**Práctica 1**

**Instanciación del Proceso Software Elegido y Creación de un proyecto para ser gestionado aplicando PUD**

1. **Introducción**

El objetivo de esta primera práctica es múltiple cómo puede verse en la ilustración 1.

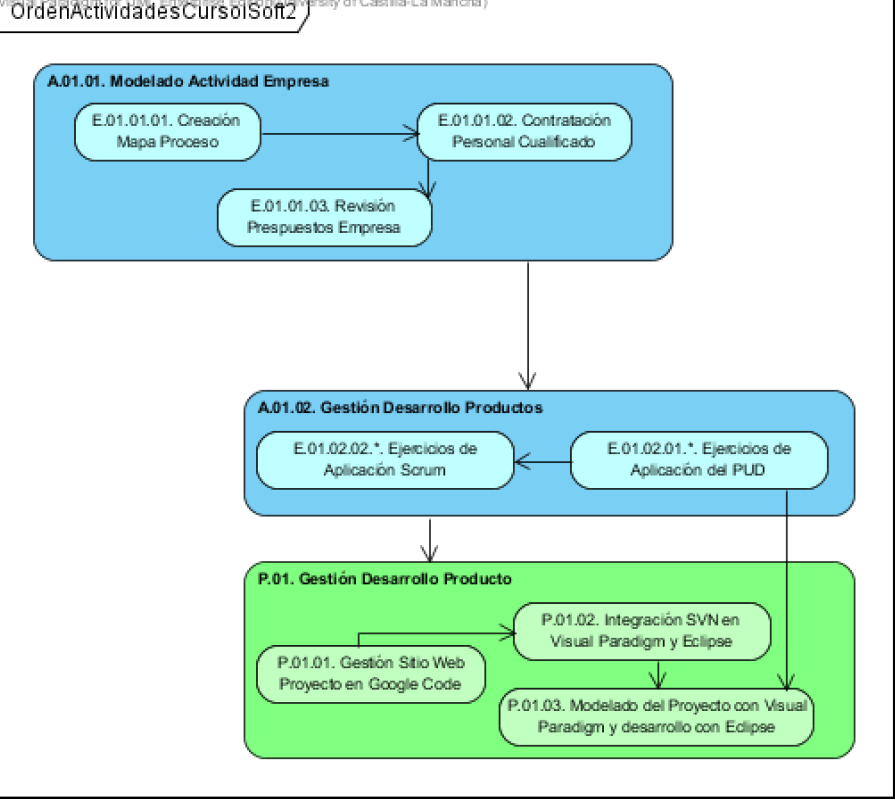


Ilustración 1. Apartados

“Básicamente se tienen dos apartados que se agrupan en tres ejercicios:

1. **Instanciar el Proceso de Desarrollo**, que implica dos tareas:

a. Elección de las herramientas que dan soporte a los procesos elegidos

b. Integración de dichas herramientas

2. **Aplicación del Proceso de Desarrollo a un proyecto específico.**

Básicamente consiste en establecer y configurar un entorno de desarrollo para la ejecución de proyectos en la organización que estáis creando. Una vez configurado el entorno, se trata de simular la ejecución de un proyecto

Consulte el apartado 2 para una descripción detallada del trabajo que hay que realizar en cada apartado y en qué consiste el entregable de la práctica.

En los siguientes subapartados se da una visión global de algunos tipos de herramientas que se pueden utilizar según los tipos de proyectos.

* 1. ***Repositorios para trabajo en grupo de Gestión de Proyectos***

Como se sabe, el desarrollo de software suele ser una actividad hecha en equipos de trabajo.

Por esa razón es fundamental contar con los mecanismos adecuados de comunicación, control y gestión de proyectos. Un concepto interesante es el de **Application Lifecycle Management (ALM)**, cuya descripción se puede encontrar en http://en.wikipedia.org/wiki/Application\_lifecycle\_management. Como puede apreciarse en ese enlace, en el mercado existen numerosas iniciativas tanto de pago (e.g. JIRA) como de uso sin pago (GoogleCode, SourceForge, BitBucket,…).

Se pide buscar las características comunes.

De entre todas las opciones, en las prácticas se optará por Git, que se explicará en clase de prácticas.

Git facilita herramientas para la documentación del proyecto mediante wikis, compartición de archivos y gestión de errores. Se recomienda consultar algunos ejemplos en Git (https://git-scm.com/) utilizando el motor de búsqueda las etiquetas “Academic”, “Java”.

* 1. ***Herramientas CASE***

Las herramientas CASE dan soporte al proceso de desarrollo del software. Este soporte se manifiesta en la posibilidad de gestionar aspectos técnicos del proyecto (como gestión de requisitos, análisis, diseño, generación de código) llamadas herramientas CASE verticales, así como aspectos puramente de gestión (planificación), llamadas herramientas CASE horizontales.

Entre las herramientas CASE verticales más importantes destacan Rational Rose, Enterprise Architect, Visual Paradigm, o Microsoft Visio dentro de las de pago, o bien la herramienta Dia, dentro de las libre uso.

Entre las herramientas CASE verticales se puede encontrar Microsoft Project dentro de las de pago, o la herramienta Gantt Project (http://www.ganttproject.biz/), dentro de las de libre distribución.

* 1. ***Entornos Integrados de Desarrollo***

Los Entornos Integrados de Desarrollo (IDE) permiten automatizar las tareas relativas a la edición, compilación y depurado de código fuente.

Existen numerosas herramientas para las distintas tecnologías de desarrollo: Visual Studio para .Net, o Eclipse, Netbeans, JDeveloper para tecnologías Java,…

Una de las características deseables es la posibilidad de expandir las funcionalidades con múltiples funcionalidades (*plug-ins*) que permitan entre otras cosas la posibilidad gestión de la configuración, o la trazabilidad de los errores. En este sentido Eclipse es un referente en entornos de desarrollo software centrados en Java.

* 1. ***Herramientas para la gestión y construcción de proyectos.***

Estas herramientas permiten gestionar las dependencias entre los módulos del proyecto así como la automatización de las pruebas. Herramientas como *makefile, ant*, o más recientemente *maven,* permiten automatizar numerosas tareas en la construcción del software.

* 1. ***Herramientas para la gestión de la configuración***

Uno de los procesos de gestión más importantes es el de gestión de los cambios y configuración. Las herramientas que soportan este proceso permiten guardar las distintas versiones del software que se va generando.

En la actualidad existen varias alternativas como SVN, CVS, Mercurial o GitHub. “

1. **Trabajo a realizar** 
   1. ***~~Elección y configuración de una herramienta específica: Git~~***

~~El trabajo que se pide realizar para esta práctica consiste en lo siguiente: se trata de recrear en un repositorio de Git la ejecución de los proyectos que desarrollasteis en el trimestre pasado.~~

* 1. ***Integración de Git con Visual Paradigm y con Eclipse***

En lo que respecta a Visual Paradigm, importad los proyectos y configurad en la pestaña gestión del proyecto la fase y la iteración en la que se debe acometer la realización de los componentes (véase ilustración 2).

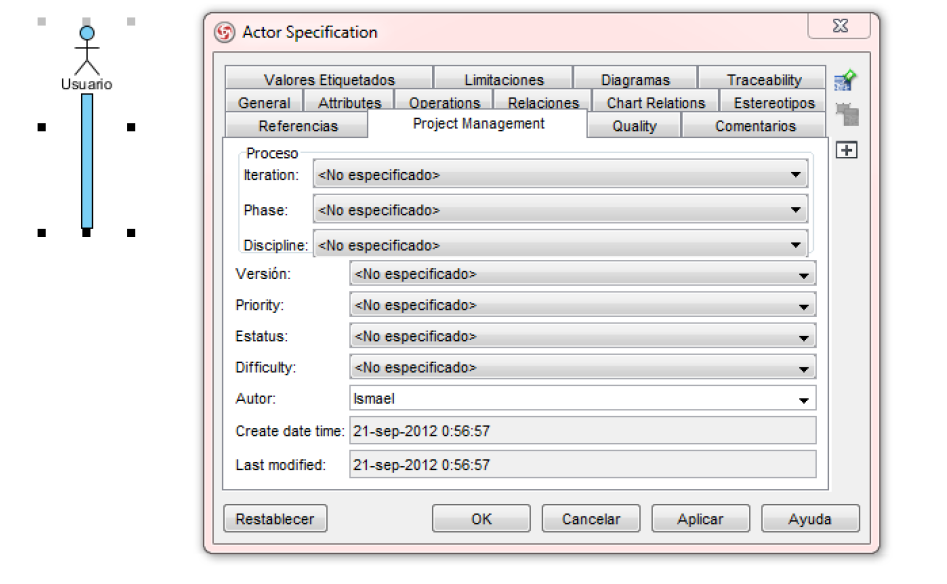


Ilustración 2. Configuración de un proyecto

Para gestionar el control de Versiones, Visual Paradigm, también permite configurar el repositorio en el equipo de trabajo.

Instalar la versión de Eclipse que se corresponda con tu sistemas operativo, instala los plugins de Git, mylyn.

* 1. ***Creación de un proyecto específico que pueda ser desarrollado por la empresa***

El objetivo de este tercer apartado es la simulación de un proyecto usando las herramientas y aplicando el Proceso Unificado de Desarrollo.

No hace falta que sea un proyecto con muchísimos requisitos funcionales, pues el objetivo de la práctica no es el desarrollo en sí, si no que comprendáis el mecanismo de creación y seguimiento de la planificación.

Por otro lado, aunque sería muy interesante, no es necesario que generéis código de momento, si bien lo necesitaréis para las prácticas 4 y 5.

Podéis usar el proyecto del año pasado del trimestre pasado o cualquier otro nuevo, pero siempre con la condición de que se adecúe al tipo de empresa que habéis creado.

Debéis hacer una planificación al estilo de los ejercicios que hemos hecho en clase (bloque E.07.02), con todos los condicionantes que se identificaban con respecto a recursos humanos, arquitecturas, … **pero usando las herramientas correspondientes para desarrollar los distintos modelos.**

Tan importante como la planificación es el seguimiento, por eso es necesario que en aras de mejorar vuestra capacidad de estimación de proyectos, llevéis una imputación de las horas reales de trabajo, para poder comparar lo que habéis previsto con lo que al final ha salido. Recordad lo que hemos hablado en clase: “lo peor” (no “lo malo”) no es hacer una mala estimación inicial, sino el no ser capaz de identificar las causas de la variabilidad entre lo estimado y lo ejecutado. **Pero esto sólo lo podéis realizar si hacéis una imputación de horas durante el proyecto.**

Todo esto debe reflejarse en el Git de vuestro proyecto mediante las correspondientes actualizaciones.