

**UNIVERSIDAD DE CASTILLA-LA MANCHA ESCUELA SUPERIOR DE INFORMÁTICA**

**INGENIERÍA**

**EN INFORMÁTICA**

**PROYECTO FIN DE CARRERA**

*“Línea de productos software: Juegos de mesa”*

Juan Yánez García-Catalán

**Septiembre, 2012**



**UNIVERSIDAD DE CASTILLA-LA MANCHA ESCUELA SUPERIOR DE INFORMÁTICA** (Departamento de Tecnologías y Sistemas de Información)

**PROYECTO FIN DE CARRERA**

*“Línea de productos software: Juegos de mesa”*

Autor: Juan Yánez García-Catalán

Director: Macario Polo Usaola

**Septiembre, 2012**

**TRIBUNAL:**

**Presidente:   
  
Vocal 1:   
  
Vocal 2:**

**Secretario:**

**FECHA DE DEFENSA:**

**CALIFICACIÓN:**

**PRESIDENTE VOCAL 1 VOCAL 2 SECRETARIO**

Fdo.: Fdo.: Fdo.: Fdo.:

## RESUMEN

asd

## ABSTRACT

Y aquí es cuando empiezo a escrbir a ver que pasa.

## ÍNDICE GENERAL

[RESUMEN I](#_Toc332765433)

[ABSTRACT II](#_Toc332765434)

[ÍNDICE GENERAL V](#_Toc332765435)

[ÍNDICE DE FIGURAS VI](#_Toc332765436)

[1. INTRODUCCIÓN 1](#_Toc332765437)

[1.1. Introducción al tema 1](#_Toc332765438)

[1.2. Estructura del documento 4](#_Toc332765439)

[2. OBJETIVOS DEL PROYECTO 7](#_Toc332765440)

[2.1. Objetivo principal 7](#_Toc332765441)

[2.2. Objetivo secundario 7](#_Toc332765442)

[2.3. Herramientas y medios empleados 7](#_Toc332765443)

[2.3.1. Middleware de comunicaciones 7](#_Toc332765444)

[2.3.2. Lenguaje de programación 8](#_Toc332765445)

[2.3.3. Entorno de desarrollo 8](#_Toc332765446)

[2.3.4. Base de datos 9](#_Toc332765447)

[2.3.5. Herramienta de análisis y diseño 9](#_Toc332765448)

[2.3.6. Herramienta de gestión 9](#_Toc332765449)

[3. ANTECEDENTES 11](#_Toc332765450)

[4. MÉTODO DE TRABAJO 13](#_Toc332765451)

[5. RESULTADOS 15](#_Toc332765452)

[6. CONCLUSIONES Y PROPUESTAS 17](#_Toc332765453)

[BIBLIOGRAFÍA 19](#_Toc332765454)

## ÍNDICE DE FIGURAS

[Figura 1.1 Modelo básico de una línea de productos software 3](#_Toc332729064)

## INTRODUCCIÓN

En este capítulo, se introduce la problemática en la que se centra el Proyecto, perfilando una posible solución. Además se explica el contexto donde se engloba el proyecto y las relaciones de éste con el entorno en el que se desarrolla.

### Introducción al tema

Cuando una empresa ofrece un producto software a distintos clientes surge toda la problemática de las versiones y la evolución acompasada del producto. En este escenario, no es raro encontrarse con alguna de las siguientes situaciones (1):

* **Relaciones conflictivas entre los equipos de marketing y de ingeniería** a causa de la incapacidad de los segundos de adecuarse a las solicitudes de nuevas variaciones de producto provenientes del negocio.
* **Coordinación compleja y costosa** de múltiples tareas de desarrollo en paralelo que comparten software común.
* **Código fuente poco robusto**, que resulta difícil de extender con variaciones del nuevo producto propenso al error.

**La producción en serie** (*mass production*) –la capacidad para crear eficientemente múltiples copias del mismo producto– constituyó un gran avance en el mundo de la fabricación. Las líneas de producto llevan años aplicándose en otros entornos industriales, como la fabricación de vehículos en cadenas de montaje: las diferentes versiones de un mismo modelo de coche difieren en ciertas características de su equipamiento (presencia o ausencia de aire acondicionado, elevalunas eléctricos, etc.), lo cual no impide que todos ellos compartan las mismas cadenas de montaje y producción. Más recientemente, las líneas de producto se utilizan también para el desarrollo y construcción de teléfonos móviles que, además del conjunto básico de funcionalidades, ofrecen diferentes tamaños de pantalla, presencia o ausencia de bluetooth, varias resoluciones de la cámara, etc.

Pero crear múltiples copias de un producto software es trivial. Sin embargo, **la personalización en serie** (*mass customization*) –la capacidad para crear eficientemente múltiples variaciones de un producto– es un importante reto tanto en la fabricación de lavadoras como en la venta de un ERP o cualquier otro producto software. La pregunta es cómo se plasma el enfoque de personalización en serie en el desarrollo de productos software.

En la producción de software, el proceso ha venido estando centrado en el producto antes que en la línea de montaje. Las herramientas de desarrollo (IDE) y las metodologías ayudaban a agilizar y sistematizar la creación de un único producto. Sí que existe una inquietud por reutilizar pero en la mayor parte de los casos, la reutilización es oportunista, es decir, surgía la posibilidad de reutilizar a posteriori, no era algo que se supiera positivamente que se iba a poder reutilizar. Por ello, muchos esfuerzos de re-utilización no se amortizaban ya que no terminaba de surgir la oportunidad para poder reutilizarlo.

Esta situación cambia con las líneas de producto. Ahora ya no se quiere producir un único producto, sino una cadena de montaje que gestione eficiente y eficazmente las diferentes variaciones que pueden existir entre los productos. La empresa ya no se centra en un producto para un cliente (por ejemplo, construir un portal para Iberia), sino en un dominio (por ejemplo, construir portales para líneas aéreas). El reto está en delimitar el ámbito de este dominio, identificar las variaciones que se van a soportar, y dotarse de la infraestructura que permita producir el producto a bajo coste pero manteniendo altas cotas de calidad. Es decir, aplicar los principios de la producción en serie también al software. Este enfoque resulta en mejoras tanto en la eficiencia (reducción del time-to-market) como en la eficacia (mejora de la calidad del software).

**Una línea de productos software (*LPS*)** se define por tanto como “*un conjunto de sistemas software, que comparten un conjunto común de características (features), las cuales satisfacen las necesidades específicas de un dominio o segmento particular de mercado, y que se desarrollan a partir de un sistema común de activos base (core assets) de una manera preestablecida*”. (2) Uno de los aspectos distintivos de las LPS frente al desarrollo tradicional es la importancia de la variabilidad a lo largo de todo el proceso de desarrollo: los productos de la línea comparten un **conjunto de características** (*commonalities*) y difieren en determinados **puntos de variación** (*variation points*), que representan la variabilidad entre los productos.

Un aspecto central en el desarrollo de LPS es la división de los procesos de ingeniería: la Ingeniería de Dominio, responsable de desarrollar los elementos comunes a todos los integrantes de la línea y su mecanismo de variabilidad, y la Ingeniería del Producto (o Ingeniería de la Aplicación), cuyos cometidos incluyen desarrollar los productos para clientes concretos, reutilizando los recursos creados en la Ingeniería del Dominio (véase figura 1.1).



Figura 1.1 Modelo básico de una línea de productos software

Además existen dos enfoques para la construcción de los productos de la LPS:

* **Enfoque sustractivo**: Todas las variabilidades se contemplan en el nivel de ingeniería de dominio y, a nivel de producto, se eliminan aquellas que no corresponden al producto concreto que se está desarrollando.
* **Enfoque aditivo:** A nivel de ingeniería de dominio se trabaja únicamente con las *common features*. Luego, a nivel de producto, se le incorporan las variabilidades que correspondan.

### Estructura del documento

El presente documento ha sido redactado siguiendo la normativa del proyecto fin de carrera de la Escuela Superior de Informática de la universidad de Castilla-La-Mancha. Éste consta de los siguientes capítulos:

**Capítulo 1. Introducción**

En este capítulo se ha introducido la problemática en la que se centra este proyecto fin de carrera, dando unas pinceladas de que se pretende resolver. Además se explica el contexto donde se engloba y las relaciones de éste con el entorno en que se desarrolla.

**Capítulo 2. Objetivos**

En este capítulo se hablará de los objetivos que se pretenden conseguir en el desarrollo de este proyecto fin de carrera.

**Capítulo 3. Antecedentes**

En este capítulo se explican las bases matemáticas y el funcionamiento general de los principales sistemas de vigilancia inteligente, haciendo especial hincapié en las capas de segmentación y tracking.

**Capítulo 4. Método de trabajo**

En este capítulo se presenta el método de trabajo seguido para el desarrollo de la Línea de productos software. Se explicará la metodología de desarrollo para el análisis, diseño e implementación de la aplicación y las herramientas necesarias para llevarlo a cabo.

**Capítulo 5. Resultados**

En este capítulo veremos los resultados obtenidos, el juegacooooo.

**Capítulo 6: Conclusiones y propuestas**

En este capítulo lo vamos a petar.

**Bibliografía**

**Anexo I.** Manual de usuario

## OBJETIVOS DEL PROYECTO

En este capítulo hablaremos de los objetivos que se pretenden conseguir en el desarrollo de este proyecto fin de carrera.

### Objetivo principal

El objetivo principal de este PFC es la aplicación del paradigma de desarrollo basado en línea de productos software que, además es un paradigma de desarrollo que empieza a implantarse en muchas empresas. Para ello, se desarrollará una línea de productos software sobre juegos de mesa debido a que comparten un amplio conjunto de características comunes (existencia de tablero, gestión del turno, posibilidad de jugar con dados, etc.).

### Objetivo secundario

El objetivo secundario de este proyecto es la implementación de una aplicación cliente-servidor, como resultado de uno de los posibles productos de la línea de productos software diseñada, que permita a los usuarios jugar a distintos juegos de mesa, gestionando la creación de partidas de diferentes tipos de juegos, la posibilidad estar unido a varias partidas a la vez, permitir tanto el juego online como el juego contra jugadores controlados por el ordenador, etc.

### Herramientas y medios empleados

#### Middleware de comunicaciones

Un middleware es un software de conectividad (basado en el modelo cliente-servidor) que aporta abstracción sobre la complejidad y heterogeneidad de las comunicaciones de un sistema dado (3). Es decir, a través de la definición de unas interfaces de comunicación entre los distintos sistemas que forman una aplicación distribuida se eliminan los problemas de comunicación entre ellos pudiendo tener ambos diferentes arquitecturas hardware, sistemas operativos, protocolos de red o lenguajes de programación. Además dichas características dan al sistema una escalabilidad mucho mayor, lo cual es uno de los objetivos de este proyecto.

Por tanto se ha elegido en este caso middleware **ZeroC Ice 3.4.2** (4) que cumple con todas las características comentadas anteriormente.

#### Lenguaje de programación

Se ha elegido como lenguaje de programación **Java** (5)tanto para el cliente como para el servidor debido a que se consigue la independencia de la arquitectura y del sistema operativo de la máquina donde se despliegue. La versión del kit de desarrollo de Java (**JDK**) utilizado ha sido la **1.7**.

Para el desarrollo de la interfaz de comunicaciones entre cliente y servidor se ha utilizado el lenguaje **Slice** (6) que proporciona el middleware de comunicaciones ZeroC Ice y que permite realizar la definición de los tipos y las interfaces de los objetos utilizados por una aplicación. Esta descripción es independiente del lenguaje de implementación por lo que no importa el lenguaje en el que estén escritos tanto el cliente como el servidor.

#### Entorno de desarrollo

El entorno de desarrollo elegido ha sido **Eclipse 4.2 Juno** (7) por ser una herramienta de desarrollo “open source”, multiplataforma y que ofrece gran cantidad de plugins para la mayoría de tecnologías que puedan necesitarse en el desarrollo de aplicaciones. En este caso se han utilizado los siguientes plugins:

* **Windows Builder Pro** (8) es un plugin que facilita el diseño y creación de las interfaces en java mediante la tecnología Swing.
* **Slice2Java** es un plugin aportado por ZeroC Ice que genera automáticamente a partir de la definición de la interfaz de comunicaciones y objetos comunes entre cliente y servidor mediante el lenguaje Slice, todas las clases necesarias para el funcionamiento de la aplicación.

#### Base de datos

Para la implementación de la base de datos se ha elegido la utilización del sistema gestor de base de datos relacional **MySQL Server 5.5** (9) y su herramienta de gestión y diseño **MySQL Workbench 5.2 CE**.

Además para facilitar las transacciones con la base de datos se ha decidido la utilización del framework de persistencia para Java llamado **Hibernate** (10)que permite manejar de forma transparente dichas transacciones.

#### Herramienta de análisis y diseño

Debido a las características de este proyecto fin de carrera, se necesita una herramienta con potencia suficiente para desarrollar de forma adecuada la línea de productos software. En este caso se ha elegido **Visual Paradigm** ya que permite el uso del lenguaje de modelado **UML,** así como la creación de perfiles necesarios para el desarrollo de este proyecto.

#### Herramienta de gestión

Para la gestión del proyecto se ha decidido la utilización de la herramienta **JIRA** (11)de la compañía Atlassian, que es una aplicación basada en web para el seguimiento de errores, incidentes y gestión operativa de proyectos. Debido a la magnitud del proyecto y los recursos humanos del mismo, esta herramienta permite organizar de manera ágil las diferentes tareas del proyecto, así como llevar un control de las mismas y el tiempo dedicado a la consecución de cada una de ellas.

## ANTECEDENTES

Aquí voy a escribir los antecedentes porque soy más guay que nadie.

## MÉTODO DE TRABAJO

## RESULTADOS

## CONCLUSIONES Y PROPUESTAS

## BIBLIOGRAFÍA

1. **Diez, Oscar y Trujillo, Salva.** *Fábricas de Software: experiencias, tecnologías y organización.* s.l. : Ra-Ma, 2010.

2. **Clements, Paul C. y Northrop, Linda M.** *Salion, Inc.: A Software.* 2002.

3. **Colouris, G., Dollimore, J. y Kindberg, K.** *Distributed System: Concepts and Design.* s.l. : Addison Wesley, 2005.

4. **Documentación ZeroC Ice.** [En línea] http://doc.zeroc.com/display/Ice/Ice+Manual.

5. **Página web de Java.** [En línea] http://www.oracle.com/technetwork/java/index.html.

6. **Documentación sobre el lenguaje Slice.** [En línea] http://doc.zeroc.com/display/Ice/The+Slice+Language.

7. **Página web de Eclipse.** [En línea] http://www.eclipse.org.

8. **Página web de Windows Builder Pro.** [En línea] https://developers.google.com/java-dev-tools/wbpro/.

9. **Página web de MySQL.** [En línea] http://dev.mysql.com/doc/.

10. **Página web de Hibernate.** [En línea] http://hibernate.org/docs.