

Polimorfismo Familiar con CaesarJ como Soporte a la Integración de Componentes para la Herramienta CASE para Desarrollo de Software Educativo basado en la Metodología DESED

Ulises Jesús López-Maldonado¹, Gustavo Pelaez-Camarena¹, Ulises Juarez-Martinez¹, José Cartas Orozco², María Antonieta Abud-Figueroa¹, Alejandro Domingo Velázquez-Cruz¹

¹Instituto Tecnológico de Orizaba, Departamento de Posgrado e Investigación, Avenida Instituto Tecnológico #852, Colonia Emiliano Zapata. Orizaba, Veracruz.

²Instituto Politécnico Nacional, Escuela Superior de Ingeniería Mecánica y Eléctrica, Sección de Estudios de Posgrado e Investigación
Avenida Instituto Politécnico Nacional s/n. Delegación Gustavo A. Madero, México, D.F.

lopezu@acm.org,sgpelaez@itorizaba.edu.mx,ujuarez@itorizaba.edu.mx,jcartaso@gmail.com,
mabud@itorizaba.edu.mx, lap.alejandro@acm.org

Abstract. This paper takes family polymorphism as the necessary base to construct a CASE tool based on DESED methodology for Educational Software Development. DESED requires a high level of flexibility that cannot be accomplished by traditional inheritance and polymorphism; it is possible to achieve such flexibility using collaboration interfaces, composing class hierarchies and polymorphic usage of class families to extend the functionality of traditional polymorphism.

Keywords: Herramienta CASE, polimorfismo familiar, CaesarJ, DESED.

1 Introducción

La programación orientada a componentes es una de los paradigmas de la ingeniería de software que intenta posicionarse como una solución para reducir el tiempo y costo de desarrollo[1], gracias al soporte que se da en el desarrollo de sistemas al facilitar su construcción sin necesidad de partir de cero a diferencia de la manera tradicional en la que se realiza este proceso.

El polimorfismo familiar[2] es un concepto novedoso para la construcción de componentes, que extiende la funcionalidad del polimorfismo convencional de la programación orientada a objetos, por medio de mecanismos provistos por el lenguaje CaesarJ (interfaz de colaboración, familias de clases, clases virtuales), en el que una

clase será contendor de otras clases, lo que permite que se tenga acceso a cada una de las partes de la clase, consiguiendo que las instancias sean únicas y pertenezcan a una instancia protegida. El polimorfismo familiar define los componentes base, y proporciona el soporte para que estos y otros componentes se integren de manera más transparente dentro de una aplicación.

2 Metodología para el Desarrollo de Software Educativo (DESED)

En la actualidad las tecnologías de la informática son adoptadas por los profesionales de la educación como un recurso didáctico dentro y fuera de los salones de clases, como apoyo al aprendizaje de los alumnos. Hoy en día existen productos de software que proporcionan una forma novedosa de presentar la información, dichos productos emplean la tecnología multimedia (texto, voz, imagen, video), con lo cual logran captar la atención de los usuarios, y proporcionan una forma agradable de obtener conocimientos.

DESED [3] establece que el proceso de desarrollo de software cumpla las reglas didácticas para que el software educativo a desarrollar a partir de éstas satisfaga las necesidades del proceso de enseñanza-aprendizaje.

La metodología consta de 14 pasos fundamentales, que describen aspectos de ingeniería de software, educación, didáctica y diseño gráfico, entre otros. Es importante resaltar, que el desarrollador de SE planee su producto de software y considere las características que se plantea en cada fase del desarrollo.

3 Desarrollo de Software Asistido por Computadora (CASE)

El concepto de utilizar herramientas para asisitir, ayudar y/o automatizar las actividades en el campo del desarrollo de software ha avanzado con el paso de los años, la ingeniería de software asistida por computadora (CASE) sin lugar a dudas ha venido a revolucionar la ingeniería de software ya que aumenta la productividad en el desarrollo y reducen considerablemente los costos en términos de tiempo y de dinero, la tecnología CASE brinda soporte en todos los aspectos del ciclo de vida del desarrollo de software, o en etapas del mismo, en tareas como el diseño del proyecto, implementación del código de manera automática, documentación o detección de errores, entre otras. En [4] se establece que una herramienta CASE se compone de 6 elementos básicos: Interfaz de usuario, Repositorio o Diccionario de datos, Metamodelo, Generador de informes, Herramientas de carga / descarga de datos, y Facilidades de comprobación.

El Metamodelo (No siempre visible al usuario), constituye el marco en el que se definen las técnicas, y metodologías soportadas por la herramienta, para este trabajo enfocado en el desarrollo de SE, se implementa la metodología DESED como base del Metamodelo y se proveerá de un Repositorio a la herramienta, donde se almacenen los elementos definidos o creados por la misma (diagramas, diseños, algoritmos, recursos digitales), con sus estados correspondientes, así como una Interfaz gráfica de usuario para la interacción con la herramienta para el desarrollo de

las aplicaciones. Para la construcción de los productos finales que se obtendrán con ésta herramienta se implementa el polimorfismo familiar, como Herramienta de carga / descarga de datos, de la misma manera se establecerán los componentes Generador de informes y Facilidades de comprobación, así como la integración general de los componentes gráficos y de acceso al Repositorio.

4 Polimorfismo Familiar

Es un conjunto de clases internas, contenidas en una clase externa, formando familias de tipos. Los métodos de una familia se conocen como interfaz de colaboración y sus clases internas también son llamadas clases virtuales. Las asociaciones de tipos de una familia se heredan a los subtipos de esa familia, de esta manera se forman conjuntos de clases que colaboran en una unidad nueva [5]. En CaesarJ[6] el polimorfismo familiar se representa por medio de la interfaz de colaboración (ACI)[7], de tal manera que las clases que integran un componente se manipulan polimórficamente para ser utilizadas en los diferentes contextos donde se requiere dicho componente, en [8] el polimorfismo familiar es considerado la mejor forma aprovechar los beneficios del agrupamiento de las clases.

El polimorfismo familiar, provee a los componentes la capacidad de ser reutilizados en diferentes contextos, al reemplazar parte de los mismos, pero sin perder su funcionalidad original. Esto es posible gracias a las características de las interfaces (clases abstractas) que al ser implementadas por otras clases, éstas últimas definen su comportamiento dependiendo de la funcionalidad específica que se requiere, pero sin alterar su estructura original. Las ACI son interfaces, cuyas declaraciones de métodos se dividen en provistos, lo que se espera obtener, y requeridos, lo que se brinda al contexto donde se aplique.

Las características particulares de CaesarJ en la programación orientada a componentes contribuyen a la integración de los mismos. Los componentes son colaboraciones de clases en CaesarJ, las clases virtuales y mixin[9] definidos como una nueva colaboración a partir de las propiedades de clases anteriores, permiten la creación de familias de clases con funcionalidad común de componentes, las interfaces de colaboración proveen las interfaces de comunicación necesarias para la integración de los componentes, los cuales implementan esas interfaces y forman construcciones más complejas que permiten el desarrollo de aplicaciones de software.

5 Desarrollo

La arquitectura de la herramienta se define mediante un estilo arquitectónico en capas, considerando los componentes básicos de una herramienta CASE (Repositorio, Interfaz de usuario, Metamodelo, Herramienta de carga / descarga de datos, Facilidades de comprobación, Generador de informes), como se muestra en la Fig. 1.

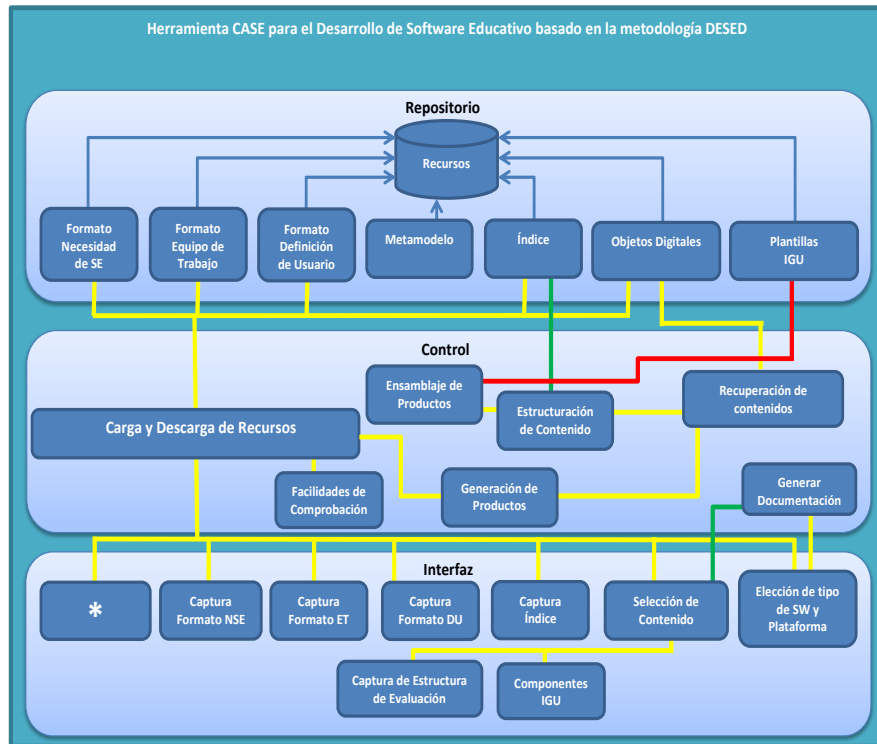


Fig. 1. Arquitectura.

El Repositorio es la base de datos central de la herramienta, la Interfaz que es utilizada por los demás componentes para la interacción con el usuario. Cabe mencionar, que estos dos componentes son claramente definidos dentro de la arquitectura en capas independientes, en la capa media de la herramienta denominada “Control”, están integrados los elementos restantes de la misma. El Metamodelo, se establece mediante la metodología DESED que es la encargada de definir las técnicas y métodos para el desarrollo de las aplicaciones, las Herramientas de carga/descarga de recursos, las cuales integran los componentes de: Interfaz gráfica de usuario, Objetos digitales y desarrollo de contenidos así como las rutinas necesarias para el funcionamiento general de la misma. El Generador de informes que proporcionará la documentación tanto de la de herramienta CASE, como la de las aplicaciones. Y las Facilidades de comprobación que garantiza la integridad y consistencia de los esquemas generados.

Como se mencionó en el párrafo anterior, la integración de los componentes, es una etapa esencial para el desarrollo de las aplicaciones de manera automatizada, de igual manera, la herramienta propone el desarrollo de contenidos educativos, con diferentes plataformas destino (Móvil, Web, Escritorio), por lo que surge la necesidad de proveer una estructura adaptable a las necesidades de desarrollo de cada una de ellas.

Para la construcción de los componentes de software se propone la utilización de CaesarJ, las familias polimórficas en CaesarJ permiten establecer una estructura base, y extender en cada familia las características propias de cada desarrollo, es decir, definir una interfaz de colaboración, y extender de esta las funcionalidades, de acuerdo a la problemática que se presente. Las diferentes implementaciones del núcleo de la familia se llevan a cabo mediante la refinación a lo largo de la herencia, en la jerarquía de las familias y colaboraciones, que da lugar a múltiples combinaciones de estas implementaciones lo que permite al conjunto general de clases manipularse polimórficamente.

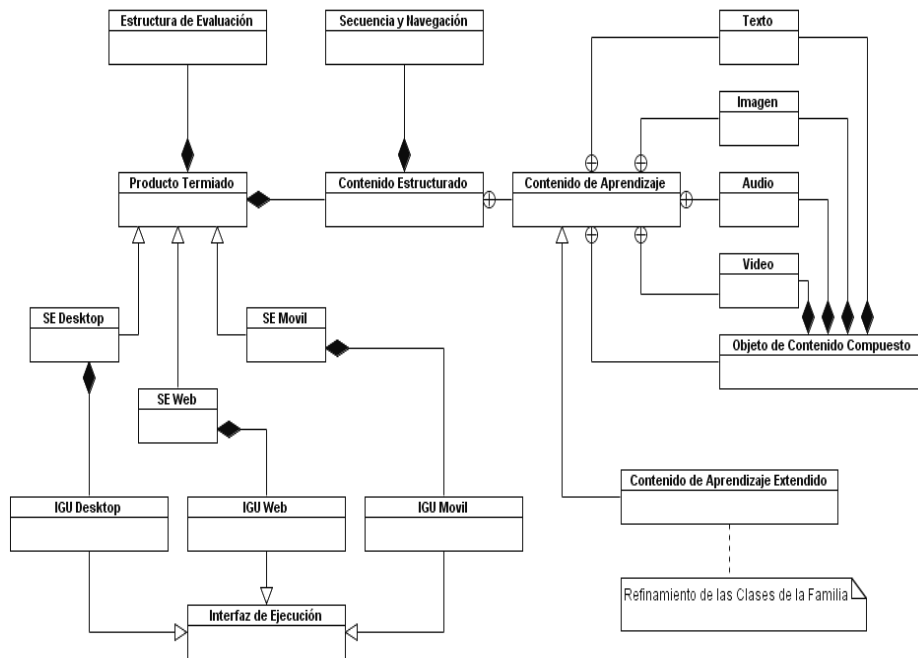


Fig. 2. Implementación de Familia polimórfica.

La creación de una familia polimórfica para el desarrollo de software educativo, con plataformas destino en diferentes modalidades (Móvil, Web, Escritorio), queda representado en la Fig. 2, en donde la clase “Contenido Estructurado” es un ACI, ya que es una clase que contiene clases internas (clases virtuales), con la familia de tipo denominada “Contenido de Aprendizaje”, la cual también forma una ACI, esta estructura representa el núcleo de nuestra herramienta, que implementarán cada una de las familias que heredan de la misma. La funcionalidad original persiste, pero las implementaciones pueden variar de acuerdo a las necesidades particulares de cada una de ellas. Para la obtención del “Producto Terminado”, cuyo objetivo es la obtención de un SE, se incorporan una clase de “Secuencia y Navegación” que permite al usuario definir la organización del contenido sin inhibir la capacidad de reutilización de los contenidos ya creados, y una “Evaluación” para comprobar el aprendizaje del estudiante mientras avanza a lo largo de los cursos. De igual manera se incrementa la

funcionalidad al agregar las clases de Interfaz: Móvil, Web y Escritorio para cada dominio en particular. De la misma manera se pueden integrar diferentes familias de tipos a esta estructura para integrar los componentes necesarios para las Facilidades de comprobación y Generación de informes, ya sea al núcleo de nuestra herramienta o particulares a cada dominio.

6 Herramienta CASE para el desarrollo de software educativo basado en la metodología DESED (HDESED)

La HDESED consta de tres módulos de desarrollo, es decir, uno por cada capa de la arquitectura de la herramienta: Repositorio, Control, Interfaz. Cabe mencionar que el módulo de Control, denominado “Módulo de ensamblaje de objetos” es el primero de los tres. Una vez definida la arquitectura de la misma, así como la Interfaz de programación de aplicaciones o API (Application Programming Interface), el módulo de control se encuentra en la etapa de implementación la cual concluirá con el desarrollo del prototipo como versión inicial de la HDESED y la etapa de prueba de la misma, posteriormente se incorporan los módulos restantes que concluyeron su etapa de desarrollo, para seguir con la implementación de los mismos.

Conclusiones

En este trabajo se expone el polimorfismo familiar como alternativa a la integración de componentes en el desarrollo de una herramientas con características CASE a través de técnicas de ingeniería de software para su desarrollo.

Por otro lado se establece que para el desarrollo de software educativo de calidad es necesario seguir una metodología formal de desarrollo que contemple aspectos educativos, e implementar correctamente las técnicas de ingeniería de software en un lenguaje de programación que provea el soporte necesario de este tipo de desarrollos.

Por lo tanto se concluye que la combinación de estos elementos garantiza el desarrollo de una aplicación lo suficientemente robusta y flexible, para obtener aplicaciones educativas que cumplan con los requisitos pedagógicos y las necesidades de los programas educativos en diversas plataformas: Móvil, Web y Escritorio más allá de un simple gestor de contenidos.

Trabajo futuro

Como trabajo futuro se realizará la integración de los 2 módulos restantes y establecer un caso de estudio para validar la integridad y eficiencia de la HDESED. Además se evaluará la opción del desarrollo de SE con base en la iniciativa de accesibilidad web y la flexibilidad que nos ofrece el polimorfismo familiar como soporte a la integración de componentes, para poder incluir aplicaciones con escenarios para personas con capacidades diferentes.

Referencias

- [1] Grundy, J., Patel R.: Developing Software Components with the UML, Enterprise Java Beans and Aspects. In: Proceedings of 2001 Australian Software Engineering Conference. Canberra, Australia (1998)
- [2] Ernst, E.: Family polymorphism. In: Proceedings of ECOOP 2001. LNCS 2072. Springer-Verlag (2001)
- [3] López, B.: Metodología para el desarrollo de software educativo (DESED). Tesis, Instituto Tecnológico de Orizaba. Orizaba, Veracruz, México (2002)
- [4] Piattini, M., Calvo-Manzano, J., Cervera, J., Fernández, L.: Análisis y diseño detallado de Aplicaciones Informáticas de Gestión. Ed. Ra-Ma (1996)
- [5] Aracic I, Gasiunas, V., Mezini, M., Ostermann, K.: Overview CaesarJ. In: Darmstadt University of Technology. D-64283 Darmstadt, Alemania (2006)
- [6] CaesarJ,<http://caesarj.org/> (2011)
- [7] Mezini, M., Ostermann, M.: Integrating independent components with ondemand modularization. In: Proceedings of OOPSLA (2002)
- [8] Wittmann, A.: Towards Caesar: Family Polymorphism for Java. In: Darmstadt University of Technology. Alemania (2003)
- [9] Sousa, E., Monteiro, M.: Overview CaesarJ. In: Darmstadt University of Technology. D-64283 Darmstadt, Alemania (2006)