Proyecto de Ingeniería de Software I - 2013

FaMAF - UNC

Presentación del problema

Este proyecto consiste en el desarrollo web de un juego, basado en el clásico de papel y lápiz "Batalla Naval". En este caso se usará una variante del original, en el que se agregan algunas características y reglas adicionales. Se juega a través de la red, de a dos o más jugadores, cada uno de los cuales cuenta con una flota inicial. Los jugadores alternan sus movimientos en turnos, y el objetivo es destruir todos los barcos del oponente (u oponentes), antes de perder los propios.

Desarrollo del proyecto

El proyecto se va a dividir en 3 grandes iteraciones, donde cada una está compuesta por todas las etapas del desarrollo cascada:

- Análisis de requerimientos y especificación
- Arquitectura y Diseño
- Implementación
- Testing

Por cada iteración se presentará un enunciado, detallando las características y funcionalidad esperadas al final de la misma.

El objetivo es que al final de cada etapa se presente un sistema completo y funcional, que será la base para la siguiente etapa. De esta manera (y de forma incremental), se deberá completar el proyecto al cierre de la última entrega.

Fechas de entrega y defensa

- Entrega 1: Lunes 16/09 / Defensa: Martes 17/09
- Entrega 2: Martes 15/10 / Defensa: Jueves 17/10
- Entrega 3: Martes 19/11 / Defensa: Jueves 21/11

Qué se espera en cada entrega?

Análisis de requerimientos y especificación

Cada grupo deberá proveer, como mínimo, el documento de requerimientos siguiendo la estructura descripta en las figuras 3.13 y 3.14 de las páginas 126 y 127 de "An integrated approach to software engineering" (Jalote 2005), y la planificación de las distintas etapas del desarrollo.

El documento de requerimientos, junto con los diagramas de clases de análisis, las consideraciones y demás comentarios de soporte que consideren necesarios deberán estar almacenados en la wiki del Trac de cada grupo (y por supuesto en el repositorio svn de cada grupo).

Además se sugiere, aunque no es requerido, registrar por cada milestone algunas tareas propias de cada etapa en forma de tickets (los que durante la evolución del proyecto posiblemente reciban comentarios y/o cambios de estado).

Arquitectura y Diseño

Cada grupo deberá proveer, como mínimo, documentos conteniendo descripciones de la arquitectura de su sistema, según la vista de componentes y conectores, y el diagrama de clases detallado del diseño de la aplicación.

Además, deberá proveerse un diagrama de secuencia por cada caso de uso especificado como parte del documento de requerimientos. Cada uno de estos diagramas de secuencia debe mostrar cómo, a través de la interacción entre instancias de las clases descriptas en el diseño, se consigue la funcionalidad asociada al caso de uso correspondiente.

Estos documentos, junto con las consideraciones y demás comentarios de soporte que consideren necesarios, y versiones preliminares de estos documentos, deberán estar almacenadas en la wiki del Trac de cada grupo (y en el svn).

Implementación

Cada grupo deberá proveer la codificación terminada de la iteración en curso: el programa funcionando con instrucciones de dónde y cómo ejecutarlo.

Todo el código debe estar al día en el repositorio svn, y cada alumno debe trabajar con su propio usuario, para que los docentes podamos tener visibilidad del trabajo individual de cada uno.

Testing

Se espera que escriban tests de unidad para cada vista y para cada modelo (al menos para la

ejecución exitosa), y tests de sistema (al menos un test por cada caso de uso identificado en las etapas iniciales del proyecto).

Otras observaciones

El enunciado es una descripción general de lo que se espera, y como en (casi) todo desarrollo puede resultar ambiguo, inconsistente, incompleto y/o ambicioso. Durante las distintas etapas deberán interactuar con el usuario "final", que para el caso será el docente que se designe a cargo del grupo. Este docente será además el principal responsable de evaluar cada etapa.

Herramientas a utilizar

La administración (management) del proyecto también será evaluada, y es indispensable para la aprobación del mismo. Es decir que el proceso involucra tanto actividades propias del desarrollo como actividades referidas al manejo del proyecto (planificación, control, etc). Para ello se brindarán las siguientes herramientas (por grupo):

Trac: permite el mantenimiento de documentos en general -wiki-, además de poseer un sistema de tickets y milestones -i.e., planning- e integrarse con svn
Subversion: sistema de control de versiones
Django: framework web sobre Python para la implementación del proyecto

Cabe mencionar nuevamente que la buena utilización de las herramientas, y el respetar los formatos y plantillas de los distintos documentos que se vayan requiriendo serán esenciales para la aprobación del proyecto.

Vale remarcar que toda la documentación del proyecto debe estar en el Trac, para lo cual será importante mantener una organización adecuada del mismo, aprovechando las ventajas que éste ofrece.

Organización interna del grupo

Dentro de cada grupo se debe distribuir los siguientes roles entre los integrantes:

- Responsable de requerimientos
- Responsable de diseño y arquitectura
- Responsable de codificación
- Responsable de testing

¿Qué implica ser el responsable de una etapa? Significa que esa persona es la encargada de llevar registro de las tareas pendientes, completadas y/o por corregir de la etapa en cuestión. Es quien tiene la responsabilidad de coordinar la etapa y asegurar que esas tareas se realicen en tiempo y forma. El docente (el "cliente") va a interactuar principalmente con el responsable de una etapa dada cuando haya que coordinar tareas o correcciones relacionadas a dicha etapa.

Además, es la persona que tiene la última palabra ante un caso de duda o en el que haya que tomar una decisión y no se llegue a consenso dentro del grupo.

¿Qué **NO** implica ser responsable? Ser responsable no significa que a ese integrante se le vaya a exigir algo distinto durante la evaluación o defensa de una entrega. Todos los miembros de un grupo deben trabajar, conocer y ser capaces de defender y/o corregir cualquiera de las partes que conforman una entrega.

El responsable de cada etapa no cambia a lo largo del cuatrimestre, es decir que será fijo para las 3 iteraciones del proyecto.

Enunciado Iteración I

En una partida de batalla naval pueden participar tantos jugadores como se quiera. Cada uno de los jugadores cuenta con una tablero o mapa, compuesto por una cuadrícula de 9x9 celdas. El tablero tiene identificada cada celda por una letra y un número, donde la primera indica la columna y la segunda, la fila.

En un mapa se debe permitir ubicar una flota, que puede estar conformada por 0 o más de los siguientes barcos:

- Portaaviones (5 celdas)
- Acorazado (4 celdas)
- Fragata (3 celdas)
- Submarino (3 celdas)
- Bote de patrulla (2 celdas)

Los barcos son posicionados en cualquier lugar del tablero, sin superposiciones de los mismos, y de manera que cada unidad ocupe celdas continuas lineales, vertical u horizontalmente (es decir, las unidades de N celdas ocupan regiones de 1xN o Nx1 celdas), pero nunca en diagonal.

Los barcos pueden estar sobre los bordes del tablero, pero no pueden tocarse entre sí. Por ejemplo, la figura de la izquierda es un layout correcto, pero la de la derecha no:

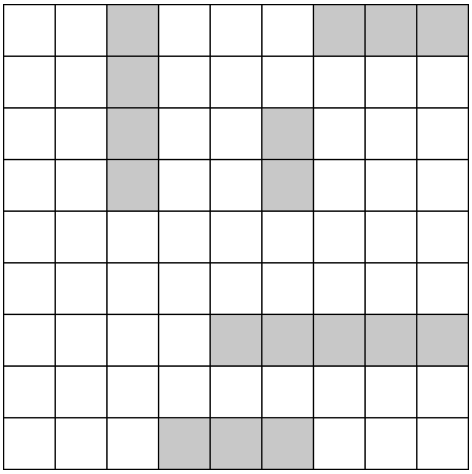

Figura 1: layout correcto

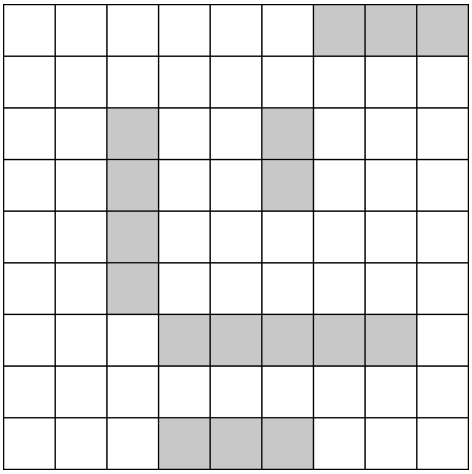

Figura 2: layout incorrecto

En esta primera iteración el objetivo es que, vía la interfaz de administración de Django, se pueda:

- crear mapas
- para un mapa particular, agregar barcos cumpliendo las restricciones descriptas

También se debe proveer una manera de acceder públicamente a una página (fuera del admin) que muestre la distribución de los barcos para un mapa dado.