Programación Matemática y Software (2014) 6 (1): 1-6. ISSN: 2007-3283

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

Family pholymorphism with CaesarJ as support for component integration in educational software production based on Desed methodology

Ulises Jesús López-Maldonado,[1]* Gustavo Peláez-Camarena,[1] Ulises Juárez-Martínez,[1] José Cartas-Orozco,[2] María Antonieta Abud-Figueroa,[1] Alejandro Domingo Velázquez-Cruz[1]

[1] Instituto Tecnológico de Orizaba, Departamento de Posgrado e Investigación.
Av. Instituto Tecnológico 852, col. Emiliano Zapata. Orizaba, Veracruz, México
[2] Sección de Estudios de Posgrado e Investigación, Escuela Superior de Ingeniería Mecánica y Eléctrica, Instituto Politécnico Nacional.
Av. Instituto Politécnico Nacional s/n, del. Gustavo A. Madero. México, DF, México
* Correo-e: lopezu@acm.org

PALABRAS CLAVE:

herramienta CASE, polimorfismo familiar, CaesarJ, Desed, software educativo

RESUMEN

El presente trabajo presenta al polimorfismo familiar como la base necesaria para la construcción de una herramienta de desarrollo de software asistido por computadora (CASE, en inglés), basada en la metodología Desed para el desarrollo de software educativo. Desed requiere un alto nivel de flexibilidad que no puede lograrse mediante la herencia y el polimorfismo tradicionales. Sin embargo, es posible lograr esa flexibilidad al utilizar interfaces de colaboración que componen la jerarquía de clases, y el polimorfismo de las familias de clases, para extender la funcionalidad del polimorfismo tradicional.

KEYWORDS:

CASE tool, family polymorphism, CaesarJ, Desed, educational software

ABSTRACT

This paper takes family polymorphism as the necessary basis to construct a computer aided software engineering (CASE) tool based on Desed methodology for educational software development. Desed requires a high level of flexibility that cannot be accomplished by traditional types of inheritance and polymorphism. It is possible to achieve such flexibility, however, through the use of collaboration interfaces, the composition of class hierarchies and the polymorphic utilization of class families to extend the functionality of traditional polymorphism.

1 INTRODUCCIÓN

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

El polimorfismo familiar [2] es un concepto novedoso para la construcción de componentes, 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 éste una clase es contendor de otras clases, lo que permite que se tenga acceso a cada una de las partes de la clase, y así se consigue que las instancias sean únicas y pertenezcan a una instancia protegida. El polimorfismo familiar define los componentes base y proporciona el soporte para que éstos y otros componentes se integren de manera más transparente dentro de una aplicación.

2 METODOLOGÍA PARA EL DESARROLLO DE SOFTWARE EDUCATIVO

En la actualidad las tecnologías de la informática son adoptadas por los profesionales de la educación como recurso didáctico dentro y fuera de las aulas, como apoyo al aprendizaje de los alumnos. Actualmente algunos productos de software proporcionan una forma novedosa de presentar la información empleando la tecnología multimedia (texto, voz, imagen, video) para captar la atención de los usuarios y proporcionar una forma atractiva de aprender.

De acuerdo con Desed [3], el proceso de desarrollo de software debe cumplir las reglas didácticas para que el software educativo que se va a desarrollar 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

La utilización de herramientas para asistir, ayudar 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, por sus siglas en inglés) sin lugar a dudas ha venido a revolucionar este campo, ya que aumenta la productividad en el desarrollo y reduce considerablemente los costos en términos de tiempo y de dinero, esta tecnología 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.

De acuerdo con Piattini y colaboradores [4], una herramienta CASE se compone de seis 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) es 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 Desed como base del metamodelo y se provee 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 esta 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, forma 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, de acuerdo con Whittmann [8], el polimorfismo familiar es considerado la mejor forma aprovechar los beneficios del agrupamiento de las clases.

Este polimorfismo provee a los componentes la capacidad de ser reutilizados en diferentes contextos, al reemplazar parte de los mismos, pero sin perder su funcionalidad original; y es posible gracias a las características de las interfaces (clases abstractas), que al ser implementadas por otras clases, estas 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 figura 1.

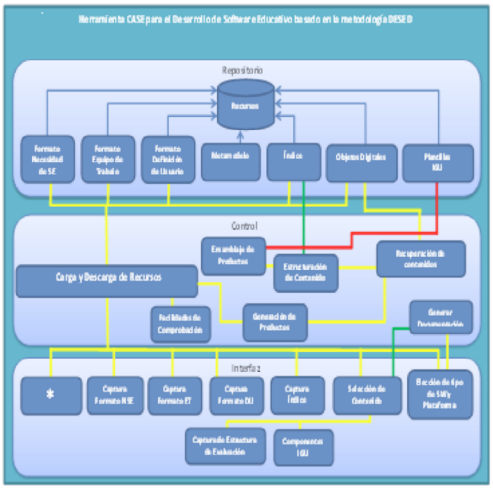

Figura 1. Arquitectura

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 garantizan 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 ésta las funcionalidades, de acuerdo con 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. Esto da lugar a múltiples combinaciones de dichas implementaciones y permite al conjunto general de clases manipularse polimórficamente.

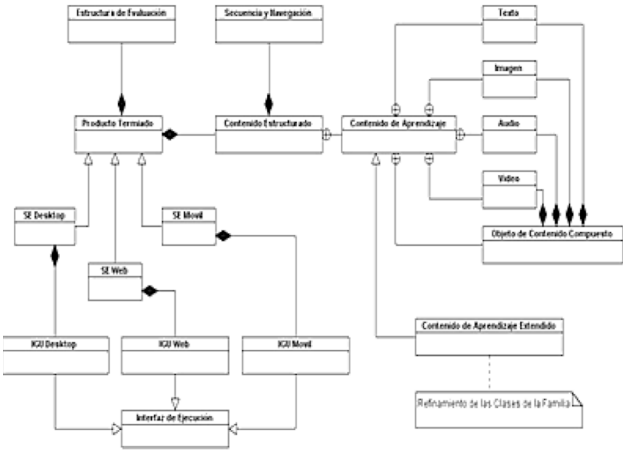

Figura 2. Implementación de familia polimórfica

La creación de una familia polimórfica para el desarrollo de software educativo, con plataformas destino en diferentes modalidades (móvil, web, escritorio) está representada en la figura 2, en donde la clase "Contenido estructurado" es un ACI, es decir, 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 implementará cada una de las familias que herede de la misma. La funcionalidad original persiste, pero las implementaciones pueden variar de acuerdo con las necesidades particulares de cada una de ellas.

Para la obtención del "Producto terminado", cuyo objetivo es la obtención de un SE, se incorpora 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

Esta herramienta denominada HDSED consta de tres módulos de desarrollo, uno por cada capa de su arquitectura: repositorio, control, interfaz. El módulo de control, denominado "Módulo de ensamblaje de objetos" es el primero de los tres.

La integración de los componentes es una etapa esencial para el desarrollo de las aplicaciones de manera automatizada, de igual modo, 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, como ya se mencionó, 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 con 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. Esto da lugar a múltiples combinaciones de estas implementaciones y permite al conjunto general de clases manipularse polimórficamente.

Una vez definida la arquitectura de la misma, así como la interfaz de programación de aplicaciones (API, por sus siglas en inglés), el módulo de control se encuentra en la etapa de implementación. Ésta concluirá con el desarrollo del prototipo como versión inicial de la HDSED 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.

7 CONCLUSIONES

En este trabajo se expuso el polimorfismo familiar como alternativa a la integración de componentes en el desarrollo de una herramienta 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 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.

8 TRABAJO FUTURO

Como trabajo futuro se realizará la integración de los dos módulos restantes y se establecerá un caso de estudio para validar la integridad y eficiencia de la HDSED. Además se evaluará la opción del desarrollo de SE con base en la iniciativa de accesibilidad web y la flexibilidad que 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. (1998). En Developing Software Components with the UML, Enterprise Java Beans and Aspects. *Proceedings of 2001 Australian Software Engineering Conference*. Canberra, Australia.
2. Ernst, E. (2001). Family polymorphism. En: *Proceedings of ECOOP 2001*. LNCS 2072. Springer-Verlag.
3. López, B. (2002). Metodología para el desarrollo de software educativo (Desed). Tesis. Instituto Tecnológico de Orizaba. Orizaba, Veracruz, México.
4. Piattini, M., Calvo-Manzano, J., Cervera, J., Fernández, L. (1996). Análisis y diseño detallado de aplicaciones informáticas de gestión. Ra-Ma.
5. Aracic I., Gasiunas, V., Mezini, M., Ostermann, K. (2006). Overview CaesarJ. En: Darmstadt University of Technology. D-64283 Darmstadt, Alemania.
6. CaesarJ, (2011). http://caesarj.org/
7. Mezini, M., Ostermann, M. (2002). Integrating independent components with ondemandremodularization. En: *Proceedings of OOPSLA*
8. Wittmann, A. (2003). Towards Caesar: family polymorphism for Java. En: Darmstadt University of Technology, Alemania.
9. Sousa, E., Monteiro, M. (2006). Overview CaesarJ. En: Darmstadt University of Technology. D-64283 Darmstadt, Alemania.

Acerca de los autores

Ulises Jesús López Maldonado se graduó como Ingeniero en Sistemas Computacionales en el Instituto Tecnológico de Orizaba. Sus intereses incluyen el desarrollo de sistemas desde el enfoque del paradigma orientado a componentes, programación orientada a aspectos, ingeniería de software y software educativo.

Silvestre Gustavo Peláez Camarena se graduó como Ingeniero Industrial en Producción en el Instituto Tecnológico de Orizaba (1978) y como Maestro en Computo Estadistico en el Colegio de Posgraduados de Chapingo (1980). Actualmente es profesor en la Maestría en Sistemas Computacionales dentro de la División de Estudios de Posgrado e Investigación del Intituto Tecnológico de Orizaba. Sus intereses incluyen la ingeniería de software y software educativo. Es también profesor investigador en la Maestría en Sistemas Computacionales de la División de Estudios de Posgrado e Investigación del Instituto Tecnológico de Orizaba.

Ulises Juárez Martínez obtuvo el grado de Doctor en Ciencias en la especialidad de Computación por parte del Cinvestav-IPN en 2008. Su trabajo doctoral consistió en el desarrollo de un marco de trabajo para la programación orientada a aspectos de grano fino. Sus intereses incluyen el desarrollo de software orientado a aspectos, arquitecturas de software, líneas de productos de software, compiladores y TRIZ aplicado al software. Actualmente es profesor investigador en el Grupo de Ingeniería de Software del Instituto Tecnológico de Orizaba, Veracruz, México.

José Cartas Orozco realizó estudios de Ingeniería Mecánica y especialización en Ingeniería de Proyectos en la ESIME Unidad Zacatenco del Instituto Politécnico Nacional. Tiene estudios de posgrado en Ingeniería Industrial Management y Dirección de Operaciones en la Universidad de Houston, Texas. En los últimos años ha participado en proyectos relacionados con el uso de las tecnologías de información y comunicación como por ejemplo: "Método de aprendizaje móvil bilingüe", un caso de uso. Subsidio de la Office of Naval Research Global, con sede en Londres, Inglaterra; "Metodología para el desarrollo de materiales educativos", diseñados como objetos de contenido reutilizables e interoperables bajo el modelo SCORM, para ser utilizados a través de herramientas de aprendizaje para e-learning, entre otros. Actualmente es profesor investigador en la Coordinación de Ingeniería de Sistemas, en la Sección de investigación y Posgrado de la ESIME Unidad Zacatenco del IPN.

Ma. Antonieta Abud Figueroa nació en Orizaba, Veracruz, México. Se graduó como Ingeniera en Electrónica en la Universidad Autónoma Metropolitana Iztapalapa (1984), y como Maestra en Ciencias en Sistemas de Información en el ITESM-Campus Morelos (1991). Actualmente es profesora investigadora en la Maestría en Sistemas Computacionales dentro de la División de Estudios de Posgrado e Investigación del Instituto Tecnológico de Orizaba, y sus áreas de interés son la ingeniería de software y la computación educativa. Es miembro del ACM desde el año 2001 y del IEEE desde 2009.

Alejandro Domingo Velázquez Cruz es estudiante del Doctorado en Ciencias y Tecnologías de la Información en la Universidad Autónoma Metropolitana. Sus intereses incluyen redes, sistemas distribuidos y la ingeniería de software. Actualmente trabaja como docente en la University of the People en la Licenciatura en Ciencias de la Computación.