



**Promoción del desarrollo de SW libre en un entorno de
calidad y confianza adaptando las metodologías, procesos,
modelos de negocio y últimas tecnologías**

FIT-350503-2007-7

D4.1.1
Arquitectura de referencia de Vulcano/EzForge

Editor: TID
Revisor: URJC

Fecha límite del entregable: 29/02/2008

Fecha de entrega: 29/02/2008

Este trabajo se licencia bajo Creative Commons Attribution-Share Alike 3.0.

Para ver una copia de esta licencia, visite <http://creativecommons.org/licenses/by-sa/3.0/> o envíe una carta a Creative Commons, 171 Second Street, Suite 300, San Francisco, California, 94105, USA.

Este trabajo está parcialmente financiado por el Ministerio de Industria, Turismo y Comercio español.

Historial de Cambios

Versión	Fecha	Estado	Autor (Partner)	Descripción
0.1	10/12/2007	Borrador	Luis de la Fuente (TID)	Introducción, Modelos de Integración, Arquitectura de EzForge
0.2	20/12/2007	Borrador	Irenka Redondo (TID)	Funcionalidades del núcleo EzForge, Integración de nuevas herramientas
1.0	28/02/2008	Final	Irenka Redondo (TID) y Daniel González (URJC)	Revisión del documento

RESUMEN EJECUTIVO

Este documento pretende dar una visión general de la arquitectura diseñada para la forja de nueva generación que se va a desarrollar en el proyecto Vulcano. Como documentación complementaria, el resto de entregables generados durante este año ahondan más profundamente en los distintos niveles de esta arquitectura.

Información del Documento

Proyecto FIT Número	FIT-350503-2007-7	Acrónimo	Vulcano
Título completo	Promoción del desarrollo de SW libre en un entorno de calidad y confianza adaptando las metodologías, procesos, modelos de negocio y últimas tecnologías		
URL	http://www.ines.org.es/vulcano		
URL del documento			

Entregable	Número	D4.1.1	Título	Arquitectura de referencia de Vulcano/EzForge
Paquete de Trabajo	Número	PT4	Título	Arquitectura E2.0 para la forja Vulcano
Tarea	Número	T4.1	Título	Arquitectura de referencia

Fecha de Entrega	Contractual	/02/2008	Entregado	/02/2008
Estado	Versión X, fecha dd/mm/yyyy		final	<input checked="" type="checkbox"/>
Tipo	Informe <input checked="" type="checkbox"/> Demo <input type="checkbox"/> Otro <input type="checkbox"/>			
Nivel de Diseminación	Público <input checked="" type="checkbox"/> Consorcio <input type="checkbox"/>			
Resumen (para diseminación)	Este documento pretende dar una visión general de la arquitectura diseñada para la forja de nueva generación que se va a desarrollar en el proyecto Vulcano.			
Palabras Clave	Arquitectura, REST, forja de nueva generación			

Autores (Partner)	Luis de la Fuente (TID), Irenka Redondo (TID)		
Responsable de Autoría	Irenka Redondo	Email	iredondo@tid.es
	Partner	TID	Tfno

TABLA DE CONTENIDOS

RESUMEN EJECUTIVO.....	3
TABLA DE CONTENIDOS.....	5
1 INTRODUCCIÓN.....	6
2 MODELOS DE INTEGRACIÓN.....	7
3 ARQUITECTURA DE EzFORGE.....	10
3.1 Lado Cliente: Plataforma de Mashup.....	12
3.2 Lado Servidor.....	14
3.2.1 Núcleo de EzForge	14
3.2.2 Adaptadores de los sistemas de backend	19
4 FUNCIONALIDADES DEL NÚCLEO EzFORGE	21
4.1 Funcionalidades básicas de una forja de desarrollo	21
4.2 Funcionalidades habituales en una forja de desarrollo.....	21
5 INTEGRACIÓN DE NUEVAS HERRAMIENTAS	23
5.1 Integración de herramientas cuya funcionalidad ya está incorporada	23
5.1.1 Desarrollo de un adaptador REST para EzForge	23
5.2 Integración de herramientas que cubren nuevas funcionalidades	23
6 CONCLUSIONES Y LÍNEAS FUTURAS.....	25
REFERENCIAS	26

1 INTRODUCCIÓN

El aumento en la producción de software de código abierto y la participación de las empresas como miembros activos de las comunidades ha puesto de manifiesto la necesidad de herramientas de desarrollo colaborativo que integren recursos alojados en fuentes diversas y nuevas aplicaciones que den respuesta al tipo de gestión que las empresas demandan.

El proyecto EzForge (PT4 de Vulcano) aborda la definición, desarrollo e implantación de una **arquitectura de integración de recursos** basada en tecnologías emergentes **Web 2.0**, que soluciona los problemas de las forjas actuales. La modelización de los servicios de una forja mediante recursos REST permitirá una integración más sencilla y flexible de herramientas heterogéneas. Por otra parte, el empleo de clientes ligeros y mash-up ofrecerá a los usuarios una interfaz fácilmente configurable de acuerdo a sus necesidades, permitiendo acceder desde la misma a distintas forjas y herramientas, proporcionando así una gestión conjunta en una misma interfaz de todos los proyectos en los que esté involucrado el usuario, independientemente de que se encuentren en diferentes repositorios.

Hemos denominado EzForge al núcleo de forja que contendrá las funcionalidades básicas y habituales de una forja de desarrollo, y a partir del cual se podrán integrar nuevas herramientas que permitan aumentar la calidad de los proyectos desarrollados y formando todo en conjunto la forja Vulcano. Por esta razón, a lo largo del documento se puede hablar indistintamente de EzForge, Vulcano o núcleo de la forja.

2 MODELOS DE INTEGRACIÓN

Al integrar la información procedente de varios sistemas heterogéneos en un único sistema con el fin de ofrecer una vista homogénea de la misma, existen dos modelos de integración dependiendo del lugar en el que se realice la tarea de integración:

- **Integración en el lado del cliente:** Este tipo de integración permite obtener y transformar la información directamente en el cliente mediante el agente que proporcione el acceso al sistema. Típicamente se tratará de un navegador web u otro agente como un plugin del entorno de desarrollo Eclipse. Esta técnica empleará métodos de acceso a la información a través de la web y sus propios sistemas de conversión de datos. Su ventaja fundamental es la de proporcionar al usuario los mecanismos necesarios para crear sus propios integradores de información, de tal manera que sea posible integrar en su espacio servicios que no han sido contemplados por los desarrolladores del sistema EzForge o por los administradores de la forja. Un ejemplo de integración en cliente es el desarrollo de un gadget (elemento básico de un mashup) cuya lógica de negocio permita obtener información de un servicio Bugzilla y relacionarla con un repositorio de código Subversion, todo ello empleando componentes JavaScript desde el propio cliente.

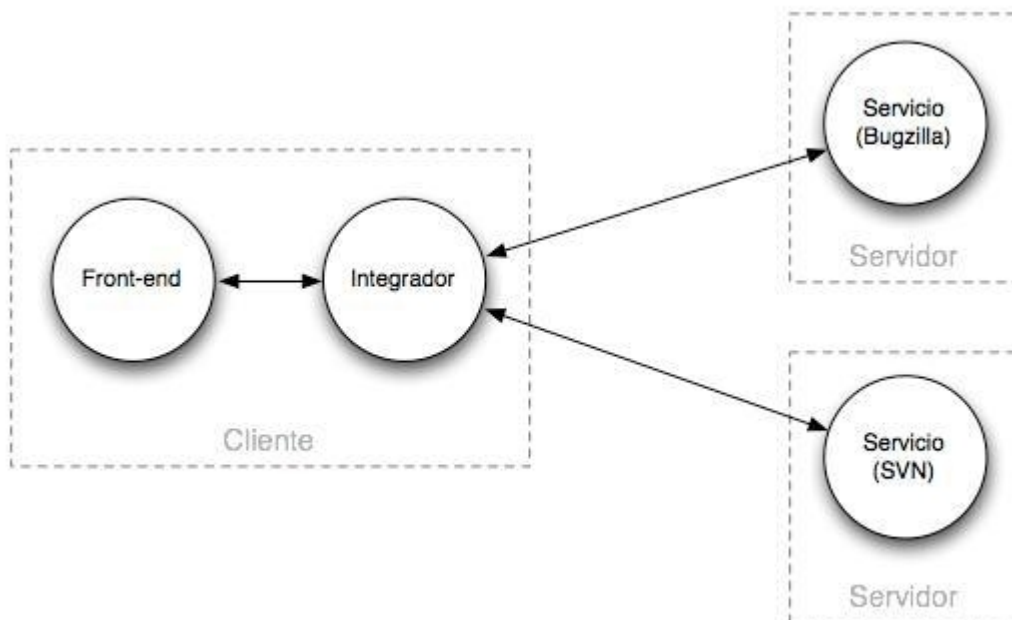


Figura 1: Integración en el cliente

- **Integración en el lado del servidor:** Este tipo de integración se caracteriza por situar la lógica de integración en el lado del servidor, de tal modo que al cliente ya le llega la información homogeneizada, de esta manera se garantiza una total transparencia en el cliente respecto a este proceso.

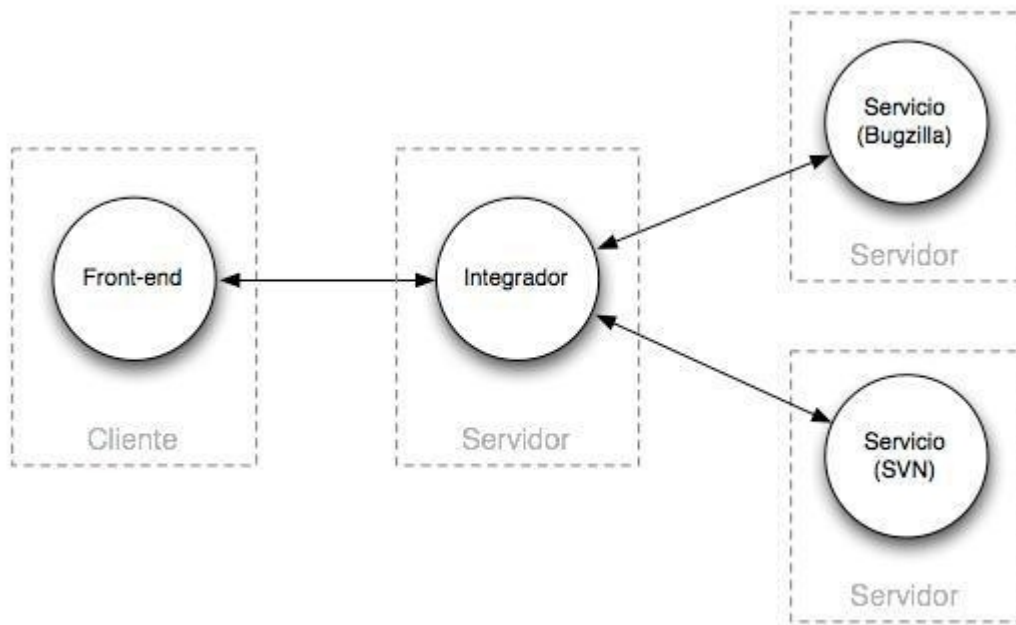


Figura 2: Integración en el lado del servidor

El modelo de integración en el lado del servidor ha sido el elegido para la definición de la arquitectura de integración de EzForge, si bien esta decisión no cierra la posibilidad de emplear una integración en el lado del cliente si el usuario así lo desea.

3 ARQUITECTURA DE EzFORGE

Los CDEs desarrollados hasta la fecha siguen un modelo de integración basado en componentes integrados de manera estática. Esto impide a los usuarios de la forja escoger los servicios a los que desean acceder, añadir nuevos componentes o desechar aquellos que no les sean necesarios. Este problema puede enfocarse desde dos puntos de vista:

- La naturaleza monolítica de los CDEs conocidos dificulta el desarrollo de nuevos servicios para ser incorporados al mismo. La ausencia de una arquitectura que permita añadir componentes (a modo de plugins) supone un grave obstáculo a la adición de nueva funcionalidad, ésta dependerá del equipo que desarrolló el CDE, el cual extenderá las funcionalidades de su software en función de sus propios objetivos e intereses, los cuales pueden no coincidir con los de sus usuarios (o al menos, no con todos ellos). Incluso si ese CDE se distribuye con una licencia open source, la extensión del mismo por parte del cliente puede conllevar unos costes de desarrollo demasiado elevados. En ambos casos, añadir servicios a un CDE monolítico puede resultar a menudo inviable.
- La naturaleza estática de los CDEs conocidos dificulta la elección de los servicios a emplear por parte de los usuarios. Incluso en el caso de un CDE no monolítico que permita la adición de nuevos servicios de manera sencilla, si estos son integrados de manera estática, los desarrolladores dependen del proveedor del CDE para añadir o eliminar funcionalidad. Esto supone que la autoridad para integrar los componentes depende de los administradores del sistema en el cual el CDE es encuentra alojado. Dentro de una misma organización, la problemática se reduce a contactar con los responsables encargados del sistema dentro de la misma. Sin embargo, en escenarios en los que intervengan diversas organizaciones o en el caso de que el servicio CDE sea provisto por terceras partes, las barreras entre organizaciones pueden llegar a suponer verdaderos escollos a la hora de escoger qué servicios serán integrados.

Estos dos problemas ilustran la necesidad de optar por un modelo arquitectónico más flexible, que garantice a los usuarios la integración de servicios en función de sus necesidades. Una forma de lograrlo es emplear arquitecturas orientadas a servicios que nos permitan evitar modelos de integración estáticos y monolíticos. Estas arquitecturas nos permiten que los componentes del sistema se integren unos con otros a través de APIs soportadas por protocolos independientes del lenguaje y la plataforma. Para el diseño de Ezforge se ha optado por seguir una arquitectura REST, *Representational State Transfer*, utilizando como infraestructura los protocolos HTTP y XML. El protocolo HTTP, *Hyper Text Transfer Protocol*, con sus cuatro operaciones básicas (*GET*, *POST*, *PUT*, *DELETE*), es el utilizado para el transporte, mientras que XML (*eXtensible Markup Language*) es utilizado para la representación de la información.

EzForge es una aplicación web 2.0 que presenta una arquitectura multinivel en la cual se han diseñado las siguientes capas:

- **Presentación:** capa a nivel de cliente, para esta se ha decidido utilizar una plataforma de mashups desarrollada dentro del proyecto Morfeo, la plataforma *EzWeb*. Puesto que la plataforma *EzWeb* se estaba desarrollando paralelamente al proyecto Vulcano, para la primera versión de EzForge se ha optado por la utilización de un framework (plataforma *EXT*) que si bien no es directamente la plataforma de integración en cliente definitiva que se utilizará en EzForge, permite ir diseñando la interfaz que va a proporcionar la forja. En diciembre se puso a disposición la primera versión de la plataforma *EzWeb*, por lo que comenzamos la migración de la interfaz a esta plataforma.
- **Lógica de Negocio:** lógica del funcionamiento de la forja, incluye tanto la lógica necesaria para el acceso a los sistemas integrados con la forja como a los mecanismos de validación y autenticación necesarios. Esta capa está basada en una serie de recursos y operadores que han sido implementados utilizando como lenguaje de programación Python, apoyándose también en otros estándares.
- **Datos:** información gestionada por la aplicación, se utiliza como soporte una base de datos relacional. Inicialmente se está utilizando MySQL por su naturaleza opensource (licencia GPL), si bien podría utilizarse cualquier otro gestor de datos relacional.
- **Adaptadores:** esta capa es la que permite la integración de herramientas con Ezforge. Al tratarse de una arquitectura REST, estos adaptadores tendrán como requisitos devolver la información (XML) siguiendo un determinado esquema y reponder por unas determinadas URIs (HTTP).

De la anterior división en capas, a la a la agrupación de las capas de lógica de negocio y datos es lo que se denomina núcleo (Core) de EzForge.

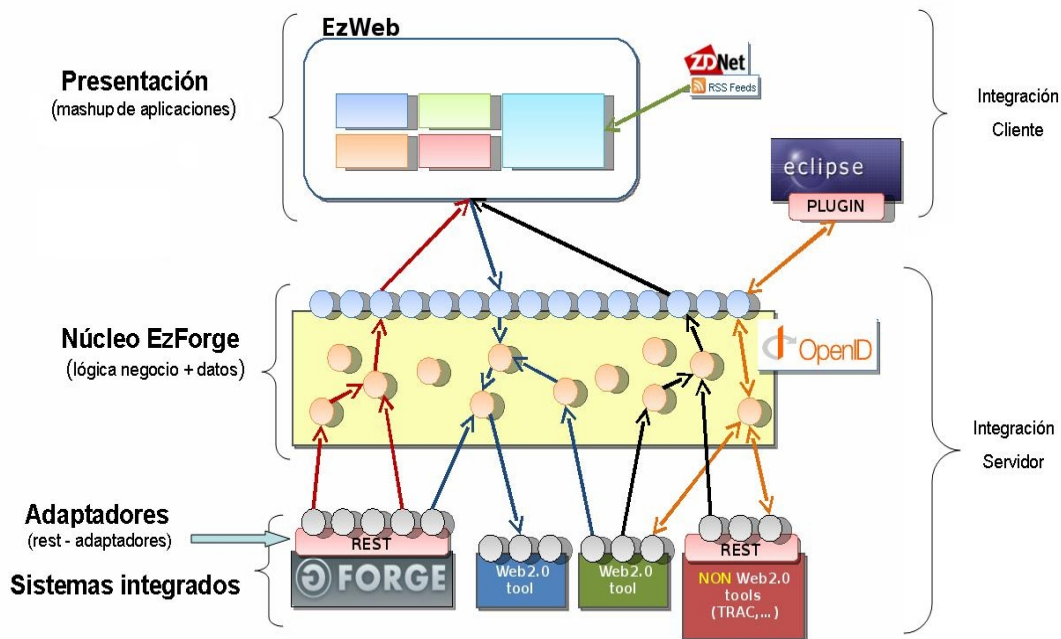


Figura 3: Arquitectura de EzForge

A continuación se detallan las capas de la arquitectura Ezforge.

3.1 Lado Cliente: Plataforma de Mashup

EzForge es una aplicación que se encuadra dentro de las arquitecturas tipo *Cliente Ligero*, es decir, la mayor carga de proceso de la aplicación se realiza en el servidor, dejando únicamente para la capa cliente la tarea de presentación de los datos. En esta capa cliente también es posible disponer de lógica muy sencilla realizada con javascript, y relacionada fundamentalmente con la presentación de los datos, por ejemplo la actualización de determinados elementos de la presentación en base a eventos (borrar, añadir, seleccionar, etc.) producidos en otros elementos de la misma.

En EzForge en lugar de optarse por utilizar la típica solución en arquitecturas de este tipo para la capa cliente, que consiste en utilizar un navegador que nos muestra una aplicación hecha a medida y genérica para todos los usuarios (Web 1.0), lo cual implica una mínima capacidad de personalización, se ha optado por una solución mucho más flexible e innovadora. Se ha diseñado un repositorio constituido por pequeñas unidades gráficas (*gadgets*) que ofrecen funcionalidades de la forja, de este modo el usuario puede seleccionar de entre todas ellas las que considere oportunas para construir su propia aplicación final, totalmente personalizada a nivel de usuario. Relacionando esto con el apartado de los modelos de integración, se puede afirmar que el diseño planteado para la capa de cliente facilita la integración en el mismo, ya que si el usuario no ve en el repositorio ningún gadget que se ajuste a sus necesidades,

puede desarrollar uno propio e integrarlo en la aplicación sin coste alguno (que no sea el del desarrollo del mismo).

EzForge en el lado cliente utiliza un navegador web, que actúa como contenedor para una plataforma de mashup que sirve a su vez como contenedor y entorno de ejecución de los gadgets. Existen en el mercado diversas plataformas de mashup, si bien por la que se ha optado para el desarrollo de EzForge es por la plataforma de mashup **EzWeb**, desarrollada dentro del propio proyecto Morfeo. Puesto que EzWeb estaba en desarrollo, se realizó una primera interfaz estática con EXT que permitiera comenzar a realizar pruebas de integración de los recursos de EzForge; en diciembre se comenzó la migración a la plataforma EzWeb.

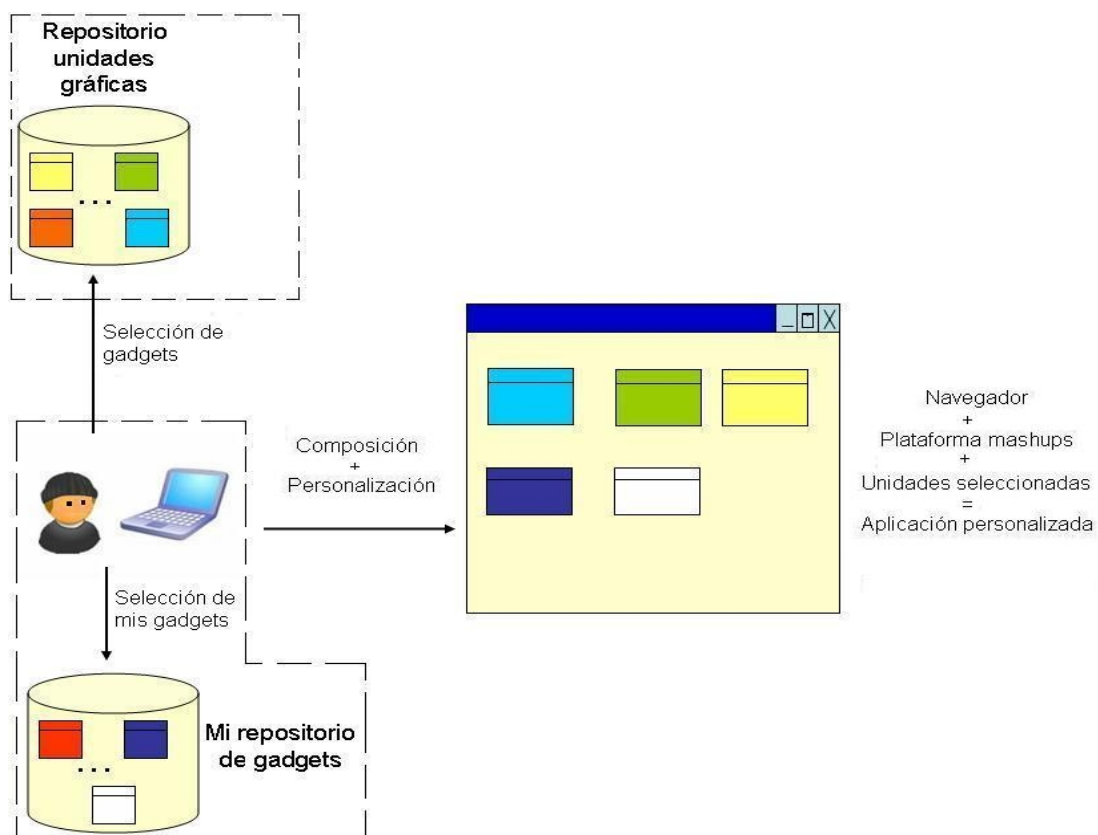


Figura 4: Lado cliente de EzForge

3.2 Lado Servidor

En este apartado se detallan los elementos que componen la arquitectura de EzForge en el lado del servidor, tanto lo que se ha denominado núcleo de EzForge como la capa de adaptadores propia de cada sistema a integrar con EzForge.

3.2.1 Núcleo de EzForge

El núcleo de EzForge se ha desglosado en tres apartados, los cuales se exponen brevemente a continuación para luego pasar a explicarlos más detalladamente:

- **Recursos EzForge:** puesto que se ha planteado una arquitectura REST, el elemento fundamental son los recursos. Estos recursos modelan las funcionalidades que ofrece la forja o CDE. A parte de los recursos que representan elementos de la forja, existen otro tipo de recursos operadores (transformaciones, agregaciones, etc.).
- **Modelo de Datos:** es la infraestructura de datos que utiliza EzForge para su propia gestión.
- **Autenticación:** es el mecanismo de validación y autenticación de usuarios que utiliza EzForge. El tema de autenticación en EzForge es complejo porque aparte de gestionar los usuarios de EzForge, debe proporcionar algún mecanismo que permita asociar a estos usuarios 'EzForge' credenciales de usuarios de sistemas de backend, con el fin de poder acceder a esos sistemas integrados con EzForge con los permisos pertinentes.

➤ Recursos EzForge

La capa de recursos permite crear una abstracción de los servicios proporcionados por la forja, de manera que se ofrezca una API común independientemente de la herramienta de backend que proporcione ese servicio. Esta capa de recursos ofrece además servicios necesarios para el propio funcionamiento de la forja, como gestión de usuarios, proyectos, funcionalidades. Estas interfaces tienen implementaciones cuya lógica de negocio puede desencadenar diversas invocaciones sobre otros recursos que finalmente desembocan en el acceso a los backend definidos en la capa de integración, logrando así la consecución de sus acciones. Esta capa, por tanto, sigue un patrón arquitectónico del tipo *pipes and filters* que recuerda estrechamente a los principios de diseño de los sistemas UNIX, en los cuales es posible implementar operadores complejos a partir de operadores más sencillos.

Los recursos pueden ser accedidos mediante un identificador global, la URI (*Uniform Resource Identifier*) del recurso. La forma de acceder al recurso es mediante HTTP, y la forma en que se haga (GET, POST, PUT, DELETE) denotará la semántica de la operación a realizar sobre el mismo:

- **GET:** implica la consulta del recurso, devuelve el XML que lo representa
- **POST:** implica la creación de un nuevo recurso
- **PUT:** implica la actualización de un recurso ya existente

- **DELETE:** implica el borrado de un recurso ya existente

En la definición de los recursos se ha establecido una categorización atendiendo a las distintas áreas de negocio que se han identificado, estableciéndose las siguientes categorías:

- **Usuarios:** recursos relacionados con la gestión de los usuarios registrados en la forja y sus habilidades.
- **Privilegios:** recursos relacionados con la gestión de los roles existentes en la forja así como los permisos que estos roles conllevan.
- **Wiki:** recursos relacionados con la gestión de las wikis asociadas a los proyectos albergados en la forja.
- **Código Fuente:** recursos relacionados con la gestión de las fuentes almacenadas en la forja (código fuente, versiones, ...).
- **Tareas:** recursos relacionados con la gestión y planificación de las tareas a realizar por los equipos de desarrollo de los proyectos.
- **Proyectos:** recursos relacionados con la gestión de los proyectos almacenados en la forja.
- **Sistemas de Backend:** recursos relacionados con la gestión de las funcionalidades, sistemas de backend y adaptadores integrados con EzForge.

➤ **Modelo de datos de EzForge**

EzForge posee un modelo de datos relacional, totalmente independiente de los modelos de datos que posean los sistemas que se puedan integrar en la forja. Si se necesita acceder a los modelos de datos de estos sistemas de backend se hará a través de adaptadores que los recubran, ofreciendo una interfaz REST del sistema en cuestión.

El modelo de datos de EzForge está enfocado a mantener información que se puede dividir en tres categorías:

- **Usuarios:** información relativa a los usuarios registrados en EzForge. También se gestiona la correspondencia entre usuarios Ezforge y usuarios de los sistemas de backend, ya que cuando se acceda a estos, será necesario acceder con cuentas de usuarios existentes en los mismos.
- **Proyectos:** información relativa a proyectos registrados directamente en EzForge. Los proyectos que estén en sistemas de backend, serán accedidos a través de éstos, más en concreto a través de sus adaptadores.
- **Sistemas de backend:** información referente a las funcionalidades proporcionadas por la forja, así como con los sistemas de backend que proporcionan dichas funcionalidades y los adaptadores que es posible utilizar para acceder a los mismos.

A continuación se muestra el modelo E/R diseñado en EzForge:

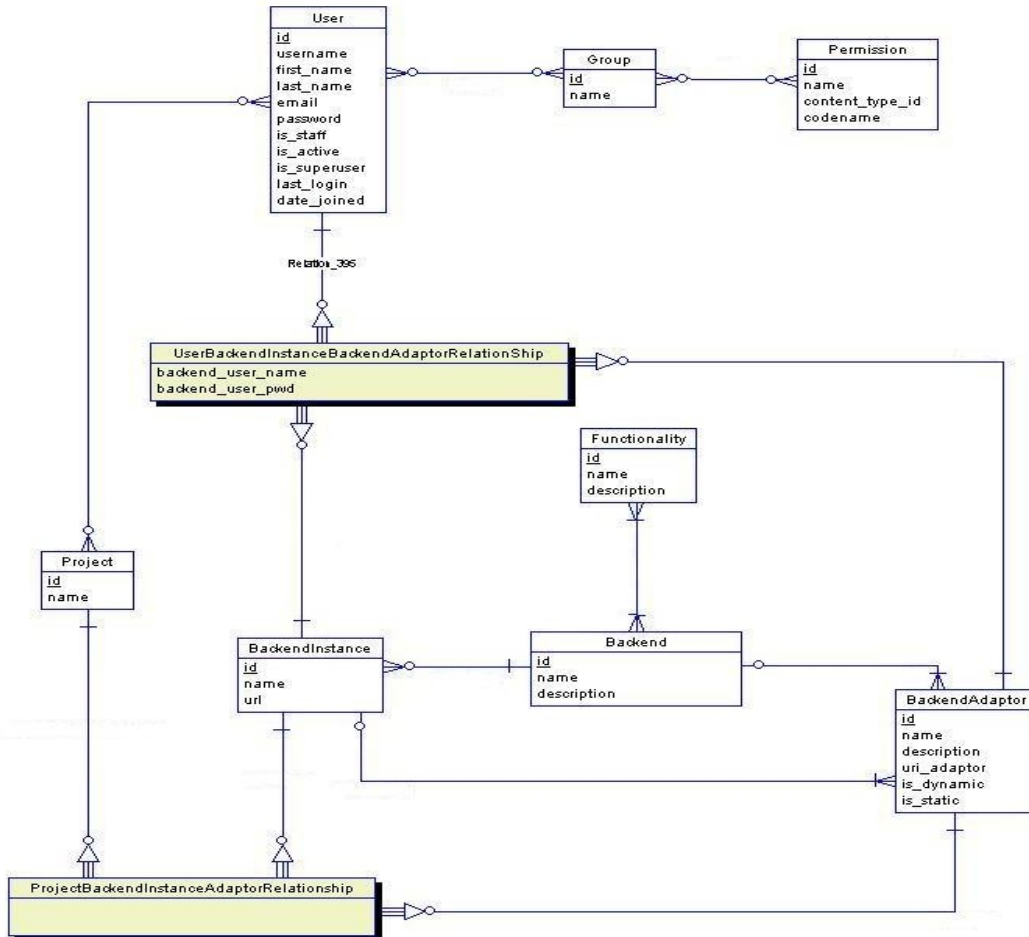


Figura 5: Modelo Entidad/Relación de EzForge

➤ Modelo de autenticación de EzForge

En sistemas distribuidos como EzForge uno de los principales puntos conflictivos es el tema de la validación y autenticación de usuarios. Los requisitos que se han tenido en cuenta a la hora de diseñar el sistema de validación-autenticación de EzForge son los siguientes:

- **Validación única (*single sign on*) y federación de identidades:** EzForge posee una arquitectura distribuida, en la cual los servicios que se encuentran dispersos por la red tienen sus propios repositorios de identidad. Se desea que el usuario solamente se valide una vez, y que el sistema sea capaz de acceder a los servicios a los que el usuario tenga acceso con las credenciales propias de ese servicio, sin necesidad de tener que validarse nuevamente.

- **Encapsulación de los proveedores de identidad de los sistemas legacy:** Algunos de los servicios integrados en EzForge han sido encapsulados para poder ser accedidos vía REST, pero siguen teniendo sus sistemas de autenticación y gestión de identidad. Con el fin de que sea posible federar las identidades se hace necesaria la integración de estos sistemas en la infraestructura de autenticación de EzForge.
- **Autenticación a lo largo de cadenas de confianza:** El núcleo de Ezforge se basa en la definición de una serie de recursos que pueden ser adaptadores, agregadores y conectores, los cuales se irán componiendo para conseguir las funcionalidades deseadas. En este sentido se hace necesaria la existencia de un mecanismo que gestione las credenciales de modo que no sea necesario ir pasándolas a lo largo de toda esa cadena de posibles composiciones de servicios.

Estos requisitos son los que han guiado el diseño de la arquitectura de autenticación de EzForge, para la cual se ha planteado:

- **Gestor de autenticación:** posee una cartera que almacena las credenciales de los usuarios EzForge en los sistemas integrados con EzForge, el acceso a esta cartera está protegido mediante el típico mecanismo de usuario/clave (usuario y clave EzForge).
- **Autenticación delegada:** algunos recursos por la información que manejan necesitan ser protegidos de accesos no autorizados, por esta razón y para cumplir el requisito de la validación única, el gestor de autenticación es utilizado por los recursos para obtener las credenciales de usuario, las cuales indican si detrás de una determinada petición hay un usuario que tenga permiso para realizarla.

Estas ideas quedan reflejadas en la siguiente figura.

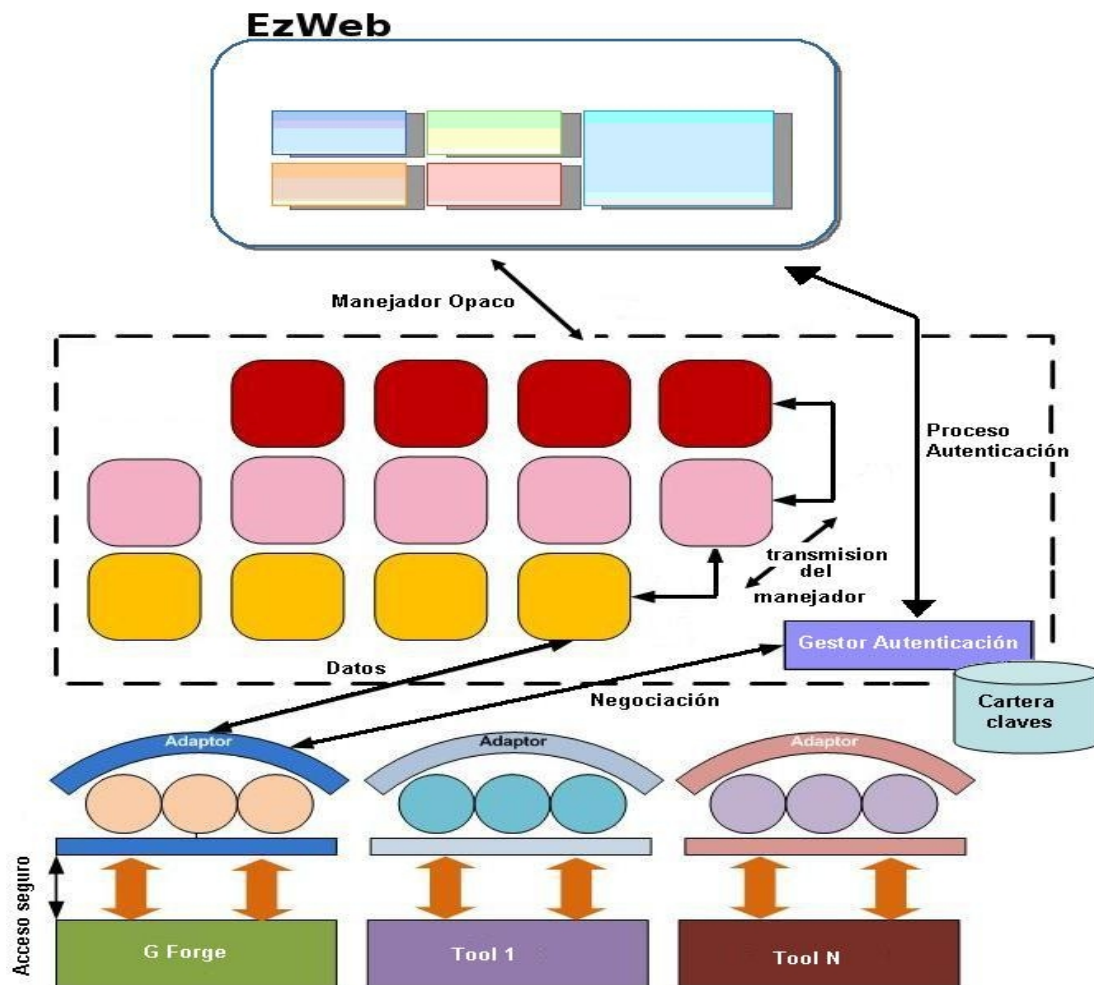


Figura 6: Modelo de autenticación de EzForge

3.2.2 Adaptadores de los sistemas de backend

Los adaptadores de los sistemas de backend son los elementos que deben aportar las herramientas que deseen integrarse con EzForge para garantizar la correcta intercomunicación entre ellos (EzForge <-> sistemas de backend). La complejidad del desarrollo de estos adaptadores dependerá de la arquitectura del sistema a integrar, según esto podemos establecer la siguiente clasificación:

- **Sistemas no Web 2.0:** si el sistema a integrar no tiene una arquitectura REST, será necesario realizar un proceso de 'RESTificación' que tiene como finalidad recubrirlo para que exporte una interfaz REST. A continuación será necesario aplicar una serie de flitros (podrían consistir simplemente en una transformación XSL) para que los recursos de estos sistemas, sus XMLs, se ajusten a lo que los recursos de EzForge esperan. Para ello en EzForge se han definido una serie de esquemas

que serán utilizados para guiar los procesos de transformación a realizar.

- **Sistemas Web 2.0:** si el sistema a integrar tiene una arquitectura REST, estaríamos como en el caso anterior una vez se ha realizado el proceso de 'RESTificación', con lo cual solo sería necesario desarrollar los filtros de adaptación a EzForge.

4 FUNCIONALIDADES DEL NÚCLEO EzFORGE

El primer paso antes de abordar el diseño y desarrollo de EzForge ha sido analizar las funcionalidades que debe cubrir el núcleo de forja. Para ello se han analizado un conjunto de forjas existentes y ampliamente usadas (como son G-Forge, Trac, Savannah) para ver cuales son las funcionalidades que suelen proporcionar las forjas de desarrollo. Estas funcionalidades se han dividido en dos grupos, las funcionalidades básicas necesarias para el funcionamiento de la forja en sí, y funcionalidades que aunque no son necesarias son bastante útiles y habituales en las forjas de desarrollo.

4.1 Funcionalidades básicas de una forja de desarrollo

Se entienden por funcionalidades básicas de una forja de desarrollo aquellas funcionalidades que son necesarias para el funcionamiento básico de la forja. Dentro de este grupo se han identificado las siguientes funcionalidades:

- Gestión de usuarios: proporciona las funcionalidades necesarias para gestionar los usuarios de la forja, tanto su registro como su autenticación en la forja y el control de la realización de ciertas operaciones en función de si está o no autenticado.
- Gestión de roles y permisos: permite la creación de roles y la asignación de roles a los distintos usuarios, los cuales en función del rol puede realizar unas operaciones u otras. Las operaciones que se pueden llevar a cabo con cada rol se definirán mediante los permisos.
- Gestión de proyectos : los proyectos son uno de los elementos fundamentales en las forjas de desarrollo, por esta razón se ha identificado la gestión de proyectos como una funcionalidad básica que debe proporcionar el núcleo de la forja. Se permitirá la creación de proyectos, que un usuario se pueda registrar en un proyecto, el seguimiento de los mismos.

4.2 Funcionalidades habituales en una forja de desarrollo

Las funcionalidades habituales en una forja de desarrollo son aquellas que no siendo clave para el funcionamiento de la forja proporcionan características bastante útiles para el desarrollo colaborativo de proyectos. Dentro de este grupo hay bastantes funcionalidades, pero de momento nos hemos orientado a identificar aquellas que se van a cubrir durante esta primera fase del proyecto, y que son:

- Wiki: esta herramienta de documentación colaborativa es muy utilizada entre las comunidades de software libre para poder realizar la documentación de los proyectos en paralelo y entre los distintos desarrolladores.

- Gestión de tickets: esta funcionalidad permitirá el registro y seguimiento de tickets en un proyecto. El concepto de ticket engloba los conceptos de bug, tarea y mejora. Esta herramienta permitirá la notificación de un bug o tarea a realizar así como peticiones de mejora por parte de los usuarios, estos tickets podrán ser asignados a un miembro del proyecto para que se encargue de llevarlos a cabo.
- Navegador de código: esta herramienta permitirá navegar por los ficheros de código existentes en un proyecto de desarrollo, dando la posibilidad de visualizar el código o descargarse el fichero correspondiente.

5 INTEGRACIÓN DE NUEVAS HERRAMIENTAS

Una de las principales ventajas de esta arquitectura es que permite la integración de nuevas herramientas de una forma sencilla. La idea es proporcionar un núcleo de forja que pueda ser ampliado como con nuevas funcionalidades que vayan cubriendo necesidades que puedan ir surgiendo y no hubieran sido tenidas en cuenta en el desarrollo del núcleo de EzForge.

El proceso a seguir para la introducción de nuevas herramientas en la forja depende de si la funcionalidad cubierta por dicha herramienta está ya incorporada o no. En este apartado se describirán ambos casos.

5.1 Integración de herramientas cuya funcionalidad ya está incorporada

Puede ocurrir el caso de que se quiera integrar una herramienta cuya funcionalidad ya está modelada en la capa de recursos genéricos, porque se ha considerado interesante o porque hay alguna otra herramienta similar ya integrada en la forja. En este caso el nivel de recursos genéricos Vulcano o EzForge contendría un conjunto de recursos modelando esta funcionalidad, con lo cual para integrar la nueva herramienta sólo tendríamos que desarrollar el adaptador REST correspondiente.

5.1.1 Desarrollo de un adaptador REST para EzForge

Cuando se integra una herramienta en la forja es necesario proporcionar una API para que la capa de recursos genéricos pueda comunicarse con las herramientas o servicios subyacentes. Esta API tiene que ser una API REST que ofrezca acceso a los recursos requeridos por los recursos genéricos y mediante las operaciones que se especifiquen. Para ello cada vez que se incorpore una funcionalidad nueva en EzForge es necesario especificar las operaciones y recursos que se requieren de los adaptadores.

5.2 Integración de herramientas que cubren nuevas funcionalidades

Si se va a integrar una herramienta que proporciona alguna funcionalidad aún no cubierta por la forja, el proceso es algo más complejo, ya que además de realizar el trabajo indicado en el punto anterior, es necesario diseñar e implementar los recursos en la capa EzForge o Vulcano que modelan la nueva funcionalidad.

Como ejemplo, si se desea proporcionar desde la forja la funcionalidad de foros, en primer lugar es necesario modelar con recursos los conceptos genéricos para ello, como puede ser el concepto mensaje, topic, etc. Una vez definidos e implementados estos recursos e identificados los recursos y operaciones que se necesita que proporcionen los adaptadores de las

herramientas subyacentes, se puede proceder a desarrollar el adaptador REST para la herramienta concreta que se quiera integrar en la forja para dar esta funcionalidad.

6 CONCLUSIONES Y LÍNEAS FUTURAS

Se irán generando nuevas versiones de este documento en los sucesivos años de proyecto, los apartados en los que se espera se puedan producir cambios son:

- Arquitectura de EzForge: este apartado podrá sufrir modificaciones para incorporar nuevos servicios que se incluyan en EzForge, o por avances en la investigación en torno a la autenticación e integración de herramientas, aspectos en los que nos centraremos bastante en el próximo año.
- Funcionalidades del núcleo EzForge: sufrirá modificaciones si se incorporan nuevas funcionalidades al núcleo.
- Integración de nuevas herramientas: sufrirá modificaciones a medida que se vayan integrando nuevas herramientas en la forja.

REFERENCIAS