Oficina Tècnica de
Direcció de Projecte

Unión Europea
Fondo Europeo
de Desarrollo Regional

2007-2013 illes balears una manera de hacer europa

Govern de les Illes Balears
Conselleria d'Economia i Competitivitat
Direcció General d'Innovació
i Desenvolupament Tecnològic

[ANÀLISI TÈCNIC – GUSITE2014]

Índex

1. Arquitectura actual

La arquitectura actual de GUSITE sigue el esquema de aplicaciones webcaib, con múltiples dependencias con librerías de otros proyectos.

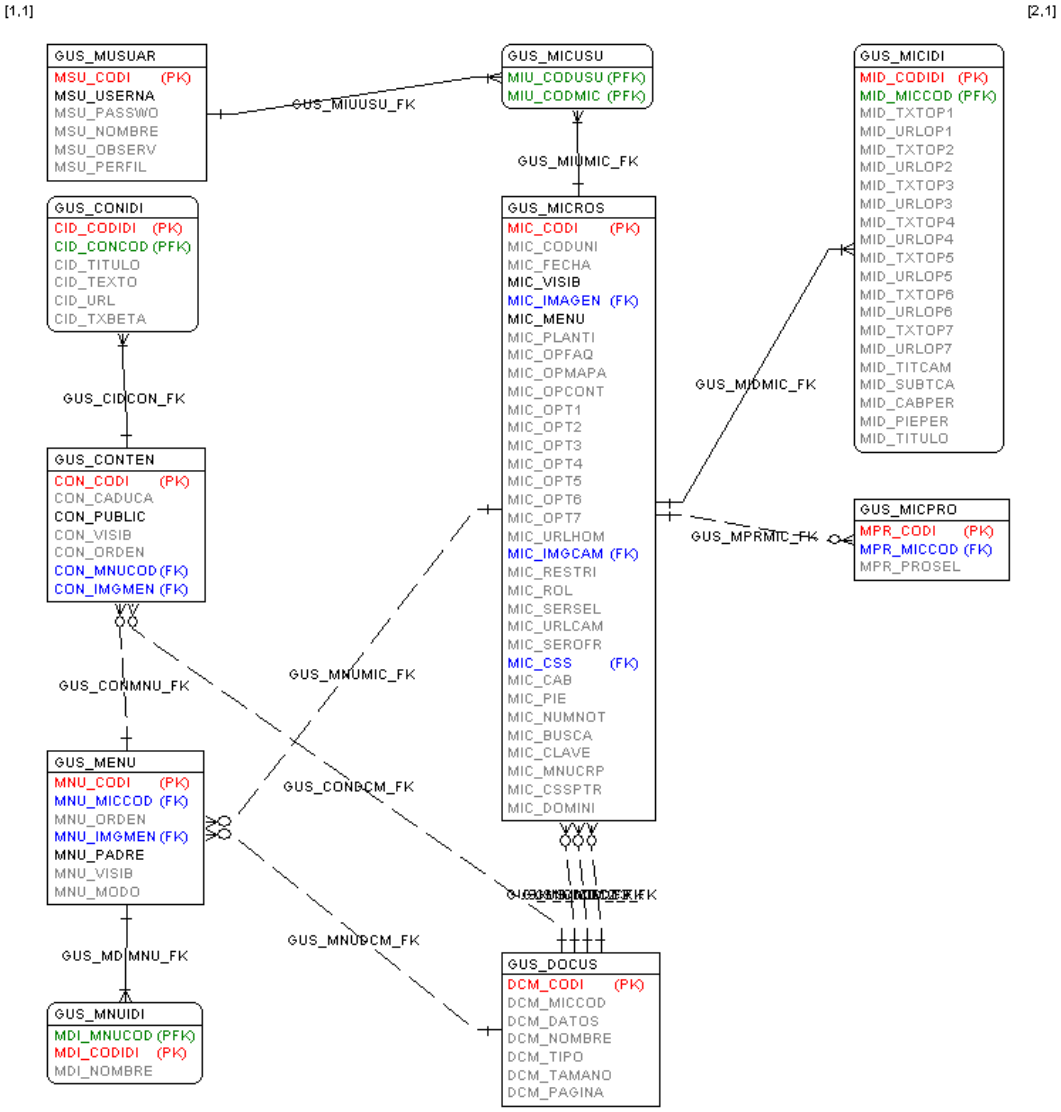
1.1. Modelo físico del esquema de datos

Para información más detallada, ver el informe adjunto: GUSITE-modelo-de-datos.zip

1.1.1. Microsite

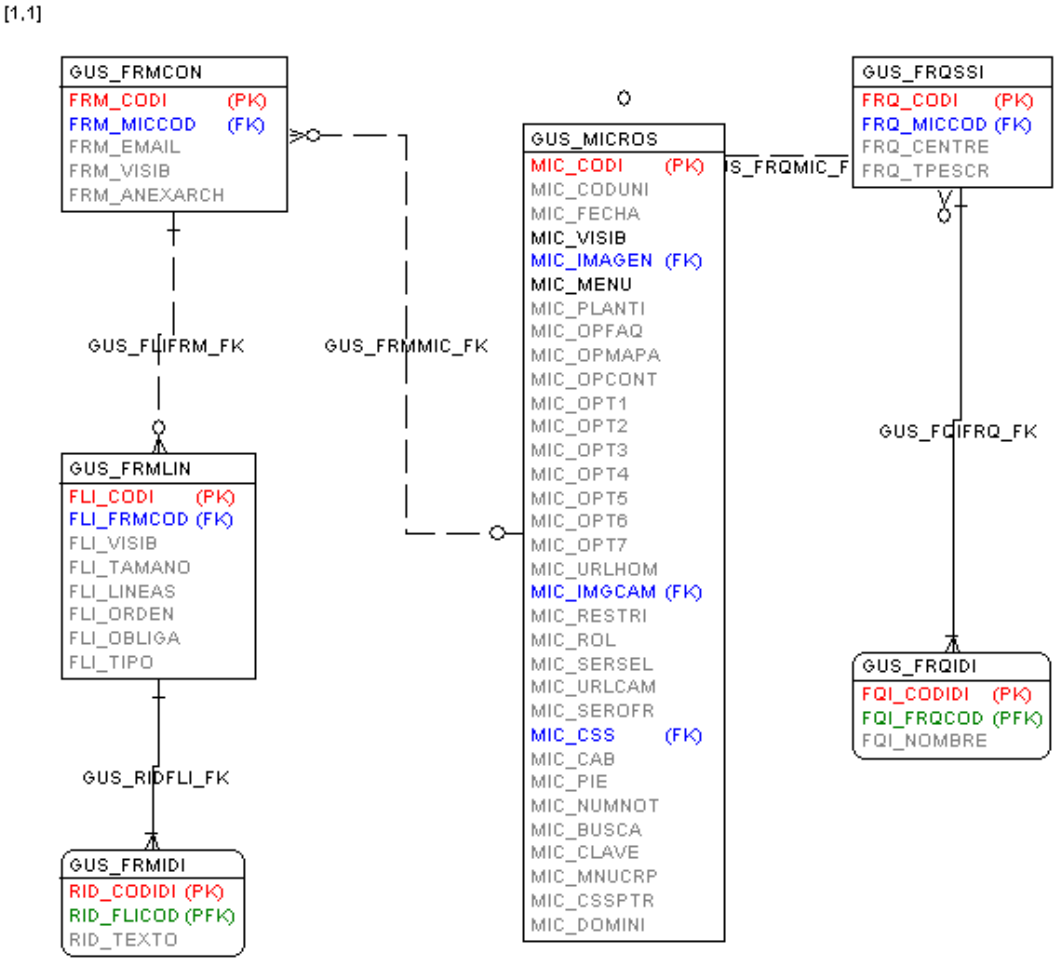
MICROS	definición del microsite con todos los atributos de caracter general que determinan su funcionamiento.
MICIDI	atributos dependientes de idioma para la definición de un elementos de un tipo de listado de un microsite.
MICPRO	relación de procedimientos o trámites asociados a un microsite.
MUSUAR	ficha de los usuarios del entorno de los microsite. El alta se realiza coincidir con un usuario de CAIB. Se utiliza para controlar que un usuario CAIB acceda un microsite con un perfil determinado.
MICUSU	relaciona un usuario con todos los microsites en los que tiene permiso, y el perfil con el que actual sobre él.
DOCUS	archivos (imágenes, pdf, doc, etc..) utilizados en los distintos elementos de un microsite.
MENU	definición del arbol de menú asociado a un microsite. Define los nodos de menú, su tipo y funcionamiento y enlaza con los contenidos del microsite.
MNUIDI	atributos dependientes de idioma para el menú de un microsite.
CONTEN	contenidos de un microsite. Son los contenidos estáticos o los enlaces externos asociados al menú.
CONIDI	atributos dependientes de idioma de los contenidos de un microsite.
COMPOS	registro de componentes sobre los listados definidos para un microsite. Son distintas presentaciones para un informe, que pueden incluirse en un contenido.
CMPIDI	atributos dependientes de idioma del registro de componentes de un microsite.
IDIOMA	lista de idiomas gestionados por los microsites. La activación de un nuevo idioma no solo supone el registro de esta tabla.
IDIMIC	indica los idiomas en que está disponible el microsite

STATS	registro de las estadísticas de acceso a los distintos elementos de un microsite.
W3C	contiene información sobre la accesibilidad de un microsite



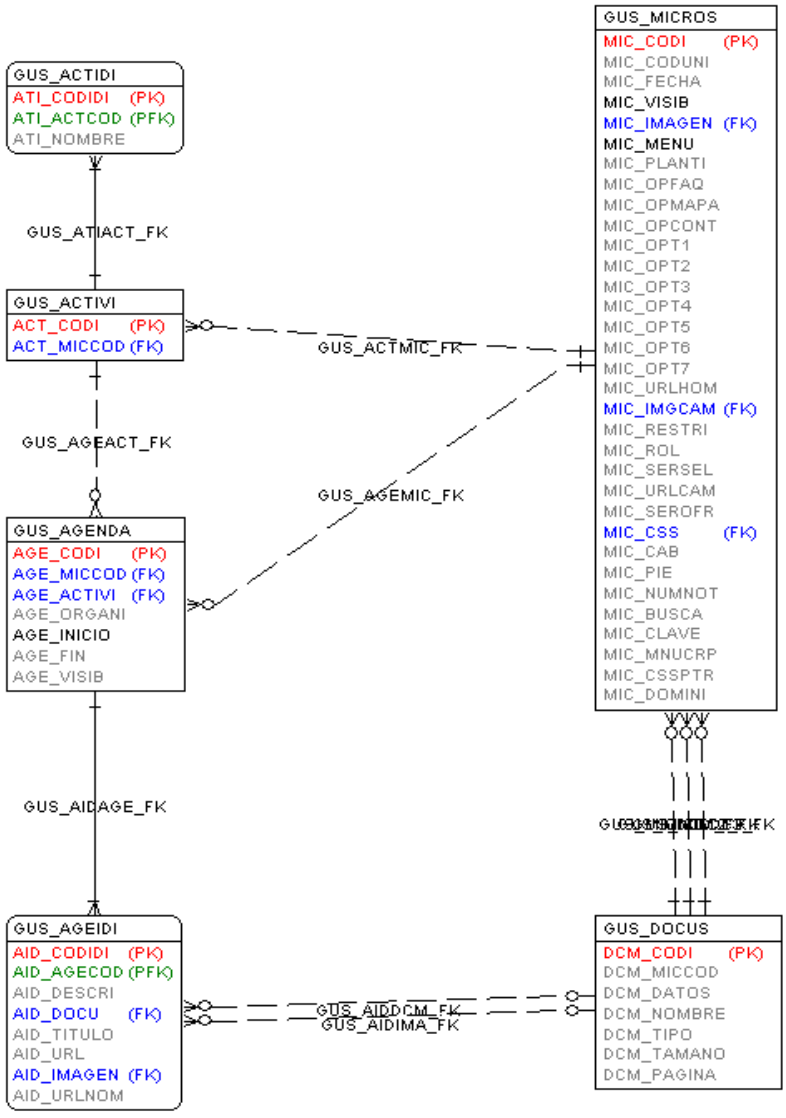
1.1.2. Formularios

FRMCON	registro de los formularios de contacto definidos para un microsite. Se compone de un número de campos que incluirá el formulario.
FRMIDI	atributos dependientes de idioma del registro de campos de un formulario de un microsite.
FRMLIN	relación de campos y su tipo asociados a un formularios de contacto definidos para un microsite.
FRQSSI	define un formulario QSSI
FRQIDI	atributos dependientes de idioma de un formulario QSSI



1.1.3. Agenda

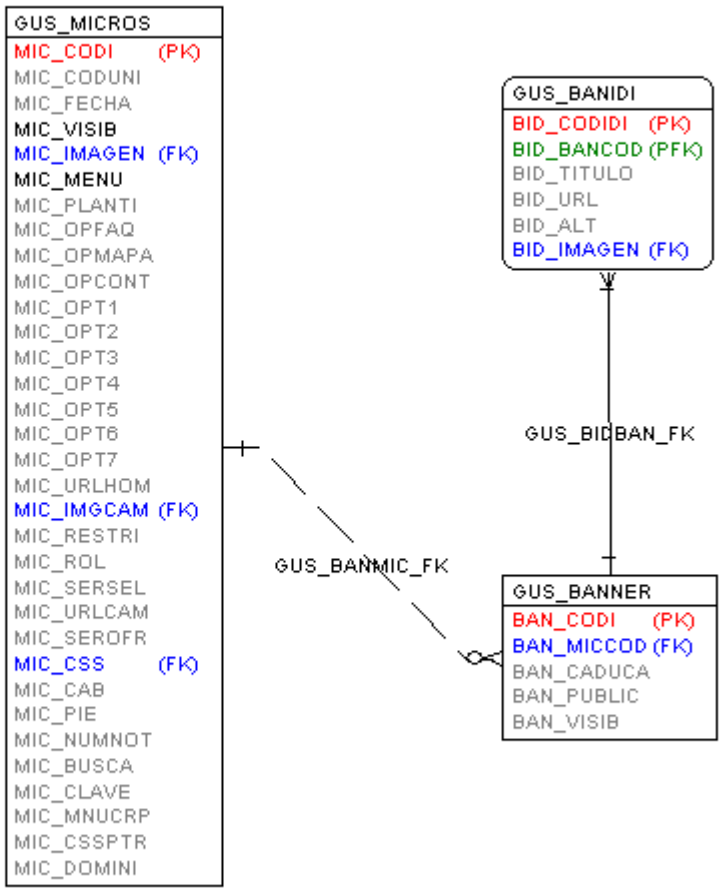
AGENDA	registros de la agenda de un microsite.
AGEIDI	atributos dependientes de idioma de los registros que incluiye una agenda de un microsite.
ACTIVI	tipos de registros que incluiye una agenda de un microsite.
ACTIDI	atributos dependientes de idioma de los tipos de registros que incluiye una agenda de un microsite.



1.1.4. Banners

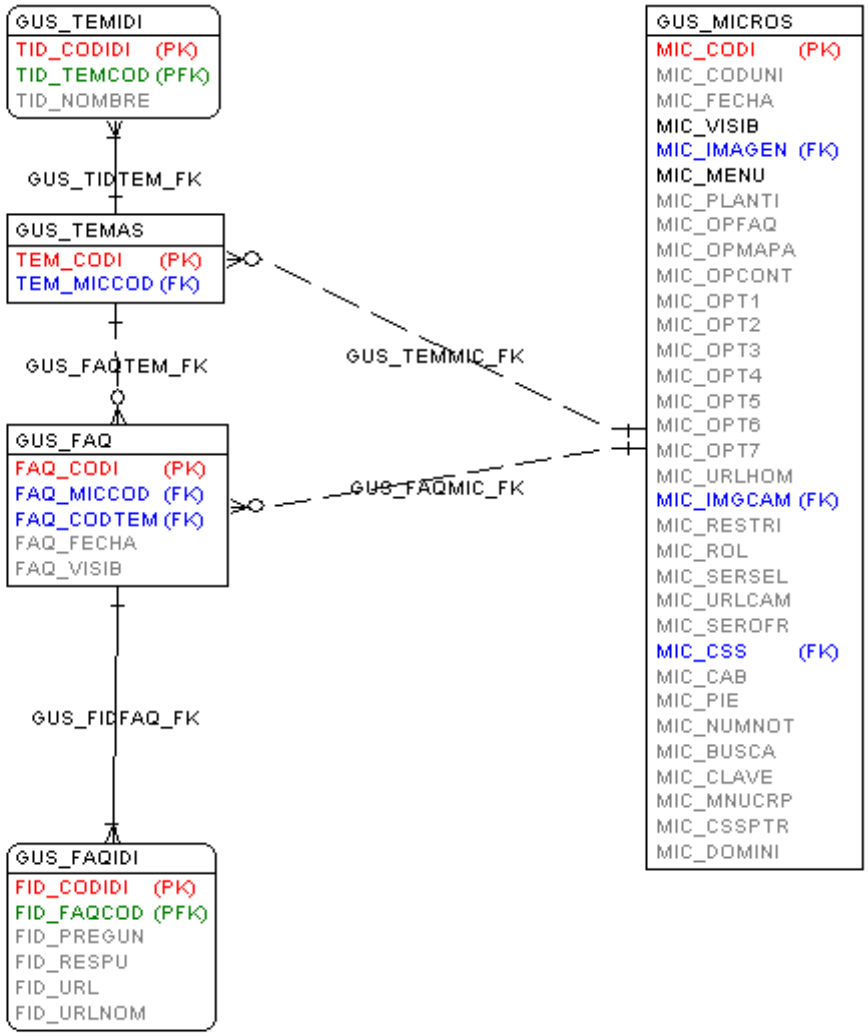
BANIDI	atributos dependientes de idioma de los elementos de tipo banner de un microsite.
BANNER	elementos de tipo banner definidos para un microsite.

[1,1]



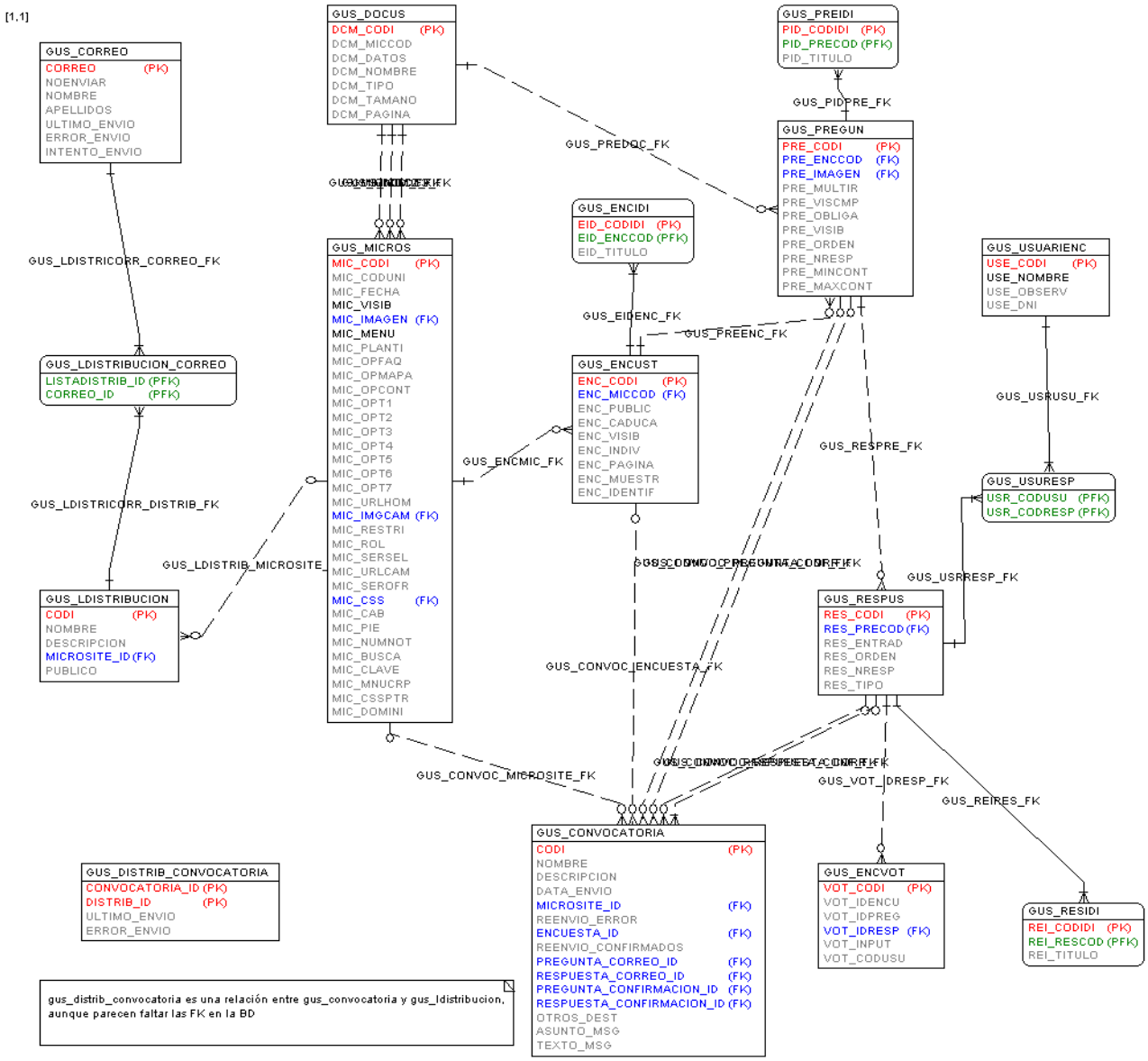
1.1.5. Faq

FAQ	registro de preguntas frecuentes creadas para un microsite.
FAQIDI	atributos dependientes de idioma para las preguntas frecuentes de un microsite.
TEMAS	registro de agrupadores definidos para las preguntas frecuentes que se creen para un microsite.
TEMIDI	atributos dependientes de idioma para la definición de los agrupadores de preguntas frecuentes de un microsite.



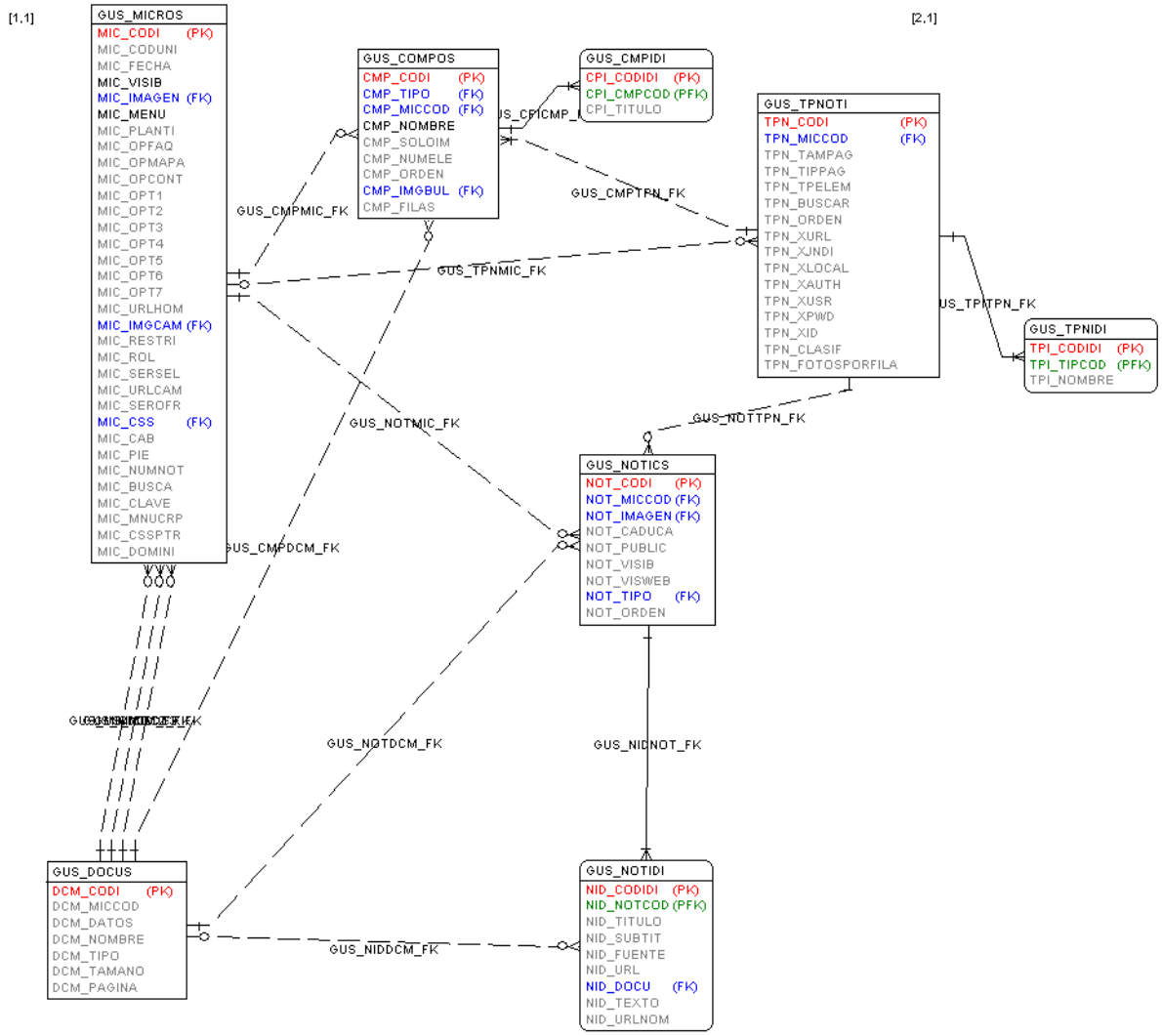
1.1.6. Encuestas

ENCUST	registro de encuestas definidas para un microsite. Se compone de una relación de preguntas y respuestas.
ENCIDI	atributos dependientes de idioma para las encuestas de un microsite.
PREGUN	relación de preguntas asociadas a una encuesta definidas para un microsite.
PREIDI	atributos dependientes de idioma para las preguntas asociadas a una encuesta de un microsite.
RESPUS	relación de respuestas tabuladas a una pregunta asociadas a una encuesta definidas para un microsite.
RESIDI	atributos dependientes de idioma para las respuestas tabuladas de las preguntas asociadas a una encuesta de un microsite.
ENCVOT	resultados de las contestaciones sobre las preguntas asociadas a una encuesta publicadas para un microsite.
USUARIENC	ficha de los usuarios de los usuarios que rellenan una encuesta.
USURESP	relaciona un usuario que rellena una encuesta con las respuestas.
CONVOCATORIA	Convocatoria para enviar por correo
CORREO	Correo electrónico de un usuario
LDISTRIBUCION	Lista de distribución
LDISTRIBUCION_CORREO	Asignación de un correo de usuario a una lista de distribución
DISTRIB_CONVOCATORIA	relaciona una lista de distribución con una convocatoria


[1,1]
GUS_CORREO
CORREO (PK)
NOENVIAR
NOMBRE
APELLIDOS
ULTIMO_ENVIO
ERROR_ENVIO
INTENTO_ENVIO
GUS_DOCUS
DCM_CODI (PK)
DCM_MICCOD
DCM_DATOS
DCM_NOMBRE
DCM_TIPO
DCM_TAMANO
DCM_PAGINA
GUS_PREIDI
PID_CODIDI (PK)
PID_PRECOD (PFK)
PID_TITULO
GUS_PIDPRE_FK
GUS_PREDOC_FK
GUS_PREGUN
PRE_CODI (PK)
PRE_ENCCOD (FK)
PRE_IMAGEN (FK)
PRE_MULTIR
PRE_VISCMP
PRE_OBLIGA
PRE_VISIB
PRE_ORDEN
PRE_NRESP
PRE_MINCONT
PRE_MAXCONT
GUS_ENCIDI
EID_CODIDI (PK)
EID_ENCCOD (PFK)
EID_TITULO
GUS_USUARIENC
USE_CODI (PK)
USE_NOMBRE
USE_OBSERV
USE_DNI
GUS_LDISTRICORR_CORREO_FK
GUS_MICROS
MIC_CODI (PK)
MIC_CODUNI
MIC_FECHA
MIC_VISIB
MIC_IMAGEN (FK)
MIC_MENU
MIC_PLANTI
MIC_OPFAQ
MIC_OPMAPA
MIC_OPCONT
MIC_OPT1
MIC_OPT2
MIC_OPT3
MIC_OPT4
MIC_OPT5
MIC_OPT6
MIC_OPT7
MIC_URLHOM
MIC_IMGCAM (FK)
MIC_RESTRI
MIC_ROL
MIC_SERSEL
MIC_URLCAM
MIC_SEROFR
MIC_CSS (FK)
MIC_CAB
MIC_PIE
MIC_NUMNOT
MIC_BUSCA
MIC_CLAVE
MIC_MNUCRP
MIC_CSSPTR
MIC_DOMINI
GUS_EIDENC_FK
GUS_PREENC_FK
GUS_ENCUST
ENC_CODI (PK)
ENC_MICCOD (FK)
ENC_PUBLIC
ENC_CADUCA
ENC_VISIB
ENC_INDIV
ENC_PAGINA
ENC_MUESTR
ENC_IDENTIF
GUS_USUSU_FK
GUS_USURESP
USR_CODUSU (PFK)
USR_CODRESP (PFK)
GUS_RESPRE_FK
GUS_LDISTRICORR_DISTRIB_FK
GUS_ENCMIC_FK
GUS_LDISTRIB_MICROSITE_FK
GUS_LDISTRIBUCION_CORREO
LISTADISTRIB_ID (PFK)
CORREO_ID (PFK)
GUS_LDISTRIBUCION
CODI (PK)
NOMBRE
DESCRIPCION
MICROSITE_ID (FK)
PUBLICO
GUS_USRRESP_FK
GUS_RESPUS
RES_CODI (PK)
RES_PRECOD (FK)
RES_ENTRAD
RES_ORDEN
RES_NRESP
RES_TIPO
GUS_CONVOC_ENCUESTA_FK
GUS_CONVOC_MICROSITE_FK
GUS_VOT_IDRESP_FK
GUS_REIRES_FK
GUS_CONVOCATORIA
CODI (PK)
NOMBRE
DESCRIPCION
DATA_ENVIO
MICROSITE_ID (FK)
REENVIO_ERROR
ENCUESTA_ID (FK)
REENVIO_CONFIRMADOS
PREGUNTA_CORREO_ID (FK)
RESPUESTA_CORREO_ID (FK)
PREGUNTA_CONFIRMACION_ID (FK)
RESPUESTA_CONFIRMACION_ID (FK)
OTROS_DEST
ASUNTO_MSG
TEXTO_MSG
GUS_ENCVOT
VOT_CODI (PK)
VOT_IDENCU
VOT_IDPREG
VOT_IDRESP (FK)
VOT_INPUT
VOT_CODUSU
GUS_RESIDI
REI_CODIDI (PK)
REI_RESCOD (PFK)
REI_TITULO
GUS_DISTRIB_CONVOCATORIA
CONVOCATORIA_ID (PK)
DISTRIB_ID (PK)
ULTIMO_ENVIO
ERROR_ENVIO
gus_distrib_convocatoria es una relación entre gus_convocatoria y gus_ldistribucion, aunque parecen faltar las FK en la BD

1.1.7. Noticias

NOTICS	registro de elemetos a publicar en la lista para un tipo de listado definidos en un microsite.
TIPSER	registro de los servicios de un microsite. La creación de un nuevo servicio no solo supone el registro de esta tabla.
TPNIDI	atributos dependientes de idioma para la definición de un tipo de listado de un microsite.
TPNOTI	registro de los distintos tipos de listados definidos para un microsite y que se utilizarán para la publicación de listas de elementos de los distintos tipo posible.
NOTIDI	atributos dependientes de idioma para una noticia



1.2. Módulos existentes

La aplicación se empaqueta en un archivo ear. El código se divide en diferentes módulos de aplicación web siguiendo el modelo de capas y aplicaciones web.

1.2.1. extractor (herramientas de texto)

Contiene los paquetes
org.ibit.rol.sac.extractor.tidy
org.ibit.rol.sac.extractor.taw
para realizar parseo y análisis de HTML: TawHtmlParser y TawResultBean
Se usa desde:
- org.ibit.rol.sac.microback.utils.w3c.Testeador

No confundir con el paquete org.ibit.rol.sac.extractor, importado desde ROLSAC con el jar extractor.jar y usado al parecer en los índices y/o búsqueda lucene.

1.2.2. micromodel (modelo de datos)

Contiene las clases del modelo de datos
Implementadas usando hibernate 2.0, mapeadas en archivos hbm.xml

1.2.3. micropersistence (capa de persistencia)

La capa de persistencia se forma con EJBs de sesión.
El EJB se implementa en micropersistence.ejb y el Delegate en micropersistence.delegate; el resto de
artefactos se generan con xdoclet.

1.2.4. microintegracion

Clases que implementan lo necesario para la integración con el traductor de rolsac.
Se usa también la librería importada de ROLSAC integracion.jar

1.2.5. microback (backoffice)

Administrador de microsites.
Basado en struts. Usa el modelo de Actions y ActionForms para realizar los mantenimientos.

1.2.6. microfront (frontend)

Frontal de microsites
Basado en struts.

El control de acceso se hace en UserRequestProcessor
(org.ibit.rol.sac.microfront.home.UserRequestProcessor)

Campos clave:

- Microsite.restringido (MIC_RESTRI). Indica el tipo de microsite

- 'S' -> site de intranet
- 'N' -> site público v1 (versió corporativa anterior)
- '2' -> site público v4 (versió corporativa nova)

- Microsite.rol (MIC_ROL). En caso de site de intranet, permite restringir a un rol

- Microsite.plantilla (MIC_PLANTI). Indica el tipo de home.
  - 1,2: portada de la plantilla (proporcionada por la aplicación)
  - 4 página específica del Microsite, indicada en Microsite.urlhome (MIC_URLHOM)
  - 3 y 5 no los he visto, pero la documentación inline en Bdhome.java indica:
    //1 y 2=plantillas, 3=imagen, 4=url, 5=despliege servicios, -1=error

Vistas JSP:

- Hay 2 conjuntos de jsps
  htdocs/v1: Maneja los sites de tipo intranet y público v1
  htdocs/v4: Maneja los sites de tipo público v4

Esquema de petición:

Todas las peticiones *.do la maneja struts
org.apache.struts.action.ActionServlet, configurado web.xml y definido en struts-config.xml

- El primer paso es UserRequestProcessor, que hace el control de acceso, esto es así porque UserRequestProcessor está configurado como processorClass en struts-config.xml. De modo que UserRequestProcessor.processPreprocess actúa como un filtro de todos los actions.
  - 1 Si el site es público no hace nada.
  - 2 Si el usuario no está logueado, redirige al login.
  - 3 Si está logueado y no tiene el rol correcto, indica el error.
- Struts encauza la petición al action correspondiente mapeado en struts-config.xml
  (La mayoría de actions siguen el siguiente esquema)

  org.ibit.rol.sac.%%nombre%%.actions.*: Actions struts, controlan la ejecución

  org.ibit.rol.sac.%%nombre%%.util.Bd*: Acceso a datos, preparan los datos a mostrar

  Así la ejecución de un Action típico sigue los pasos siguientes:
  - Obtiene los datos del BD correspondiente
  - Fija los atributos en el request (en ocasiones, ver apartado "problemas detectados" estos atributos son HTML generado)
  - Redirige a la plantilla correspondiente (forward v1 o v4) según el resultado de la obtención de datos y la versión del microsite.

1.3. Problemas detectados

1.3.1. sacmicrofront

- Se guardan datos en sesión que no deberían. Por ejemplo, el microsite actual.
  Esto hace que la aplicación no sea session-safe (hay errores cuando se navega en diferentes microsites en diferentes pestañas de navegador)
  Hay que evitarlo, si lo que se pretendía era optimizar rendimiento, es mejor usar técnicas de caching.

- La semántica de Microsite.restringido es confusa. Por un lado indica si un site es restringido o público, pero también la plantilla. Sería mejor tener 2 campos diferenciados, lo que permitiría indicar que un site basado en plantilla v4 sea restringido.
- En algún caso, como en Bdhome, éste genera directamente HTML, que se fija en el request para ser mostrado después. Para ello se usan clases en org.ibit.rol.sac.microfront.util.microtag. Estas clases incluso realizan el acceso a datos, por ejemplo en MparserAgenda.getHtmlAgendaListado.
- En el control de acceso que UserRequestProcessor, al menos en nuestra versión, parece que hay un error. Si entras sin tener el rol y luego haces reload al final te deja entrar.

1.3.2. sacmicroback

- No dispone de roles propios, sino que usa los de rolsac RSC_OPER, RSC_SUPER, RSC_ADMIN, RSC_SYSTEM. Estos roles están además "hard-coded" en muchos actions: String[] roles = new String[]{"sacsystem", "sacadmin", "sacoper", "sacsuper"};
- La lógica de los enlaces internos no está tipificada. Cuando en un elemento se incluye un enlace a un componente interno al microsite, se incluye un enlace completo que incluye información de detalle de la url destino del front. Esto hace poco flexible la estructura del frontal.
- El exportador de microsites exporta todo el contenido del microsite (incluidos archivos) a un archivo xml en memoria. Esto no es escalable y debe modificarse para solucionar uno de los Bugs detectados.

1.4. Dependencias con otros proyectos CAIB

La implementación actual de GUSITE incluye dependencias con otros proyectos CAIB. Dado que uno de los requerimientos del proyecto indica la necesidad de convertir GUSITE en un proyecto independiente, se listan aquí las librerías usadas y las acciones necesarias para eliminar su vinculación.

1.4.1. ROLSAC/extractor.jar

Contiene el paquete org.ibit.rol.sac.extractor* (excepto tidy y taw que son locales al proyecto)
USO: En el indexador (org.ibit.rol.sac.micromodel.IndexObject)
TAREAS:

- Incluir en el proyecto las clases necesarias como es.caib.gusite.extractor.*
- Eliminar el jar.

1.4.2. ROLSAC/model.jar

Clases del modelo de rolsac.

USO: Acceso a datos de rolsac que se usan en gusite:

- UnidadAdministrativa, TraduccionUnidadAdministrativa: Un microsite está relacionado con una UA
- Ficha, TraduccionFicha: En rolsac se "vinculan" fichas a microsites. Bueno, la vinculación se hace por url, de modo que la búsqueda de fichas relacionadas a un microsite es así:
  "where trad.url like '%sacmicrofront%' AND trad.url like '%idsite="+idsite+"%'");

Y no está muy claro dónde de gusite se usan las fichas, porque sólo lo hemos visto en la indexación de microsites.

- Materia, TraduccionMateria: No está muy claro dónde se usan, pero parece que se obtienen a partir de las fichas. ¿en la indexación?
- AgrupacionMateria: Se cargan las agrupaciones en MenuCabecera.java, pero parece que luego no se usan.
- Idioma: Parece más un error, creo que debería usarse el Idioma de gusite.
- ProcedimientoLocal, TraduccionProcedimientoLocal: Gusite permite crear una página con el listado de procedimientos de la UA del Microsite

TAREA: Sustituir por API v2 EJB de Rolsac. Eliminar los usos obsoletos (como parece ser AgrupacionMateria).

1.4.3. ROLSAC/sac-persistence.jar

Capa de persistencia de acceso al modelo de rolsac.

Se aplica lo dicho para ROLSAC/model.jar

1.4.4. ROLSAC/integracion.jar

Contiene los paquetes es.indra.rol.sac.integracion.*

USO: En microintegracion se usa la clase es.indra.rol.sac.integracion.traductor.Traductor para realizar traducciones.

TAREA: Incluir en el proyecto las clases necesarias como es.caib.gusite.microintegracion.*

1.4.5. WEBCAIB/boib.jar

Api de webcaib para acceso al antiguo BOIB (ya obsoleto)

USO: Se utilizaba en MenuCabecera.java para obtener la url del último boib. Ahora ya no se usa.

TAREA: Eliminarlo del proyecto.

1.4.6. WEBCAIB/link.jar

USO: Parece que no se utiliza.

TAREA: confirmarlo y eliminarlo del proyecto.

1.4.7. WEBCAIB/sac.jar

Esta es la API de webcaib para acceso a la BD de rolsac.

USO:

- en Bdbase.java para construir el pie de página (dirección de la uo, etc...)
- en MenuCabecera.java para contruïr el menú del sitio web

TAREA: sustituir por API v2 EJB de Rolsac

1.4.8. WEBCAIB/caib-utilities.jar (en el ear de librerías)

Utilidades webcaib, en webcaib-1.2 está el código

USO:

- En microback y microfront
  <listener-class>es.caib.utilities.sesion.NsesionesMicroback</listener-class>
  <listener-class>es.caib.utilities.sesion.NsesionesMicrofront</listener-class>
- En microback.action.GraficoAction, para generar las gráficas de estadísticas del microsite:
  import es.caib.utilities.chart.*;
- En ProcesoW3CTesteandoValorStatusBar, AjaxValorBarraEstadoAction, Proceso y ProcesoW3C. Para mostrar la barra de progreso del proceso de validación W3C
  import es.caib.utilities.statusbar.StatusBar
  import es.caib.utilities.statusbar.StatusBarFactory
- En BdEnvioencuesta.java y Fechas.java
  es.caib.utilities.date.Fechas

TAREA Incluir las clases necesarias nativamente en el proyecto

1.4.9. LIBRERIAS/lucene-caib.jar

Es la base de la implementación del indexador, **se desconoce la ubicación del código fuente original.** Puesto que el proyecto lucene-caib no se evoluciona, parece lógico incluir las clases necesarias nativamente en el proyecto, para poder evolucionarlas a voluntad, previa obtención del código fuente original.

TAREA: Incluir las clases necesarias nativamente en el proyecto

1.4.10. seyconsession-common.jar

Librería de login CAIB.

Se usa en Bdbase.getIdUser y Bdbase.getPrincipal, pero estos métodos parecen no usarse así que debería poder eliminarse esta dependencia del proyecto, ya que el control de acceso se configura en el container.

TAREA: Eliminar del proyecto si su uso ya está obsoleto.

2. Mejoras de arquitectura

2.1. Migración a JPA + hibernate 3x + annotations

Esta tarea se puede abordar de manera progresiva en las fases secuenciales siguientes.

2.1.1. Hibernate 3.x

No debería ser muy problemático, hay una pequeña guía aquí:

https://community.jboss.org/wiki/HibernateCoreMigrationGuide30

Proponemos migrar inicialmente a hibernate 3.6, ya que la migración de 3.0 a 3.6 debería ser "drop-in-replacement".

El mantener hibernate 2 limita bastante el desarrollo, puesto que la versión está muy limitada en funcionalidades y no está mantenida por hibernate.

La versión actual de hibernate es la 4.3.1.

Esta tarea afectará a los módulos micromodel y micropersistence donde todas las clases EJB extienden una base HibernateEJB y el diseño modular del proyecto debería facilitar la tarea.

2.1.2. Hibernate 3.x + anotaciones JPA

Se trata de ir migrando clase a clase los descriptores *.hbm.xml por anotaciones JPA.

http://docs.jboss.org/hibernate/annotations/3.5/reference/en/html_single/#setup-configuration

"Note: You can mix annotated persistent classes and classic hbm.cfg.xml declarations with the same SessionFactory. You can however not declare a class several times (whether annotated or through hbm.xml). You cannot mix configuration strategies (hbm vs annotations) in an entity hierarchy either."

2.1.3. JPA

Migrar el descriptor hibernate-cfg.xml de hibernate a descriptor JPA persistente.xml

http://docs.jboss.org/hibernate/orm/3.6/quickstart/en-US/html/hibernate-gsg-tutorial-jpa.html

2.2. Cambios en la nomenclatura de paquetes

Cambiar org.ibit.rol.sac.

micromodel
micropersistence
extractor.tidy
extractor.taw
microfront
microback
micropersistence

por es.caib.gusite.

micromodel
micropersistence
extractor.tidy
extractor.taw
microfront
microback
micropersistence

Cambiar es.indra.gusite.

microintegracion

por es.caib.gusite.

microintegracion

2.3. Migrar el proyecto a UTF-8.

Convertir todos los archivos del proyecto a UTF-8.

Incluir los filtros necesarios en las aplicaciones web.

2.4. Eliminar dependencias con otras aplicaciones CAIB y liberación de la aplicación.

Para conseguir esto es necesario:

- Tareas indicadas en el punto 1.4, así se conseguirá que la aplicación sea independiente del frontal webcaib.
- Incluir roles propios para la aplicación, dado que en la actualidad se están usando los de ROLSAC.
- Eliminar la dependencia con el ear de librerías, configurando un classloader propio e incluyendo las librerías necesarias en el ear propio.

Por otro lado podrá funcionar en un entorno con el único requerimiento de tener acceso a un API de ROLSAC, que ya es una aplicación liberada.

Eliminar el requerimiento de ROLSAC es también sencillo, pero recomendamos hacerlo en una fase posterior.

2.5. Migrar el proyecto a JBOSS 5.

Realizar las adaptaciones necesarias para que el proyecto funcione bajo el entorno jboss5.

Revisar las librerías incluidas y eliminar del proyecto las que ya aporte de base jboss5.

2.6. Sistema de plantillas

GUSITE incorporará un sistema de plantillas que permitirá crear (y subir al sistema) nuevas plantillas de sitio web, así como editar y modificar las plantillas individuales tanto de páginas como de módulos.

Así, se creará un frontal nuevo basado en este sistema de plantillas, en lugar de jsp-jstl como lo está el frontal actual.

2.6.1. Elección del motor de plantillas

De entre la multitud de sistemas de plantillas existentes, hemos pre-seleccionado 3 (thymeleaf, freemarker y velocity) y en principio a falta validarlo con un juego de pruebas de capacidades, nos decantamos por thymeleaf.

2.6.1.1. thymeleaf

Es moderno y está optimizado para generar XHTML, permitiendo asegurar que se genera XHTML válido. Su sintaxis basada en tags, permite escribir plantillas de un modo bastante "natural".

De base no incluye la capacidad de leer plantillas de base de datos y de procesar cadenas con fragmentos de plantilla XHTML; aunque es fácilmente extensible para que así sea.

intro: http://blog.zenika.com/index.php?post/2013/01/18/introducing-the-thymeleaf-template-engine

ejemplo: http://www.thymeleaf.org/petclinic.html

2.6.1.2. freemarker

Es un sistema de plantillas robusto y usado ampliamente, que puede usarse para generar cualquier tipo de archivo de texto. Está basado en marcas de sustitución ({$})

De base no incluye la capacidad de leer plantillas de base de datos aunque es fácilmente extensible para que así sea.

Tutorial: http://www.vogella.com/tutorials/FreeMarker/article.html

Página oficial: http://freemarker.org/

2.6.1.3. velocity

Tal vez el template system más extendido en java, lo usa liferay entre otros.

Descartado por varios motivos: es bastante pesado, su última actualización fue en 2010.

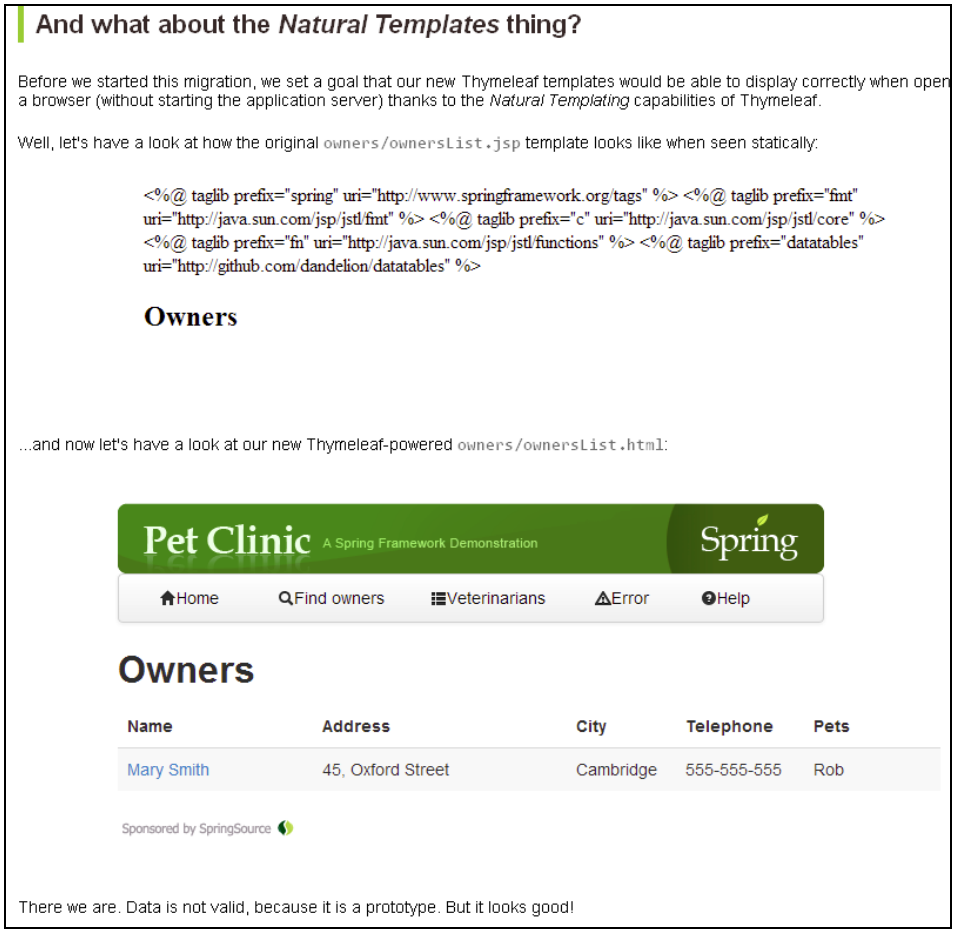
2.6.1.4. Elección final: thymeleaf

Se usara finalmente thymeleaf para el sistema de plantillas. Antes de continuar, algunas definiciones:

- **fragmento**: Llamaremos fragmento, a una porción de código que visualiza un apartado determinado de una página, pero a menudo reutilizado en múltiples páginas. Por ejemplo: cabecera y pie.
- **plantilla**: La plantilla es el archivo que contiene la vista de una página completa. Por ejemplo: la portada. Una plantilla puede a su vez incluir diversos fragmentos.
- **tema**: Llamaremos tema al conjunto completo de plantillas y fragmentos que implementa la vista completa del sitio web.

Entre muchas otras funcionalidades interesantes, he aquí algunas:

- Natural templates: las plantillas thymeleaf, se visualizan como prototipos cuando no se ejecutan en entorno de servidor, es decir, en modo estático (ver captura de ejemplo más abajo). Esto simplifica y agiliza la maquetación. Usaremos thymol (http://sourceforge.net/u/jjbenson/wiki/thymol/) para la ejecución estática de fragmentos, etc. en tiempo de diseño y maquetación.
- Integración con spring: thymeleaf se integra fuertemente con spring, que es la tecnología que usaremos en el nuevo front de GUSITE. http://doanduyhai.wordpress.com/2012/04/14/spring-mvc-part-iii-thymeleaf-integration/
- Extremadamente extensible: puede ser utilizado como un *framework* de motor de plantillas si es necesario.
- Traducción automática DOCTYPE —de plantilla DTD a resultado DTD— para validaciones (opcional) de plantillas y código resultante.
- Muchas plantillas listas para ser utilizadas (extensibles): XML, XHTML, HTML5.
- Soporte completo (y extensible) a la internacionalización.



And what about the *Natural Templates* thing?

Before we started this migration, we set a goal that our new Thymeleaf templates would be able to display correctly when open a browser (without starting the application server) thanks to the *Natural Templating* capabilities of Thymeleaf.

Well, let's have a look at how the original `owners/ownersList.jsp` template looks like when seen statically:

```
<%@ taglib prefix="spring" uri="http://www.springframework.org/tags" %> <%@ taglib prefix="fmt"
uri="http://java.sun.com/jsp/jstl/fmt" %> <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ taglib prefix="fn" uri="http://java.sun.com/jsp/jstl/functions" %> <%@ taglib prefix="datatables"
uri="http://github.com/dandelion/datatables" %>
```

Owners

...and now let's have a look at our new Thymeleaf-powered `owners/ownersList.html`:

Pet Clinic A Spring Framework Demonstration Spring

Home Find owners Veterinarians Error Help

Owners

Name	Address	City	Telephone	Pets
Mary Smith	45, Oxford Street	Cambridge	555-555-555	Rob

Sponsored by SpringSource

There we are. Data is not valid, because it is a prototype. But it looks good!

2.6.2. Configurar el motor de plantillas en el nuevo front

Configurar de modo estándar TemplateResolver, ViewResolver y TemplateEngine y testear el funcionamiento.

2.6.3. Reescribir la plantilla jsp-jstl v4 usando el nuevo motor de plantillas

Thymeleaf sustituye completamente la capa jsp. El componente sacmicrofront consta actualmente de ~40 archivos jsp. Esta tarea se realizará en paralelo con la tarea 2.7.4.

2.6.4. Programar y configurar las extensiones necesarias

Extender thymeleaf creando un dialecto propio para las funcionalidades GUSITE. Además debemos incluir sólo los tags de thymeleaf que queremos permitir usar a los gestores de sitios web.

http://www.thymeleaf.org/doc/html/Extending-Thymeleaf.html#scenario-3-creating-your-own-template-system

Crear una implementación de iResourceResolver y de iTemplateResolver. De este modo, según la configuración del sitio web, será posible ejecutar una determinada plantilla base o una plantilla adaptada para el sitio web. No hablamos sólo de modificar todo un tema, sino también de extender un tema modificando sólo ciertos fragmentos o páginas.

http://forum.thymeleaf.org/Loading-thymeleaf-template-from-Database-td4025164.html
http://stackoverflow.com/questions/13543877/modular-template-resolver-in-thymeleaf

2.7. Nuevo módulo de front

Requisitos iniciales:

- Url-friendly para posicionamiento SEO.
- Flexible y configurable para evolución futura.
- Inicialmente, se implementará la plantilla v4 de sacmicrofront usando el sistema de plantillas elegido. De este modo, la migración de los sites antiguos que usen esta plantilla será lo más directa posible.

Implementaremos el nuevo módulo de front usando spring MVC, tecnología ya usada con éxito en el proyecto de SEUCAIB. De entre las características de spring, usaremos:

- @RequestMapping parametrizado para implementar urls amigables. http://www.codingpedia.org/ama/seo-friendly-url-construction-with-spring-mvc/
- Configuración usando anotaciones, "autowiring" y "component scan".
- Integración con thymeleaf para las vistas.
- Conversión de los actions de struts en controllers de spring MVC.

2.7.1. Crear el nuevo módulo

Crear un nuevo componente en la aplicación GUSITE, consistente en un módulo web. Configurar la compilación y la integración en el ear actual del componente, además de incluirlo en el proceso de construcción "ant".

2.7.2. Configurar spring e integrar las librerías necesarias

Crear la aplicación base spring, y la infraestructura de base necesaria para el módulo web.

2.7.3. Inventario de actions y uris existentes

Realizar el inventario de actions y uris existentes en el actual front. Derivado de struts-config y web.xml.

2.7.4. Programar los nuevos controladores spring

La funcionalidad a implementar se extraerá del código actual struts, para las acciones que deben conservarse en la nueva versión. Esta tarea se realizará en paralelo con la tarea 2.6.3

2.7.5. Filtro en sacmicrofront.

Añadir un filtro al front antiguo para que redireccione al nuevo front los microsites configurados para ejecutarse en la nueva versión. NOTA: en principio, serán todos los configurados con plantilla v4.

2.7.6. Migración SEO.

Añadir un filtro al nuevo front para que realice la conversión de urls antiguas a urls modernas, con el objetivo de aprovechar el posicionamiento en buscadores de las páginas y secciones de los antiguos sitios web, y evitar problemas de errores 404 con enlaces antiguos.

2.8. Mover los archivos de blobs de base de datos a filesystem.

Permitir configurar como propiedad en la aplicación si se desea almacenar los archivos en filesystem o en base de datos.

Realizar el proceso de volcado de los archivos actuales.

Modificar las aplicaciones para que usen la propiedad de la aplicación para recuperar los archivos de la fuente correcta.

3. Resolución de problemas existentes

3.1. BUG: Exportació de microsites

"L'exportació de microsites es basa en la generació d'un fitxer XML on s'inclouen els binaris del documents adjunts al microsite. Si el microsite té molts de documents adjunts, o bé no s'arriba a generar el fitxer XML o bé es genera un fitxer de centenars de MB que no s'arriba a importar."

Solución:

Crear el xml sin los archivos, exportar los archivos a filesystem y componer un zip con xml y archivos. Pista: para excluir los datos de los archivos del xml añadir a ModelSuppressionStrategy:

addSupressionList(Archivo.class, new String[] {"datos"});

Modificar el importador para que soporte los nuevos tipos de archivo de importación (además de los antiguos).

Nota:

Uno de los cambios arquitectónicos requeridos es sustituir el almacenamiento de blobs en base de datos por filesystem, lo cual afecta directamente a esta funcionalidad. Así, sería preferible solucionar este problema conjuntamente con el cambio de blobs en bd a filesystem.

3.2. BUG: L'arbre de menú no funciona bé amb IE7

"A vegades no es graven els canvis sobre el menú principal. Pareix que només passa amb IE7."

Solución:

Revisar el javascript y los estilos el control.

3.3. Eliminar la funcionalidad de procedimientos

"No es necesaria esta funcionalidad tan concreta, de hecho no se emplea y nunca ha funcionado al 100%."

Solución:

Eliminar las clases y archivos vinculados.

Eliminar el tipo de servicio de la base de datos.

4. Nuevas funcionalidades

4.1. MILLORA: Posicionamiento SEO y estadísticas

"Definir un conjunt de camps obligatoris que facilitin el posicionament dels microsites en els cercadors (títol, metadescripcions, metadades)"

Permitir vincular una cuenta genérica de Google Analytics y una cuenta específica por microsite.

Solución:

Añadir los campos (multi-idoma) al modelo, backoffice y frontales. Habría que repensar los requisitos de obligatoriedad, toda vez que estos campos tampoco son muy determinantes para el SEO.

4.2. MILLORA: Microsites semipúblics

"Permetre que part d'un microsite tingui accés restringit. Actualment només tenim microsites públics o interns"

Solución:

Crear el concepto de "grupo de usuarios" del microsite. Cada grupo de usuarios tendrá asociado uno o varios roles y se permitirá restringir el acceso al microsite o a una rama del árbol de menús por grupo de usuarios.

Esta tarea se realizará sólo en el nuevo frontal, por tanto es dependiente de las tareas de creación del nuevo frontal.

4.3. MILLORA: Cercador de microsites al backoffice

"Facilitar el procés d'assignació de microsites a usuaris. Actualment es presenten tots el microsites (200 aprox.) en un desplegable"

Solución:

Sustituir el desplegable por un campo search-as-you-type.

4.4. MILLORA: Auditories per microsites

"Control de canvis per usuari i per microsite"

Solución:

Interceptar los cambios en la capa de datos o persistencia para guardar histórico de cambios en la base de datos.

Crear las pantallas en el backoffice para gestionar las auditorías.

4.5. Editor markdown

GUSITE2014 permitirá editar contenidos usando markdown (http://daringfireball.net/projects/markdown/) de forma que el usuario podrá elegir entre usar el editor HTML integrado o un editor markdown (o elegir de entre más de uno).

Existen varias implementaciones de markdown en java. En tiempo de desarrollo probaremos:

- pegdown: https://github.com/sirthias/pegdown/
- markdownj: http://code.google.com/p/markdownj/

Así, se debe incluir un nuevo atributo a los contenidos editables que indique su formato y la capacidad a los frontales de que muestren el contenido editado en el formato correcto.

Se permitirá pasar la edición de un contenido de markdown a HTML, pero no es posible al contrario.

4.6. Diseño y maquetación web. Funcionalidades BASE.

4.6.1. Facilitar que el diseño resultante sea multidispositivo. Responsive Design.

- Actualizar las plantillas GUSITE a html5 + css3
- Aplicar técnicas de diseño responsive para que el código de base de GUSITE se adapte a diversos tamaños de pantalla (multi-dispositivo), tal y como se ha realizado con éxito en la aplicación SEUCAIB.

4.6.2. Aplicación y uso del concepto de plantillas

GUSITE incorporará desde la implementación del punto 2.6.1.4 el concepto de temas, plantillas y fragmentos. En este punto, se pretende extenderlo de modo que los usuarios puedan cargar temas nuevos y modificar las plantillas existentes.

- Los temas soportarán herencia, de modo que se pueda extender un tema existente modificando sólo ciertos archivos o plantillas.
- Se permitirá la edición de la plantilla en línea.
- El usuario podrá crear sus propios temas y hacerlos públicos para otros usuarios.
- Se diseñarán dos variaciones del tema base para dejarlas disponibles para los usuarios.

4.7. Eliminar el componente de banners

Este componente es muy poco usado en la actualidad. Por otro lado el nuevo sistema de plantillas hace que puedan conseguirse sus objetivos por otros medios así que se eliminará este componente.

4.8. Sistema de plugins (GUSITE como frontal de multiples backoffices)

Es necesaria una arquitectura extensible que permita que otros desarrolladores puedan incluir de forma ordenada y sencilla nuevos módulos.

El caso de uso principal es la integración de datos de otras aplicaciones, de modo que no sea necesario el desarrollo de un frontal completo para una nueva aplicación departamental, sino utilizar GUSITE como frontal, desarrollando únicamente un módulo de integración.

Por otro lado, esta funcionalidad debe dar soporte también al desarrollo de los nuevos componentes que se desean en la siguiente fase:

- Componente de suscripción RSS a los microsites.
- Componente de redes sociales.
- Componente de contacto con el gestor para usuarios autenticados / no autenticados.

A la hora de escoger la implementación del sistema de plugins, hay que tener en cuenta los requisitos del sistema:

- Simplicidad: el problema inicial a resolver está relativamente acotado y su alcance no es la funcionalidad núcleo del proyecto.
- Integración con el entorno de ejecución de la DGIDT.
- Facilidad de mantenimiento y extensibilidad futura.

Existen diversas alternativas en cuanto a la implementación:

1. Implementar OSGi.

   OSGi es un Framework de componentes para java en el cual se pueden instalar unidades de recursos llamados paquetes (bundles). Estos paquetes pueden exportar servicios o ejecutar procesos, y tienen sus dependencias gestionadas por el contenedor. Cada paquete también puede tener su propia ruta de clase interna, de modo que pueda servir como una unidad independiente. Todo esto se encuentra estandarizado en OSGi, de modo que un paquete OSGi válido, sería en teoría instalable en cualquier contenedor OSGi.

   Este enfoque requiere de un contenedor OSGi y encontramos que está sobredimensionado para el problema inicial a resolver.

2. Integrar un Framework ligero existente.

   a. *Java Simple Plugin Framework (https://code.google.com/p/jspf/)*

   b. jin-plugin (https://code.google.com/p/jin-plugin/)

   c. Java Plug-in Framework (JPF) Project (http://jpf.sourceforge.net/)

   d. Impala (https://code.google.com/p/impala/)

   e. Springlime (https://code.google.com/p/springlime/)

   Una vez estudiados éstos, no hemos localizado ninguno que se ajuste exactamente a las necesidades del proyecto. Así que, aunque se trate de la fuente de inspiración principal, utilizarlos pondría en riesgo el requisito de facilidad de mantenimiento y extensibilidad futura.

3. Crear un framework ligero. En este caso, la arquitectura de SPRING y sus capacidades de auto-configuración y auto-descubrimiento de componentes, forman un entorno idóneo para implementar esta funcionalidad. A modo de ejemplo, se pueden consultar las siguientes referencias:

   - http://www.devx.com/Java/Article/31835

- http://dinukaroshan.blogspot.com.es/2010/12/plugin-based-architecture-with-spring.html
- http://classpathhelper.sourceforge.net/lightweightplugins/
- https://github.com/spring-projects/spring-plugin

Versió

0.8

Observacions

Control de canvis

Versió	Data	Autor	Modificació
0.5	28/01/2014	agarcia@at4	Creació del document amb la compilació de les tasques d'anàlisi i disseny.
0.55	26/02/2014	agarcia@at4	Ampliació de la tasca 2.1 per incloure adaptació a JPA
0.6	14/03/2014		Revisió de totes les tasques d'arquitectura segons l'última reunió tècnica.