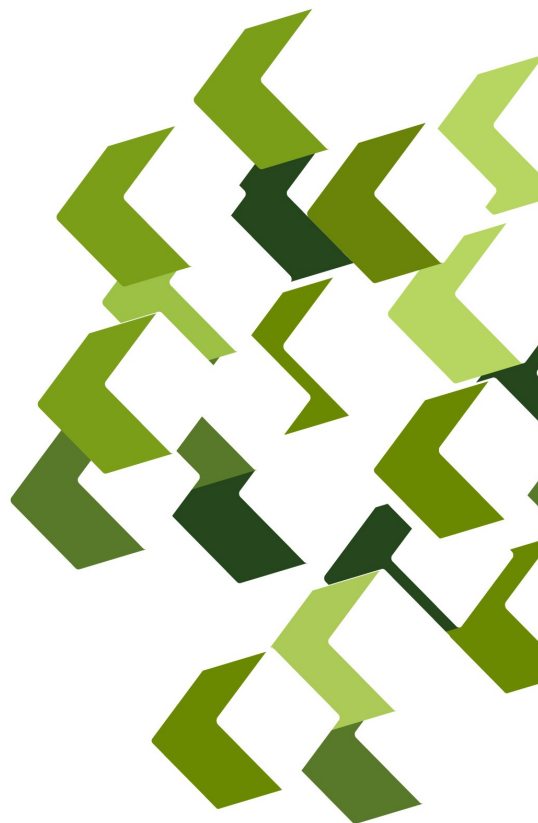




**cenatic**

Centro Nacional de Referencia  
de Aplicación de las TIC basadas  
en fuentes abiertas



## 11PRO001CT015

Documento de conclusiones de análisis y definición de la solución

[www.cenatic.es](http://www.cenatic.es)

Versión: v01r00  
Fecha:19/07/11



# 11PRO001CT015

Documento de conclusiones de análisis y definición de la solución



## Índice

1 INTRODUCCIÓN .....	5
2 CATALOGO DE REQUISITOS.....	6
2.1 Anotaciones a los requisitos iniciales.....	7
3 DEFINICIÓN FUNCIONAL DE LA SOLUCION.....	10
3.1 Descripción funcional.....	10
3.2 Esquema bloques funcionales.....	13
3.3 Descripción grupos funcionales o subsistemas .....	15
3.3.1 Almacenamiento.....	15
3.3.2 Virtualización-Cloud.....	15
3.3.3 Repositorios.....	16
3.3.4 Infraestructuras de proyectos.....	16
3.3.5 Búsquedas.....	16
3.3.6 Capa de aplicación.....	17
3.3.7 Minería de datos .....	21
3.3.8 Orquestación de servicios.....	21
3.3.9 Comunicaciones asíncronas.....	22
3.3.10 Asistentes.....	22
3.3.11 Herramientas colaborativas.....	23
3.3.12 Gestión de comunidades.....	24
3.3.13 Redes sociales.....	24
3.3.14 API.....	25
3.3.15 Conectores.....	27
4 ESTUDIO PREVIO DE ALTERNATIVAS DE SOLUCIÓN.....	27
4.1 Diagrama de arquitectura.....	27
4.2 Propuesta de tecnologías a incluir en el estudio.....	28
4.2.1 Almacenamiento.....	28
4.2.2 Virtualización - cloud.....	29
4.2.3 Repositorios.....	29
4.2.4 Buscador.....	30
4.2.5 Minería de datos.....	31
4.2.6 Infraestructura de proyectos.....	31



## 11PRO001CT015

Documento de conclusiones de análisis y  
definición de la solución



4.2.7 Orquestación de servicios.....	31
4.2.8 Comunicaciones asíncronas.....	32
4.2.9 Herramientas colaborativas.....	32
4.2.10 Capas de aplicación, asistentes, API, gestión de comunidades y redes sociales.....	33
4.3 Criterios de evaluación.....	35
5 MATRIZ DE TRAZABILIDAD DE REQUISITOS FUNCIONALES .....	36
6 PROPUESTA DE TAXONOMÍAS DE PROYECTOS.....	42
6.1 ¿Para qué y para quién va destinado?.....	47
6.1.1 Audiencia.....	47
6.1.2 Sector .....	47
6.1.3 Clasificación funcional .....	47
6.2 ¿Situación del proyecto?.....	48
6.2.1 Estado de desarrollo.....	48
6.2.2 Tamaño de la Comunidad.....	48
6.3 Información Técnica.....	48
6.3.1 Entorno.....	48
6.3.2 Idioma principal .....	49
6.3.3 Localizaciones del proyecto.....	49
6.3.4 Lenguaje de programación.....	49
6.3.5 Tipo de licencia.....	49
6.3.6 Sistema operativo.....	50
6.3.7 Base de Datos usada.....	50
7 CONCLUSIONES.....	51



## 1 INTRODUCCIÓN

Las tecnologías basadas en software de fuentes abiertas son cada vez más apreciadas por las empresas, perfilándose como parte importante de sus estrategias de desarrollo competitivo en una sociedad en red, globalizada, interconectada e interdependiente; al mismo tiempo, cada vez es mayor el número de empresas que adoptan el software de fuentes abiertas como modelo de negocio.

En este sentido, CENATIC ha considerado emprender un proyecto tecnológico dirigido fundamentalmente a :

- Seleccionar tecnologías existentes y evolucionarlas hasta las llamadas forjas de nueva generación, más cerca de un entorno colaborativo moderno, modular y orientado a servicios que de las forjas tradicionales.
- Evolucionar las tecnologías de forja de nueva generación para soportar los nuevos retos de cambio de paradigma.
- Dotar a las AAPP de un espacio donde compartir eficientemente el software y conocimiento desarrollado a otras AAPP del ámbito nacional e internacional.
- Utilizar arquitecturas abiertas y orientadas a servicio que permitan personalización, interconexión y evolución futura.

En este sentido, CENATIC ya ha abordado el proyecto: **Forja de Nueva creación. Análisis técnico, funcional y organizativo** con referencia **10PRO002108CT043**. Este proyecto, en sus diferentes fases, ha dado como resultado la obtención de entregables que habilitan nuevas iniciativas concretas dentro de la estrategia general. El más destacado de éstos es el **Análisis de Requerimientos de la Forja de Nueva Generación**.

Como conclusión más importante, se ha establecido que **no existe ninguna solución de forja actual que cumpla con los requerimientos establecidos** Es por ello que Cenatic, lanza el actual proyecto cuyo objetivo es continuar la hoja de ruta hacia la Forja de Nueva Generación que cubra las necesidades detectadas.

El actual proyecto consistirá en la construcción de un prototipo que demuestre las funcionalidades principales de dicha forja, detallando también la arquitectura de referencia que va a darle soporte. Asimismo este proyecto servirá para determinar las condiciones técnicas de futuras fases del mismo como el diseño y la construcción.



## 11PRO001CT015

Documento de conclusiones de análisis y definición de la solución



### 2 CATALOGO DE REQUISITOS.

El punto de partida del actual proyecto es el mencionado análisis de necesidades del proyecto anterior, que ya conformó un primer catálogo de requerimientos.

A continuación se presenta dicho conjunto de requerimientos técnicos y funcionales para el desarrollo a medida de una Forja de Nueva Generación.

Estos requerimientos han superado seis fases:

1. Relación abierta de necesidades por parte de CENATIC con arreglo a condicionantes legislativos de las AA.PP y la AGE
2. Relación abierta de necesidades de grupos de expertos a través de sesiones de trabajo
3. Entrevistas a expertos nacionales e internacionales en materia del presente y el futuro de las forjas
4. Análisis de mercado sobre las forjas existentes
5. Análisis sobre las tendencias y buenas prácticas observadas en el empleo de forjas
6. Análisis de las respuestas a cuestionarios sobre la realidad de las entidades potencialmente usuarias de una FNG

Fruto de este trabajo se han obtenido un número de requerimientos funcionales. Cada uno de ellos identifica:

**Autor:** Persona, entidad o grupo que propone el requerimiento en una fase inicial

**ID:** Código unívoco

**Categoría:** Categoría a la que pertenece

**Puntos:** Importancia sobre 100 puntos

**Descripción:** Enunciado sencillo, redactado en lenguaje no necesariamente técnico, que propone un requerimiento a ser abordado por la FNG.

**Motivación:** Justificación detalla de la necesidad de abordar este requerimiento. En ocasiones se presenta una reflexión con origen en los análisis contrastados de las



## 11PRO001CT015

Documento de conclusiones de análisis y definición de la solución



fases mencionadas. La motivación sirve en ocasiones para detallar más el objetivo del requerimiento (cuando esto sucede, se marca en negrita).

**Casos de uso:** Listado de los casos de uso (cuando aplican o no son triviales de obtener) sobre ese requerimiento y que permiten ver con ejemplos de uso el funcionamiento específico más relevante del requerimiento.

**Suposiciones:** Prerrequisitos o escenarios de partida que condicionan la viabilidad del requerimiento tal y como se ha expresado en la Motivación y Casos de uso.

**Otros comentarios:** Cualquier otro comentario que pueda ser merecedor de un capítulo aparte

La información detallada (codificación, peso, correlación con otros requisitos funcionales, casos de uso, suposiciones, etc...) de cada uno de los requerimientos puede verse en el documento: FPR\_Cenatic\_FNGv1.1.odt

### 2.1 Anotaciones a los requisitos iniciales

Dichos requerimientos han sido posteriormente revisados a fin de aclarar posibles malinterpretaciones y efectuar una valoración más equilibrada con los objetivos del proyecto. A fin de identificar aquellos requerimientos que han sufrido algún tipo de modificación o han necesitado de una aclaración de los objetivos, se les ha asignado un número de versión.

A continuación se detallan las modificaciones a los requerimientos iniciales.

La tabla con el detalle de las modificaciones puede consultarse en el documento anexo [Anexo I. Catalogo de requisitos versionado V01r00.ods](#)



# 11PRO001CT015

## Documento de conclusiones de análisis y definición de la solución



Anexo I. Catálogo de requisitos versionado						
REQ Puntuación 90-100	Descripción	Peso	Versión	Relacionado	Anotaciones	
FNGREQ43	La FNG debe adherirse a la tendencia observada de usar sistemas de control de versiones distribuidos al tiempo que soportar el uso de los centralizados (SVN) mediante hooks traductores	100	0		El sistema de control de versiones será GIT, Bazaar o Mercurial. Se dispondrá de sistemas de exportación del código de estos a sistemas Subversion.	
FNGREQ33	La FNG, con el objetivo de integrar en todo el ciclo de vida de un proyecto el aspecto legal, debería contar con un gestor de licencias tipo y un potente buscador alrededor de ellas.	100	2		La ForjaNG deberá de contar con un asistente para ayudar a la selección del tipo de licencia mediante preguntas eliminatorias. Asociación de cabeceras de licencias en cada fichero en base a la licencia Asistente de selección en base a preguntas simples. Generar metadatos de la licencia	
FNGREQ39	La FNG debe facilitar la definición de subforjas con Identidad visual diferenciada.	100	2		Se acuerda que cuando se esté navegando por proyectos alojados en diferentes subforjas se mantendrá la línea gráfica de la forja desde la que se está buscando. El comportamiento será similar al actual de Morfeo. Se debe añadir como restricción la no obligatoriedad de mantener múltiples dominios para las subforjas puesto que podría ser difícil de gestionar por el MPTAP. También se especifica que el nivel normal de subforjas será el de Comunidades Autonomas, no se prevén subforjas de Administraciones Locales. En el caso de componentes que no estén soportados directamente por la forja, se estudiará al menos la creación de un marco que indique que el servicio ofrecido es externo.	
FNGREQ42	La FNG debe estar disponible en multilinguaje y soportar internamente localización	100	1		Se debe proveer soporte de localización para todos los elementos de libre descripción de las fichas de proyecto, por lo que el alta de proyecto debe permitir y favorecer para todos los campos de libre descripción (Descripción del proyecto, about us, contacta, etc...) la introducción de los literales en múltiples idiomas.	
FNGREQ45	Se requiere la inclusión de herramientas 2.0 de gestión del conocimiento en la forja.	100	0			
FNGREQ32	En línea con las necesidades de agregar información de forjas existentes, la FNG requiere una meta agregador que añada semántica (tags, categorías, taxonomías) a los proyectos sindicados.	100	2		Se explican los conceptos de sindicación y federación, indicando MPTAP que no van a hacer uso del nivel de sindicación, sino del de federación. Sindicación cuando consume información de forma anónima. Federación, cuando se consume con autenticación y Se es capaz de hacer push hacia otra forja. El sistema de tags no debería ser excesivamente abierto al usuario, se deberían proponer, en principio los tags mas comunes de las forjas actuales.	
FNGREQ46	La FNG debe tener una arquitectura y una vocación de descentralización de servicios de modo que puedan elegirse fácilmente proveedores externos para cada una de las grandes patas funcionales verticales en ella contenidas. Esa descentralización requiere la capacidad de agregarlos en la FNG.	100	1		Se propone la ampliación del requisito incluyendo la descripción de los tres tipos de proyecto que se alojarán en la forja: - Proyectos Internos. Todos sus servicios estarán soportados en la FNG - Proyectos Mixtos. Algunos de los servicios pueden estar soportados por proveedores externos. - Proyectos Externos. Solo se incluirá en la forja la ficha de proyecto, el resto de Servicios serán externos.	
FNGREQ60	La FNG debe disponer de un completo set de herramientas de gobierno de proyectos. Aprobación de hitos, generación y publicación de versiones/releases, tiempos de liberación, gestión de tareas y bugs. Este gobierno de proyecto debe tener vocación upstream cuando sea posible	100	2		Generar una visión del proyecto a nivel de gestión del mismo, no del código. Es decir, asociar hitos a releases y otras buenas prácticas de gestión de proyectos. Esta información será mostrada en la pagina de presentación del proyecto	
FNGREQ63	La FNG debe disponer de indicadores claros y precisos sobre la salud de un proyecto	100	1		Dar valor también a la parte social (Comunidad), a la misma importancia que las estadísticas del código. Si son proyectos mixtos o externos, el encargado deberá de cumplimentar los datos a fin de que sea consumido por la forja. Sin estos, no se computan	
FNGREQ01	De forma similar al funcionamiento de plataformas de redes sociales, debe resultar muy fácil y sencilla la comunicación entre los integrantes de la forja.	100	0			
FNGREQ02	La FNG debe facilitar en todo lo posible la colaboración de todos los interesados, aportando información precisa; también debe potenciar especialmente la incorporación de código fuente a un proyecto.	95	1		Al dar de alta un proyecto, el sistema de recomendaciones quién y cómo puede ayudarme a determinados acciones.	
FNGREQ04	Se requiere que la FNG disponga de las herramientas necesarias para facilitar el trabajo colaborativo tanto sincrónico como asincrónico. Ejemplos válidos son Google Docs o Etherpad.	90	1		Podría ser una aplicación de escritorio que se sincronizase cuando tuviera conexión, Al estilo Dropbox	
FNGREQ07	Para facilitar la transferencia completa de proyectos hacia/desde la FNG, ésta debe permitir importación y exportación de proyectos entre forjas ya existentes y con las que exista posibilidad por compatibilidad de modelo de entidades.	100	2		Al proceso de ETL sobre Gorge ya mencionado en el requisito se debe añadir la necesidad de al menos proveer soluciones para la migración de Mantis, SVN y evaluar posibles migraciones de otros elementos como wikis y blogs.	
REQ Puntuación 70-90	Descripción		Versión	Versión	Relacionado	Anotaciones
FNGREQ06	La gestión administrativa de la FNG debe resultar ágil, liviana e integral en cuanto a todos los componentes funcionales.	85	1			El perfil del admin NO es un admin de sistema, tiene que ser alguien sin conocimientos técnicos pero sí gestiones proyectos, usuarios, grupos, tags, ...
FNGREQ08	La FNG debe alcanzar un grado de usabilidad muy alto en el proceso específico de subir nuevos proyectos. Un ejemplo puede ser asistentes avanzados.	78	2			Ampliar el modelo de alta sencilla de proyecto indicando los datos mínimos a incluir: - Nombre completo de la iniciativa. - Nombre corto de la iniciativa: identificador único del sistema - Resumen: descripción corte de uno o varios párrafos - Descripción: cuentan con más detalle lo que hace el proyecto, sus objetivos, finalidad, funcionalidades, aspectos técnicos etc - Organismo Responsable. - Contacto. - Explicación de razón de solicitud y en la forja herramientas que necesitarán inicialmente - Administradores de la iniciativa (nombre, apellidos, y cuentas de usuario) Se recogerán automáticamente. Las clasificaciones en categorías serán rellenas mediante formulario por el usuario y verificadas por terceros
FNGREQ44	La FNG debe poder usar autenticación vía OpenID	70	2			Añadir la necesidad de identificación mediante certificado o dni electrónico, así como la Integración con directorios LDAP
FNGREQ52	La FNG debe presentarse al usuario gestor del proyecto en términos en los que pueda realizar esa tarea sin conocimientos técnicos	70	1			Aprobar usuarios, buscar colaboradores, desplegar nuevos servicios (wiki, blog,...) Posiblemente con uso de asistentes en tareas complejas
FNGREQ09	La FNG debe promover un modelo de federación entre forjas mediante la definición de un conjunto de metadatos muy completos que facilita la comunicación desatendida entre ellas.	70	1	FNGREQ32		Relacionado con el FNGREQ32 se indica que la integración débil entre forjas sería similar a la sindicación y la integración fuerte se asocia con la federación. Esta integración afectará a personas y proyectos
FNGREQ47	La FNG debe disponer de una completa API de programación para facilitar la creación de plugins orientados a gestión de metadatos y al aprovechamiento de la información generada en la comunidad de usuarios.	70	0			



# 11PRO001CT015

## Documento de conclusiones de análisis y definición de la solución



REQ Puntuación 50-60	Descripción	Peso	Versión	Relacionado	Anotaciones
FNGREQ10	La FNG debe contar con un Cuadro de mando orientado a perfiles de gestión y dirección.	68	0		
FNGREQ15	Se requiere personalización de la visualización de la información de la FNG dependiendo de diferentes perfiles de usuario. Esta personalización debe estar contemplada con mentalidad mashup y modular.	68	1	FNGREQ15-BIS	Se separa la parte de mashup modular, que pasa a ser un requerimiento de menor peso
FNGREQ12	La FNG debe disponer de un guión claro para animar y facilitar el cubrir toda la información necesaria de los proyectos. Uno de los objetivos es dar a conocer la realidad del proyecto de forma que su valor sea conocido fácilmente y sea usado. Se evitará así parte de los nuevos proyectos nacidos del desconocimiento de otros similares en marcha.	65	1		Se enviarán notificaciones a fin de que se complemente la información y se motive la introducción de metadatos
FNGREQ11	La FNG debe proponer y facilitar un ciclo de vida completo del proyecto, documentando cada fase y teniendo en consideración hitos como la propia finalización de la construcción del producto en los términos que sean apropiados.	65	1		Sugerir hitos mínimos para que el proyecto se le pueda realizar un seguimiento. Sin fechas, sin hitos, no hay forma de hacer el seguimiento del proyecto.
FNGREQ14	La FNG debe orientar en las mejores prácticas de elección de metodologías de gestión, desarrollo y pruebas al tiempo que alerta de posibles riesgos del proyecto.	63	1		Se matiza que está enfocado a best practice o BBDD de conocimiento y se podría resolver mediante una wiki o mediante una serie de checklist
FNGREQ18	Se requiere incorporar ciertas características del Entorno social/2.0.	60	0		
FNGREQ19	La FNG debe facilitar en extremo que la carga de información vía documentos pueda realizarse en procesos batch o por lotes.	60	0		
FNGREQ35	La FNG requiere facilitar o construir herramientas de carga de documentos desplegadas en entorno escritorio.	60	1		Se hará un prototipo a la carga de documentos mediante carpetas remotas, tipo webdav
FNGREQ38	Se requiere la existencia de un módulo funcional en la FNG que sea capaz, en función de métricas específicas, de medir la calidad del código fuente de los proyectos.	60	1		Implementación similar a Sonar. No existen actualmente herramienta con soporte para múltiples lenguajes.
FNGREQ40	La FNG debe poder integrarse fácilmente con portales de redes sociales.	60	1		A través del bloque funcional de redes sociales, se procederá a efectuar acciones de Difusión mediante redes sociales. Ejemplo: si se crea un proyecto, este proyecto deberá de registrarse automáticamente en una de ellas.
FNGREQ41	La FNG debe adoptar las mejores prácticas en el reconocimiento del trabajo y la aportación de sus usuarios.	60	0		Debería haber una clasificación automática de usuarios y empresas, de forma que el sistema pueda luego hacer proposiciones de colaboradores a proyectos en función de características comunes.
FNGREQ48	La FNG debe disponer de un listado de desarrolladores, competencias y empresas con perfiles fáciles de analizar.	60	0		Esto incluirá el Karma, y niveles de capacitación en tecnologías. Se acuerda mostrar simplemente proyectos relacionados y a partir de ahí se podrá acceder a información sobre desarrolladores o empresas.
FNGREQ20	La FNG debe proporcionar soluciones y herramientas para que los proyectos puedan integrar directamente las traducciones y las localizaciones e informar del estado de estas dos actividades.	55	0		Transifex.org ofrece esas características
FNGREQ23	La FNG debe poder facilitar la personalización de la apariencia tanto para las entidades como para los proyectos.	53	0		Logotipo del proyecto Colores asociados a la hoja de estilos personalizable Imagen o color de fondo
FNGREQ24	Se requiere que exista en la FNG una predisposición a ofrecer ciertos servicios de automatización en dos puntos: - Generadores automáticos de código - Creación de esqueletos de proyecto tipo	53	0		- Generadores automáticos de código: que incluyen estructuras iniciales de repositorio de código (trunk, rama estable, rama experimental), ficheros de licencia, README, estructura de directorios (src, doc, etc) y, excepcionalmente y sólo si la tecnología del proyecto está bien determinada, el volcado de un snapshot concreto de un framework.  - Creación de esqueletos de proyecto tipo: que incluyen fases e hitos de ejemplo con fechas ficticias, tipología de tareas, wikis iniciales, entrada del blog inicial, etc.
FNGREQ25	La FNG debe tener una orientación clara hacia la nube tanto en su construcción interna como en la naturalidad en la integración con servicios de terceros ofrecidos en modalidad SaaS.	53	0		
REQ Puntuación 30-50	Descripción	Peso	Versión	Relacionado	Anotaciones
FNGREQ30	La FNG debe tener en mente siempre al usuario final, no necesariamente tecnológico y establecer fórmulas para establecer comunidad con ellos dentro de la FNG con el objetivo de acercarlos fácilmente a los diferentes productos o proyectos.	38	1		Los proyectos no tienen que ser necesariamente de software. Puede haber de muchos tipos.
FNGREQ34	La FNG requiere facilitar o construir herramientas de seguimiento de proyectos desplegables en entorno escritorio.	33	0		Estas herramientas harán uso de la API, incluyendo cuadros de mando y Serán multiplataforma
FNGREQ31	La FNG deberá incluir elementos de validación de hitos y entregables para posibilitar la traslación de los elementos de vinculación contractual entre proveedores y clientes.	35	1		Podría ser un campo del proyecto que será necesario validar por una persona a fin de que se consolide determinado ítem.
FNGREQ37	La FNG debe disponer de un servicio de creación de contenidos esencialmente formativos alrededor del proyecto. Se propone analizar el uso de terceras herramientas como LMS Moodle para cubrir esta necesidad.	50	2	FNGREQ48 FNGREQ41	Matizar con la necesidad de poder incluir información sobre procesos de certificación o empresas certificadas en la ficha de proyecto. Subir el nivel a 50.
FNGREQ49	La FNG debe facilitar la generación de bounties o recompensas por resolver determinadas necesidades	20	0		
FNGREQ58	La FNG debe estar construida siguiendo patrones de alta disponibilidad	20	0		
FNGREQ15-BIS	Se facilitará la personalización de la interfaz gráfica, que Debe de estar diseñada con mentalidad mashup y modular.	40	1	FNGREQ15	



### 3 DEFINICIÓN FUNCIONAL DE LA SOLUCION

Tomando los requisitos y las condiciones técnicas detalladas anteriormente, y tras realizar un análisis del *estado del arte* de las arquitecturas de las forjas que actualmente son referencia en el ámbito del Software de Fuentes Abiertas, y aunándola al tipo de arquitectura de sistemas que actualmente se desarrolla en sitios web de elevada concurrencia, se propone la siguiente definición funcional para el sistema.

#### 3.1 Descripción funcional

Se han encontrado deficiencias comunes en las actuales forjas, tales como: arquitecturas monolíticas, elementos duplicados, falta de orquestación de servicios, escasa orientación al modelo colaborativo. y un largo etcétera que hacen que la modificación de cualquiera de las forjas actuales a fin de satisfacer estas carencias sea una tarea más costosa que la creación de una nueva forja que solucione éstas deficiencias.

La Nueva Forja va a mantener un núcleo de componentes básicos comunes a la mayoría de las forjas actuales consistente en blogs, sistema de seguimientos de errores, wikis, listas de correos, sistemas de control de versiones, etc.

Y va a añadir una serie de elementos que se esperan obtener: gestión de métricas, gestión de licencias, sistemas de generación automática de código y documentación a partir de los contenidos en el mismo, herramientas de traducción, gestión de comunidades y su integración con redes sociales.

Se identifican tres tipos de proyectos que pueden hacer uso de la forja, sean éstos de generación de código o no. A saber:

- **1º Propios:** soportados íntegramente por los servicios propios a Proyectos internos de la forja
- **2º Externos:** completamente alojados en forjas externas compatibles (principalmente basadas en GForge o derivadas) pero manteniendo la ficha de proyecto en la Nueva Forja
- **3º Mixtos:** aprovechan servicios propios y externos, a consideración del gestor del proyecto

Este modelo de proyectos tan diversos va a requerir de la forja una gran capacidad de modulación, flexibilidad, intercomunicación, etc...

Como capacidad imprescindible de la forja se permitirá la creación de subforjas que podrán tener identidades visuales específicas para cada una. A la vez la forja debe tener la capacidad



de proveer servicios externos de forma transparente al usuario, aún siendo este informado de que el servicio es ofrecido por un ente externo.

La nueva Forja estará dotada de un mecanismo automático de tratamiento de la información, a fin de proveer capacidades de

- **Recomendación:** a partir de la información que los usuarios o proyectos generan, el sistema intentará encontrar ítems que pueden interesarle.
- **Agrupación:** a partir de la información generada, el sistema tratará de identificar ítems que compartan determinadas características, e intentar agruparlas.
- **Clasificación:** a partir de información ya categorizada, el sistema intentará clasificar correctamente aquellos que no lo han sido aún.
- **Frecuencia:** a partir de los grupos de ítem clasificados, el sistema intentará identificar qué ítems individuales usualmente aparecen juntos o relacionados.

Fruto del tratamiento automatizado de gran cantidad de información, el sistema será capaz de realizar una clasificación automática de usuarios, empresas, entidades, proyectos y otros ítem, a fin de proponer colaboradores para proyectos, contribuciones interesantes para un determinado perfil de desarrollador, etc...

La nueva Forja no va a imponer ninguna metodología de gestión de proyectos, pero sí se propondrán una serie de buenas prácticas comunes entre todo tipos proyectos (sean de índole informática o no), tales como objetivos, marca de hitos, previsiones de releases, ...

Como elemento fundamental de la forja se va a barajar el concepto de asistente, entendiendo como tal un sistema en el que mediante una sencilla interacción un usuario sin necesidad de conocimientos técnicos elevados pueda resolver algún aspecto de complejidad relevante. Ejemplos de asistentes que considerará la forja serán el de alta de proyectos, importación-exportación de proyectos, reconfiguración de proyectos y colaboración y búsqueda de perfiles.

Algo que no podría entenderse en una forja pensada para el futuro sería dejar de lado la orientación 2.0. Además de las herramientas colaborativas ya habituales en las forjas actuales, wikis, blogs, foros, etc... La nueva forja hará un uso intensivo de las redes sociales permitiendo la integración con las principales del mercado.

Por otra parte si hay un elemento básico detrás del mundo de las forjas es el de la comunidad. Esta nueva forja no puede concebirse sin una serie de capacidades enfocadas a la gestión e interrelación de esta comunidad. La nueva forja contará con herramientas basadas en la teoría de juegos que permitirá el establecimiento de recompensas o bounties que clasificará a los



## 11PRO001CT015

Documento de conclusiones de análisis y definición de la solución



usuarios en función de su actividad, facilitará la gestión de campañas informativas y que permitirá asimismo la integración de recursos formativos online para extender los conocimientos a la comunidad mediante herramientas LMS.

Otras funcionalidades aportadas por la forja, serán medir la calidad del código, utilización de herramientas de escritorio para el trabajo con la forja y a modo de cuadro de mandos, que la administración de los proyectos y la plataforma en sí sea realizable por personal no técnico, ...

Aunque sea entrar ya en un aspecto técnico y no funcional, se considera básico reseñar la capacidad de orquestación de la nueva forja porque esta capacidad es la que va a proveer la gran diferencia con respecto a los modelos de forjas actuales.

La forja contará con un orquestador de servicios, que basándose en un sistema de intercambio de mensajes, realice determinadas acciones. El diseño de este subsistema es simple: una serie de agentes se conectan al bus de mensajes, y se suscriben a las colas de mensajería. Cada agente queda a la espera de recepcionar el mensaje específico con las acciones que sabe interpretar. En caso de que el agente no esté disponible, y no existan más agentes capaces de interpretar y ejecutar este tipo de mensajes, el sistema de colas almacena el mismo, de modo que no se pierda ninguna acción.

Como casos de uso ejemplares sobre el funcionamiento de la forja y en concreto de la orquestación se proponen los siguientes:

1º Alta de un repositorio externo y extracción de la información del mismo.

Al dar de alta un proyecto, se selecciona la opción de utilizar un repositorio externo. Dicha orden es encolada y consumida por el correspondiente agente, que se encargará de extraer la información necesaria del repositorio remoto (probablemente a través de git clone, svn checkout o similar) a un espacio de almacenamiento temporal. Una vez terminado el clonado del proyecto remoto, el agente informará de que la acción ha sido correctamente terminada, enviando un mensaje. Éste mensaje será a su vez consumido por otro agente que examinará el repositorio a fin de recabar información y estadísticas (lenguajes, commits, ...). Al terminar de procesar el repositorio, la información resultante será consumida por un agente cuya misión será almacenar dicha información, y borrar el repositorio del espacio de almacenaje temporal.

2º Cierre de un hito y publicación de una noticia en un blog y Twitter.

Leopoldo, a través de su herramienta de escritorio, quiere indicar que se ha alcanzado el hito "Versión 3.0" del desarrollo de un proyecto y publicar una noticia en el blog del proyecto, y en Twitter. Para ello, redacta la noticia y cierra el hito en su herramienta de escritorio. La herramienta de escritorio realiza mediante API una llamada a un agente que se encarga de gestionar la información del proyecto, que indica que se ha terminado el hito "Versión 3.0" y



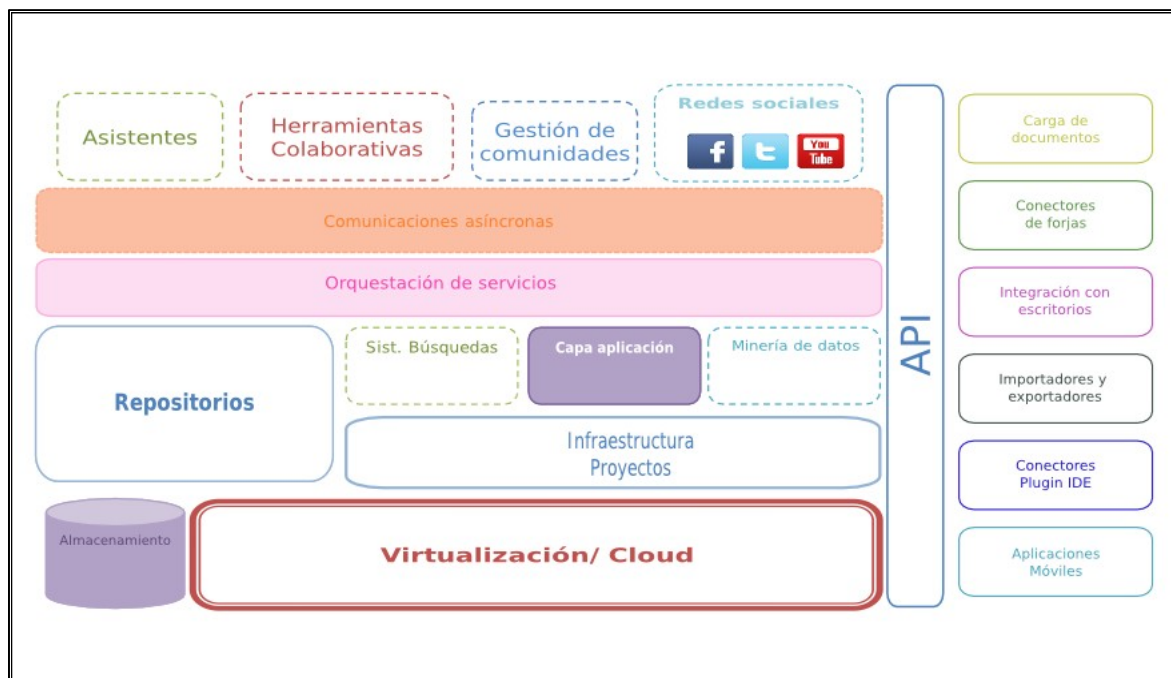
## 11PRO001CT015

Documento de conclusiones de análisis y definición de la solución

debe de cerrarse dicho hito y anunciar a todos los interesados el hecho. Una vez el agente realiza la misión de cerrar el hito en la forja, se envía un mensaje a una cola de tipo publish/subscribe, a través de la cual se identifica quién está interesado en el proyecto, y se le notifica según el método de notificación especificado por cada interesado (sea humano o programa). Uno de los notificados es un agente que se encarga de crear una entrada en el blog del proyecto (usando las credenciales adecuadas, posiblemente mediante una interfaz como python-wordpress) con los datos introducidos en la aplicación de escritorio, y en paralelo (si así se configuró), usar la API de Tweeter para postear un tweet.

### 3.2 Esquema bloques funcionales

La arquitectura de referencia de la ForjaNG ha sido descompuesta en bloques funcionales que responden a un paradigma típico de las infraestructuras que dan soporte a sistemas web de última generación.



La visión simplificada de la ForjaNG es que sirva de bus de agregación de los servicios prestados por otras forjas, así como de los prestados por la misma. A tal fin, y dado que estos servicios necesitan ser recolectados, agregados, procesados, y finalmente consumidos por los clientes, cuentan indefectiblemente con un retraso desde que la información es recolectada, hasta que ésta está finalmente disponible para el consumo. Para poder abordar la tarea de mantener una experiencia de usuario cómoda, la arquitectura se ha separado en dos entornos desacoplados, uno que interacciona con el usuario y otro en el que se ejecutan en procesos de lotes o que se estiman que pueden interferir en la experiencia de usuario.



## 11PRO001CT015

Documento de conclusiones de análisis y definición de la solución



A fin de que ambos sistemas inicialmente desacoplados puedan colaborar es necesario que estén intercomunicados de manera asíncrona. Esta capa va a permitir no sólo la independencia de los entornos, sino que además va a facilitar la integración de otros sistemas de información posteriormente, incluyéndose aquí otras forjas y llamadas al núcleo de la forja a través de APIs

Quede claro que la actual propuesta de la forja es más amplia que la requerida en el Pliego Técnico, por lo que es posible que la forja pueda ser construida con menor número de componentes, sin embargo, la arquitectura propuesta va un poco más allá de la clásica aplicación web, y propone también determinadas prácticas y tecnologías que implementan arquitecturas distribuidas a escala masiva que dan soporte a redes sociales, y que sitúan a esta forja un paso adelante del resto en el tratamiento de la información que de los usuarios y proyectos se obtienen.

Así tenemos tres bloques principales:

1. **Frontend**, que conforma la parte visible por el usuario final de la Forja de Nueva Generación, donde la criticidad se encuentra en que la experiencia de usuario sea lo más fluida posible. Según datos de SourceForge.net, casi el 90% del tráfico que reciben se centraliza únicamente en las áreas de páginas de información, repositorios y descargas de cada uno de los proyectos, lo que hace que la disponibilidad de éstas sea un factor crítico. De igual modo, la API para interactuar con los subsistemas de la forja están sometidos a unos requisitos de disponibilidad muy altos, pues ya que posibilitan el trabajo de los desarrolladores y la interacción de éstos con el resto de la forja (incluyendo a los clientes finales), cualquier tipo de incidente que degrade la prestación del servicio no es aceptable.

Bloques funcionales: asistentes, repositorios, capa de aplicación, herramientas colaborativas, gestión de comunidades, redes sociales y API

2. **Backend**, que realiza una amplia variedad de tareas, desde recolección de información de sitios web remotos, almacenamiento de datos de los usuarios, indexación de información, hasta tareas analíticas y administrativas, y que posibilita que estos datos finalmente sean puestos a disposición para el consumo.

Bloques funcionales: orquestación de servicios, infraestructura de proyectos, sistemas de búsquedas, comunicaciones asíncronas

3. **Las extensiones**, que no siendo necesarias para el funcionamiento básico de la forja, pueden ofrecer funcionalidades que aportan valor añadido a la propia forja.

Bloques funcionales: minería de datos, virtualización y/o cloud, almacenamiento



## 11PRO001CT015

Documento de conclusiones de análisis y definición de la solución



### 3.3 Descripción grupos funcionales o subsistemas

A continuación se detalla la funcionalidad que presta cada uno de los bloques, a efectos de conseguir una visión de cada uno de los subsistemas.

#### 3.3.1 Almacenamiento

La capa de almacenamiento virtualiza los recursos disponibles y los expone a los clientes como pools de almacenamiento, que serán usado por los clientes para almacenar archivos y objetos, utilizando distintos métodos de acceso dependiendo de la finalidad. Físicamente, estos recursos pueden estar dispersos entre varios servidores.

Se espera que aporte a la arquitectura:

- Un sistema de ficheros distribuido, con escalado automático y sin necesidad de reconfiguración al añadir nuevos nodos de almacenamiento.
- Un almacén de objetos el cual maneje internamente la replicación y recuperación de éstos.
- Interfaz REST al almacenamiento de objetos, compatible con S3, lo que permitiría una elevada compatibilidad y la posibilidad de escalar off-site.
- Una interfaz de dispositivos de bloques para que las máquinas virtualizadas puedan aprovecharse de forma transparente de todas las ventajas del sistema de ficheros distribuido.

#### 3.3.2 Virtualización-Cloud

Este bloque permite la abstracción de las instancias del sistema operativo que ejecutan procesos relacionados con la forja de la plataforma hardware donde se ejecuta. Dado que la concepción de la forja es eminentemente distribuida, una instancia del SO que presta determinados servicios debe de ser fácilmente migrables, replicable y escalable.

Se espera que aporte a la arquitectura:

- Enfoque modular, que permita la sustitución de alguno de los componentes por sistemas diseñados a medida
- Sistema distribuido y flexible, donde cada componente puede ejecutarse estratégicamente cerca de los recursos que usa o necesita, de manera que mejore la escalabilidad de la plataforma. De este modo, se facilitará sobremanera la administración, pudiendo ser casi completamente automatizada en lo referente a la provisión de servicios.
- Compatibilidad con otras plataformas, de modo que sea posible desligar parte de



la plataforma off-site o simplemente facilitar la migración de una plataforma a otra, incluyendo, la independencia del hipervisor, y posibilitando la creación de nubes privadas, públicas o mixtas.

### 3.3.3 Repositorios

En las forjas clásicas, el sistema de control de versiones es el componente clave, ya que posibilita la colaboración en el desarrollo de software, y sobre el que el resto de la forja órbita.

Un sistema de control de versiones debe proporcionar:

- Mecanismo de almacenamiento de los elementos que deba gestionar (ej. archivos de texto, imágenes, documentación...)
- Posibilidad de realizar cambios sobre los elementos almacenados (ej. modificaciones parciales, añadir, borrar, renombrar o mover elementos)
- Registro histórico de las acciones realizadas con cada elemento o conjunto de elementos (normalmente pudiendo volver o extraer un estado anterior del producto)
- Generación de informes con los cambios introducidos entre dos versiones, informes de estado, marcado con nombre identificativo de la versión de un conjunto de ficheros, etcétera.
- Exportación e importación de otros repositorios de tecnologías distintas.
- Integración con otras herramientas, tanto traductores de repositorios como aplicaciones destinadas a la notificación de cambios, seguimiento de errores, ...

### 3.3.4 Infraestructuras de proyectos

El bloque funcional provee toda la infraestructura necesaria para que un usuario o proyecto utilice la forja. Por ejemplo, cuando un usuario crea un proyecto, a parte del repositorio, se le dan las opciones de elegir la licencia, si tiene que provisionarse un blog local, si se va a crear un wiki o si por el contrario se va a utilizar uno albergado por un tercero.

Este bloque recoge la información suministrada en la capa de aplicación o mediante API y ejecuta las acciones necesarias para que el proyecto tenga a su disposición todas las herramientas que ha seleccionado. Está conformado por agentes que esperan un mensaje, y al recibirlo, ejecutan la acción, y está en relación casi directa con la capa de orquestación de servicios

### 3.3.5 Búsquedas

El bloque funcional responde a las características de un buscador tipo Enterprise, término utilizado para describir software que busca información federada en otros



entornos. A diferencia de los buscadores web, éstos indexan datos y documentos de una amplia variedad de fuentes, como sistemas de ficheros, DMS, blogs, y cuentan con un control de acceso de usuarios dependiente de una política de seguridad

El sistemas de búsquedas debe de seguir un flujo de trabajo bien definido:

- Ingestión de contenido, siguiendo un modelo push-pull, donde bien se puede inyectar el nuevo contenido a procesar mediante una API (crítico para búsquedas en tiempo real), o bien dicho contenido se extrae de las fuentes mediante arañas u otros medios.
- Procesamiento y análisis, pues el contenido extraído de diferentes fuentes puede presentarse en distintos formatos ha de transformarse a fin de que puedan ser interpretados por el buscador.
- Indexación del resultado, que estará optimizado para realizar rápidas búsquedas , aportando también información como ranking o frecuencia del término
- Capacidad de realización de consultas complejas, mediante interfaz web o API y su comparación contra el índice y presentación de las mismas.

### 3.3.6 Capa de aplicación

Solución basada en un framework de desarrollo web en SFA que permita un rápido desarrollo de aplicaciones, que haga uso extensivo de la reutilización de código, haga uso de plugins para extender las funcionalidades, facilite la construcción de aplicaciones a medida a usuarios no expertos y disponga de una amplia variedad de conectividad con otros componentes web, basado en el paradigma Modelo-Vista-Controlador. También debe de contar con una aplicación administrativa que permita la creación, actualización y eliminación de objetos de contenido, llevando un registro de todas las acciones realizadas sobre cada uno, y proporciona una interfaz para administrar los usuarios y los grupos de usuarios (incluyendo una asignación detallada de permisos). Por supuesto, estas tareas también deben poder ser automatizadas.

El framework a utilizar debe incluir

- Un núcleo que se ajuste al paradigma Modelo-Vista-Controlador (MVC)
- Capa de abstracción para el acceso a datos
- Un sistema de templates, que permita la personalización de la presentación.
- Un sistema de formularios con serialización y validación.
- Poder actuar sobre todo el flujo de proceso de peticiones y extenderse mediante código



## 11PRO001CT015

Documento de conclusiones de análisis y definición de la solución



- Un sistema de serialización que pueda producir e interpretar representaciones en formatos JSON y XML.
- Un completo juego de herramientas para la creación de APIs REST.
- Una interfaz de administración
- Herramientas para generar feeds de sindicación de noticias en RSS y ATOM
- Sistema de comentarios flexible y extensible
- La capacidad de ejecutar múltiples sitios web sobre una única instalación del framework
- Herramientas para generar mapas del sitio web dinámicamente
- Protección contra vulnerabilidades XSS
- Soporte para microformatos y web semántica. Soporte para ActivityStreams.
- Herramientas para la revisión de código entre pares.
- Disposición (no sólo visual) inicial estándar de proyectos para mantener la consistencia y fácil despliegue. Una vez desplegado será posible personalizarlo usando métodos como plantillas, CSS, páginas HTML estáticas
- Proyectos tipo que pueden ser usados como la base de cualquier proyecto
- Plantillas por defecto para facilitar el rápido prototipado
- Widgets que pueden instalarse en cualquier página.
- Un sistema incorporado de "vistas genéricas" que ahorra tener que escribir la lógica de ciertas tareas comunes.
- Un sistema extensible de plantillas basado en etiquetas, con herencia de plantillas.
- Un dispensador de URLs basado en expresiones regulares.
- Soporte de internacionalización i18n y l10n, incluyendo traducciones incorporadas de la interfaz de administración. Las páginas deben de poder soportar varios idiomas y las peculiaridades relativas a monedas, fecha y hora. Además, deberá de poder validar que los datos introducidos en los formularios están localizados con el idioma que en que se presenta la página.



## 11PRO001CT015

Documento de conclusiones de análisis y definición de la solución



- Documentación incorporada accesible a través de la aplicación administrativa (incluyendo documentación generada automáticamente de los modelos y las bibliotecas de plantillas añadidas por las aplicaciones).
- Dado que la mayoría de las operaciones son calculadas dinámicamente (cuando se le realiza una petición), es necesario el uso extensivo de caching de aquellas operaciones que resultan costosas en términos computacionales, de modo que no sea necesario volver a realizar los cálculos la próxima vez. También debe de contar con soporte para múltiples métodos de cacheo
- Extensibilidad: mediante el uso de plugins deberá de poder incluir funcionalidades no incluidas en el core de la aplicación
- Semántica: uso de microformatos dentro del (X)HTML, que permitirá agregar más información y significado a los datos que luego puede ser utilizado por aplicaciones como por ejemplo buscadores o agregadores de contenidos.
- Sistema de autenticación: Debe de permitir la autenticación contra bases de datos, directorios LDAP, OpenID, Oauth y un largo abanico de fuentes de autenticación. Al menos debe de permitirse la gestión de :
  - Usuarios
  - Permisos que definan si un usuario puede realizar determinadas tareas
  - Grupos de usuarios, a fin de aplicar etiquetas y permisos a más de un usuario.
  - Otros permisos
- Debe de poseer un sistema de logging de eventos en las aplicaciones, a fin de que se recopile información propia de la aplicación (tal como rendimiento, errores, avisos) que puede ser posteriormente analizada a fin de obtener distintas métricas de la aplicación.
- Debe de poder manejar de manera simple y gestionable por el usuario, las redirecciones hacia páginas externas, debiéndose identificar de forma clara que el usuario va a proceder a abandonar la forja y acceder a un contenido dependiente de un tercero.
- Proveer soporte completo para sesiones, tanto anónimas como autenticadas y permitir el almacenamiento y recuperación de datos arbitrarios, de forma



## 11PRO001CT015

Documento de conclusiones de análisis y definición de la solución



individual para cada usuario, desde el servidor.

- Señalización, para permitir a ciertas entidades notificar a otras entidades cuando una acción a tenido lugar (alta de usuario, creación de repositorios, etcétera). Es especialmente útil cuando varias partes del código están interesadas en los mismos eventos, y al estar contenidas dentro de la propia aplicación, no es necesario el uso de un sistema de intercambios de mensajes completo.
- Deberá de poder crear dinámicamente el mapa del sitio web, archivo XML que describe a los indexadores de los buscadores web qué ha cambiado y cuánta importancia tienen determinadas páginas en relación con otras del sitio web, lo que optimiza la indexación de la web y el posicionamiento en los buscadores
- Deberá contar con un generador de feeds de alto nivel que facilite la creación de feeds, tanto RSS como Atom. Esto permitirá que tanto la misma forja como otras entidades puedan suscribirse a éstos y puede contener entradas, que pueden ser encabezados, artículos completos, resúmenes, y/o enlaces al contenido de un sitio web, todo listo para ser tratado adecuadamente por programas externos, como lectores de feeds o programas de extracción de contenidos.
- Dado que la interacción entre usuarios es un área crítica en la forja, el framework debe de contar con las herramientas web 2.0 necesarias, tales como comentarios, botones de "Me gusta" o "No me gusta", chat, Twitter, así como integración con los servicios externos mediante API
- Debe de permitir la fácil creación de asistentes, incluyendo la vista previa de estos. El workflow debe de ser similar a:
  - El usuario accede a la primera página del asistente, rellena el formulario y lo envía.
  - El servidor valida los datos. Si son inválidos, el formulario es mostrado de nuevo, con un mensaje de error asociado. En el caso de ser válido, el servidor almacena el dato (preferiblemente cifrado) en un campo oculto del formulario.
  - Los puntos 1 y 2 se repiten sucesivamente para cada formulario del asistente. Paralelamente se va mostrando información relativa a los datos que se han introducido en el asistente, a fin de proporcionar recomendaciones, proyectos similares, o simplemente consejos a fin de que la información quede completada de la forma más óptima para que el proyecto cuente con la información necesaria



## 11PRO001CT015

Documento de conclusiones de análisis y definición de la solución



- Si se han completado todos los formularios presentados, los datos son validados y se presentan al usuario una vista previa de los datos introducidos.
- Una vez terminado el proceso, los datos son validados y el asistente procesa los datos, completándose así el flujo de trabajo.
- Debe de poder manejar datos geospaciales, a fin de poder trabajar con éstos y mostrar información referente a la geolocalización, tal como usuarios que compartan geolocalizaciones cercanas.

### 3.3.7 Minería de datos

Dado que la cantidad de información con la que cuenta la plataforma es potencialmente elevada, cierta información tiene que estar en el frontend (principalmente métricas y cuadros de mando), y el histórico de dicha información debe de almacenarse en un sistema que permita bucear en la información de manera ágil, así como descubrir información que no es evidente a primera vista.

El sistema de minería de datos debe proporcionar:

- Aprendizaje en la asociación de reglas, buscando relaciones entre dos ítem.
- Agrupación, para descubrir ítem que son de alguna manera similares, incluso sin conocer la estructura de los datos.
- Clasificación, aplicando el conocimiento que se tiene para identificar y asociar contenido nuevo con contenido relacionado
- Regresión, intentando encontrar una función que modele los datos con el menor número de errores posibles.

### 3.3.8 Orquestación de servicios

Los servicios de la forja deben de ser automatizados al ciento por ciento, dado que las tareas que requieren intervención humana entorpecen la experiencia de usuario. La orquestación de servicios aúna la monitorización de la plataforma con el mantenimiento de la infraestructura.

Un ejemplo: un usuario crea un nuevo proyecto, al cual asocia un blog local a la forja, la acción se encola y termina llegando al agente que despliega blogs. Este agente consulta el estado del servidor dedicado a los blogs, y se percata de que se ha llegado a la capacidad máxima de blogs que puede albergar una instancia, definida en una base de datos global. El agente, en vez de fallar la acción, notifica a otro agente que tiene que provisionar una nueva instancia del servidores dedicado a albergar blogs, el cual provisiona de manera pertinente la instancia del S.O. configurado mediante un sistema de control de la configuración. Una vez terminada la provisión de la máquina, se notifica



al agente que puede proceder a crear el blog en tal máquina.

### 3.3.9 Comunicaciones asíncronas

Este bloque actúa como capa de comunicaciones entre los procesos participantes de la forja, usando el concepto de Message Oriented Middleware, permite a los módulos de una aplicación encontrarse distribuidos sobre plataformas homogéneas y crear una capa de comunicaciones que facilite la creación de programas distribuidos abstrayendo la complejidad de la intercomunicación de los componentes.

### 3.3.10 Asistentes

Este bloque funcional, a pesar de estar soportado técnicamente por la capa de aplicación, tiene suficiente entidad como para ser tratado específicamente. Un asistente es un tipo de interfaz de usuarios que presenta a un usuario una secuencia de cuadros de diálogo que guían al mismo a través de una serie de pasos bien definidos, haciendo más amigables al usuario tareas que son complejas o con las que no se encuentra familiarizado. A diferencia de los asistentes, un sistema experto guía al usuario proporcionándole información relacionada con la temática del asistente. Con los sistemas expertos se busca una mejor calidad y rapidez en las respuestas, dando así lugar a una mejora de la productividad y calidad de la información introducida por el usuario.

Estos asistentes deberán de estar conformados por:

- Base de conocimientos: Contiene conocimiento modelado extraído del diálogo con un experto o sistema automatizado de procesamiento de información, como por ejemplo un sistema de recomendación, o un catálogo de licencias albergado en una wiki.
- Interfaz de usuario: es la interacción entre el sistema experto y el usuario, y cuando sea posible, utilizará lenguaje natural a fin de facilitar la tarea al usuario.
- Los asistentes podrán utilizar las capacidades de búsqueda en tiempo real proporcionadas por la propia forja o sistemas externos a fin de enriquecer el contexto de la información con metadatos relativos al tiempo.
- Se deberán de crear al menos los siguientes asistentes, aunque se aboga por el uso intensivo de éstos a fin de facilitar la interacción de los usuarios no técnicos con la forja:
  - Alta de proyecto: al dar de alta un proyecto o usuario, se completarán una serie de datos básicos (como tecnologías, objetivos, ...) que serán enriquecidos con propuestas de proyectos relacionados o similares, usuarios con experiencia en las tecnologías determinadas, usuarios que recomiendan a esta persona para determinadas áreas tecnológicas, asistir a que la información del proyecto sea completada en idiomas relacionados,



## 11PRO001CT015

Documento de conclusiones de análisis y definición de la solución



recomendará *best practices* relativas a la gestión de proyectos como uso de sistemas de integración continua, ...

- Importación y exportación. Este asistente guiará al usuario a realizar tanto la exportación del proyecto albergado en la forja, como en la importación de proyectos, sean estos internos, mixtos o externos. En el caso de que se dependan de terceros, el asistente proporcionará información de qué servicios externos soporta la extracción de información de forma automatizada. Por ejemplo, conectores para blogs en Wordpress, sistemas de bug tracking como Mantis, diferentes tipos de wikis, siempre y cuando estos sistemas de terceros provean un método estándar para extraer dicha información.
- Reconfiguración del proyecto. Este asistente permitirá la reconfiguración del proyecto o información relacionada con el usuario, incluyendo capacidades técnicas, información de difusión, gestión de comunidades, adición de herramientas al proceso metodológico de desarrollo.
- Colaboración y búsqueda de perfiles y capacidades. Muy enfocado a la interacción entre usuarios, este asistente permitirá encontrar posibles colaboradores para una tarea determinada, como por ejemplo, ayuda con código de una determinada tecnología, buscar traductores para un idioma y demás acciones que tienden a promover la colaboración entre usuarios.

### 3.3.11 Herramientas colaborativas

Estas herramientas facilitarán el desarrollo del trabajo en grupo, proveyendo a los usuarios de canales de comunicación (blogs, listas de correo, chat, video-conferencia), herramientas de edición colaborativa (editores), repositorios de conocimiento (wikis, FAQs), y otros.

Se espera que este bloque funcional ayude a

- Organizar la información que tiene el usuario relacionada con uno o varios proyectos, facilitándole el uso y búsqueda de archivos, calendarios o contactos de actividades de estos proyectos.
- Compartir la información de forma segura y simple con proveedores, colaboradores o clientes, con disponibilidad de la información y los archivos relacionados de forma inmediata y desde cualquier lugar.
- Colaborar con proveedores, colaboradores o clientes a través de herramientas que ofrezca el sistema, como las pizarras virtuales, sistemas de comentarios, etc.
- Controlar accesos y permisos de uso y vista de los elementos relacionados a los proyectos a través de sistemas de permisos y bitácoras de actividades.



Además, y casi entrando en el área de gestión de comunidades que se verá a continuación, potenciará la construcción cooperativa de conocimientos, haciendo uso de herramientas LMS (como Moodle) a fin de expandir el conocimiento y establecer métodos para la futura certificación en tecnologías a los usuarios.

### 3.3.12 Gestión de comunidades

Estas herramientas facilitarán la difusión y gestión del conocimiento, a fin de animar a crear una comunidad activa e implicada en la consecución de los objetivos del proyecto, así como encontrar potenciales colaboradores en ámbitos determinados, sean éstos tecnológicos o no.

Inicialmente se clasifican las herramientas según su ámbito.

1. Herramientas de transmisión inmediata: Son herramientas que permiten transmitir el conocimiento explícito de forma fácil al conjunto de miembros de la forja.
2. Herramientas y servicios de gestión del conocimiento interno: Son aquellos componentes que gestionan, analizan, buscan y distribuyen información. Por ejemplo el sistema de recomendación.
3. Herramientas y servicios de gestión del conocimiento externo: son componentes que gestionan, analizan, buscan y distribuyen, pero en este caso también hay que añadir que localizan y extraen, dado que su misión principal es la localización y extracción de información relacionada con el proyecto pero que está en el exterior de ésta (principalmente en otras forjas) y que por lo tanto en algunas ocasiones puede ser ajena a esta y no tener conocimiento de su existencia.

Otras estrategias de gestión de la comunidad incluyen:

- Recompensas (bounties) para motivar la colaboración en determinadas áreas
- Podcast: contar historias como medio de transferir conocimiento
- Casos de uso como medio de efectuar una demostración de la aplicabilidad del proyecto
- Directorio de Expertos por un campo específico y que actúan como primera referencia sobre con quién hablar sobre un tema específico.
- Organizar talleres y sesiones formativas
- Software social, como redes sociales, wikis.

### 3.3.13 Redes sociales

Las redes sociales son estructuras sociales compuestas de grupos de personas, las



## 11PRO001CT015

Documento de conclusiones de análisis y definición de la solución



cuales están conectadas por uno o varios tipos de relaciones, tales como amistad, parentesco, intereses comunes o que comparten conocimientos. Mediante el análisis de redes sociales, se estudia esta estructura social aplicando la Teoría de Grafos e identificando las entidades como "nodos" o "vértices" y las relaciones como "enlaces" o "aristas". Esta teoría puede ser utilizada para medir el el valor que un individuo obtiene de los recursos accesibles a través de su red social.

En estas comunidades, un número inicial de participantes envían mensajes a miembros de su propia red social invitándoles a unirse al sitio. Los nuevos participantes repiten el proceso, creciendo el número total de miembros y los enlaces de la red. Los sitios ofrecen características como actualización automática de la libreta de direcciones, perfiles visibles, la capacidad de crear nuevos enlaces mediante *servicios de presentación* y otras maneras de conexión social en línea con otros proveedores de sitios sociales.

La forja debe de contener las herramientas informáticas para potenciar la eficacia de las redes sociales:

- Comunicación, que nos ayudan a poner en común conocimientos.
- Comunidad, nos ayudan a encontrar e integrar comunidades.
- Cooperación, nos ayudan a hacer cosas juntos

Como se puede observar, tanto las redes sociales, la gestión de la comunidad, como las herramientas colaborativas tiene una intersección importante, y es precisamente esa el área que esta forja intenta potenciar.

### 3.3.14 API

La forja deberá de exponer los servicios internos mediante una API REST que permita la integración con otras herramientas, como herramientas de seguimiento de proyecto para escritorio, aplicaciones para dispositivos móviles. Esta API Proveerá acceso a recursos (entidades de datos) vía rutas URI, sobre una capa de transporte HTTP de cara a los clientes, y preferiblemente AMQP a nivel interno. Por defecto el formato de respuesta es JSON, aunque también estará disponible XML .

Dado que la API REST está basada en estándares libres, será posible la utilización de esta API sobre un amplio abanico de lenguajes.

Se deberán de proveer, al menos, los siguientes recursos, usando los verbos típicos de HTTP conocidos como CRUD (Create, Read, Update, Delete):

**Changesets:** da acceso a la lista de cambios de los repositorios.

**Emails:** da acceso acceso a la creación, modificación y eliminación de cuentas de correo asociadas con las cuentas.



## 11PRO001CT015

Documento de conclusiones de análisis y definición de la solución



**Eventos:** da acceso a la información acerca de eventos de usuarios/proyectos y repositorios

**Seguidores:** provee la funcionalidad de listar seguidores de un usuario/proyecto, repositorio o bug

**Grupos:** da acceso a listar información de grupos, crear nuevos, actualizar su participantes o borrarlos.

**Permisos de grupo:** da acceso a manipular los permisos concedidos a un grupo.

**Invitaciones:** proporciona una manera de permitir a un determinado usuario acceso en modo lectura, escritura o de administrador a un repositorio, vía correo. Si el invitado accede a la URL adjunta en el correo, automáticamente se le asignan los permisos recogidos en la invitación.

**Bugs:** permite la manipulación de los bugs recogidos en el sistema de seguimiento de errores.

**Permisos:** permite la manipulación de los permisos de los repositorios, permitiendo asignarles permisos de lectura, escritura o administración a usuarios específicos.

**Repositorios:** permite la manipulación de creación, modificación y eliminación de repositorios

**Servicios:** permite la adición, eliminación o modificación de agentes conectores en los repositorios.

**Fuentes:** permitirá la navegación remota por las fuentes del repositorio.

**Claves SSH:** permitirá la adición, modificación, eliminación y consulta de claves SSH asociadas a una cuenta de usuario.

**Usuarios:** permitirá la adición, modificación, eliminación y consulta de cuentas de usuarios

**Wiki:** permitirá la adición, modificación, eliminación y consulta de páginas en las wikis

**Blog:** permitirá la adición, modificación, eliminación y consulta de páginas en los blogs

Además, ante la reciente adopción por el IETF del RFC 5789 en relación al nuevo verbo HTTP PATCH, el cual permite la actualización una parte de un recurso especificado de una manera predecible y de modo transaccional, se evaluará la posibilidad de diseñar la API para implementar el manejo dicho verbo.



## 11PRO001CT015

Documento de conclusiones de análisis y definición de la solución



### 3.3.15 Conectores

Los conectores son programas externos a la ForjaNG que interactúa con la API de la aplicación de forma que pueda ser utilizado por terceros de manera simple. Un conector se comunica mediante la API con la aplicación o con un agente. Una petición que procede de un conector no es más que una llamada REST a la API de la aplicación.

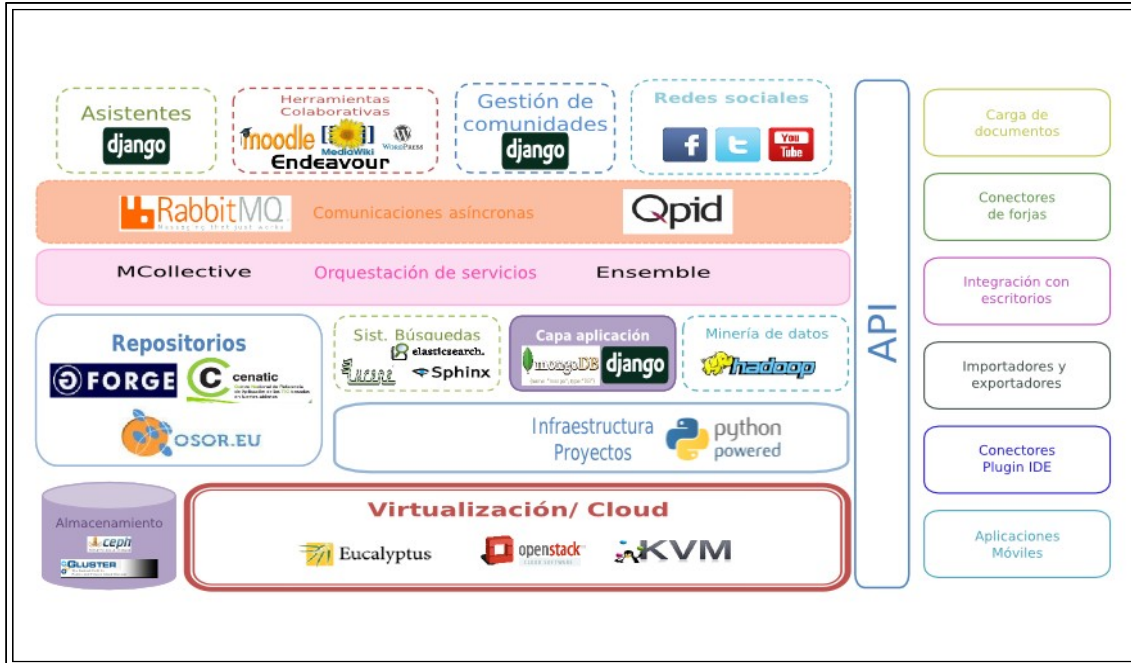
## 4 ESTUDIO PREVIO DE ALTERNATIVAS DE SOLUCIÓN

A continuación se detalla la arquitectura técnica, mostrando algunas alternativas evaluadas para cubrir cada bloque funcional.

Estos componentes **deberán ser validados y verificados durante el proyecto** a fin de adecuarse a los requerimientos identificados finalmente como viables para la solución ForjaNG.

### 4.1 Diagrama de arquitectura.

En el diagrama inferior se pueden observar cómo algunos bloques funcionales sí tienen una propuesta tecnológica. Esto es debido a que siguiendo la filosofía del SFA, la reutilización de componentes especializados usando estándares abiertos, nos permite el aprovechamiento de la tecnología y experiencia en la materia, pudiendo entonces concentrarnos en los componentes que el Análisis de Requerimientos de la Forja de Nueva Generación identificaba que no estaban cubiertos por ninguna forja. Siguiendo las recomendaciones indicadas anteriormente respecto al framework de desarrollo web, se ha propuesto la utilización de Django para acometer el desarrollo de estos componentes, incluyendo la API.



Esta arquitectura inicial no contempla el licenciamiento de los componentes, que habrá de examinar con detalle en busca de posibles incompatibilidades durante la realización de la matriz de trazabilidad.

## 4.2 Propuesta de tecnologías a incluir en el estudio

Para aquellos bloques funcionales que cuenten con productos ya identificados que puedan ser usados como base para sustentar la FNG, se proponen las siguientes tecnologías para ser tenidas en cuenta en la evaluación.

### 4.2.1 Almacenamiento

**GlusterFS:** un sistema de archivos NAS desarrollada por Gluster. Agrega varios servidores de almacenamiento a través de Ethernet o Infiniband de interconexión RDMA en un gran sistema paralelo de archivos de red. GlusterFS se basa en un diseño de espacio de usuario (no en el kernel) apilable sin comprometer el rendimiento. Se ha encontrado una variedad de aplicaciones que incluyen computación en la nube, las ciencias biomédicas y de almacenamiento de archivos. GlusterFS es software libre, licenciado bajo GNU AGPL v3 licencia.

**Ceph:** constituye una red de almacenamiento y un sistema de ficheros distribuidos, diseñado para proveer un excelente rendimiento, disponibilidad y escalabilidad. Ceph se basa en un almacén de objetos fiable y escalable distribuido, con un grupo de gestión de metadatos distribuidos en capas superiores para proporcionar un sistema de archivos distribuido con la semántica POSIX. Hay una variedad de maneras de interactuar con el



## 11PRO001CT015

Documento de conclusiones de análisis y definición de la solución



sistema: sistema de ficheros distribuidos, objetos de almacenamiento, mediante API compatible con Amazon S3 y dispositivos de bloque. Ceph es software libre, licenciado bajo la licencia LGPL

### 4.2.2 Virtualización - cloud

**Eucalyptus:** es una plataforma open source para la implementación de computación en nube privada en clústers de ordenadores. **Eucalyptus** es compatible con Amazon Web Services (Amazon EC2 y S3). Puede usar gran variedad de tecnologías de virtualización de hardware incluyendo hipervisores VMware, Xen y KVM para implementar las abstracciones de nube que soporta. Actualmente posee una interfaz orientada al usuario que es compatible con los servicios pero la plataforma está modularizada para poder utilizar un conjunto de interfaces diferentes simultáneamente. Hay 2 ediciones básicas: una propietaria, y otra de código abierto, con licencias GPL3 y BSD

**Openstack:** es un proyecto de cloud computing que intenta proporcionar todas las herramientas necesarias para la creación de servicios IaaS, creado por Rackspace Cloud y NASA. Está formado OpenStack Compute (Nova) como controlador de nube IaaS. Está escrita en Python, usando los frameworks Eventlet and Twisted, AMQP como protocolo de mensajería, y SQLAlchemy acceso a datos. OpenStack Object Store (Swift) es un sistema de almacenamiento masivo escalable diseñado para soluciones cloud. Finalmente Glance es el servicio de gestión de imágenes de máquinas virtuales, con soporte para Open Virtualization Format (OVF) support. Es software libre bajo la licencia Apache.

**KVM:** KVM permite ejecutar máquinas virtuales utilizando imágenes de disco que contienen sistemas operativos sin modificar. Cada máquina virtual tiene su propio hardware virtualizado: una tarjeta de red, discos duros, tarjeta gráfica, etc. Licenciado bajo GPL y LGPL

### 4.2.3 Repositorios

**GIT:** es un software de control de versiones diseñado por Linus Torvalds, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando estas tienen un gran número de archivos de código fuente. Al principio, Git se pensó como un motor de bajo nivel sobre el cual otros pudieran escribir la interfaz de usuario o front end como Cogito. Sin embargo, Git se ha convertido desde entonces en un sistema de control de versiones con funcionalidad plena. Licenciado bajo GPLv2

**Bazaar:** es un sistema de control de versiones distribuido patrocinado por Canonical Ltd., diseñado para facilitar la contribución en proyectos de software libre y open source. Bazaar puede ser usado por un usuario único trabajando en múltiples ramas de un contenido local, o por un equipo colaborando a través de la red. Bazaar está escrito en lenguaje de programación Python, es software libre licenciado bajo GPLv2

**Mercurial:** es un sistema de control de versiones multiplataforma, para desarrolladores de software. Está implementado principalmente haciendo uso del lenguaje de



programación Python, pero incluye una implementación binaria de diff escrita en C. Las principales metas de desarrollo de Mercurial incluyen un gran rendimiento y escalabilidad; desarrollo completamente distribuido, sin necesidad de un servidor; gestión robusta de archivos tanto de texto como binarios; y capacidades avanzadas de ramificación e integración, todo ello manteniendo sencillez conceptual. El código fuente se encuentra disponible bajo los términos de la licencia GNU GPL versión 2.

**Subversion:** es un sistema de control de versiones centralizado diseñado específicamente para reemplazar al popular y antiguo CVS. Es software libre bajo una licencia de tipo Apache/BSD y se le conoce también como *svn* por ser el nombre de la herramienta utilizada en la línea de órdenes.

#### 4.2.4 Buscador

**Lucene/Nutch:** es un API de código abierto para recuperación de información, originalmente implementada en Java. Está apoyado por el Apache Software Foundation y se distribuye bajo la Apache Software License. Lucene tiene versiones para otros lenguajes incluyendo Delphi, Perl, C#, C++, Python, Ruby y PHP.

Es útil para cualquier aplicación que requiera indexado y búsqueda a texto completo. Lucene ha sido ampliamente usado por su utilidad en la implementación de motores de búsquedas. Nutch es un software que, sobre la base aportada por Lucene, integra todo lo que hace falta para completar un motor de búsqueda de páginas web.

**ElasticSearch:** Se trata de un software de código abierto (Apache 2), que implementa un motor de búsqueda integrado sobre Lucene, capaz de indexar los datos simplemente usando JSON a través de HTTP, Soporta búsquedas en tiempo real, permite multitenancia y su diseño es distribuido. Es actualmente usado en sitios como Mozilla, StumbleUpon o Sony.

**Solr:** Sus características principales incluyen potentes búsqueda de texto completo, destacando hit, búsqueda por facetes, agrupamiento dinámico, la integración de bases de datos, ficheros de documentos (por ejemplo, Word, PDF), manipulación y búsqueda geoespacial. Solr es altamente escalable, proporcionando búsquedas distribuidas y replicación de índices, y les proporciona búsquedas y funciones de navegación de muchos de los sitios más grandes del mundo de Internet. Solr está escrito en Java y se ejecuta como un servidor de búsqueda dentro de un contenedor de servlets como Tomcat. Solr utiliza la biblioteca de Java Lucene búsqueda en su base para la indexación de texto completo y búsqueda. Utiliza tanto REST como HTTP / XML y JSON APIs que hacen que sea fácil de usar desde cualquier lenguaje de programación. Solr permite que se adapten a casi cualquier tipo de aplicación sin codificación Java, y tiene una arquitectura de plugin que permite una amplia personalización más avanzada cuando es necesario. Licencia Apache 2

**Sphinx:** es un motor de búsquedas de código, diseñado desde cero con el desempeño, la relevancia (búsqueda de calidad), integración y simplicidad en mente. Está escrito en



C + + y funciona en Linux. Permite índices por lotes y los datos son almacenado una base de datos SQL, almacenamiento NoSQL, o simplemente los archivos de forma rápida y sencilla. Buscar a través de SphinxAPI es simple, y permite consultas a través de SphinxQL, con consultas de búsqueda expresadas en SQL. Es software libre disponible sobre licencia GPLv2

#### 4.2.5 Minería de datos

**Hadoop:** es un proyecto de SFA, licencia Apache, que provee una plataforma para combinar y analizar datos de un tamaño descomunal. Técnicamente, consiste en dos servicios clave: almacenamiento escalable de datos, mediante HDFS, y un sistema de procesamiento paralelo de alto rendimiento mediante técnicas MapReduce . Alrededor de éstos, se han creado herramientas que aprovechan las características de estos servicios para proveer una serie de herramientas que permitan trabajar con enormes cantidades de información, como por ejemplo, una base de datos escalable (Hbase), un sistema de coordinación de sistemas distribuidos (Zookeeper), un sistema de aprendizaje y minería de datos (Mahout). Al ser un conglomerado de proyectos auspiciados por la fundación Apache, está licenciado bajo Apache 2.0

**Disco:** una plataforma de análisis para grandes volúmenes de datos, que soporta computación paralela mediante MapReduce, que ofrece un framework ligero para el procesamiento de datos distribuidos. Disco ha sido utilizado con éxito en Nokia y en otros lugares para una variedad de propósitos: analizar, reformatear, análisis de registros, agrupación, modelos probabilísticos, minería de datos, la indexación de texto completo, y el aprendizaje máquina. Los usuarios suelen escribir trabajos en Python, por lo que es posible expresar algoritmos más complejos en tan sólo unas decenas de líneas de código. La licencia es <https://github.com/tuulos/disco/blob/master/LICENSE>

#### 4.2.6 Infraestructura de proyectos

La infraestructura de proyectos, por lo particular de las funcionalidades que van a prestar, va a tener que diseñarse a medida que se vayan definiendo los servicios de la FNG. Se recomienda el lenguaje de programación Python para la creación de estos agentes.

También se puede estudiar como alternativas el uso de Node.js, un entorno de JavaScript del lado del servidor que utiliza un modelo asíncrono orientado a eventos. Esto permite a Node.js obtener un rendimiento excelente sobre la base de las arquitecturas típicas de aplicaciones web.

#### 4.2.7 Orquestación de servicios

**Mcollective:** es un proyecto derivado de Puppet, que provee un framework para construir sistemas de orquestación y ejecución de tareas de forma paralela. Utiliza STOMP como modelo de intercambio de mensajes, pero también tiene soporte para AMQP y otros protocolos. Las acciones se pueden definir en distintos lenguajes. Utiliza filosofías modernas como descubrimiento en tiempo real de recursos usando metadatos, en vez de nombres DNS. Al derivarse de Puppet obtenemos también un completo



## 11PRO001CT015

Documento de conclusiones de análisis y definición de la solución



sistema de configuración automática, que contiene herramientas para la completa automatización de la plataforma, y permite la gestión remota vía API REST. Está licenciado bajo Apache 2.0

**Ensemble:** herramienta de despliegue de servicios y orquestación. Con Ensemble, diferentes autores son capaces de crear servicios independientemente, y hacer estos servicios comunicarse a través de un protocolo de configuraciones simple. Entonces, el usuario puede coger el producto de ambos autores y cómodamente desplegar estos servicios en un entorno, de la misma forma que la gente es capaz de instalar una red de dependencias de paquetes con un simple comando vía APT. Licencia GNU Affero GPLv3

### 4.2.8 Comunicaciones asíncronas

**RabbitMQ:** es la implementación de referencia del protocolo AMQP, un sistema de mensajería que ofrece alta disponibilidad, escalabilidad y latencias mínimas. Soporta además los protocolos SMTP, STOMP, HTTP y XMPP, que permiten usarlo como sistema de mensajería web ligero. Es software libre bajo licencia Mozilla Public License (MPL1.1)

**Celery:** es una cola de tareas y trabajos basada en un sistema distribuido de intercambio de mensajería, que se centra en la operativa en tiempo real, pero también soporta tareas programadas. Las unidades de ejecución, llamadas tareas, son ejecutadas concurrentemente en uno o más servidores, y de forma asíncrona o síncrona. Está desarrollado en python, pero permite la integración con otros lenguajes vía una interface HTTP que expone datos en formato JSON. Es software libre bajo licencia New BSD

**Qpid:** es un sistema multi-plataforma de mensajería de empresa que implementa el protocolo de AMQP, proporcionando en C++ y Java, junto con los clientes de C++, Java JMS, .Net, Python y Ruby. Implementa la especificación más reciente AMQP, proporcionando la gestión de transacciones, colas, distribución, seguridad, gestión, agrupación, federación y soporte heterogéneo de múltiples plataformas. Licenciado Apache 2.0

### 4.2.9 Herramientas colaborativas

La amplia variedad de herramientas disponibles (cada una con sus ventajas e inconvenientes) para cubrir este bloque funcional, hace inviable realizar una evaluación de cada uno de los productos. Por ello, se expone un ejemplo de cada tipo de herramienta, con el objetivo de en próximas sesiones identificar cuáles son las herramientas más utilizadas por los stakeholders y realizar la posterior evaluación, tanto para posible integración como para posible exportación de contenidos. Así mismo, algunas de estas herramientas, por su simplicidad, serán implementadas dentro de la capa de aplicación y serán accesible vía API. En este bloque encontraremos herramientas como:

- Blog MultiSite
- Listas de correo



## 11PRO001CT015

Documento de conclusiones de análisis y definición de la solución



- Mensajería Instantánea y chat, con soporte para videoconferencia
- Editores colaborativos
- Wiki
- FAQ
- Seguimiento de errores
- Catálogo de widgets disponibles de la propia forja.
- Encuestas
- herramientas para lluvia de ideas
- herramientas para mapas mentales
- Herramientas para aprendizaje (Moodle)

### 4.2.10 Capas de aplicación, asistentes, API, gestión de comunidades y redes sociales.

Estos bloques funcionales agrupan las principales carencias que el documento de análisis de forjas había detectado en las forjas actuales., y por lo tanto, las que son principal objeto de este proyecto y habrá que desarrollar. Siguiendo las premisas descritas anteriormente, se propone el estudio de:

**Django:** es un framework de desarrollo web de código abierto, escrito en Python, que cumple en cierta medida el paradigma del Modelo Vista Controlador. Fue desarrollado en origen para gestionar varias páginas orientadas a noticias de la World Company de Lawrence, Kansas, y fue liberada al público bajo una licencia BSD en julio de 2005. La meta fundamental de Django es facilitar la creación de sitios web complejos. Django pone énfasis en el re-uso, la conectividad y extensibilidad de componentes, el desarrollo rápido y el principio No te repitas (DRY, del inglés Don't Repeat Yourself). Python es usado en todas las partes del framework, incluso en configuraciones, archivos, y en los modelos de datos. Tiene soporte para gran variedad de bases de datos y cuenta con numerosas módulos que extienden su funcionalidad, y entre ellos varios para diseño e implementación de APIs. Existe un fork, django-nonrel, que será integrado en la versión 1.4, que da soporte completo a almacenes NoSQL.

**Ruby on rails:** también conocido como **RoR** o **Rails** es un framework de aplicaciones web de código abierto escrito en el lenguaje de programación Ruby, siguiendo el paradigma de la arquitectura Modelo Vista Controlador (MVC). Trata de combinar la simplicidad con la posibilidad de desarrollar aplicaciones del mundo real escribiendo menos código que con otros frameworks y con un mínimo de configuración. El lenguaje de programación Ruby permite la metaprogramación, de la cual Rails hace uso, lo que resulta en una sintaxis que muchos de sus usuarios encuentran muy legible. Rails se



## 11PRO001CT015

Documento de conclusiones de análisis y definición de la solución



distribuye a través de RubyGems, que es el formato oficial de paquete y canal de distribución de bibliotecas y aplicaciones Ruby. Utiliza la licencia MIT.

**Symfony:** es un completo framework PHP diseñado para optimizar el desarrollo de las aplicaciones web mediante algunas de sus principales características. Separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web (MVC). Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. El resultado de todas estas ventajas es que no se debe reinventar la rueda cada vez que se crea una nueva aplicación web. Licencia MIT.

Implícito a esta capa se encuentra el almacén de información relacionada con el propio proyecto, y que también es usado por los framework como almacén de datos de la propia aplicación.

**Mongodb:** es una base de datos orientada a documentos, de alto rendimiento y sin esquemas, escrito en C++. Maneja colecciones de documentos en formato JSON, con esquemas dinámicos que ofrecen simplicidad y potencia (no son necesarias las migraciones de datos), lo que la hace tener un modelo de datos más natural para ciertas aplicaciones, dado que los datos pueden ser ordenados en jerarquías complejas y aún ser indexados y consultados. Otras características son índice de todos los atributos, replicación y alta disponibilidad ,replica a través de redes LAN y WAN, auto-sharding, escalado horizontal sin comprometer la funcionalidad, consultas enriquecidas basada en documentos consulta, permite modificaciones atómicas, Map/Reduce flexible de agregación y procesamiento de datos. GridFS para almacenar archivos de cualquier tamaño, sin complicar su stack. Licenciado bajo GNU AGPL v3.0

**PostgreSQL:** Una base de datos de tipo empresarial, que cuenta con características avanzadas tales como Control Multi-Version de concurrencia (MVCC), tablespaces, replicación asíncrona, transacciones anidadas (puntos de retorno), backups en línea , un optimizador de consultas, un sofisticado planificador , registro de acciones para la tolerancia a fallos. Es compatible con conjuntos de caracteres internacionales, las codificaciones de caracteres multibyte, Unicode, y es la configuración regional para la ordenación, mayúsculas y minúsculas, y el formato. Es altamente escalable, tanto en la gran cantidad de datos que puede manejar y en el número de usuarios concurrentes que puede acomodar. Licencia PostgreSQL

**MySQL:** es un sistema de gestión de bases de datos relacional, multihilo y multiusuario. MySQL es muy utilizado en aplicaciones web, como Drupal o phpBB, en plataformas (Linux/Windows-Apache-MySQL-PHP/Perl/Python), y por herramientas de seguimiento de errores como Bugzilla. Su popularidad como aplicación web está muy ligada a PHP, que a menudo aparece en combinación con MySQL. MySQL es una base de datos muy rápida en la lectura cuando utiliza el motor no transaccional MyISAM, pero puede



## 11PRO001CT015

Documento de conclusiones de análisis y definición de la solución



provocar problemas de integridad en entornos de alta concurrencia en la modificación. En aplicaciones web hay baja concurrencia en la modificación de datos y en cambio el entorno es intensivo en lectura de datos, lo que hace a MySQL ideal para este tipo de aplicaciones. Licencia GNU GPL

### 4.3 Criterios de evaluación

Los criterios que se van a utilizar para la evaluación de las diferentes tecnologías, son bastante específicos de las tecnologías, dado que el ámbito de los productos va a marcar las evaluaciones a realizar. Como norma general se especifican los siguientes:

1. **Licencias** de las tecnologías evaluadas
2. **Coste:** Coste de licencias, coste horas hombre, costes hardware, estimación de desglose de costes entre despliegue y mantenimiento. Costes de desarrollos a medida necesarios, costes de servicios asociados, etc...
3. **Salud:** Indicará el nivel de salud del entorno a evaluar midiendo indicadores como; número de commits, número de versiones, número de desarrolladores, Desarrolladores activos, Existe o no un Roadmap de desarrollo, Formato de comunidad (Fundación, Asociación, Corporación, Asamblea,...), Productos derivados, Idiomas soportados, etc...
4. **Mantenibilidad:** Medirá el esfuerzo necesario para su mantenimiento una vez en operación. Existencia de comunidad que de soporte a la herramienta, empresas u organizaciones que realicen mantenimiento, complejidad del lenguaje de programación usada en la herramienta, complejidad de la herramienta en si de administración.
5. **Integración:** Medirá las posibilidades de integración de la herramienta a evaluar, en un sentido amplio, interfaces que presenta, servicios web, posibilidades de integración visual, integración con sistemas de seguridad, SSO, disponibilidad de APIs, lenguajes de programación que soporta,
6. **Accesibilidad:** Se medirá mediante estándares conocidos: el nivel de cumplimiento de accesibilidad debe ser AA y que donde debe ser obligatorio el cumplimiento del mismo es en todas aquellas partes de la forja que no requieran autenticación. En las partes que requieran autenticación, el peso de la accesibilidad puede ser menor.
7. **Usabilidad:** Dado que no hay un estándar de medición de usabilidad, se detallarán perfectamente las medidas a realizar y la forma de ponderación de los resultados para dar un valor a esta medida. El criterio de **usabilidad** se referirá a tanto la localización como la percepción general del usuario.
8. **Funcionalidad:** Esta medida nos dará una indicación del nivel de ajuste entre



## 11PRO001CT015

Documento de conclusiones de análisis y definición de la solución



las funcionalidades aportadas por la herramienta y las requeridas por los requisitos funcionales.

### 5 MATRIZ DE TRAZABILIDAD DE REQUISITOS FUNCIONALES

A fin de identificar que los bloques funcionales están alineados con los requisitos de la aplicación, se ha cruzado ambos en una matriz que identifica qué relación de dependencia existe entre un requerimiento y el bloque funcional que lo provee o que depende de éste.

Se han reflejado dos tipos de cruce en la matriz adjunta:



Cuando el requisito está soportado por el bloque Funcional.



Cuando el requisito está relacionado o parcialmente implementado en el bloque funcional.

El detalle de la matriz de trazabilidad se puede consultar en el documento anexo [Anexo II. Matriz de trazabilidad requisitos-bloques v01r00.ods](#)

Se incluye también a continuación de esta un ejemplo de matriz de trazabilidad que resultaría una vez evaluadas las tecnologías.

Asimismo esta tabla se puede consultar en detalle en el documento anexo [Anexo III. Matriz de trazabilidad requisitos-productos v01r00.ods](#)

### Matriz de Trazabilidad de la Arquitectura

REQ Puntuación 90-100	Descripción	API	Almacenamiento	Asistentes	Capa Aplicación	Comunicaciones	Asíncronas Gestión de Comunidades	Herramientas Colaborativas	Infraestructuras De Proyectos	Metodologías Software Factory	Minería de Datos	Redes Sociales	Repositorios	Sistema Búsquedas Virtualización	Orquestación Servicios	Conectores	Ponderación
FNGREQ43	La FNG debe adherirse a la tendencia observada de usar sistemas de control de versiones distribuidos al tiempo que soportar el uso de los centralizados (SVN) mediante hooks traductores		✓	✓	✓	✓			✓				✓		✓	✓	12
FNGREQ33	La FNG, con el objetivo de integrar en todo el ciclo de vida de un proyecto el aspecto legal, debería contar con un gestor de licencias tipo y un potente buscador alrededor de ellas.			✓	✓						✓			✓			8
FNGREQ39	La FNG debe facilitar la definición de subforjas con identidad visual diferenciada.			✓	✓		✓	✓	✓				✓				8
FNGREQ42	La FNG debe estar disponible en multilinguaje y soportar internamente localización			✓	✓		✓	✓			✓			✓			10
FNGREQ45	Se requiere la inclusión de herramientas 2.0 de gestión del conocimiento en la forja.	✓					✓	✓				✓		✓		✓	10
FNGREQ32	En línea con las necesidades de agregar información de forjas existentes, la FNG requiere un meta agregador que añada semántica (tags, categorías, taxonomías) a los proyectos sindicados.	✓			✓	✓			✓		✓		✓	✓	✓	✓	13
FNGREQ46	La FNG debe tener una arquitectura y una vocación de descentralización de servicios de modo que puedan elegirse fácilmente proveedores externos para cada una de las grandes patas funcionales verticales en ella contenidas. Esa descentralización requiere la capacidad de agregarlos en la FNG.	✓		✓	✓	✓		✓	✓		✓		✓	✓	✓	✓	16
FNGREQ60	La FNG debe disponer de un completo set de herramientas de gobierno de proyectos. Aprobación de hitos, generación y publicación de versiones/releases, tiempos de liberación, gestión de tareas y bugs. Este gobierno de proyecto debe tener vocación upstream cuando sea posible	✓			✓	✓	✓	✓		✓					✓		10
FNGREQ63	La FNG debe disponer de indicadores claros y precisos sobre la salud de un proyecto				✓		✓	✓	✓	✓	✓		✓	✓		✓	14
FNGREQ01	De forma similar al funcionamiento de plataformas de redes sociales, debe resultar muy fácil y sencilla la comunicación entre los integrantes de la forja.	✓			✓	✓	✓	✓				✓		✓		✓	13
FNGREQ02	La FNG debe facilitar en todo lo posible la colaboración de todos los interesados, aportando información precisa; también debe potenciar especialmente la incorporación de código fuente a un proyecto.				✓		✓	✓			✓		✓	✓			11
FNGREQ04	Se requiere que la FNG disponga de las herramientas necesarias para facilitar el trabajo colaborativo tanto síncrono como asíncrono. Ejemplos válidos son Google Docs o Etherpad.	✓			✓	✓	✓	✓								✓	9
FNGREQ07	Para facilitar la transferencia completa de proyectos hacia/desde la FNG, ésta debe permitir importación y exportación de proyectos entre forjas ya existentes y con las que exista posibilidad por compatibilidad de modelo de entidades.		✓	✓	✓	✓	✓	✓	✓				✓	✓	✓		15

REQ Puntuación 70-90	Descripción	API	Almacenamiento	Asistentes	Capa Aplicación	Comunicaciones Asíncronas	Gestión de Comunidades	Herramientas Colaborativas	Infraestructuras De Proyectos	Metodologías Software Factory	Minería de Datos	Redes Sociales	Repositorios	Sistema Búsquedas	Virtualización	Orquestación Servicios	Conectores	Ponderación
FNGREQ06	La gestión administrativa de la FNG debe resultar ágil, liviana e integral en cuanto a todos los componentes funcionales.	✓	✓		✓	✓			✓						✓	✓		12
FNGREQ08	La FNG debe alcanzar un grado de usabilidad muy alto en el proceso específico de subir nuevos proyectos. Un ejemplo puede ser asistentes avanzados.			✓	✓				✓				✓	✓			✓	10
FNGREQ44	La FNG debe poder usar autenticación vía OpenID	✓			✓		✓	✓				✓	✓				✓	11
FNGREQ52	La FNG debe presentarse al usuario gestor del proyecto en términos en los que pueda realizar esa tarea sin conocimiento técnicos			✓	✓		✓	✓									✓	9
FNGREQ09	La FNG debe promover un modelo de federación entre forjas mediante la definición de un conjunto de metadatos muy completos que facilita la comunicación desatendida entre ellas.	✓	✓		✓				✓		✓			✓			✓	12
FNGREQ47	La FNG debe disponer de una completa API de programación para facilitar la creación de plugins orientados a gestión de metadatos y al aprovechamiento de la Información generada en la comunidad de usuarios.	✓			✓				✓		✓		✓	✓			✓	10

REQ Puntuación 50-60	Descripción	API	Almacenamiento	Asistentes	Capa Aplicación	Comunicaciones	Asíncronas	Gestión de Comunidades	Herramientas Colaborativas	Infraestructuras De Proyectos	Metodologías Software Factory	Minería de Datos	Redes Sociales	Repositorios	Sistema Búsquedas Virtualización	Orquestación Servicios	Conectores	Ponderación
FNGREQ10	La FNG debe contar con un Cuadro de mando orientado a perfiles de gestión y dirección.	✓		✓	✓			✓	✓		✓	✓					✓	12
FNGREQ15	Se requiere personalización de la visualización de la información de la FNG dependiendo de diferentes perfiles de usuario. Esta personalización debe estar contemplada con mentalidad mashup y modular.	✓		✓	✓			✓	✓						✓		✓	10
FNGREQ12	La FNG debe disponer de un guión claro para animar y facilitar el cubrir toda la información necesaria de los proyectos. Uno de los objetivos es dar a conocer la realidad del proyecto de forma que su valor sea conocido fácilmente y sea usado. Se evitará así parte de los nuevos proyectos nacidos del desconocimiento de otros similares en marcha.				✓	✓			✓		✓	✓	✓		✓			11
FNGREQ11	La FNG debe proponer y facilitar un ciclo de vida completo del proyecto, documentando cada fase y teniendo en consideración hitos como la propia finalización de la construcción del producto en los términos que sean apropiados.	✓			✓			✓	✓	✓							✓	9
FNGREQ14	La FNG debe orientar en las mejores prácticas de elección de metodologías de gestión, desarrollo y pruebas al tiempo que alerta de posibles riesgos del proyecto.				✓	✓		✓	✓			✓						8
FNGREQ18	Se requiere incorporar ciertas características del Entorno social/2.0.	✓			✓			✓	✓	✓			✓				✓	11
FNGREQ19	La FNG debe facilitar en extremo que la carga de información vía documentos pueda realizarse en procesos batch o por lotes.	✓	✓	✓	✓	✓				✓						✓	✓	12
FNGREQ35	La FNG requiere facilitar o construir herramientas de carga de documentos desplegadas en entorno escritorio.	✓		✓		✓										✓	✓	7
FNGREQ38	Se requiere la existencia de un módulo funcional en la FNG que sea capaz, en función de métricas específicas, de medir la calidad del código fuente de los proyectos.	✓			✓					✓		✓	✓		✓			10
FNGREQ40	La FNG debe poder integrarse fácilmente con portales de redes sociales.	✓			✓			✓	✓				✓				✓	11
FNGREQ41	La FNG debe adoptar las mejores prácticas en el reconocimiento del trabajo y la aportación de sus usuarios.				✓			✓	✓			✓	✓	✓	✓			10
FNGREQ48	La FNG debe disponer de un listado de desarrolladores, competencias y empresas con perfiles fáciles de analizar.				✓			✓	✓			✓			✓		✓	10
FNGREQ20	La FNG debe proporcionar soluciones y herramientas para que los proyectos puedan integrar directamente las traducciones y las localizaciones e informar del estado de estas dos actividades.			✓	✓			✓	✓					✓	✓			8
FNGREQ23	La FNG debe poder facilitar la personalización de la apariencia tanto para las entidades como para los proyectos.			✓	✓			✓	✓				✓		✓			7
FNGREQ24	Se requiere que exista en la FNG una predisposición a ofrecer ciertos servicios de automatización en dos puntos: - Generadores automáticos de código - Creación de esqueletos de proyecto tipo	✓		✓	✓	✓			✓	✓	✓	✓		✓			✓	15
FNGREQ25	La FNG debe tener una orientación clara hacia la nube tanto en su construcción interna como en la naturalidad en la integración con servicios de terceros ofrecidos en modalidad SaaS.	✓	✓		✓	✓									✓	✓		11

REQ Puntuación 30-50	Descripción	API	Almacenamiento	Asistentes	Capa Aplicación	Comunicaciones Asíncronas	Gestión de Comunidades	Herramientas Colaborativas	Infraestructuras De Proyectos	Metodologías Software Factor	Minería de Datos	Redes Sociales	Repositorios	Sistema Búsquedas	Virtualización	Orquestación Servicios	Conectores	Ponderación
FNGREQ30	La FNG debe tener en mente siempre al usuario final, no necesariamente tecnólogo y establecer fórmulas para establecer comunidad con ellos dentro de la FNG con el objetivo de acercarlos fácilmente a los diferentes productos o proyectos.			✓	✓		✓	✓				✓						9
FNGREQ34	La FNG requiere facilitar o construir herramientas de seguimiento de proyectos desplegables en entorno escritorio.	✓			✓	✓			✓								✓	7
FNGREQ31	La FNG deberá incluir elementos de validación de hitos y entregables para posibilitar la traslación de los elementos de vinculación contractual entre proveedores y clientes.	✓		✓	✓		✓	✓		✓		✓					✓	12
FNGREQ37	La FNG debe disponer de un servicio de creación de contenidos esencialmente formativos alrededor del proyecto. Se propone analizar el uso de terceras herramientas como LMS Moodle para cubrir esta necesidad.	✓			✓			✓			✓	✓					✓	9
FNGREQ49	La FNG debe facilitar la generación de bounties o recompensas por resolver determinadas necesidades				✓		✓				✓	✓						6
FNGREQ58	La FNG debe estar construida siguiendo patrones de alta disponibilidad	✓	✓		✓	✓			✓						✓	✓		14
FNGREQ15-BIS	Se facilitará la personalización de la interfaz gráfica, que debe de estar diseñada con mentalidad mashup y modular.			✓	✓		✓	✓				✓		✓				10





**11PRO001CT015**

Documento de conclusiones de análisis y  
definición de la solución



## 6 PROPUESTA DE TAXONOMÍAS DE PROYECTOS

Para realizar la propuesta se ha realizado un estudio entre las forjas mas usadas a nivel nacional e internacional, buscando en ellas una taxonomía que pudiera ser el mínimo común múltiplo de todas ellas (ver documento [Anexo IV. Estudio taxonomias forjas v01r00.odt](#)).



11PRO001CT015

Documento de conclusiones de análisis y definición de la solución

	OSO-R	JdA	JdA AALL	CTT	Morfeo	laFarga.cat	JdE	Mancomún	RedIRIS	Taxonomía propuesta
Audiencia	X	X	X	X	X		X	X	X	Obligatorio
Idioma	X				X	X	X		X	Obligatorio
Entorno	X	X			X		X	X	X	Obligatorio
Lenguaje Programación	X	X	X	X	X		X	X	X	Obligatorio
Licencia	X	X		X	X	X	X	X	X	Obligatorio
Estado desarrollo	X		X	X	X	X	X	X	X	Obligatorio
Sistema operativo	X		X	X	X		X	X	X	Obligatorio
Temática	X	X	X	X	X	X	X	X	X	Obligatorio
Comunidad	X									
Tipo Mantenimiento		X								
Consejería Org. Utiliza		X								Opcional
Consejería Org. Impulsa		X	X							Opcional
Tamaño Comunidad		X								Opcional
Base de datos		X	X							Opcional
Tipo Desarrollo		X								
Clasificación orgánica				X						
Clasificación Técnica				X						
Ámbito						X				
Concursos								X		
Proyecto Localización								X		



11PRO001CT015

Documento de conclusiones de análisis y definición de la  
solución

	OSO-R	JdA	JdA AALL	CTT	Morfeo	laFarga.cat	JdE	Mancomún	RedIRIS	Taxonomía propuesta
Proy. Documentación					X					
Entidad Origen			X							
Provincia Origen			X							
Taxonomía IDABC			X							
Empresas Involucradas En El Desarrollo		X	X							
Orientación Del Servicio Del Sistema De Información			X							
Orientación A Tipo De Administración Local			X							
Concurso Universitario de Software Libre									X	



Para la taxonomía propuesta se ha realizado una comparación con las principales forjas “comerciales” a nivel internacional que se refleja en la siguiente tabla:

	Taxonomía propuesta				
Audiencia	X				
Idioma	X				
Entorno	X	X			
Lenguaje Programación	X	X			X
Licencia	X	X	X		
Estado desarrollo	X	X			
Sistema operativo	X				
Temática	X	X		X	
Consejería/Org. Utiliza	X				
Consejería/Org. Impulsa	X				
Tamaño Comunidad	X				
Base de datos	X				

Tras el análisis de las distintas taxonomías, se ha detectado que:

- Existen casos que un mismo concepto es denominado de forma distinta en cada forja (por ejemplo “Orientación Del Servicio Del Sistema De Información” de JdA AALL es equivalente a “Clasificación Técnica” de CTT).
- La taxonomía temática es utilizada en determinadas ocasiones como una especie de “cajón de sastre” perdiendo su utilidad de clasificación.
- Muchas forjas comparten taxones al estar basados principalmente en Gforge.
- Las taxonomías no están agrupadas siguiendo ningún tipo de lógica.

Por todo esto, y con el objetivo de clarificar las taxonomías propuestas para la Forja de Nueva Generación, se han agrupado los taxones en base al tipo de respuesta que ofrecen según la siguiente tabla:



## 11PRO001CT015

Documento de conclusiones de análisis y definición de la solución



AGRUPACIÓN / PREGUNTA	TAXÓN
¿PARA QUÉ Y PARA QUIÉN?	Audiencia
	Sector (temática)
	Clasificación funcional(temática)
¿SITUACIÓN?	Estado de desarrollo
	Tamaño de la comunidad
INFORMACIÓN TÉCNICA	Lenguaje de programación
	Entorno
	Sistema Operativo
	Base de datos
	Tipo de Licencia
	Idioma principal
	Localizaciones del proyecto

De igual forma y con el objetivo de sencillez de clasificación y búsqueda de proyectos, la taxonomía temática se ha desagregado en dos taxones:

- **Sector:** su objetivo será clasificar los proyectos en base a sector de actividad al cual va destinado (*Aeroespacial, Agricultura, ...*). Sería similar a una clasificación “vertical”.
- **Clasificación funcional:** su objetivo será clasificar los proyectos en base a la funcionalidad aportada (*CRM, ERP, RRHH, ...*) , equivalente a la clasificación “horizontal”.

Se ha llegado al consenso de que los apartados “Consejería / Organismo que impulsa” y “Consejería / Organismo que utiliza”, no serán taxonomías con valores a seleccionar, sino que serán características de los proyectos, y se introducirán en un campo abierto, sobre el que no se harán clasificaciones.

Por último, indicar que para cada taxón, buscando de nuevo usabilidad y sencillez, se limitarán el conjunto de valores posibles a aquellos más relevantes evitando así que los creadores de proyectos se pierdan realizando la clasificación y que los usuarios se aturden con tal volumen de posibilidades.

A continuación se describe cada uno de los taxones propuestos para la Forja de Nueva Generación así como un ejemplo de posibles valores a contener.



## 11PRO001CT015

Documento de conclusiones de análisis y definición de la solución



### 6.1 ¿Para qué y para quién va destinado?

#### 6.1.1 Audiencia

Mediante esta clasificación se definirá el público objetivo del proyecto.

A continuación se muestra un ejemplo de posibles valores propuestos:

- AA.PP.
  - Interestatal
  - Estatal
  - Regional
  - Local
    - Mancomunidad
    - Ayuntamiento
    - Diputación Provincial/Cabildo Insular.
- Ciudadanos
- Empresa
- Comunidades de desarrollo
- Tipo Usuarios
  - Usuarios escritorio
  - Desarrolladores
  - Administradores

#### 6.1.2 Sector

El objetivo será clasificar los proyectos en base a sector de actividad al cual va destinado (*Aeroespacial, Agricultura, ...*). Sería similar a una clasificación “vertical”.

A continuación se muestra un ejemplo de posibles valores propuestos:

- Aeronáutica
- Agricultura
- Automoción
- Telecomunicaciones
- Banca
- Transporte y almacenamiento
- Hostelería
- ...

#### 6.1.3 Clasificación funcional

El objetivo será clasificar los proyectos en base a la funcionalidad aportada (*CRM, ERP, RRHH, ...*), equivalente a la clasificación “horizontal”

Mediante esta clasificación se identificará sobre qué versa el proyecto.

- ERP
- RRHH
- CRM
- ...



## 11PRO001CT015

Documento de conclusiones de análisis y definición de la solución



### 6.2 ¿Situación del proyecto?

#### 6.2.1 Estado de desarrollo

Mediante esta clasificación se identificará en que situación o estado se encuentra el proyecto.

A continuación se muestra un ejemplo de posibles valores propuestos:

- Planificación
- En desarrollo
- Beta
- Producción / Estable / Maduro
- En vías de extinción

#### 6.2.2 Tamaño de la Comunidad

Mediante esta clasificación se identificará el el tamaño de la comunidad que está soportando el proyecto diferenciando usuarios de desarrolladores.

A continuación se muestra un ejemplo de posibles valores propuestos:

- Desarrolladores
  - Desarrolladores :: 1
  - Desarrolladores :: 1-5
  - Desarrolladores :: 5-10
  - Desarrolladores :: 10-50
  - Desarrolladores :: 50-100
  - Desarrolladores :: 100\+
  - Desarrolladores :: 1000\+
  - Desarrolladores :: 10000\+
- Usuarios
  - Usuarios :: 1
  - Usuarios :: 1-5
  - Usuarios :: 5-10
  - Usuarios :: 10-50
  - Usuarios :: 50-100
  - Usuarios :: 100\+
  - Usuarios :: 1000\+
  - Usuarios :: 10000\+

### 6.3 Información Técnica

#### 6.3.1 Entorno

Mediante esta clasificación se identificará el entorno de ejecución del proyecto.

A continuación se muestra un ejemplo de posibles valores propuestos:

- Consola (Basado en texto)



## 11PRO001CT015

Documento de conclusiones de análisis y definición de la solución



- Sin entrada/salida (Servicio/Demonio/API)
- Entornos de escritorio
- Otros

### 6.3.2 Idioma principal

Mediante esta clasificación se identificará el idioma principal del proyecto.

A continuación se muestra un ejemplo de posibles valores propuestos:

- Español
  - Catalán
  - Valenciano
  - Balear
  - Gallego
  - Euskera
  - Inglés
  - Francés
  - Alemán
- (Resto de idiomas oficiales)

### 6.3.3 Localizaciones del proyecto

Mediante esta clasificación se identificará el/los idiomas a los que ha sido localizado el proyecto.

A continuación se muestra un ejemplo de posibles valores propuestos:

- Español
  - Catalán
  - Valenciano
  - Balear
  - Gallego
  - Euskera
  - Inglés
  - Francés
  - Alemán
- (Resto de idiomas oficiales)

### 6.3.4 Lenguaje de programación

Mediante esta clasificación se identificará el/los lenguajes de programación utilizados en el desarrollo del proyecto.

Al ser este un apartado muy extenso proponemos la utilización de cualquiera de las extensas listas utilizadas en otras forjas, aunque recomendamos reducir esta a aquellos que realmente podrían tener una representación significativa dentro la forja.

### 6.3.5 Tipo de licencia

Mediante esta clasificación se identificará el tipo de licenciamiento del producto generado



## 11PRO001CT015

Documento de conclusiones de análisis y definición de la solución



por el proyecto.

A continuación se muestra un ejemplo de posibles valores propuestos:

- Licencia Pública General de GNU (GPL)
- Licencia Pública General de Biblioteca o "Inferior" de GNU (LGPL)
- Licencia Apache 2.0
- Nueva licencia BSD
- Licencia MIT
- Licencia Pública Mozilla 1.1 (MPL)
- Licencia de desarrollo común y distribución
- Licencia pública común 1.0
- Licencia pública Eclipse
- Licencia Pública de la Unión Europea (EUPL)
- CeCILL
- Otras

### 6.3.6 Sistema operativo

Mediante esta clasificación se identificará el tipo de sistema operativo soportado por el proyecto.

A continuación se muestra un ejemplo de posibles valores propuestos:

- Distribuciones Linux
- Windows
- Mac OS
- Dispositivos Móviles
- Multiplataforma Java
- Otros

A esta clasificación se le podría generar una subdivisión con los Sistemas operativos concretos de cada grupo.

### 6.3.7 Base de Datos usada.

Mediante esta clasificación se identificará la base de datos usada por el proyecto, como medio para almacenar los datos. Para las AAPP suele ser muy importante a la hora de seleccionar una u otra aplicación.

A continuación se muestra un ejemplo de posibles valores propuestos:

- MySQL
- PostgreSQL
- ORACLE
- SQL Server
- MS Access
- Otros SGBD



## 7 CONCLUSIONES

Del trabajo llevado a cabo hasta el momento se ha llegado a una serie de conclusiones que pueden ser de interés. Está demostrada la alta demanda de las capacidades mencionadas para las forjas de nueva generación por parte de la comunidad. Esto hace que exista un alto número de stakeholders dispuestos a apoyar el proyecto desde diversas perspectivas. Este apoyo institucional sumado al liderazgo y alta implicación de la dirección de proyecto de Cenatic hace augurar un buen futuro al mismo.

Se ha llegado a una base de requisitos que conforman un conjunto ambicioso que de llevarse a efecto al cien por cien, superaría en todos los sentidos cualquier forja actual o en construcción y por tanto satisfaría una gran parte de las demandas sobre las forjas que tiene la comunidad. Pero dicho conjunto de requisitos a la vez parece un conjunto realista, y que según se ha podido apreciar por el estudio técnico inicial podrían estar cubiertos en parte por tecnologías existentes, con lo que el trabajo desde cero sería mínimo.

A partir de ahora, dentro del proyecto se comenzará un estudio exhaustivo de cada una de las tecnologías mencionadas, evaluando cada una de ellas según el conjunto de criterios establecidos. Con los indicadores obtenidos de este estudio se tomarán una serie de decisiones sobre la arquitectura conformando un documento que se convertirá en la arquitectura de referencia de la plataforma.

En paralelo se irá abordando el estudio del prototipo funcional, definiendo para cada uno de los requisitos establecidos una serie de pantallas e interfaces. Así como el esquema de navegación por los mismos.

Como gran reto observado en el proyecto se tiene que se está definiendo hoy la forja del mañana, con las dificultades que ello conlleva. Por ejemplo en cuanto a la volatilidad de los requisitos. En tres semanas de trabajo con el conjunto de requisitos con diferentes actores, se han generado nuevas versiones de un alto porcentaje de los mismos.

Esto que puede ser comprensible en un proyecto de carácter tan innovador puede suponer un grave handicap en el entorno de una contratación pública en la que el conjunto de los requisitos debe ser cerrado y no sujeto a cambios.

A este reto se debe sumar la posible dificultad de trabajar con diferentes proveedores a lo largo del proyecto que puede hacer que las concepciones que se tienen de la forja no se transmitan de forma adecuada entre las diferentes empresas que están colaborando en el proyecto.

Este riesgo en cualquier caso siempre se podrá mitigar manteniendo una dirección fuerte de proyecto como la actual así como una oficina técnica de proyecto que asegure que el soporte documental del proyecto refleja todos los aspectos cruciales del mismo.



## 11PRO001CT015

Documento de conclusiones de análisis y definición de la solución



Debido a que el proyecto cuenta con una marcada tendencia innovadora, habrá que evaluar adecuadamente que las tecnologías utilizadas no sólo cumplen con los requisitos, sino que también habrá que analizar los posibles riesgos tecnológicos que supone seleccionar una tecnología.

Si bien a día de hoy determinadas tecnologías parecen novedosas y vienen precedidas de cierto nivel de “hype”, será muy valorable que existan implementaciones de esos servicios en producción y en entornos de tamaño medio-grande. De idéntica forma, habrá otras que no tengan el nivel de madurez suficiente como para cumplir con producción, pero habrá que tener en cuenta el *timeline* del producto, pues para cuando se inicie la codificación de la solución, el producto puede contar ya con una versión estable y aprovechar las novedosas características que aporte.

Así mismo, y aunque ya fuera introducido con anterioridad, hay que incidir en dos puntos claves:

**Licencias:** puede darse el caso de que aún teniendo herramientas que cumplan una determinada misión, el licenciamiento del software de lugar a incompatibilidades de licencias, lo que haría necesario el desarrollo de ese componente desde cero

**Tecnologías:** el uso de diferentes lenguajes de base en una solución software de nueva generación es una práctica bastante habitual, pues cada una de las tecnologías suele brillar en unas áreas determinadas y flaquear en otras. Sin embargo, esto hace que la complejidad de la plataforma aumente considerablemente. A fin de simplificar el stack de tecnologías y los requerimientos técnicos de aquellos que participen en el desarrollo de la FNG, la selección de herramientas también ha de contar con un criterio enfocado a la no proliferación de dependencias.