UNIVERSIDAD NACIONAL
"PEDRO RUIZ GALLO"
ESCUELA DE POSGRADO

MAESTRÍA EN INGENIERÍA DE SISTEMAS CON MENCIÓN EN GERENCIA DE TECNOLOGÍAS DE LA INFORMACIÓN Y GESTIÓN DEL SOFTWARE

"Diseño de un framework para la creación automática de piezas de software para la implementación de sistemas informáticos empresariales"

TESIS

Presentada para optar el Grado Académico de Maestro en Ingeniería de Sistemas con mención en Gerencia de Tecnologías de la Información y Gestión del Software

AUTOR:

Ing. Terán Santa Cruz, Franklin Edinson

ASESOR:

Ing. Haro Maldonado, Edward Ronald

LAMBAYEQUE - PERÚ

2021

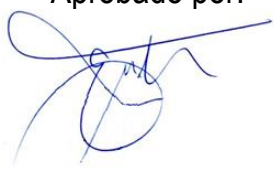
“Diseño de un framework para la creación automática de piezas de software para la implementación de sistemas informáticos empresariales”

Ing. Franklin Edinson Terán Santa Cruz
Autor

Ing. Edward Ronald Haro Maldonado
Asesor

Tesis presentada a la Escuela de Posgrado de la Universidad Nacional Pedro Ruiz Gallo para optar el Grado Académico de: MAESTRO EN INGENIERÍA DE SISTEMAS CON MENCIÓN EN GERENCIA DE TECNOLOGÍAS DE LA INFORMACIÓN Y GESTIÓN DEL SOFTWARE

Aprobado por:



M. Sc. Ernesto Karlo Celi Arevalo

Presidente del Jurado

M. Sc. Jesús Olavarría Paz

Secretario del Jurado

M. Sc. Martin Ampuero Pasco

Vocal del Jurado

Lambayeque, 2021

ACTA DE SUSTENTACIÓN

ACTA DE SUSTENTACIÓN DE TESIS 037

Siendo las 4:00 PM horas del día 25 de Mayo del año Dos Mil Dieciocho, en la Sala de Sustentaciones de la Escuela de Postgrado de la Universidad Nacional Pedro Ruiz Gallo de Lambayeque, se reunieron los miembros del jurado, designados mediante Resolución Nº 279-2017-EPG de fecha 20 DE Marzo de 2017., conformado por:

M.Sc. Ernesto Karlo Celi Arévalo ..... PRESIDENTE (A)
M.Sc. Jesús Bernardo Navarria Paz ..... SECRETARIO (A)
M.Sc. Martin Ampuero Pasco ..... VOCAL
M.Sc. Edward Ronald Haro Maldonado ..... ASESOR (A)

con la finalidad de evaluar la tesis titulada Diseño de un framework para la Creación automática de Pieças de software para la implementación de sistemas informáticos empresariales

presentado por el (la) tesista FRANKLIN EDINSON TEJAN SANTA CRUZ

sustentación que es autorizada mediante Resolución Nº 1054-2018-EPG de fecha 18 DE Mayo de 2018

El Presidente del jurado autorizó el inicio del acto académico y después de la sustentación, los señores miembros del jurado formularon las observaciones y preguntas correspondientes, las mismas que fueron absueltas por el (la) sustentante, quien obtuvo 81 puntos que equivale al calificativo de MUY BUENO

En consecuencia el (la) sustentante queda apto (a) para obtener el Grado Académico de Maestro en Ingeniería de Sistemas con Mención en Gerencia de Tecnologías de la información y Gestión del Software

Siendo las 6:05 PM horas del mismo día, se da por concluido el acto académico, firmando la presente acta.

PRESIDENTE

SECRETARIO

VOCAL

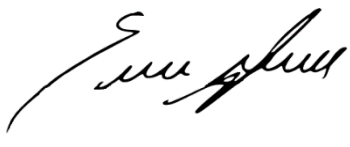
ASESOR

DECLARACIÓN JURADA DE ORIGINALIDAD

Yo, Franklin Edinson Terán Santa Cruz investigador principal, y Edward Ronald Haro Maldonado, asesor del trabajo de investigación "Diseño de un framework para la creación automática de piezas de software para la implementación de sistemas informáticos empresariales", declaramos bajo juramento que este trabajo no ha sido plagiado, ni contiene datos falsos. En caso se demostrara lo contrario, asumo responsablemente la anulación de este informe y por ende el proceso administrativo a que hubiere lugar. Que puede conducir a la anulación del título o grado emitido como consecuencia de este informe.

Lambayeque, 14 de marzo del 2021

Ing. Franklin Edinson Terán Santa Cruz



Ing. Edward Ronald Haro Maldonado

DEDICATORIA

Dedico el fruto de años de trabajo en primer lugar a mis padres quienes me enseñaron a decidir con responsabilidad y calma pensando siempre en salir adelante cumpliendo cada uno mis sueños.

A mi esposa por ser la persona que me entiende y permanece a mi lado apoyándome como sólo ella lo sabe hacer en la consecución de uno de tantos objetivos trazados como profesional y persona.

AGRADECIMIENTOS

Lograr este anhelado objetivo no hubiese posible sin el invalorable apoyo de mi familia, amigos y en especial el de mi esposa que me motiva y apoya permanentemente.

Un agradecimiento especial a mi asesor y docentes jurados que permitieron mejorar el proyecto así como varios aspectos del presente informe.

Es importante también agradecer a los programadores que conformaron los grupos experimentales y permitieron realizar las pruebas necesarias; reiterarles mi agradecimiento por la paciencia y apoyo.

TABLA DE CONTENIDOS

RESUMEN

El presente proyecto está enfocado en la mejora del proceso de desarrollo de software centrándose específicamente en la implementación de sistemas informáticos con acceso a bases de datos relacionales. Este tipo de rutinas implican algoritmos de acceso que son exclusivas para la estructura de la tabla con la que se está trabajando transformando a los programadores en personal casi indispensable para el mantenimiento del software elaborado debido a la complejidad que implica contratar nuevo personal. Así mismo se ha detectado que aunque las piezas de software involucradas son únicas en su código comparten características comunes; por lo que actualmente es común que los programadores copien código existente y lo adapten de acuerdo a sus necesidades.

La propuesta implica estructurar el trabajo de los programadores estableciendo reglas que permitan crear código de calidad sin perder la esencia que hace única a cada pieza del software. La consecuencia por esencia de estructurar la labor de codificación es la posibilidad de automatizar dicho proceso. Para conseguir el objetivo planteado se ha definido un conjunto de reglas para estandarizar el proceso de codificación de procedimientos almacenados y las clases necesarias (entidades y reglas de negocio) que accederán a la base de datos. Dichas reglas han sido el punto de partida para crear la herramienta CASE TheCoder responsable de generar automáticamente las piezas de código indicadas.

Con el fin de verificar la validez de la hipótesis se ha elegido a un grupo de programadores conformados por estudiantes y egresados de la profesional de Ingeniería en Computación e Informática con quienes se ha realizado mediciones del tiempo requerido para codificar el acceso a tablas de bases de datos relacionales. El grupo que ha utilizado la herramienta TheCoder ha disminuido sensiblemente el tiempo requerido de codificación manteniendo la calidad del código obtenido.

Palabras Clave: ingeniería del software, generación de código, herramienta CASE, calidad de código, mantenibilidad

ABSTRACT

The present project is focused on the improvement of the software development process focusing specifically on the implementation of computer systems with access to relational databases. This type of routines implies access algorithms that are unique to the structure of the table with which work is being done, transforming programmers into almost indispensable personnel for the maintenance of the software developed due to the complexity involved in hiring new personnel. Likewise, it has been detected that although the pieces of software involved are unique in their code, they share common characteristics; so it is currently common for programmers to copy existing code and adapt it according to their needs.

The proposal involves structuring the work of programmers establishing rules that allow creating quality code without losing the essence that makes each piece of software unique. The essential consequence of structuring the coding work is the possibility of automating this process. To achieve the stated objective, a set of rules has been defined to standardize the process of coding stored procedures and the necessary classes (entities and business rules) that will access the database. These rules have been the starting point to create the CASE TheCoder tool responsible for automatically generating the indicated pieces of code.

In order to verify the validity of the hypothesis, a group of programmers made up of students and graduates of the computer and IT engineering professional has been chosen with whom measurements have been made of the time required to encode access to base tables of relational data. The group that has used the TheCoder tool has significantly decreased the coding time required while maintaining the quality of the code obtained.

Keywords: software engineering, code generation, CASE tool, code quality, maintainability

INTRODUCCIÓN

El presente proyecto tiene el objetivo de brindar a las empresas de desarrollo de una herramienta específica orientada a disminuir los tiempos de desarrollo necesarios para la creación de sistemas empresariales que consumen datos de un gestor de base de datos relacional. La importancia de las aplicaciones empresariales es indudable y las empresas modernas ejercen una enorme presión sobre los analistas y desarrolladores para entregar el producto solicitado en el menor tiempo posible cumpliendo con las exigencias propias del negocio para el que se trabaja. Es importante mencionar que no sólo se debe cumplir con los requerimientos solicitados sino otorgar el nivel de flexibilidad mínimo necesario para soportar cambios futuros.

El capítulo I describe los problemas que enfrentan las empresas desarrolladoras entre los cuales se puede mencionar la constante rotación de su personal de programación por lo que la dependencia con ellos representa un enorme problema. Estos inconvenientes presentan con gran claridad la necesidad de ordenar el proceso de desarrollo estableciendo normas y recomendaciones que permitan limitar las consecuencias de la tan conocida frase PROGRAMAR ES UN ARTE.

El capítulo II presenta el sustento teórico de la propuesta objeto del presente proyecto: la estandarización del proceso de codificación y la generación automatizada del mismo. La propuesta de estandarización se sustenta en la legibilidad del código lo cual a su vez conlleva a facilitar la mantenibilidad del mismo. Otro de los pilares de la estandarización es la aplicación de buenas prácticas de codificación lo cual se sustenta en la experiencia del investigador en su rol de programador.

En el capítulo III se aborda el proceso de estandarización propuesta detallando las reglas de codificación que se deben tener en cuenta tanto en la creación de los procedimientos almacenados así como en la definición de las clases. Posterior al mencionado proceso se inició el análisis e implementación de la herramienta CASE TheCoder con el objetivo de generar automáticamente el código de acceso a tablas de bases de datos relacionales. Las recomendaciones de codificación así como TheCoder utilizan Visual Basic .Net como lenguaje de programación y MS SQL Server como motor relacional. Asimismo en este ítem se presentan las variables involucradas y sus características así como los indicadores identificados para demostrar la validez de la hipótesis planteada. Para la realización de las pruebas se ha conformado dos grupos experimentales conformados por egresados y estudiantes de la escuela profesional de ingeniería en computación e informática.

El capítulo IV presenta las dos mediciones realizadas a los grupos experimentales a los cuales se les proporcionó bases de datos relacionales con similares características en dos oportunidades. Uno

de los grupos fue capacitado en la utilización de TheCoder para establecer la influencia de la misma en la codificación requerida. Como se puede observar en las mediciones que se presentan el grupo que utiliza la herramienta CASE evidencia una disminución de su tiempo de codificación. Así mismo los integrantes manifiestan que el código generado es fácil de entender porque se sustenta en los estándares detallados en el capítulo previo.

La estandarización del proceso de codificación y la creación de la herramienta CASE han proporcionado al investigador conclusiones importantes acerca de las facilidades que brinda la estructuración del mencionado proceso. Se puede sintetizar lo detallado en el capítulo V indicando que los beneficios no se limitan a disminuir el tiempo requerido.

El capítulo VI describe algunas recomendaciones que proporcionan nuevas aristas y posibilidades de la ingeniería del software en el proceso de codificación como son la creación automatizada de interfaces y la posibilidad de extender la funcionalidad de la herramienta CASE creada.

CAPÍTULO I. ANÁLISIS DEL OBJETO DE ESTUDIO

1.1 CÓMO SURGE EL PROBLEMA

Las tecnologías de información son hoy en día un componente esencial en el desarrollo de nuestra, considerándose como la herramienta más importante de nuestra era, denominada la era de la información.

En el ámbito empresarial los sistemas de información se han consolidado como un recurso de mucha importancia no sólo por agilizar las labores administrativas del personal sino, y más importante aún por generar nuevas posibilidades de negocio, mayor eficiencia en el trato con el cliente, e incluso la creación de nuevos mercados y tipos de empresas. En consecuencia, las empresas de desarrollo de software están constantemente sometidas a la exigencia de desarrollar sistemas de información de calidad en el menor tiempo posible. Es decir, los sistemas no sólo deben cumplir con los requerimientos funcionales y no funcionales solicitados, sino también deben ser entregados en tiempos cada vez más cortos.

La ingeniería del software tiene como objetivo principal identificar los problemas a los que se encuentra sometido el desarrollo de sistemas de información así como determinar las mejores prácticas y criterios para establecer la calidad de un sistema informático. Al respecto varias instituciones, organizaciones, entidades gubernamentales y académicas han enfocado sus esfuerzos en la implementación de estándares y normas internacionales que cubren muchos aspectos relacionados con el ciclo de vida del desarrollo de sistemas de información como por ejemplo las normas ISO 9000, ISO 8402, ISO 14764, CMMI, PMBOK, ITIL entre otras. Así mismo también podemos encontrar recomendaciones para el diseño de la arquitectura de los sistemas (patrones de diseño) e inclusive lo que no se debe realizar (anti patrones de diseño).

Por otro lado, desde el punto de vista empresarial se ha propuesto una corriente enfocada en hacer más eficiente el proceso de desarrollo de software, llamada Fábrica de Software. Esta perspectiva propone estandarizar cada uno de los procesos involucrados, mejorar la capacidad de verificación del trabajo realizado por el personal y lograr una transferencia de know-how más transparente y en consecuencia facilitar el ingreso de personal nuevo así como disminuir la dependencia con el personal existente.

Las empresas de desarrollo y las instituciones de educación de superior de nuestro país están exclusivamente inmersas en el desarrollo de sistemas; y como respuesta a las exigencias del mercado de proveer de sistemas de información de calidad, utilizan herramientas de desarrollo modernas que utilizan las arquitecturas y recomendaciones brindadas por la ingeniería de software.

SOLTI SAC es una empresa dedicada al desarrollo de sistemas informáticos proveyendo a las empresas de la región de sistemas con acceso a bases de datos necesarios para da soporte a sus operaciones diarias. El área actualmente está conformada por un líder de proyectos y dos programadores, los cuales no están exceptuados de las exigencias e inconvenientes expuestos en los párrafos anteriores. En entrevistas realizadas al personal del área y al gerente de la empresa se ha confirmado el retraso en la entrega de los productos solicitados, así como el dificultoso proceso de transferencia de know-how debido a la renuncia y posterior contratación de programadores o analistas.

Como respuesta para atenuar los problemas de codificación existentes se plantea la utilización de herramientas que automaticen la generación de código fuente y por lo tanto disminuya la implementación de sistemas informáticos.

1.2 FORMULACIÓN DEL PROBLEMA

¿De qué manera la herramienta CASE TheCoder puede disminuir el tiempo de codificación de piezas de software con acceso a tablas de bases de datos relacionales?

1.3 JUSTIFICACIÓN E IMPORTANCIA

El presente proyecto se plantea el reto de acortar el proceso de desarrollo de un sistema empresarial con acceso a datos, tomando como punto de partida disminuir el tiempo de codificación de las piezas de software necesarias para gestionar datos de tablas de bases de datos relacionales. Esto permitirá satisfacer al exigente mercado empresarial actual quienes solicitan, en periodos cada vez más cortos, sistemas informáticos de calidad tanto en funcionamiento como en rendimiento.

Así mismo la arquitectura que se presentará es en sí un aporte importante, enmarcado en la Ingeniería del Software; por que incentiva la aplicación de normas de codificación que

contribuyen a mejorar la legibilidad del mismo y realizar un proceso de transferencia de know-how más eficiente entre programadores.

Como consecuencia de establecer normas de codificación las empresas de desarrollo podrán realizar asignaciones eficientes del tiempo de su personal de desarrollo, asignándole tareas con mayor exigencia de acuerdo a su experiencia y capacidad. Es importante destacar también que el periodo de capacitación de nuevo personal se verá drásticamente reducido por la estandarización del proceso de codificación.

1.4 OBJETIVOS

1.4.1 Objetivo general

Desarrollar una herramienta CASE denominada TheCoder que permita disminuir el tiempo necesario para la codificación de piezas de software de un sistema con acceso a datos.

1.4.2 Objetivos específicos

1. Identificar las pautas de codificación de las piezas de software necesarias para el acceso a tablas de base de datos relacional.
2. Estandarizar la codificación de las piezas de software necesarias para el acceso a tablas de una base de datos relacional.
3. Medir los tiempos de codificación de una pieza de software de acceso a tablas de base de datos relacional.
4. Desarrollar la herramienta CASE de generación de código de piezas de software TheCoder, de acceso a tablas de base de datos relacional.
5. Establecer el tiempo de codificación de las piezas de software de acceso a tablas de una base de datos relacional después de utilizar la herramienta CASE TheCoder.

CAPÍTULO II. MARCO TEÓRICO

2.1 ANTECEDENTES

2.1.1 Generación de código a partir de modelos parametrizados

El autor en su proyecto de tesis, enfatiza que la responsabilidad de los programadores está basada principalmente en su capacidad analítica; sin embargo se pueden identificar partes del programa que son similares y en algunos casos idénticos. Estas partes, denominadas assets, sirven como base para la crear modelos los cuales son parametrizados y luego utilizados para generar código que respete la funcionalidad del asset adicionándole características particulares. Así mismo, desarrolla un lenguaje, el cual es utilizado para insertar los parámetros necesarios en la generación del código fuente. (Paiva, 2011)

2.1.2 Herramienta CASE para la generación de código C++ a partir de diagramas de clase UML

Como concluye y se evidencia en el producto entregado por los autores de la tesis, el uso de normas de codificación y buenas prácticas de desarrollo facilitan la creación de sistemas informáticos de calidad. Así mismo, la utilización de herramientas CASE de generación de código ayudan a disminuir el tiempo de codificación necesario para desarrollar un software. (Cruz, 2003)

2.1.3 Generación automática de prototipos funcionales a partir de esquemas preconceptuales

El autor de este proyecto de investigación hace énfasis en los inconvenientes inherentes al proceso de recolección y diseño de interfaces, así como los retrasos que se dan en esta etapa del desarrollo de sistemas informáticos. Como alternativa propone el uso de esquemas preconceptuales, los cuales a su vez permitirán generar los diagramas UML, que formaran parte de la documentación; así como el código fuente necesario para su utilización en la etapa de programación. El código fuente generado es para una aplicación web. (Chaverra Mojica, 2011)

2.1.4 Una herramienta CASE para el diseño y la generación de la estructura estática de la base de datos

En esta publicación los autores hacen énfasis en la utilización no sólo de una herramienta CASE, sino también en la aplicación de un método de diseño de base de datos. La utilización de ambas herramientas permite al grupo de desarrollo no disminuir

el tiempo de codificación así como la creación de la base de datos y al mismo tiempo asegurar un nivel mínimo de calidad en su construcción en cuanto a la coherencia del diseño de la base de datos. Además, concluye la importancia de hacer uso de notaciones estándares y herramientas comerciales actuales, lo cual permite mayor interoperabilidad y menor impacto en el aprendizaje de la herramienta CASE.

2.2 BASE TEÓRICA

2.2.1 Ingeniería del software

Es la disciplina orientada a analizar, detectar debilidades y proponer mejoras en cada una de las etapas del proceso de desarrollo de software. (Sommerville, 2011, pág. 7)

Para alcanzar su objetivo hace uso de diferentes herramientas de manera coherente buscando no sólo alcanzar soluciones sino, y principalmente, de identificar soluciones eficientes.

Como consecuencia de la labor realizada por los expertos se han propuesto normas que facilitan la comunicación del personal encargado del desarrollo favoreciéndose de esta manera el mantenimiento correctivo y preventivo del software.

a. Reutilización de código fuente

Esta técnica permite utilizar el código fuente creado por un grupo de trabajo en el desarrollo de un nuevo software. De esta forma se disminuye el tiempo de codificación necesario así como la inversión económica a realizar. (Kendall & Kendall, 2005, pág. 714)

b. Patrones de diseño

Los patrones de diseño representan la reutilización del trabajo de otras personas, la cual no sólo está limitada a hacer uso del código generado, sino también la abstracción realizada y la arquitectura consecuencia de ésta.

Los patrones proponen pautas de solución para problemas conocidos mas no la solución completa. Los patrones de diseño se sustentan en la programación orientada a objetos y utiliza UML para su representación. (Stelting & Maassen, 2003)

c. Notación

Es un conjunto de recomendaciones enfocadas en mejorar la legibilidad del código estableciendo normas para la escritura de los diferentes nombres de elementos que constituyen un programa: variables, funciones, clases y paquetes. Estas recomendaciones determinan los caracteres a utilizar y en que combinaciones son

permitidas para cada uno de los nombre de los elementos indicados. (Martin, 2009)

A lo largo de los años han existido diferentes notaciones íntimamente vinculadas a los lenguajes de programación sobre los cuales se proponen y con un objetivo específico dando prioridad en ocasiones a brindar información y en otros casos para facilitar la escritura. Hoy en día los IDE modernos proveen herramientas que facilitan la escritura y la obtención de información de los elementos involucrados por lo que la prioridad de las notaciones se ha trasladado a facilitar la legibilidad del código.

Actualmente existen dos notaciones importantes: Camell y Pascal.

2.2.2 Fábrica de software

Existen dos puntos de vista con respecto al concepto de Fábrica de Sotware. Una que hace referencia a un software responsable de generar código fuente de forma parcial o, idealmente, total para la creación de un nuevo software.

Sin embargo el concepto más ampliamente aceptado, proviene desde el ámbito empresarial y caracteriza a una empresa dedicada a la creación de software en periodos cortos, que cumple con los requerimientos funcionales y no funcionales; y que además satisface las normas de codificación. (Piattini & Garzas, 2007)

Las fábricas de software tienen su origen en la insatisfacción del mercado con respecto a la generación de nuevas aplicaciones. Actualmente la industria del software atraviesa los siguientes problemas:

- Falta de personal cualificado
- Bajo porcentaje de éxito en los proyectos
- Evolución constante de las tecnologías

2.2.3 Herramienta CASE

a. Concepto

Computer Aided Software Engineering: Ingeniería de Software Asistida por Computadora.

Herramientas software creadas para facilitar el proceso de desarrollo de sistemas informáticos en alguna o varias de las etapas del ciclo de vida del desarrollo reduciendo el tiempo necesario y por lo tanto disminuyendo tiempo y costo. (Herramienta CASE, 2017)

b. Objetivos

- Mejorar la productividad
- Aumentar calidad del software
- Reducir tiempo y costo de desarrollo y mantenimiento
- Mejorar planificación del proyecto
- Automatizar el proceso de desarrollo
- Aumentar la biblioteca de conocimiento
- Automatizar el desarrollo de software
- Ayuda a la reutilización, portabilidad y estandarización de la documentación
- Gestión global en todas las fases del desarrollo
- Facilitar el uso de las distintas metodologías

c. Clasificación

Las herramientas CASE puede clasificarse por diversos criterios. Para el presente proyecto se considera la clasificación en base a las fases del ciclo de desarrollo que cubre la herramienta. (Herramienta CASE, 2017)

- Upper Case (U-CASE): enfocadas en las etapas de planificación, análisis de requisitos y estrategias de desarrollo.
- Middle Case (M-CASE): Sirven de apoyo en las fases de análisis y diseño de la aplicación.
- Lower Case (M-CASE): se utilizan en la etapa de codificación y permiten la generación de código, detección de errores, depuración de programas y pruebas de las mismas

CAPÍTULO III. MARCO METODOLÓGICO

3.1 DESARROLLO DE LA PROPUESTA

3.1.1 Estandarización de código

La propuesta central del proyecto implica estructurar el trabajo de codificación de los programadores por lo que el primer paso que se realizó es establecer un conjunto de reglas y recomendaciones tanto a nivel de base de datos como a nivel del lenguaje de programación que se utiliza.

Las recomendaciones establecidas se detallan a continuación:

BASE DE DATOS

El acceso a tablas de base de datos relacionales implica según la experiencia del investigador tres operaciones fundamentales: registro, actualización y lectura. No se recomienda la creación de rutinas de eliminación por la potencial pérdida de datos que generen problemas de seguridad.

Procedimientos almacenados

Considerando estas operaciones se ha definido las siguientes reglas para los nombres de procedimientos almacenados:

pr_<Prefijo de operación><Nombre de Tabla>

<*Prefijo de operación*>*:* determina la operación que se va a realizar. Puede tener los siguientes valores:

- i : procedimiento de inserción
- a: procedimiento de actualización
- l: procedimiento de lectura. Esta operación implica la recuperación de una solo fila de datos o ninguna es decir se utiliza la clave primaria como parte de la condición
- li: procedimiento de lectura que sirve para obtener cero o más filas de datos

<*Nombre de Tabla*>: es el nombre de la tabla sobre la cual se realizará la operación

Ejemplo: si se está trabajando con la tabla Persona los procedimientos deberían crearse con los siguientes nombres: pr_iPersona, pr_aPersona, pr_lPersona y pr_liPersona

La cantidad y tipo de los parámetros de los procedimientos almacenados depende completamente de la estructura de la tabla sobre la cual se está trabajando.

LENGUAJE DE PROGRAMACIÓN

El proyecto se ha centrado en la especificación de reglas haciendo uso de Visual Basic Net como lenguaje de programación. Se ha tomado como referencia la arquitectura cliente/servidor para el desarrollo de aplicaciones de escritorio la cual define 3 capas: interfaz, negocio y datos. La estandarización se ha trabajado en la capa de negocio es decir en la creación de las entidades y reglas de negocio en base a la experiencia del investigador en su rol de programador.

Teniendo en cuenta estas consideraciones se han establecido las siguientes reglas:

Clases Entidad

- Se crea una entidad para cada tabla de la base de datos utilizando el nombre la misma como nombre de la clase.
- Se crea una propiedad de lectura y escritura por cada campo de la tabla utilizando su nombre. El tipo de dato se obtiene en base al tipo de dato de la columna y la compatibilidad documentada por el Net Provider correspondiente. Esta regla no aplica a los campos que se han definido como clave foránea.
- Los campos creados como clave foránea se representan como un atributo cuyo tipo de dato es una referencia a una clase que representa a la tabla con la cual se ha creado la clave foránea.

Clases Reglas de Negocio

- Se crea una clase de regla de negocio por cada tabla cuyo nombre está compuesto de la siguiente manera: RN<Nombre de tabla>. Por ejemplo si se trabaja con la tabla Producto el nombre de la regla de negocio será RNProducto.
- La clase contendrá 04 funciones cuyos nombres y firmas se detallan a continuación:
  - Public Void Registrar(nombreTabla AS Entidad)
  - Public Void Actualizar (nombreTabla AS Entidad)
  - Public Function Leer(nombreTabla AS Entidad) AS Entidad
  - Public Function Listar ( ) AS List(Of Entidad)

Es importante aclarar que las funciones antes indicadas utilizarán los procedimientos almacenados cuyos nombres y características han sido establecidos en el ítem Base de datos.

3.1.2 Especificaciones de funcionalidad

La creación de la herramienta CASE TheCoder implica satisfacer los siguientes requerimientos funcionales:

Base de datos: MS SQL Server 2008

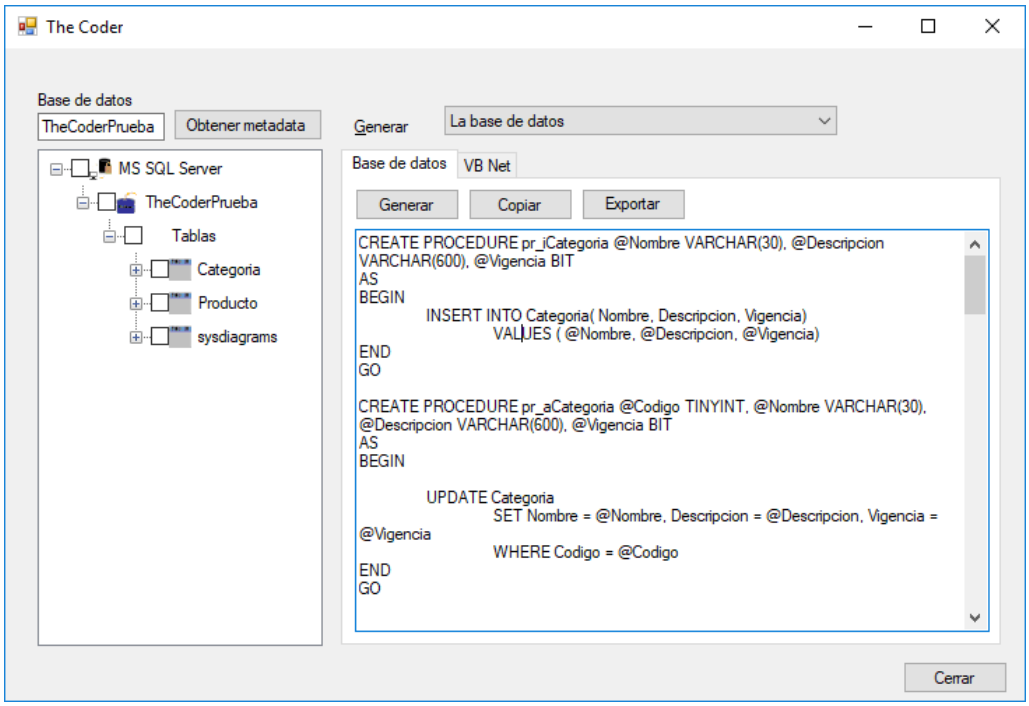
- Flexibilidad para elegir la base de datos.
- Identificación de tablas así como las columnas que la conforman y los tipos de datos correspondientes.
- Identificar la clave primaria de cada tabla.
- Identificar los campos que conforman las claves foráneas y las tablas con las cuales se crea la dependencia.
- La interfaz debe generar el código indicado y permitir las siguientes posibilidades de interacción:
    - Presentarlo en pantalla para seleccionarlo, editarlo o copiarlo manualmente
    - Opción para almacenar el código generado en el Portapapeles de MS Windows facilitando su edición en los editores de su preferencia
    - Generar un archivo de texto con el código generado considerando el tipo de codificación que utiliza MS SQL Server Management Studio 2008

Lenguaje de programación: Visual Basic Net 2010

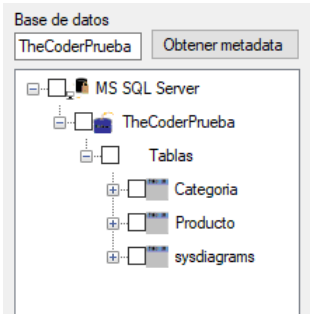
- Crear las clases entidades de negocio considerando las tablas y características de las columnas de cada una.
- Crear las clases reglas de negocio considerando las tablas. La generación del código debe considerar la entidad correspondiente así como los procedimientos almacenados con los que debe interactuar.
- La interfaz debe generar el código indicado y permitir las siguientes posibilidades de interacción:
    - Presentarlo en pantalla para seleccionarlo, editarlo o copiarlo manualmente
    - Opción para almacenar el código generado en el Portapapeles de MS Windows facilitando su edición en los IDE que el usuario desee
    - Generar un archivo de texto con el código generado considerando el tipo de codificación que utiliza el IDE Visual Studio 2010 así como la identación correspondiente que facilite su edición. Esta opción facilita la integración de los archivos generados en las soluciones y proyectos que VS 2010 gestiona.

3.1.3 Prototipos

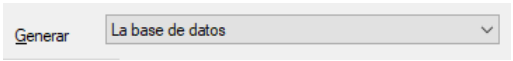
La herramienta CASE TheCoder creada es una aplicación SDI es decir consta de una ventana principal que proporciona todas las herramientas descritas en los ítem Especificaciones de funcionalidad.



A continuación se describen los componentes de la interfaz

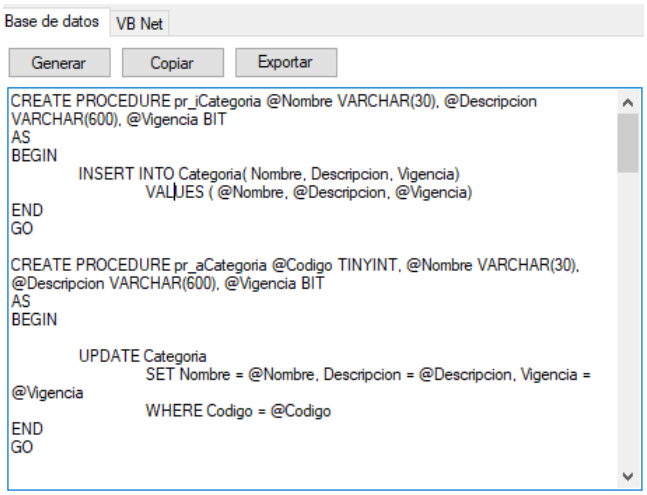


Esta parte de la interfaz permite indicar la base de datos y haciendo uso del botón "Obtener metadata" se listarán las tablas que conforman la base de datos elegida así como las columnas que componen cada una de las tablas. Se ha utilizado iconos que permiten identificar visualmente las características principales de las columnas: clave primara y foránea.



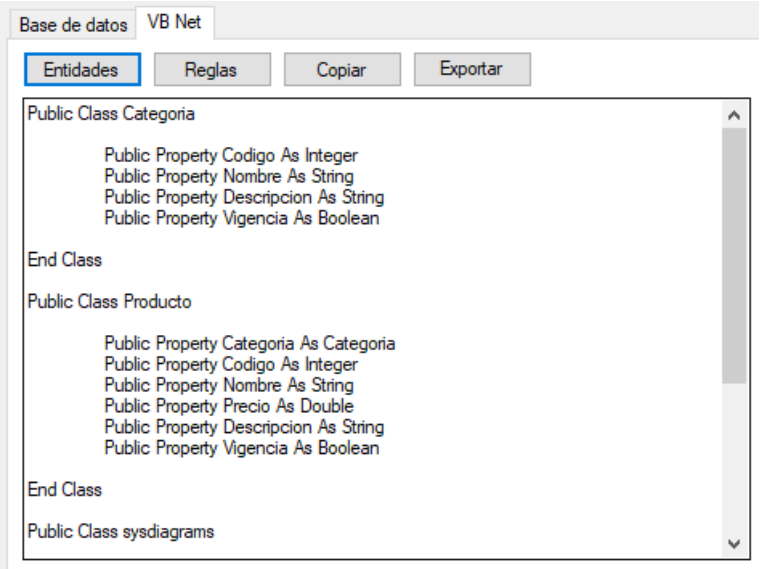
Este componente permite elegir como se realizará la generación de código. Presenta las siguientes opciones:

- La base de datos: genera el código necesario (procedimientos y clases) para todas las tablas de la base de datos seleccionada.
- Sólo las tablas seleccionadas: genera el código requerido (procedimientos y clases) para las tablas que han sido seleccionadas en el árbol que se presenta en el componente de la izquierda.



En esta parte de la interfaz haciendo uso del botón “Generar” se presenta el código de los procedimientos almacenados que se crearán para MS SQL Server 2008.

Esta interfaz también contiene los botones “Copiar“ que almacena el código generado en el portapapeles de MS Windows y el botón “Exportar” para generar el archivo con extensión sql compatible con MS SQL Server Management Studio.



Este es el último componente de la interfaz que permite la generación de las clases que se utilizaran para acceder a las tablas de la base de datos seleccionada. Presenta los botones “Entidades” y “Reglas” que permiten crear las clases entidades y reglas de negocio según las especificaciones funcionales.

También muestra el botón “Copiar” que permite almacenar el código que se visualiza en el portapapeles de MS Windows.

Por último se encuentra el botón “Exportar” para generar los archivos necesarios y compatibles con MS Visual Studio 2010; todos los archivos generados tendrán la extensión vb característico del lenguaje Visual Basic Net 2010.

Es importante mencionar que TheCoder generará un archivo por cada clase requerida.

3.2 VARIABLES

3.2.1 Variable independiente

Herramienta CASE ad-hoc de generación de código: TheCoder

3.2.2 Variable dependiente

Tiempo de codificación de piezas de software de acceso a datos de tablas de base de datos relacional

3.2.3 Definición conceptual de variables

- ✓ **Herramienta CASE ad-hoc de generación de código (TheCoder):** herramienta CASE a desarrollar para generar código con las características estandarizadas

- ✓ **Tiempo de codificación de piezas de software de acceso a datos de tablas de bases de datos relacional:** tiempo necesario para la creación de los componentes necesarios para realizar las operaciones fundamentales (registro, actualización, eliminación y consulta), de tablas de bases de datos relacionales.

3.2.4 Definición operacional de variable

VARIABLE	DIMENSIÓN	INDICADORES	TÉCNICAS
Tiempo de codificación de piezas de software de acceso a datos de tablas de bases de datos relacional	Base de datos	- Tiempo de codificación de procedimientos almacenados y clases (entidades y reglas)	- Observación
	Lenguaje de programación		

3.2.5 Caracterización de la variable: herramienta CASE TheCoder

TheCoder es una herramienta de generación de código que debe cumplir con los siguientes requerimientos:

- ✓ Detectar las tablas de la base de datos de un servidor de base de datos
- ✓ Generar los archivos necesarios con la codificación de los procedimientos almacenados necesarios para la gestión de los datos de las tablas solicitadas
- ✓ Generar los archivos con la codificación de las clases necesarias para la gestión de los datos de las tablas seleccionadas. Esta codificación debe ser coherente con la codificación de los procedimientos almacenados
- ✓ Los archivos generados deben ser compatibles con las herramientas de desarrollo correspondientes al servidor de base de datos, así como del entorno de desarrollo
- ✓ La codificación debe respetar los estándares de codificación establecidos

3.3 DISEÑO DE CONTRASTACIÓN DE HIPÓTESIS

La presente investigación al ser de tipo pre experimental, que tendrá como uno de sus productos la herramienta CASE TheCoder; se plantea contrastar la hipótesis de la siguiente manera:

GE1: O1 → X → O2

GE2: O1 → O2

O1 = Tiempo de codificación manual a realizar antes de la implantación TheCoder

X = Herramienta CASE TheCoder

O2 = Tiempo de codificación apoyado por TheCoder, a realizarse después de la capacitación en el uso de la herramienta

Las unidades de análisis están conformados por alumnos y egresados de Ing. Computación e Informática.

3.4 POBLACIÓN Y MUESTRA

La población la constituyen todos los programadores que implementan sistemas informáticos haciendo uso de gestores de bases de datos relacionales.

La muestra ha sido elegida intencionalmente entre estudiantes y egresados de la escuela profesional de Ingeniería en Computación e Informática con la finalidad de garantizar su experiencia en programación.

Los grupos han sido seleccionados al azar considerando su homogeneidad en velocidad promedio de tipeo. Las mediciones se han realizado utilizando la herramienta Test de velocidad de la página web www.cursomeca.com cuyos certificados forman parte de los anexos del presente proyecto.

GE1: Grupo experimental 1

Nº	Desarrollador	Velocidad promedio (ppm)
1.	Sirlopu Cumpa Angel	289
2.	Altamirano Balcazar, Christopher	248
3.	Urupeque Sanchez Richard	198
	Velocidad promedio	**245**

GE2: Grupo experimental 2

Nº	Desarrollador	Velocidad promedio (ppm)
1.	Bobadilla Farfán Cristina	394
2.	Ruiz Figueroa, Adrián	270
3.	Gonzales Reluz, Manuel	220
	Velocidad promedio	**249.66**

3.5 MATERIALES, TÉCNICAS E INSTRUMENTOS DE RECOLECCIÓN DE DATOS

- ✓ Observación
- ✓ Reloj
- ✓ Entrevistas
- ✓ Formatos de recolección de datos de mediciones
- ✓ Test de velocidad: www.cursomeca.com

3.6 MÉTODOS Y PROCEDIMIENTOS PARA LA RECOLECCIÓN DE DATOS

Observación de campo.

La recolección de las mediciones realizadas se ha registrado en una hoja de cálculo de Microsoft Excel

3.7 ANÁLISIS Y DISCUSIÓN DE LOS RESULTADOS DE LOS INSTRUMENTOS UTILIZADOS

La investigación planteada tiene como objetivo disminuir el tiempo de codificación de las piezas de software. Para la demostración se presentará lo siguiente:

- Promedios
- Máximos y mínimos

CAPÍTULO IV. RESULTADOS Y DISCUSIÓN

Con la finalidad de verificar la hipótesis planteada se han realizado dos pruebas experimentales a los dos grupos que conforman la muestra.

GE1: conformado por tres programadores que utilizarán TheCoder en la segunda prueba para establecer las diferencias con el primer experimento en el cual programarán de manera tradicional.

GE2: conformado por tres programadores que codificarán de manera tradicional en ambas pruebas.

PRIMER EXPERIMENTO

En las tablas se presentan los tiempos requeridos por cada programador para implementar una ventana que permita realizar las operaciones básicas de una tabla: registro, actualización y búsqueda de datos.

Para una comprensión más detallada el indicador Tiempo de codificación de procedimientos almacenados y clases se ha dividido en dos ítems: codificación de tablas sin clave foránea y con claves foráneas.

Programadores GE 1	Tablas de bases de datos					Tiempo Promedio de codificación Tabla sin FK (minutos)	Tiempo Promedio de codificación Tabla con FK (minutos)
	Personal	Usuario	Categoria	Producto	Tiempo de codificación (minutos)		
Sirlopu Cumpa Angel	85	22	21	21	149	53.00	21.50
Altamirano Balcazar, Christopher	45	27	13	25	110	29.00	26.00
Urupeque Sanchez Richard	35	23	18	26	102	26.50	24.50
				Promedio	**120.33**	**36.17**	**24.00**

Esta tabla presenta los datos del segundo grupo experimental obteniéndose que el tiempo de codificación máximo es de 149 y el más corto es de 102 minutos.

Programadores GE 2	Tablas de bases de datos					Tiempo Promedio de codificación Tabla sin FK (minutos)	Tiempo Promedio de codificación Tabla con FK (minutos)
	Personal	Usuario	Categoria	Producto	Tiempo de codificación (minutos)		
Bobadilla Farfan Cristina	40	33	30	18	121	35.00	25.50
Ruiz Figueroa Adrian	37	14	58	37	146	47.50	25.50
Gonzales Reluz Manuel	40	15	33	61	149	36.50	38.00
				Promedio	**138.67**	**39.67**	**29.67**

Como se puede corroborar en la tabla el tiempo máximo de codificación es 149 minutos y el tiempo más corto es 121 minutos.

Al comparar los resultados de ambos grupos se puede evidenciar tiempos similares para realizar la codificación requerida. Al comparar las dos últimas columnas que agrupan los tiempos requeridos por tablas con similares características podemos verificar que hay una diferencia de 210 segundos para codificar las tablas sin clave foránea y de 340 segundos en las tablas que incluyen clave foránea.

RESUMEN	GE1	GE2	Diferencia	%
Tiempo total promedio	120.33	138.67	-18.33	86.78
Tiempo promedio máximo	149.00	149.00	0.00	100.00
Tiempo promedio mínimo	102.00	121.00	-19.00	84.30
Tiempo promedio para tablas sin FK	36.17	39.67	-3.50	91.18
Tiempo promedio para tablas con FK	24.00	29.67	-5.67	80.90

Esta tabla muestra un resumen de los valores obtenidos en la primera medición. Se puede destacar que los tiempos de codificación de ambos grupos son similares evidenciándose un máximo de 19 minutos de diferencia en el tiempo promedio mínimo de codificación.

SEGUNDO EXPERIMENTO

Para finalizar el proyecto se realizó un experimento adicional el cual incluye en el primer grupo la utilización de TheCoder como herramienta para agilizar el proceso de codificación. Es importante mencionar que los integrantes de este grupo fueron capacitados previamente para que sea utilizado.

Programadores GE 1	Tablas de bases de datos					Tiempo Promedio de codificación Tabla sin FK (minutos)	Tiempo Promedio de codificación Tabla con FK (minutos)
	Personal	Usuario	Categoria	Producto	Tiempo de codificación (minutos)		
Sirlopu Cumpa Angel	13	12	10	10	45	11.50	11.00
Altamirano Balcazar, Christopher	18	18	9	13	58	13.50	15.50
Urupeque Sanchez Richard	20	15	15	13	63	17.50	14.00
				Promedio	**55.33**	**14.17**	**13.50**

Esta tabla presenta los datos del segundo grupo experimental obteniéndose que el tiempo de codificación máximo es de 63 y el más corto es de 45 minutos.

Programadores GE 2	Tablas de bases de datos					Tiempo Promedio de codificación Tabla sin FK (minutos)	Tiempo Promedio de codificación Tabla con FK (minutos)
	Personal	Usuario	Categoria	Producto	Tiempo de codificación (minutos)		
Bobadilla Farfan Cristina	35	22	15	25	97	25.00	23.50
Ruiz Figueroa Adrian	30	17	11	20	78	20.50	18.50
Gonzales Reluz	39	25	13	28	105	26.00	26.50
				Promedio	**93.33**	**23.83**	**22.83**

Como se puede corroborar en la tabla el tiempo máximo de codificación es 105 minutos y el tiempo más corto es 78 minutos.

Como se puede corroborar en las tablas del segundo experimento existe una diferencia notoria en el tiempo promedio total de codificación así como en las mediciones por cada tipo de tabla.

RESUMEN	GE1	GE2	Diferencia	%
Tiempo total promedio (minutos)	55.33	93.33	-38.00	59.29
Tiempo promedio máximo (minutos)	63.00	105.00	-42.00	60.00
Tiempo promedio mínimo (minutos)	45.00	78.00	-33.00	57.69
Tiempo promedio para tablas sin FK (minutos)	14.17	23.83	-9.67	59.44
Tiempo promedio para tablas con FK (minutos)	13.50	22.83	-9.33	59.12

La tabla resumen presenta los resultados de ambos grupos en el segundo experimento; se puede constatar que los tiempos necesarios vistos desde diferentes perspectivas demuestran que la estandarización del proceso de codificación y sobretodo su automatización afectan positivamente el tiempo requerido.

La utilización de la herramienta CASE Thecoder libera en casi su totalidad la creación del código necesarios para entidades, reglas y procedimientos almacenados dejando al programador el diseño de las interfaces de usuario y las llamadas a las funciones generadas automáticamente. Case resaltar que el código generado es fácil de editar y adaptar dado que está basado en las reglas y recomendaciones indicadas en el ítem Marco Metodológico.

CAPÍTULO V. CONCLUSIONES Y RECOMENDACIONES

5.1 CONCLUSIONES

Durante el desarrollo del proyecto se ha evidenciado las ventajas que proporciona la estandarización del proceso de codificación y su automatización. Entre las principales conclusiones se pueden mencionar las siguientes:

1) El uso de estándares de codificación (notaciones) facilitó el proceso de codificación así como su legibilidad.

2) La automatización del proceso de codificación es completamente dependiente de la formalización del mismo y él código fuente generado debe respetar completamente las normas establecidas para facilitar su integración.

3) El modelo relacional de base de datos y el diagrama de clases que se utilizan al implementar un sistema representan aspectos diferentes de una aplicación sin embargo están íntimamente relacionados y su interrelación fue considerada minuciosamente al implementar TheCoder.

5.2 RECOMENDACIONES

El desarrollo de aplicaciones y en especial de sistemas empresariales debe cumplir con las exigencias del mercado. Esta incesante necesidad de mejorar los procesos no es exclusiva de las instituciones tradicionales sino también están afectando a las empresas proveedoras de sistemas informáticos por lo que se recomienda:

1) Utilizar UML para representar los elementos estáticos y dinámicos de los sistemas empresariales con el objetivo de posibilitar una mayor flexibilidad en la generación de código.

2) Formalizar el proceso del diseño de interfaces para su posterior automatización y vinculación con la herramienta TheCoder expuesta en el presente proyecto.

3) Crear herramientas CASE generadoras de código personalizadas a las características de cada empresa de desarrollo.

Bibliografía general

- Chaverra Mojica, J. J. (2011). *Generación automática de prototipos funcionales a partir de esquemas preconceptuales.* Universidad Nacional de Colombia. Medellin: Universidad Nacional de Colombia.

- Cruz, I. (2003). *Herramientas CASE para la generación de código C++ a partir de diagramas de clase UML.* Mexico: Universidad Tecnológica de Mixteca.

- *Herramienta CASE*. (20 de 10 de 2017). Recuperado el 26 de 11 de 2017, de Herramienta CASE: https://es.wikipedia.org/wiki/Herramienta_CASE

- Kendall, K. E., & Kendall, J. E. (2005). *Análisis y Diseño de Sistemas* (6 ed.). (A. Nuñez Ramos, Trad.) Mexico, Mexico: Pearson Educación.

- Martin, R. C. (2009). *Código limpio. Manual de estilo para el desarrollo ágil de software.* (J. L. Gomez Celador, Trad.) Madrid, España: Ediciones Anaya Multimedia.

- Monjes, S. (1996). *Herramientas CASE para ingeniería de software.* Guatemala: Universidad Francisco Marroquín.

- Paiva, A. (2011). *Generación de código a partir de modelos parametrizados.* Chile: Universidad de Concepción.

- Piattini, M., & Garzas, J. (2007). *Fábricas de software: Experiencias, tecnologías y organización.* Mexico, Mexico: Alfaomega Grupo Editor. doi:978-970-15-1315-6

- Sommerville, I. (2011). *INGENIERÍA DE SOFTWARE* (9 ed.). (L. M. Cruz Castillo, Ed.) Estado de México, México,.

- Stelting, S., & Maassen, O. (2003). *Patrones de diseño aplicados a Java.* Madrid, España: Pearson Educación.

ANEXOS

ANEXO 01: CERTIFICADOS q DE VELOCIDAD

Para la selección de los integrantes de cada grupo experimental se ha utilizado la herramienta Test de velocidad de la página www.cursomeca.com la cual expide un certificado acerca de las características de tipeo de la persona que realiza la prueba. Se anexa los dieciocho (18) certificados de los participantes a los cuales se les evaluó tres veces para obtener su velocidad promedio.

Cursomeca
www.cursomeca.com

CERTIFICADO

Christopher Jeffrey Altamirano Balcazar ha alcanzado una velocidad de **254 PPM** (Pulsaciones Por Minuto) en nuestros tests y por el cual se extiende el presente certificado. Test de velocidad de mecanografía realizado en cursomeca.com

Fecha de emisión: 01 de agosto del 2017

Número de certificado: C00222475
Código de validación: 860487

Para comprobar la autenticidad de este certificado introduzca el número de certificado y código de validación en www.cursomeca.com/certificados/

Cursomeca
www.cursomeca.com

CERTIFICADO

Christopher Jeffrey Altamirano Balcazar ha alcanzado una velocidad de **250 PPM** (Pulsaciones Por Minuto) en nuestros tests y por el cual se extiende el presente certificado. Test de velocidad de mecanografía realizado en cursomeca.com

Fecha de emisión: 01 de agosto del 2017

Número de certificado: C00222472
Código de validación: 696414

Para comprobar la autenticidad de este certificado introduzca el número de certificado y código de validación en www.cursomeca.com/certificados/

Cursomeca
www.cursomeca.com

CERTIFICADO

Christopher Jeffrey Altamirano Balcazar ha alcanzado una velocidad de **239 PPM** (Pulsaciones Por Minuto) en nuestros tests y por el cual se extiende el presente certificado. Test de velocidad de mecanografía realizado en cursomeca.com

Fecha de emisión: 01 de agosto del 2017

Número de certificado: C00222473
Código de validación: 749649

Para comprobar la autenticidad de este certificado introduzca el número de certificado y código de validación en www.cursomeca.com/certificados/

Cursomeca
www.cursomeca.com

CERTIFICADO

Claudia Cristina Bobadilla Farfán ha alcanzado una velocidad de **386 PPM** (Pulsaciones Por Minuto) en nuestros tests y por el cual se extiende el presente certificado. Test de velocidad de mecanografía realizado en cursomeca.com

Fecha de emisión: 25 de julio del 2017

Número de certificado: C00221647
Código de validación: 846291

Para comprobar la autenticidad de este certificado introduzca el número de certificado y código de validación en www.cursomeca.com/certificados/

Cursomeca
www.cursomeca.com

CERTIFICADO

Claudia Cristina Bobadilla Farfán ha alcanzado una velocidad de **396 PPM** (Pulsaciones Por Minuto) en nuestros tests y por el cual se extiende el presente certificado. Test de velocidad de mecanografía realizado en cursomeca.com

Fecha de emisión: 25 de julio del 2017

Número de certificado: C00221646
Código de validación: 791418

Para comprobar la autenticidad de este certificado introduzca el número de certificado y código de validación en www.cursomeca.com/certificados/

Cursomeca
www.cursomeca.com

CERTIFICADO

Claudia Cristina Bobadilla Farfán ha alcanzado una velocidad de **400 PPM** (Pulsaciones Por Minuto) en nuestros tests y por el cual se extiende el presente certificado. Test de velocidad de mecanografía realizado en cursomeca.com

Fecha de emisión: 25 de julio del 2017

Número de certificado: C00221645
Código de validación: 738001

Para comprobar la autenticidad de este certificado introduzca el número de certificado y código de validación en www.cursomeca.com/certificados/

Cursomeca
www.cursomeca.com

CERTIFICADO

Adrián Ernesto Ruiz Figueroa ha alcanzado una velocidad de **273 PPM** (Pulsaciones Por Minuto) en nuestros tests y por el cual se extiende el presente certificado. Test de velocidad de mecanografía realizado en cursomeca.com

Fecha de emisión: 29 de agosto del 2017

Número de certificado: C00225580
Código de validación: 930375

Para comprobar la autenticidad de este certificado introduzca el número de certificado y código de validación en www.cursomeca.com/certificados/

Cursomeca
www.cursomeca.com

CERTIFICADO

Adrián Ernesto Ruiz Figueroa ha alcanzado una velocidad de **277 PPM** (Pulsaciones Por Minuto) en nuestros tests y por el cual se extiende el presente certificado. Test de velocidad de mecanografía realizado en cursomeca.com

Fecha de emisión: 29 de agosto del 2017

Número de certificado: C00225581
Código de validación: 995622

Para comprobar la autenticidad de este certificado introduzca el número de certificado y código de validación en www.cursomeca.com/certificados/

Cursomeca
www.cursomeca.com

CERTIFICADO

Adrián Ernesto Ruiz Figueroa ha alcanzado una velocidad de **261 PPM** (Pulsaciones Por Minuto) en nuestros tests y por el cual se extiende el presente certificado. Test de velocidad de mecanografía realizado en cursomeca.com

Fecha de emisión: 29 de agosto del 2017

Número de certificado: C00225582
Código de validación: 1062325

Para comprobar la autenticidad de este certificado introduzca el número de certificado y código de validación en www.cursomeca.com/certificados/

Cursomeca
www.cursomeca.com

CERTIFICADO

segundo manuel gonzales reluz ha alcanzado una velocidad de **223 PPM** (Pulsaciones Por Minuto) en nuestros tests y por el cual se extiende el presente certificado. Test de velocidad de mecanografía realizado en cursomeca.com

Fecha de emisión: 20 de julio del 2017

Número de certificado: C00221317
Código de validación: 500400

Para comprobar la autenticidad de este certificado introduzca el número de certificado y código de validación en www.cursomeca.com/certificados/

Cursomeca
www.cursomeca.com

CERTIFICADO

segundo manuel gonzales reluz ha alcanzado una velocidad de **229 PPM** (Pulsaciones Por Minuto) en nuestros tests y por el cual se extiende el presente certificado. Test de velocidad de mecanografía realizado en cursomeca.com

Fecha de emisión: 20 de julio del 2017

Número de certificado: C00221318
Código de validación: 542260

Para comprobar la autenticidad de este certificado introduzca el número de certificado y código de validación en www.cursomeca.com/certificados/

Cursomeca
www.cursomeca.com

CERTIFICADO

segundo manuel gonzales reluz ha alcanzado una velocidad de **208 PPM** (Pulsaciones Por Minuto) en nuestros tests y por el cual se extiende el presente certificado. Test de velocidad de mecanografía realizado en cursomeca.com

Fecha de emisión: 20 de julio del 2017

Número de certificado: C00221320
Código de validación: 280999

Para comprobar la autenticidad de este certificado introduzca el número de certificado y código de validación en www.cursomeca.com/certificados/

Cursomeca
www.cursomeca.com

CERTIFICADO

SIRLOPU CUMPA MIGUEL ANGEL ha alcanzado una velocidad de **278 PPM** (Pulsaciones Por Minuto) en nuestros tests y por el cual se extiende el presente certificado. Test de velocidad de mecanografía realizado en cursomeca.com

Fecha de emisión: 20 de julio del 2017

Número de certificado: C00221326
Código de validación: 505405

Para comprobar la autenticidad de este certificado introduzca el número de certificado y código de validación en www.cursomeca.com/certificados/

Cursomeca
www.cursomeca.com

CERTIFICADO

SIRLOPU CUMPA MIGUEL ANGEL ha alcanzado una velocidad de **255 PPM** (Pulsaciones Por Minuto) en nuestros tests y por el cual se extiende el presente certificado. Test de velocidad de mecanografía realizado en cursomeca.com

Fecha de emisión: 20 de julio del 2017

Número de certificado: C00221327
Código de validación: 547902

Para comprobar la autenticidad de este certificado introduzca el número de certificado y código de validación en www.cursomeca.com/certificados/

Cursomeca
www.cursomeca.com

CERTIFICADO

SIRLOPU CUMPA MIGUEL ANGEL ha alcanzado una velocidad de **333 PPM** (Pulsaciones Por Minuto) en nuestros tests y por el cual se extiende el presente certificado. Test de velocidad de mecanografía realizado en cursomeca.com

Fecha de emisión: 20 de julio del 2017

Número de certificado: C00221328
Código de validación: 591855

Para comprobar la autenticidad de este certificado introduzca el número de certificado y código de validación en www.cursomeca.com/certificados/

Cursomeca
www.cursomeca.com

CERTIFICADO

steyner ha alcanzado una velocidad de **218 PPM** (Pulsaciones Por Minuto) en nuestros tests y por el cual se extiende el presente certificado. Test de velocidad de mecanografía realizado en cursomeca.com

Fecha de emisión: 20 de julio del 2017

Número de certificado: C00221322
Código de validación: 349977

Para comprobar la autenticidad de este certificado introduzca el número de certificado y código de validación en www.cursomeca.com/certificados/

Cursomeca
www.cursomeca.com

CERTIFICADO

steyner ha alcanzado una velocidad de **203 PPM** (Pulsaciones Por Minuto) en nuestros tests y por el cual se extiende el presente certificado. Test de velocidad de mecanografía realizado en cursomeca.com

Fecha de emisión: 20 de julio del 2017

Número de certificado: C00221324
Código de validación: 424779

Para comprobar la autenticidad de este certificado introduzca el número de certificado y código de validación en www.cursomeca.com/certificados/

Cursomeca
www.cursomeca.com

CERTIFICADO

steyune ha alcanzado una velocidad de **173 PPM** (Pulsaciones Por Minuto) en nuestros tests y por el cual se extiende el presente certificado. Test de velocidad de mecanografía realizado en cursomeca.com

Fecha de emisión: 20 de julio del 2017

Número de certificado: C00221321
Código de validación: 314760

Para comprobar la autenticidad de este certificado introduzca el número de certificado y código de validación en www.cursomeca.com/certificados/

ANEXO 02: Código fuente de la herramienta TheCoder

Como parte del proyecto el investigador implementó la herramienta CASE TheCoder cuyo código fuente se entrega en el CD adjunto.