Resumen

El objetivo de este Proyecto Fin de Carrera es desarrollar una serie de generadores de código que automaticen parte de la metodología TE.NET. TE.NET es un proyecto mucho más amplio que se está llevando a cabo dentro del Departamento de Matemáticas, Estadística y Computación y cuyo objetivo es crear una versión de la metodología TENTE para la plataforma .NET.

TENTE es una metodología dirigida por modelos y orientada a características para el desarrollo y configuración de Líneas de Productos Software. TENTE combina diversas tecnologías novedosas provenientes de diversos campos, tales como los lenguajes orientados a características, las transformaciones de modelos o la generación de código.

La utilización en la primera versión de TENTE de *CaesarJ*, un moderno y avanzado lenguaje de programación orientado a características, resultó ser un serio obstáculo para su transferencia tecnológica. La mayoría de las empresas interesadas en adoptar una metodología para el desarrollo de Líneas de Productos Software como TENTE se mostraban muy reticentes a cambiar su lenguaje de programación habitual por CaesarJ (u otro lenguaje similar).

Por tanto se decidió rehacer la metodología TENTE para que fuera compatible con un lenguaje de programación convencional. Dado que un amplio número de empresas del tejido TIC cántabro desarrollan software sobre la plataforma .NET, se decidió rediseñar TENTE para que fuese compatible con C#, el lenguaje bandera de la plataforma .NET.

Siguiendo dicha línea de trabajo, se creó un patrón de diseño denominado *Slicer Pattern* que facilitaba la implementación de Líneas de Productos Software en C# utilizando un mecanismo denominado *clases parciales*. Utilizando el *Slicer Pattern* se podían implementar y configurar con cierta facilidad modelos de diseño orientados a características, descritos en UML.

No obstante, el paso de dichos modelos a código debía realizarse de forma completamente manual, lo que es un proceso tedioso, costoso y no libre de errores. El objetivo de este proyecto es crear una serie de generadores de código que transformen automáticamente dichos modelos orientados a características en una implementación en C# basada en el *Slicer Pattern*. Ello ayuda a reducir tiempo y coste de desarrollo, a la vez que evita que puedan aparecer errores debidos a la intervención humana.

Palabras Clave: Líneas de Productos Software, Generación de Código, TENTE, Slicer Pattern, C#, .NET, Epsilon, Desarrollo Software Dirigido por Modelos.

Preface

The aim of this project is to develop an Information System that allows the management of students'records who are enrolled in exchange programmes within the School of Nursing, at the request of the management staff and the secretarial services of the school, who managed these records manually until now. In order to automate and facilitate the management, the system must enable the administration of all the data necessary to carry out this management. For example, the system must allow to register foreign universities, centers associated with these universities as well as subjects taught in these centers to be considered within the processes of recognition among subjects.

The main goal of the application must be to allow the creation of detailed recognition plans for each student in the most suitable way for administrative staff. Such recognition plans must specify the subjects studied by students in the University destination during the exchange program, and for each of these subjects, which subject of the student's origin university wants to be recognised. In addition, the application must allow to export records from students to various formats, such as PDF.

The application, due to technical restrictions, will be developed as a distributed desktop application that communicates remotely using MySQL as a database management system. The system will be develop using a three-layer architecture which favours its evolution and adaptation.

Keywords: Information Systems, Three-Layer Software Architecture, Record Management, MySQL

Capítulo 1

Introducción

Este capítulo sirve de introducción a la presente Memoria de Proyecto Fin de Carrera. En él se describen los objetivos generales del proyecto, así como el contexto donde se enmarca. Por último, se describe como se estructura el presente documento.

1.1. Introducción

El principal objetivo de este Proyecto de Fin de Carrera es implementar un conjunto de generadores de código que permitan transformar modelos orientados a características para una línea de productos software en una implementación de dichos diseños para la plataforma .NET utilizando para ello las facilidades que ofrecen las clases parciales del lenguaje C# basadas en el Patrón Slicer. A continuación intentaremos introducir de forma breve al lector en estos conceptos.

El objetivo de la *Línea de Producto Software* **??**, de un segmento particular de mercado, es reutilizar una base común de tecnología y proporcionar productos personalizados, acordes a las necesidades del cliente de manera eficiente, eficaz y a un coste razonable.

El uso de las líneas de producto software permite la reducción de costes de desarrollo por la reutilización de la tecnología en los distintos sistemas, a mayor cantidad de productos a desarrollar mayor rentabilidad respecto a los sistemas creados individualmente. Ofrece alta calidad en el producto resultante porque se realizan pruebas de los componentes de la plataforma en diferentes tipos de producto para ayudar a detectar y corregir errores. Reduce el tiempo de creación debido a la reutilización de los componentes ya existentes para cada nuevo producto y reduce también el esfuerzo requerido por el mantenimiento ya que cuando se cambia algo de un componente de la plataforma, ese cambio se propaga a todos los componentes que lo empleen, y de esta forma se reduce el esfuerzo de aprender cómo funciona cada elemento individualmente.

En contraposición a la flexibilidad que ofrece el desarrollo de software individual, específico para cliente pero que supone grandes costes, las líneas de producto software delimitan las variaciones de sus productos a un conjunto prefijado y optimizan, por tanto, los procesos para dichas variaciones.

La línea de productos software se puede extrapolar a otros ámbitos de producción. Un ejemplo clásico de línea de productos es la fabricación de automóviles, donde se ofrece al cliente un modelo base al cual puede añadir aquellos extras que así desee, personalizando el vehículo y adaptándolo a sus necesidades. De esta forma partiendo del mismo modelo y de unas variaciones adicionales preestablecidas, y diseñadas de tal forma que se adaptan perfectamente al modelo seleccionado, se puede obtener gran cantidad de variaciones en el modelo final de manera automática.

En el ámbito del desarrollo software, las empresas ya no se centran en la creación de un producto específico para un cliente (por ejemplo, diseñar y construir un portal para la Universidad de Cantabria), sino en un domino (por ejemplo, diseñar y construir un portal para universidades). Los principales desafíos a los que se enfrentan las empresas son: delimitar dicho dominio, identificar las distintas variaciones que se van a permitir y desarrollar la infraestructura que permita realizar los productos a bajo coste sin reducir la calidad.

El proceso de desarrollo de la línea de productos software se divide en dos procesos**??**: Ingeniería de Dominio e Ingeniería de la Aplicación. Por un lado la *Ingeniería del Dominio* se encarga de la construcción de la plataforma mediante la delimitación del conjunto de aplicaciones para las que está creada, además de definir y construir qué características serán reusables y cuales específicas para cada uno de los productos que se desean fabricar.

Por otra parte, la *Ingeniería de la Aplicación* se encarga de la creación de los productos para clientes concretos. Partiendo de la plataforma creada en la fase de Ingeniería de Dominio, y reutilizando tantos componentes como fuera necesario, se crea una especialización del producto base acorde a los requisitos del cliente.

Una de las técnicas más utilizadas para realizar dicho análisis es la *programación orientada a características***??**, que intenta encapsular porciones coherentes de funcionalidad proporcionadas por una aplicación en módulos independientes llamados características. Lo que convirte la obtención de diferentes versiones de una misma aplicación combinando diferentes conjuntos de características en una tarea sencilla. Los *lenguajes orientados a características* deben asegurar que el resultado de la composición de un conjunto de características produce como resultado una aplicación correcta y segura.

Tal como se ha descrito al inicio de este apartado, el objetivo del presente Proyecto Fin de Carrera consiste en el desarrollo e implementación de unos generadores de código que permitan la tranformación del diseño de los modelos en una implementación en código C# de dichos diseños, para ello se usarán las prestaciones que ofrecen el uso de las clases parciales del

lenguaje C# basadas en el patrón Slicer.

Las *clases parciales* permiten a los desarrolladores fragmentar la implementación de una clase en un conjunto de ficheros, cada uno de los cuales contiene una porción, o incremento, de una funcionalidad de la clase. Sin embargo, no ofrecen ningún mecanismo para agrupar o encapsular características, por lo que no es posible ocultar clases y métodos que pertenecen a una característica específica de aquellas clases y métodos que pertenecen a características independientes. Además, permiten añadir nuevos atributos y métodos a existentes clases parciales pero no permite sobreescribir o extender métodos ya existentes.

Para solventar dichos problemas, el profesor Pablo Sánchez, dentro del Departamento de Matemáticas, Estadística y Computación, ha desarrollado un patrón de diseño llamado *Patrón Slicer***??** que parte de la siguiente idea: todos los problemas que se pretenden solucionar tienen origen en el hecho de no poner tener métodos con el mismo nombre en distintas clases parciales, hay que evitar dicha situación. Estos fragmentos de clases parciales, son combinados en tiempo de compilación para crear una única clase que auna todas las características seleccionadas inicialmente por el cliente.

Por ejemplo, supongamos que un cliente quiere un vehículo con varias características adicionales entre las que se encuentran: aire acondicionado, sensor de lluvia, medidor de temperatura en grados Celsius y GPS integrado en idioma español e inglés. La base de nuestro producto final será el vehículo, al cual iremos añadiendo las distintas características requeridas por el cliente. Hay algunas peculiaridades, la clase del medidor de temperatura puede estar a su vez fragmentada en varios ficheros (temperatura en Celsius, temperatura en Farentheit) y de los cuales en el modelo final solo usaremos uno de ellos, el de temperatura en Celsius. Lo mismo ocurre con el selector de idiomas para el GPS, solo se elegirá el idioma español e inglés. De esta forma, el producto final juntará todas estas características dentro de un mismo elemento que será el vehículo entregado al usuario final atendiendo a sus requisitos.

El objetivo de este Proyecto Fin de Carrera es implementar generadores de código que abordarán tanto la implementación de la familia de productos software cubierta por la línea de productos, como la configuración de productos concretos pertenecientes a dicha familia utilizando las prestaciones de las clases parciales en C# y el Patrón Slicer. Con esto esperamos haber aclarado el primer párrafo de esta sección al lector no familiarizado con las líneas de productos software, clases parciales en lenguaje C# y/o el Patrón Slicer.

Tras esta introducción, el resto del presente capítulo se estructura como sigue: La Sección [] proporciona []. Por último, la Sección 1.2 describe la estructura general del presente documento.

1.2. Estructura del Documento