ESCUELA SUPERIOR DE INGENIERÍA

GRADO EN INGENIERÍA INFORMÁTICA

PLANTILLA PARA PROYECTO DE FIN DE GRADO

Alumno Alumno Alumno

13 de marzo de 2012

ESCUELA SUPERIOR DE INGENIERÍA

GRADO EN INGENIERÍA EN INFORMÁTICA

PLANTILLA PARA PROYECTO DE FIN DE GRADO

- Departamento: Lenguajes y Sistemas informáticos
- Director del proyecto: Profesor Profesor Profesor
- Autor del proyecto: Alumno Alumno Alumno

Cádiz, 13 de marzo de 2012

Fdo: Alumno Alumno Alumno

Agradecimientos

Introduzca aquí, si lo desea, los agradecimientos.

Resumen

Introduzca aquí un resumen no superior a 500 palabras, que servirá de descripción pública del trabajo realizado.

Palabras clave: Lista de palabras clave que reflejen el contenido del trabajo en aras de facilitar su búsqueda en sistemas bibliográficos.

Índice general

Índice de figuras

Índice de cuadros

Parte I

Prolegómeno

La primera parte de la memoria del PFC debe contener una introducción y una planificación del proyecto.

La introducción es un capítulo que, a modo de resumen, debe contener una breve descripción del contexto de la disciplina en la que el proyecto tiene aplicación, la motivación para su desarrollo, una explicación de la situación actual previa al proyecto y de otros sistemas existentes, en su caso, y la problemática existente que se pretende solucionar. Al final del capítulo introductorio se deben enunciar claramente los objetivos y el alcance del proyecto.

El segundo capítulo debe incluir una planificación del proyecto. La planificación deberá ajustarse a las prácticas de ingeniería en general, y de la ingeniería del software en particular. Deberá tener en cuenta los plazos, los entregables (documentos y software), los recursos (humanos y de equipamiento inventariable) y el método de ingeniería de software a emplear.

Capítulo 1

Introducción

A continuación se describe la motivación del presente proyecto, la descripción del sistema actual (si existiese), los objetivos y cómo se ha estructurado la documentación.

1.1. Motivación

Qué motivación nos ha llevado a su desarrollo. Contexto y ámbito en el que se desarrolla el proyecto.

1.2. Descripción del sistema actual

Estudio de las características del sistema actual, si existiese, profundizando en las carencias que presenta y la problemática que pretende resolver.

1.3. Objetivos y alcance del proyecto

Los objetivos conviene describirlos a varios niveles de abstracción. Es buena práctica describir un objetivo general y luego desglosar éste en objetivos específicos o subobjetivos. Dependiendo de la complejidad técnica del proyecto, el número de niveles de abstracción y/o el número de subobjetivos será mayor o menor. Los objetivos específicos se deben enumerar o denominar con algún acrónimo corto, porque esto facilitará hacer referencia a su cumplimiento en el capítulo de pruebas y evaluación más adelante en la memoria.

El alcance del proyecto señalará cuál es el grado esperado de cumplimiento de los objetivos. Un objetivo puede ser alcanzado en mayor o menor medida, dependiendo de cómo se planteen más adelante los requisitos del sistema. El alcance del proyecto podrá ser luego determinado en forma de requisitos funcionales y/o no funcionales, dependiendo de lo ambicioso que sea el proyecto y del entorno en que el sistema deba operar.

1.4. Organización del documento

Descripción de los contenidos de la presente memoria, así como del software entregado en soporte informático.

Capítulo 2

Planificación

En esta sección se describen todos los aspectos relativos a la planificación del proyecto: metodología, organización, costes, planificación y gestión de riesgos.

2.1. Metodología de desarrollo

Definición del proceso de desarrollo, ciclo de vida y metodología empleada durante la elaboración del proyecto. Las fases y/o iteraciones que proponga el método empleado deberán quedar recogidas en la planificación que se detalle más adelante.

2.2. Planificación del proyecto

Estimación temporal y definición del calendario básico (hitos principales e iteraciones). Desarrollo de la planificación detallada, utilizando un diagrama de Gantt. Análisis de la desviación temporal, comparando tiempos estimados con los finalmente empleados.

Los diagramas de Gantt que se vean (girados y divididos si hace falta).

2.3. Organización

Relación de las personas (roles) involucradas en el proyecto y de cómo se estructuran las relaciones entre las mismas para ejecutar el proyecto. Relación de los recursos inventariables utilizados en el proyecto: equipamiento informático (hardware y software), herramientas empleadas, etc.

2.4. Costes

Estudio y presupuesto de los costes de los recursos (humanos y materiales) descritos anteriormente, necesarios para el proyecto.
Para el cálculo de costes de personal pueden consultarse las tablas salariales de la UCA para el personal técnico de apoyo contratado laboral [CCOO, 2010], o bien otras más ajustadas a la realidad. El cálculo del coste del personal del proyecto debe hacerse en personas-mes, y luego hacer la correspondencia al coste monetario.

2.5. Gestión de riesgos

Enumeración de los riesgos del proyecto, indicando su posible impacto (efecto que la ocurrencia del citado riesgo tendría en el desarrollo del proyecto) y la probabilidad de ocurrencia. Una vez los riesgos son identificados y priorizados, hay que definir los planes necesarios para reducir los efectos del riesgo una vez se haya materializado o disminuir que este ocurra.

Parte II

Desarrollo

En esta parte se debe describir el desarrollo del proyecto siguiendo la metodología empleada. Sus capítulos no deben ser una descripción exhaustiva de todos los documentos, diagramas, código fuente y, en general, entregables generados, sino más bien una explicación resumida del desarrollo, estructurada según las etapas principales del proceso de ingeniería. Deben seleccionarse aquellos diagramas, fragmentos de código y secciones de los entregables que sean más significativos para dicha explicación. La totalidad de los entregables resultado del proyecto se ubicarán en los anexos y/o en el material en CD/DVD que acompañe al proyecto.

Capítulo 3

Análisis de Requisitos

En esta sección se presenta el catálogo de requisitos del sistema de información. Para ello se detallarán los actores del sistema, los requisitos funcionales, los requisitos de información, los requisitos no funcionales y las reglas de negocio. Luego se describen las diferentes alternativas tecnológicas y el análisis de la brecha entre los requisitos planteados y la solución base seleccionada.

3.1. Catálogo de actores

En este apartado se describirán los diferentes roles que juegan los usuarios que interactúan con el sistema. Los actores pueden ser roles de personas físicas, sistemas externos o incluso el tiempo (eventos temporales).

3.2. Requisitos funcionales

Descripción completa de la funcionalidad que ofrece el sistema. Para ello se emplearan casos de uso como mecanismo para representar las interacciones entre los actores y el sistema bajo estudio. Para cada caso de uso deberá indicarse los actores implicados, las precondiciones y postcondiciones, los pasos que conforman el escenario principal y el conjunto de posibles escenarios alternativos.

3.3. Requisitos de información

En esta sección se describen los requisitos de gestión de información (datos) que el sistema debe gestionar. Para ello, se desarrollará un diagrama conceptual de clases UML, identificando las clases, atributos, relaciones, restricciones adicionales y reglas de derivación necesarias.

3.4. Requisitos no funcionales

Descripción de otros requisitos (relacionados con la calidad del software) que el sistema deberá satisfacer: portabilidad, seguridad, auditoría, monitorización, fiabilidad, comunicaciones con sistemas externos, extensibilidad, rendimiento, escalabilidad, estándares de obligado cumplimiento, accesibilidad, usabilidad, aspectos de la interfaz de usuario, utilización de un determinado entorno tecnológico, etc.

3.5. Reglas de negocio

En el desarrollo del sistema, hay que tener en cuenta las denominadas reglas de negocio, es decir, el conjunto de restricciones, normas o políticas de la organización que deben ser respetadas por el sistema, las cuales suelen ser cambiantes.

3.6. Estudio de alternativas tecnológicas

En esta sección, se debe ofrecer un estudio del arte de las diferentes alternativas tecnológicas que permitan satisfacer los requerimientos del sistema, para luego seleccionar la herramienta o conjunto de herramientas que utilizaremos como base para el software a desarrollar.

3.7. Análisis GAP

Una vez seleccionado el software de base, debemos identificar y medir las diferencias entre lo que proporciona este software y los requisitos definidos para el proyecto. El resultado de este análisis permitirá identificar cuáles de éstos requisitos ya están solventados total o parcialmente por el sistema base y cuales tendremos que diseñar e implementar.

Capítulo 4

Diseño del Sistema

En este capítulo se recoge la arquitectura general del sistema de información, el diseño de la interfaz de usuario, el diseño físico de datos, el diseño de componentes software y la parametrización del software base.

4.1. Diseño de la arquitectura

En esta sección se define la arquitectura general del sistema de información, especificando las distintas particiones físicas del mismo, la descomposición lógica en subsistemas de diseño y la ubicación de cada subsistema en cada partición, así como la especificación detallada de la infraestructura tecnológica necesaria para dar soporte al sistema de información.

4.1.1. Arquitectura física

En este apartado, describimos los principales componentes hardware que forman la arquitectura física de nuestro sistema, recogiendo por un lado los componentes de servidor y los componentes de sistemas externos con los que colabora nuestro sistema y por otro, los componentes hardware de cliente.

4.1.2. Arquitectura lógica

La arquitectura lógica del sistema está formada por los elementos software (servicios, aplicaciones, librerías, frameworks, etc.) que componen el software base, más el software desarrollado para cumplir los requisitos de la aplicación. También, se recogen los componentes de sistemas externos con los que interactúa nuestro sistema, así como los componentes software del lado cliente.

4.1.3. Arquitectura de diseño

La arquitectura de diseño especifica la forma en que los artefactos software de más bajo nivel, interactúan entre sí para lograr el comportamiento deseado en el sistema. Utilizaremos el patrón arquitectónico Layers (Capas), con el cual estructuramos el sistema en un número apropiado de capas, de forma que todos los componentes de una misma capa trabajan en el mismo nivel de abstracción y los servicios proporcionados por la capa superior utilizan internamente los servicios proporcionados por la capa inmediatamente inferior.

Capa de presentación

Este grupo de artefactos software conforman la capa de presentación del sistema, incluyendo tanto los componentes de la vista como los elementos de control de la misma.

Capa de negocio

Este grupo de artefactos software conforman la capa de negocio del sistema, incluyendo los elementos del modelo de dominio y los servicios (operaciones del sistema).

Capa de integración

Este grupo de artefactos software conforman la capa de integración del sistema, incluyendo las clases de abstracción para el acceso a datos (BD o sistema de ficheros) o a sistemas heredados.

Servicios transversales

Este grupo de artefactos software pueden ser usados por elementos de cualquiera de las capas del sistema y fundamentalmente proporcionan servicios relacionados con requisitos no funcionales (calidad).

4.2. Diseño de la interfaz de usuario

En esta sección se especifican las interfaces entre el sistema y el usuario, detallando el aspecto y el comportamiento de las diferentes pantallas e informes, de acuerdo con el entorno tecnológico definido. Con respecto a las pantallas e informes, es preciso realizar un prototipo o mockup gráfico. Junto a estos bocetos hay que definir qué ocurre en los distintos componentes visuales de la interfaz cuando aparecen y qué acciones se disparan cuando el usuario trabaja con ellas. Además, es preciso elaborar un diagrama de navegación, reflejando la secuencia de pantallas a las que tienen acceso los diferentes roles de usuario y la conexión entre éstas.

4.3. Diseño de datos

En esta sección se define la estructura física de datos que utilizará el sistema, a partir del modelo de conceptual de clases, de manera que teniendo presente los requisitos establecidos para el sistema de información y las particularidades del entorno tecnológico, se consiga un acceso eficiente de los datos. La estructura física se compone de tablas, índices, procedimientos almacenados, secuencias y otros elementos dependientes del SGBD a utilizar.

4.4. Diseño de componentes

En esta sección se definen los componentes software necesarios para la implementación del sistema. Para facilitar la comprensión y la posterior implementación del sistema, es recomendable organizarlo en forma de subsistemas, los cuales se compondrán a su vez de módulos. Estos módulos (o paquetes) contendrán un conjunto de artefactos software, que representaremos en forma de clases y que corresponderán a una de las capas identificadas en la arquitectura. Para cada uno de los módulos funcionales del sistema debemos realizar un diagrama de secuencia, para definir la interacción existente entre las clases de objetos que permitan responder a eventos externos.

A partir de este diagrama, se genera el diagrama de clases de diseño, incluyendo los elementos del modelo conceptual, enriquecidos con las nuevas clases, relaciones, atributos y operaciones resultantes. Asimismo, se detallará el comportamiento de las operaciones más relevantes.

4.5. Parametrización del software base

En esta sección, se detallan las modificaciones a realizar sobre el software base, que son requeridas para la correcta construcción del sistema. En esta sección incluiremos las actuaciones necesarias sobre la interfaz de administración del sistema, sobre el código fuente o sobre el modelo de datos.

Capítulo 5

Implementación del Sistema

Este capítulo trata sobre todos los aspectos relacionados con la implementación del sistema en código, haciendo uso de un determinado entorno tecnológico.

5.1. Entorno tecnológico

En esta sección se debe indicar el marco tecnológico utilizado para la construcción del sistema: entorno de desarrollo (IDE), lenguaje de programación, bibliotecas y frameworks de desarrollo empleados, herramientas de ayuda a la construcción y despliegue, control de versiones, repositorio de componentes, integración contínua, etc.

5.2. Código fuente

Organización del código fuente, describiendo la utilidad de los diferentes ficheros y su distribución en paquetes o directorios.

5.3. Calidad de código

Resumen de las comprobaciones de análisis de código fuente, llevadas a cabo mediante el uso de herramientas automáticas de comprobación de calidad de software.

Capítulo 6

Pruebas del Sistema

En este capítulo se documentan los diferentes tipos de pruebas que se han llevado a cabo, ya sean manuales (mediante listas de comprobación) o automatizadas mediante algún software específico de pruebas.

6.1. Pruebas unitarias

Las pruebas unitarias tienen por objetivo localizar errores en cada nuevo artefacto software desarrollado, antes que se produzca la integración con el resto de artefactos del sistema.

6.2. Pruebas de integración

Este tipo de pruebas tienen por objetivo localizar errores en módulos o subsistemas completos, analizando la interacción entre varios artefactos software.

6.3. Pruebas de sistema

En esta actividad se realizan las pruebas de sistema de modo que se asegure que el sistema (tanto el software como el hardware) cumple con todos los requisitos establecidos: funcionales, de almacenamiento, reglas de negocio y no funcionales. Se suelen desarrollar en un entorno específico para pruebas.

6.3.1. Pruebas funcionales

Con estas pruebas se analiza el buen funcionamiento de la implementación de los flujos normales y alternativos de los distintos casos de uso del sistema.

6.3.2. Pruebas no funcionales

Estas pruebas pretenden comprobar el funcionamiento del sistema, con respecto a los requisitos no funcionales definidos en la etapa de análisis: rendimiento, accesibilidad, etc.

6.4. Pruebas de aceptación

El objetivo de estas pruebas es demostrar que el producto está listo para el paso a producción. Suelen ser las mismas pruebas que se realizaron anteriormente pero en el entorno de producción. En estas pruebas, es importante la participación del cliente final.

Parte III

Epílogo

En esta última parte quedarán recogidas las conclusiones y los manuales necesarios para el manejo de la aplicación resultado del desarrollo. Si se ha realizado algún tipo de evaluación de la solución proporcionada, más allá de las pruebas del sistema, también deberá venir recogida en un capítulo separado dentro de esta parte. Pueden consultarse diversos tipos de evaluaciones sobre sistemas de información en [Hevner et al., 2004]: casos de estudio, análisis estático, análisis dinámico, simulación, experimento controlado, etc.

Capítulo 7

Manual de usuario

Las instrucciones de uso del sistema se detallan a continuación.

7.1. Introducción

Resumen de los principales objetivos, ámbito y alcance del software desarrollado.

7.2. Características

Recopilación de las principales funcionalidades del sistema.

7.3. Requisitos previos

Requisitos hardware y software para el correcto uso del sistema.

7.4. Utilización

Descripción del área de trabajo del sistema y las instrucciones concretas para hacer uso de las funcionalidades del sistema.

Capítulo 8

Manual de instalación y explotación

Las instrucciones de instalación y explotación del sistema se detallan a continuación.

8.1. Introducción

Resumen de los principales objetivos, ámbito y alcance del software desarrollado.

8.2. Requisitos previos

Requisitos hardware y software para la correcta instalación del sistema.

8.3. Inventario de componentes

Lista de los componentes hardware y software que se incluyen en la versión del producto.

8.4. Procedimientos de instalación

Procedimientos de instalación y configuración de cada componente hardware y software (base y desarrollado) para asegurar la correcta instalación y explotación del sistema, así como aquellos procedimientos necesarios de migración/carga de datos.

8.5. Procedimientos de operación y nivel de servicio

Procedimientos necesarios para asegurar el correcto funcionamiento, rendimiento, disponibilidad y seguridad del sistema: back-ups, chequeo de logs, etc. También, es preciso indicar claramente aquellas actuaciones precisas necesarias para el mantenimiento preventivo del sistema y así prevenir posibles fallos en el mismo.

8.6. Pruebas de implantación

Descripción de las pruebas a realizar después de la instalación del sistema.

Capítulo 9

Conclusiones

En este último capítulo se detallan las lecciones aprendidas tras el desarrollo del presente proyecto y se identifican las posibles oportunidades de mejora sobre el software desarrollado.

9.1. Objetivos

Este apartado debe resumir los objetivos generales y específicos alcanzados, relacionándolos con todo lo descrito en el capítulo de introducción.

9.2. Lecciones aprendidas

A continuación se detallan las buenas prácticas adquiridas, tanto tecnológicas como procedimentales, así como cualquier otro aspecto de interés.

Este apartado debe recoger una comparación cuantitativa del tiempo y el esfuerzo realmente invertido frente al estimado y planificado. Estos datos pueden recogerse del sistema de gestión de tareas empleado para el seguimiento del proyecto. Es mejor resumir cuantitativamente el tiempo y esfuerzo dedicados al proyecto a lo largo de su desarrollo y medido de esta forma que escribir un sencillo "he trabajado mucho en este proyecto".

9.3. Trabajo futuro

En esta sección, se presentan las diversas áreas u oportunidades de mejora detectadas durante el desarrollo del proyecto y que podrán ser abarcadas en futuras versiones del software.

Los elementos aquí descritos deben estar en relación con lo relatado en el apartado de objetivos y alcance del proyecto descritos en la introducción.

Bibliografía

Para las referencias bibliográficas puede elegirse cualquier estilo de bibtex. Por comodidad, es recomendable usar un estilo que genere las citas usando el nombre de sus autores y ordene las referencias alfabéticamente. Por ejemplo, se recomienda el estilo apalike, que no usa citas numéricas. Si se desea confeccionar un estilo propio adaptado al castellano, puede emplearse el paquete makebst [Daly, 2003].

En las referencias hay que cuidado de que bibtex trate adecuadamente las eñes y los caracteres acentuados del castellano. Revisar también que bibtex haya generado correctamente las citas y referencias para los autores que tienen dos apellidos (como suele ser el caso en los españoles).

En la lista de referencias, conviene no abusar de las citas a páginas y sitios web. En caso necesario, incluir las referencias a la web siguiendo algún esquema estandarizado, como por ejemplo el de APA[1] [Land, 1998].

[CCOO, 2010] CCOO (2010). Tablas salariales 2010 IV Convenio Colectivo. http://www.uca.es/sindicato/ccoo/documentos/tabla-salarial-pas-laboral-2010.pdf.

[Daly, 2003] Daly, P. W. (2003). Customizing Bibliographic Style Files. http://www.ctex.org/documents/packages/bibref/makebst.pdf.

[Hevner et al., 2004] Hevner, A. R., March, S. T., Park, J., and Ram, S. (2004). Design Science in Information Systems Research. *MIS Quarterly*, 28(1):75–105.

[Land, 1998] Land, T. (1998). Estándar para hacer referencia a documentos de red en publicaciones científicas. http://www2.udec.cl/~gnavarro/2000_2/weapas.html.

[1] *American Psychological Association*

Información sobre Licencia

Incluir aquí la información relativa a la licencia seleccionada para la documentación y software del presente proyecto.