

Proyecto Software - Práctica 6

Planificación

Introducción

A lo largo de las diferentes sesiones de teoría hemos ido viendo diferentes aspectos que se deben tener en cuenta a la hora de hacer la planificación de un proyecto. Así mismo, hemos visto que la planificación debe ser revisada periódicamente al objeto de ir modificando de acuerdo con la evolución del proyecto.


Por otro lado, en vuestro proyecto habéis tenido que hacer una planificación inicial que supone el punto de partida para la organización y gestión del mismo. Esta planificación ha sido incluida y revisada en la primera entrega del proyecto, y deberá ser actualizada en la entrega final en la que se espera encontrar la planificación inicial y su revisión al final del proyecto (junto con un análisis de las discrepancias). Para todo ello, no os hemos dado ninguna referencia de herramienta concreta a utilizar, más allá de que un diagrama de Gantt puede ser uno de los soportes gráficos más utilizados y fáciles de comprender.

En esta práctica vamos a profundizar en técnicas y herramientas de planificación que, si bien ya no tiene sentido aplicar a vuestro proyecto porque está cerca de ser completado, es útil que conozcáis cuando tengáis que planificar cualquier proyecto con una mínima complejidad, cierto número de tareas interdependientes y recursos limitados para abordarlas.

Esta práctica debe realizarse de manera individual y entregarse al final de la sesión a través de Moodle. Su evaluación positiva supondrá 0,1 puntos directos a sumar a la nota que se obtenga entre trabajo y examen.

Herramientas

[TaskJuggler](#) es una herramienta de gestión de proyectos libre y de código abierto. A diferencia de muchas otras alternativas de pago, como p.ej., Microsoft Project, su interfaz de usuario principal no es un editor gráfico para diagramas de Gantt, sino que permite definir un proyecto en cualquier editor de texto en términos de tareas, recursos, restricciones, dependencias etc. utilizando un lenguaje declarativo y luego, a partir de eso, calcula un plan optimizado para llevar a cabo esas tareas, con esos recursos y bajo esas restricciones y dependencias en el menor



tiempo posible. Este plan se puede exportar a distintos formatos, incluyendo un diagrama de Gantt.

Como guía para esta práctica, se ha desarrollado una versión bastante simple de la planificación de un proyecto consistente en un juego que tiene un backend, que da soporte a la lógica del mismo, y un frontend web.

Fundamentos de TaskJuggler

El trabajo con TaskJuggler se basa en la especificación de la planificación del proyecto a través de un lenguaje propio. De ese modo, mediante un fichero en texto plano, y con extensión .tjp, se detallan recursos, tareas, incidencias, costes, etc. Una vez introducida toda esta información en el fichero, la herramienta permite generar informes sobre el proyecto y su planificación.

Los principales elementos de TaskJuggler que se van a trabajar en esta práctica son:

- **Recursos:** Son las personas que van a aportar esfuerzo en el desarrollo del proyecto. Se definen usando la instrucción “resource” seguida de una declaración de la variable con la que se referencias a lo largo de todo el código (en este ejemplo “boss”), un nombre de identificación en la realidad (que tiene que ir entre comillas) y, ya entre llaves, una serie de variables de caracterización del recurso. En algunos casos, estas variables pueden reescribir variables globales. Para todo el proyecto se ha fijado un coste diario por defecto (“rate”) de 145,2 €, mientras que para el recurso “boss” se ha fijado en 160 €.

```
resource boss "Liz Bullock" {  
    email "iiii@unizar.es"  
    rate 160  
}
```

- **Control de costes:** TaskJuggler permite ir controlando los costes y los ingresos del proyecto y en cada tarea de los mismos. Para ello se tienen que definir una serie de variables de tipo “account”, que pueden ser la suma de otras que servirán para el control de partes del proyecto. En el ejemplo se ha establecido una cuenta para costes (“cost”) y otra para ingresos (“rev”). La de costes, además, se ha dividido en los costes de gestión, los costes de desarrollo del backend y los costes de desarrollo del frontend. Finalmente, se ha establecido que el seguimiento económico del proyecto (“balance”) se fijará como ingresos (“rev”) menos costes (“cost”).

```
account cost "Project Cost" {  
    account manag "Management"  
    account wFront "Web Front-End"  
    account back "Back-End"  
}  
account rev "Payments"
```

balance cost rev

- **Tareas:** Son las diferentes actividades en las que se va a descomponer el proyecto. Se

van estableciendo de manera anidada, siendo la primera de ellas el propio proyecto. Se definen mediante la instrucción “task” seguida de una declaración de la variable con la que lo manejaremos a lo largo de todo el código (en este ejemplo “tback”), un nombre de identificación en la realidad (que tiene que ir entre comillas) y, ya entre llaves, una serie de variables de caracterización de la tarea y su estado, así como las subtareas que se fijen. En el este ejemplo se indica que los costes de esta tarea se asignarán a la cuenta “back” (con la instrucción “chargeset”), se ha fijado un responsable, se establece que no se puede empezar la tarea hasta que no haya terminado la tarea “tgdesign”, y se fija una subtarea (“tBEd”). Para esta subtarea se fija un esfuerzo de 5 días y se asigna su desarrollo al recurso “dev1”. Cuando una tarea puede ser desarrollada indistintamente por más de un recurso, éstos se incluyen en líneas adicionales o separándolos por comas.

```
task tback "Back End" {
    chargeset back
    responsable dev1
    depends WMGame.tgdesign
    task tBEd "Back End General Design" {
        effort 5d
        allocate dev1
    }
}
```

- **Hitos:** Son momentos importantes del proyecto que se deben gestionar. TaskJuggler no los maneja de forma explícita por lo que hay que especificarlos como si fueran tareas sin duración. Para ello, no se les asigna ningún esfuerzo (no se usa la instrucción “effort”). Sin embargo, es necesario fijar un momento de inicio o de fin para poder programarla. En este ejemplo se utiliza la macro “\${projectstart}” que lo que hace es buscar en el proyecto la fecha de inicio que se ha fijado. El uso de este tipo de macros permite manejar algunas variables de configuración del proyecto por referencia a las mismas. En este caso, se han fijado dos hitos: uno al inicio del proyecto y otro a la finalización de la tarea “ttest” (en el proyecto de ejemplo se ha vinculado la tarea “ttest” a los test de aceptación). Para cada uno de ellos se ha establecido una asignación económica que se asocia a los pagos por parte del cliente cuando se logren estos hitos. Estos pagos se asignan a la cuenta “rev” mediante la instrucción “chargeset”. Como ya estaba hecha una asignación previa, es necesario hacer una “limpieza” de esa asignación mediante la instrucción “purge”.

```
task deliveries "Milestones" {
```

```

    purge chargeset
    chargeset rev
    task start "Project start" {
      start ${projectstart}
      charge 13000.0 onstart
    }
    task done "Ship Product to Customer" {
      depends WMGame.ttest
      charge 12000.0 onstart
    }
  }
}

```

- **Seguimiento del proyecto:** TaskJuggler define como escenarios (“scenario”) los diferentes momentos de control que se pueden establecer dentro de un proyecto. De este modo, se define un escenario a nivel general para la planificación inicial del proyecto, y diferentes subescenarios para determinar momentos de control con el proyecto en marcha. Para establecer un cambio en la planificación asignada a uno de estos momentos de control, es necesario llevar a cabo una mención explícita al mismo. Así, por ejemplo, si dentro de una tarea se indica “effort 12d”, se está fijando que el esfuerzo necesario es de 12 días. Si la posterior realización del proyecto ha requerido, por ejemplo, tan solo 11 días, se podrá indicar añadiendo previamente la referencia al escenario que se está detallando. Por ejemplo, de este modo: “actual:effort 11d”.

```

scenario plan "Plan" {
  scenario actual "Actual"
}

```

TaskJuggler ofrece muchas más posibilidades y herramientas que podrás explorar a través de la web del proyecto.

Tareas

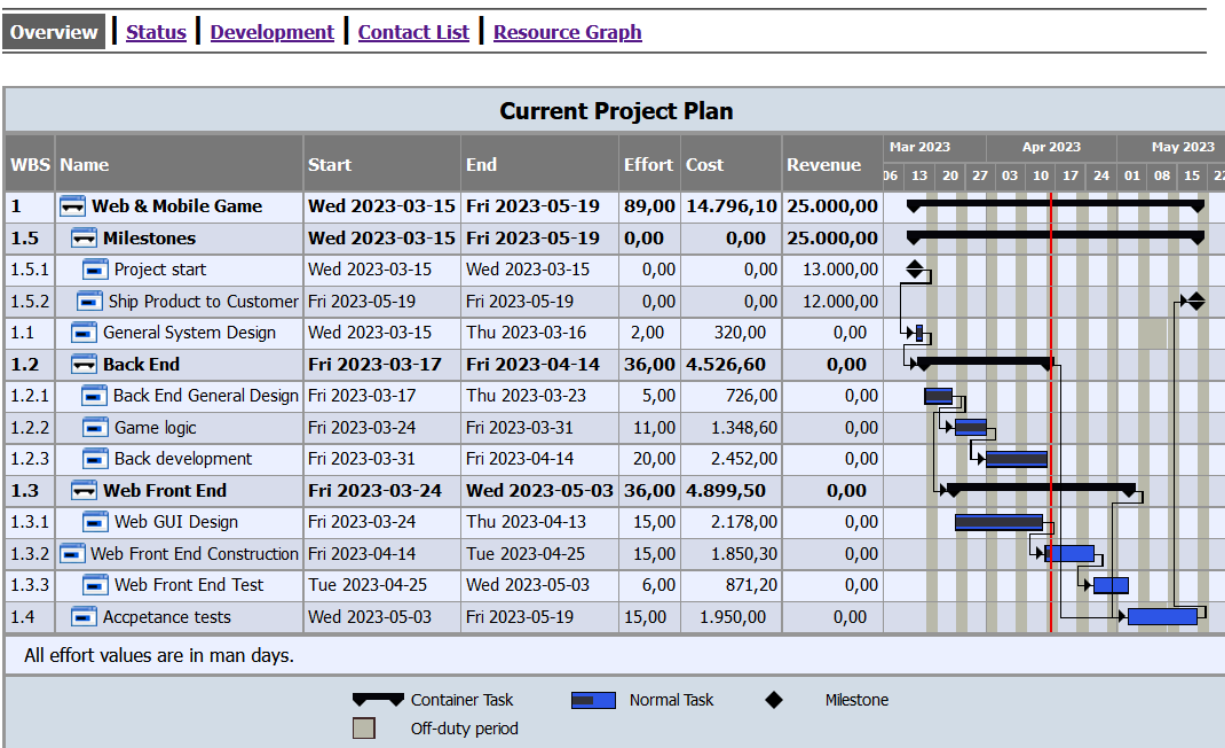
Instalación puesta en marcha de la herramienta

Puedes instalar la aplicación en tu ordenador siguiendo las instrucciones que puedes encontrar en la web de [TaskJuggler](#). En Linux esto es moderadamente rápido, pero en Windows es un proceso un poco largo que necesita, entre otras cosas, la instalación de [ruby](#), que es el lenguaje de programación en el que está desarrollado.

Dado que tan solo dispones de dos horas para hacer esta práctica, hemos preparado un repositorio en GitHub configurado con GitHub Actions para que se ejecute TaskJuggler allí. De este modo, haciendo un fork del repo y clonando luego ese fork en tu equipo, podrás hacer que

sea GitHub Actions el que ejecute la herramienta cada vez que hagas un push, generando los resultados en un fichero results.zip (que incluirá todos los reports que genera el TaskJuggler). Ese fichero zip lo puedes descargar desde la pestaña de Actions de GitHub (tal y como se explicó en la práctica 3). El repositorio que tenéis que “forkear” para luego clonar en vuestros equipos es este: <https://github.com/UNIZAR-30226-PROYS/taskjuggler>.

Accounting Project Web Game



En ese repo tienes también disponible el fichero con el ejemplo (extensión .tjp) que vas a tener que modificar. Esta planificación genera una serie de informes:

- **Overview:** Presenta una visión general y resumida del proyecto en su estado actual (a la fecha definida para el informe).
- **Status:** Presenta detalle del estado del proyecto a la fecha de generación del informe.
- **Development:** Presenta detalles de qué recurso ha llevado a cabo cada una de las tareas y cuál es la asignación prevista hasta final de proyecto.
- **Contact list:** Presenta detalle de los recursos y cuándo has estado trabajando en el proyecto, así como cuándo está previsto que vuelvan a estarlo.
- **Resource Graph:** Presenta detalle de en qué proyecto ha estado asignado cada recurso y a cuáles está previsto asignarlos hasta final de proyecto.

Modificaciones sobre las planificación

Sobre la base de la planificación anterior, debes introducir las siguientes modificaciones:

- Añade dos tareas más de primer nivel. Una de ellas para llevar a cabo el desarrollo de la versión móvil del frontend del juego y otra para dar soporte al despliegue en la nube del back end. Deberás incluir subtareas vinculadas con los diferentes trabajos a desarrollar en la versión móvil, así como para la evaluación y decisión de la nube a usar, y el despliegue sobre la misma. Establece las dependencias con otras tareas que consideres más apropiadas. Así mismo, fija las duraciones de las subtareas que creas que serían las razonables.
- Fija, como modificación a la planificación inicial, el inicio de las pruebas de aceptación para que sea, como pronto, el 1 de junio de 2024. Tendrás que consultar [la sintaxis de task](#) para averiguar cómo.
- Modifica la fecha del informe para que se ajuste al día en el que estás desarrollando la práctica (dale un vistazo a la definición de “project” en el fichero .tjp y averigua lo que tienes que cambiar), y añade una incidencia/anotación en alguna de las tareas del proyecto en esta misma fecha.
- Genera un informe adicional que permita mostrar la planificación inicial del proyecto.
- Inclúyete en el equipo de desarrollo, con una dedicación de 4 horas al día (consulta [la sintaxis de resource](#)), y asígnate a las tareas que consideres que podrías llevar a cabo.

Resultado de la práctica

Como resultado final de la práctica tienes que entregar el fichero .tjp con las modificaciones en la planificación que se ha solicitado, así como el fichero comprimido que genera el repo de GitHub. Esta entrega se tiene que efectuar al final de la sesión de prácticas y vía Moodle.