Artículo **Revista del Desarrollo Tecnólogico**
Diciembre 2017 Vol.1 No.4 1-11

Valoración de una herramienta de generación semiautomática de código para agilizar el desarrollo de software

MARTÍNEZ-LÓPEZ, Fernando José*†, VEGA-FLORES, Patricia y ALCÁNTAR-ORTÍZ, Patricia

Instituto Tecnológico Superior del Sur de Guanajuato. Educación Superior 2000, Benito Juárez, 38980 Uriangato, Gto

Recibido Octubre 5, 2017; Aceptado Diciembre 20, 2017

Resumen

En el presente artículo se muestran resultados de un estudio sobre la evaluación de una herramienta de ingeniería de software asistida por computadora (CASE) utilizada con el objetivo de impactar en la etapa de codificación en proyectos del Centro de Desarrollo de Software (CDS) del Instituto Tecnológico Superior del Sur de Guanajuato (ITSUR). El ITSUR es una institución de educación superior pública descentralizada, perteneciente al Tecnológico Nacional de México. El CDS se destaca por haber sido una de las pocas células albergadas en una institución educativa en haber alcanzado el nivel 3 del modelo CMMI. El CDS en su etapa de codificación, promueve la elaboración de miles de líneas de código fuente, que siendo analizadas históricamente, resultan similares. Líneas de código fuente correspondientes a clases de software de acceso a datos y servicios, por su similaridad entre proyectos son candidatas para plantear su generación automatizada mediante tecnologías CASE. En la actualidad algunas tecnologías CASE facilitan la elaboración semiautomática de código fuente repetible y originan una reducción significativa en tiempos de producción y un aumento en la calidad respecto a reducción de defectos en el software, cuestiones que motivaron el presente estudio y de las cuales se obtuvieron resultados satisfactorios.

Codificación, automatización, productividad, calidad de software

Abstract

This paper presents results of a study on the evaluation of a computer - aided software engineering tool (CASE) used with the purpose of impacting the coding stage in projects of the Centro de Desarrollo de Software (CDS) of the Instituto Tecnológíco Superior del Sur de Guanajuato (ITSUR). ITSUR is a decentralized institution of public higher education, belonging to the National Technological of Mexico. The CDS stands out because it was one of the few cells housed in an educational institution to have reached level 3 of the CMMI model. The CDS in its coding stage, promotes the development of thousands of lines of source code, which being analyzed historically, are very similar. Source code lines corresponding to classes of data access software and services, due to their similarity between projects are candidates to propose their automated generation using CASE technologies. Today, some CASE technologies facilitate the semi-automated production of repeatable source code and lead to a significant reduction in production times and an increase in quality with respect to software defects reduction, issues that motivated the present study and of which satisfactory results were obtained.

Coding, automation, productivity, software quality

Citación: MARTÍNEZ-LÓPEZ, Fernando José, VEGA-FLORES, Patricia y ALCÁNTAR-ORTÍZ, Patricia. Valoración de una herramienta de generación semiautomática de código para agilizar el desarrollo de software. Revista del Desarrollo Tecnológico 2017, 1-4: 1-11

* Correspondencia al Autor (Correo Electrónico: fj.martinez@itsur.edu.mx)
† Investigador contribuyendo como primer autor.

Introducción

En la actualidad el Centro de Desarrollo de Software (CDS) del Instituto Tecnológico Superior del Sur de Guanajuato (ITSUR), institución de educación superior pública descentralizada perteneciente al Tecnológico Nacional de México, se ha convertido en una instancia reconocida a nivel regional y estatal debido al desarrollo de tecnologías mediante productos de software a la medida de alta calidad para clientes de instituciones públicas y privadas. El CDS ha destacado por haber sido una de las pocas células albergadas en una institución educativa en haber alcanzado el nivel 3 del modelo CMMI y con esto haber logrado integrar satisfactoriamente el desarrollo profesionalizante docente y de su alumnado con el desarrollo tecnológico vinculado al sector productivo (Martínez López, Vega Olvera, & Morales Orozco, 2015; Morales Orozco & Gutiérrez Torres, Profesores y alumnos inmersos en la implementación de un modelo internacional de procesos de software, 2014; Morales Orozco & Gutiérrez Torres, Industria y academia, uniendo, 2012)

En los últimos años el CDS ha desarrollado aplicaciones siguiendo estilos arquitectónicos, patrones de diseño, estándares de programación y buenas prácticas en sus productos de software tanto en plataforma web como escritorio (stand-alone), apegándose a procesos definidos mediante el modelo CMMI. Lo anterior ha dado pauta a que las áreas de ingeniería, relacionadas finalmente con la codificación de productos de software, también se hayan desarrollado buscando mejorar la calidad a la par que las áreas administrativas.

Se tiene un nivel institucionalizado en el apego a estándares de programación y existe una cantidad considerable de código fuente que ha sido analizado buscando la elaboración de librerías propias, catálogos de código reutilizable y la posible creación de arquetipos base.

Al comenzar una nueva solución correspondiente a un proyecto de software se utiliza un arquetipo base, que constituye el esqueleto de la aplicación y que conserva de manera inherente el uso de las buenas prácticas adquiridas en los proyectos generados anteriormente. Durante el proceso de elaboración y perfeccionamiento de estos arquetipos ha podido ser observada la particularidad de que en estos se siguen estilos arquitectónicos y patrones de diseño de alto y bajo nivel comunes, patrones que se ven reflejados en código fuente que resulta ser demasiado semejante proyecto tras proyecto, esencialmente el código fuente correspondiente a clases de software de acceso a datos y servicios, que comúnmente debe ser generado manualmente siendo adaptado al modelo de datos correspondiente a la solución diseñada.

En la actualidad los marcos de trabajo de programación más destacados ofrecen la posibilidad de generar código fuente estructural base a partir de un esquema de datos, principalmente aquellos que se dirigen por el patrón arquitectónico conocido como Modelo-Vista-Controlador. Esta técnica de generación semiautomática de código es popularmente conocida como Scaffolding, una técnica que permite mejorar la productividad del desarrollo y su nivel de calidad generando las clases de acceso a datos y algunas veces de servicios e inclusive las de Interfaz Gráfica del Usuario.

ISSN-2531-2960

No obstante, la generación de código mediante estas herramientas se adapta únicamente a los estilos arquitectónicos y patrones de diseño adoptados por los creadores del marco de trabajo.

Asumir una adopción de un marco de trabajo popular representaría el abandono de años de trabajo y adopción de estilos, patrones de diseño y buenas prácticas en el CDS. Por esta razón se consideró apropiado comenzar a introducir herramientas que faciliten las labores de ingeniería en cuanto a la generación automatizada de código, teniendo impacto en las áreas de ingeniería de productos de software de forma que pudiesen obtenerse los beneficios de las herramientas populares pero conservando los estilos, patrones y buenas prácticas de codificación que han sido perfeccionados tras más de 10 años de experiencias en el CDS.

Justificación

El desarrollo del presente proyecto representa un impacto principalmente en aspectos de calidad y capacidad de producción en todos los productos de software que el CDS del ITSUR pueda llegar a elaborar en el futuro próximo. De manera primordial, el impacto en la reducción del tiempo de producción, pues originalmente el CDS invierte por cada catálogo del producto de software a desarrollar de 5 a 16 horas, según la complejidad de las entidades de datos relacionadas a su modelo de Casos de Uso. Gran parte de este tiempo es invertido en la elaboración de clases de software de acceso a datos y clases de servicios correspondientes a las operaciones de altas, lectura, actualizaciones y bajas (CRUD, por sus siglas en inglés: Create, Read, Update, Delete). Las operaciones CRUD generan gran parte del código que se considera semejante en todos los sistemas de información que han sido desarrollados por el CDS y su código fuente es de los principales candidatos a una generación automática o semiautomática según la herramienta CASE lo permita.

La herramienta CASE que se integra en el estudio, logra reducir este tiempo de manera sustancial significativamente.

Adicionalmente, el impacto en la calidad de los productos de software es manifiesto, ya que la herramienta CASE, al generar cientos de líneas de código de manera automatizada, libera al software de errores producidos comúnmente por el factor humano durante este proceso, además de que entrega un código fuente debidamente sangrado, comentado de manera estandarizada, y que respeta los estándares de codificación utilizados por el CDS.

Por otra parte, al generar código de diferentes capas de la arquitectura, el proyecto proporciona un marco de referencia arquitectónica del software de los productos elaborados por el CDS, que integra los estilos y patrones comúnmente utilizados como una estructura a seguir para generar nuevas arquitecturas para futuros productos de software.

Problema

El CDS del ITSUR ha dado pauta a que las áreas de ingeniería relacionadas con la codificación de productos de software mejoren la calidad al mismo nivel de madurez que las áreas administrativas. Su alto nivel de apego a estándares de programación, estilos arquitectónicos y patrones de diseño institucionalizados, y el deseo de conservarlos en lugar de una adopción de un marco de trabajo popular que represente el abandono de años de trabajo motiva a introducir herramientas que faciliten las labores de ingeniería en cuanto a la generación automática de código, para obtener beneficios que permitan optimizar el tiempo de producción y nivel de calidad de los productos generados que actualmente apenas se considera suficiente.

ISSN-2531-2960

Hipótesis

La hipótesis planteada en el presente estudio es la siguiente: La utilización de una herramienta CASE para la generación automática de código fuente permite minimizar la cantidad tiempo invertido en la fase de codificación en proyectos de desarrollo de software e incrementar la calidad de los productos disminuyendo la cantidad de defectos encontrados durante las revisiones técnicas de la misma fase.

Objetivos

Objetivo General

Valorar el uso de una herramienta CASE para la generación de código base en los productos de software del CDS del ITSUR.

Objetivos específicos

- Determinar el esquema común de la arquitectura respecto a estilos y patrones de diseño de los productos de software del CDS.
- Utilizar una herramienta CASE para generan código funcional libre de errores acorde a las capas lógicas identificadas y a las clases de software que las constituyen, alineados a estilos, patrones y estándares del CDS.
- Valorar el trabajo en práctica respecto a la minimización de la cantidad de tiempo invertido en la fase de codificación.
- Valorar el trabajo en práctica respecto a la minimización de la cantidad de defectos encontrados durante las revisiones técnicas de la fase de codificación.

Marco Teórico

Hoy en día, la demanda de software en el mundo, la creciente complejidad del mismo y el progresivo aumento en la capacidad del hardware han traído desde hace tiempo la famosa “crisis del software” en la que como lo menciona (Dijkstra, 1972) ya no es suficiente una única persona para realizar el proceso de construcción del software. Actualmente el proceso de la ingeniería del software requiere mucho más que simplemente programar, es indispensable la aplicación sistemática de procesos, métodos y herramientas CASE para lograr satisfacer las demandas de calidad y costo que requiere el software.

Por lo anterior, existe hoy por hoy una necesidad fundamental de promover la reusabilidad de productos de software mediante herramientas CASE, que se apeguen a métodos y a su vez que estos agilicen los procesos de la ingeniería, tal como lo sugiere (Storr & Jarvis, 1996).

En todo caso es necesario tener una visión clara del significado de la ingeniería de software. El Software Engineering Body of Knowledge en (IEEE Computer Society, 2004) define la ingeniería de software como: “La aplicación de un enfoque de desarrollo sistemático, disciplinado y cuantificable, para el desarrollo, operación y mantenimiento del software; la aplicación de la ingeniería al software”.

En la actualidad existen algunas varias metodologías de ingeniería de software, definidas con la finalidad de aplicar las mejores prácticas a un entorno específico. Estas metodologías incluyen procesos agiles, métodos, y en ocasiones herramientas, para el desarrollo del proceso de ingeniería de software, tal es el caso del famoso Rational Unified Procces (RUP) (Jacobson, Booch, & James, 1999).

ISSN-2531-2960

No obstante, el CDS cuenta con su propia metodología y estilo de trabajo destilado de las buenas prácticas de diversos marcos de trabajo, estilos y patrones, que contiene los procesos, métodos y varias herramientas de apoyo definidas bajo el modelo CMMI® (Capability Maturity Model® Integration). Este tipo de modelos dirigen sus esfuerzos a documentar las mejores prácticas de calidad del mundo del software, de manera que puedan ser utilizados como marco de referencia para evaluar la capacidad y madurez de las empresas del ramo, tal es el caso del CDS (Chrissis, Konrad, & Shrum, 2011).

Para entender un poco mejor el proceso de ingeniería de software del CDS hay que conocer que el CMMI es un modelo de mejora de madurez de procesos para el desarrollo de productos y servicios. Se compone de las mejores prácticas que componen las actividades de desarrollo y mantenimiento que cubren el ciclo de vida del producto desde su concepción hasta la entrega y mantenimiento. El propósito de CMMI es ayudar a las organizaciones a mejorar su desarrollo y mantenimiento de procesos para productos y servicios. Por otro lado, y aunque no es la finalidad directa del modelo, permite que las organizaciones puedan ser ubicadas en alguno de los diferentes niveles de madurez que presenta el modelo y utilizar esto como marco de referencia para asegurar que aquellas que cuenten con un alto nivel de madurez brindarán productos y servicios de mayor calidad. La manera en que se acredita el grado de madurez de una organización conforme a la estructura de CMMI es realizada mediante una evaluación con base en el "Standard CMMI Appraisal Method for Process Improvement" (SCAMPI). (Carnegie Mellon Software Engineering Institute, 2006).

No obstante, los modelos de madurez como CMMI, incluyen un gran apoyo para controlar la parte administrativa y no para el aspecto de ingeniería.

El modelo CMMI en niveles superiores brinda las guías para desarrollar la capacidad de las áreas de ingeniería de la empresa con mayor calidad, no obstante nunca presenta una clasificación tan clara como la que sugiere de manera similar la IEEE en el cuerpo de conocimiento de la ingeniería de software.

La herramienta CASE que se propone para el estudio en este proyecto aborda el área de codificación de software y se denomina CodeSmith Generator, la cual, mediante el uso de plantillas y un esquema de base de datos inicial, permite generar de manera automática la estructura completa de las clases de software necesarias para ejecutar las operaciones básicas CRUD que se consideren pertinentes.

CodeSmith Generator (CodeSmith Tools, LLC., 2017) es una herramienta de software, generadora de código fuente dirigido por plantillas, que automatiza la creación de código fuente común de aplicaciones para prácticamente cualquier lenguaje de programación. Conforme a (CodeSmith Tools, LLC., 2017) CodeSmith Generator cuenta con las siguientes ventajas en el desarrollo de software:

- Reducción de la codificación repetitiva.
- Generación de código fuente en menos tiempo y con menos defectos.
- Producción de código fuente consistente que se adhiere a los estándares de la empresa.
- Creación de plantillas personalizadas para cualquier lenguaje de programación

ISSN-2531-2960

Actualmente empresas de reconocimiento internacional han demostrado la utilidad práctica de CodeSmith Generator, tal es el caso de nuSoft-solutions, que en el caso de estudio presentado por (Anderson, 2009) menciona que ha logrado reducir en semanas la implementación del marco de trabajo arquitectónico, asegurando haber obtenido de manera inmediata el retorno de su inversión.

La ventaja que distinguió a CodeSmith Generator respecto a otras herramientas revisadas en el estudio fue principalmente la de contar con la posibilidad de definir plantillas para la creación automatizada de código fuente aprovechando los estilos arquitectónicos y patrones de diseño con los que actualmente se construyen los productos de software del CDS, apegándose a sus propios estándares de calidad.

Metodología de Investigación

El estudio realizado se dirigió por la realización de tres grandes actividades alineadas a diversos tipos de investigación:

1. Análisis documental de la arquitectura de un grupo selecto de productos de software del CDS.
2. Evaluación del código fuente generado a partir de la herramienta CASE.
3. Evaluación cuantitativa de la optimización de tiempo y defectos causada por el código fuente generado a partir de las plantillas CASE.

A continuación se presenta el detalle de estas actividades.

Análisis documental de la arquitectura de un grupo selecto de productos de software del CDS

Durante esta etapa se tomó una muestra de productos de software realizados por el CDS, siendo seleccionados aquellos que:

a) Su estructura, estilos y patrones arquitectónicos son similares.

b) Cuentan con una estructura modular con tendencia a ser reutilizada en futuros proyectos.

c) El lenguaje de programación con el cual han sido programados, se presta para volver a ser utilizado en futuros proyectos.

d) Hacen uso de bases de datos relacionales.

Una vez seleccionados los productos de software se realizó un estudio detallado para identificar la estructura común del despliegue de su arquitectura y patrones de diseño para determinar el esquema común clasificando las capas lógicas de los productos de software y clases de software que las constituyen, así como sus responsabilidades, generando un listado de sus características comunes.

Las características comunes de las arquitecturas estudiadas fueron las siguientes:

- Todos los proyectos se dividen en capas lógicas, comúnmente: Datos, Negocios, Servicios y Presentación.
- Es común utilizar los términos Back-End para referirse a las capas lógicas: Datos, Negocios y Servicios.
- Es común utilizar el término Front-End para referirse a la capa lógica de Presentación.

ISSN-2531-2960

- El Front-End es generado con la ayuda de herramientas de diseño integradas en la herramienta del Entorno de Desarrollo integrado (IDE por sus siglas en inglés Integrated Development Environment).

- El código fuente del Back-End es comúnmente generado manualmente siguiendo estilos y patrones adoptados por el CDS cómo buenas prácticas.

- El Back-End difiere en algunos proyectos al ser utilizado un estilo de diseño dirigido por dos tipos de clases de software denominadas Objetos de Acceso a Datos y Objetos planos. En otros casos la mezcla de estos dos tipos de objetos se fusiona en clases denominadas Objetos de Negocio.

- Todas las clases del Back-End utilizan nombres correspondientes a las entidades de la base de datos para generar sus respectivas clases en un modelo orientado a objetos y servicios.

- El código fuente común en todos los proyectos a nivel de Back-End es el correspondiente a los métodos: get, getAll, insert-update y delete que corresponde directamente a las operaciones CRUD anteriormente planteadas.

- Todos los proyectos cuentan con adaptadores propios al sistema manejador de base de datos relacional, predominando el correspondiente a MySQL Server.

- El lenguaje de programación dominante en los proyectos es Visual Basic y se considera pertinente pues permite una migración transparente hacia el lenguaje C# en futuros proyectos bajo la misma plataforma Microsoft .Net.

Evaluación del código fuente generado a partir de la herramienta CASE

Cómo se mencionó anteriormente, la herramienta CASE denominada CodeSmith Generator fue utilizada para cumplir el propósito de generar de manera semiautomática el código fuente que se considerase pertinente. Por lo que fue diseñada una plantilla propia para la herramienta, como puede apreciarse en fragmento en la Figura 1, que permitiese generar código fuente principalmente de las operaciones identificadas a nivel de los métodos de las clases de software de Acceso a Datos en sus respectivas operaciones: get, getAll, insert-update y delete, tanto de entidades de la base de datos simples como aquellas complejas, considerando un sistema gestor de base de datos MySQL Server por defecto y respetando el lenguaje de Visual Basic, con la finalidad de probar los resultados en un proyecto ya construido.

Figura 1 Fragmento de la plantilla elaborada para la herramienta CASE

Fuente: Propia.

Una vez desarrollada la plantilla y puesta en marcha sobre el esquema de base de datos de uno de los proyectos muestra se realizó una revisión técnica del código fuente generado a partir de esta, esto con el fin de verificar lo siguiente:

- El código fuente generado es completo.

- El código fuente generado puede ser cargado directamente por el IDE.

ISSN-2531-2960

- El código fuente generado se encuentra libre de errores (sintácticos, lógicos, etc.)

- El código fuente generado opera de la misma manera que el código fuente del proyecto muestra.

- El código fuente cumple con los estándares de codificación del CDS.

Tras una serie de 3 revisiones, realizando los ajustes correspondientes finalmente se alcanzaron resultados de evaluación satisfactorios.

Evaluación cuantitativa de la optimización de tiempo y defectos causada por el código fuente generado a partir de las plantillas CodeSmith

Una vez realizada la revisión del código fuente generado por la herramienta, se realizó una valoración sobre la posibilidad de uso en una de las etapas de codificación dentro de un proyecto de desarrollo de software. Determinándose, por los resultados mostrados, que la herramienta podría utilizarse en un proyecto de software del CDS considerando que los resultados esperados serían los siguientes:

- Reducción de la cantidad tiempo invertido en la fase de codificación en al menos un 30%.

- Reducción de la cantidad de defectos encontrados durante las revisiones técnica de la fase de codificación en al menos un 30%

El proyecto se puso en marcha, considerando utilizar la herramienta durante las 3 iteraciones del proyecto.

Para realizar esto se tuvo que capacitar de antemano al personal de programación correspondiente para que pudiese utilizar la herramienta en el momento adecuado, es decir una vez superada la etapa de diseño, sobre todo en el diseño del esquema de base de datos. Se plantearon esquemas de trabajo respecto al manejo de la herramienta en casos diversos cómo: Entidades de datos compartidas, manejo de versionamiento de código fuente e integración de cambios.

Se pudo realizar la medición de tiempos e incidencias conforme a las bitácoras de tiempo y registro de revisiones técnicas utilizadas por el CDS, durante las iteraciones de desarrollo del software.

El registro de tiempo se llevó a cabo de acuerdo a lo manejado históricamente conforme al modelo de estimación del CDS, documentado en (Gutiérrez Torres, Martínez López, & Vega Chavez, 2013), en el que se considera modularizar los requisitos de software en Casos de Uso delimitados y que fungen como los bloques de construcción de acuerdo al tipo que se le confiere, teniendo 3 tipos de Casos de Uso: Catálogos, Especiales y Reportes en tres tipos de complejidades distintas: Simple, Promedio y Complejo. Haciendo un total de 9 posibles combinaciones. Incidiendo las mediciones en nuestro estudio principalmente en aquellos Casos de Uso de tipo Catálogo, pues a estos les corresponden directamente las operaciones de objetos de Acceso a Datos las cuales fueron seleccionadas para automatizar su generación de código fuente mediante la herramienta CASE.

ISSN-2531-2960

Resultados

Los hallazgos que se pudieron apreciar durante las iteraciones del proyecto en el que fue utilizada la herramienta de generación automatizada de código fuente CodeSmith Generator y en las que se implementaron principalmente las operaciones comunes de Back-End correspondientes a las clases de Objetos de Acceso a Datos fueron los siguientes:

- Durante las primeras iteraciones aún fueron identificadas algunas anomalías menores en la plantilla de generación de código las cuales fueron corregidas y perfeccionadas sin pormenores con la ayuda del personal del CDS.

- Al finalizar el periodo del estudio se reveló un decremento significativo de aproximadamente 60% del tiempo requerido en la codificación de Casos de Uso de tipo simple (referentes a aquellos con entidades de datos no complejas). Partiendo del dato histórico promedio de 5 horas por Caso de Uso simple sin la herramienta, siendo ahora estos realizados en promedio en aproximadamente 2 horas.

- Se reveló un decremento menor, pero aún significativo de 39.3% del tiempo requerido en la realización de Caso de Uso de tipo promedio (referentes a aquellos con entidades de datos de mediana complejidad). Partiendo del dato histórico promedio de 10.7 horas por Caso de Uso promedio sin la herramienta, siendo estos realizados durante el proyecto en aproximadamente 6.5 horas.

- Se reveló un decremento menor pero aún significativo de un 36% del tiempo requerido en la realización de Caso de Uso de tipo complejo (referentes a aquellos con entidades de datos complejas). Partiendo del dato histórico promedio de 16.43 horas por Caso de Uso complejo sin la herramienta siendo estos realizados en promedio durante el proyecto en aproximadamente 10.5 horas.

- Respecto a las mediciones realizadas sobre defectos encontrados en el proyecto, se pudieron determinar valores que pueden considerarse altamente significativos pues prácticamente no hubo un caso en el que algún programador presentase ajustes por defectos de sintaxis o semántica del lenguaje, o el incumplimiento de estándares de codificación durante las revisiones técnicas de las clases generadas con la herramienta. Respecto a esto se puede aseverar que se logró erradicar en un 100% este tipo de defectos en lo que respecta a clases de los Objetos de Acceso a Datos. No obstante, los defectos detectados durante las revisiones técnicas de código comúnmente no revelan en ambiente de prueba los posibles defectos de las clases generadas una vez que estas entran en ejecución en ambiente de producción, además la integración de las mismas con el Front-End desarrollado implica consideraciones más allá de las mediciones realizadas durante las revisiones técnicas para estas clases, teniendo que escalar esto a realizar un estudio más completo respecto a los defectos encontrados en pruebas de sistema y de aceptación y que puedan ser posiblemente adjudicados a problemas relacionados con el código generado en el Back-End mediante la herramienta CASE.

ISSN-2531-2960

La Tabla 1 muestra un resumen de los resultados presentados finalmente tras la utilización de la herramienta case, mostrando el impacto que fue alcanzado respecto a la diferencia en horas comparando el tiempo histórico del modelo de estimación y el tiempo promedio actual de los distintos tipos de complejidad para casos de uso de tipo Catálogo.

Complejidad de CU	Tiempo histórico (horas)	Tiempo actual (horas)	Diferencia (%)
Simple	**5**	**2**	**60%**
Promedio	**10.7**	**6.5**	**39.3%**
Complejo	**16.43**	**10.5**	**36%**

Tabla 1 Impacto de la herramienta CASE en casos de uso de tipo catálogo

Conclusiones

Los resultados presentados finalmente logran demostrar que la utilización de una herramienta CASE para la generación automática o semiautomática de código fuente, como la que se utilizó para el estudio, permite minimizar de manera significativa la cantidad tiempo invertido en la fase de codificación en proyectos de desarrollo de software, siempre que se tenga una definición clara e institucionalizada de los estilos arquitectónicos y patrones utilizados en la empresa. Es importante mencionar que se realizó la automatización de generación de código fuente de únicamente las operaciones CRUD comunes en el Back-End de los proyectos, dejando esto una posibilidad de ir más allá e intentar un paso más cercano a la generación completa semejante al Scaffolding que realizan algunos marcos de trabajo populares haciendo un barrido desde el Back-End hasta el Front-End.

Por otra parte, mediante el presente trabajo ha sido posible aseverar que es potencial el logro del incremento en la calidad de los productos de software mediante el uso de este tipo de herramientas CASE, pues se ha logrado disminuir la cantidad de defectos encontrados durante las revisiones técnicas de la fase de codificación, al menos en los aspectos revisados respecto al código fuente correspondiente a las operaciones CRUD. Esto nos deja también una posibilidad, como fue mencionado, de realizar un estudio más profundo sobre la relación existente entre los defectos operativos en pruebas de sistema y aceptación y su vinculación con el código fuente generado, el cual podría ser desarrollado en futuros estudios.

Referencias

Anderson, B. (2009). *CodeSmithTools Case Studies nuSoft-solutions.* Dallas, TX: CodeSmithTools .

Carnegie Mellon Software Engineering Institute. (2006). *CMMI for Development v 1.2.* Pittsburgh,PA: Carnegie Mellon University.

Chrissis, M. B., Konrad, M. D., & Shrum, S. (2011). *CMMI: guidelines for process integration and product improvement.* Addison-Wesley Professional.

CodeSmith Tools, LLC. (2017). *CodeSmith Tools*. Recuperado el junio de 2017, de CodeSmith Generator: http://www.codesmithtools.com/product/generator#overview

Dijkstra, E. W. (1972). The Humble Programmer. *Communications of the ACM*, 859-866.

Gutiérrez Torres, L. G., Martínez López, F. J., & Vega Chavez, E. (2013). Estimación Temprana y Seguimiento en Proyectos Iterativos de Desarrollo de Software.

Academia Journals Celaya. Celaya, Guanajuato: Academia Journals .

IEEE Computer Society. (2004). *Guide to the Software Engineering Body of Knowledge.* IEEE Computer Society.

Jacobson, I., Booch, G., & James, R. (1999). *The Unified Software Development Process.* Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.

Martínez López, F. J., Vega Olvera, G. I., & Morales Orozco, D. (2015). Proceso e Impacto Institucional de la Acreditación de CMMI-DEV L3 del Centro de Desarrollo de Software. *ANFEI DIGITAL*, 635-644.

Morales Orozco, D., & Gutiérrez Torres, L. G. (2012). Industria y academia, uniendo. Celaya, Guanajuato: Academia Journals.

Morales Orozco, D., & Gutiérrez Torres, L. G. (2014). Profesores y alumnos inmersos en la implementación de un modelo internacional de procesos de software. *Academia Journals Celaya.* Celaya, Guanajuato: Academia Journals.

Storr, A., & Jarvis, D. (1996). *Software engineering for manufacturing systems: Methods and CASE tools.* Springer. doi:10.1007/978-0-387-35060-8

Wagner, R. (2009). *CodeSmithTools Case Studies Farm Credit Services of America.* Dallas, TX: CodeSmithTools .