Universidad Complutense

Facultad de Informática

eGorilla

Plan de Gestión de la Configuración

Asignatura: *Ingeniería del Software*

Curso Académico: *2008/2009*

Grupo: *4º B*

Índice

1. Introducción

1.1. Propósito

El objetivo de este documento es establecer las **actividades de Gestión de la Configuración del Software** que se van a realizar en el marco del proyecto eGorilla, cómo se van a realizar, quién las va a realizar, cuando y con qué recursos. Todo ello con el fin último de poder **controlar en cada momento**, a lo largo de todo el ciclo de vida, el estado del sistema que se está desarrollando y la evolución seguida.

Los destinatarios de este documento son **los miembros del equipo de desarrollo de la aplicación**, ya que son ellos los interesados en determinar cómo se van a coordinar todos los elementos.

Se van a definir cuales serán los **elementos de configuración del software**, todas las piezas constitutivas de los productos finales que se van a entregar y cuales van a ser las **herramientas de desarrollo**. Ejemplos de esto último pueden ser los entornos de programación, sistemas de desarrollo de documentación y sistemas de control de versiones de dichos **elementos de configuración del software**.

Además se determinará **qué tipo de modelo de desarrollo** se seguirá durante la implementación, y qué objetivos, revisiones, modificaciones, etc. han de hacerse para cumplir con lo esperado en cada una de las diferentes versiones del software y documentación. Esto hasta la entrega final, y después puede extenderse si hubiera un mantenimiento o ampliación de la aplicación.

1.2. Alcance del plan

El ámbito de este **Plan de Gestión de la Configuración** abarca un único proyecto: el desarrollo de una aplicación P2P, a la que se ha bautizado como **eGorilla**, a realizar como práctica de la asignatura de **Ingeniería del Software** durante el curso **académico 2008/2009** en la **Facultad de Informática** de la **Universidad Complutense de Madrid**.

En lo referente a detalles básicos acerca de su funcionalidad y características generales, se remite al lector a consultar el documento de **Especificación de Requisitos.**

Nos basaremos para el desarrollo del producto en un **Modelo de Proceso en Espiral,** que en sucesivas iteraciones, nos permitirá ir pasando por todas las fases, avanzando así **en paralelo** en todos los frentes, hasta

obtener el grado de completitud deseado en la entrega final. De cara a la gestión de la configuración, se identificarán inicialmente un conjunto de elementos de configuración del software, que se irán ampliando con otros nuevos a medida que se van completando iteraciones.

A continuación se presenta una lista con los **elementos de configuración del software** que se han identificado para esta primera iteración:

- **Documento relativos a la documentación (incluido el presente documento)**
- **Código Fuente de la aplicación (Cliente P2P y Servidor P2P)**

No se va a incluir dentro de la gestión de configuración ningún otro software (de apoyo) diferente del que se va a desarrollar.

Se va a aplicar un **grado de formalidad medio** sobre las actividades de gestión de la configuración del software, ya que no se trata de un proyecto excesivamente grande o complejo y se enmarca dentro de la realización de la práctica de una asignatura.

Estas actividades cubrirán el ciclo de vida del producto desde la concepción hasta su puesta en funcionamiento (entrega final). Después podría extenderse si hubiera un mantenimiento o ampliación de la aplicación.

Las **limitaciones** a tener en cuenta en este proyecto son de índole temporal, de hardware y de personal:

- **La primera** viene determinada por tener como **tiempo máximo** el correspondiente a un curso académico (nueve meses). Los objetivos de desarrollo final deben poder ajustarse dentro de ese plazo.
- **La segunda** viene determinada por el **software/hardware** disponible en los **laboratorios de la facultad**. Si bien el software será desarrollado en Java, y por tanto es multiplataforma, la aplicación debe poder funcionar y utilizar la red de computadores existente en el laboratorio. Además la disponibilidad de herramientas software es limitada.
- **La tercera** es relativa al **número de personas y tiempo de dedicación al proyecto**. Se debe tener en cuenta que dicha dedicación no es total y que el equipo está formado por trece personas, a la hora de ajustar la planificación temporal.

Para este proyecto, partimos de dos tipos de supuestos:

- **Supuestos de usuario:** Para el manejo de la aplicación, se presuponen unos **conocimientos mínimos de informática** a nivel de usuario, para el correcto manejo del programa cliente. Éstos son relativos básicamente al manejo intuitivo de una aplicación gráfica.

Sin embargo, para el programa servidor, se presuponen conocimientos de informática más avanzados: manejo de consolas de comandos, configuraciones de red, y familiaridad con plataformas basadas no sólo en sistemas Windows, sino también tipo Unix(Linux) y Mac.

El usuario no va a participar en las actividades de gestión de la configuración.

- **Supuestos de desarrollador:** El equipo de desarrollo deberá estar familiarizado (o familiarizarse) con el lenguaje de programación **Java**, entornos de desarrollo para el mismo (en particular **NetBeans**), conocimientos de **redes** a nivel suficiente como para comprender el diseño de la aplicación, etc. De cara a la gestión de la configuración tendrá que familiarizarse con la **herramienta de control de versiones SVN**.

1.3. Definiciones y acrónimos

Se remite al lector al apartado con el mismo nombre relativo al documento **Glosario_De_Terminos.doc.**

1.4. Referencias

Los principales documentos que usamos como apoyo para la elaboración del presente documento son:

1. **Standard IEEE 828 – rev. 2005**.
2. Documentación de **Subversion**.
3. **Transparencias** vistas en clase.

2. Especificaciones de gestión

2.1. Organización

Como ya se ha comentado anteriormente, el desarrollo de este proyecto se enmarca en el contexto de la realización de una práctica de carácter docente. **Solamente los miembros del equipo de desarrollo** van a participar en o ser **responsables** de las actividades de **gestión de la configuración**. Para esta primera iteración, la estructura de trabajo que se está siguiendo obedece a un esquema horizontal. Los miembros de equipo colaborarán en todas las partes que comprende el desarrollo. Las aportaciones concretas a cada parte pueden consultarse en el documento de **Plan de Iteraciones**, y en el documento de **Arquitectura del Software**, en el punto de **Vista de Asignación de Trabajo**.

De cara a la **cuarta iteración**, con fecha de 27-02-09, se ha abandonado la estructura horizontal, y se ha establecido un **jefe de proyecto,** el cual coordina las actividades agilizando la toma de decisiones y reparto de tareas dentro del grupo de trabajo.

2.2. Responsabilidades

La responsabilidad de realizar las actividades de gestión de configuración recaerá solidariamente sobre los **miembros del equipo de desarrollo**. Al trabajar en un contexto de colaboración completa, no se han establecido responsabilidades jerárquicas ni organizativas. No hay miembros “dueños” de ninguna parte de la estructura de desarrollo, ni de los productos que se van generando. **Todos pueden colaborar en todo**.

A fecha de la cuarta iteración (ver punto anterior), la colaboración de cada miembro en cada ECS, **debe ser aprobada por el jefe de proyecto**.

2.3. Plan de implementación

Consultar el subapartado de **enumeración de las líneas base del apartado 5**, y el documento de **planificación** (de este último principalmente el **plan de fase**) para concebir una idea general de cómo se va a ir implementando el proyecto, y en qué momento se establecen los hitos importantes de la elaboración, pruebas y entregas.

3. Políticas, directivas, procedimientos

Como consecuencia del desarrollo que se ha seguido en la realización del proyecto, este apartado no ha sido aplicable en su totalidad y se tendrá en cuenta en posteriores iteraciones.

3.1. Niveles del software en un árbol jerárquico

La estructura y los componentes software que van a conformar este proyecto no se ha establecido todavía en el momento de redactar este documento. Este apartado se rellenará más adelante, previsiblemente al final de la fase de elaboración, cuando ya se tenga una visión de la arquitectura general del sistema.

A fecha de la cuarta iteración se han comenzado a establecer niveles software en lo relativo al diseño de la aplicación, aunque aún a alto nivel.

3.2. Nombrado de programas y módulos

Se ha determinado que los nombres de programas, módulos, paquetes y clases no contengan acentos y, si se trata de nombres compuestos, utilizar el guión bajo '_' como separador. Además los paquetes se han de nombrar exclusivamente en minúsculas, las clases deben empezar con la primera letra en mayúscula y **se utilizarán nombres en castellano**.

En cuanto al código fuente del proyecto, los convenios anteriores también se aplicarán a la nomenclatura de los atributos, métodos, etc. Los atributos siempre comenzarán por el carácter '_'. Además de esto, todas las funciones/métodos han de incluir una documentación mínima (descripción del método, de sus atributos, etc.) necesaria para generar fácilmente ayuda en formato HTML mediante javadoc.

3.3. Identificación de productos software

Hasta el momento se han distinguido **dos productos principales**: el **cliente P2P** y **servidor P2P**.

El primero de ellos es el que será manejado por el usuario y el segundo de ellos será controlado por alguna persona con unos conocimientos más específicos en el manejo de la aplicación.

3.4. Identificación de documentación

Toda la documentación generada ha de ser **subida al repositorio**, siguiendo los mismos criterios anteriores de nombrado para programas y módulos, es decir, nombres de los documentos sin acentos y con el carácter '_' para separar palabras. Se ha decidido así para evitar problemas que pueden acarrear algunos clientes de SVN.

Por otra parte, los documentos han de seguir un **formato homogéneo,** en apartados como el encabezado y pie de página, fuente de texto, tamaño, estilo, etc.

3.5. Identificación de medios y ficheros

Por el momento el único medio de almacenamiento es el repositorio de código eGorilla dentro del servicio **googlecode** (ver punto 4.4 para más información).

En cuanto a las **herramientas de desarrollo**, finalmente se ha optado por **NetBeans 6.1** para el desarrollo de software y diseño UML, y Microsoft Word 2003 para la documentación. Comentar que no sólo se consideró la utilización de Eclipse, sino que tuvo su breve período de utilización, pero se descartó principalmente por las posibilidades ofrecidas por NetBeans para el desarrollo de diagramas UML.

Pero a fecha de 28-01-09 se está considerando cambiar Microsoft Word por OpenOffice Writer y demás programas de esa suite, pues ofrecen mayor compatibilidad. Cualquier decisión final que está por llegar, será consignada entre otros sitios en este documento.

Se ha decidido para la **entrega del día 27-02-09** (revisión de la entrega efectuada el día 30-01-09 y corregida tras la presentación del día 19-02-09, ver apartado 4.2) que hasta nueva orden, la documentación será desarrollada con Microsoft Word 2003. Se sigue manteniendo **NetBeans 6.1** para el diseño UML que irá integrado en los documentos correspondientes.

4. Actividades de GCS

4.1. Identificación de la configuración

Una lista detallada de los ECS identificados hasta el momento puede verse en el subapartado **Alcance del proyecto** del apartado Introducción. Solamente van a ponerse bajo control de la configuración **productos obtenidos del proceso de desarrollo** (no otros elementos del entorno de desarrollo). De momento no es necesario descomponer dichos ECS en otros más sencillos. Complicaría la gestión de la configuración, ya que a estas alturas serían demasiados objetos a tener en cuenta a la vez para la magnitud de la entrega.

De momento, las relaciones que se establecen entre los ECS, son relaciones de sucesión, es decir, tenemos ya nuevas versiones de ECS que han sido revisados. Esto viene reflejado explícitamente en el número de versión de cada ECS (ver TECS, **Anexo A**).

4.2. Enumeración de las líneas base

Con cada entrega parcial que se realice se generará **una nueva línea base**, que estará compuesta por los ECS especificados por el profesor.

Se remite al lector a la lectura del documento **Plan_De_Iteraciones.doc** donde se especifica en profundidad lo anteriormente comentado.

4.3. Bibliotecas y repositorios a utilizar

Para el control automático de las versiones de los *ECS* a lo largo de todo el proyecto, vamos a utilizar un repositorio de **Google Code** basado en el sistema de control de versiones **Subversion**.

La **url** del repositorio está disponible en:

http://code.google.com/p/egorilla/

Accesible en modo **sólo lectura** para cualquiera que desee consultar los *ECS.* Los miembros del equipo de desarrollo tienen acceso completo a dicho repositorio.

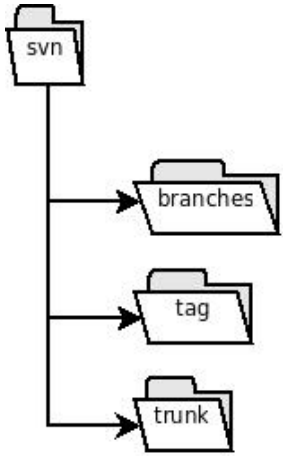
Para gestionar este sistema de versiones, se usará tanto el cliente de **Subversion** que tiene integrado el **IDE NetBeans**, como los clientes **svn** (línea de comandos) y **RapidSvn** (interfaz gráfica) de Linux, **Tortoise** para Windows, y sus equivalentes para Mac. La recuperación, modificación y almacenamiento de los ECS se realizará mediante los comandos ofrecidos por el cliente de **Subversion**.

Así contaremos con las siguientes bibliotecas de trabajo:

- **Biblioteca de desarrollo:** Se almacena en la carpeta **branches** del repositorio (ver punto 4.4) En ella, cada miembro del equipo tendrá asignado un directorio personal, donde llevará a cabo aquello que le corresponda.
- **Biblioteca de integración:** Se almacena en la carpeta **trunk** del repositorio (ver punto 4.4). Aquí se irán unificando todos los ECS realizados en la carpeta **branches**, de acuerdo con la configuración necesaria para cada una de las entregas del producto. Esta integración deberá ser **supervisada y aprobada por el jefe del proyecto**.
- **Biblioteca de proyecto:** Se almacena en la carpeta **tag** del repositorio (ver punto 4.4). Contendrá el producto con su configuración tras las iteraciones más importantes. Sólo podrá modificarse bajo la estricta aprobación de *todos los miembros* del equipo, **supervisada por el jefe de proyecto**.

4.4. Mecanismos para el control de versiones

La estructura de directorios del repositorio, es la siguiente:



La configuración interna de los directorios es la siguiente (susceptible a cambios, es muy importante revisar este punto y el anterior muy frecuentemente):

- **Directorio branches:** Es la biblioteca de desarrollo (ver 4.3). Deberá contener **una carpeta por cada miembro del grupo**. Dentro de cada directorio personal.
- **Directorio trunk:** Es la biblioteca de integración (ver apartado 4.3). Su disposición interna es la siguiente:
  - **Directorio doc:** Aquí se realizará la integración de los ECS relativos a la documentación. Contendrá a su vez los directorios:
    - **/formato_doc** para la unificación de las diferentes partes todavía en formato .doc.
    - **/formato_pdf** para almacenar los ECS de documentación ya convertidos al formato .pdf de entrega.
    - **/formato_mpp** que incluye los archivos con MS Proyect 2003.
    - **/presentaciones** que incluye los archivos realizados con MS Power Point correspondientes a las distintas presentaciones realizadas durante el curso.

- **Directorio eGorilla:** Donde se lleva a cabo la **integración de los ECS relativos al software** (Código Fuente).

- **Directorio tag:** Es la **biblioteca de proyecto** (ver apartado 4.3). Su contenido será el que corresponda a **versiones significativas del proyecto**, es decir, la configuración del mismo tras iteraciones principales, en las que haya un buen grado de funcionalidad. La primera prevista para **el día 27 de Marzo**.

Es indispensable que todos los miembros cuenten con un cliente SVN como los especificados en el apartado 4.3, para tener copias locales debidamente actualizadas de la configuración del proyecto. Cada actualización del repositorio debe ir acompañada de un **log explicativo**, además de un **mensaje a la lista de correo**, para que todos los miembros sean conscientes de la necesidad de actualizar su copia local, y por qué.

Para la identificación del **número de versión del proyecto** se ha decidido elegir el **número de revisión global otorgado por subversion**, para así determinar rápidamente cuál debería ser la configuración más reciente que todos los miembros del equipo deberían tener.

Ésta nueva configuración viene a sustituir a la anterior en la que se optaba por un directorio único de documentación **/trunk/doc** y dentro de él dos subcarpetas:

- **Generales:** Contenía todos los documentos de la documentación (.doc, .pdf, .mpp y .ppt)
- **ArquitecturaDelSistema:** Todos los documentos de integración de la Arquitectura del Sistema.