2012

UNC, Ingeniería en Computación
"Ingeniería de Software"

Mgr. Martín Miceli

[PRÁCTICO DE ING. DE SW]

Este documento describe los requisitos mínimos que debe tener el trabajo práctico final para la materia "Ingeniería de SW". Dicho trabajo consiste en la aplicación de las prácticas más importantes de ingeniería de software en base al trabajo final de concurrentes (previo o en paralelo).

Descripción del trabajo práctico

Acerca del informe final

El alumno, en grupos de aproximadamente 3 personas (máx. 4, sólo en aquellos casos pre aprobados), deberá usar el trabajo actual o pasado de concurrentes para aplicar las metodologías y conceptos vistos en clase. Se deberá presentar un reporte hecho en Word 2003/2007 o PDF (preferentemente este último) que incluya los requisitos mínimos listados en las sección ("Lista de requisitos del informe"). Si bien pueden elegir por producir documentos separados para cada área, es recomendable incluir cada grupo de requisitos como secciones separadas de un mismo documento que tenga su historial propio de versiones de modificación (ej., 1 sólo informe con secciones como "Requerimientos", "Manejo de las Configuraciones" etc.).

Acerca del formato de la entrega

El/los archivos deberán ser comprimidos en formato *.zip y enviados por email a martinmiceli@gmail.com antes de la fecha final de cierre de recepción (11-Jul-2012, 19:00 hs).

Acerca de la presentación

Además, los gráficos y tablas claves deberán ser incluidos en una presentación que se usará para defender el trabajo el día posterior a la entrega de los trabajos (12-Jul-2012). Cada equipo deberá defender el trabajo durante 10 minutos (7' presentación y 3' de preguntas y respuestas). Cada equipo recibirá justo antes de presentar 1 o 2 temas en los cuales deberán focalizarse.

Acerca de la propiedad intelectual

Es importante notar que el trabajo práctico debe ser realizado enteramente por el grupo, esperándose contribuciones similares de cada integrante (debe mencionarse el autor de cada sección y los revisores y contribuidores). Cualquier intento de copia o modificación de trabajos presentados por otros en el presente o con anterioridad, derivará a la desaprobación directa y consecuentes acciones por fraude del grupo y sus participantes.

Lista de requisitos del informe:

Manejo de las Configuraciones

1. Incluir una sección para el **plan de manejo de las configuraciones** que incluya la forma de acceder al código fuente y a los documentos incluidos en la herramienta de control de versiones. Esta sección es el índice a toda la información del proyecto incluyendo:
    a. Dirección y forma de accesos a la herramienta de control de versiones

b. Esquema de directorios y propósito de cada uno
c. Normas de etiquetado y de nombramiento de los archivos.
d. Plan del esquema de ramas a usar (y en uso). Mencionar la lógica de elección de dicho esquema.
e. Políticas de fusión de archivos (a.k.a. mergeo) y de etiquetado de acuerdo al progreso de calidad en los entregables.
f. Forma de entrega de los "releases", instrucciones mínimas de instalación y formato de entrega.
g. Listado y forma de contacto de los integrantes del equipo, así como sus roles en la CCB. También incluir periodicidad de las reuniones y miembros obligatorios.
h. Herramienta de seguimiento de bugs usado para reportar los defectos descubiertos y su estado.
i. Cualquier otra información relevante (ej. Direcciones del servidor de integración continua, etc.)

Requerimientos

2. Requerimientos que incluya los "**Diagrama de Casos de Usos**", "**Diagrama de Actividades**" y "**Diagramas de secuencias**" para introducir al lector en los requerimientos de usuario para el sistema a construir. También debería detalle de los **requerimientos funcionales** y **no funcionales de sistema**.
3. Se deberán incluir un **diagrama de arquitectura preliminar** que permita asociar requerimientos y casos de usos con los sistemas, subsistemas y módulos identificados.
4. **Matriz de trazabilidad** entre los **casos de usos y los requerimientos**, así como la trazabilidad entre los **req. de sistemas entre sí**.

Arquitectura

5. Un **gráfico de arquitectura** general para mostrar los **componentes y sus relaciones** con las **interfaces externas**. Explicitar que **patrón de arquitectura** fue usado y porqué, haciendo énfasis **como resuelve los requerimientos no funcionales**. Adicionalmente se deben incluir al menos los gráficos **UML de despliegue** y de **componentes**. Opcionalmente se puede incluir un **diagrama de contexto** (**composite: actividades y sub-sistemas**) que incluya la relación de los sistemas/subsistemas con las actividades del dominio del conocimiento de la aplicación.

Diseño e implementación

6. **Diagramas de clases** y de **objetos**. Opcionalmente generar los diagrama de secuencia y/o estados según sea aplicable con alguna parte del diseño).
7. **Diagramas de paquetes**.
8. Implementar al menos un **Patrón de diseño** (preferentemente un **Observer y/o un Strategy**). Resaltar el patrón de diseño en el diagrama de clases e indicar porqué se optó por este (explicar qué problema soluciona).

Pruebas unitarias y del sistema

9. Generar **pruebas unitarias** automáticas (a.k.a. Unit test) para el código, explicar cómo correrlos y verificar su estado. Opcionalmente: proveer los porcentajes de cobertura de código desde la herramienta.
10. Generar los **Casos de Prueba de Sistema** contra los **requerimientos funcionales** y **no funcionales**. Incluir **casos de prueba alternativos** que prueban valores límites o inusuales y que tratan de generar errores no esperados.
11. Seleccionar un subconjunto de los **casos de prueba** para actúen como **Smoke o Sanity T**ests y permitan la aceptación por parte del cliente.
12. Actualizar la **Matriz de trazabilidad** para incluir el mapeo entre casos de usos, requerimientos, **módulos o clases y casos de pruebas** (unitarias y de sistema).
13. Calcular el **"Pass/Fail Ratio** (i.e. % Pruebas Pasadas/Falladas)**"** para **cada tipo de pruebas**.
14. Proveer el **número de bugs identificados y corregidos** (agrupados por severidad). Incluir el **link a la herramienta de administración de bugs (e**.g. Bugzilla, Jira, etc.)
15. Opcional: incluir cualquier otra herramienta de gestión de la calidad de SW que haya sido usada (ej. FindBugs for análisis estático de código, etc.)

Entrega

16. Proveer la "**Nota de Entrega** (a.k.a Release Note)" con el estado del entregable al momento de la presentación del trabajo práctico. Incluir los siguientes puntos:
    a. Breve **listado de la funcionalidad** incluida (con el **estado de implementación** de c/u)
    b. **Pass/Fail Ratio** de sistema
    c. Bugs conocidos (i.e. no resueltos) en la entrega
    d. **Lugar/link** del **entregable y de las instrucciones de instalación**

Estimaciones y Datos Históricos

1. **Detalle de dedicación de esfuerzo** para realizar el trabajo (en Personas Horas), distinguiendo la contribución personal de cada miembro del grupo y el esfuerzo invertido por cada tarea realizada. Esto incluye todas las tareas para poder construir la aplicación, documentarla y elaborar los trabajos de Ing. de SW y Concurrentes.

Información Adicional:

1. Adjuntar la descripción del proyecto de concurrentes, ya sea el del año actual o el pasado en el que se haya basado el trabajo.
2. **Lecciones aprendidas** durante la elaboración del práctico y errores cometidos.