GESTIÓN DE LA CONFIGURAción del software

Asignatura: Ampliación Ingeniería del Software. Curso: 2019-2020.

Javier Montes Hermosilla

Daniel Rodríguez Borreguero

Celia Rodríguez Mata

Alberto Millán Roldán Sierra

**Índice**

**Introducción.**

-Propósito.

-Alcance.

-Definiciones.

-Referencias.

**Gestión de la GCS.**

-Organización.

-Responsabilidades.

-Políticas, directivas y procedimientos aplicables.

**Actividades de la GCS.**

-Descripción de las tareas de GCS.

-Control de la interfaz software.

-Proceso del control de cambios

**Calendario de la GCS.**

**Recursos de la GCS.**

-Hardware

-Software

-Personal

-Entrenamiento necesario

**Mantenimiento del plan de la GCS.**

**Introducción.**

**-Propósito.**

Mediante este proyecto se pretende satisfacer las necesidades del cliente mediante la adición a la App Tic Tac Toe de un sistema de estadísticas para que los usuarios puedan ver sus estadísticas y las del resto de usuarios.

**-Alcance.**

Este proyecto software consiste en la modificación del ya existente Tic Tac Toe añadiéndole una nueva funcionalidad.

Los elementos de configuración de los que haremos uso son, entre otros, nuevo código fuente, además de un nuevo modelo funcional que permita al jugador ver las estadísticas y al servidor almacenar las estadísticas de los jugadores al acabar la partida y la especificación de requisitos también varía, al añadir como requisito funcional que el usuario debe ver sus estadísticas y el servidor almacenarlas.

No requerimos de nuevo software, dado que el actual ya nos permite, modificándolo, satisfacer las necesidades del cliente.

El coste económico del proyecto será nulo, ya que es un proyecto universitario, pero sí se emplearán un numero de horas, repartidas entre los 59 días de duración del proyecto. La fecha de entrega del proyecto es el día 20 de abril y la de inicio es el día 21 de febrero.

**-Definiciones. (Preguntar qué quiere)**

ACC: Autoridad de control de cambios.

Elemento de configuración: Agregación de hardware y/o software diseñada para la gestión de configuración y que se trata como una única entidad en el proceso de gestión de configuración

**-Referencias.**

Documento del Plan de GCS IEEE Std. 828-2005: <https://standards.ieee.org/standard/828-2005.html>

Configuración operativa del sistema software IEEE Std. 610.12-1990 <https://standards.ieee.org/standard/610_12-1990.html>

**Gestión de la GCS.**

**-Organización.**

La organización encargada de este proyecto es el Grupo 4 de la asignatura Ampliación de Ingeniería de Software, del Grado en Ingeniería Informática en la URJC de Vicálvaro.

Hemos creado una única unidad organizativa técnica y de gestión, encargada de realizar tantos los cambios en el software como la documentación. La organización esta compuesta por Celia Rodríguez Mata, Javier Montes Hermosilla, Daniel Rodríguez Borreguero y Alberto Millán Roldan Sierra.

**-Responsabilidades.**

En este caso al tratarse de un proyecto con muy pocos cambios, toda la unidad organizativa ha colaborado en la realización del proyecto, a través de videoconferencias y realizando las tareas secuencialmente.

**-Políticas, directivas y procedimientos aplicables.**

Para este proyecto utilizaremos el estándar de Gestión de Configuración del Software IEEE Std. 828-2005.

**Actividades de la GCS.**

**-Descripción de las tareas de GCS.**

**I. Identificación de la configuración.**

**Identificación de los EC y las LB:** Especificación de requisitos, modelo funcional del sistema, código fuente, plan de gestión de configuración del software. No aplica LB.

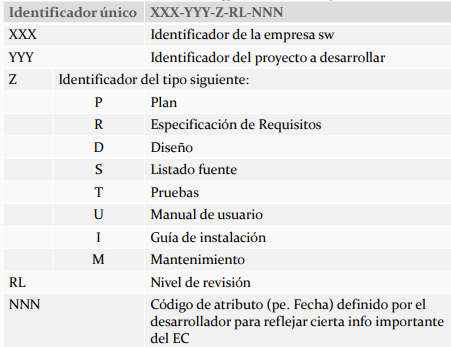
**Nombrado de los EC:**

* **Especificación de requisitos (Nueva):** URJC-TTT-R-1-2502
* **Modelo funcional del sistema (Nueva):** URJC-TTT-D-1-2502
* **Código fuente (Nuevo):** URJC-TTT-S-1-2502
* **PGC Software (Nuevo):** URJC-TTT-P-1-2502

**Obtención de los EC:**

En este caso no hemos recibido documentación previa del software a modificar, por lo que no hemos podido recuperar antiguas versiones de la documentación de los elementos controlados, exclusivamente código fuente.

**Nombrado de identificación (Leyenda):**



**II. Identificación de la necesidad del cambio y documentación.**

**Preparación de la documentación:** Hemos realizado tres directorios, Evolución del Software, Gestión de la Configuración del Software y Gestión de Proyecto, en esos tres directorios debe quedar reflejada documentación suficiente para realizar este proyecto.

**Completar la documentación**: A lo largo del proyecto se irá completando la documentación de manera secuencial hasta la finalización del proyecto, donde esta documentación quedará exclusivamente en modo lectura y no será modificada.

**Revisión:** Además de hacer revisiones intermedias antes de realizar cada commit en el GitHub, al finalizar el proyecto, la supervisora Paloma Cáceres, hará la revisión final, y evaluará de acuerdo a esta.

**III. Análisis y evaluación de la petición.**

**Evaluación del impacto del cambio:** El cambio en el software permitirá a los usuarios consultar sus estadísticas.

**Creación de procedimientos para la revisión del análisis de resultados:** Se generarán una serie de test en la segunda parte de la asignatura que probarán el correcto funcionamiento del software y la correcta implementación.

**IV. Aprobación o desaprobación.**

Paloma Cáceres es la encargada de aprobar este documento y por consiguiente los cambios que se quieren realizar en este proyecto.

**-Control de la interfaz software.**

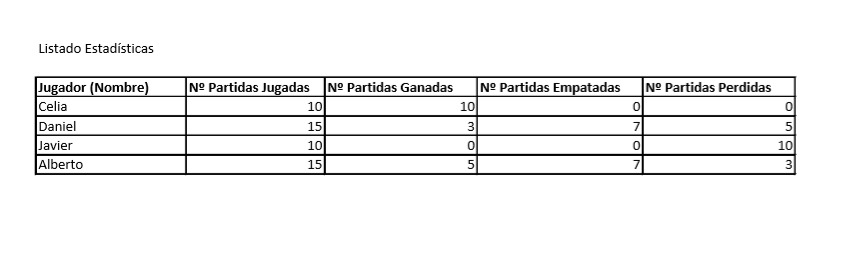
**Index.html:** Se genera un nuevo html en el cual el jugador podrá elegir entre jugar una partida o consultar las estadísticas del juego.

**Estadisticas.html:** Se genera este html para poder hacer display de todas las estadísticas almacenadas.

**TicTacToeHandler.java:** En el método **handleTextMessage**, más específicamente en el case MARK, hay que añadir dos comprobaciones, game.checkDraw(msg.data.PlayerId), para saber si la partida esta empatada, y por lo tanto la partida finaliza y manda las estadísticas al listado de estadísticas como un empate para cada jugador y una partida jugada más para cada jugador. La segunda comprobación es game.checkWinner(msg.data.PlayerId), que comprueba si el usuario ha ganado, en caso de que sea TRUE, ese jugador recibirá un +1 en partidas ganadas y el rival, por consiguiente, un +1 en partidas perdidas, además los dos aumentan en 1 sus partidas jugadas.

**EstadisticasController.java:** Esta clase es necesaria para hacer el Launch de estadísticas.html a través de una anotación de RequestMapping.

**EstadisticasRepository.java:** Esta es una clase repositorio necesaria para pasar el listado de las estadísticas al html, esta conectado con EstadisticasController.java y con Estadisticas.html.

**Estadisticas.java:** En esta clase se define el tipo estadística y se almacena el listado con todos los atributos que deseamos conocer, como se aprecia en el ejemplo.

**-Control de la interfaz hardware.**

No aplica.

**-Control del subcontratista/vendedor.**

No aplica.

**-Proceso del control de cambios (Se ha decidido junto con la profesora dejar este proceso de control de cambios).**

**“**Añadir estructura de datos”

**Necesidad de un cambio**

Para poder consultar las estadísticas de las partidas jugadas debemos añadir una estructura de datos para almacenar la información de cada partida y poder así realizar las estadísticas requeridas.

**Petición de cambio**

Requerimos implantar la estructura de datos para poder estructurar correctamente la información y poder almacenarla de una forma clara.

**Evaluación**

Hemos comprobado que, al añadir nuestra Estructura de Datos, todos los requerimientos del cliente se cumplen. La información se muestra correctamente y nuestra función de “Consultar Estadísticas” muestra tal y como se han desarrollado las partidas jugadas.

**Generación de la petición de cambio**

Una vez analizado y evaluado si el cambio era posible y, además, cumplía las expectativas del cliente hemos podido afirmar que era lo mejor añadir está estructura de datos al software.

**Decisión de ACC (autoridad de control de cambios)**

Al ser necesario implementar esta funcionalidad, el cambio ha sido aprobado y aceptado por la ACC.

**Identificación personalizada EC (elementos configuración)**

Código fuente, modelo funcional, **e**specificación de requisitos

**Dar de baja los EC**

Código fuente, modelo funcional, especificación de requisitos

**Realizar el cambio**

Código Fuente implementación clase estadística

Modelo funcional ahora soporta estructura de datos para almacenar estadísticas

Especificación de requisitos del cliente han cambiado, ahora requiere poder consultar las estadísticas del juego.

**Auditoría**

Se comprueba un correcto funcionamiento de la aplicación

**Dar de alta los EC modificados**

Especificación de requisitos (Nueva): URJC-TTT-R-1-2502

Modelo funcional del sistema (Nueva): URJC-TTT-D-1-2502

Código fuente (Nuevo): URJC-TTT-S-1-2502

**Crear una LB para hacer las pruebas**

No aplica.

**Realizar pruebas y actividad de garantía cal**

Realizadas semanalmente para comprobar su funcionamiento correcto.

**Promoción cambios para incluir en +versión (revisión)**

Se realizará la revisión cuando el numero de estadísticas sea muy elevado y poder comprobar el funcionamiento de la estructura.

**Reconstrucción de la versión adecuada**

Se aplicarán siguiendo las mismas directrices hasta ahora.

**Revisión de los cambios (auditoría en los EC)**

Volvemos a realizar revisiones, esta vez diarias, tras la evaluación final del cambio.

**Distribución de la nueva versión.**

Se renombrarán los EC en concordancia con la fecha y numero de versión que le corresponde.

**“**Modificación TIC TAC TOE HANDLER – Método handleTextMessage”

**Necesidad de un cambio**

Para poder comprobar el resultado de la partida necesitamos añadir en el método handleTextMessage(), un checkDraw() y checkWinner() que comprueban si el jugador empata o gana respectivamente.

**Petición de cambio**

Requerimos modificar el código para que nuestro software pueda realizar correctamente la función de comprobación de los resultados de la partida y posteriormente mostrar las estadísticas correctas.

**Evaluación**

Hemos comprobado que al modificar nuestro código el software implementa correctamente las partidas empatadas y ganadas.

**Generación de la petición de cambio**

Una vez analizado y evaluado si el cambio era posible y, además, cumplía las expectativas del cliente hemos podido comprobar que la modificación de estos métodos es necesaria para mostrar el resultado de la partida.

**Decisión de ACC (autoridad de control de cambios)**

Al ser necesario implementar esta funcionalidad, el cambio ha sido aprobado y aceptado por la ACC.

**Identificación personalizada EC (elementos configuración)**

Código fuente, especificación de requisitos

**Dar de baja los EC**

Código fuente, **e**specificación de requisitos

**Realizar el cambio**

Código Fuente modificación del método handleTextMessage(), de la clase TIC TAC TOE HANDLER. Dentro del método handleTextMessage(), en el apartado switch, tenemos este caso MARK en el que comprobamos el resultado de la partida con los métodos checkWinner() y checkDraw().

Especificación de requisitos del cliente han cambiado, ahora requiere poder consultar las estadísticas del juego y habiendo comprobado el resultado obtenido en la partida.

**Auditoría**

Se comprueba un correcto funcionamiento de la aplicación

**Dar de alta los EC modificados**

Especificación de requisitos (Nueva): URJC-TTT-R-1-2502

Código fuente (Nuevo): URJC-TTT-S-1-2502

**Crear una LB para hacer las pruebas**

No aplica.

**Realizar pruebas y actividad de garantía cal**

Realizadas semanalmente para comprobar su funcionamiento correcto.

**Promoción cambios para incluir en +versión (revisión)**

Se realizará la revisión tras implementar las modificaciones y si se comprueba algún error en el funcionamiento.

**Reconstrucción de la versión adecuada**

Se aplicarán siguiendo las mismas directrices hasta ahora.

**Revisión de los cambios (auditoría en los EC)**

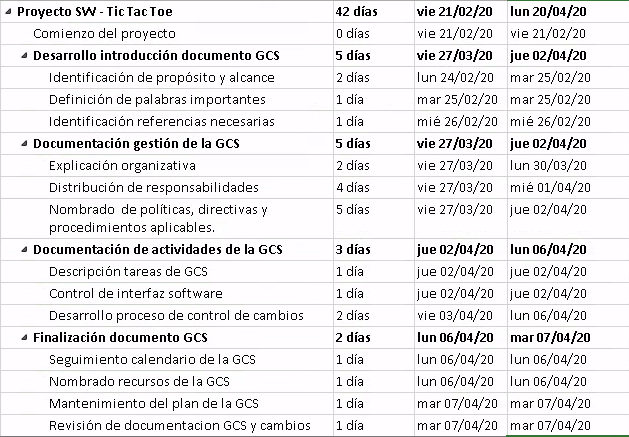
Volvemos a realizar revisiones, esta vez semanalmente incluso si no se comprobaran errores de funcionamiento.

**Distribución de la nueva versión.**

Se renombrarán los EC en concordancia con la fecha y numero de versión que le corresponde.

**Calendario de la GCS.**

Aquí vemos una versión simplificada de Microsoft Project, aunque el calendario de la GCS está mejor reflejado en el documento Gestión de Proyecto Software, junto con Diagramas de Gantt y redes de precedencia (información análoga a PERT).



**Recursos de la GCS.**

**-Hardware**

Ordenador portátil: HP ProBook 640 G1.

Ordenador portátil: MacBook Pro 2011.

**-Software**

Aula Virtual URJC: Plataforma accedida a través de Google Chrome.

Eclipse IDE for Enterprise Java Developers - 2019-12: Version 2019-12 (4.14.0)

LonelyScreen Airplay Reciever: Software para duplicación de pantalla de iPad.

Star UML: Programa para realizar esquemas UML.

Discord.gg: Necesario para la comunicación y puesta en común del trabajo.

Microsoft Project 2016: Herramienta que permite realizar la planificación del proyecto.

Git: Repositorio online que permite guardar los cambios y actualizar las versiones.

Microsoft Word 2016: Para realizar los documentos, licencia cedida por la URJC.

Microsoft Teams: Para la realización de tutorías debido al Covid-19. Licencia cedida por la URJC.

**-Personal**

Javier Montes Hermosilla

Daniel Rodríguez Borreguero

Celia Rodríguez Mata

Alberto Millán Roldán Sierra.

**-Entrenamiento necesario**

Para realizar las tareas propias de GCS y la gestión del proyecto es necesario la lectura y el estudio de los contenidos de la asignatura.

**Mantenimiento del plan de la GCS.**

Test: Se harán test con herramientas como Mockito, Selenium y librerías JUnit.

Reuniones por Discord: Para revisar periódicamente el software y la documentación.