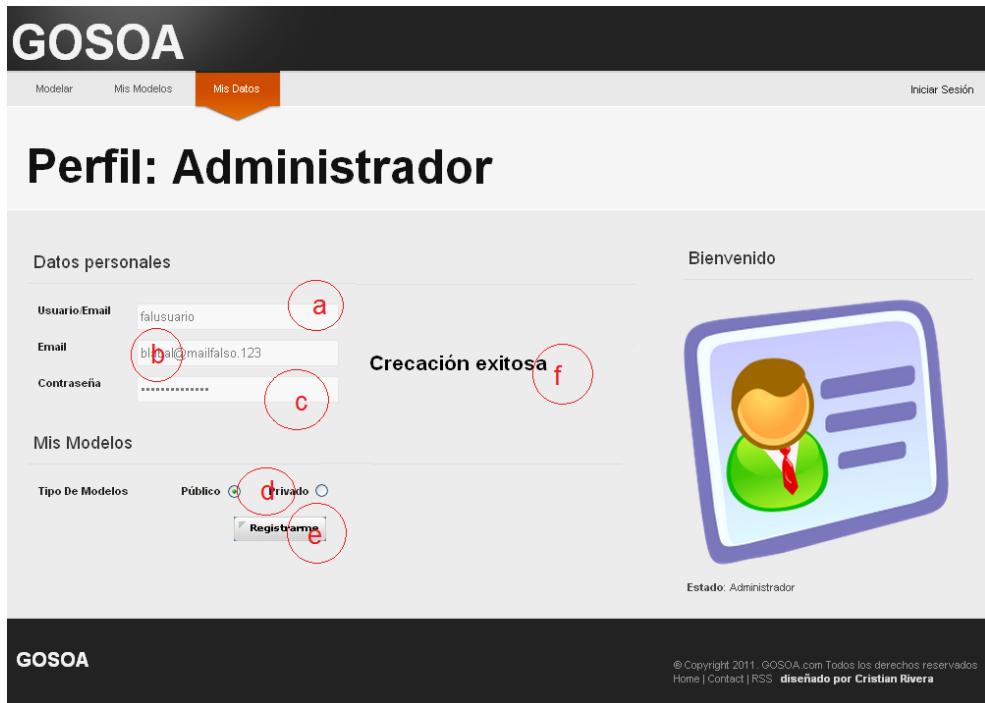


Figura B.9: Interfaz Modificación datos de usuario

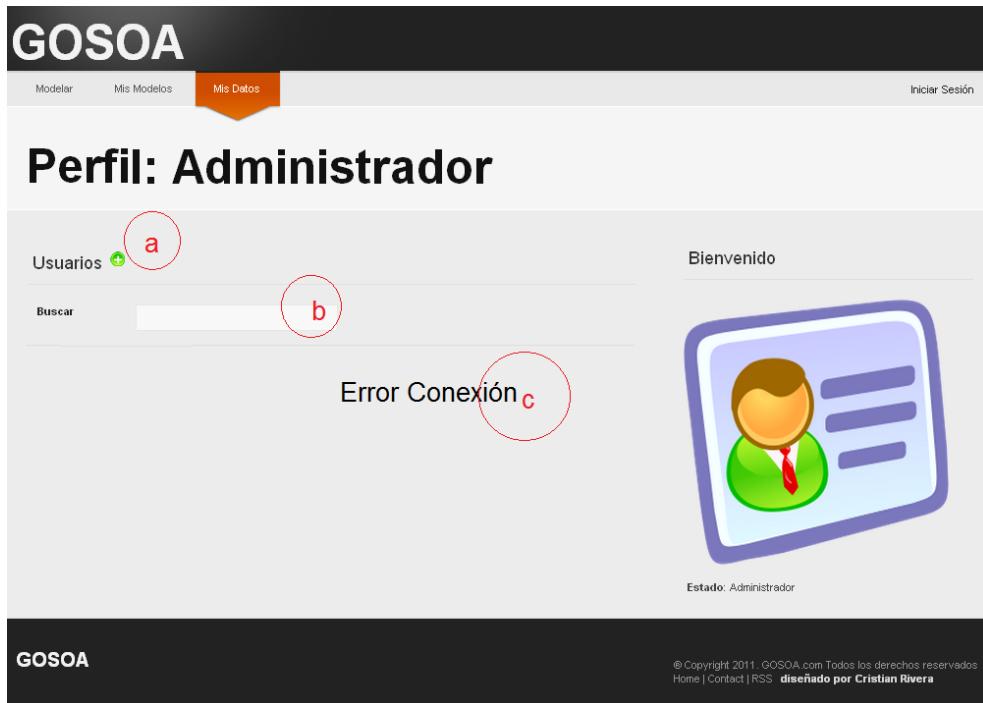


Figura B.10: Interfaz de problema conexión al buscar usuario

Nombre Caso de Uso	Gestión de Usuarios
Actores	Usuario
Propósito	Eliminar usuario
Resumen	El administrador debe poder crear, eliminar, modificar y buscar a los usuarios
Tipo	Opcional
Referencias Cruzadas	
Precondiciones	El Administrador debe estar autenticado en el sistema
Curso Normal (Usuario)	Curso Normal (Sistema)
El usuario accede a la opción buscar usuario (campo b Figura B.11)	El sistema queda a la espera del ingreso de caracteres
El usuario escribe el nombre en el campo y presiona enter o lo busca entre la lista de usuarios (campo e Figura B.11)	El sistema despliega los datos del usuario seleccionado junto a las opciones de modificar (campo f Figura B.11) y eliminar (campo c Figura B.11)
El usuario selecciona eliminar (campo c Figura B.11)	El sistema muestra un mensaje de operación exitosa (campo g Figura B.11)
Curso Alternativo (Usuario)	Curso Alternativo (Sistema)
El usuario accede a la opción buscar usuario (campo b Figura B.12)	El sistema muestra un error de conexión a la base de datos (campo c Figura B.12)

Tabla B.4: Caso de uso Real Eliminar Usuario



Figura B.11: Interfaz de eliminación de usuarios

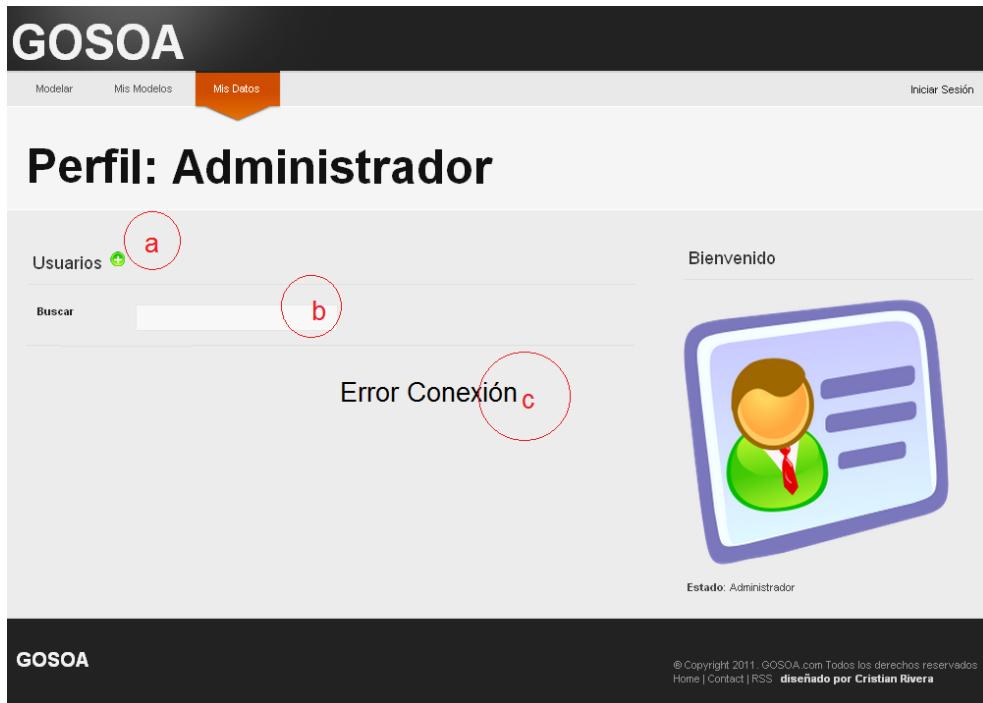


Figura B.12: Interfaz de problema conexión al buscar usuario

Nombre Caso de Uso	Selección de Diagramas
Actores	Usuario
Propósito	Seleccionar el diagrama a realizar
Resumen	El usuario puede escoger entre los distintos diagramas especificados en la metodología
Tipo	Obligatorio
Referencias Cruzadas	
Precondiciones	El Usuario debe estar autentificado en el sistema
Curso Normal (Usuario)	Curso Normal (Sistema)
El usuario ingresa al sistema El usuario escoge unos de los diagramas (Campos a, b, c, d, e Figura B.13)	El sistema muestra una pantalla con los modelos para que el usuario escoja uno de ellos (Figura B.13) El sistema muestra una ventana para generar los diagramas (campo c Figura B.14) y las herramientas (campo d Figura B.14) respectivas para cada uno de ellos (Global Model: Figura B.14, Process Model: Figura B.15, Protocol Model: Figura B.16, Design Pattern View: Figura B.17, Composition Design Pattern View: Figura B.18)
Curso Alternativo (Usuario)	Curso Alternativo (Sistema)

Tabla B.5: Caso de Uso Real de Selección de diagramas

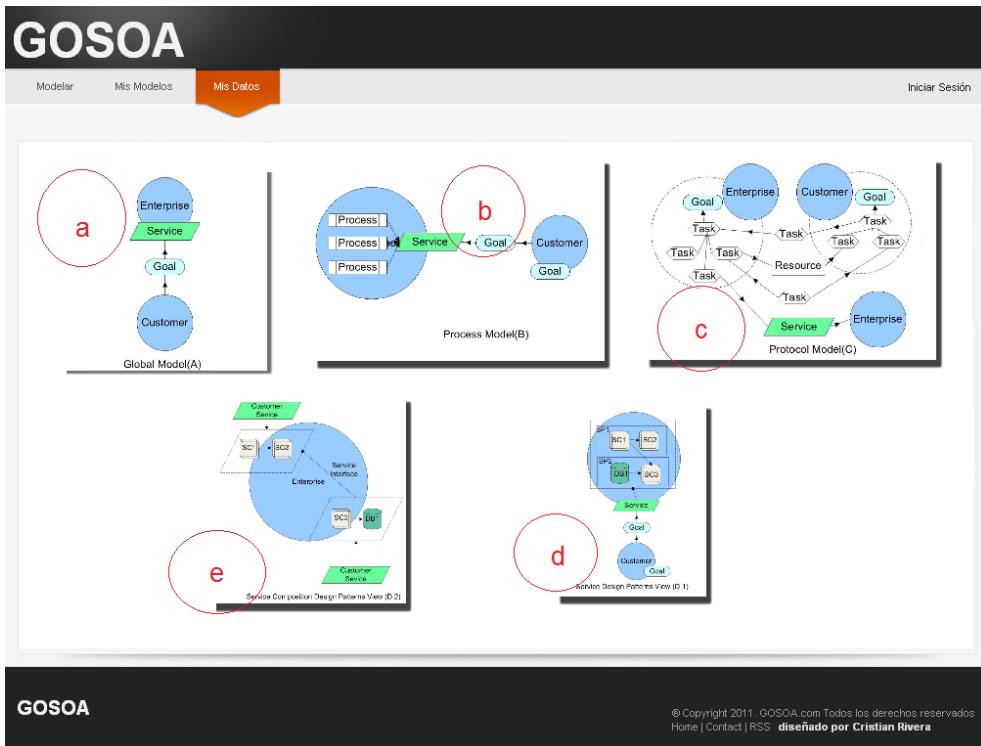


Figura B.13: Interfaz de selección de diagramas

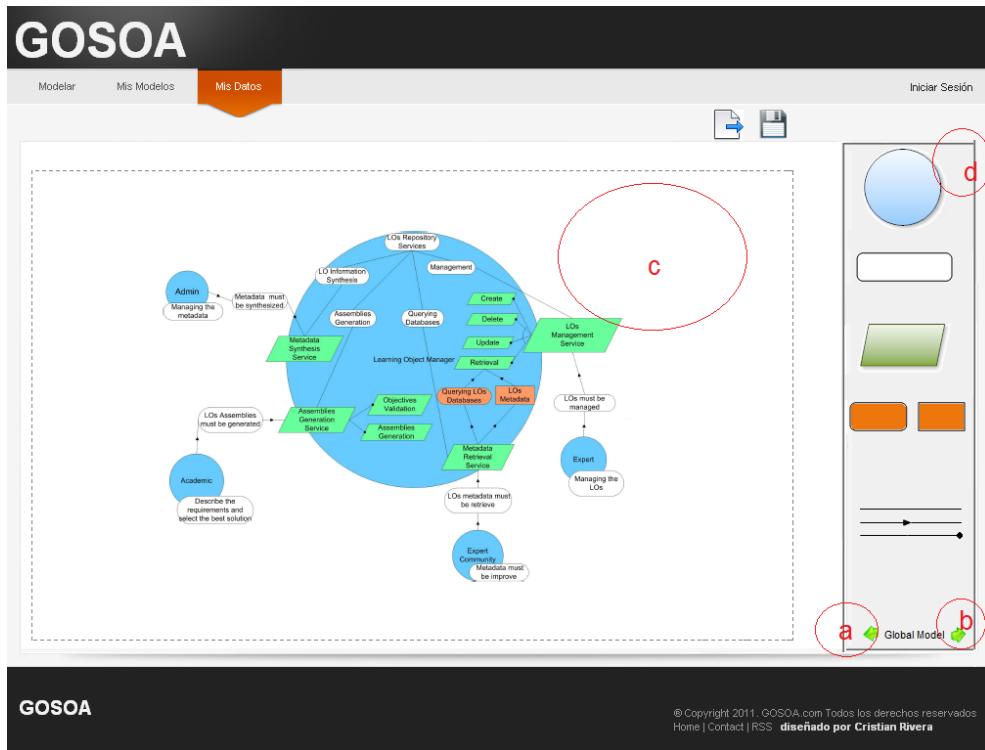


Figura B.14: Interfaz de Diagrama Global Model

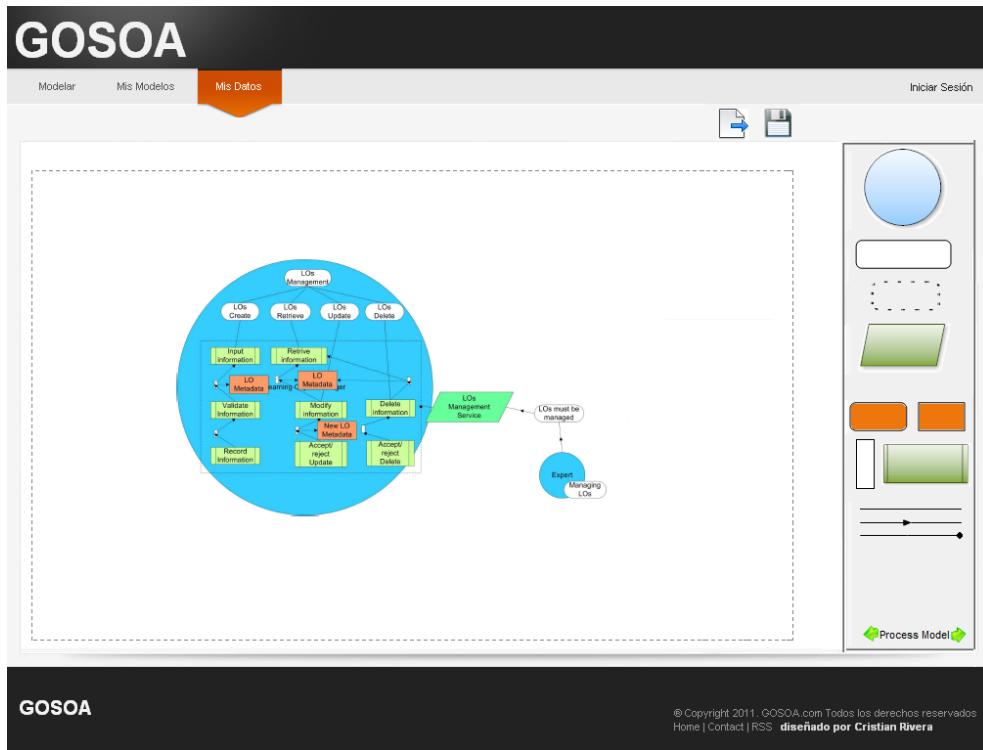


Figura B.15: Interfaz de Diagrama Process Model

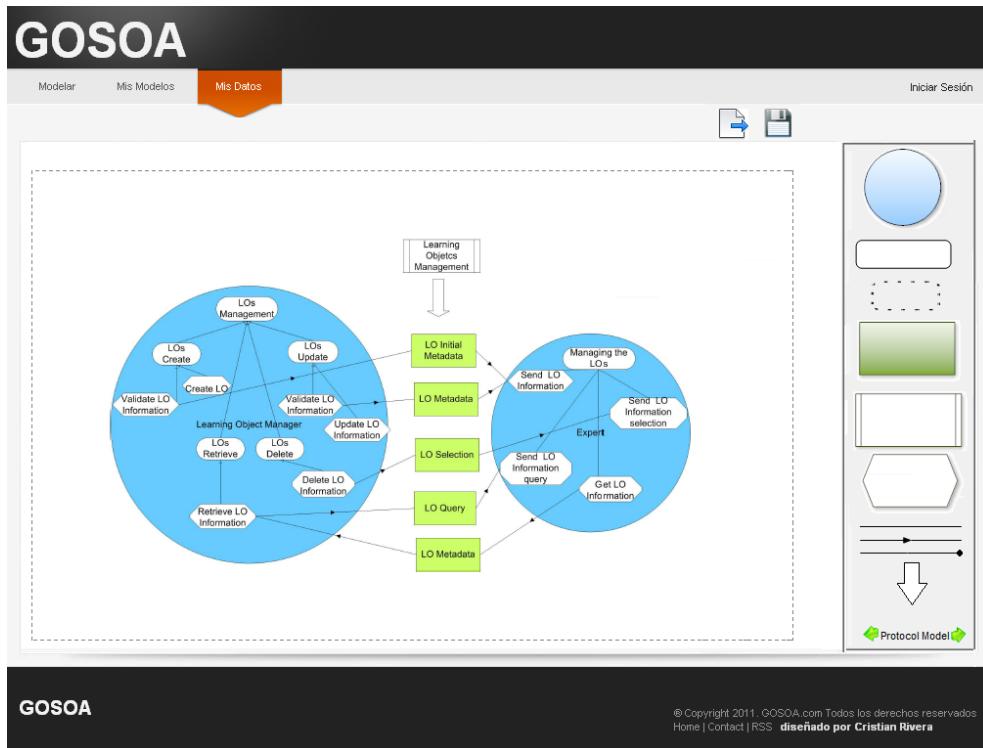


Figura B.16: Interfaz de Diagrama Protocol Model

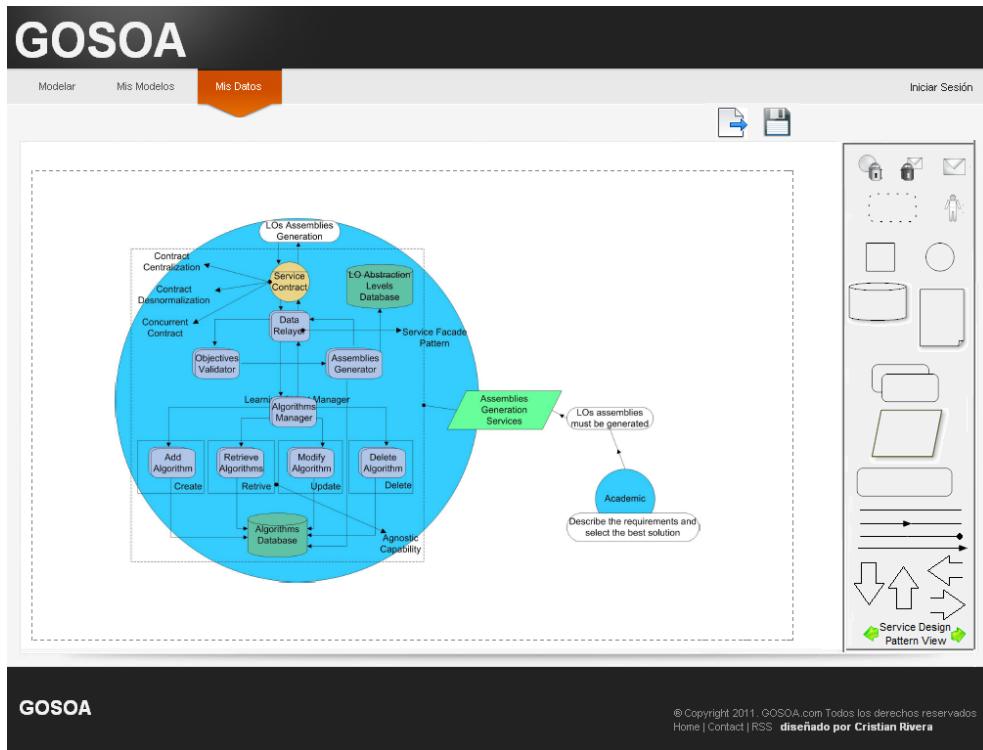


Figura B.17: Interfaz de Diagrama Service Desing Pattern View Model

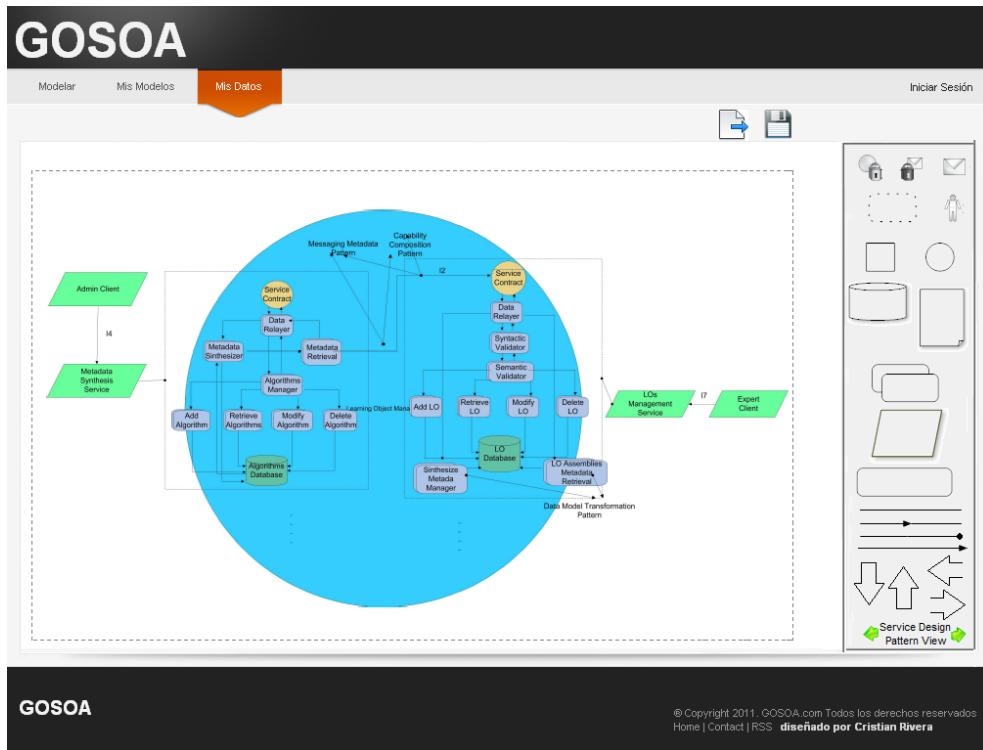


Figura B.18: Interfaz de Diagrama Service Desing Pattern View Model

Nombre Caso de Uso	Pasar a diagrama siguiente o anterior
Actores	Usuario
Propósito	Navegar entre diagramas
Resumen	El usuario puede navegar entre los diagramas posteriores y anteriores según la metodología que se ha descrito
Tipo	Opcional
Referencias Cruzadas	
Precondiciones	El Usuario debe estar autenticado en el sistema
Curso Normal (Usuario)	Curso Normal (Sistema)
El usuario se encuentra en un Modelo y presiona el botón siguiente (campo a Figura B.19)	El sistema carga las herramientas del siguiente modelo
Curso Alternativo (Usuario)	Curso Alternativo (Sistema)
El usuario presiona el botón atrás (campo b Figura B.19)	El sistema carga las herramientas del modelo anterior

Tabla B.6: Caso de Uso Real para avanzar y retroceder

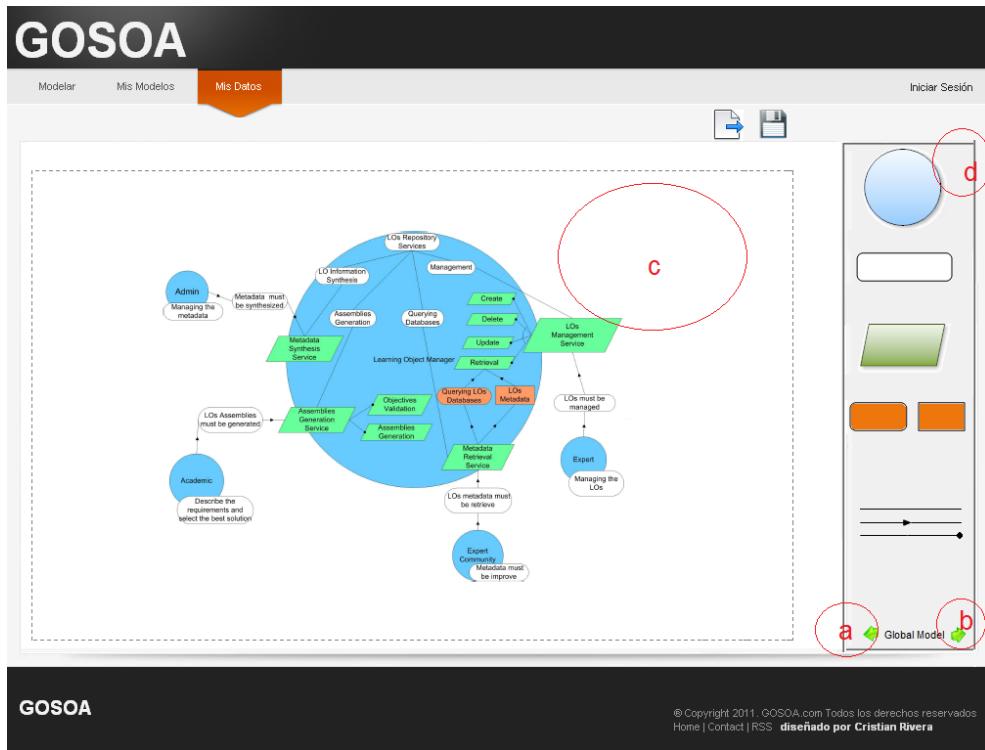


Figura B.19: Interfaz de para cambiar de Modelo

Nombre Caso de Uso	Exportar modelo
Actores	Usuario
Propósito	Exportar diagramas de modelos determinados
Resumen	El usuario puede exportar los modelos según los formatos que se predefinan
Tipo	Opcional
Referencias Cruzadas	
Precondiciones	El Usuario debe estar autentificado en el sistema
Curso Normal (Usuario)	Curso Normal (Sistema) El sistema muestra las opciones de exportación de los modelos (Figura B.21) El sistema exporta el modelo
Curso Alternativo (Usuario)	Curso Alternativo (Sistema)
El usuario presiona escape	El sistema vuelve al modelo (Figura B.20)

Tabla B.7: Caso de Uso Real de Exportación

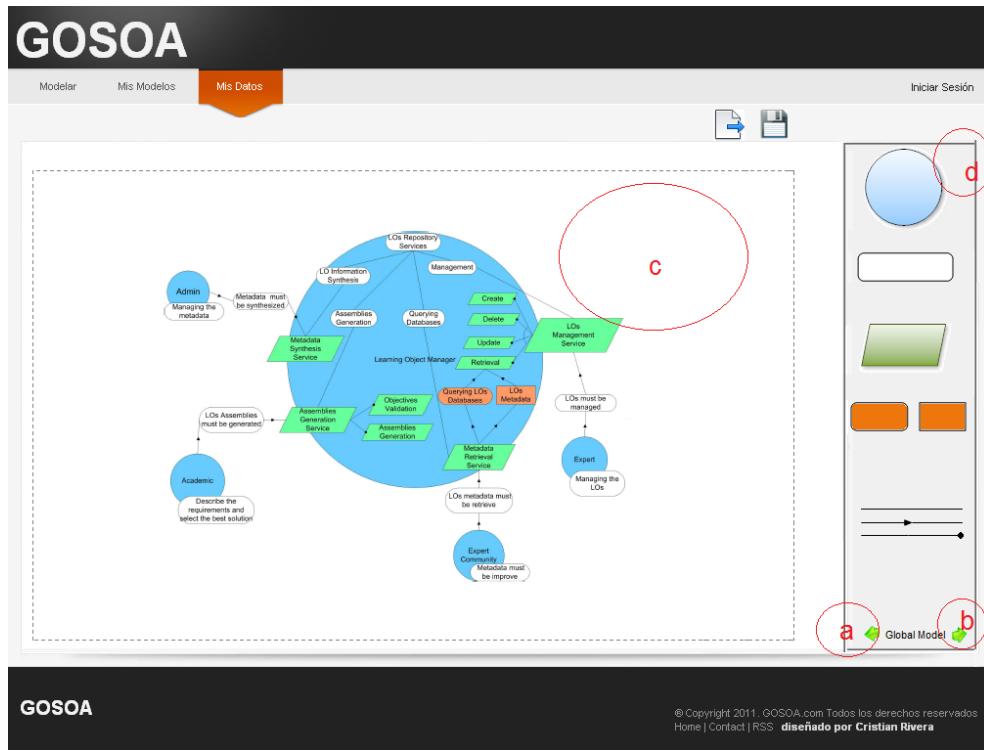


Figura B.20: Interfaz Para empezar la exportación

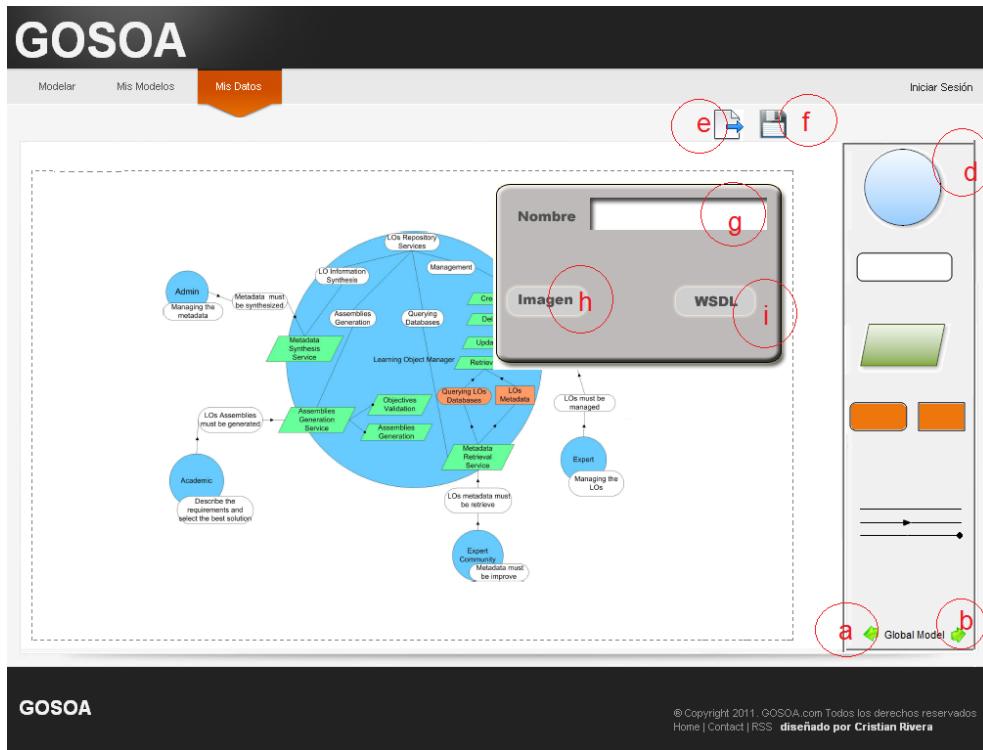


Figura B.21: Interfaz de imagen de exportación

Nombre Caso de Uso	Configurar Datos	
Actores	Usuario	
Propósito	Configurar las componentes del sistema	
Resumen	En el sistema existen varios componentes que tienen campos como nombres, y estos deben ser configurados. También existen valores generar el WSDL que debe ser configurados previamente	
Tipo	Obligatorio	
Referencias Cruzadas		
Precondiciones	El Usuario debe estar autenticado en el sistema	
Curso Normal (Usuario)	<p>El usuario hace doble click en un elemento del modelo</p> <p>El usuario configura los datos de los elementos: nombre (campo e Figura B.23) y color (campo f Figura B.23). luego presiona la configuración avanzada (campo g Figura B.23)</p> <p>El usuario configura los datos avanzados de los elementos (campo h Figura B.24) y luego el usuario presiona escape</p>	<p>El sistema genera una ventana para configurar los datos del elemento (Figura B.23)</p> <p>El sistema muestra una ventana para la configuración avanzada (Figura B.24)</p> <p>El sistema vuelve al modelo (Figura B.22)</p>
Curso Alternativo (Usuario)	Curso Alternativo (Sistema)	
El usuario presiona escape	El sistema vuelve al modelo (Figura B.22)	

Tabla B.8: Caso de Uso Real de Configuración de datos

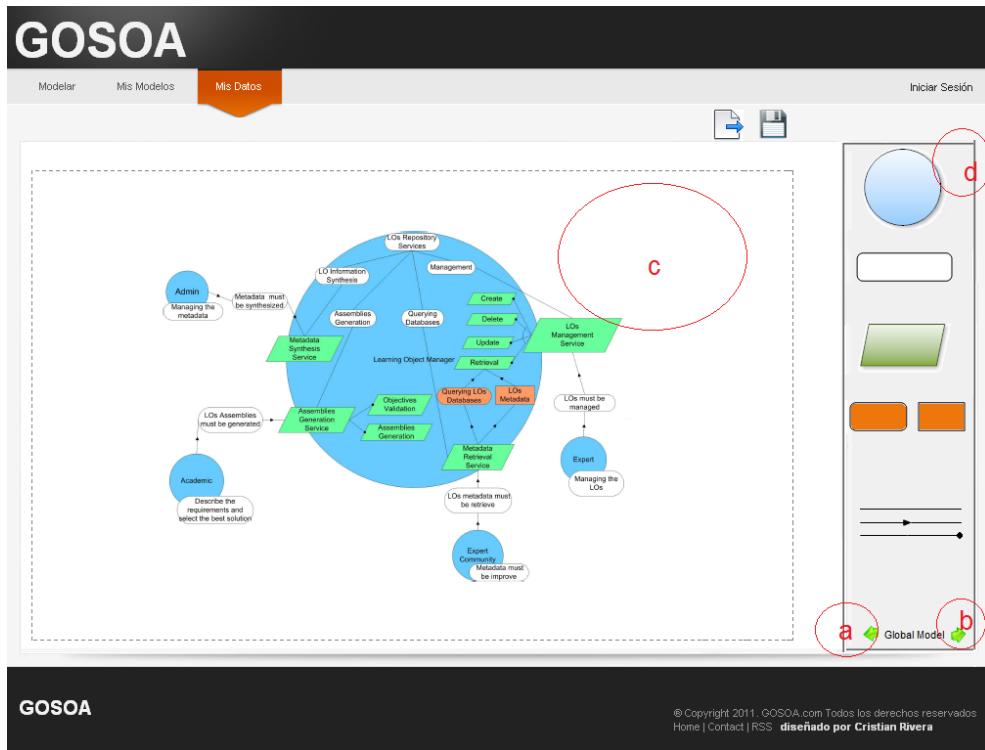


Figura B.22: Interfaz volver al diagrama

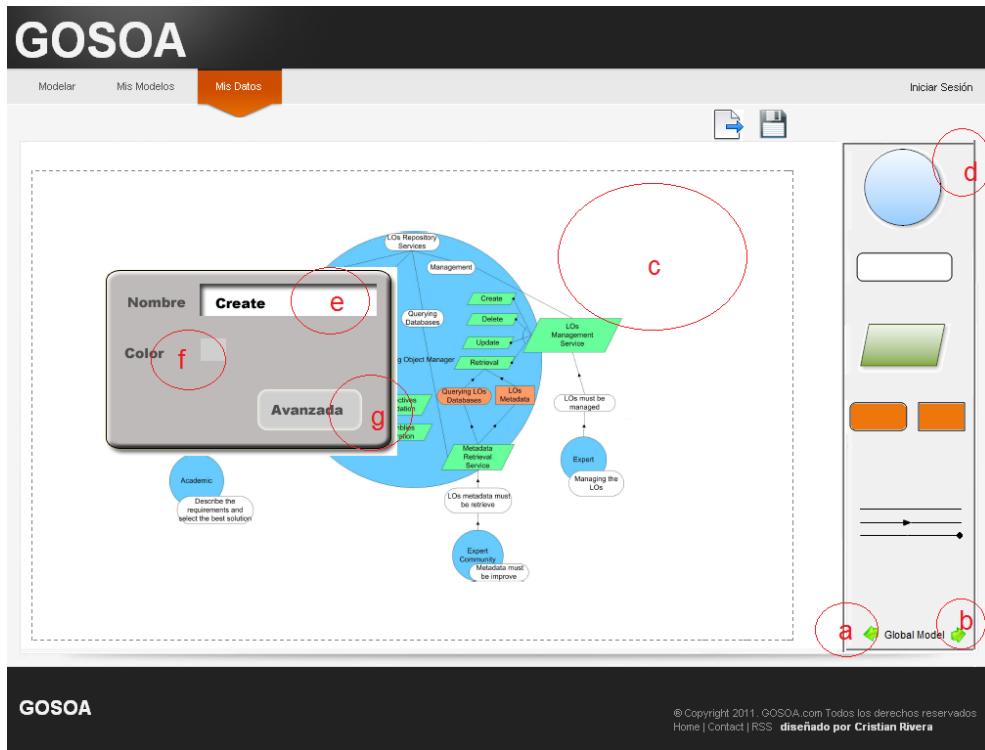


Figura B.23: Interfaz de opciones de elemento

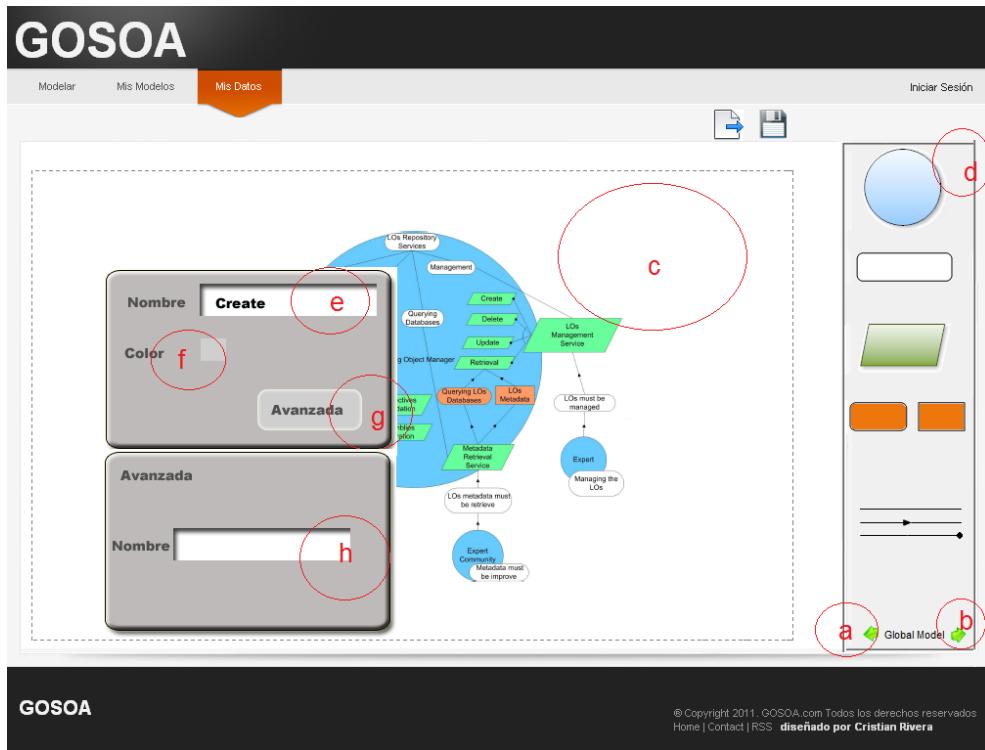


Figura B.24: Interfaz de opciones de elemento avanzadas

Apéndice C

Pruebas Unitarias

C.1. Modulos a probar

Explicaremos cuales son los módulos a probar junto a sus entradas y salidas esperadas. También las pruebas han sido separadas para su mejor comprensión.

A nivel de Interfaz		
Caso de Uso	Entradas	Salida esperada
Acceder al Sistema	Nombre de Usuario. Contraseña.	<code>\$_POST['nombre']</code> . <code>\$_POST['pass']</code> .
Crear Usuario	Nombre de usuario. Contraseña. Email. Modelos{público, privado}	<code>\$_POST['nombre']</code> <code>\$_POST['pass']</code> <code>\$_POST['mail']</code> <code>\$_POST['modelo']</code>
Modificar datos Usuario	Nombre de usuario. Contraseña. Email. Modelos{público, privado} Id_usuario.	<code>\$_POST['nombre']</code> <code>\$_POST['pass']</code> <code>\$_POST['mail']</code> <code>\$_POST['modelo']</code> <code>\$_POST['Id']</code>
Eliminar Usuario	{se presiona botón eliminar}	<code>\$_POST['Id']</code>

Figura C.1: Pruebas Unitarias a nivel de interfaz

A nivel de Lógica		
Caso de Uso	Entradas	Salida esperada
Acceder al Sistema	<code>\$_POST['nombre']</code> . <code>\$_POST['pass']</code> .	Genera consulta SQL para validar usuario
Crear Usuario	<code>\$_POST['nombre']</code> <code>\$_POST['pass']</code> <code>\$_POST['mail']</code> <code>\$_POST['modelo']</code>	Genera consulta SQL para crear nuevo usuario
Modificar datos Usuario	<code>\$_POST['nombre']</code> <code>\$_POST['pass']</code> <code>\$_POST['mail']</code> <code>\$_POST['modelo']</code> <code>\$_POST['Id']</code>	Genera consulta SQL para editar datos del usuario
Eliminar Usuario	<code>\$_POST['Id']</code>	Genera consulta SQL para eliminar usuario

Figura C.2: Pruebas Unitarias a nivel de Lógica

A nivel de Persistencia		
Caso de Uso	Entradas	Salida esperada
Acceder al Sistema	Genera consulta SQL para validar usuario	Ejecuta consulta en MySQL de validación de usuario
Crear Usuario	Genera consulta SQL para crear nuevo usuario	Ejecuta consulta en MySQL de creación de usuario
Modificar datos Usuario	Genera consulta SQL para editar datos del usuario	Ejecuta consulta en MySQL de edición de datos de usuario
Eliminar Usuario	Genera consulta SQL para eliminar usuario	Ejecuta consulta en MySQL de eliminación de usuario

Figura C.3: Pruebas Unitarias a nivel de Persistencia

A nivel de Interfaz de Módulo Modelador		
Caso de Uso	Entradas	Salida esperada
Selección de Diagramas	{click en diagramas}	Var tipo=<tipo de modelo>
Pasar Siguiente o anterior	{click en botón siguiente o anterior}	Var próximo=<siguiente o anterior> Var tipo=<tipo de modelo>
Exportar Modelo	{click en formato de modelo a exportar}	Var nombre=<nombre del modelo> Var formato=<formato del modelo>
Configurar Datos	{click en elemento a configurar}	Var elemento=<elemento seleccionado> Var valor=<valor a cambiar>

Figura C.4: Pruebas Unitarias del modelador a nivel de Interfaz

A nivel de Lógica de Módulo Modelador		
Caso de Uso	Entradas	Salida esperada
Selección de Diagramas	Var tipo=<tipo de modelo>	Genero class Modelo. Modelo.tipo=tipo Creo class HerramientaDeModelado Ejecuto: Carga de Conectores() Carga de Elementos()
Pasar Siguiente o anterior	Var próximo=<siguiente o anterior>	Si próximo es siguiente Ejecuto: Siguiente Modelo() sino Anterior Modelo().
Exportar Modelo	Var nombre=<nombre del modelo> Var formato=<formato del modelo>	Ejecuto: Si formato es WSDL Generar WSDL() sino Exportar(nombre, formato)
Configurar Datos	Var elemento=<elemento seleccionado> Var valor=<valor a cambiar>	Ejecuto: Si elemento es conector Conector.Setnombre(valor) sino Elemento.setnombre(valor)

Figura C.5: Pruebas Unitarias del modelador a nivel de Lógica

A continuación se detallan una a una las pruebas unitarias aplicadas al sistema GO-SOA

C.2. Acceder Al Sistema - Interfaz

Prueba Unitaria														
Fecha	29 de agosto de 2011	Nº	1											
Nombre	Acceder al sistema (Interfaz)	Módulo	Sistema Login											
Información Caso de Prueba														
Descripción: Se prueba la Interfaz del sistema de login para verificar su correcto funcionamiento.		Enfoque: Inicio de sesión												
Datos de Entrada: Nombre de usuario. Contraseña.		Datos De Salida (Esperados): <code>\$_POST['nombre']</code> <code>\$_POST['pass']</code>												
Procedimiento														
1.Se ingresa a la página de inicio de sesión 2.Se ingresa el nombre de usuario y la contraseña 3.Se muestra la página con las variables seteadas														
Precondiciones:														
Se modifica la página de salida para ver el valor de las variables														
Resultados														
Resultados Obtenidos: Se muestra correctamente las variables		Estado <table border="1"> <tr> <td>Aprobado</td> <td>X</td> </tr> <tr> <td>No Aprobado</td> <td></td> </tr> <tr> <td colspan="2">De no aprobar especificar falla:</td></tr> <tr> <td>Grave</td> <td></td> </tr> <tr> <td>Menor</td> <td></td> </tr> </table>			Aprobado	X	No Aprobado		De no aprobar especificar falla:		Grave		Menor	
Aprobado	X													
No Aprobado														
De no aprobar especificar falla:														
Grave														
Menor														
Observaciones														
Sin observaciones														

Figura C.6: Acceder al Sistema - Interfaz

C.3. Acceder Al Sistema - Lógica

Prueba Unitaria					
Fecha	29 de agosto de 2011	Nº	5		
Nombre	Acceder al sistema (lógica)	Módulo	Sistema Validar Usuario		
Información Caso de Prueba					
Descripción: Se prueba la lógica del sistema de Validar usuario para verificar su correcto funcionamiento.		Enfoque: Acceder al sistema			
Datos de Entrada: \$_POST['nombre'] \$_POST['pass']		Datos De Salida (Esperados): \$sql="select * from usuarios where nombre=\$_POST['nombre'] and pass=\$_POST['pass']"			
Procedimiento					
1.se envían las variables a la página para procesar el acceso al sistema 2.Se muestra la página con las variables seteadas					
Precondiciones:					
Se modifica la página de salida para ver el valor la consulta sql					
Resultados					
Resultados Obtenidos: Se muestra correctamente las variables	Estado				
	Aprobado	X			
	No Aprobado				
	De no aprobar especificar falla:				
	Grave				
	Menor				
Observaciones					
Sin observaciones					

Figura C.7: Acceder al Sistema - Lógica

C.4. Acceder Al Sistema - Persistencia

Prueba Unitaria					
Fecha	29 de agosto de 2011	Nº	9		
Nombre	Acceder al Sistema (persistencia)	Módulo	Sistema Conex		
Información Caso de Prueba					
Descripción: Se prueba la persistencia del sistema acceder al sistema para verificar su correcto funcionamiento.		Enfoque: Acceder al Sistema			
Datos de Entrada: <code>\$sql="select * from usuarios where nombre=\$_POST['nombre'] and pass=\$_POST['pass']"</code>		Datos De Salida (Esperados): Mensaje de acceso exitoso			
Procedimiento					
1.se envían las sentencias sql a la página que se comunica con mysql 2.Se muestra la página con el mensaje del acceso a la base de datos					
Precondiciones:					
Se modifica la página de salida para ver el valor la consulta sql					
Resultados					
Resultados Obtenidos: Se muestra correctamente las variables	Estado				
	Aprobado	X			
	No Aprobado				
	De no aprobar especificar falla:				
	Grave				
	Menor				
Observaciones					
Sin observaciones					

Figura C.8: Acceder al Sistema - Persistencia

C.5. Crear Usuario - Interfaz

Prueba Unitaria					
Fecha	29 de agosto de 2011	Nº	2		
Nombre	Crear Usuario (Interfaz)	Módulo	Sistema Registro		
Información Caso de Prueba					
Descripción: Se prueba la Interfaz del sistema de Registro para verificar su correcto funcionamiento.		Enfoque: Crear nuevo usuario			
Datos de Entrada: Nombre de usuario. Contraseña. Email Mis Modelos {público, privado}		Datos De Salida (Esperados): <code>\$_POST['nombre']</code> <code>\$_POST['pass']</code> <code>\$_POST['email']</code> <code>\$_POST['modelo']</code>			
Procedimiento					
1.Se ingresa a la página de Registro 2.Se ingresa el nombre de usuario, contraseña, email y elegir tipo de modelo 3.Se muestra la página con las variables seteadas					
Precondiciones:					
Se modifica la página de salida para ver el valor de las variables					
Resultados					
Resultados Obtenidos: Se muestra correctamente las variables	Estado				
	Aprobado	X			
No Aprobado De no aprobar especificar falla:					
Grave					
Menor					
Observaciones					
Sin observaciones					

Figura C.9: Crear Usuario - Interfaz

C.6. Crear Usuario - Lógica

Prueba Unitaria					
Fecha	29 de agosto de 2011	Nº	6		
Nombre	Crear Usuario (lógica)	Módulo	Sistema Manejador de registro		
Información Caso de Prueba					
Descripción: Se prueba la lógica del sistema manejador de registro para verificar su correcto funcionamiento.		Enfoque: Crear Usuario			
Datos de Entrada: <code>\$_POST['nombre']</code> <code>\$_POST['pass']</code> <code>\$_POST['email']</code> <code>\$_POST['modelo']</code>		Datos De Salida (Esperados): <code>\$sql="insert into usuarios (nombre, pass, email, modelos)</code> <code>values(\$_POST['nombre'],</code> <code>\$_POST['pass'], \$_POST['email'],</code> <code>\$_POST['modelo'])"</code>			
Procedimiento					
1.se envían las variables a la página para procesar el acceso al sistema 2.Se muestra la página con las variables seteadas					
Precondiciones:					
Se modifica la página de salida para ver el valor la consulta sql					
Resultados					
Resultados Obtenidos: Se muestra correctamente las variables	Estado				
	Aprobado	X			
	No Aprobado				
	De no aprobar especificar falla:				
	Grave				
	Menor				
Observaciones					
Sin observaciones					

Figura C.10: Crear Usuario - Lógica

C.7. Crear Usuario - Persistencia

Prueba Unitaria					
Fecha	29 de agosto de 2011	Nº	10		
Nombre	Crear Usuario (persistencia)	Módulo	Sistema Conex		
Información Caso de Prueba					
Descripción: Se prueba la persistencia del sistema crear usuario para verificar su correcto funcionamiento.		Enfoque: Crear usuario			
Datos de Entrada: <code>\$sql="insert into usuarios (nombre, pass, email, modelo) values ('\$_POST['nombre']', '\$_POST['pass']', '\$_POST['email']', '\$_POST['modelos']'"</code>		Datos De Salida (Esperados): Mensaje de acceso exitoso			
Procedimiento					
1.se envían las sentencias sql a la página que se comunica con mysql 2.Se muestra la página con el mensaje del acceso a la base de datos					
Precondiciones:					
Se modifica la página de salida para ver el valor la consulta sql					
Resultados					
Resultados Obtenidos: Se muestra correctamente las variables	Estado				
	Aprobado	X			
	No Aprobado				
	De no aprobar especificar falla:				
	Grave				
	Menor				
Observaciones					
Sin observaciones					

Figura C.11: Crear Usuario - Persistencia

C.8. Modificar Datos Usuario - Interfaz

Prueba Unitaria					
Fecha	29 de agosto de 2011	Nº	3		
Nombre	Modificar Datos Usuario (Interfaz)	Módulo	Sistema Perfil		
Información Caso de Prueba					
Descripción: Se prueba la Interfaz del sistema de Perfil para verificar su correcto funcionamiento.		Enfoque: Editar Datos de Usuario			
Datos de Entrada: Nombre de usuario. Contraseña. Email Mis Modelos {público, privado}		Datos De Salida (Esperados): <code>\$_POST['nombre']</code> <code>\$_POST['pass']</code> <code>\$_POST['email']</code> <code>\$_POST['modelo']</code>			
Procedimiento					
1.Se ingresa a la página de Registro 2.Se ingresa el nombre de usuario, contraseña, email y elegir tipo de modelo 3.Se muestra la página con las variables seteadas					
Precondiciones:					
Se modifica la página de salida para ver el valor de las variables Se debe haber iniciado sesión como usuario o administrador					
Resultados					
Resultados Obtenidos: Se muestra correctamente las variables	Estado				
	Aprobado	X			
No Aprobado De no aprobar especificar falla:					
Grave Menor					
Observaciones					
Sin observaciones					

Figura C.12: Modificar Datos Usuario - Interfaz

C.9. Modificar Datos Usuario - Lógica

Prueba Unitaria					
Fecha	29 de agosto de 2011	Nº	7		
Nombre	Modificar datos Usuario (lógica)	Módulo	Sistema Editar Datos Personales		
Información Caso de Prueba					
Descripción: Se prueba la lógica del sistema editar datos personales para verificar su correcto funcionamiento.		Enfoque: Modificar datos de Usuario			
Datos de Entrada: \$_POST['nombre'] \$_POST['pass'] \$_POST['email'] \$_POST['modelo']		Datos De Salida (Esperados): \$sql="update usuarios set nombre=\$_POST['nombre'], pass=\$_POST['pass'], email=\$_POST['email'], modelos=\$_POST['modelo']"			
Procedimiento					
1.se envían las variables a la página para procesar la edición de datos 2.Se muestra la página con las variables seteadas					
Precondiciones:					
Se modifica la página de salida para ver el valor la consulta sql Se debe estar autenticado en el sistema como usuario o administrador					
Resultados					
Resultados Obtenidos: Se muestra correctamente las variables	Estado				
	Aprobado	X			
No Aprobado De no aprobar especificar falla:					
Grave Menor					
Observaciones					
Sin observaciones					

Figura C.13: Modificar Datos Usuario - Lógica

C.10. Modificar Datos Usuario - Persistencia

Prueba Unitaria					
Fecha	29 de agosto de 2011	Nº	11		
Nombre	Modificar Datos Usuario (persistencia)	Módulo	Sistema Conex		
Información Caso de Prueba					
Descripción: Se prueba la persistencia del sistema modificar datos usuario para verificar su correcto funcionamiento.		Enfoque: Modificar datos del usuario			
Datos de Entrada: <code>\$sql="update usuarios set nombre=\$_POST['nombre'], pass=\$_POST['pass'], email=\$_POST['email'], modelo=\$_POST['modelos']"</code>		Datos De Salida (Esperados): Mensaje de acceso exitoso			
Procedimiento					
1.se envían las sentencias sql a la página que se comunica con mysql 2.Se muestra la página con el mensaje del acceso a la base de datos					
Precondiciones:					
Se modifica la página de salida para ver el valor la consulta sql Se debe estar logueado como usuario o administrador					
Resultados					
Resultados Obtenidos: Se muestra correctamente las variables	Estado				
	Aprobado	X			
	No Aprobado				
	De no aprobar especificar falla:				
	Grave				
Observaciones					
Sin observaciones					

Figura C.14: Modificar Datos Usuario - Persistencia

C.11. Eliminar Usuario - Interfaz

Prueba Unitaria					
Fecha	29 de agosto de 2011	Nº	4		
Nombre	Eliminar Usuario (Interfaz)	Módulo	Sistema Perfil		
Información Caso de Prueba					
Descripción: Se prueba la Interfaz del sistema de Perfil para verificar su correcto funcionamiento.		Enfoque: Eliminar Usuario			
Datos de Entrada: Click en botón eliminar		Datos De Salida (Esperados): \$_POST['id']			
Procedimiento					
1.Se ingresa a la página de administrador 2.Se hace click en el botón borrar al lado del nombre del usuario que se desea eliminar 3.Se muestra la página con las variables seteadas					
Precondiciones:					
Se modifica la página de salida para ver el valor de las variables Se debe haber iniciado sesión como administrador					
Resultados					
Resultados Obtenidos:		Estado			
Se muestra correctamente las variables		Aprobado	X		
		No Aprobado			
		De no aprobar especificar falla:			
		Grave			
		Menor			
Observaciones					
Sin observaciones					

Figura C.15: Eliminar Usuario - Interfaz

C.12. Eliminar Usuario - Lógica

Prueba Unitaria					
Fecha	29 de agosto de 2011	Nº	8		
Nombre	Eliminar Usuario (lógica)	Módulo	Sistema Eliminar Usuarios		
Información Caso de Prueba					
Descripción: Se prueba la lógica del sistema eliminar usuarios para verificar su correcto funcionamiento.		Enfoque: Eliminar Usuario			
Datos de Entrada: \$_POST['id']		Datos De Salida (Esperados): \$sql="delete from usuarios where id=\$_POST['id']"			
Procedimiento					
1.se envían las variables a la página para procesar la edición de datos 2.Se muestra la página con las variables seteadas					
Precondiciones:					
Se modifica la página de salida para ver el valor la consulta sql Se debe estar autenticado en el sistema administrador					
Resultados					
Resultados Obtenidos: Se muestra correctamente las variables	Estado				
	Aprobado	X			
	No Aprobado				
	De no aprobar especificar falla:				
	Grave				
	Menor				
Observaciones					
Sin observaciones					

Figura C.16: Eliminar Usuario - Lógica

C.13. Eliminar Usuario - Persistencia

Prueba Unitaria					
Fecha	29 de agosto de 2011	Nº	12		
Nombre	Eliminar Usuario (persistencia)	Módulo	Sistema Conex		
Información Caso de Prueba					
Descripción: Se prueba la persistencia del sistema eliminar usuario para verificar su correcto funcionamiento.		Enfoque: Eliminar Usuario			
Datos de Entrada: <code>\$sql="delete from usuarios where id=\$_POST['id']"</code>		Datos De Salida (Esperados): Mensaje de acceso exitoso			
Procedimiento					
1.se envían las sentencias sql a la página que se comunica con mysql 2.Se muestra la página con el mensaje del acceso a la base de datos					
Precondiciones:					
Se modifica la página de salida para ver el valor la consulta sql Se debe estar logueado como administrador					
Resultados					
Resultados Obtenidos: Se muestra correctamente las variables	Estado		X		
	Aprobado				
No Aprobado De no aprobar especificar falla:					
Grave					
Menor					
Observaciones					
Sin observaciones					

Figura C.17: Eliminar Usuario - Persistencia

C.14. Pasar Siguiente/Anterior - Interfaz

Prueba Unitaria					
Fecha	29 de agosto de 2011	Nº	13		
Nombre	Pasar siguiente/anterior (Interfaz)	Módulo	Sistema Modelado		
Información Caso de Prueba					
Descripción: Se prueba la Interfaz del sistema pasar siguiente anterior para verificar su correcto funcionamiento.		Enfoque: Pasar diagrama siguiente y anterior			
Datos de Entrada: {click en diagramas}		Datos De Salida (Esperados): Var proximo=<siguiente o anterior> Var tipo=<modelo actual>			
Procedimiento					
1.se realiza click en las flechas que se encuentran al lado del nombre del modelo 2.Se muestra la página con los alert de las variables					
Precondiciones:					
Se modifica la página de salida para ver el valor de las variables a través de alert. Se debe estar logueado en el sistema					
Resultados					
Resultados Obtenidos: Se muestra correctamente las variables	Estado				
	Aprobado	X			
	No Aprobado				
	De no aprobar especificar falla:				
	Grave				
	Menor				
Observaciones					
Sin observaciones					

Figura C.18: Pasar Siguiente/Anterior - Interfaz

C.15. Pasar Siguiente/Anterior - Lógica

Prueba Unitaria					
Fecha	29 de agosto de 2011	Nº	18		
Nombre	Pasar siguiente/anterior (lógica)	Módulo	Sistema Modelado		
Información Caso de Prueba					
Descripción: Se prueba la Interfaz del sistema pasar siguiente/anterior para verificar su correcto funcionamiento.		Enfoque: Pasar siguiente/anterior modelo			
Datos de Entrada: Var proximo=<siguiente o anterior> Var tipo=< modelo actual>		Datos De Salida (Esperados): Modelo siguiente o anterior			
Procedimiento					
1. Se envían las variables para pasar de modelo 2. Se muestra el nuevo modelo					
Precondiciones:					
Se modifica la página para entregarle las variables para ver los modelos Se debe estar logueado en el sistema					
Resultados					
Resultados Obtenidos: Se muestra correctamente el modelo	Estado				
	Aprobado	X			
	No Aprobado				
	De no aprobar especificar falla:				
	Grave				
	Menor				
Observaciones					
Sin observaciones					

Figura C.19: Pasar Siguiente/Anterior - Lógica

C.16. Exportar Modelo - Interfaz

Prueba Unitaria					
Fecha	29 de agosto de 2011	Nº	14		
Nombre	Exportar Modelo (Interfaz)	Módulo	Sistema Modelado		
Información Caso de Prueba					
Descripción: Se prueba la Interfaz del sistema exportar modelo para verificar su correcto funcionamiento.		Enfoque: Exportar modelo			
Datos de Entrada: Nombre modelo {click en botón exportar}		Datos De Salida (Esperados): Var nombre=<nombre del modelo> Var format=<formato de exportación>			
Procedimiento					
1. Se llena el campo del nombre del modelo (por defecto es nombre proyecto) 2. Se presiona el botón para exportar el modelo 3. Se muestra la página con los alert de las variables					
Precondiciones:					
Se modifica la página de salida para ver el valor de las variables a través de alert. Se debe estar logueado en el sistema					
Resultados					
Resultados Obtenidos: Se muestra correctamente las variables	Estado				
	Aprobado	X			
	No Aprobado				
	De no aprobar especificar falla:				
	Grave				
	Menor				
Observaciones					
Sin observaciones					

Figura C.20: Exportar Modelo - Interfaz

C.17. Exportar Modelo - Lógica

Prueba Unitaria					
Fecha	29 de agosto de 2011	Nº	17		
Nombre	Exportar Modelo (lógica)	Módulo	Sistema Modelado		
Información Caso de Prueba					
Descripción: Se prueba la Interfaz del sistema configurar datos para verificar su correcto funcionamiento.		Enfoque: Configurar Datos			
Datos de Entrada: Var nombre=<Nombre del modelo> Var format=< formato de exportación>		Datos De Salida (Esperados): imagen			
Procedimiento					
1. Se en envían las variables para generar la imagen 2. Se muestran los directorios para guardar la imagen 3. Se genera la Imagen					
Precondiciones:					
Se modifica la página para entregarle las variables para generar la imagen Se debe estar logueado en el sistema					
Resultados					
Resultados Obtenidos: Se muestra correctamente la Imagen	Estado				
	Aprobado	X			
	No Aprobado				
	De no aprobar especificar falla:				
	Grave				
	Menor				
Observaciones					
Sin observaciones					

Figura C.21: Exportar Modelo - Lógica

C.18. Configurar Datos - Interfaz

Prueba Unitaria					
Fecha	29 de agosto de 2011	Nº	15		
Nombre	Configurar Datos (Interfaz)	Módulo	Sistema Modelado		
Información Caso de Prueba					
Descripción: Se prueba la Interfaz del sistema configurar datos para verificar su correcto funcionamiento.		Enfoque: Configurar Datos			
Datos de Entrada: Nombre y tipo (type) Message input name Message output name Porttype name Porttype operation name Service name Documentation name Port name services		Datos De Salida (Esperados): Var nombretipo=<Nombre tipo(type)> Var input=< Message input name> Var input=< Message output name> Var port=< Porttype name> Var portop=< Porttype operation name> Var ser=< Service name> Var doc=< Documentation name> Var portser=< Port name services>			
Procedimiento					
1. Se presiona en un vínculo 2. Se elige la opción para generar WSDL 3. Se llenan todos los campos para diseñar el WSDL 3. Se muestra la página con los alert de las variables					
Precondiciones: Se modifica la página de salida para ver el valor de las variables a través de alert. Se debe estar logueado en el sistema					
Resultados					
Resultados Obtenidos: Se muestra correctamente las variables	Estado				
	Aprobado	X			
	No Aprobado				
	De no aprobar especificar falla:				
	Grave				
	Menor				
Observaciones					
Sin observaciones					

Figura C.22: Configurar Datos - Interfaz

C.19. Configurar Datos - Lógica

Prueba Unitaria					
Fecha	29 de agosto de 2011	Nº	16		
Nombre	Configurar Datos (lógica)	Módulo	Sistema Modelado		
Información Caso de Prueba					
Descripción: Se prueba la Interfaz del sistema configurar datos para verificar su correcto funcionamiento.		Enfoque: Configurar Datos			
Datos de Entrada: Var nombretipo=<Nombre tipo(type)> Var input=< Message input name> Var input=< Message output name> Var port=< Porttype name> Var portop=< Porttype operation name> Var ser=< Service name> Var doc=< Documentation name> Var portser=< Port name services>		Datos De Salida (Esperados): WSDL			
Procedimiento					
1. se envían las variables para generar el WSDL 2. Se muestra la página con el WSDL en formato XML					
Precondiciones:					
Se modifica la página para entregarle las variables para generar al WSDL Se debe estar logueado en el sistema					
Resultados					
Resultados Obtenidos: Se muestra correctamente las variables	Estado				
	Aprobado	X			
	No Aprobado				
	De no aprobar especificar falla:				
	Grave				
Observaciones					
Sin observaciones					

Figura C.23: Configurar Datos - Lógica

Apéndice D

Pruebas de Integración

A continuación se detallan los resultados obtenidos al realizar las pruebas de integración:

Prueba de Integración						
Fecha	1 de Septiembre de 2011	Nº	1			
Nombre	Integración 1					
Información Caso de Prueba de Integración						
Descripción: Se prueba el funcionamiento de los módulos según el diagrama de integración propuesta en la sección de diseño	Enfoque: Validar integración de los módulos					
Procedimiento						
1. Se toman Ambos Módulos y se valida su funcionamiento						
Precondiciones:						
Los Módulos deben estar aprobados con cada caso de prueba						
Resultados						
Resultados Obtenidos: Se muestra correctamente el modelo	Estado					
	Aprobado	X				
	No Aprobado					
	De no aprobar especificar falla:					
	Grave					
	Menor					
Observaciones						
Sin observaciones						

Figura D.1: Prueba Integración 1

Prueba de Integración					
Fecha	1 de Septiembre de 2011	Nº	2		
Nombre		Integración 2			
Información Caso de Prueba de Integración					
Descripción: Se prueba el funcionamiento de los módulos según el diagrama de integración propuesta en la sección de diseño		Enfoque: Validar integración de los módulos			
Procedimiento					
1. Se toman Ambos Módulos y se valida su funcionamiento					
Precondiciones:					
Los Módulos deben estar aprobados con cada caso de prueba					
Resultados					
Resultados Obtenidos: Se muestra correctamente el modelo	Estado				
	Aprobado		X		
	No Aprobado				
	De no aprobar especificar falla:				
	Grave				
Observaciones Sin observaciones	Menor				

Figura D.2: Prueba Integración 2

Prueba de Integración					
Fecha	1 de Septiembre de 2011	Nº	3		
Nombre		Integración 3			
Información Caso de Prueba de Integración					
Descripción: Se prueba el funcionamiento de los módulos según el diagrama de integración propuesta en la sección de diseño		Enfoque: Validar integración de los módulos			
Procedimiento					
1. Se toman Ambos Módulos y se valida su funcionamiento					
Precondiciones:					
Los Módulos deben estar aprobados con cada caso de prueba					
Resultados					
Resultados Obtenidos: Se muestra correctamente el modelo	Estado				
	Aprobado		X		
	No Aprobado				
	De no aprobar especificar falla:				
	Grave				
Observaciones Sin observaciones	Menor				

Figura D.3: Prueba Integración 3

Prueba de Integración					
Fecha	1 de Septiembre de 2011	Nº	4		
Nombre		Integración 4			
Información Caso de Prueba de Integración					
Descripción: Se prueba el funcionamiento de los módulos según el diagrama de integración propuesta en la sección de diseño		Enfoque: Validar integración de los módulos			
Procedimiento					
1. Se toman Ambos Módulos y se valida su funcionamiento					
Precondiciones:					
Los Módulos deben estar aprobados con cada caso de prueba					
Resultados					
Resultados Obtenidos: Se muestra correctamente el modelo	Estado				
	Aprobado		X		
	No Aprobado				
	De no aprobar especificar falla:				
	Grave				
Observaciones Sin observaciones	Menor				

Figura D.4: Prueba Integración 4

Prueba de Integración					
Fecha	1 de Septiembre de 2011	Nº	5		
Nombre		Integración 5			
Información Caso de Prueba de Integración					
Descripción: Se prueba el funcionamiento de los módulos según el diagrama de integración propuesta en la sección de diseño		Enfoque: Validar integración de los módulos			
Procedimiento					
1. Se toman Ambos Módulos y se valida su funcionamiento					
Precondiciones:					
Los Módulos deben estar aprobados con cada caso de prueba					
Resultados					
Resultados Obtenidos: Se muestra correctamente el modelo	Estado				
	Aprobado		X		
	No Aprobado				
	De no aprobar especificar falla:				
	Grave				
Observaciones Sin observaciones	Menor				

Figura D.5: Prueba Integración 5

Prueba de Integración					
Fecha	1 de Septiembre de 2011	Nº	6		
Nombre		Integración 6			
Información Caso de Prueba de Integración					
Descripción: Se prueba el funcionamiento de los módulos según el diagrama de integración propuesta en la sección de diseño		Enfoque: Validar integración de los módulos			
Procedimiento					
1. Se toman Ambos Módulos y se valida su funcionamiento					
Precondiciones:					
Los Módulos deben estar aprobados con cada caso de prueba					
Resultados					
Resultados Obtenidos: Se muestra correctamente el modelo	Estado				
	Aprobado		X		
	No Aprobado				
	De no aprobar especificar falla:				
	Grave				
Observaciones Sin observaciones	Menor				

Figura D.6: Prueba Integración 6

Prueba de Integración					
Fecha	1 de Septiembre de 2011	Nº	7		
Nombre		Integración 7			
Información Caso de Prueba de Integración					
Descripción: Se prueba el funcionamiento de los módulos según el diagrama de integración propuesta en la sección de diseño		Enfoque: Validar integración de los módulos			
Procedimiento					
1. Se toman Ambos Módulos y se valida su funcionamiento					
Precondiciones:					
Los Módulos deben estar aprobados con cada caso de prueba					
Resultados					
Resultados Obtenidos: Se muestra correctamente el modelo	Estado				
	Aprobado	X			
	No Aprobado				
	De no aprobar especificar falla:				
	Grave				
Observaciones Sin observaciones	Menor				

Figura D.7: Prueba Integración 7

Prueba de Integración					
Fecha	1 de Septiembre de 2011	Nº	8		
Nombre		Integración	8		
Información Caso de Prueba de Integración					
Descripción: Se prueba el funcionamiento de los módulos según el diagrama de integración propuesta en la sección de diseño		Enfoque: Validar integración de los módulos			
Procedimiento					
1. Se toman Ambos Módulos y se valida su funcionamiento					
Precondiciones:					
Los Módulos deben estar aprobados con cada caso de prueba					
Resultados					
Resultados Obtenidos: Se muestra correctamente el modelo	Estado				
	Aprobado		X		
	No Aprobado				
	De no aprobar especificar falla:				
	Grave				
Observaciones Sin observaciones	Menor				

Figura D.8: Prueba Integración 8

Prueba de Integración					
Fecha	1 de Septiembre de 2011	Nº	9		
Nombre		Integración	9		
Información Caso de Prueba de Integración					
Descripción: Se prueba el funcionamiento de los módulos según el diagrama de integración propuesta en la sección de diseño		Enfoque: Validar integración de los módulos			
Procedimiento					
1. Se toman Ambos Módulos y se valida su funcionamiento					
Precondiciones:					
Los Módulos deben estar aprobados con cada caso de prueba					
Resultados					
Resultados Obtenidos: Se muestra correctamente el modelo	Estado				
	Aprobado		X		
	No Aprobado				
	De no aprobar especificar falla:				
	Grave				
Observaciones Sin observaciones	Menor				

Figura D.9: Prueba Integración 9

Prueba de Integración						
Fecha	1 de Septiembre de 2011	Nº	10			
Nombre	Integración 10					
Información Caso de Prueba de Integración						
Descripción: Se prueba el funcionamiento de los módulos según el diagrama de integración propuesta en la sección de diseño	Enfoque: Validar integración de los módulos					
Procedimiento						
1. Se toman Ambos Módulos y se valida su funcionamiento						
Precondiciones:						
Los Módulos deben estar aprobados con cada caso de prueba						
Resultados						
Resultados Obtenidos: Se muestra correctamente el modelo	Estado					
	Aprobado	X				
	No Aprobado					
	De no aprobar especificar falla:					
	Grave					
	Menor					
Observaciones						
Sin observaciones						

Figura D.10: Prueba Integración 10

Prueba de Integración						
Fecha	1 de Septiembre de 2011	Nº	11			
Nombre	Integración 11					
Información Caso de Prueba de Integración						
Descripción: Se prueba el funcionamiento de los módulos según el diagrama de integración propuesta en la sección de diseño	Enfoque: Validar integración de los módulos					
Procedimiento						
1. Se toman Ambos Módulos y se valida su funcionamiento						
Precondiciones:						
Los Módulos deben estar aprobados con cada caso de prueba						
Resultados						
Resultados Obtenidos: Se muestra correctamente el modelo	Estado					
	Aprobado	X				
	No Aprobado					
	De no aprobar especificar falla:					
	Grave					
	Menor					
Observaciones						
Sin observaciones						

Figura D.11: Prueba Integración 11

Prueba de Integración					
Fecha	1 de Septiembre de 2011	Nº	12		
Nombre	Integración 12				
Información Caso de Prueba de Integración					
Descripción: Se prueba el funcionamiento de los módulos según el diagrama de integración propuesta en la sección de diseño		Enfoque: Validar integración de los módulos			
Procedimiento					
1. Se toman Ambos Módulos y se valida su funcionamiento					
Precondiciones:					
Los Módulos deben estar aprobados con cada caso de prueba					
Resultados					
Resultados Obtenidos: Se muestra correctamente el modelo	Estado				
	Aprobado	X			
	No Aprobado				
	De no aprobar especificar falla:				
	Grave				
Observaciones Sin observaciones	Menor				

Figura D.12: Prueba Integración 12

Prueba de Integración						
Fecha	1 de Septiembre de 2011	Nº	13			
Nombre	Integración 13					
Información Caso de Prueba de Integración						
Descripción: Se prueba el funcionamiento de los módulos según el diagrama de integración propuesta en la sección de diseño	Enfoque: Validar integración de los módulos					
Procedimiento						
1. Se toman Ambos Módulos y se valida su funcionamiento						
Precondiciones:						
Los Módulos deben estar aprobados con cada caso de prueba						
Resultados						
Resultados Obtenidos: Se muestra correctamente el modelo	Estado					
	Aprobado	X				
	No Aprobado					
	De no aprobar especificar falla:					
	Grave					
	Menor					
Observaciones						
Sin observaciones						

Figura D.13: Prueba Integración 13

Prueba de Integración						
Fecha	1 de Septiembre de 2011	Nº	14			
Nombre	Integración 14					
Información Caso de Prueba de Integración						
Descripción: Se prueba el funcionamiento de los módulos según el diagrama de integración propuesta en la sección de diseño	Enfoque: Validar integración de los módulos					
Procedimiento						
1. Se toman Ambos Módulos y se valida su funcionamiento						
Precondiciones:						
Los Módulos deben estar aprobados con cada caso de prueba						
Resultados						
Resultados Obtenidos: Se muestra correctamente el modelo	Estado					
	Aprobado	X				
	No Aprobado					
	De no aprobar especificar falla:					
	Grave					
	Menor					
Observaciones						
Sin observaciones						

Figura D.14: Prueba Integración 14

Prueba de Integración						
Fecha	1 de Septiembre de 2011	Nº	15			
Nombre	Integración 15					
Información Caso de Prueba de Integración						
Descripción: Se prueba el funcionamiento de los módulos según el diagrama de integración propuesta en la sección de diseño	Enfoque: Validar integración de los módulos					
Procedimiento						
1. Se toman Ambos Módulos y se valida su funcionamiento						
Precondiciones:						
Los Módulos deben estar aprobados con cada caso de prueba						
Resultados						
Resultados Obtenidos: Se muestra correctamente el modelo	Estado					
	Aprobado	X				
	No Aprobado					
	De no aprobar especificar falla:					
	Grave					
	Menor					
Observaciones						
Sin observaciones						

Figura D.15: Prueba Integración 15

Prueba de Integración					
Fecha	1 de Septiembre de 2011	Nº	16		
Nombre	Integración 16				
Información Caso de Prueba de Integración					
Descripción: Se prueba el funcionamiento de los módulos según el diagrama de integración propuesta en la sección de diseño		Enfoque: Validar integración de los módulos			
Procedimiento					
1. Se toman Ambos Módulos y se valida su funcionamiento					
Precondiciones:					
Los Módulos deben estar aprobados con cada caso de prueba					
Resultados					
Resultados Obtenidos: Se muestra correctamente el modelo	Estado				
	Aprobado	X			
	No Aprobado				
	De no aprobar especificar falla:				
	Grave				
Observaciones Sin observaciones	Menor				

Figura D.16: Prueba Integración 16

Prueba de Integración						
Fecha	1 de Septiembre de 2011	Nº	17			
Nombre	Integración 17					
Información Caso de Prueba de Integración						
Descripción: Se prueba el funcionamiento de los módulos según el diagrama de integración propuesta en la sección de diseño	Enfoque: Validar integración de los módulos					
Procedimiento						
1. Se toman Ambos Módulos y se valida su funcionamiento						
Precondiciones:						
Los Módulos deben estar aprobados con cada caso de prueba						
Resultados						
Resultados Obtenidos: Se muestra correctamente el modelo	Estado					
	Aprobado	X				
	No Aprobado					
	De no aprobar especificar falla:					
	Grave					
	Menor					
Observaciones						
Sin observaciones						

Figura D.17: Prueba Integración 17

Prueba de Integración						
Fecha	1 de Septiembre de 2011	Nº	18			
Nombre	Integración 18					
Información Caso de Prueba de Integración						
Descripción: Se prueba el funcionamiento de los módulos según el diagrama de integración propuesta en la sección de diseño	Enfoque: Validar integración de los módulos					
Procedimiento						
1. Se toman Ambos Módulos y se valida su funcionamiento						
Precondiciones:						
Los Módulos deben estar aprobados con cada caso de prueba						
Resultados						
Resultados Obtenidos: Se muestra correctamente el modelo	Estado					
	Aprobado	X				
	No Aprobado					
	De no aprobar especificar falla:					
	Grave					
	Menor					
Observaciones						
Sin observaciones						

Figura D.18: Prueba Integración 18

Apéndice E

Pruebas de Aceptación

E.1. Encuesta Usuario

A continuación mostraremos las respuestas de los usuarios por pregunta. Para ello elegimos la siguiente nomenclatura:

- Color Azul: Es el puntaje que cada usuario dio a la pregunta.
- Color Rojo: Es el resultado esperado mínimo que corresponde al numero 3.

Lo que se muestra entre la Figura E.1 y Figura E.11 son los resultados por pregunta de cada uno de los usuarios.

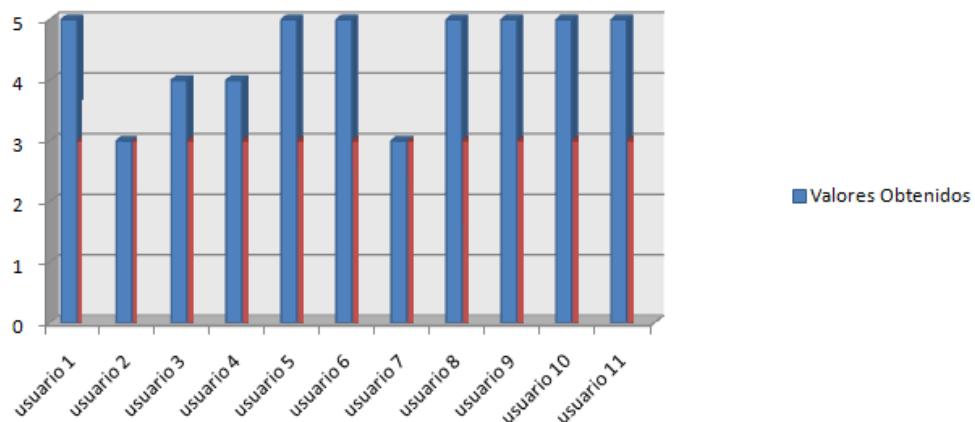


Figura E.1: Pregunta 1

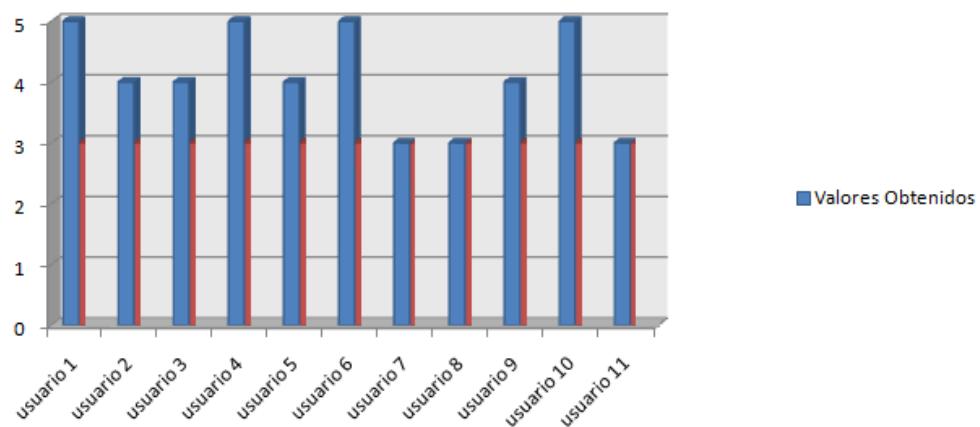


Figura E.2: Pregunta 2

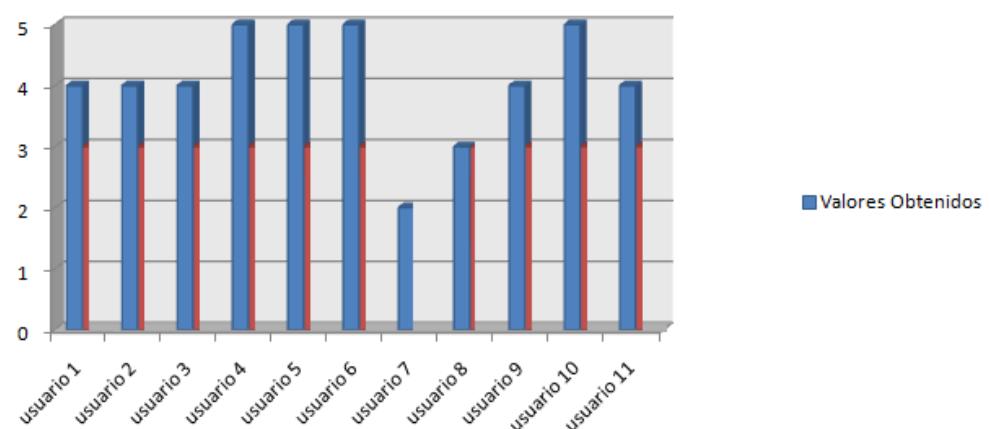


Figura E.3: Pregunta 3

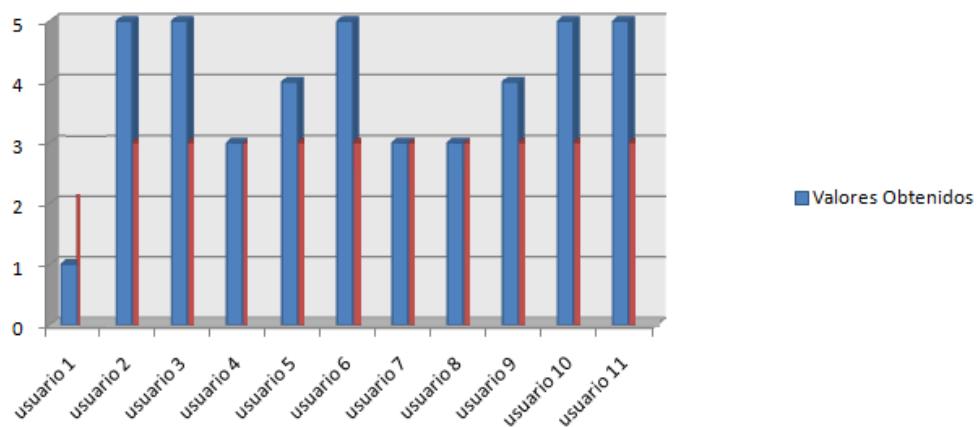


Figura E.4: Pregunta 4

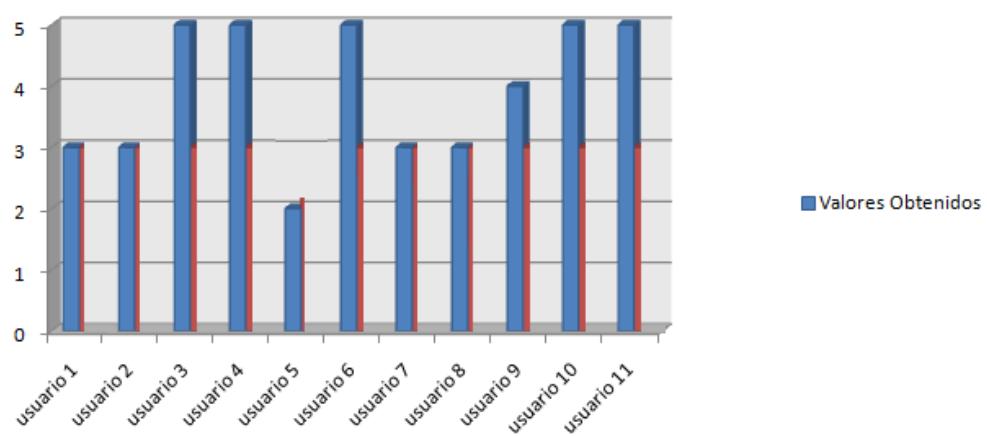


Figura E.5: Pregunta 5

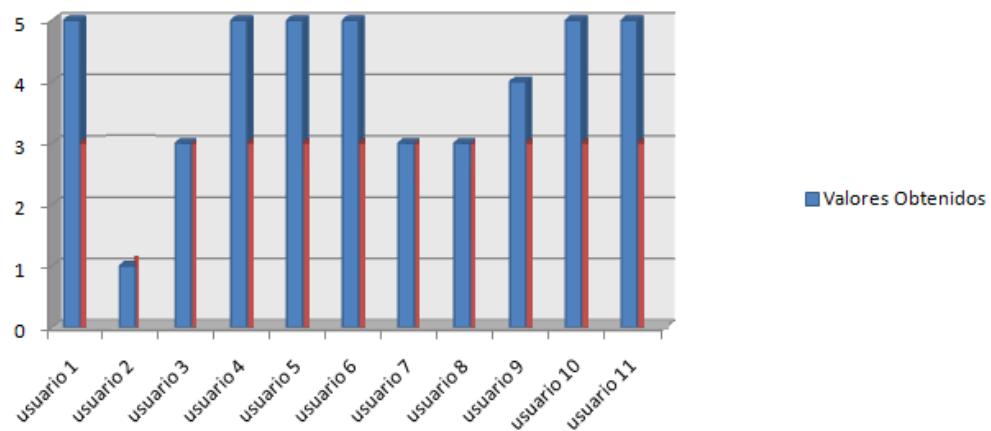


Figura E.6: Pregunta 6

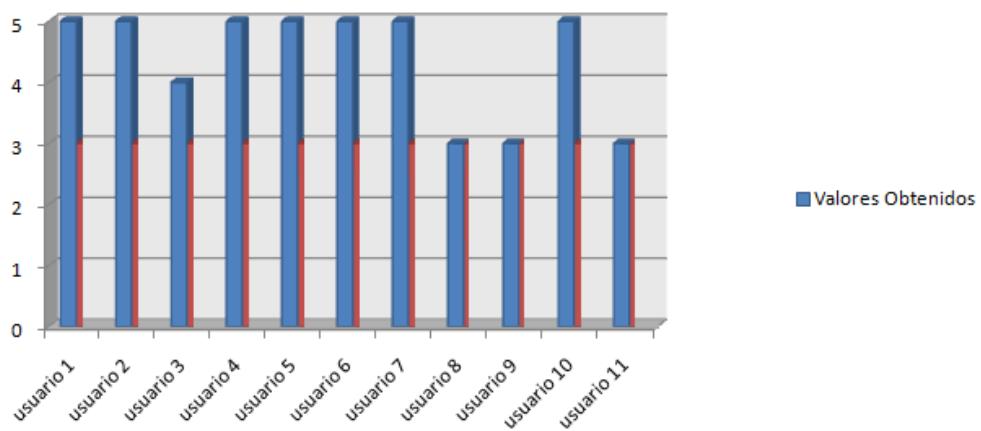


Figura E.7: Pregunta 7

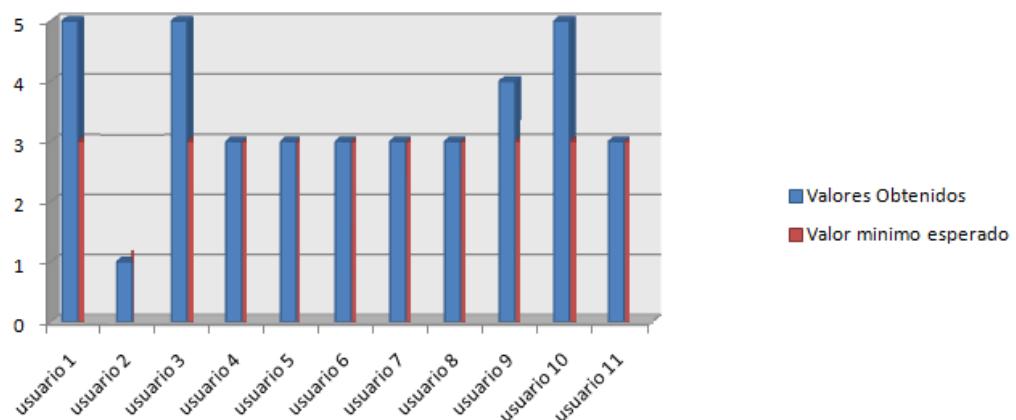


Figura E.8: Pregunta 8

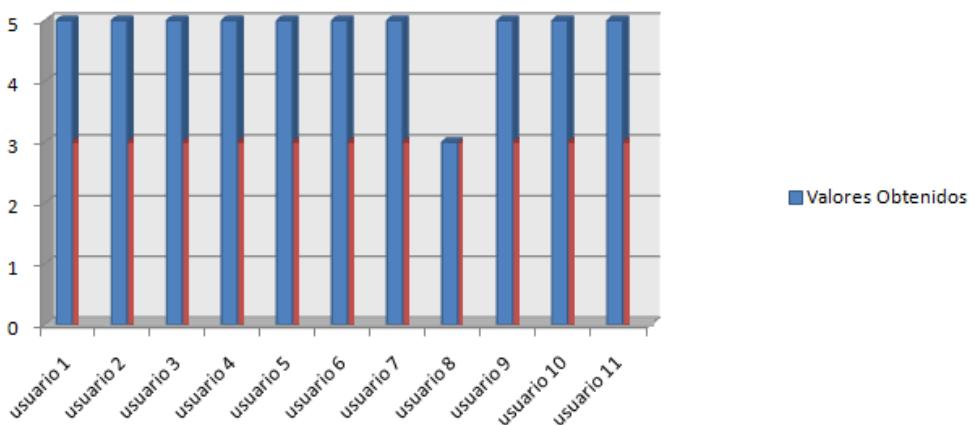


Figura E.9: Pregunta 9

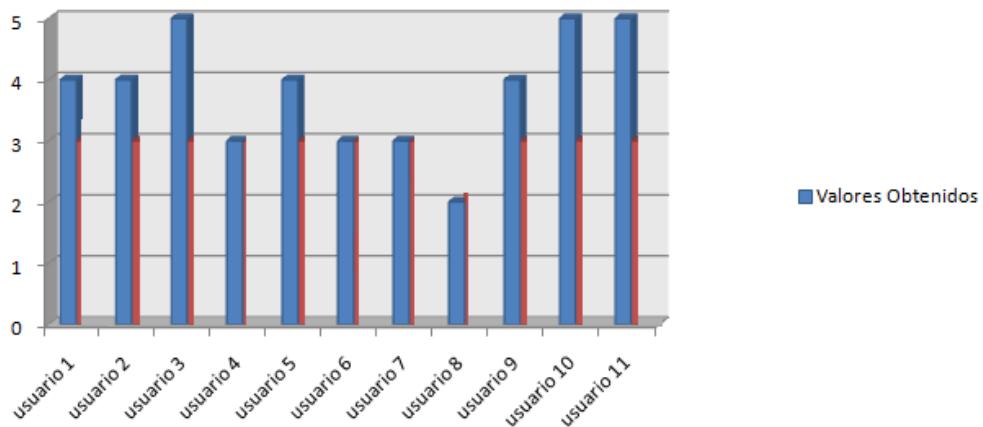


Figura E.10: Pregunta 10

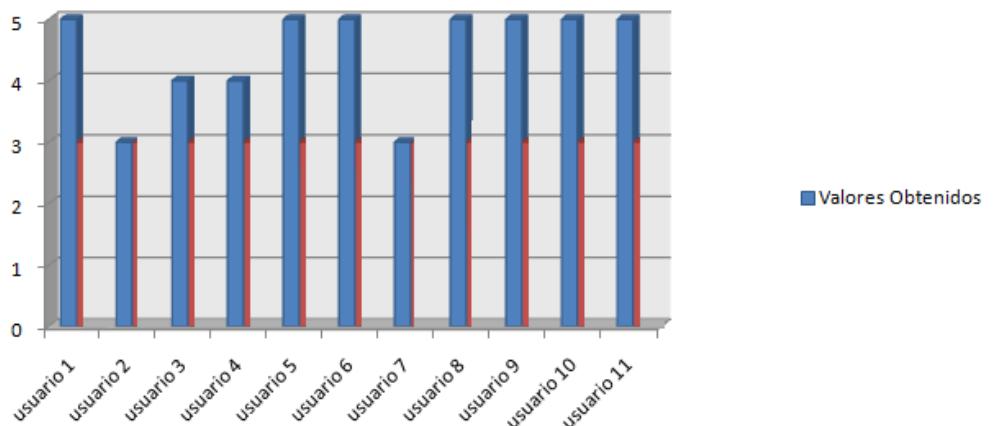


Figura E.11: Pregunta 11

También podemos ver el resumen en la Figura E.12.

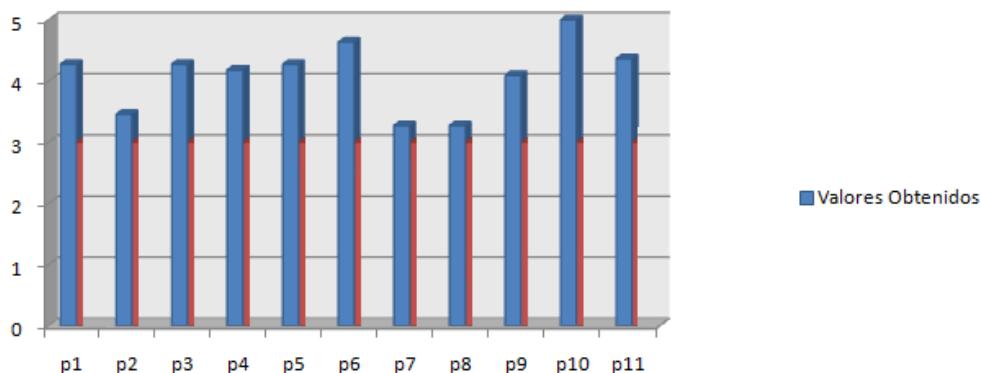


Figura E.12: Resumen de Preguntas de los usuarios

Podemos concluir que los resultados obtenidos fueron exitosos ya que estuvieron sobre el resultado mínimo esperado de 60 % en cada pregunta. Además se puede decir que el porcentaje final de aprobación estuvo sobre el 82 %.

También podemos determinar la mejor y peor pregunta, que fueron:

■ Peor Pregunta:

- ¿Se entiende que significa que los modelos sean públicos o privados?
- Tuvo una valoración promedio de 3.5 lo que equivale a un 70 % de aprobación.
- Los usuarios no entendieron que era un modelo público y privado hasta que se les dijo de que se trataba.

■ Mejor Pregunta:

- ¿Evalué la Simplicidad de la realización del registro?
- Tuvo una valoración promedio de 4.8 lo que equivale a un 96 % de aprobación.

E.2. Encuesta Administrador

A continuación mostraremos las respuestas de los usuarios por pregunta. Para ello elegimos la siguiente nomenclatura:

- Color Azul: Es el puntaje que cada usuario dio a la pregunta.

- Color Rojo: Es el resultado esperado mínimo que corresponde al 55

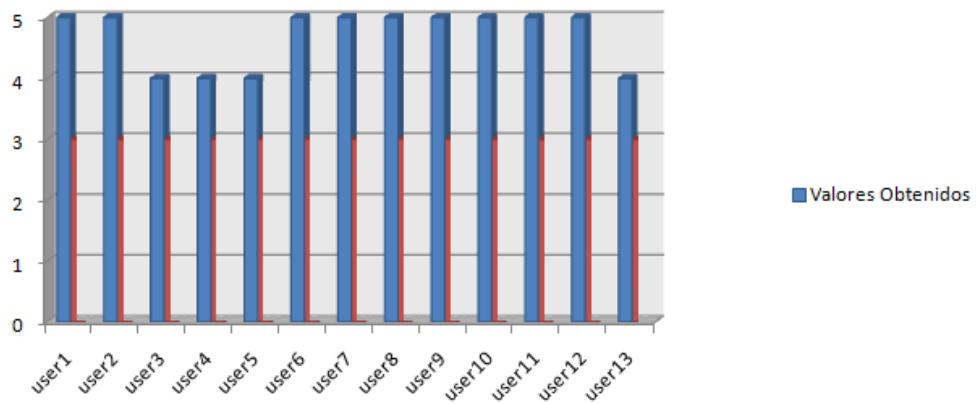


Figura E.13: Pregunta 1 Administrador

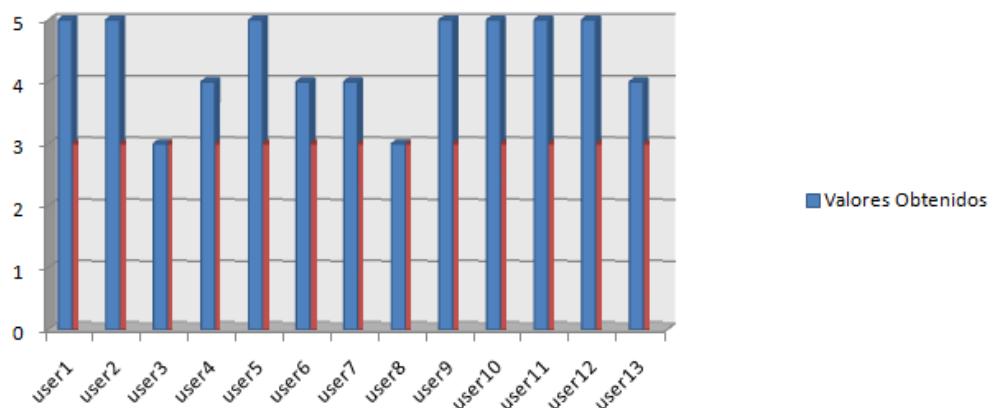


Figura E.14: Pregunta 2 Administrador

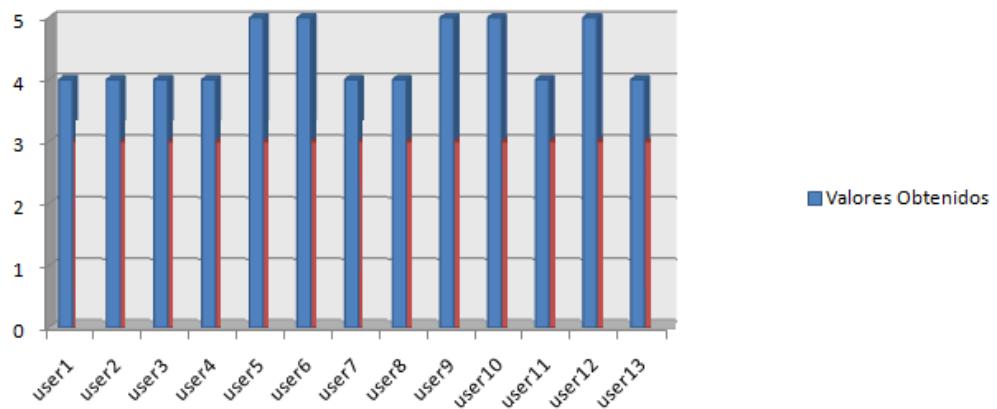


Figura E.15: Pregunta 3 Administrador

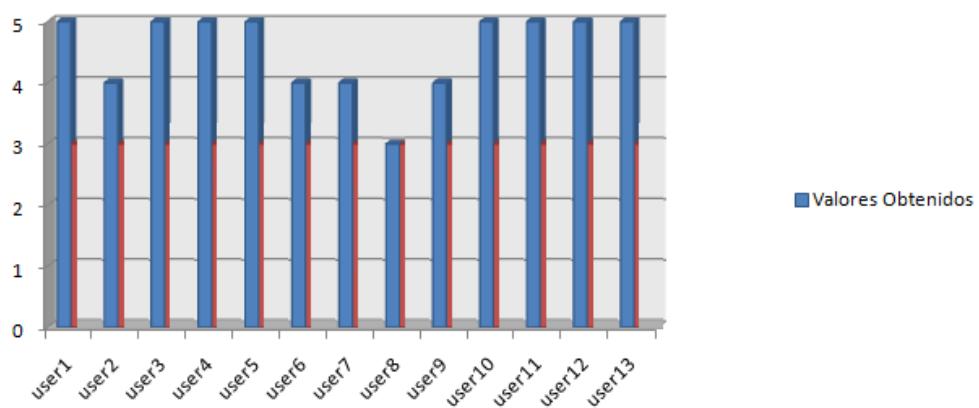


Figura E.16: Pregunta 4 Administrador

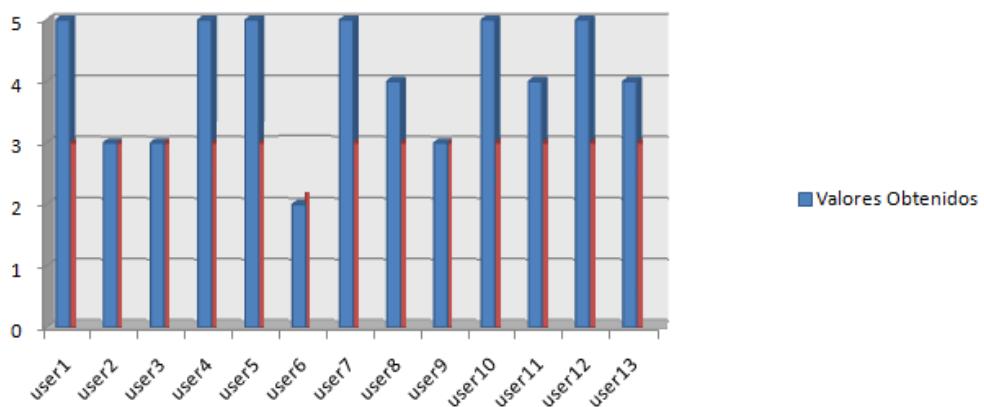


Figura E.17: Pregunta 5 Administrador

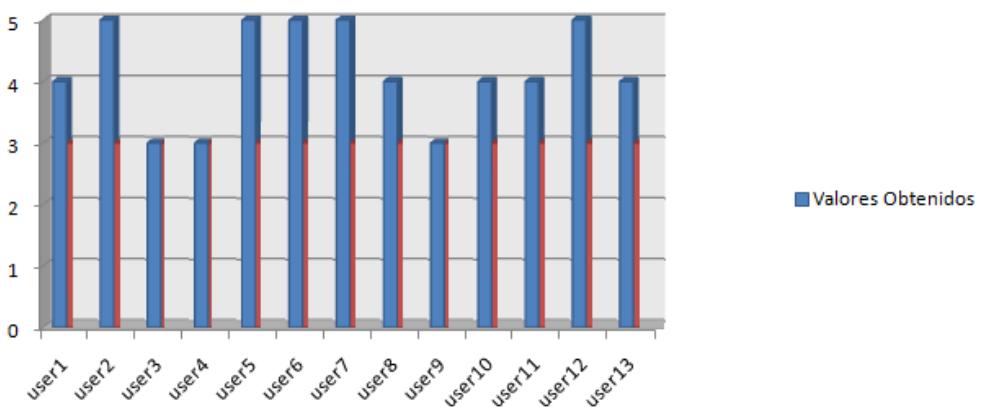


Figura E.18: Pregunta 6 Administrador

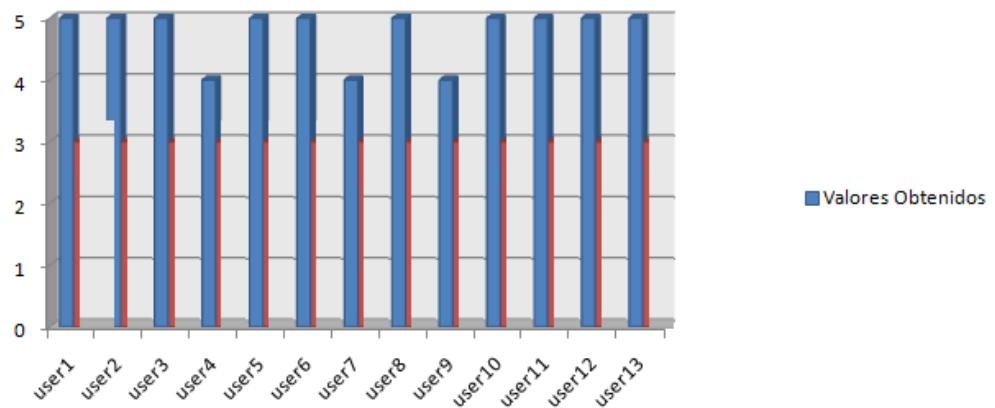


Figura E.19: Pregunta 7 Administrador

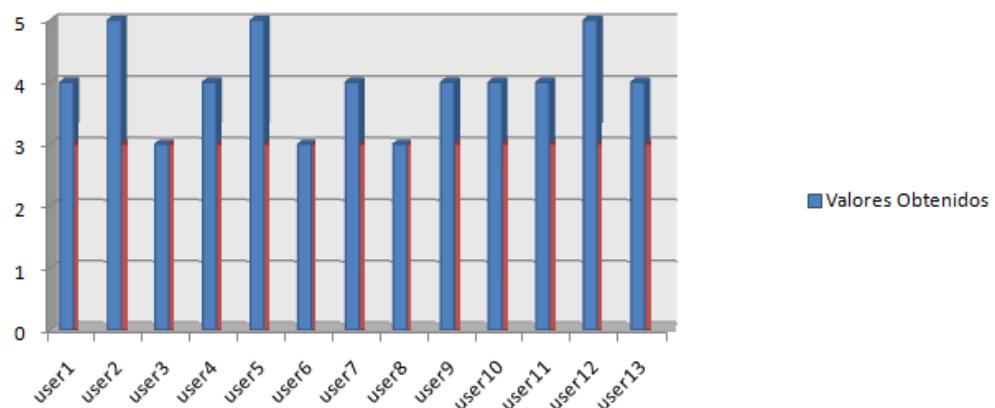


Figura E.20: Pregunta 8 Administrador

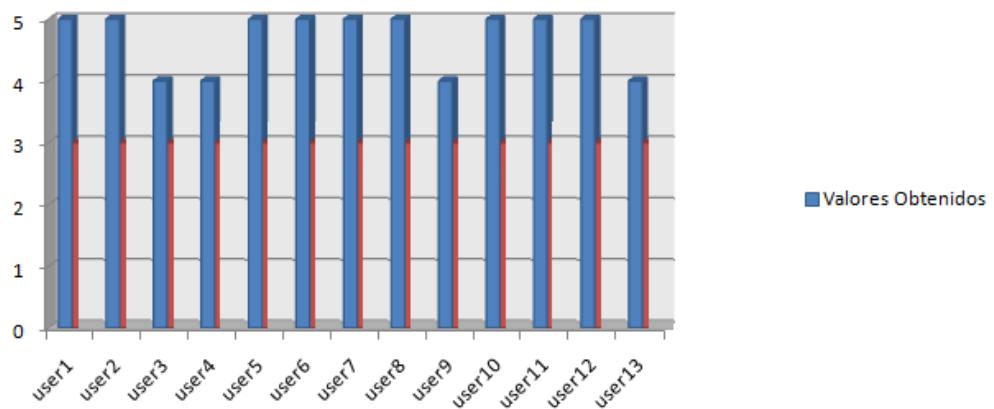


Figura E.21: Pregunta 9 Administrador

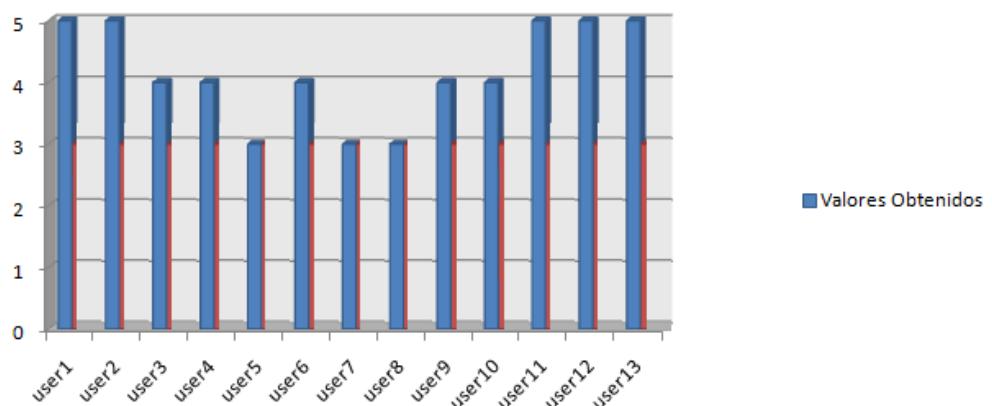


Figura E.22: Pregunta 10 Administrador

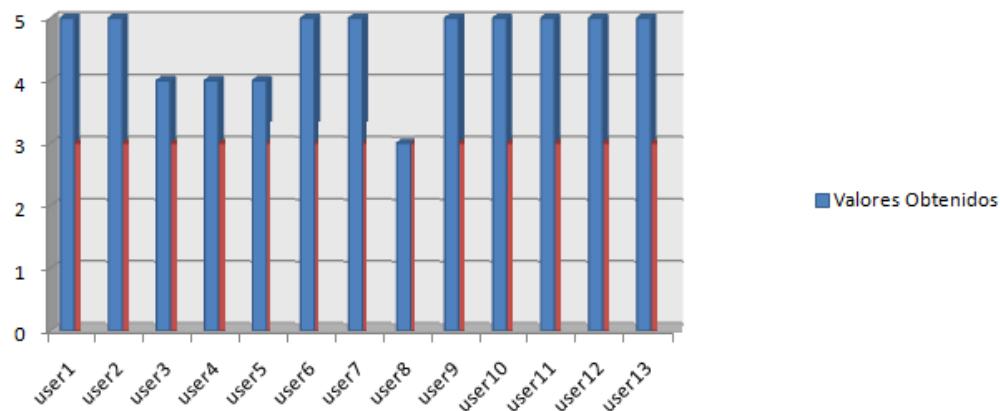


Figura E.23: Pregunta 11 Administrador

También podemos ver el resumen en la Figura E.24.

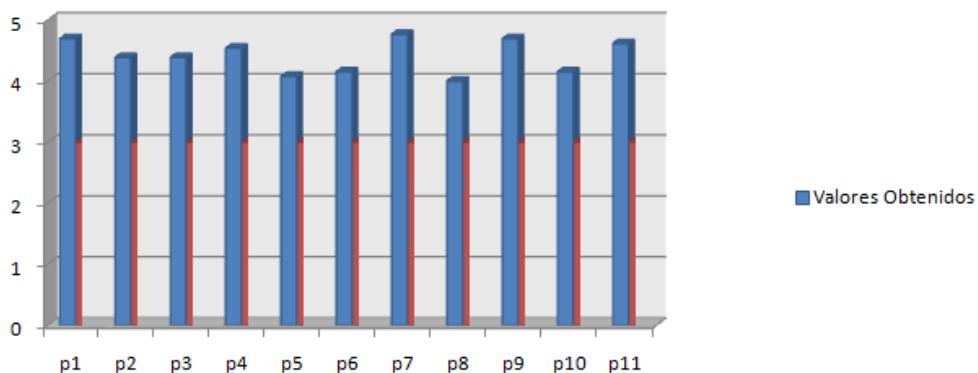


Figura E.24: Resumen de Preguntas de La sección Administrador

Podemos concluir que los resultados obtenidos fueron exitosos ya que estuvieron sobre el resultado esperado de cada pregunta. También podemos determinar la mejor y peor pregunta, que fueron:

- Peor Pregunta:
 - ¿Le parece bien los campos que se le exigen al usuario para el registro (enfoque seguridad)?

- Tuvo una valoración promedio de 4.0 lo que equivale a un 80 % de aprobación.
 - Los usuarios pensaron que los campos pedidos eran insuficientes y que se debiesen agregar otros como Nombre, Apellido, fecha de nacimiento, etc.
- Mejor Pregunta:
- ¿Evalué la gestión de usuarios (buscar, crear, eliminar, editar)?
 - Tuvo una valoración promedio de 4.8 lo que equivale a un 96 % de aprobación.