UNIVERSIDAD RICARDO PALMA

FACULTAD DE INGENIERÍA
PROGRAMA DE TITULACIÓN POR TESIS
ESCUELA PROFESIONAL DE INGENIERÍA INFORMÁTICA

FRAMEWORK PARA APLICACIONES CON BASE DE DATOS RELACIONAL ORIENTADO A DESARROLLADORES DE SOFTWARE

TESIS

PARA OPTAR EL TÍTULO PROFESIONAL DE
INGENIERO INFORMÁTICO

PRESENTADO POR

Bach. PALMA SERRANO, CARLOS WALTHER
Bach. VELASQUEZ ASTUCURI, CHRISTIAN EDUARDO

ASESORES: MG. ING. LINÁREZ COLOMA, HUMBERTO VICTOR

LIMA – PERÚ
2019

Dedicatoria

El presente documento está dedicado a mi familia, mis padres Walther y María, mi hermano Walther, su esposa Yoselin y la pequeña Sofía, y a mi enamorada Ana Karina, por todo su amor incondicional y por todo el apoyo que me han brindado día a día en todo este tiempo, brindándome las fuerzas necesarias para poder lograr este objetivo tan importante para mí.

Carlos Palma

A mi abuela materna, por estar conmigo, por enseñarme a crecer y a que si caigo debo levantarme, por apoyarme y guiarme, por ser las bases que me ayudaron a llegar hasta aquí.

El presente trabajo es dedicado también a mis padres, a mi esposa y a mis hijos quienes han sido parte fundamental para escribir este trabajo, ellos son quienes me dieron grandes enseñanzas y los principales protagonistas de este “sueño alcanzado”.

Christian Velasquez

Agradecimiento

A toda mi familia, por hacer de mí una persona correcta inculcándome sus valores para ser una persona de bien y no rendirme jamás ante las adversidades. A mis compañeros del entorno laboral, social, educativo, por esos ánimos que siempre me han brindado para no desistir y seguir siempre adelante. A la Universidad Ricardo Palma, por haberme brindado los conocimientos necesarios para llegar a cumplir la meta de ser un profesional.

Carlos Palma

A mis padres, quienes me apoyaron en todo lo indispensable, a mi esposa, que me dio apoyo moral y afectivo en todo momento y a mis hijos Elion y Kathleen, que me apoyaron con darme la motivación permanente de ser mejor persona y darles las mejores oportunidades.

A la Universidad Ricardo Palma, que, con sus docentes, lograron potenciar mis facultades y mis aficiones para lograr mi titulación en la carrera de Ingeniería informática.

Christian Velasquez

ÍNDICE GENERAL

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

El presente trabajo de investigación se motivó en el estudio de los procesos que realizan los desarrolladores de código de software cuando crean aplicaciones diversas, con acceso a base de datos relacionales. Durante el proceso de la codificación los desarrolladores se tomaban una gran parte del tiempo en tareas de acceso y alteración de los datos. Propuesta del modelo espiral (Boehm, 1988).

Debido a ello, se propone en este trabajo, enfocarse en el rubro de construcción de programas con uso de base de datos relacionales. Que en el año 2019 el uso de base de datos relacionales (SQL) es mayor al 50 por ciento, de todas las tecnologías que usan los desarrolladores de software (Stack Overflow, 2019; Jet Brains, 2019).

La propuesta por lo tanto es un framework, que automatiza la codificación de esas tareas repetitivas de acceso a datos, usando estándares internacionales de escritura de código; haciendo una mejora sustancial en los tiempos y la calidad de la construcción de aplicaciones de software con acceso a base de datos relacionales.

Logrando así, reducción de tiempos, aumento en la productividad de los desarrolladores, reducción de errores y la mejora de la calidad del código, debido a que se aplicó el uso de patrones de programación reconocidos internacionalmente por mejorar la calidad, el fácil entendimiento del código y la integridad de los datos.

Mejorando la satisfacción de los creadores de software, reduciendo los errores (bugs) por acceso, alteración e integridad de datos, así como también la satisfacción de los usuarios, que usan el producto final construido.

Palabras claves: Base de datos relacionales, SQL, Codificación, SQL Server, Oracle, MySql, desarrolladores de software, programadores.

ABSTRACT

The present research work was motivated in the study of the processes carried out by software code developers when creating diverse applications, with access to relational database. During coding process, developers took a large part of the time in tasks of accessing and altering the data. Proposal of the spiral model (Boehm, 1988).

Due to this, it is proposed in this work to focus on the construction of programs with the use of relational databases. That in 2019 the use of relational database (SQL) is greater than 50 percent, of all the technologies used by software developers (Stack Overflow, 2019; Jet Brains, 2019).

The proposal is therefore a framework, which automates the codification of those repetitive tasks of access to data using international code writing standards, making a substantial improvement in the times and the quality of the construction of software applications with access to relational database.

Achieving this, reducing time, increasing the productivity of developers, reducing errors and improving the quality of the code, because we apply the use of internationally recognized programming patterns to improve quality, easy understanding of the code and the integrity of the data.

Improving the satisfaction of the creators of software, reducing errors (bugs) by access, alteration and integrity of data, as well as the satisfaction of users, who use the final product built.

Keywords: Relational database, SQL, Coding, SQL Server, Oracle, MySql, software developers, programmers.

INTRODUCCIÓN

Desde que se tiene registros de las civilizaciones humanas, para todos los campos de estudio, siempre están en un proceso constante de automatización, lo cual se mueve conforme avanza la tecnología. Siendo el principal objetivo reducir el error humano, y mejorar los tiempos en la ejecución de tareas repetitivas, siendo estas tareas delegadas por ejemplo a animales, máquinas, vehículos, robots, programas de computadora u otros.

En el caso de la construcción de programas de software, en ese campo, la automatización que existe se inclina más a rutinas, subrutinas, intérpretes, librerías reutilizables y entornos de desarrollo; así pues, se puede automatizar tareas concretas, cada vez más amplias que abarcan mayores capacidades que hasta ahora se consideraban exclusivamente humanas, como identificar una voz o un rostro, etiquetar una fotografía, conducir un coche o predecir un comportamiento de acuerdo con unas reglas. Pero aún queda mucho camino para automatizar totalmente el trabajo de los desarrolladores.

En el capítulo uno, del presente trabajo de investigación, está motivado en el estudio de los procesos que realizan los desarrolladores de código de software cuando crean aplicaciones diversas con acceso a base de datos relacionales, especialmente en la etapa de codificación de la creación del software, en esta investigación pretendemos dar un avance más en este ámbito, buscando el objetivo de reducir los tiempos que emplean los desarrolladores en crear programas informáticos y promover el uso de estándares de escritura de código que mejoran la calidad de este.

En el capítulo dos, se detalla el marco de esta investigación, tomando en cuenta que los desarrolladores que usan base de datos relacionales para el año 2019 son más de la mitad del global de desarrolladores de código de software (Stack Overflow, 2019; Jet Brains, 2019); Para demostrar la eficacia de esta propuesta, se delimita este trabajo a un manejador de base de datos y un lenguaje de programación, no siendo una restricción, añadir posteriormente otros lenguajes y manejadores de base de datos para ampliar el universo objetivo.

En el capítulo tres y cuatro, se detalla la forma como se realiza técnicamente el proceso, en resumen, el framework emplea los patrones al proyecto del usuario, sin necesidad que el desarrollador tenga conocimientos de estos o como aplicarlos, de tal manera que el software escribe directamente código sobre su proyecto y su estructura, adicionalmente crea una aplicación de demostración, donde se muestra la manera más eficiente de acceder y alterar los datos.

Debido a ello, esta investigación concluye que, de aplicar la solución propuesta, se logra la reducción de tiempos, aumento de la productividad de los desarrolladores, reducción de errores y la mejora de la calidad del código, al plantear el uso de estándares de programación reconocidos internacionalmente por mejorar la calidad, el fácil entendimiento del código y la integridad de los datos.

Mejorando la satisfacción de los usuarios, que usan el producto final construido, consiguiendo un beneficio a los creadores de software, reduciendo los errores por acceso, alteración e integridad de datos.

CAPÍTULO I: VISIÓN DEL PROYECTO

1.1 Antecedentes del problema

1.1.1 Descripción del problema

Existen diversas metodologías de desarrollo de software, por mencionar algunas: cascada, espiral, iterativo. Habiendo en cada una de ellas un proceso que se encarga de la programación directa de la aplicación, en la tabla 1, mostramos las diversas metodologías y sus procesos.

Tabla 1: Metodologías de desarrollo de software

Metodología	**Procesos**
Cascada	Análisis, Diseño, Codificación, Pruebas, Operaciones (Royce, 1970)
Espiral	Determinar objetivos, Análisis de riesgos, Desarrollar y probar, Planificación (Boehm, 1988)
Iterativo	Concepción, Elaboración, Construcción, Transición (Larman & Basili, 2003)

Fuente: Elaboración propia

El presente trabajo se enfoca en el proceso de codificación en la metodología cascada, que es similar al proceso de Desarrollar en la metodología Espiral y construcción en la metodología Iterativa, en resumen: “Los programas para computadoras modernas consisten en secuencias de instrucciones que están codificadas como dígitos numéricos. Tal sistema de codificación se conoce como lenguaje de máquina.” (Brookshear, 2006, pág. 268)

En la creación de software, un porcentaje importante del tiempo de desarrollo de una aplicación, es empleado en la creación de los procesos de acceso a datos, y creación de la estructura del proyecto aplicando patrones de desarrollo.

Estos procesos que son repetitivos para los desarrolladores de código son escritos reiteradamente por cada acceso/alteración de cada entidad de la base de datos

relacional. Lo que conlleva a cometer errores menores, que consumen un tiempo sustancial.

En la creación de aplicaciones de software con acceso a base de datos relacionales existen estándares internacionales de desarrollo (patrones), con la finalidad de facilitar la lectura del código fuente, la búsqueda de errores, integridad de los datos y la aplicación de técnicas de seguridad de la información.

Se pueden englobar los patrones en una corta definición "Los patrones le ayudan a construir sobre la experiencia colectiva de ingenieros de software experimentados. Estos capturan la experiencia existente y que ha demostrado ser exitosa en el desarrollo de software, y ayudan a promover las buenas prácticas de diseño." (Buschmann, 1996, p. 1)

Ciertamente no todos los desarrolladores aplican los patrones, o no los aplican correctamente, produciendo errores en el desarrollo y en el producto final.

Si bien existen librerías que aplican un mapeo objeto-relacional, (O.R.M) y "Por herramienta ORM, marco de persistencia, mapeador relacional de objetos o mapeador OR nos referiremos a una herramienta o componente que permite mapear objetos a bases de datos relacionales. Esta herramienta generalmente agrega un nivel especial a la aplicación y generalmente realiza un comportamiento CRUD." (Orság, 2006, p. 17)

El aplicarlos plantea algunos inconvenientes, por mencionarlos:

1. Necesitan una curva de aprendizaje regular, debido a que algunos emplean una forma especial de escribir el código y una estructura determinada.
2. Es necesario referenciar una librería externa, con lo que implica dejar abierta una brecha de seguridad.
3. No se puede aplicar estándares de programación de código establecidas dentro de su propia organización.
4. Presentan complejidad al trabajar con datos poco comunes, y variables de salida y/o formuladas.

5. Un cambio en el nombre de un campo físico de las tablas de la base de datos implica volver a codificar varias líneas de código en el programa.
6. No abarcan consultas a entidades con llamada por llaves foráneas o primarias múltiples.
7. Manejo de múltiples operaciones en una sola transacción.

En la tabla 2, se describe una muestra de sistemas ORM, y los inconvenientes que plantea a desarrolladores su implementación.

Tabla 2: Sistemas ORM populares y sus desventajas

Sistema ORM	**Inconvenientes**
Microsoft Entity Framework 6.2	Punto 2: Automáticamente se añade al proyecto librerías Punto 5: Cambiar o añadir un atributo a una tabla provoca editar manualmente las clases o rehacer. Punto 6: Las consultas a entidades referenciadas son pesadas y complejas de hacer.
Hibernate 5.4	Punto 1: Es necesario un tiempo considerable de aprendizaje para manejar bien este ORM Punto 2: Se añade librerías Hibernate al proyecto Punto 5: Cambiar o añadir un atributo a una tabla, provoca editar manualmente las clases.
Dapper ORM	Punto 2: Se añade varias librerías al proyecto que tendrán acceso a la información de la base de datos. Punto 4: No trabajan con datos como imágenes, atributos calculadas o de salida. Punto 5, 6 y 7: No manejan llaves foráneas, ni transacciones en múltiples operaciones.

Fuente: Elaboración Propia

Así pues, también existen los llamados lenguajes de muy alto nivel o lenguajes declarativos, que crean los programas a partir de una representación del conocimiento, que pueden ser diagramas, bloques u otros. Reduciendo la mayor parte de la codificación del desarrollador, pero este también tiene otros inconvenientes por mencionarlos:

1. Curva de aprendizaje pronunciada, estos lenguajes usan su propio entorno de desarrollo IDE, y su propio lenguaje de codificar, por lo que requieren tiempo adicional para su estudio.
2. Están limitados a los patrones y estándares aplicados por la herramienta.
3. La seguridad del código puede ser una vulnerabilidad, debido a que están limitados a las creadas por el mismo lenguaje y en muchos casos el código debe ser almacenados en una nube administrada por estas empresas.
4. Los cambios para aplicar nuevas tecnologías, o parches de seguridad, están limitados a las actualizaciones del entorno del lenguaje y su núcleo.
5. Están limitados solo a ciertas operaciones comunes de alteración de datos simples, más no maneja operaciones más complejas como manejo de tablas cruzadas muchos a muchos

En la tabla 3, se muestra algunos de los sistemas que entran al grupo de lenguajes declarativos, en relación con los inconvenientes que presentan a los desarrolladores el implementarlos.

Tabla 3: Lenguajes declarativos y sus desventajas

Lenguajes Declarativos	**Inconvenientes**
Genexus	Punto 1, Es necesario dedicar varias horas en comprender este software. Punto 2,3 y 4. Se tiene que tener métodos externos de seguridad.
App Inventor	Punto 2, 3 y 5. Están limitados a los patrones y estándares aplicados por la herramienta.
OpenXava	Punto 1, 2 y 4. Limitados a la forma de trabajo propuesta.
Scratch	Punto 2, 3 y 5 Muy limitado para proyectos empresariales.

Fuente: Elaboración propia

Si por el contrario no se emplea ninguna de las herramientas anteriormente mencionadas en la tabla 2 y la tabla 3, los desarrolladores se embarcan en una extensa tarea de escribir código de acceso y alteración de datos.

En la siguiente comparativa (Tabla 4) se muestra la cantidad de líneas de código que escribe un programador sin usar ningún ORM ó lenguaje declarativo, al trabajar con una base de datos de tan solo 7 tablas.

Tabla 4: Líneas de código necesarias para crear los accesos a base de datos desde un programa en .Net Framework 4.6.2

Lenguaje	**Líneas de código Aproximadamente**	**Procedimientos almacenados / Clases**
Scripts SQL Server	1741	64
Código en lenguaje C#	4585	21

Fuente: Base de datos AdventureWorks usando el esquema Purchasing.

Por lo tanto, de hacer esta tarea manualmente se concluye que el trabajo es altamente deficiente.

1.2 Identificación del problema

1.2.1 Problema Principal

Baja eficiencia en el desarrollo de aplicaciones de software con acceso de base de datos relacional.

1.2.2 Problemas Específicos

a) Deficiente empleo de patrones de desarrollo de código o no los emplean adecuadamente, siendo el tiempo de implementación un valor que se debe cuantificar en el cronograma de construcción, asignando un costo por hora.

b) Deficiente creación de procedimientos almacenados para el acceso y alteración de datos. Codificar procedimientos almacenados para acceder a tablas de la base de datos o alterarlas, toma un tiempo considerable el cual solo debe limitarse a operaciones especialmente complejas de acuerdo con el negocio, o para los reportes.

c) Deficiente codificación de clases y métodos dentro de la aplicación de software, empleados para acceder y alterar información en la base de datos relacional desde el programa fuente.
d) Deficiente aplicación de cambios en el programa fuente, al realizar un cambio en las tablas de la base de datos. Cuando un software se encuentra en fase intermedia o avanzada de la construcción, el alterar un campo en una tabla de base de datos, debe de propagarse a clases, métodos y procedimientos almacenados, y esto puede tomar un tiempo considerable

1.3 Objetivos

1.3.1 Objetivos General

Mejorar la eficiencia en el desarrollo de aplicaciones de software con acceso de bases de datos relacional, automatizando la programación de los procesos CRUD (Create, Read, Update y Delete) y empleando patrones de desarrollo con el uso de una aplicación de escritorio dirigida a desarrolladores de software que crean aplicaciones conectadas a base de datos relacional.

1.3.2 Objetivos Específicos

a) Mejorar la eficiencia en el desarrollo de aplicaciones de software, aplicando patrones MVC y Unit of Works al crear la estructura de programación nueva o modificar un proyecto solución existente.
b) Mejorar la eficiencia en el desarrollo de aplicaciones de software, automatizando la codificación de procedimientos almacenados de acceso y alteración de tablas de la base de datos.
c) Mejorar la eficiencia en el desarrollo de aplicaciones de software, automatizando la codificación de clases y métodos empleados para acceder y alterar información en la base de datos relacional desde el programa fuente.
d) Mejorar la eficiencia en el desarrollo de aplicaciones de software, automatizando la propagación de cambios en tablas de base de datos, clases, métodos y procedimientos almacenados.

1.4 Descripción y sustentación de la solución

1.4.1 Descripción de la solución

1. Elaboración del asistente de configuración (Wizard), un módulo que guiará paso a paso al desarrollador a configurar su proyecto de software; con esto se cubre el objetivo a.
2. Elaboración del módulo de configuración de la conexión a la base de datos, donde se detalla servidor, conexión, base de datos, tablas, y otros relacionados con el manejador de base de datos; con esto se cubre el objetivo b.
3. Elaboración del módulo de configuración del proyecto donde se detalla el nombre del proyecto, el nombre de las capas de modelo, vista, controladoras y entidades, Así como también el interfaz de usuario, si es una aplicación web, de escritorio o ambos; con esto se cubre el objetivo c.
4. Elaboración de la interfaz de configuración de la estructura, permitiendo usar estándares personalizados para la escritura de los procedimientos almacenados, y pudiendo usar distintos por cada proyecto; con esto se cubre el objetivo a y objetivo d.
5. Elaboración de los procesos de creación de la estructura de la solución, las clases, métodos, procedimientos almacenados y la interfaz de usuario de demostración en base a los parámetros especificados; con esto se cubre el objetivo a y objetivo d.

1.4.2 Justificación de la realización del proyecto

Se procede a desarrollar un framework que permita automatizar la codificación de tareas repetitivas de acceso y alteración de datos dentro de una base de datos relacional, aplicando estándares internacionales de escritura de código.

Para justificar académicamente se resalta la importancia de reducir los tiempos en la programación de procesos CRUD que consta de una serie de procedimientos almacenados y código fuente que lee y altera las entidades de la base de datos, aplicando patrones MVC para la estructura, y *"Unit of Work"* para mantener la integridad de los datos, al usar transacciones cuando se altera múltiples entidades en una sola operación.

Beneficios tangibles:

- Aumentar la productividad de los desarrolladores de código en un 15% comparado en no usar el framework propuesto.
- Reducir los tiempos de desarrollo de aplicaciones con acceso a base de datos relacionales 15%
- Reducir los tiempos de escribir scripts SQL de acceso y alteración de los datos en un 75%, siendo solo necesario para operaciones específicas del negocio, que por su naturaleza no está dentro del alcance.
- Se reduce errores en los procesos CRUD en un 10%.

Beneficios intangibles:

- Mantiene la transparencia del código, evitando el uso de librerías cerradas añadidas al proyecto.
- Se mantiene la integridad de los datos al facilitar el uso de transacciones en múltiples operaciones conjuntas.
- Se estandariza la escritura del código.
- Promueve la entrega de un producto final de calidad, por aplicar estándares internacionales de codificación.

CAPÍTULO II: MARCO CONCEPTUAL

2.1 Marco conceptual

2.1.1 Base de datos relacional

Se denomina base de datos relacional al conjunto de tablas correctamente identificadas en las que se puede acceder a la información de las mismas. Mediante un lenguaje de consultas estructuradas (Structure Query Language), se realiza la interacción entre un software y una base de datos relacional. El concepto básico de una base de datos relacional es el de "Relación".

> Cuando se considera una relación como una tabla de valores, cada fila de la tabla representa una recopilación de valores de datos relacionados. Una fila representa un hecho que generalmente corresponde a una entidad o relación del mundo real. El nombre de la tabla y los nombres de columna se utilizan para ayuda a interpretar el significado de los valores en cada fila.
> (Elmasri & Navathe, 2010, pág. 60)

Una tabla viene a ser la representación gráfica de una entidad, el cual nos muestra los valores de una tupla de forma ordenada. En la figura N° 1, se muestra la representación gráfica de una Tabla de una base de datos relacional.

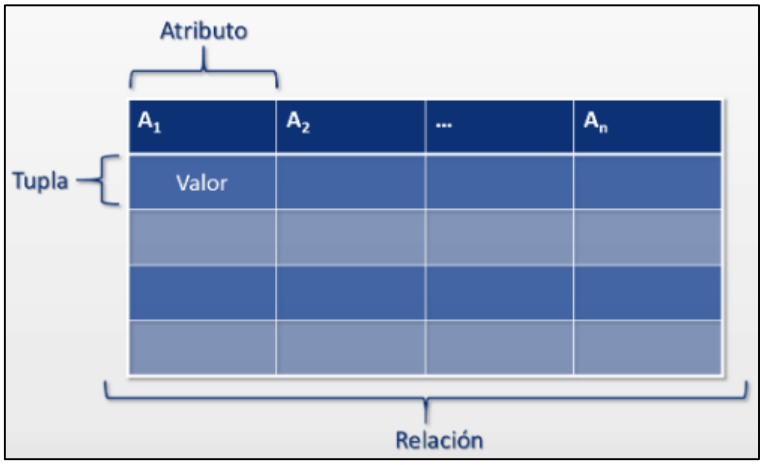


Figura 1: Representación gráfica de una tabla de un BD relacional
Fuente: https://www.ionos.es/digitalguide/hosting/cuestiones-tecnicas/bases-de-datos-relacionales/

Claves

Cada tupla de una tabla se identifica únicamente por una clave principal, el cual es representada el mayor de los casos por un número autoincrementado. “La restricción de integridad de la entidad establece que ningún valor de clave principal puede ser NULL. Esta se debe a que el valor de la clave primaria se usa para identificar tuplas individuales en una relación.” (Elmasri & Navathe, 2010, pág. 73).

Adicionalmente cada tupla, podría contener una o más claves foráneas, donde cada clave foránea es la representación de una clave principal de una tupla de otra tabla. En la figura N° 2, se muestra la representación gráfica de una llave primaria y foránea.

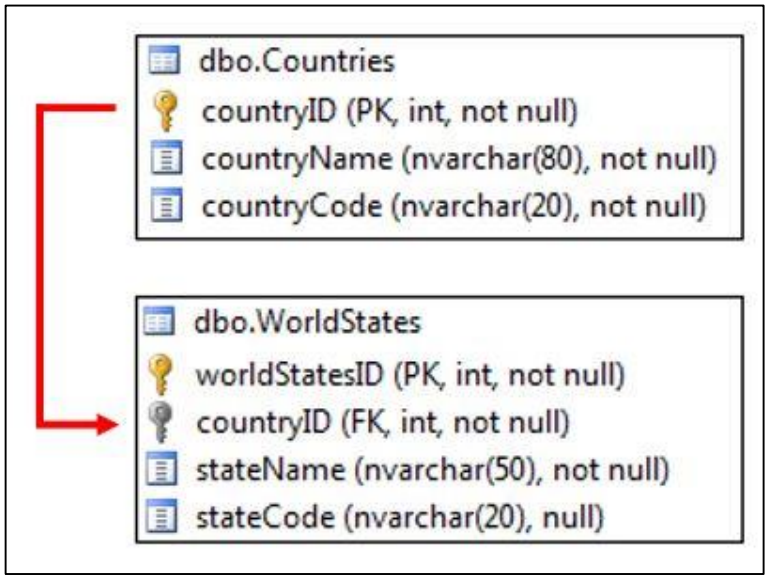


Figura 2: Representación gráfica de clave primaria y foránea
Fuente: Elaboración propia

Normalización

Al trabajar con base de datos relacionales, se utiliza la normalización, el cual consiste en la aplicación de reglas definidas a las tablas de la base de datos relacional, con la finalidad de clasificarlas en función de su significado, evitando así la duplicidad de información y asegurando la integridad de los datos.

Las reglas de normalización están diseñadas para evitar anomalías de actualización e inconsistencias de datos. Con respecto a las compensaciones de rendimiento, estas pautas están sesgadas hacia el supuesto de que todos los

> campos no clave se actualizarán con frecuencia. Tienden a penalizar la recuperación, ya que los datos que pueden haberse recuperado de un registro en un diseño no normalizado pueden tener que recuperarse de varios registros en la forma normalizada.
> (Kent, 1983, pág. 120)

En la tabla 5, se muestra las principales formas normales con sus respectivos objetivos a conseguir.

Tabla 5: Formas de Normalización

Forma Normal	Nomenclatura	Objetivo
Primera	1FN	Eliminar grupos repetitivos
Segunda	2FN	Eliminar datos redundantes
Tercera	3FN	Eliminar columnas que no dependen de la clave

Fuente: Elaboración propia

2.1.2 Desarrollo de Software

El desarrollo de software es la ingeniería que se encarga de la codificación de instrucciones, a fin de ser ejecutadas en un ordenador, para la creación de una solución tecnológica de un problema específico. A diferencia del hardware, el software viene a ser la parte intangible de un ordenador.

> El desarrollo de software es un proceso de aprendizaje social. El proceso es un diálogo en el que el conocimiento que debe convertirse en software se reúne e incorpora en éste. El proceso genera interacción entre usuarios y diseñadores, entre usuarios y herramientas cambiantes, y entre diseñadores y herramientas en evolución [tecnología].
> (Baetjer, 1998, pág. 85)

Ciclo de vida (RUP – Rational Unified Process)

El ciclo de vida de un desarrollo de software consiste en un modelo a seguir en el cual se definen diferentes enfoques con sus respectivas actividades a lo largo de todo el proceso de creación del software.

La metodología más utilizada para la realización del análisis, el diseño, la implementación y la documentación de un software es la del Proceso Unificado de Rational o RUP (Rational Unified Process). El RUP es conjunto de metodologías desarrollada por la empresa Rational Sofware, teniendo como propietario a la empresa IBM, el cual nos ofrece un conjunto de tareas que se adaptan fácilmente al contexto de cada organización.

El ciclo de vida del RUP se divide en 4 fases: Iniciación, Elaboración, Construcción y Transición. "Cada fase contiene una o más iteraciones, que se centran en producir los productos técnicos necesarios para lograr los objetivos de la organización de esa fase" (Kroll & Kruchten, 2003, pág. 39)

- **Fase de Iniciación**

  En esta fase se enfoca en la comprensión el problema, el alcance y delimitación del proyecto, administración y mitigación de riesgos.

- **Fase de Elaboración**

  En esta fase se enfoca en el modelado del negocio, modelado del sistema, análisis, diseño y la arquitectura del software a desarrollar.

- **Fase de Construcción**

  En esta fase se enfoca en el desarrollo del producto (software), enfocándose en aquellos casos de uso que se han obtenido en la fase de elaboración, para posteriormente realizar su implementación y sus pruebas finales.

- **Fase de Transición**

  En esta fase se enfoca en la preparación del producto (software) para su entrega a los usuarios finales.

En la figura N° 3, se muestra las actividades que se realizan en cada fase del ciclo de vida de RUP.

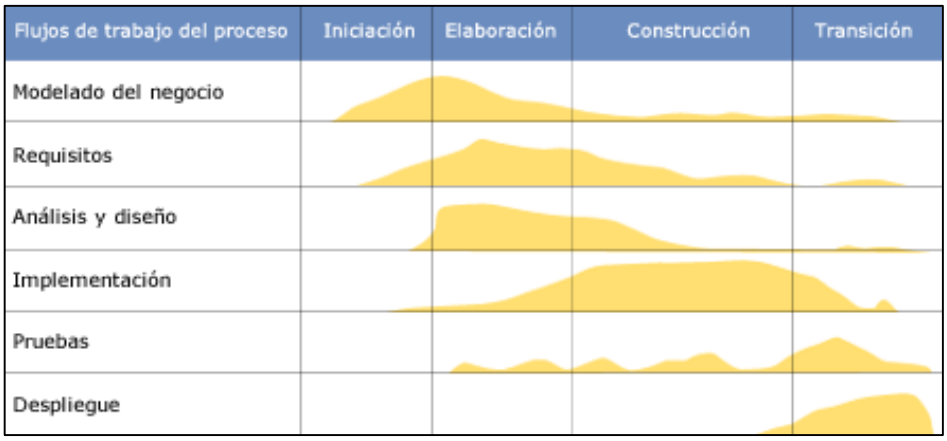


Figura 3: Actividades realizadas por fase de ciclo de vida RUP
Fuente: https://es.wikipedia.org/wiki/Proceso_Unificado_de_Rational

2.1.3 Metodologías de Desarrollo

Las metodologías de desarrollo es un marco de trabajo en el cual se utilizan técnicas y herramientas para hacer posible el desarrollo de un producto software. Estas técnicas y herramientas generalmente son documentadas ya que de esa manera facilitan al programador la visualización del ciclo de vida de un proyecto que se va a utilizar. Existen diferentes metodologías de desarrollo donde, cada una de ellas, tienen su propio enfoque en cuanto al desarrollo de software.

Desarrollo en Cascada

La metodología del desarrollo en cascada clasifica ordenadamente el proceso para el desarrollo de software en etapas, de tal manera que, para comenzar una nueva etapa, debe esperar la finalización de una etapa anterior.

Sus ventajas principales son su fácil entendimiento e implementación, está documentado al 100% y promueve al programador la realización de definir los diseños antes de realizar la codificación. Su principal desventaja es que, si existe un error en la etapa de prueba, necesariamente se tiene que rediseñar y volver a hacer la codificación de manera correcta, lo cual conlleva a un incremento de costos del desarrollo.

En la figura N° 4, se muestra los procedimientos realizados en la metodología de cascada.

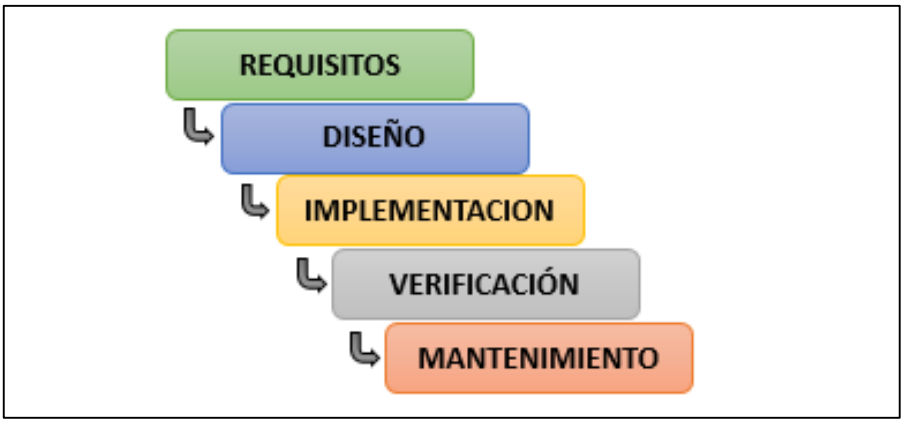


Figura 4: Ejemplo de metodología de cascada
Fuente: Elaboración propia

Desarrollo en Espiral

La metodología de desarrollo en espiral consiste en la existencia de tareas en cada bucle, las cuales están definidas en función al análisis de riesgo iniciando desde el bucle anterior. Esta metodología toma en cuenta fundamentalmente los riesgos que se pueden presentar en cada etapa del ciclo de vida del desarrollo de software.

> El modelo espiral es un enfoque realista para el desarrollo de sistemas y de software a gran escala. Como el software evoluciona a medida que el proceso avanza, el desarrollador y cliente comprenden y reaccionan mejor ante los riesgos en cada nivel de evolución. El modelo espiral usa los prototipos como mecanismo de reducción de riesgos, pero, más importante, permite aplicar el enfoque de hacer prototipos en cualquier etapa de la evolución del producto. (S. Pressman, 2010, pág. 40)

En cada espiral se tienen en cuenta los siguientes puntos:

- **Objetivos**

  Identificación de funcionalidades que tendrá el software.

- **Análisis de Riesgo**

  Identificación de riesgos del proyecto y alternativas para evitarlos.

- **Desarrollo y Pruebas**

  Codificación de las funcionalidades del software y la realización.

- **Planificación**

  Se prepara la planificación del siguiente bucle.

Su ventaja es que se tendrá un software de calidad debido a que se reduce al máximo los riesgos del proyecto, con la posibilidad de añadir mejoras y nuevos requerimientos sin la necesidad de rehacer todo el proyecto nuevamente. Su principal desventaja es que esta metodología es muy costosa y se requiere mucha experiencia en la gestión de riesgos. En la figura N° 5, se muestra la representación de la metodología descrita.

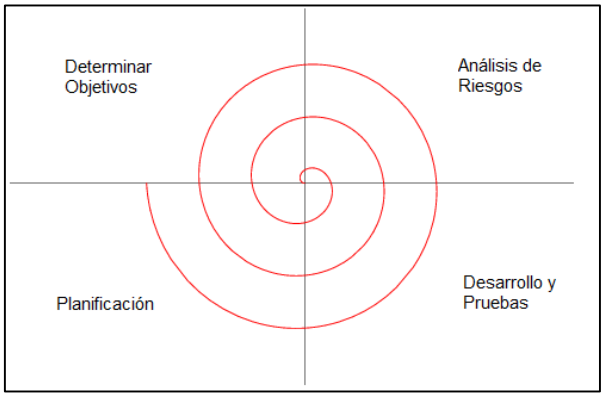


Figura 5: Representación de la metodología en espiral
Fuente: Elaboración propia

2.1.4 Mapeo Objeto – Relacional (ORM – Object Relational Mapping)

El mapeo objeto-relacional es una técnica de programación para convertir datos del sistema de tipos utilizado en un lenguaje de programación orientado a objetos al utilizado en una base de datos relacional. En la práctica esto crea una base de datos virtual orientada a objetos sobre la base de datos relacional. Esto posibilita el uso de las características propias de la orientación a objetos (esencialmente la herencia y el polimorfismo).

(Yanes & Gracia del Busto, 2011, págs. 2-3)

Las herramientas que se utilizan en los distintos lenguajes de programación utilizan la técnica de mapeo para que el programador se enfoque completamente en la realización del software separándolo por completo de la base de datos relacional que se está utilizando.

Las principales ventajas que se tiene son las siguientes:

- Se reduce el tiempo de generación del código orientado a objetos
- Se presenta una mayor seguridad en la relación de los objetos y la base de datos relacional
- Facilidad para el constante mantenimiento del código orientado a objetos
- El código orientado a objeto será independiente de la base de datos que se esté utilizando

Las principales desventajas que se tiene son las siguientes:

- Se destacan para aquellas aplicaciones que se desea realizar las funciones básicas en una base de datos (CRUD: Create, Read, Update, Delete)
- Complejidad del aprendizaje del código ORM
- Generación de librerías adicionales

Hibernate (Java)

Una de las herramientas de ORM más conocidas actualmente en el mercado es Hibernate. Esta herramienta ORM está orientado para el lenguaje de programación Java, el cual permite el mapeo de atributos de una base de datos relacional mediante archivos en formato XML.

Mediante un lenguaje de consultas denominada HQL (Hibernate Query Language), se puede realizar el almacenamiento, actualización y recuperación de la información en una base de datos relacional. En la figura N° 6 se muestra la arquitectura de la herramienta ORM Hibernate.

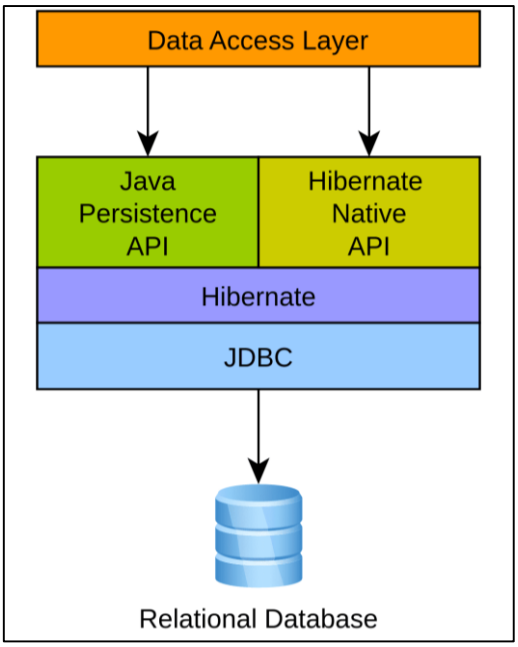


Figura 6: Arquitectura herramienta ORM Hibernate
Fuente: https://hibernate-orm.en.softonic.com/

Otras herramientas ORM conocidas en el mercado son:

- **Java**: Hibernate, iBatis, Ebean
- **.Net**: nHibernate, Entity Framework
- **Python**: SQLObject, Django

2.1.5 MVC (Model View Controller)

MVC es un patrón de arquitectura el cual separa ordenadamente los datos, la lógica del negocio y las interfaces del usuario. Este patrón consta de 3 componentes distintos los cuales hacen posible la funcionalidad de un software: el modelo, la vista y el controlador. En la figura N° 7 se muestra la arquitectura del patrón MVC.

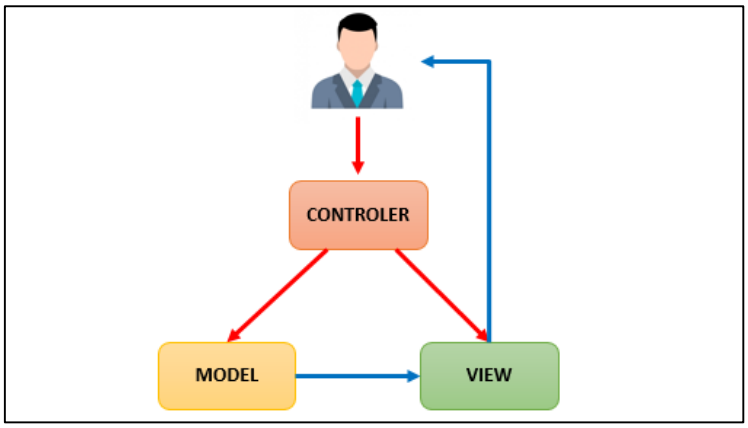


Figura 7: Comunicación MVC
Fuente: Elaboración propia

- **Vista**

  Este componente viene a ser cada elemento de la interfaz gráfica que interactúa con el usuario. Su función principal es la de obtener las peticiones, también llamados eventos, realizadas por parte del usuario. La vista muestra siempre la información del "Modelo".

- **Modelo**

  Este componente tiene la responsabilidad de manipular los datos y relacionarlos con el sistema de gestor de base de datos según la petición realizada por el usuario (consultas, actualizaciones, etc). Este componente contempla la lógica del negocio.

- **Controlador**

  Este componente es el encargado de recepcionar la petición de los usuarios y enviarlos al modelo. Es responsable de elegir que vista es la que tiene mostrar al usuario de acuerdo a su petición. El Controlador es el intermediario entre el "Modelo" y la "Vista".

Flujo de información del modelo MVC

La interacción del usuario con una aplicación MVC sigue un ciclo natural: el usuario toma una acción y en respuesta, la aplicación cambia su modelo de datos y ofrece una vista actualizada al usuario. Y luego el ciclo se repite. Este es un ajuste conveniente para aplicaciones web entregadas como una serie de Solicitudes y respuestas HTTP.
(Freeman, 2013, pág. 4)

En la figura N° 8 se muestra el flujo realizado por el modelo MVC.

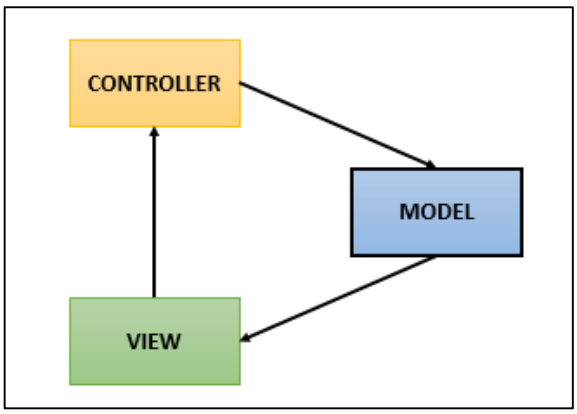


Figura 8: Arquitectura modelo MVC
Fuente: Elaboración propia

1. El usuario realiza la petición mediante la Vista (interfaz gráfica).
2. La vista notifica al Controlador la petición realizada por el usuario.
3. El Controlador recibe la petición y actualiza el modelo según la petición del usuario (insertar, actualizar, eliminar).
4. El modelo es actualizado y notifica los cambios a la vista.
5. La vista muestra los resultados al usuario.

2.2 Estado del Arte

2.2.1 Trabajos realizados (Investigación y Software)

2.2.1.1 Herramienta ORM para mejorar la productividad de la empresa Interfaces Software Group

Chingo (2016) describe el desarrollo de una herramienta ORM con el objetivo de mejorar la productividad de aplicaciones de la empresa Interfaces Software Group y así alcanzar una mayor calidad en sus productos (softwares). La propuesta es la implementación de una herramienta ORM que permita la conexión a SQL Server, MySql y PostgreSQL para posteriormente proceder a la creación de procedimientos almacenados en base a las tablas existentes de una base de datos seleccionada.

En la figura N° 9 se muestra la interfaz de acceso a la herramienta ORM en el cual se puede seleccionar una base de datos e ingresar las credenciales para su acceso.

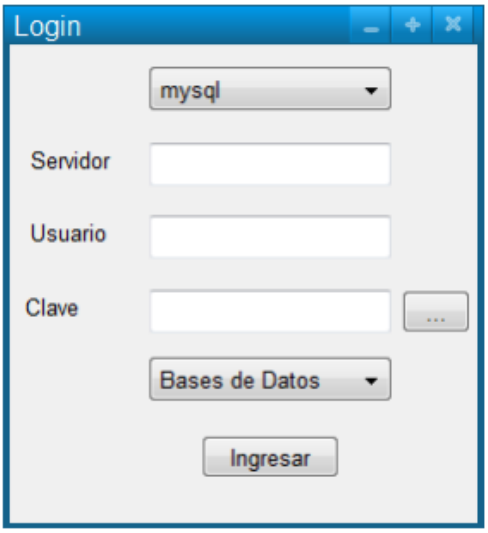


Figura 9: Formulario Login
Fuente: Tomado de Herramienta ORM para mejorar la productividad de la empresa Interfaces Software Group

En la figura N° 10 se muestra un formulario básico realizado por el autor, en el cual se va a mostrar las opciones para que los procedimientos almacenados se ejecuten en la base de datos seleccionada o de manera interna en un directorio del software. De igual forma, se muestran las opciones para la generación de las Clases (objetos) y la generación de la capa de datos.

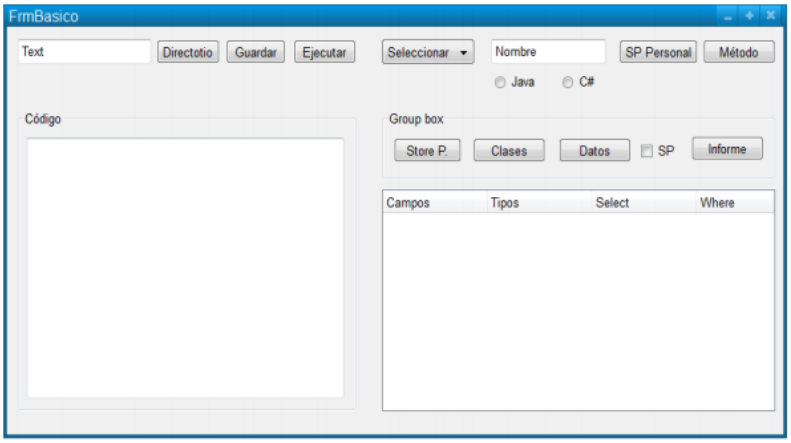


Figura 10: Formulario Básico
Fuente: Tomado de Herramienta ORM para mejorar la productividad de la empresa Interfaces Software Group

En la figura N° 11 se muestra el formulario Panel de Opciones en el cual el desarrollador muestra las tablas con sus respectivos campos y podrá generar un store procedure seleccionando las columnas respectivas en los campos “Select” y “Where”.

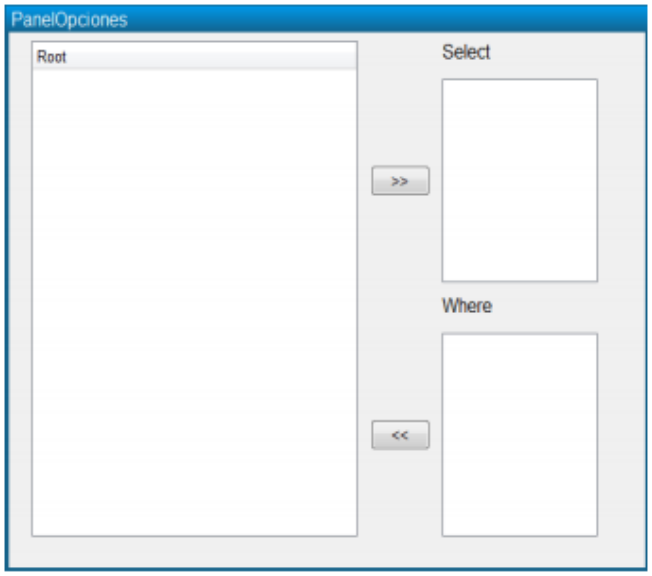


Figura 11: Formulario Panel Básico
Fuente: Tomado de Herramienta ORM para mejorar la productividad de la empresa Interfaces Software Group

2.2.1.2 Sistema de Gestión de Información Odontológica utilizando ORM para el Departamento de Bienestar Universitario de la UTN

El tesista describe el desarrollo de una herramienta ORM con el objetivo de agilizar el proceso de atención odontológica y la generación de reportes estadísticos del Departamento de Bienestar Universitario.

"Se considera ORM a cualquier capa de persistencia que proporcione autogeneración de SQL a través de metadatos (generalmente XML). No se considera ORM a una capa de persistencia creada por el propio desarrollador" (Freire, 2008, pág. 14)

Para la tesis descrita, el autor ha utilizado el patrón Facade, el cual es utilizado para minimizar la complejidad de un sistema mediante una interfaz fácil de usar. Las características principales de este patrón son las siguientes:

- El cliente se comunica con el sistema realizando peticiones al Facade.
- Facade engloba aquellas clases para responder a una determinada petición.
- Las clases que se engloban en el Facade realizan el trabajo solicitado por el cliente. Las clases no tienen conocimiento de la existencia del Facade.

En la figura N° 12 se muestra un ejemplo de una petición realizada por un cliente utilizando el patrón Facade.

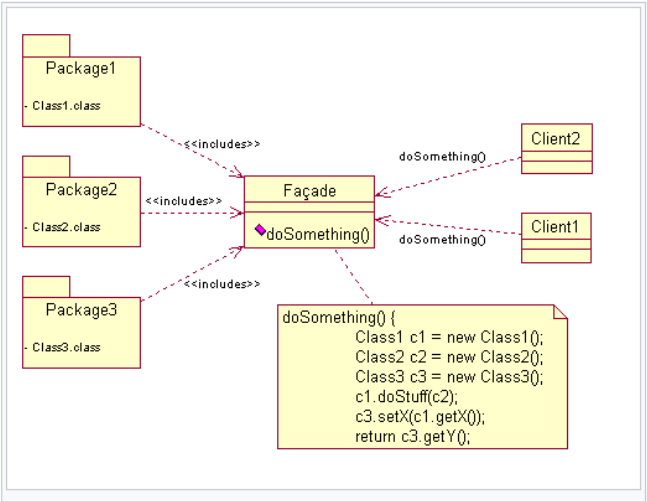


Figura 12: Ejemplo de patrón Facade
Fuente: https://es.wikipedia.org/wiki/Facade

En la figura N° 13 se detalla y se muestra el diagrama de componentes el cual fue elaborado por el mismo autor de la tesis descrita.

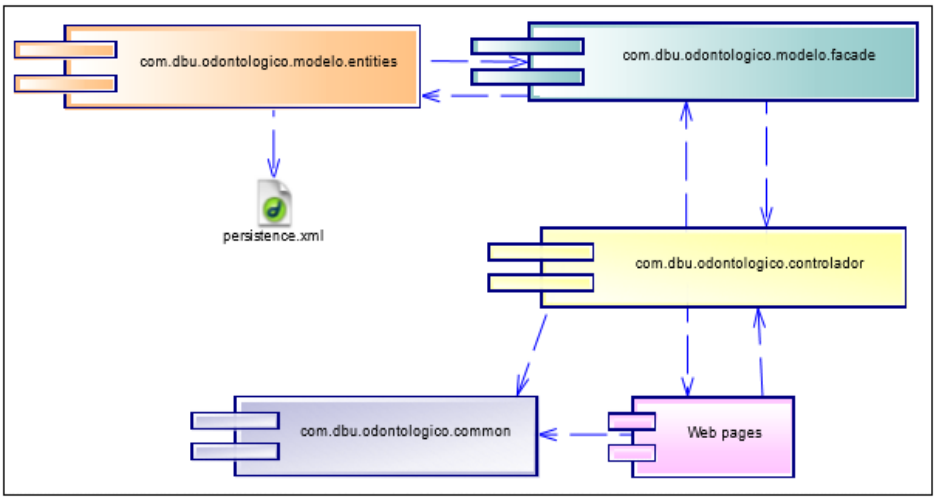


Figura 13: Diagrama de Componentes
Fuente: Tomado de Sistema de Gestión de Información Odontológica utilizando ORM para el Departamento de Bienestar Universitario de la UTN

- El componente entities contiene aquellas clases mapeadas con la base de datos.
- El componente Facade contiene aquellas clases que permiten la conexión entre el modelo y el controlador.
- El componente controlador contiene aquellas clases que implementan los procesos para gestionar la información.
- El componente common contiene aquellas clases de funciones comunes para su aplicación: getFecha.cs, getTemplate.cs, etc.
- El componente web pages contiene las interfaces html con los que el usuario va a interactuar.

2.2.1.3 Análisis de la tecnología ORM-Hibernate con respecto a la productividad, aplicado al sistema de gestión de documentación del departamento de procuraduría de la UNACH.

El tesista describe el desarrollo de una herramienta ORM con el objetivo de que el Departamento de procuraduría general pueda contar con una aplicación web para gestionar sus procesos internos, ya que actualmente realizan la gestión de su información de forma manual.

Para la creación de dicha aplicación web, el autor de la tesis analiza y realiza una comparación entre los resultados logrados utilizando la herramienta ORM-Hibernate y la arquitectura de N-Capas, basándose únicamente en el proceso de negocio de gestión de acceso de usuarios.

En la figura N° 14 se puede visualizar la comparación obtenida entre la herramienta ORM-Hibernate y la arquitectura N-Capas.

	N-Capas	ORM-Hibernate	Diferencia
Número de líneas de código	76	26	50
Número de horas empleadas	12	5	7
Porcentaje de líneas de código	100%	34%	66%
Porcentaje de horas empleadas	100%	42%	58%
		Promedio	62%

Figura 14: Datos comparativos entre la herramienta ORM-Hibernate y la arquitectura N-Capas
Fuente: Tomado de Análisis de la tecnología ORM-Hibernate con respecto a la productividad, aplicado al sistema de gestión de documentación del departamento de procuraduría de la UNACH

Después de realizar las comparaciones, se llegó a la conclusión que se obtiene una mayor productividad utilizando la herramienta ORM-Hibernate, dejando atrás la opción de la arquitectura N-Capas.

> Para el desarrollo de las aplicaciones web de forma eficiente se recomienda utilizar la tecnología ORM-Hibernate porque permite una mayor productividad enfocada al acceso de datos, gracias a la tecnología mapeo objeto/relacional este permite la creación de las clases de las tablas de la base de datos en segundos y el tratamiento de los objetos con los métodos save, update, delete y el lenguaje de consulta HQL.
> (Tixi, 2016, pág. 110)

2.2.1.4 GreenDAO

GreenDAO es una herramienta ORM de código abierto enfocado para la base de datos SQLite de Android. Nos permite la creación de acceso a datos de objetos mediante una interfaz que se encuentre entre la aplicación y la base de datos.

En la figura N° 15 se muestra la comparación entre las herramientas ORM para Android: GreenDAO y ORMLite; teniendo la primera opción como la herramienta más utilizada por los desarrolladores de aplicaciones móviles.

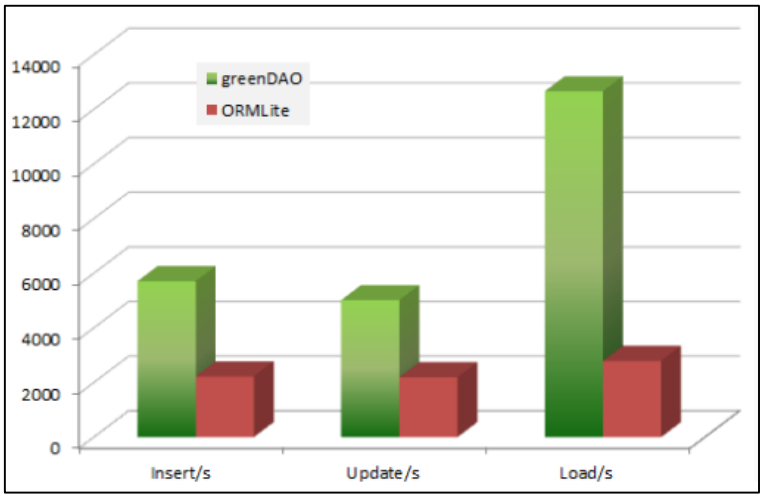


Figura 15: Comparativa de rendimiento entre greenDAO y ORMLite
Fuente: https://academiaandroid.com/introduccion-a-orm-y-greendao/

En la figura N° 16, se muestra la arquitectura de la herramienta greenDAO.

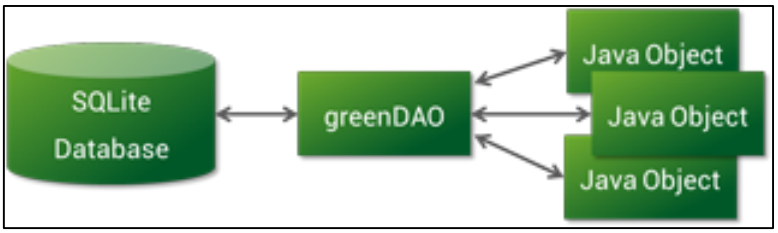


Figura 16: Arquitectura greenDAO
Fuente: https://academiaandroid.com/introduccion-a-orm-y-greendao/

Las principales ventajas del GreenDAO son:

- Brinda un alto performance
- Uso mínimo del consumo de recursos del dispositivo
- Es una herramienta de código abierto
- Permite la generación de objetos y acceso a datos.

2.2.2 Herramientas para la implementación

Para el presente desarrollo de tesis se está considerando las siguientes herramientas:

- Visual Studio .Net 2017 (Lenguaje C#)
- SQL Server 2018

2.2.3 Definición de términos

- **Automatización de datos**: Proceso por el cual se realiza la recolección para poder resguardarlos en una base de datos.

- **Base de datos**: Es el conjunto de datos almacenados los cuales representan a una determinada entidad.

- **CRUD**: (Create, Read, Update, Delete) Funciones básicas que realiza un software en relación con una base de datos.

- **Entidad**: Representación de un objeto identificado por sus atributos.

- **MVC**: Modelo Vista-Controlados. Arquitectura de software que, para un software, divide los datos, la lógica y la interfaz de usuario.

- **ORM**: Object-Relational Mapping. Técnica de programación que convierte tipo de datos de lenguaje orientado a objetos en tipo de datos de una base de datos relacional.

- **Software**: Conjunto de distintas rutinas programadas para el cumplimiento de determinadas tareas con la finalidad de solucionar un problema específico.

- **SQL server**: Gestor de base de datos relacional el cual está desarrollado por Microsoft. Este gestor es utilizado para la manipulación y recuperación de datos.

- **Unit of Work**: (Unidad de trabajo). Es un patrón el cual tiene como finalidad de representar como una unidad, aquellos objetos que se están manipulando en un software.

2.3 Benchmarking

En la figura N° 17 se muestra el benchmarking en el cual muestra el análisis comparativo entre las diferentes software existentes y la solución propuesta.

BENCHMARKING				
ANÁLISIS COMPARATIVO		ENTITY FRAMEWORK	HIBERNATE	TESIS PROPUESTA
CARACTERÍSTICAS	Autenticación de base de datos Windows y SQL	3	3	3
	Facilidad de aprendizaje	2	1	3
	Exclusión de librerías externas	1	1	3
	Propagación de cambios de la BD al código sin rehacer	1	1	3
	Consulta/Alteración a tablas por columna referenciada	1	1	3
	Múltiples operaciones en una sola transacción	2	2	3
	Manejo de columnas SQL tipo salida y tipos complejos	1	1	3
	PUNTAJE	11	10	21
FUNCIONALIDADES	Generación de clases con el código a emplear	3	3	3
	Generación de scripts de base de datos	1	1	3
	Generación de las capas (proyectos) y la solución de VS	1	1	3
	Aplicación de estándares personalizados al código	1	1	3
	Aplicación de formatos personalizados a los scripts SQL	1	1	3
	Alteración de la configuración inicial en cualquier momento	1	1	3
	Aplicación del patrón MVC y Unit of Works	1	1	3
	PUNTAJE	9	9	21
PUNTAJE FINAL		**19**	**18**	**30**

Figura 17: Benchmarking
Fuente: Elaboración propia

En la figura N° 18 se muestra la leyenda utilizada para el benchmarking.

LEYENDA	
Bueno - Cumple	3
Regular - Cumple parcialmente	2
Malo - No cumple	1

Figura 18: Leyenda de benchmarking
Fuente: Elaboración propia

CAPÍTULO III: DESARROLLO DEL PROYECTO

3.1 Alcance del proyecto

3.1.1 Estructura del desglose del trabajo y entregables

En la figura N° 19 se muestra el desglose y entregables del proyecto.

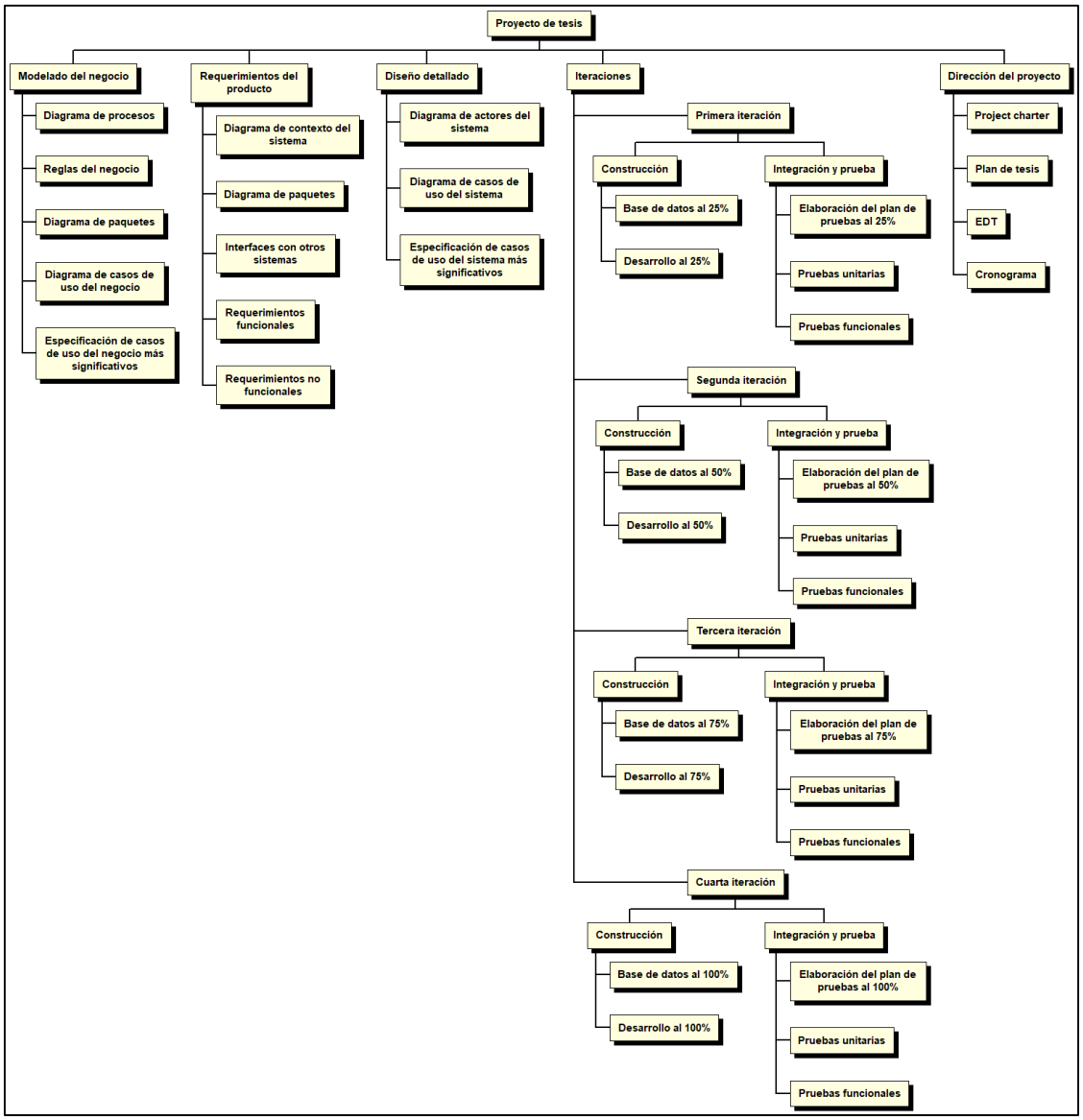


Figura 19: EDT
Fuente: Elaboración propia

3.1.2 Exclusiones del proyecto

- No abarca base de datos no relacionales (No SQL): Las bases de datos no relacionales no son parte de esta investigación.

- No se contempla el uso de lenguajes de programación distintos a C#: Si bien los procesos que realiza el framework pueden aplicarse en varios lenguajes de programación, para la demostración de esta investigación se emplea solo el lenguaje C#.
- No se contempla el gestor de base de datos distintos a SQL Server: Si bien los procesos que realiza el framework pueden aplicarse en varios motores de base de datos relacionales, para la demostración de esta investigación se emplea sólo Microsoft SQL Server.
- No se contempla el diseño del esquema de la base de datos del usuario: Como punto de partida es necesario que exista una base de datos relacional diseñada correctamente especificando sus restricciones de llaves primarias y foráneas.

3.1.3 Restricciones del proyecto

- El aplicativo, está enfocado en desarrolladores con conocimientos en programación con lenguaje C# como mínimo.
- El aplicativo requiere que el sistema operativo sea Windows 7 o superior.
- El usuario deberá tener instalado una versión del año 2015 o superior de Microsoft visual studio ó Visual studio code.
- El usuario deberá contar con su gestor de base de datos relacional para iniciar a usar el software y esta deberá ser SQL Server versión 2012 o superior.

3.1.4 Supuestos del proyecto

- Es necesario contar con las herramientas necesarias para el desarrollo del software.
- La infraestructura tecnológica deberá contar con un servidor de base de datos para realizar las pruebas.

3.1.5 Cronograma del proyecto

En la figura N° 20 se muestra el cronograma general del proyecto.

Task Name	Duration	Start	Finish
SOFTWARE DE GESTION DE PROYECTOS Y CONTROL DE BASES DE DATOS RELACIONES PARA DESARROLLADORES .NET	**128 days**	**Sat 15/06/19**	**Sun 20/10/19**
Modelado de Negocio	**7 days**	**Mon 01/07/19**	**Sun 07/07/19**
Requerimientos del producto	**7 days**	**Mon 08/07/19**	**Sun 14/07/19**
Diseño Detallado	**6 days**	**Tue 09/07/19**	**Sun 14/07/19**
Iteraciones	**97 days**	**Mon 15/07/19**	Sat 19/10/19
Dirección de proyectos	**16 days**	**Sat 15/06/19**	**Sun 30/06/19**

Figura 20: Cronograma general
Fuente: Elaboración propia

En la figura N° 21 se muestra el cronograma del modelado del negocio.

Task Name	Duration	Start	Finish
SOFTWARE DE GESTION DE PROYECTOS Y CONTROL DE BASES DE DATOS RELACIONES PARA DESARROLLADORES .NET	**128 days**	**Sat 15/06/19**	**Sun 20/10/19**
Modelado de Negocio	**7 days**	**Mon 01/07/19**	**Sun 07/07/19**
Diagrama de procesos	1 day	Mon 01/07/19	Mon 01/07/19
Reglas del negocio	1 day	Tue 02/07/19	Tue 02/07/19
Diagrama de paquetes	1 day	Wed 03/07/19	Wed 03/07/19
Diagrama de casos de uso de negocio	1 day	Thu 04/07/19	Thu 04/07/19
Especificación de casos de uso más significativos	2 days	Fri 05/07/19	Sat 06/07/19
Aprobación de documento de modelado de negocio	1 day	Sun 07/07/19	Sun 07/07/19

Figura 21: Modelado de negocio
Fuente: Elaboración propia

En la figura N° 22 se muestra el cronograma de los requerimientos del producto.

Task Name	Duration	Start	Finish
SOFTWARE DE GESTION DE PROYECTOS Y CONTROL DE BASES DE DATOS RELACIONES PARA DESARROLLADORES .NET	**128 days**	**Sat 15/06/19**	**Sun 20/10/19**
Modelado de Negocio	**7 days**	**Mon 01/07/19**	**Sun 07/07/19**
Requerimientos del producto	**7 days**	**Mon 08/07/19**	**Sun 14/07/19**
Diagrama del contexto del sistema	1 day	Mon 08/07/19	Mon 08/07/19
Diagrama de paquetes del sistema	1 day	Mon 08/07/19	Mon 08/07/19
Interfaces con otros sistemas	1 day	Mon 08/07/19	Mon 08/07/19
Requerimientos funcionales	1 day	Mon 08/07/19	Mon 08/07/19
Requerimientos no funcionales	1 day	Mon 08/07/19	Mon 08/07/19
Aprobación de documento de requerimientos del producto	1 day	Sun 14/07/19	Sun 14/07/19

Figura 22: Requerimientos del producto
Fuente: Elaboración propia

En la figura N° 23 se muestra el cronograma del diseño detallado.

Task Name	Duration	Start	Finish
SOFTWARE DE GESTION DE PROYECTOS Y CONTROL DE BASES DE DATOS RELACIONES PARA DESARROLLADORES .NET	128 days	Sat 15/06/19	Sun 20/10/19
Modelado de Negocio	7 days	Mon 01/07/19	Sun 07/07/19
Requerimientos del producto	7 days	Mon 08/07/19	Sun 14/07/19
Diseño Detallado	6 days	Tue 09/07/19	Sun 14/07/19
Diagrama de actores del sistema	2 days	Tue 09/07/19	Wed 10/07/19
Diagrama de casos de uso del sistema	2 days	Thu 11/07/19	Fri 12/07/19
Especificación de casos de uso del sistema más significativos	1 day	Sat 13/07/19	Sat 13/07/19
Aprobación de documento de diseño detallado	1 day	Sun 14/07/19	Sun 14/07/19

Figura 23: Diseño detallado
Fuente: Elaboración propia

En la figura N° 24 se muestra el cronograma de la primera iteración.

Task Name	Duration	Start	Finish
SOFTWARE DE GESTION DE PROYECTOS Y CONTROL DE BASES DE DATOS RELACIONES PARA DESARROLLADORES .NET	128 days	Sat 15/06/19	Sun 20/10/19
Modelado de Negocio	7 days	Mon 01/07/19	Sun 07/07/19
Requerimientos del producto	7 days	Mon 08/07/19	Sun 14/07/19
Diseño Detallado	6 days	Tue 09/07/19	Sun 14/07/19
Iteraciones	97 days	Mon 15/07/19	Sat 19/10/19
Primera iteración	13 days	Mon 15/07/19	Sat 27/07/19
Construcción	10 days	Mon 15/07/19	Wed 24/07/19
Base de datos al 25%	3 days	Mon 15/07/19	Wed 17/07/19
Desarrollo al 25%	7 days	Thu 18/07/19	Wed 24/07/19
Construcción del módulo de Configuración de Proyectos .NET	7 days	Thu 18/07/19	Wed 24/07/19
Aprobación de modulos al 25%	1 day	Sun 28/07/19	Sun 28/07/19
Integración y pruebas	3 days	Thu 25/07/19	Sat 27/07/19
Elaboración del plan de pruebas al 25%	1 day	Thu 25/07/19	Thu 25/07/19
Pruebas unitarias	2 days	Fri 26/07/19	Sat 27/07/19
Aprobación del plan de pruebas al 25%	1 day	Sun 28/07/19	Sun 28/07/19
Segunda iteración	27 days	Mon 29/07/19	Sat 24/08/19
Tercera iteración	27 days	Mon 26/08/19	Sat 21/09/19
Cuarta iteración	27 days	Mon 23/09/19	Sat 19/10/19
Dirección de proyectos	16 days	Sat 15/06/19	Sun 30/06/19

Figura 24: Primera iteración
Fuente: Elaboración propia

En la figura N° 25 se muestra el cronograma de la segunda iteración.

Task Name	Duration	Start	Finish
SOFTWARE DE GESTION DE PROYECTOS Y CONTROL DE BASES DE DATOS RELACIONES PARA DESARROLLADORES .NET	**128 days**	**Sat 15/06/19**	**Sun 20/10/19**
Modelado de Negocio	**7 days**	**Mon 01/07/19**	**Sun 07/07/19**
Requerimientos del producto	**7 days**	**Mon 08/07/19**	**Sun 14/07/19**
Diseño Detallado	**6 days**	**Tue 09/07/19**	**Sun 14/07/19**
Iteraciones	**97 days**	**Mon 15/07/19**	Sat 19/10/19
Primera iteración	**13 days**	**Mon 15/07/19**	**Sat 27/07/19**
Segunda iteración	**27 days**	**Mon 29/07/19**	**Sat 24/08/19**
Construcción	**21 days**	**Mon 29/07/19**	**Sun 18/08/19**
Base de datos al 50%	7 days	Mon 29/07/19	Sun 04/08/19
Desarrollo al 50%	**14 days**	**Mon 05/08/19**	**Sun 18/08/19**
Construcción del módulo de Configuración de BD SQL	14 days	Mon 05/08/19	Sun 18/08/19
Aprobación de modulos al 50%	1 day	Sun 25/08/19	Sun 25/08/19
Integración y pruebas	**6 days**	**Mon 19/08/19**	**Sat 24/08/19**
Elaboración del plan de pruebas al 50%	1 day	Mon 19/08/19	Mon 19/08/19
Pruebas unitarias	2 days	Tue 20/08/19	Wed 21/08/19
Pruebas de integración	3 days	Thu 22/08/19	Sat 24/08/19
Aprobación del plan de pruebas al 50%	1 day	Sun 25/08/19	Sun 25/08/19
Tercera iteración	**27 days**	Mon 26/08/19	Sat 21/09/19
Cuarta iteración	**27 days**	**Mon 23/09/19**	Sat 19/10/19
Dirección de proyectos	**16 days**	**Sat 15/06/19**	**Sun 30/06/19**

Figura 25: Segunda iteración
Fuente: Elaboración propia

En la figura N° 26 se muestra el cronograma de la tercera iteración.

Task Name	Duration	Start	Finish
SOFTWARE DE GESTION DE PROYECTOS Y CONTROL DE BASES DE DATOS RELACIONES PARA DESARROLLADORES .NET	**128 days**	**Sat 15/06/19**	**Sun 20/10/19**
Modelado de Negocio	**7 days**	**Mon 01/07/19**	**Sun 07/07/19**
Requerimientos del producto	**7 days**	**Mon 08/07/19**	**Sun 14/07/19**
Diseño Detallado	**6 days**	**Tue 09/07/19**	**Sun 14/07/19**
Iteraciones	**97 days**	**Mon 15/07/19**	Sat 19/10/19
Primera iteración	**13 days**	**Mon 15/07/19**	**Sat 27/07/19**
Segunda iteración	**27 days**	**Mon 29/07/19**	**Sat 24/08/19**
Tercera iteración	**27 days**	Mon 26/08/19	Sat 21/09/19
Construcción	**21 days**	**Mon 26/08/19**	**Sun 15/09/19**
Base de datos al 75%	7 days	Mon 26/08/19	Sun 01/09/19
Desarrollo al 75%	**14 days**	**Mon 02/09/19**	**Sun 15/09/19**
Construcción del módulo de Estándares SQL	14 days	Mon 02/09/19	Sun 15/09/19
Aprobación de modulos al 75%	1 day	Sun 22/09/19	Sun 22/09/19
Integración y pruebas	**6 days**	**Mon 16/09/19**	**Sat 21/09/19**
Elaboración del plan de pruebas al 75%	1 day	Mon 16/09/19	Mon 16/09/19
Pruebas unitarias	2 days	Tue 17/09/19	Wed 18/09/19
Pruebas de integración	3 days	Thu 19/09/19	Sat 21/09/19
Aprobación del plan de pruebas al 75%	1 day	Sun 22/09/19	Sun 22/09/19
Cuarta iteración	**27 days**	**Mon 23/09/19**	Sat 19/10/19
Dirección de proyectos	**16 days**	**Sat 15/06/19**	**Sun 30/06/19**

Figura 26: Tercera iteración
Fuente: Elaboración propia

En la figura N° 27 se muestra el cronograma de la cuarta iteración.

Task Name	Duration	Start	Finish
SOFTWARE DE GESTION DE PROYECTOS Y CONTROL DE BASES DE DATOS RELACIONES PARA DESARROLLADORES .NET	**128 days**	**Sat 15/06/19**	**Sun 20/10/19**
Modelado de Negocio	**7 days**	**Mon 01/07/19**	**Sun 07/07/19**
Requerimientos del producto	**7 days**	**Mon 08/07/19**	**Sun 14/07/19**
Diseño Detallado	**6 days**	**Tue 09/07/19**	**Sun 14/07/19**
Iteraciones	**97 days**	**Mon 15/07/19**	Sat 19/10/19
Primera iteración	**13 days**	**Mon 15/07/19**	**Sat 27/07/19**
Segunda iteración	**27 days**	**Mon 29/07/19**	**Sat 24/08/19**
Tercera iteración	**27 days**	Mon 26/08/19	Sat 21/09/19
Cuarta iteración	**27 days**	**Mon 23/09/19**	Sat 19/10/19
Construcción	**21 days**	**Mon 23/09/19**	**Sun 13/10/19**
Base de datos al 100%	7 days	Mon 23/09/19	Sun 29/09/19
Desarrollo al 100%	**14 days**	**Mon 30/09/19**	**Sun 13/10/19**
Construcción del módulo de Estándares .NET	14 days	Mon 30/09/19	Sun 13/10/19
Aprobación de modulos al 100%	1 day	Sun 20/10/19	Sun 20/10/19
Integración y pruebas	**6 days**	**Mon 14/10/19**	**Sat 19/10/19**
Elaboración del plan de pruebas al 100%	1 day	Mon 14/10/19	Mon 14/10/19
Pruebas unitarias	2 days	Tue 15/10/19	Wed 16/10/19
Pruebas de integración	3 days	Thu 17/10/19	Sat 19/10/19
Aprobación del plan de pruebas al 100%	1 day	Sun 20/10/19	Sun 20/10/19
Dirección de proyectos	**16 days**	**Sat 15/06/19**	**Sun 30/06/19**

Figura 27: Cuarta iteración
Fuente: Elaboración propia

En la figura N° 28 se muestra el cronograma de la dirección de proyectos.

Task Name	Duration	Start	Finish
SOFTWARE DE GESTION DE PROYECTOS Y CONTROL DE BASES DE DATOS RELACIONES PARA DESARROLLADORES .NET	**128 days**	**Sat 15/06/19**	**Sun 20/10/19**
Modelado de Negocio	**7 days**	**Mon 01/07/19**	**Sun 07/07/19**
Requerimientos del producto	**7 days**	**Mon 08/07/19**	**Sun 14/07/19**
Diseño Detallado	**6 days**	**Tue 09/07/19**	**Sun 14/07/19**
Iteraciones	**97 days**	**Mon 15/07/19**	Sat 19/10/19
Dirección de proyectos	**16 days**	**Sat 15/06/19**	**Sun 30/06/19**
Proyect Charter	1 day	Sat 15/06/19	Sat 15/06/19
EDT	1 day	Sun 16/06/19	Sun 16/06/19
Cronograma	3 days	Mon 17/06/19	Wed 19/06/19
Plan de tesis	11 days	Thu 20/06/19	Sun 30/06/19

Figura 28: Dirección de proyectos
Fuente: Elaboración propia

3.2 Alcance del Producto

3.2.1 Descripción del alcance del producto

1. Elaboración del asistente de configuración (Wizard):
   Un módulo el cual guía paso a paso a configurar el proyecto, la finalidad es facilitar la configuración inicial, debido a que son varios parámetros de configuración para la base de datos y para la codificación, y algunos dependen de una configuración previa. Si el usuario, decide omitir el asistente, siempre podrá hacerlo manualmente para esto elaboramos una interfaz de usuario para crear un nuevo proyecto o partir de un archivo de solución de visual studio existente, partiendo de esta configuración el aplicativo creará un tipo de archivo único que permitirá guardar las características del proyecto del usuario, y en este archivo se almacena toda la información relacionada con el proyecto.

2. Elaboración el módulo de configuración de la conexión a la base de datos:
   En cualquier momento el usuario podrá cambiar datos de la conexión a la base de datos, como el servidor, nombres, tablas usuarios y contraseñas. Por lo tanto, se tiene una interfaz de usuario para realizar estas operaciones de manera fácil e intuitiva.

3. Elaboración del módulo de configuración del proyecto:
   En cualquier momento el usuario podrá cambiar datos del proyecto, como son el nombre de las capas en la estructura, los prefijos y sufijos de las clases, procedimientos y aplicaciones. Por lo tanto, se tiene una interfaz de usuario para realizar estas operaciones de manera fácil e intuitiva.

4. Elaboración de la interfaz de configuración de la estructura:
   Cada usuario puede tener distintas formas de estandarizar la forma de escribir su código por eso, esta configuración se establece al codificar las clases, los métodos, y los procedimientos. Por lo tanto, se elabora una interfaz donde puedan configurarlo e incluso permitiéndole tener una configuración distinta para cada proyecto. Está estructura se muestra en un explorador de soluciones, donde visualizaremos previamente el trabajo que va realizar el software en la creación de la estructura del proyecto, los proyectos y las clases, antes de iniciar el proceso.

5. Elaboración de los procesos de creación de la estructura de la solución: Las clases, los métodos, los procedimientos almacenados, las conexiones y la interfaz de usuario de demostración en base a los parámetros especificados.
   El sistema iniciará una serie de operaciones que según las configuraciones iniciales, crea un proyecto de visual studio que puede ser web y/o de escritorio, crea la estructura de la codificación bajo el patrón Modelo Vista y Controlador, crea las clases para cada capa, crea los procedimientos almacenados y los ejecuta, crea las clases de conexión a la base de datos y las clases necesarias para aplicar el patrón de Unidad de Trabajo (Unit of Work), por último crea la aplicación de demostración, que puede ser web y/o de escritorio, esta aplicación tiene algunas operaciones básicas para demostrar al desarrollador la mejor manera de trabajar con el proyecto creado.

3.2.2 Criterios de aceptación del producto

- La demostración del software cumple la totalidad del alcance propuesto, esto se manifiesta con la culminación total del desarrollo de los casos de uso del flujo principal.
- El flujo principal debe funcionar en su totalidad, correctamente y en cualquier momento que se realice la demostración; para esto el framework propone una configuración recomendada, y un asistente para guiar al usuario en todo el proceso, pero deja abierta la posibilidad de configuración personalizada.
- El instalador del framework se dejará copia y se procederá a instalar en el ordenador que la universidad cree conveniente.
- El software tiene las validaciones para cada campo alterable por el usuario para reducir a la mínima posibilidad de ingresar datos errados.

CAPÍTULO IV: DESARROLLO DEL PRODUCTO

4.1 Modelado del Negocio

4.1.1 Diagramas de proceso

En la figura N° 29 se muestra el diagrama de procesos de negocio.

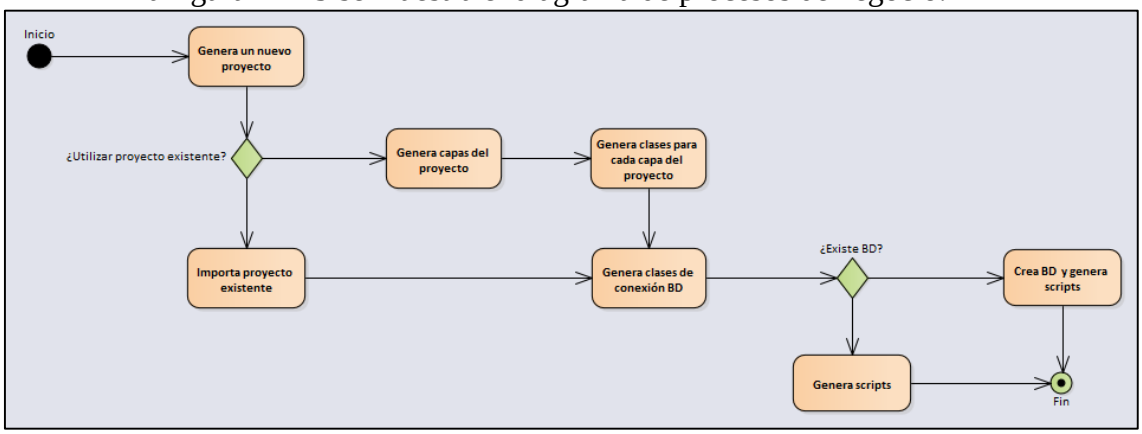


Figura 29: Diagrama de Proceso de Negocio
Fuente: Elaboración propia

4.1.2 Reglas del negocio

En la tabla 6 se muestra las reglas de negocio.

Tabla 6: Reglas de Negocio

RN1	Todo proyecto inicial parte de una base de datos relacional existente
RN2	Las llaves primarias y foráneas de las tablas de la base de datos relacional deben de estar correctamente creadas
RN3	Las tablas de la base de datos relacional existente deben estar correctamente relacionadas
RN4	Los prefijos de capas y clases podrán tener la opción de ser modificados sólo cuando se genere una nueva solución partiendo de un proyecto ya existente

Fuente: Elaboración propia

4.1.3 Diagrama de paquetes de negocio

En la figura N° 30 se muestra el diagrama general de paquetes del negocio.

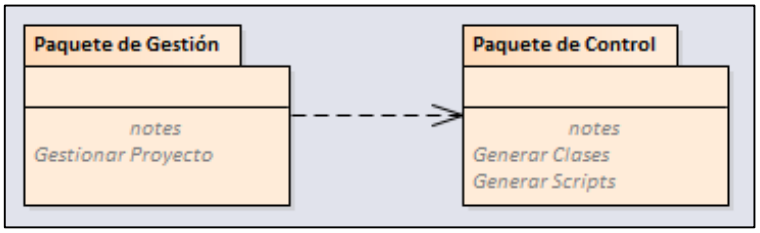


Figura 30: Diagrama General de Paquetes de Negocio
Fuente: Elaboración propia

En la figura N° 31 se muestra el diagrama de paquete de gestión.

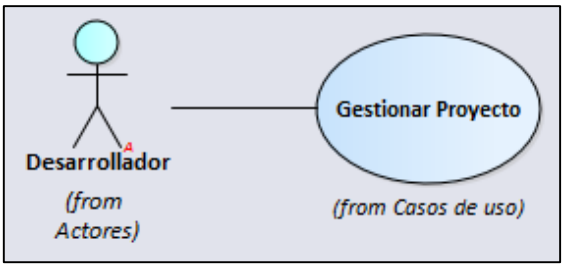


Figura 31: Diagrama de Paquete de Gestión
Fuente: Elaboración propia

En la figura N° 32 se muestra el diagrama de paquete de gestión.

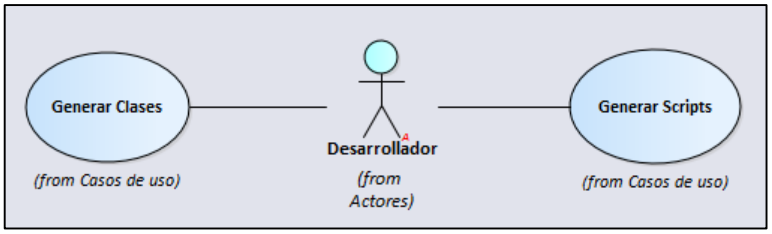


Figura 32: Diagrama de Paquete de Control
Fuente: Elaboración propia

4.1.4 Diagrama de actores de negocio

En la figura N° 33 se muestra el diagrama de actores de negocio.

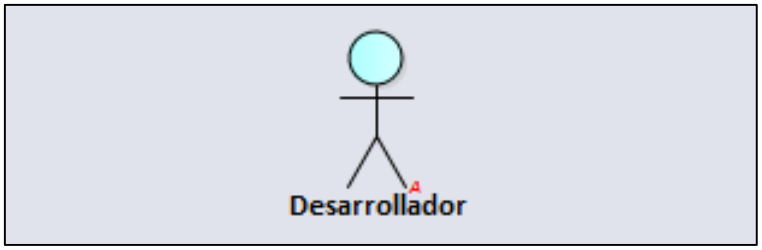


Figura 33: Diagrama de Actores de Negocio
Fuente: Elaboración propia

4.1.5 Diagrama de casos de uso de negocio

En la figura N° 34 se muestra el diagrama de casos de uso de negocio.

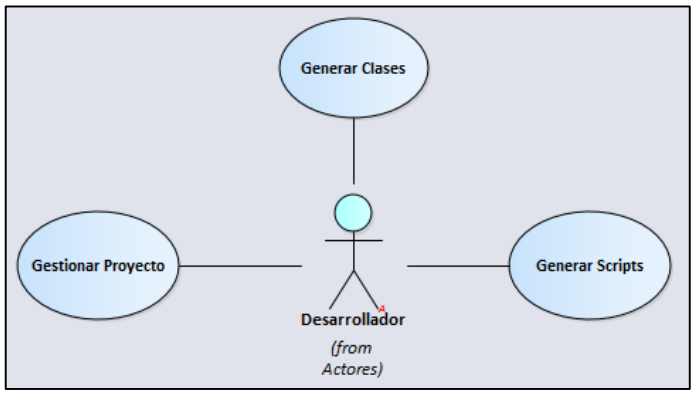


Figura 34: Diagrama de Casos de Uso de Negocio
Fuente: Elaboración propia

4.1.6 Especificación de CUN más significativos

4.1.6.1 Especificación "CUN Gestionar Proyecto"

En la tabla 7 se muestra la especificación del CUN Gestionar Proyecto.

Tabla 7: CUN Gestionar Proyecto

Nombre	CUN Gestionar Proyecto
Descripción	Proceso en el cual el Desarrollador gestiona un nuevo proyecto
Actor(es)	Desarrollador
Pre-Condición	-
Flujo de Eventos	1. El desarrollador ingresa al IDE 2. El desarrollador genera un nuevo proyecto e indica el tipo de proyecto que va desarrollar (desktop o web). 3. El desarrollador genera las capas del proyecto: - Capa de Entidades - Capa de Negocio - Capa de Acceso a Datos - Capa de Presentación
Flujo alternativo	-
Post-Condición	Se genera la solución del proyecto

Fuente: Elaboración propia

En la figura N° 35 se muestra el diagrama del CUN Gestionar Proyecto.

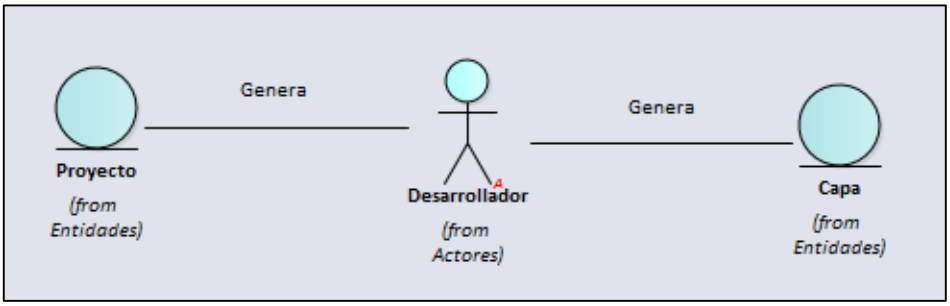


Figura 35: Diagrama de entidades y relación del CUN Gestionar Proyecto
Fuente: Elaboración propia

En la figura N° 36 se muestra el diagrama de actividades del CUN Gestionar Proyecto.

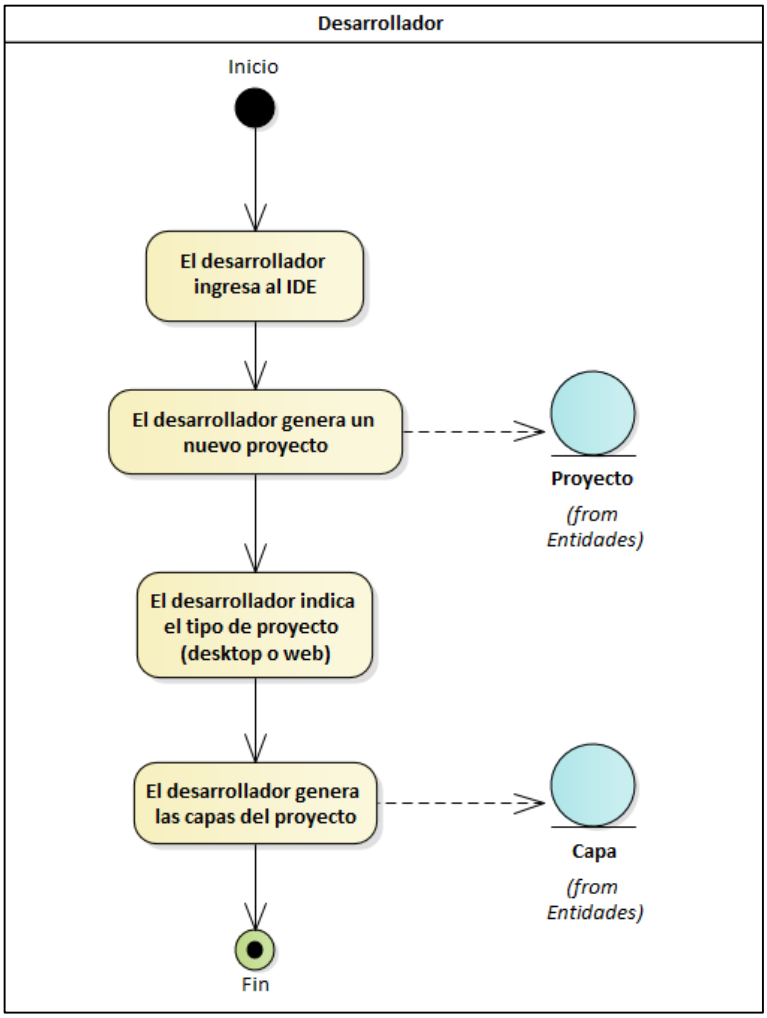


Figura 36: Diagrama de actividades del CUN Gestionar Proyecto
Fuente: Elaboración propia

4.1.6.2 Especificación “CUN Generar Clases”

En la tabla 8 se muestra la especificación del CUN Generar Clases.

Tabla 8: CUN Generar Clases

Nombre	CUN Generar Clases
Descripción	Proceso en el cual el Desarrollador genera las clases de las capas del proyecto
Actor(es)	Desarrollador
Pre-Condición	Ya existen capas generadas en la solución del proyecto
Flujo de Eventos	1. El desarrollador ingresa a la capa de entidades y genera una clase por cada entidad que se utilizarán en el proyecto. 2. El desarrollador ingresa a la capa de acceso a datos y genera los 4 métodos básicos por cada entidad descrita anteriormente: - Método de Inserción - Método de Actualización - Método de Lectura - Método de Eliminación 3. El desarrollador genera la clase de conexión a la base de datos. 4. El desarrollador ingresa a la capa de negocio y genera los métodos respectivos para realizar el llamado a la capa de acceso a datos. 5. El desarrollador ingresa a la capa de presentación, genera las interfaces y genera los métodos respectivos para realizar el llamado a la capa de negocio.
Flujo alternativo	-
Post-Condición	Se genera la clase de conexión a la BD, las entidades, interfaces y los métodos de las capas de negocio y acceso a datos

Fuente: Elaboración propia

En la figura N° 37 se muestra el diagrama del CUN Generar Clases.

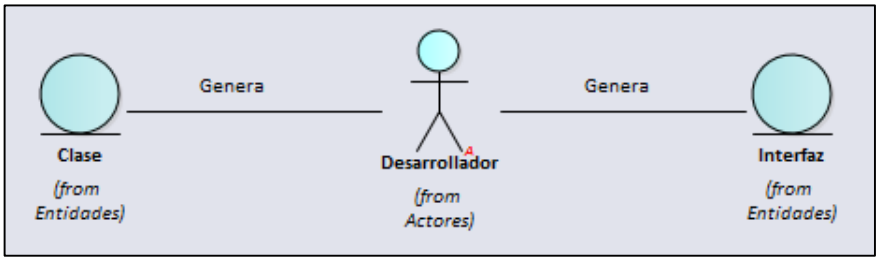


Figura 37: Diagrama de entidades y relación del CUN Generar Clases
Fuente: Elaboración propia

En la figura N° 38 se muestra el diagrama de actividades del CUN Generar Clases

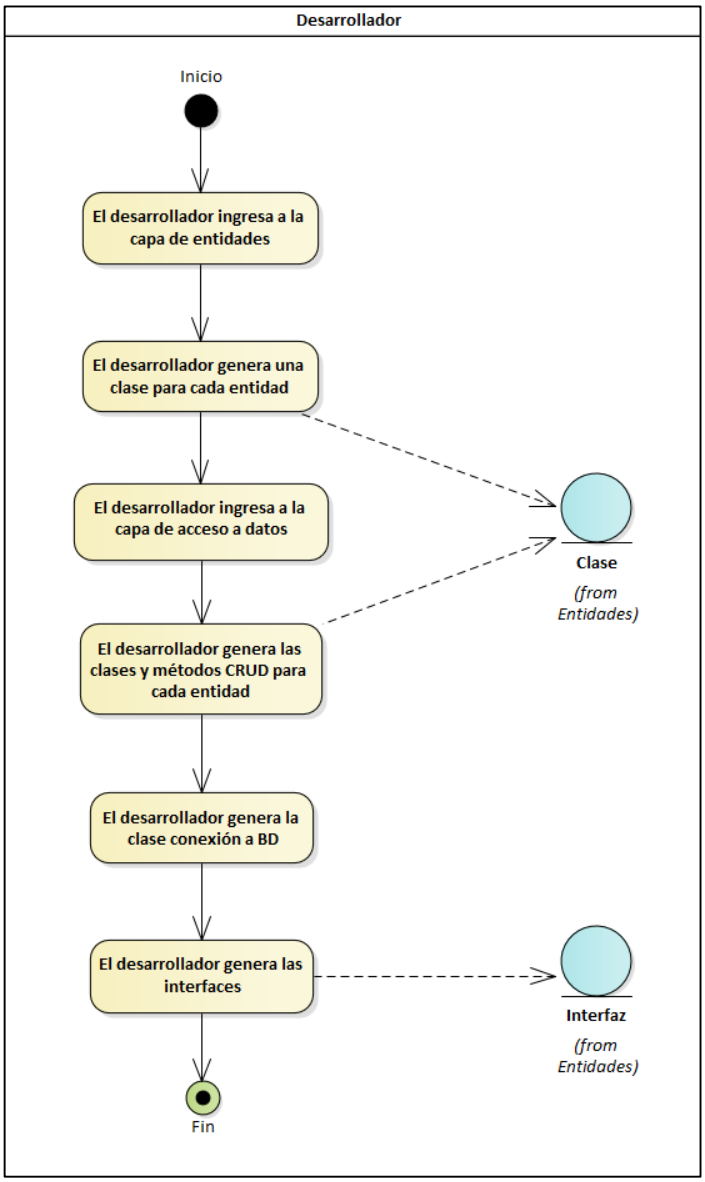


Figura 38: Diagrama de actividades del CUN Generar Clases
Fuente: Elaboración propia

4.1.6.3 Especificación “CUN Generar scripts”

En la tabla 9 se muestra la especificación del CUN Generar Scripts.

Tabla 9: CUN Generar Scripts

Nombre	CUN Generar Scripts
Descripción	Proceso en el cual el Desarrollador genera los procedimientos almacenados de inserción, actualización, lectura y eliminación para cada entidad de la BD
Actor(es)	Desarrollador
Pre-Condición	La BD cuenta con tablas creadas y con información
Flujo de Eventos	1. El desarrollador ingresa al gestor de base de datos 2. El desarrollador genera los 4 procedimientos almacenados básicos por cada tabla existente de la base de datos: - Procedimiento de Inserción - Procedimiento de Actualización - Procedimiento de Lectura - Procedimiento de Eliminación
Flujo alternativo	-
Post-Condición	Se genera la clase de conexión a la BD, las entidades, interfaces y los métodos de las capas de negocio y acceso a datos.

Fuente: Elaboración propia

En la figura N° 39 se muestra el diagrama del CUN Generar Scripts.

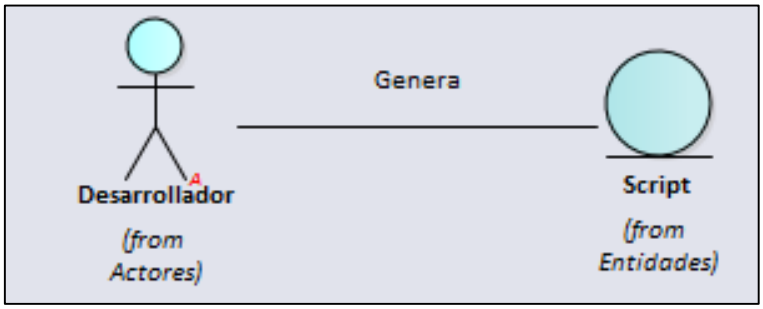


Figura 39: Diagrama de entidades y relación del CUN Generar Scripts
Fuente: Elaboración propia

En la figura N° 40 se muestra el diagrama de actividades del CUN Generar Scripts

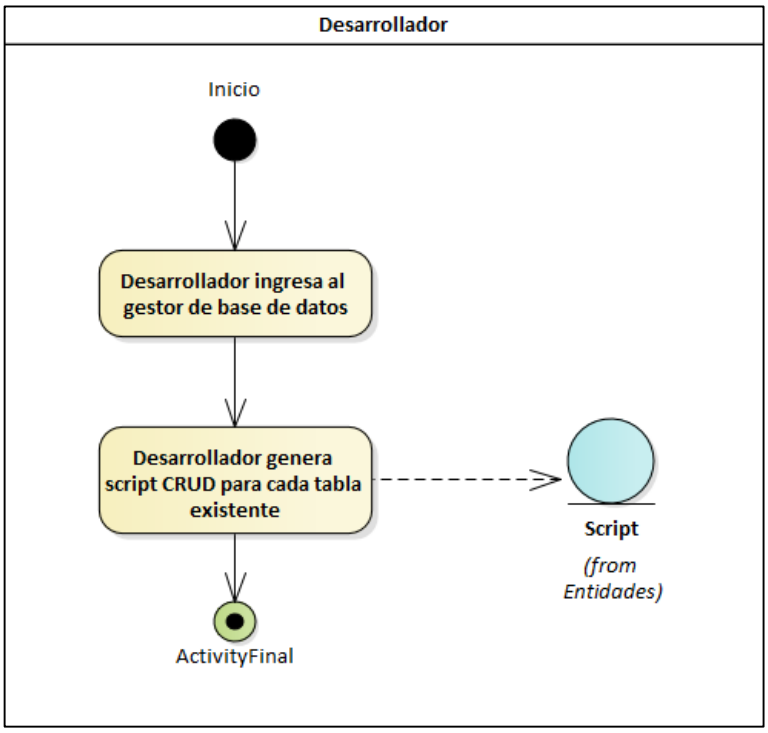


Figura 40: Diagrama de actividades del CUN Generar Scripts
Fuente: Elaboración propia

4.2 Requerimientos del producto software

4.2.1 Diagrama de paquetes

En la figura N° 41 se muestra el diagrama general de paquetes del sistema.

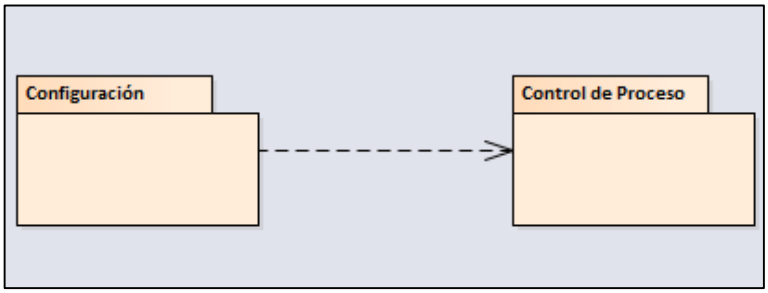


Figura 41: Diagrama General de Paquetes del Sistema
Fuente: Elaboración propia

En la figura N° 42 se muestra el diagrama del Paquete de Configuración.

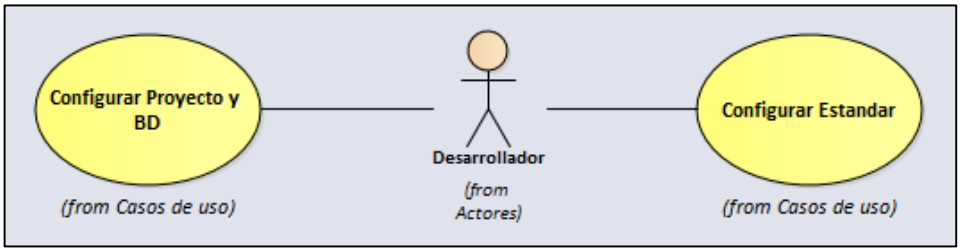


Figura 42: Diagrama del Paquete de Configuración
Fuente: Elaboración propia

En la figura N° 43 se muestra el diagrama del Paquete de Control de Proceso.

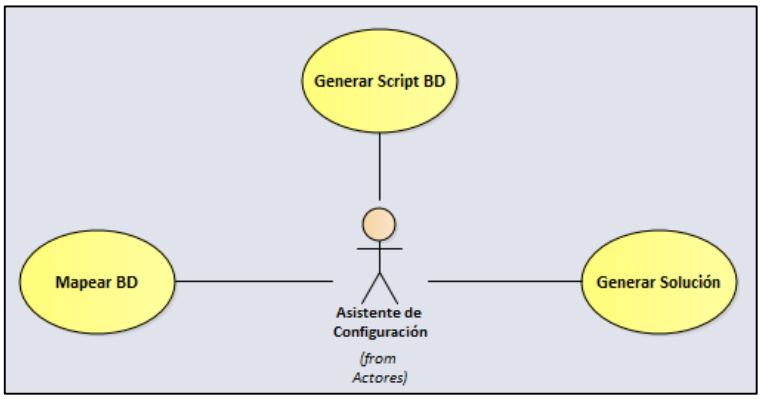


Figura 43: Diagrama de Paquete de Control de Proceso
Fuente: Elaboración propia

4.2.2 Interfaces con otros sistemas

No aplica

4.2.3 Requerimientos funcionales

a) RF01: Gestionar la información de datos para la generación de un proyecto.
b) RF02: Gestionar la información de datos para generar el mapeo con la base de datos.
c) RF03: Generar el mapeo de la base de datos y crear las clases de manera automática.
d) RF04: Generar los scripts de la base de datos, ya sea internos (ejecutable desde un directorio) o externos (ejecutable desde la base de datos.

4.2.4 Requerimientos no funcionales

a) Usabilidad: Las interfaces del proyecto son fáciles de entender para el usuario.

b) Confiabilidad: La información que se ingresa en la configuración inicial permitirá la generación de una solución con data precisa y exacta.

c) Disponibilidad: La herramienta está disponible en cualquier momento.

4.2.5 Diagrama de actores del sistema

En la figura N° 44 se muestra el diagrama de actores del sistema.

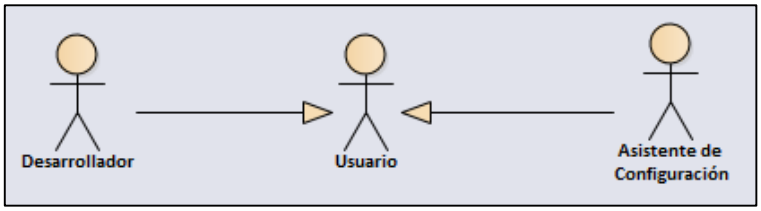


Figura 44: Diagrama de actores del sistema
Fuente: Elaboración propia

4.2.6 Diagrama de casos de uso del sistema

En la figura N° 45 se muestra el diagrama general de casos de uso del sistema.

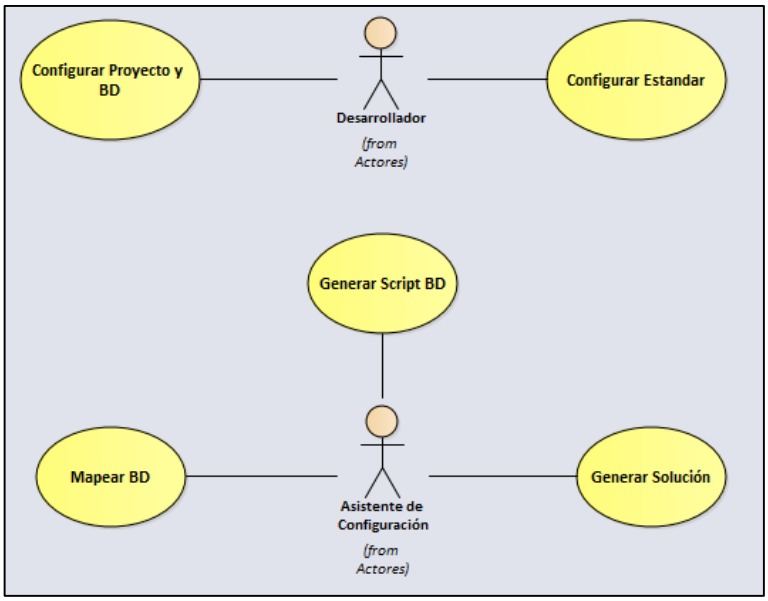


Figura 45: Diagrama general de casos de uso del sistema
Fuente: Elaboración propia

4.2.7 Especificación de CUS más significativos

4.2.7.1 Especificación “CUS Configurar Proyecto y BD”

En la tabla 10 se muestra la especificación del CUS Configurar Proyecto y BD.

Tabla 10: CUS Configurar Proyecto y BD

Nombre	CUS Configurar Proyecto y BD
Descripción	Proceso en el cual el Desarrollador configura un nuevo proyecto
Actor(es)	Desarrollador
Pre-Condición	-
Flujo de Eventos	1. El desarrollador ingresa al asistente de configuración del sistema. 2. El desarrollador selecciona un nuevo proyecto 3. El desarrollador selecciona el servidor y la base de datos que se va a utilizar en para el proyecto. 4. El desarrollador selecciona las vistas y las tablas de la base de datos seleccionada que se va a utilizar en el proyecto. 5. El desarrollador selecciona la opción para generar el script de la base de datos de forma externa.
Flujo alternativo	-
Post-Condición	Se seleccionó el proyecto y la base de datos que se va a utilizar

Fuente: Elaboración propia

En la figura N° 46 se muestra la interfaz para configurar el proyecto (1).

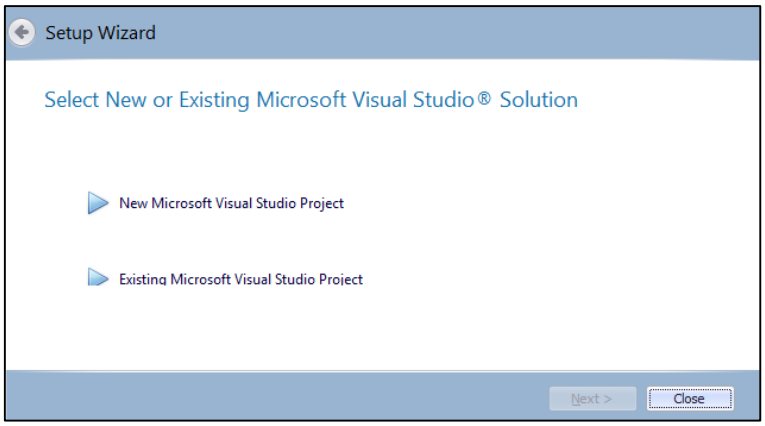


Figura 46: Interfaz Configurar Proyecto N° 1
Fuente: Elaboración propia

En la figura N° 47 se muestra la interfaz para configurar el proyecto (2).

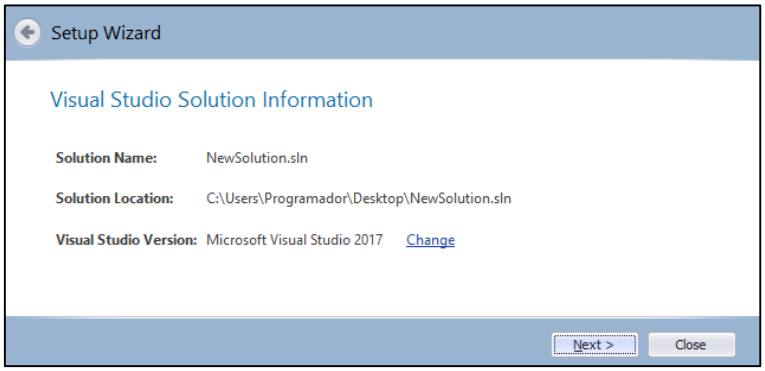


Figura 47: Interfaz Configurar Proyecto N° 2
Fuente: Elaboración propia

En la figura N° 48 se muestra la interfaz para configurar la BD (1).

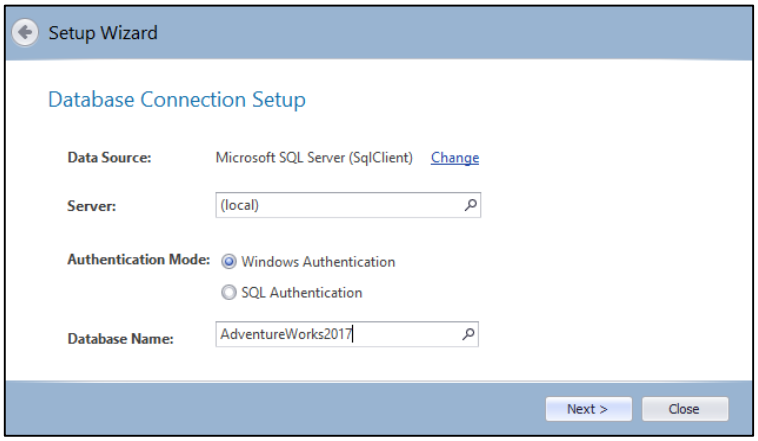


Figura 48: Interfaz Configurar BD N° 1
Fuente: Elaboración propia

En la figura N° 49 se muestra la interfaz para configurar la BD (2).

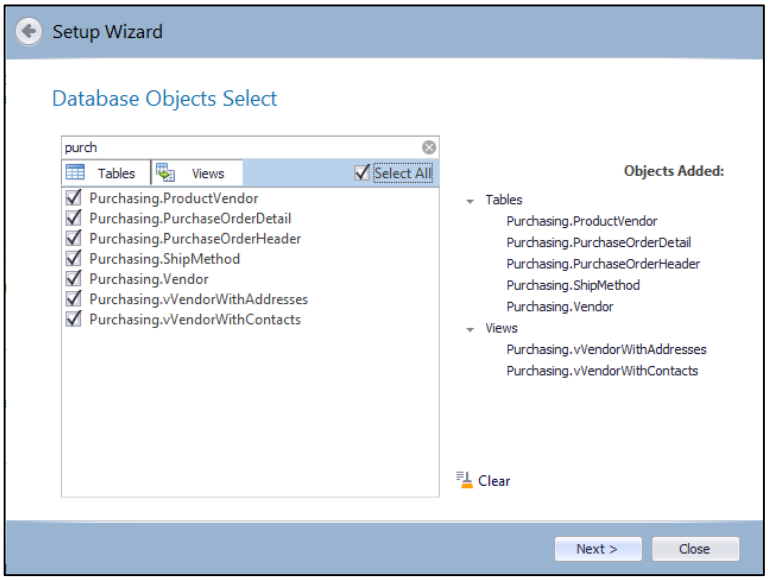


Figura 49: Interfaz Configurar BD N° 2
Fuente: Elaboración propia

4.2.7.2 Especificación "CUS Configurar Estándar"

En la tabla 11 se muestra la especificación del CUS Configurar Estándar.

Tabla 11: CUS Configurar Estándar

Nombre	CUS Configurar Estándar
Descripción	Proceso en el cual el Desarrollador configura los estándares de los prefijos de la BD y la estructura del proyecto
Actor(es)	Desarrollador
Pre-Condición	Se seleccionó el proyecto y la base de datos a utilizar
Flujo de Eventos	1. El desarrollador ingresa el prefijo que tendrán los procedimientos almacenados. (Prefijo: SP) 2. El desarrollador ingresa el prefijo que tendrán los procedimientos almacenados dependiendo a su funcionalidad: - Para inserción: SP_INS - Para actualización: SP_UPD - Para lectura: SP_SEL - Para eliminación: SP_DEL 3. El desarrollador ingresa el prefijo de las capas de la solución del proyecto: - Para capa de entidades: EN_NombreProyecto - Para capa de controlador: CTRL_NombreProyecto - Para capa de vista: VW_NombreProyecto El desarrollador finaliza la configuración de estándares y selecciona la opción de procesar.
Flujo alternativo	-
Post-Condición	Se configuró los estándares del proyecto y BD

Fuente: Elaboración propia

En la figura N° 50 se muestra la interfaz para configurar estándar de BD.

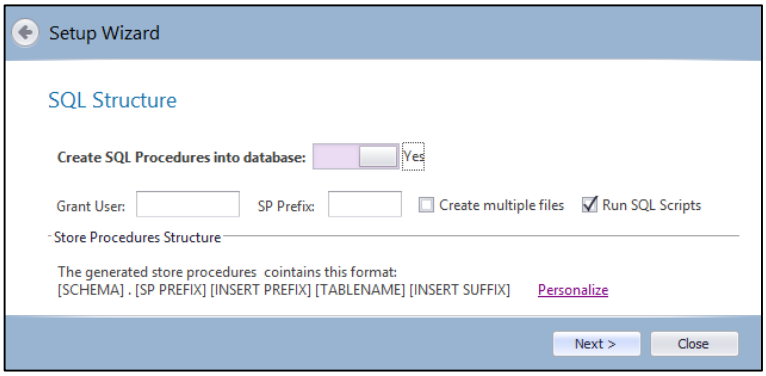


Figura 50: Interfaz Configurar estándar de BD
Fuente: Elaboración propia

En la figura N° 51 se muestra la interfaz para configurar estándar de proyecto.

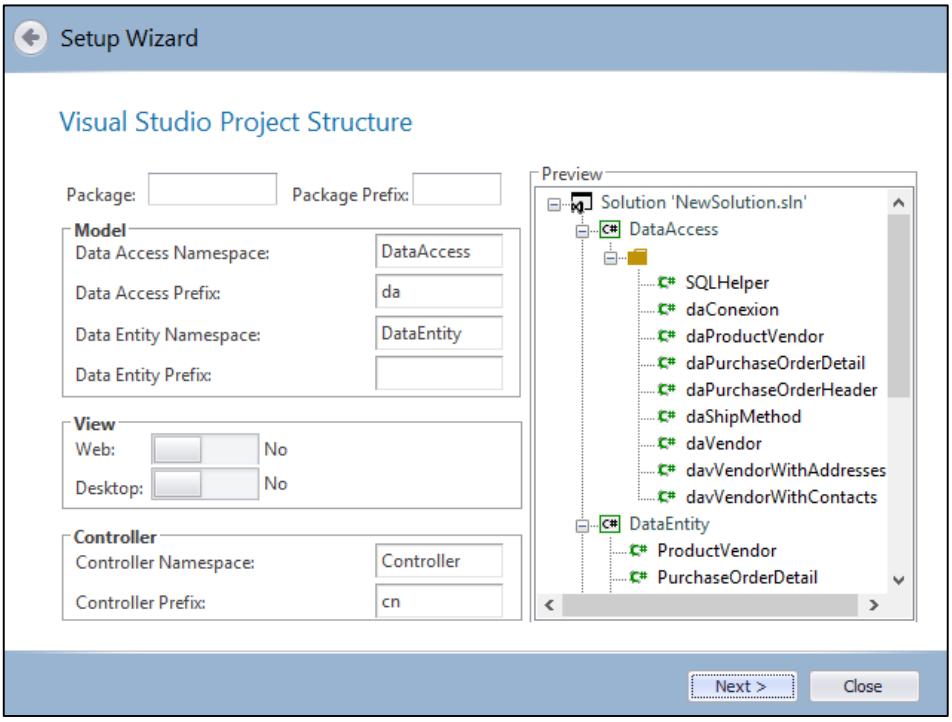


Figura 51: Interfaz Configurar estándar de Proyecto
Fuente: Elaboración propia

En la figura N° 52 se muestra la interfaz de resumen de la configuración.

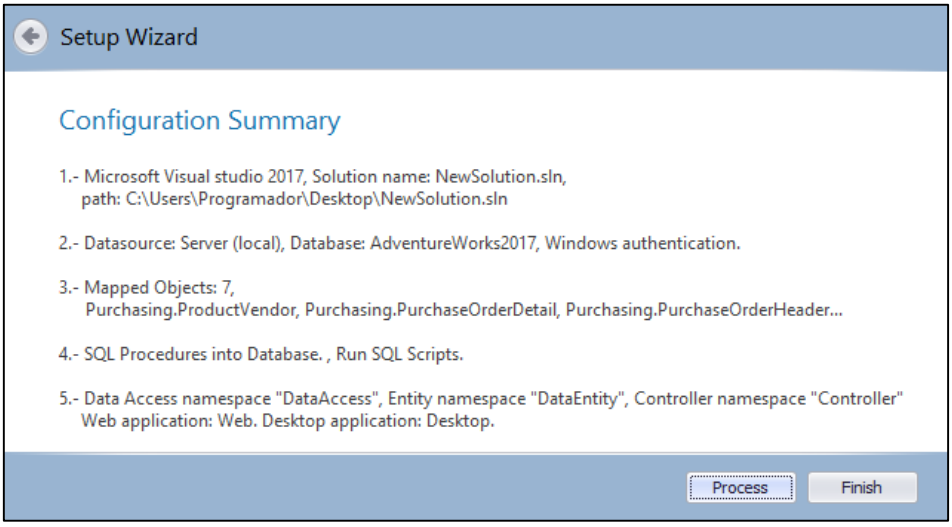


Figura 52: Interfaz Resumen de Configuración
Fuente: Elaboración propia

4.2.7.3 Especificación "CUS Mapear BD"

En la tabla 12 se muestra la especificación del CUS Mapear BD.

Tabla 12: CUS Mapear BD

Nombre	CUS Mapear BD
Descripción	Proceso en el cual el asistente de configuración convierte las tablas de la base de datos y los transforma en clases del proyecto
Actor(es)	Asistente de Configuración
Pre-Condición	Se configuró los prefijos de las clases del proyecto
Flujo de Eventos	1. El asistente de configuración, obtiene la información de la base de datos seleccionada en el caso de uso "Gestionar proyecto". 2. El asistente de configuración toma las tablas seleccionadas y realiza una lectura para obtener sus respectivos atributos. El asistente de configuración toma los atributos de cada una de las tablas de la base de datos y genera las clases con sus respectivos atributos en el proyecto.
Flujo alternativo	-
Post-Condición	Se genera una clase con sus respectivos atributos por cada tabla seleccionada de la base de datos.

Fuente: Elaboración propia

4.2.7.4 Especificación "CUS Generar Script BD"

En la tabla 13 se muestra la especificación del CUS Generar Script BD.

Tabla 13: CUS Generar Script BD

Nombre	CUS Generar Script BD
Descripción	Proceso en el que el asistente de configuración genera los scripts de la base de datos (procedimientos almacenados) de acuerdo a las clases creadas anteriormente.
Actor(es)	Asistente de Configuración
Pre-Condición	Se generó una clase con sus respectivos atributos por cada tabla seleccionada de la base de datos.
Flujo de Eventos	1. El asistente de configuración obtiene las clases con sus respectivos atributos por cada tabla seleccionada. 2. El asistente de configuración genera los procedimientos almacenados por cada clase generada. 3. El asistente de configuración ejecuta cada procedimiento almacenado en la base de datos.
Flujo alternativo	-
Post-Condición	Se generó y se ejecutó los scripts de la base de datos.

Fuente: Elaboración propia

4.2.7.5 Especificación “CUS Generar Solución”

En la tabla 14 se muestra la especificación del CUS Generar Solución.

Tabla 14: CUS Generar Solución

Nombre	CUS Generar Solución
Descripción	Proceso en el que el asistente de configuración genera las capas de acceso a datos, controlador e interfaz en el proyecto.
Actor(es)	Asistente de Configuración
Pre-Condición	Se generó una clase con sus respectivos atributos por cada tabla seleccionada de la base de datos.
Flujo de Eventos	1. El asistente de configuración obtiene las clases con sus respectivos atributos por cada tabla seleccionada. 2. El asistente de configuración genera la capa de acceso a datos con sus respectivas clases por cada entidad existente del proyecto. El asistente de configuración genera la capa de controladores con sus respectivas clases y métodos que hacen el llamado a la capa de acceso a datos.
Flujo alternativo	-
Post-Condición	Se generó la solución total del proyecto.

Fuente: Elaboración propia

4.3 Análisis y Diseño

4.3.1 Análisis

4.3.1.1 Realización de caso de uso análisis “Configurar Proyecto y BD”

En la figura N° 53 se muestra la realización de uso análisis del caso de uso “Configurar Proyecto y BD”.

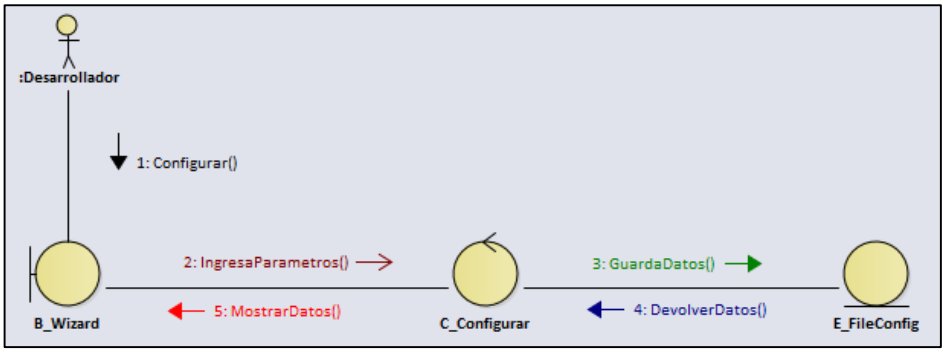


Figura 53: Realización de caso de uso análisis "Configurar Proyecto y BD"
Fuente: Elaboración propia

4.3.1.2 Realización de caso de uso análisis “Configurar Estándar”

En la figura N° 54 se muestra la realización de uso análisis del caso de uso “Configurar Estándar”.

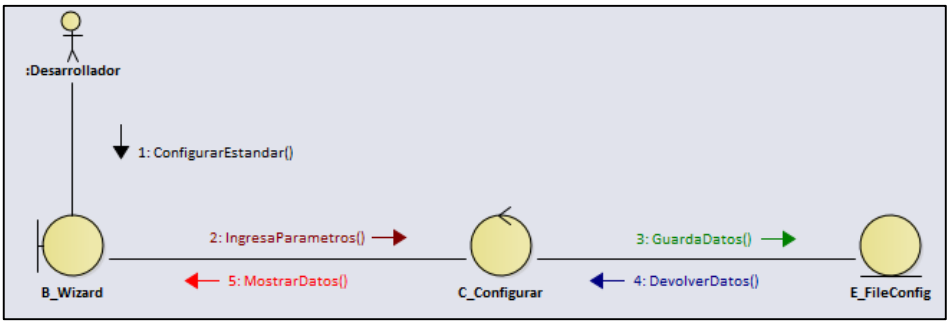


Figura 54: Realización de caso de uso análisis "Configurar Estándar"
Fuente: Elaboración propia

4.3.1.3 Realización de caso de uso análisis “Mapear BD”

En la figura N° 55 se muestra la realización de uso análisis del caso de uso “Mapear BD”.

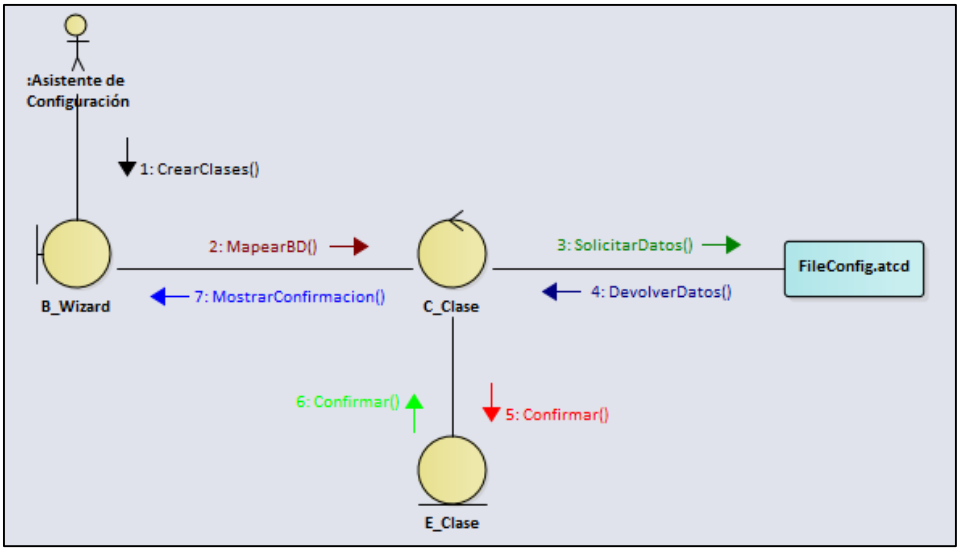


Figura 55: Realización de caso de uso análisis "Mapear BD"
Fuente: Elaboración propia

4.3.1.4 Realización de caso de uso análisis “Generar Script BD”

En la figura N° 56 se muestra la realización de uso análisis del caso de uso “Generar Script BD”.

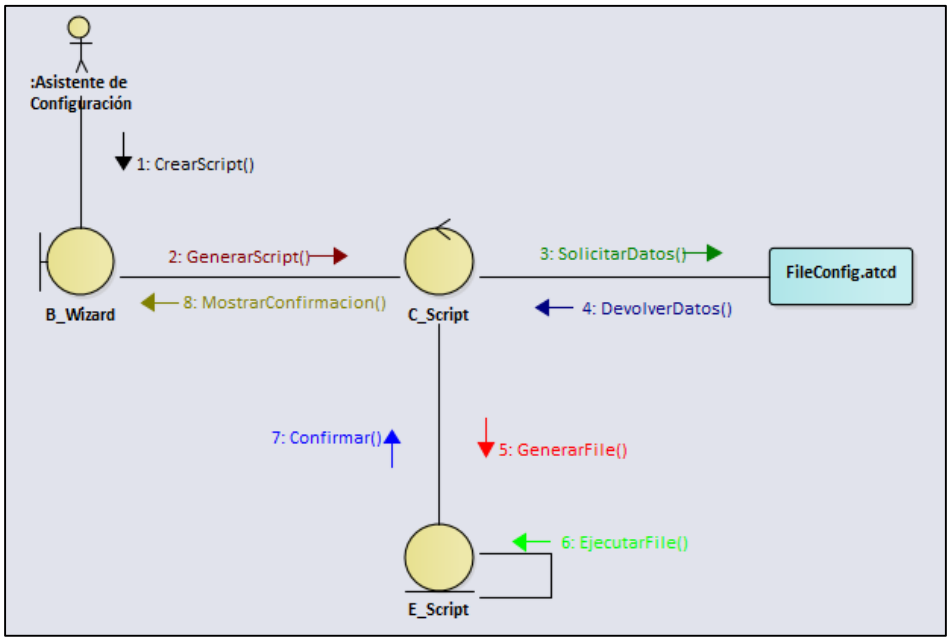


Figura 56: Realización de caso de uso análisis "Generar Script BD"
Fuente: Elaboración propia

4.3.1.5 Realización de caso de uso análisis “Generar Solución”

En la figura N° 57 se muestra la realización de uso análisis del caso de uso “Generar Solución”.

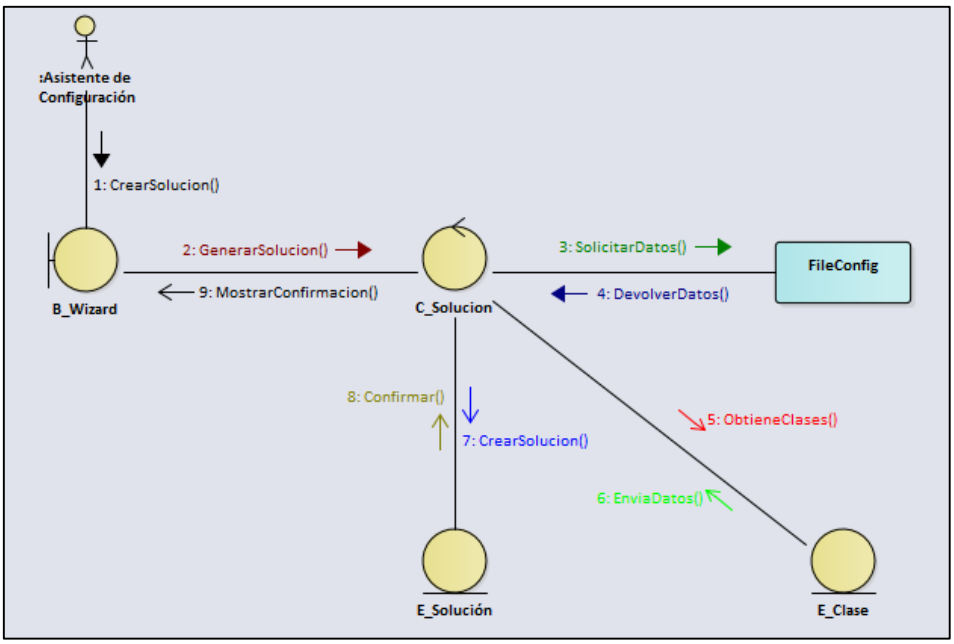


Figura 57: Realización de caso de uso análisis "Generar Solución"
Fuente: Elaboración propia

4.3.2 Diseño

4.3.2.1 Diagrama de secuencia de diseño “Configurar Proyecto y BD”

En la figura N° 58 se muestra el diagrama de secuencia de diseño del caso de uso “Configurar Proyecto y BD”.

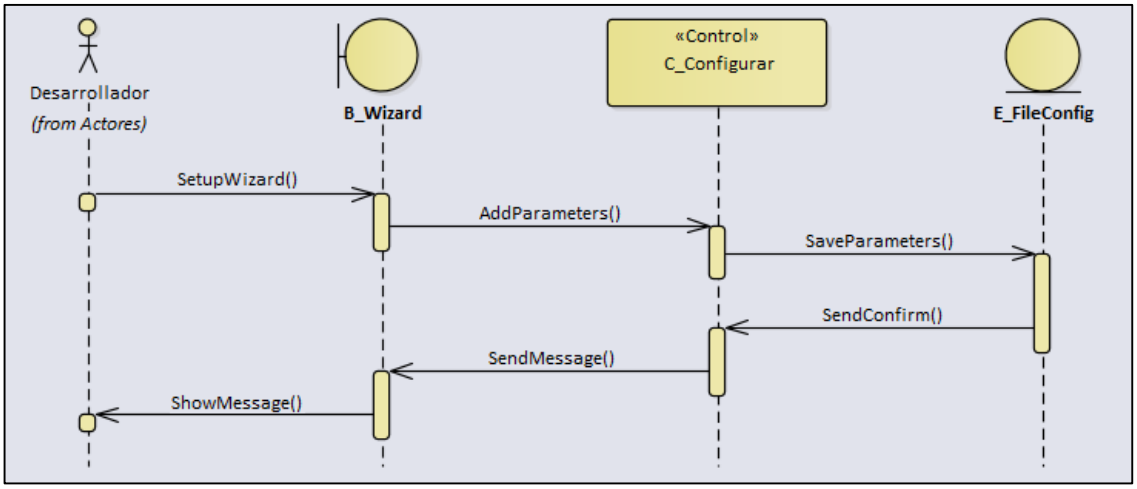


Figura 58: Diagrama de secuencia de diseño "Configurar Proyecto y BD"
Fuente: Elaboración propia

4.3.2.2 Diagrama de secuencia de diseño “Configurar Estándar”

En la figura N° 59 se muestra el diagrama de secuencia de diseño del caso de uso “Configurar Estándar”.

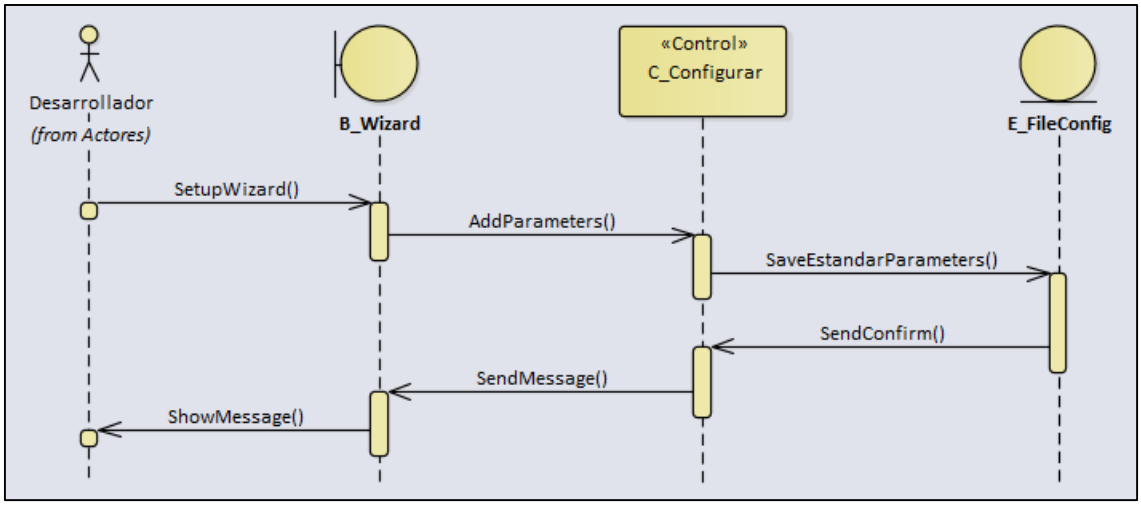


Figura 59: Diagrama de secuencia de diseño "Configurar Estándar"
Fuente: Elaboración propia

4.3.2.3 Diagrama de secuencia de diseño “Mapear BD”

En la figura N° 60 se muestra el diagrama de secuencia de diseño del caso de uso “Mapear BD”.

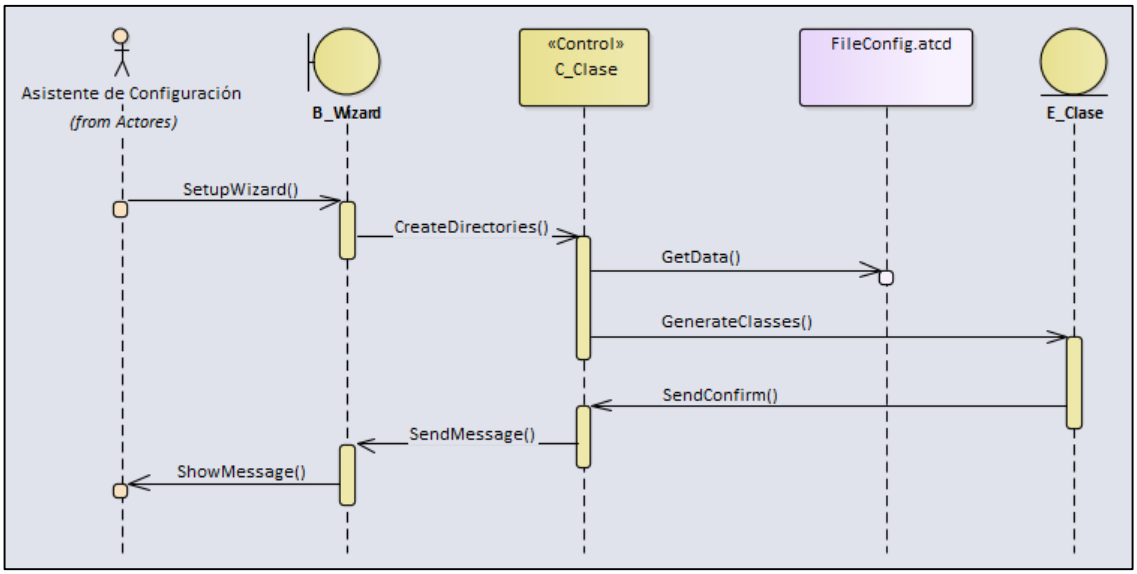


Figura 60: Diagrama de secuencia de diseño “Mapear BD”
Fuente: Elaboración propia

4.3.2.4 Diagrama de secuencia de diseño “Generar Script BD”

En la figura N° 61 se muestra el diagrama de secuencia de diseño del caso de uso “Generar Script BD”.

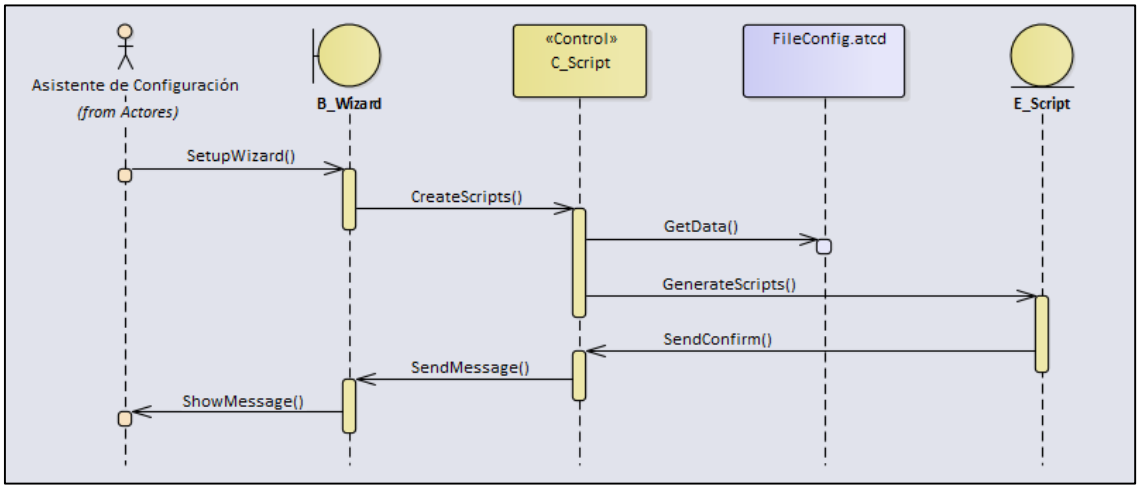


Figura 61: Diagrama de secuencia de diseño “Generar Script BD”
Fuente: Elaboración propia

4.3.2.5 Diagrama de secuencia de diseño “Generar Solución”

En la figura N° 62 se muestra el diagrama de secuencia de diseño del caso de uso “Generar Solución”.

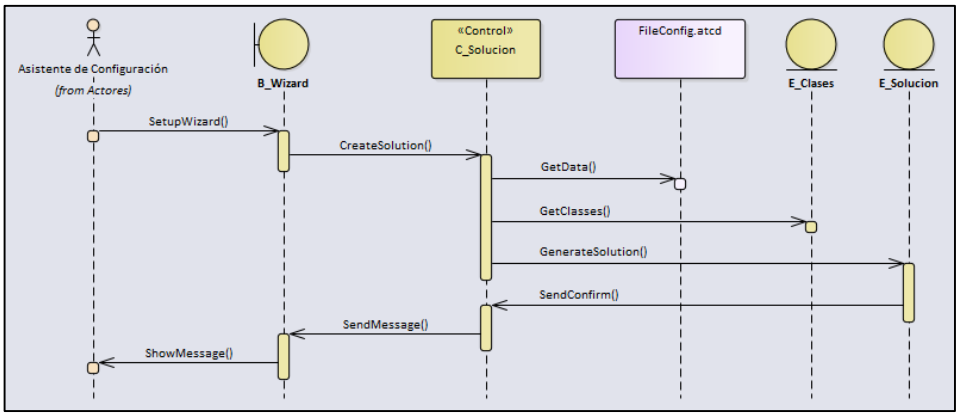


Figura 62: Diagrama de secuencia de diseño “Generar Solución”
Fuente: Elaboración propia

4.3.3 Modelo de Datos

4.3.3.1 Modelo lógico

En la figura N° 63 se muestra el modelo lógico del sistema.

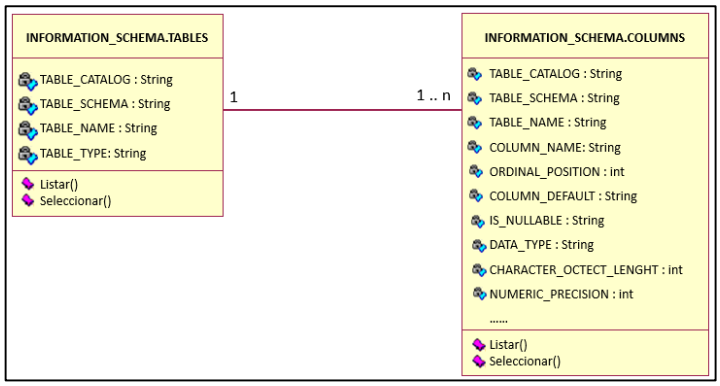


Figura 63: Modelo Lógico
Fuente: Elaboración propia

4.3.3.2 Modelo físico

En la figura N° 64 se muestra el modelo físico del sistema.

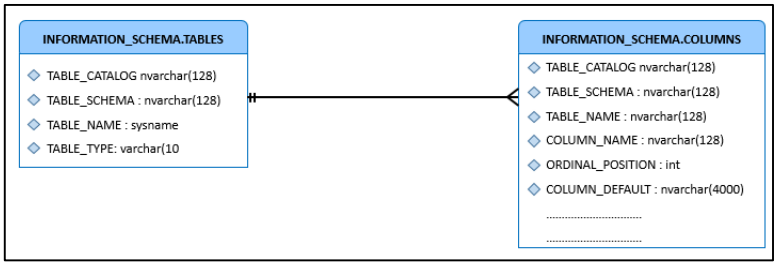


Figura 64: Modelo Físico
Fuente: Elaboración propia

4.3.3.3 Estructura archivo XML

En la figura N° 65 se muestra la estructura XML del archivo de configuración.

```xml
<?xml version="1.0" encoding="utf-8"?>
<Genesys>
    <Solution>
        <Name></Name>
        <VisualStudioVersion></VisualStudioVersion>
        <SqlServerVersion></SqlServerVersion>
        <Path></Path>
        <SolutionPath></SolutionPath>
    </Solution>
    <Project>
        <Name></Name>
        <ProjectPreffix></ProjectPreffix>
        <DataAcces>
            <Namespace></Namespace>
            <Prefix></Prefix>
        </DataAcces>
        <DataEntity>
            <Namespace></Namespace>
            <Prefix></Prefix>
        </DataEntity>
        <Controller>
            <Namespace></Namespace>
            <Prefix></Prefix>
        </Controller>
        <View>
            <WebNamespace></WebNamespace>
            <DesktopNamespace></DesktopNamespace>
            <ConsoleNamespace></ConsoleNamespace>
        </View>
        <SelectedTables></SelectedTables>
        <CustomQuerys>
            <CommonColumns>
                <ColumnName></ColumnName>
                <Query></Query>
            </CommonColumns>
        </CustomQuerys>
    </Project>
    <Connection>
        <Server></Server>
        <DatabaseName></DatabaseName>
        <Autentication>
            <LoginName>
            </LoginName>
            <Password>
            </Password>
        </Autentication>
        <Programmability>
            <GrantUser>
            </GrantUser>
            <SPPrefix>
            </SPPrefix>
        </Programmability>
    </Connection>
</Genesys>
```

Figura 65: Estructura archivo XML
Fuente: Elaboración propia

4.3.3.4 Diccionario de datos

En la tabla 15 se muestra el diccionario de datos de la tabla del sistema de gestor de base de datos INFORMATION_SCHEMA.TABLES

Tabla 15: Diccionario de datos de INFORMATION_SCHEMA.TABLES

INFORMATION_SCHEMA.TABLES		
Esta vista del sistema nos proporciona el acceso a la información de todas aquellas tablas y vistas de las diferentes bases de datos que existen.		
Columna	**Tipo de Dato**	**Descripción**
TABLE_CATALOG	nvarchar(128)	Nombre del proyecto o base de datos
TABLE_SCHEMA	nvarchar(128)	Nombre del esquema
TABLE_NAME	sysname	Nombre de la tabla
TABLE_TYPE	varchar(10)	Tabla o Vista

Fuente: Elaboración propia

En la tabla 16 se muestra el diccionario de datos de la tabla del sistema de gestor de base de datos INFORMATION_SCHEMA.COLUMNS

Tabla 16: Diccionario de datos de INFORMATION_SCHEMA.COLUMNS (Continua)

INFORMATION_SCHEMA.COLUMNS		
Esta vista del sistema nos proporciona el acceso a la información de todas las columnas que se encuentras registradas según las tablas y vistas de las diferentes bases de datos que existen.		
Columna	**Tipo de Dato**	**Descripción**
TABLE_CATALOG	nvarchar(128)	Nombre del proyecto o base de datos
TABLE_SCHEMA	nvarchar(128)	Nombre del esquema
TABLE_NAME	nvarchar(128)	Nombre de la tabla
COLUMN_NAME	nvarchar(128)	Nombre de columna
ORDINAL_POSITION	int	Identificación de la columna

Fuente: Elaboración propia

Tabla 17: Diccionario de datos de INFORMATION_SCHEMA.COLUMNS

COLUMN_DEFAULT	nvarchar(4000)	Valor por defecto de la columna
IS_NULLABLE	varchar(3)	Si valor es nulo, “YES”, sino NO”
DATA_TYPE	nvarchar(128)	Tipo de datos suministrado por el sistema
CHARACTER_MAXIMUM_LENGTH	int	Máxima longitud en caracteres. -1 para XML y tipos de datos largos.
CHARACTER_OCTET_LENGTH	int	Máxima longitud en bytes. -1 para XML y tipos de datos largos.
NUMERIC_PRECISION	tinyint	Datos numéricos o monetarios exactos
NUMERIC_SCALE	int	Escala de dato numérico o monetario
DATETIME_PRECISION	smallint	Código de subtipo para fecha y hora
CHARACTER_SET_CATALOG	nvarchar(128)	Base de datos donde el carácter se encuentra localizado
CHARACTER_SET_SCHEMA	nvarchar(128)	Siempre retorna nulo
CHARACTER_SET_NAME	nvarchar(128)	Devuelve el nombre único para el conjunto de caracteres si es datos de caracteres o texto, sino devuelve nulo
COLLATION_CATALOG	nvarchar(128)	Siempre retorna nulo
COLLATION_SCHEMA	nvarchar(128)	Siempre retorna nulo
COLLATION_NAME	nvarchar(128)	Si el carácter es de tipo texto, devuelve el nombre único, sino devuelve nulo
DOMAIN_CATALOG	nvarchar(128)	Nombre de la base de datos en la que se creó el tipo de datos definido
DOMAIN_SCHEMA	nvarchar(128)	Nombre del esquema en la que se creó el tipo de datos definido
DOMAIN_NAME	nvarchar(128)	Nombre del tipo de datos en la que se creó el tipo de datos definido

Fuente: Elaboración propia

En la tabla 17 se muestra el diccionario de datos del archivo principal de configuración inicial del proyecto.

Tabla 18: Diccionario de datos de archivo de configuración

FileConfig.atcd		
Este archivo de configuración contiene la información de base sobre el proyecto, capas y tablas de base de datos con el fin de generar una solución.		
Nodo	**Nodo Hijo**	**Descripción**
<Solution>	<Name>	Nombre de la solución
	<VisualStudioVersion>	Versión del IDE Visual Studio .Net
	<SqlServerVersion>	Versión de BD SQL Server
	<Path>	Ruta
	<SolutionPath>	Ruta de la solución
<Project>	<Name>	Nombre del Proyecto
	<ProjectPreffix>	Prefijo asignado para el nombre del proyecto
	<DataAcces>	Nombre y prefijo de la capa de acceso a datos
	<DataEntity>	Nombre y prefijo de la capa de entidades
	<Controller>	Nombre y prefijo de la capa de controlador
	<View>	Tipo de proyecto (web, desktop, console)
	<SelectedTables>	Tablas de base de datos seleccionadas para mapear al proyecto
	<CustomQuerys>	Reemplazo de campos de base de datos
<Connection>	<Server>	Servidor principal de base de datos
	<DatabaseName>	Base de datos que se realizará el mapeo
	<Autentication>	Datos de autenticación para dicha base de datos
	<Programmability>	Datos de usuario con altos privilegios para base de datos

Fuente: Elaboración propia

4.4 Arquitectura

4.4.1 Representación de la arquitectura

Para la realización de esta herramienta ORM se maneja el patrón de diseño Unit Of Work. El objetivo principal de este patrón es el de, aquellos objetos que han sido registrados, modificados y/o eliminados con respecto a una base de datos, tratarlos como si fuera una sola unidad, es decir, para un mismo objeto se pueden realizar las acciones de inserción y actualización, y cuando se desea guardar dicha información en la base de datos, dichas acciones se ejecutan como si fueran una sola, debido a que las acciones ya quedaron en memoria.

En la figura N° 66 se muestra la representación de la arquitectura que se utiliza en el presente documento.

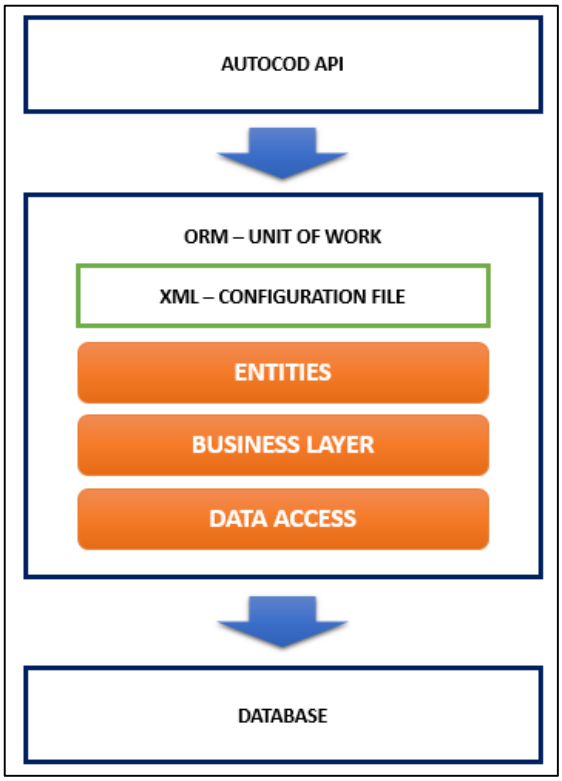


Figura 66: Representación de la arquitectura del software
Fuente: Elaboración propia

4.4.2 Vista de casos de uso

4.4.2.1 Diagrama de casos de uso más significativos

En la figura N° 67 se muestra el diagrama de casos de uso más significativos.

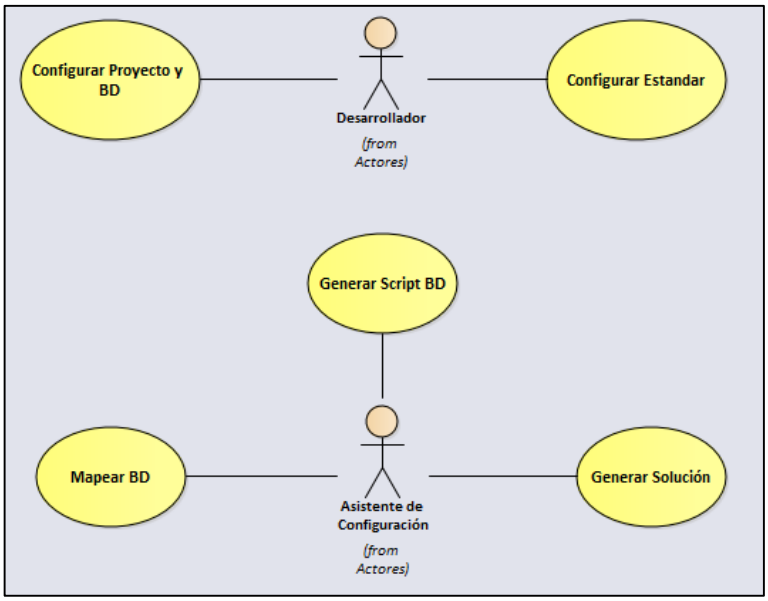


Figura 67: Diagrama de casos de uso más significativos
Fuente: Elaboración propia

4.4.2.2 Lista priorizada de casos de uso más significativos

1. Configurar proyecto y BD
2. Configurar estándar
3. Mapear BD
4. Generar Script BD
5. Generar Solución

4.4.3 Vista lógica

4.4.3.1 Diagrama de paquetes

En la figura N° 68 se muestra el diagrama de paquetes

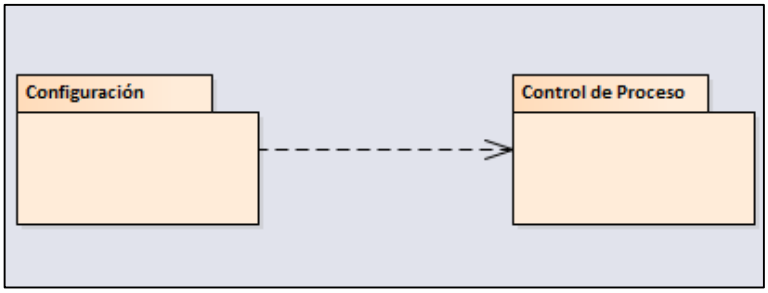


Figura 68: Diagrama de paquetes
Fuente: Elaboración propia

4.4.3.2 Clases de diseño más representativas del sistema

En la figura N° 69 se muestra las clases de diseño más representativas del sistema.

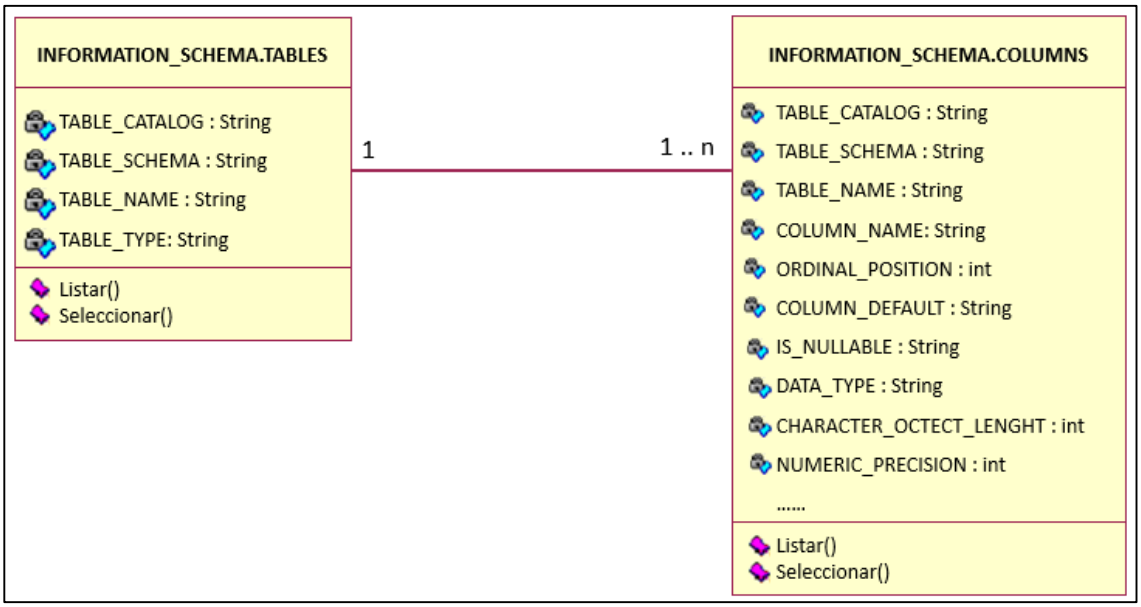


Figura 69: Clases de diseño más representativas del sistema
Fuente: Elaboración propia

4.4.4 Vista de implementación

4.4.4.1 Diagrama de componentes del sistema

En la figura N° 70 se muestra el diagrama de componentes del sistema.

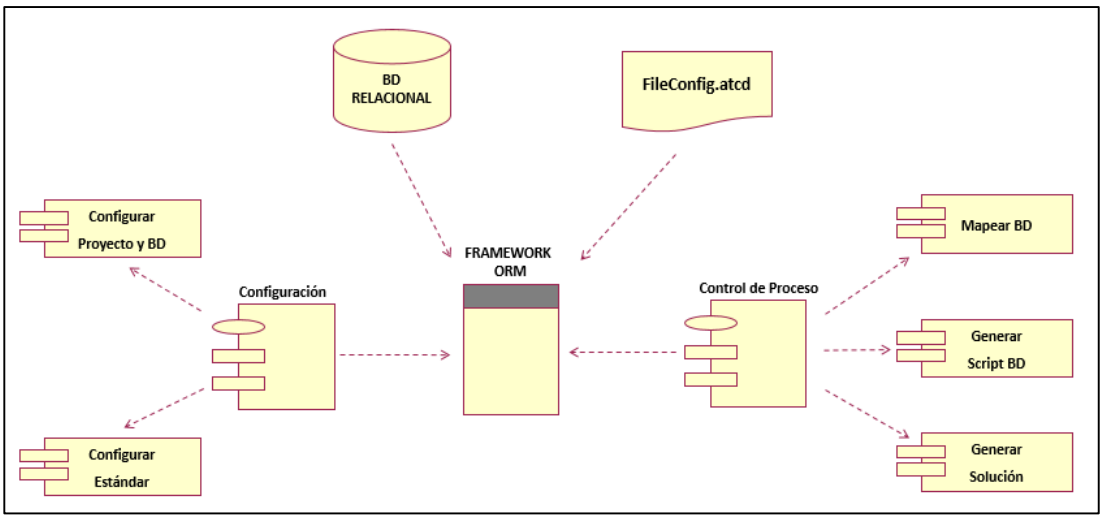


Figura 70: Diagrama de componentes del sistema
Fuente: Elaboración propia

4.4.5 Vista de despliegue

4.4.5.1 Diagrama de despliegue

En la figura N° 71 se muestra la vista de despliegue.

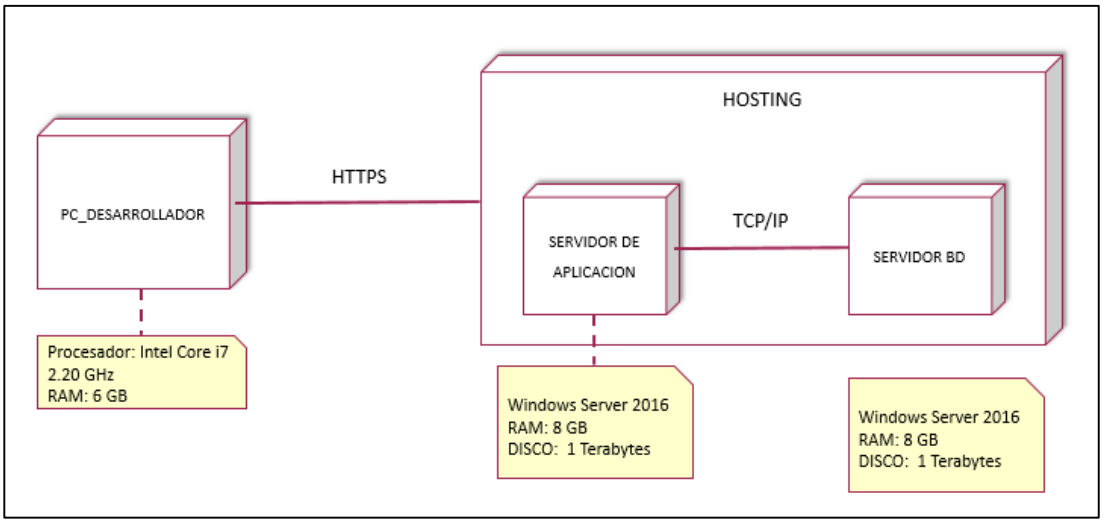


Figura 71: Diagrama de despliegue
Fuente: Elaboración propia

4.4.6 Vista de datos

4.4.6.1 Modelo físico de datos

En la figura N° 72 se muestra el modelo físico de datos.

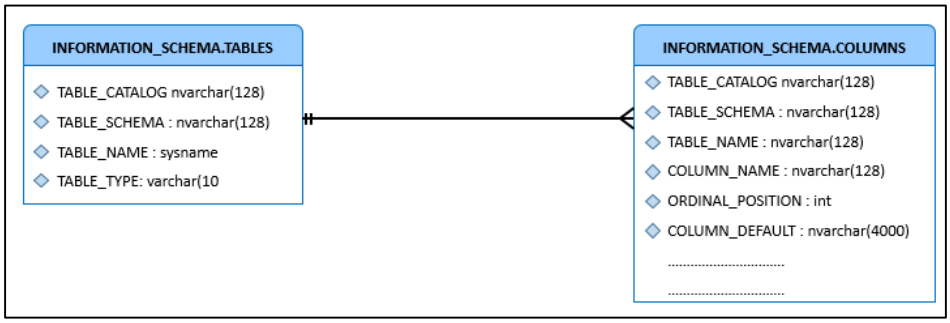


Figura 72: Modelo físico de datos
Fuente: Elaboración propia

4.5 Pruebas

4.5.1 Plan de pruebas

En la tabla 18, se detalla el plan de prueba que realizaremos para las funcionalidades descritas en los casos de uso.

Tabla 19: Plan de pruebas

CRUD APPLICATION FRAMEWORK – PLANES DE PRUEBA						
Fase	**Nº**	**Unidad de Prueba**	**Tipo**	**Descripción**	**Fecha planificada**	**Tester**
ELB	1	Prueba de Programación de CUS Configurar Proyecto y BD	DEF	En este caso de uso lo realiza el desarrollador configura un nuevo proyecto	22/09/2019	Velásquez
ELB	2	Prueba de Programación de CUS Mapear BD	DEF	En este caso de uso lo realiza el framework, a partir de las tablas de la base de datos crea las clases del proyecto.	15/09/2019	Velásquez
ELB	3	Prueba de Programación de CUS Generar Scripts de base de datos	DEF	En este caso de uso lo realiza el framework, crea los scripts de SQL que contiene las instrucciones de acceso y alteración de datos de la BD	10/09/2019	Velásquez
ELB	4	Prueba de Programación de CUS Generar Solución	DEF	En este caso de uso lo realiza el framework crea los archivos de proyecto de las capas, la solución y las referencias a las clases y otros proyectos.	20/09/2019	Velásquez

Fuente: Elaboración propia

4.5.2 Informe de pruebas

En las tablas 19, 20, 21 y 22, se muestra los planes de prueba realizados al framework con los resultados esperados.

Tabla 20: Informe de pruebas CUS Configurar Proyecto y BD

INFORME DE PRUEBA					
Unidad de Prueba:	CUS Configurar Proyecto y BD				
	Escenarios probados:				
Fecha:	22/09/2019	**Avance %**	100%		
Tester:	Christian Velasquez				
Objetivo de la Prueba:					
Verificar que el desarrollador pueda configurar nuevo proyecto.					
N°	**Tipo**	**Descripción**	**Resultado Esperado**	**Resultado**	**Detalle de los resultados**
1	Prueba Unitaria	Debe existir un formulario para configurar los parámetros del framework	Se espera que se realice con éxito el resultado.	Aprobado	Al abrir el sistema aparece el formulario de configuración
2	Punto de chequeo	Se comprobará que al llenar los datos en el asistente de configuración	Se realiza la configuración con éxito y se guarda estos datos en un archivo XML	Aprobado	Al dar presionar el botón “siguiente” se visualizará paso por paso todos los campos de configuración y al finalizar guarda el archivo XML.

Fuente: Elaboración propia

Tabla 21: Informe de pruebas CUS Mapear BD

INFORME DE PRUEBA					
Unidad de Prueba:	CUS Mapear BD				
	Escenarios probados:				
Fecha:	22/09/2019	**Avance %**	100%		
Tester:	Christian Velasquez				
Objetivo de la Prueba:					
Verificar que se cree las clases de acuerdo a las tablas seleccionadas					
Número	**Tipo**	**Descripción**	**Resultado Esperado**	**Resultado**	**Detalle de los resultados**
1	Prueba Unitaria	El framework lee la información completa de las tablas seleccionadas y crea las clases de C# a partir de esas tablas.	Que el framework cree las clases respetando los tipos de datos que contienen las tablas y en la ubicación especificada.	Aprobado	Al procesar el proyecto el sistema lee la información de las columnas de las tablas SQL y crea las clases entidades a partir de ellas
2	Punto de chequeo	Se comprueba que se culmina la generación de las clases.	Las clases se encuentran generadas dentro de los proyectos y en las carpetas especificadas	Aprobado	Al culminar el proceso se abre el directorio donde se encuentra las clases generadas

Fuente: Elaboración propia

Tabla 22: Informe de pruebas CUS Generar Scripts BD

INFORME DE PRUEBA					
Unidad de Prueba:		CUS Generar Script BD			
		Escenarios probados:			
Fecha:	22/09/2019	**Avance %**	100%		
Tester:	Christian Velasquez				
Objetivo de la Prueba:					
Validar la generación de los scripts que interactúan con la base de datos.					
Número	**Tipo**	**Descripción**	**Resultado Esperado**	**Resultado**	**Detalle de los resultados**
1	Prueba Unitaria	El framework crea los scripts de la base de datos.	Que el framework cree los scripts de acceso y alteración de datos en la ubicación especificada.	Aprobado	Al procesar se agrega los procedimientos almacenados en la carpeta SQL Scripts.
2	Punto de chequeo	Se comprueba que los scripts creados funcionen correctamente	Que la funcionalidad del producto final sea sin errores de sintaxis SQL y los scripts se ejecuten con normalidad	Aprobado	Al ejecutar el formulario de ejemplo se puede comprobar que los scripts de listar funcionan perfectamente

Fuente: Elaboración propia

Tabla 23: Informe de pruebas CUS Generar Solución

INFORME DE PRUEBA					
Unidad de Prueba:	CUS Generar Solución				
	Escenarios probados:				
Fecha:	20/09/2019	**Avance %**	100%		
Tester:	Christian Velasquez				
Objetivo de la Prueba:					
Comprobar que se genere la estructura de la solución los archivos de proyecto y el archivo solución completamente relacionado.					
Número	**Tipo**	**Descripción**	**Resultado Esperado**	**Resultado**	**Detalle de los resultados**
1	Prueba Unitaria	El framework debe de generar el proyecto solución de visual studio	Cree los proyectos y la estructura de la solución con la respectiva referencia a las clases creadas	Aprobado	Al finalizar el proceso, se crea un archivo solución y los archivos de proyectos estos a su vez tienen referencias de las clases
2	Punto de chequeo	Se comprueba que las referencias estén correctamente entrelazadas entre las distintas capas y clases	El archivo solución debe tener referencia de los proyectos que representan las capas, y estos proyectos deben tener referencias a las clases que contienen	Aprobado	Al ejecutar el proyecto solución creado no muestra ningún error y se verifica que la estructura esté correctamente creada.

Fuente: Elaboración propia

CONCLUSIONES

Tomando como unidad de medida principal, el tiempo de desarrollo empleado al interactuar con una entidad de la base de datos, desde la conexión, hasta su utilización en lo métodos de acceso y alteración de datos, medimos la productividad mediante la codificación necesaria para cada entidad, contra los recursos horas/hombre empleados, y comparando este resultado con el de no usar el framework construido, se concluye que:

1. Se mejora la eficiencia en el desarrollo de aplicaciones de software, aplicando patrones MVC y Unit of Works al crear la estructura de programación nueva o modificar un proyecto solución existente.
2. Se mejora la eficiencia en el desarrollo de aplicaciones de software, automatizando la codificación de procedimientos almacenados acceso y alteración de tablas de la base de datos.
3. Se mejora la eficiencia en el desarrollo de aplicaciones de software, automatizando la codificación de clases y métodos empleados para acceder y alterar información en la base de datos relacional desde el programa fuente.
4. Se mejora la eficiencia en el desarrollo de aplicaciones de software, automatizando la propagación de cambios en tablas de base de datos, clases, métodos y procedimientos almacenados.

Finalmente, se concluye que el framework construido cumple el 100 por ciento los objetivos específicos planteados inicialmente.

RECOMENDACIONES

Después de varias pruebas de uso en el desarrollo de aplicaciones de base de datos se recomienda que:

1. El desarrollador debe conocer a un nivel alto el lenguaje de consultas **LINQ** para potenciar el uso del framework debido a que los métodos de selección trabajan con listas de elementos.
2. Debido a la tendencia de mantener una base de datos limpia de se recomienda no abusar de la generación de scripts en la base de datos. Y por el contrario usar la generación de procedimientos almacenados dentro del proyecto que facilita el framework.
3. El diseño de la base de datos debe estar correctamente relacionada entre las tablas que cuentan con una columna en común, para que el framework cree los métodos de fácil acceso a tablas mediante la invocación por llaves foráneas o primarias múltiples.
4. Usar el patrón "unit of work" cuando se tenga que hacer más de una alteración a multiples tablas de la base de datos, usar la clase unit of work creada por el framework le facilita esta tarea, por lo que emplearlo hará que la implementación sea muy sencilla.

REFERENCIAS BIBLIOGRÁFICAS

Alarcon, J. M. (2018). *Qué es un ORM - campusMVP.* Obtenido de campusMVP: https://www.campusmvp.es/recursos/post/que-es-un-orm.aspx

Baetjer, J. (1998). *Software as Capital.*

Boehm, B. W. (1988). *A Spiral Model of Software Development and Enhancement.* TRW Defense Systems Group. Obtenido de https://csse.usc.edu/TECHRPTS/1988/usccse88-500/usccse88-500.pdf

Brookshear, J. G. (2006). *Computer Science An Overview* (9th ed.). Boston - USA: Greg Tobin - Pearson Education. Obtenido de http://infocat.ucpel.tche.br/disc/icc/docs/CSAO.pdf

Buschmann, F. (1996). *Pattern Oriented Software Architecture, Volume 1: A System of Patterns.* Willey & Sons. Obtenido de https://books.google.com.pe/books?id=0kUFZDuqvmEC

Chingo, W. (2016). *Herramienta orm para mejorar la productividad de la empresa Interfaces Software Group. (Tesis de Pregrado) Universidad Regional Autónoma de los Andes.* Obtenido de http://dspace.uniandes.edu.ec/bitstream/123456789/5258/1/PIUASIS005-2016.pdf

Elmasri, R., & Navathe, S. B. (2010). *Fundamentals of Database Systems.* Addison-Wesley. Obtenido de http://iips.icci.edu.iq/images/exam/databases-ramaz.pdf

Freeman, A. (2013). *Pro ASP.NET MVC 5.* Obtenido de http://enos.itcollege.ee/~ijogi/Nooks/Pro%20ASP.NET%20MVC%205/Pro%20ASP.NET%20MVC%205.9781430265290.pdf

Freire, T. (2008). *Sistema de Gestión de Información Odontológica utilizando ORM para el Departamento de Bienestar Universitario de la UTN. (Tesis de Pregrado) Universidad Técnica Del Norte.* Obtenido de http://repositorio.utn.edu.ec/bitstream/123456789/571/1/Tesis.pdf

Jet Brains. (2019, 06 01). *Jet Brains The State of Developer Ecosystem 2019.* Retrieved from Jet Brains: https://www.jetbrains.com/lp/devecosystem-2019/

Kent, W. (1983). *A Simple Guide to Five Normal Forms in Relational Database Theory.* Obtenido de http://www.roma1.infn.it/people/barone/metinf/database-normalization.pdf

Kroll, P., & Kruchten, P. (2003). *The Rational Unified Process Made Easy: A Practitioner's Guide to the RUP.*

Larman, C., & Basili, V. R. (2003). *Iterative and IncrementalDevelopment:A Brief History.* IEEE. Obtenido de

http://www.craiglarman.com/wiki/downloads/misc/history-of-iterative-larman-and-basili-ieee-computer.pdf

Orság, J. (2006). *OBJECT-RELATIONAL MAPPING.* Bratislava: Comenius University. Obtenido de http://www.dcs.fmph.uniba.sk/diplomovky/obhajene/getfile.php/dp.orsag.orm.pdf?id=86&fid=147&type=application%2Fpdf

Royce, W. W. (1970). *Managing The Development Of Large Software Systems.* Obtenido de http://www-scf.usc.edu/~csci201/lectures/Lecture11/royce1970.pdf

S. Pressman, R. (2010). *Ingeniería del software: Un enfoque práctico, 7ma Edición.* Obtenido de http://artemisa.unicauca.edu.co/~cardila/Libro_Pressman_7.pdf

Schwaber, K. (2004). *Agile Project Management with Scrum.* Microsoft. doi:978-0-7356-1993-7

Stack Overflow. (2019, 04 11). *Stack Overflow Developer Survey 2019.* Retrieved from Stack Overflow is the largest, most trusted online community for developers to learn, share their programming knowledge, and build their careers.: https://insights.stackoverflow.com/survey/2019

Tixi, I. (2016). *Análisis de la tecnología ORM-Hibernate con respecto a la productividad, aplicado al sistema de gestión de documentación del departamento de procuraduría de la UNACH. (Tesis de Pregrado) Universidad Nacional de Chimborazo.* Obtenido de http://dspace.unach.edu.ec/bitstream/51000/1628/1/UNACH-EC-ISC-2016-0009.pdf

Yanes, O., & Gracia del Busto, H. (2011). *Mapeo Objeto / Relacional (ORM).* Obtenido de http://revistatelematica.cujae.edu.cu/index.php/tele/article/view/23/21

ANEXOS

1. Manual de Instalación

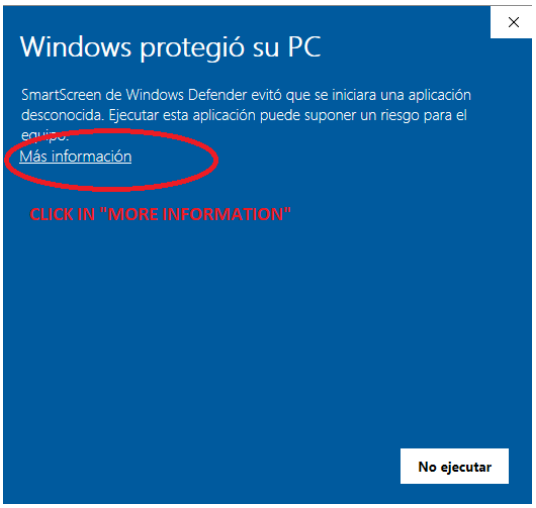
1.1 Descarga e Instalación

Este proceso consta de descargar el instalador del framework para Windows, para esto se encuentra publicado en un dominio en internet para facilitar el acceso, seguir los siguientes pasos:

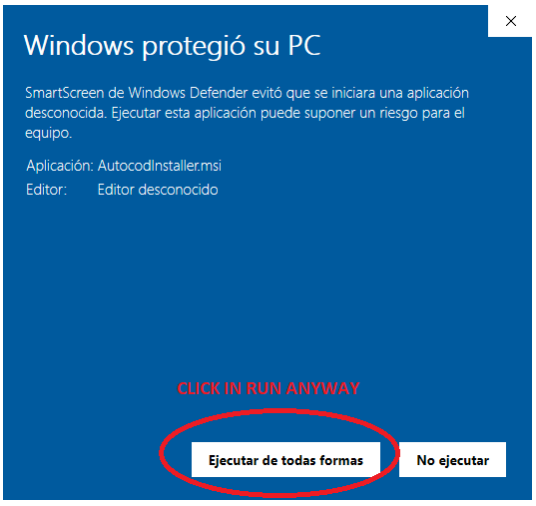
a) Dirigirse a https://autocod.net/download.

b) Descargar el paquete instalador de Windows y ejecutar el archivo AutocodInstaller.msi

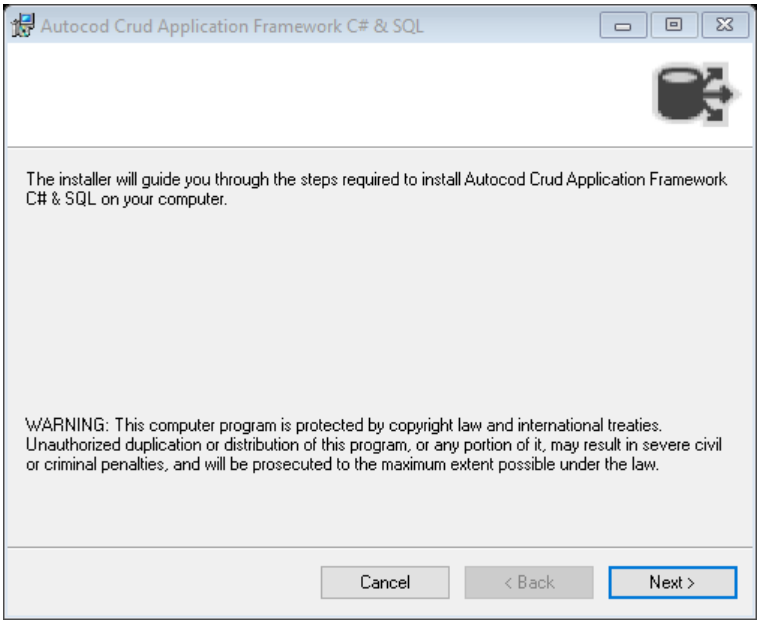
c) Debido a que el software aún no cuenta con la firma digital, Windows defender muestra una advertencia para la ejecución.



d) Dar clic en “más información” y luego en ejecutar de todas formas.

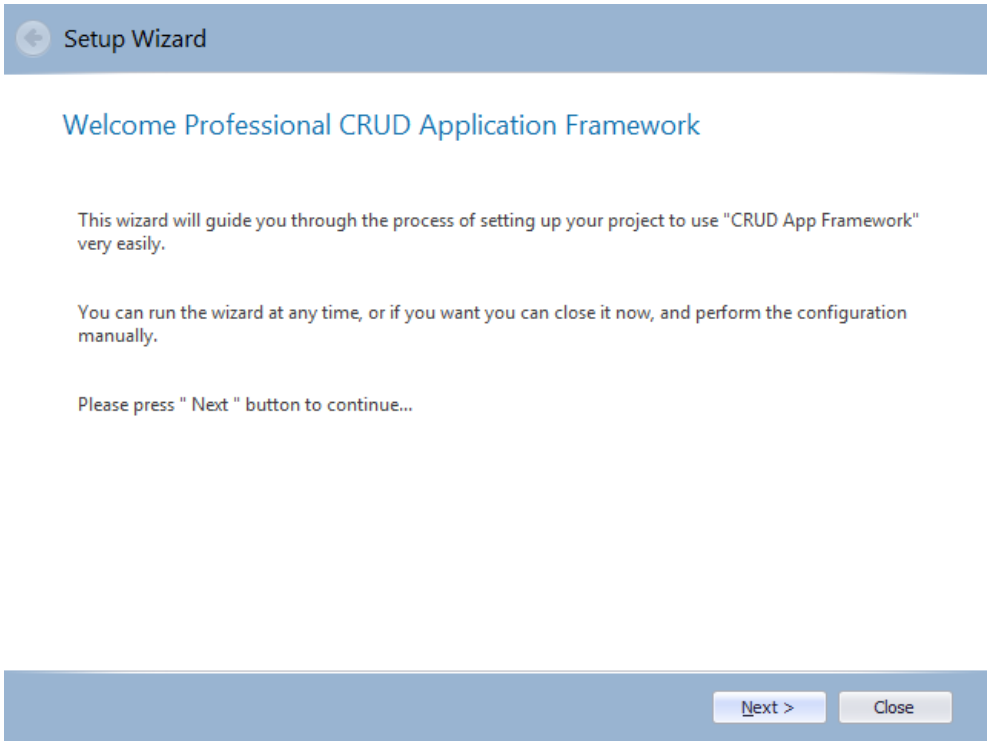


e) Por ultimo seguir los pasos del instalador.

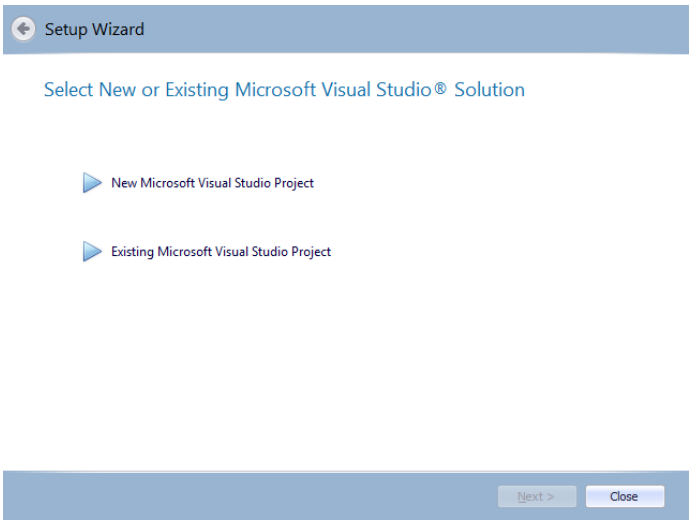


2. Manual de Inicio rápido

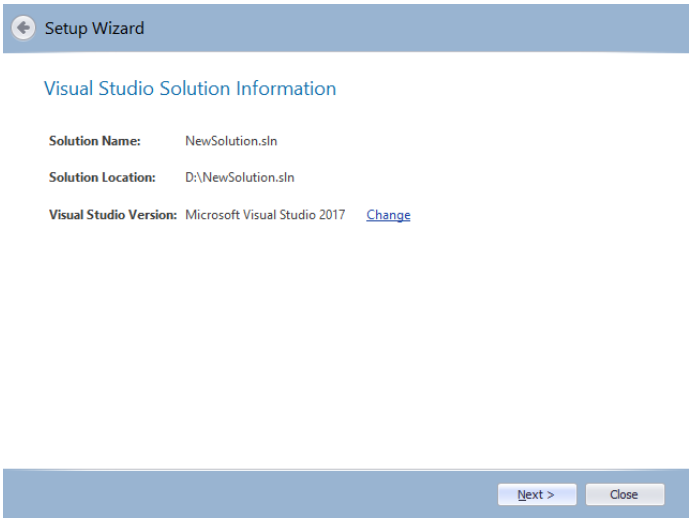
a) Una vez instalado abrir el programa, inmediatamente se abrirá un asistente de configuración que guía paso a paso a realizar la configuración de su proyecto solución.



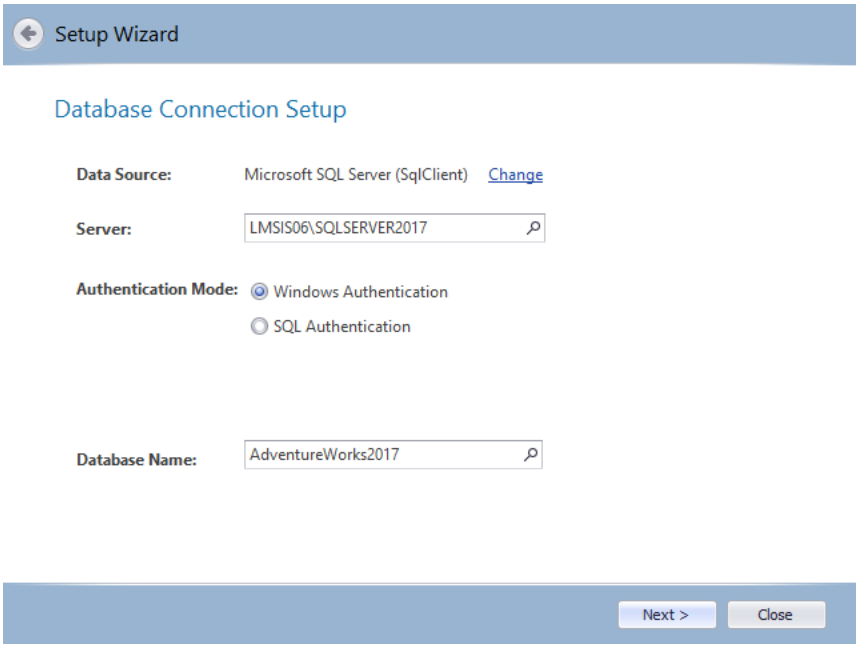
b) Se debe elegir entre crear una nueva solución de visual studio o partir desde una solución ya existente, en caso seleccione un proyecto existente, el sistema no elimina las clases actuales, lee el archivo solución y añadirá los nuevos proyectos sin afectar el funcionamiento ni los proyectos que contiene.



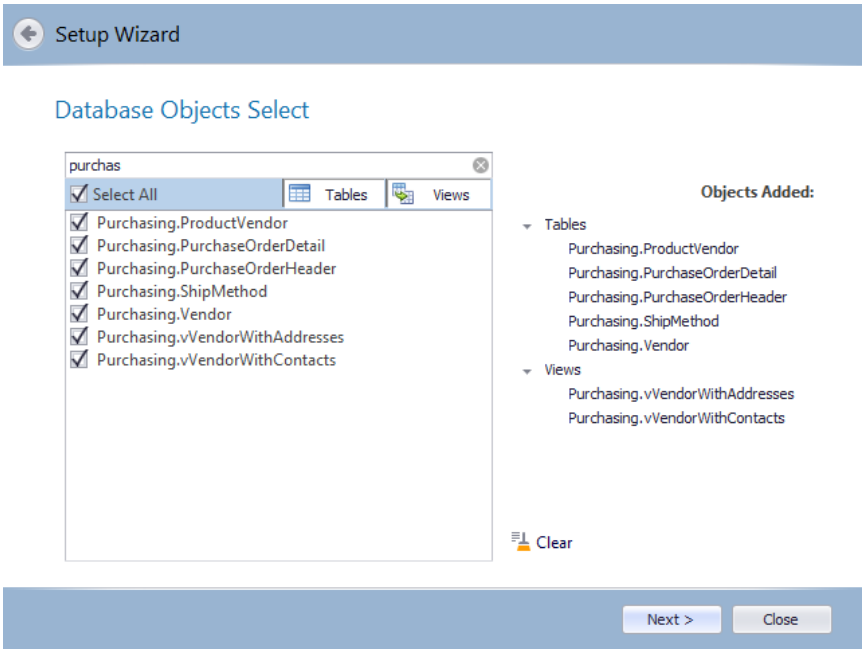
c) De acuerdo a la elección del usuario el sistema muestra la información de ubicación.



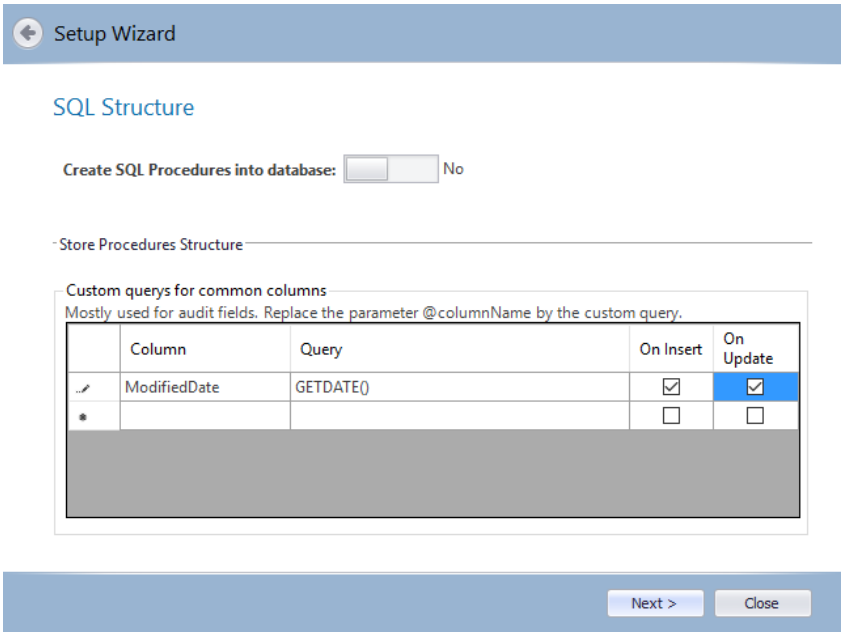
d) A continuación, procedemos a configurar la información de la conexión de la base de datos.



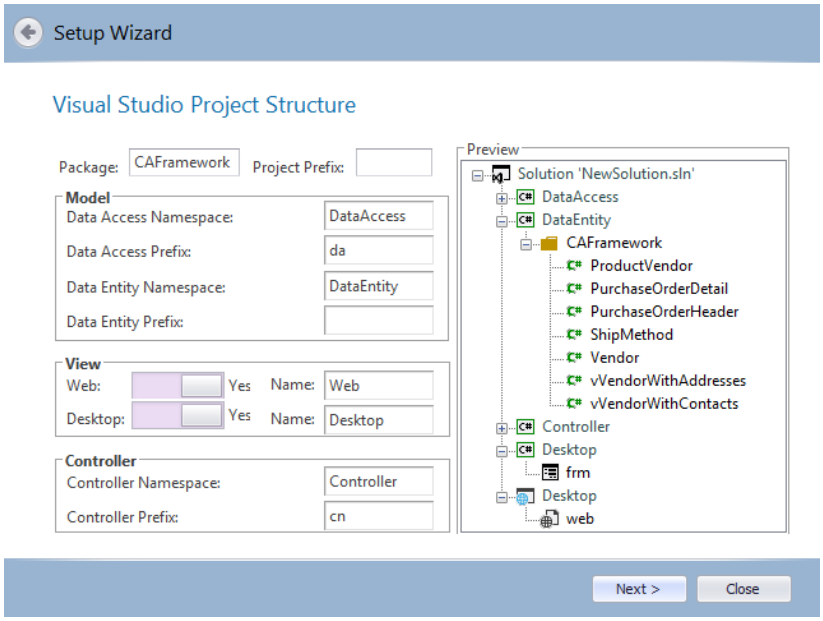
e) Seleccionar las tablas y las vistas que se van a mapear.



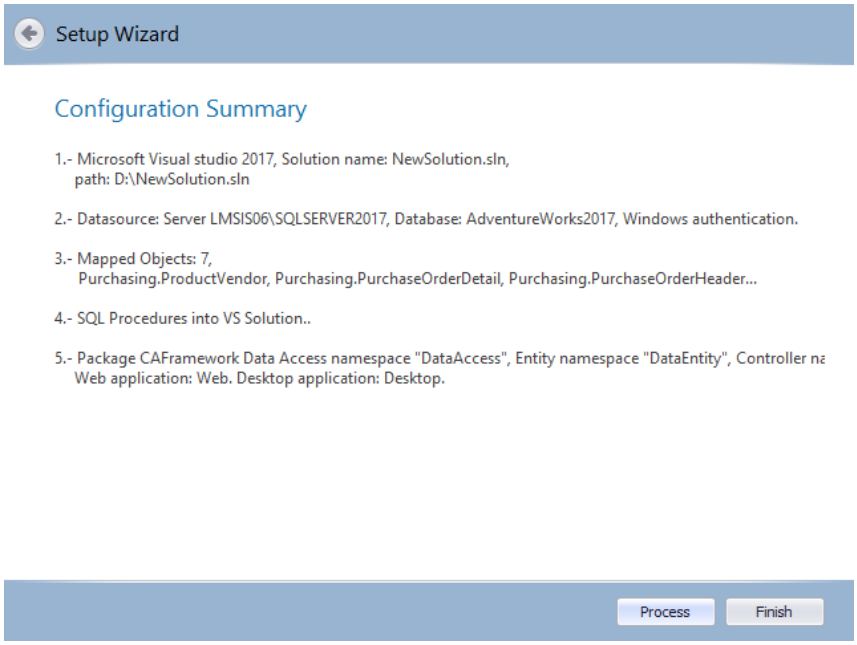
f) Elegir si los procedimientos almacenados serán creados en la base de datos o internamente en el código dentro de la capa de acceso a datos. Si dentro de las tablas existe una columna en común, el framework le permite ingresar una consulta SQL que se usará como prioridad para los métodos de inserción y/o actualización.



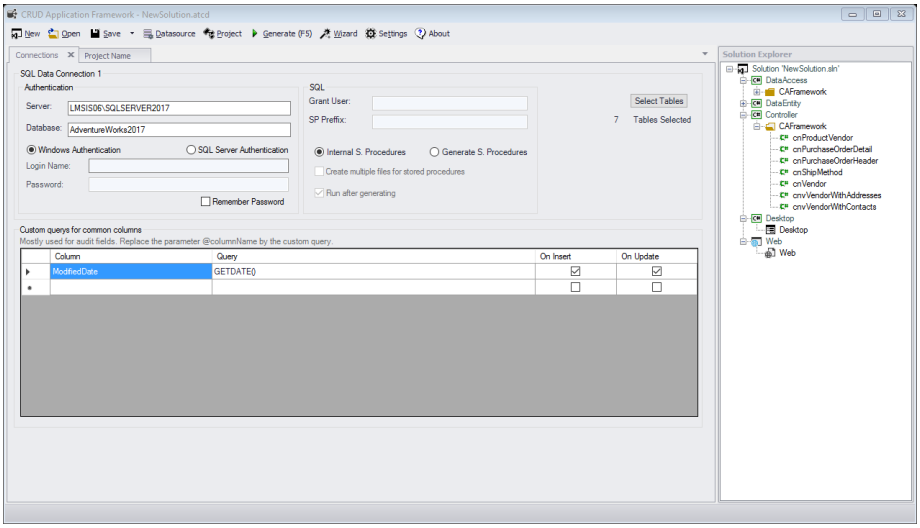
g) En el siguiente paso se detalla la estructura de la solución, se define el nombre de las capas correspondientes a la modelo, vista y controladora, y también se permite añadir el nombre del prefijo y el paquete del proyecto.



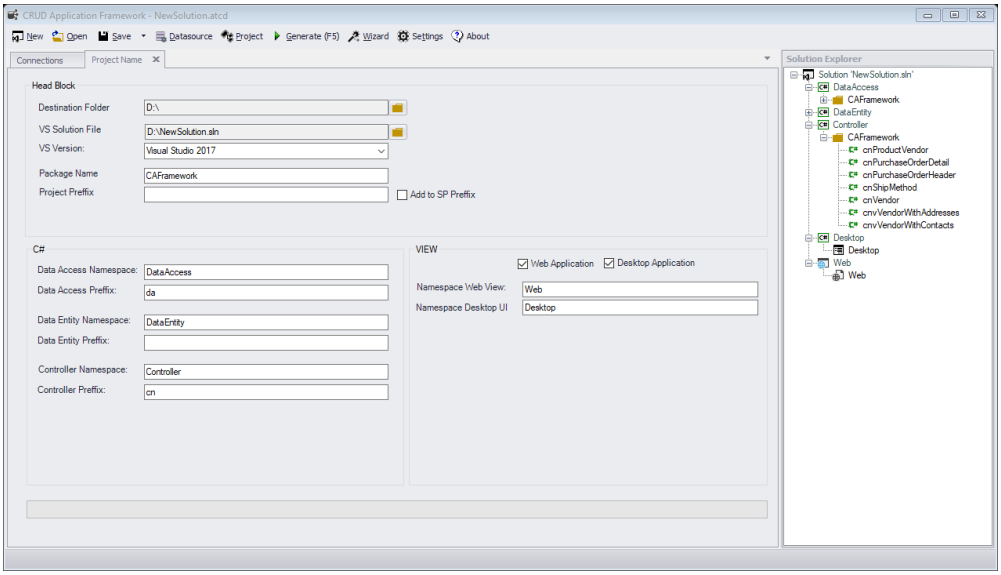
h) El último paso muestra un resumen con la configuración ingresada en el asistente y muestra dos opciones, “Procesar” y “Finalizar”, en el primer caso procesar cierra el asistente e inicia con el proceso de creación, y en el segundo caso cierra el asistente y abre el formulario IDE del framework con toda la configuración realizada.



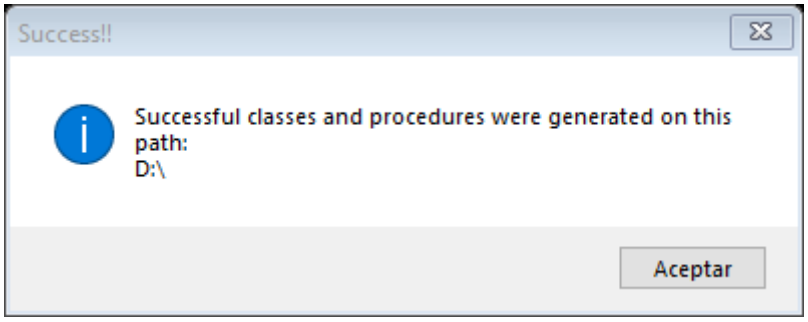
i) Al finalizar se crea un archivo XML en la misma carpeta de solución con la extensión “atcd”, donde está guardado toda la información de la onfiguración realizada. Este tipo de archivo se puede abrir en cualquier momento con el IDE del sistema, o haciendo doble clic en el archivo.

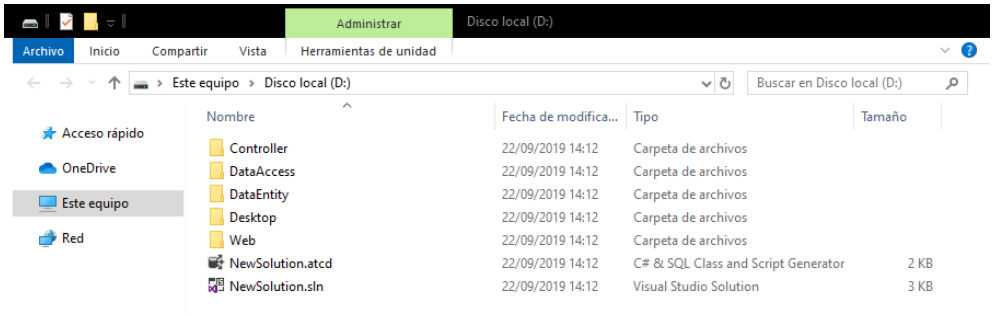


j) Dentro del IDE del sistema se puede modificar directamente la configuración guardada e iniciar el proceso mediante la tecla “F5”

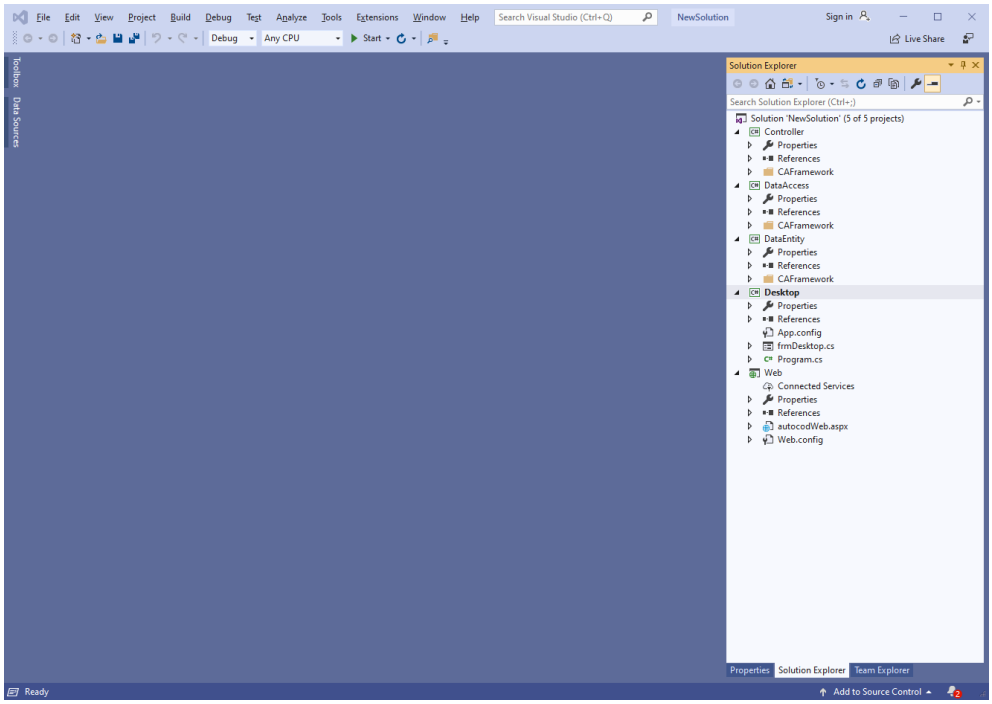


k) Después de iniciar el proceso mediante el IDE o el asistente, se crean todas las clases y archivos que corresponden a la solución.

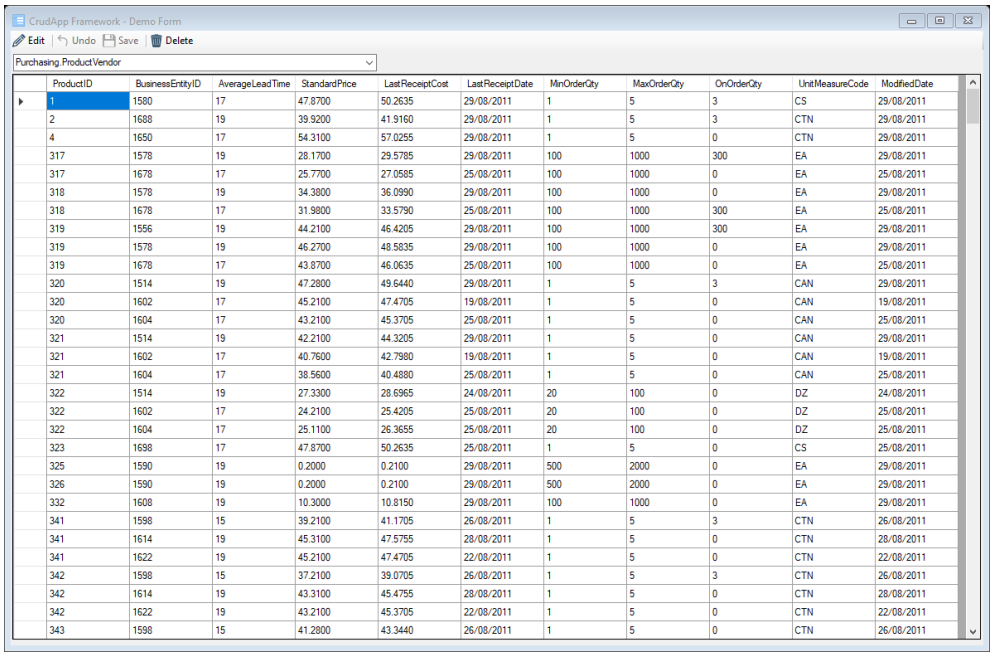




l) Al abrir el archivo solución mostrará los proyectos creados, si es una solución existente los proyectos serán añadidos a los actuales.



m) El sistema crea un formulario de ejemplo en desktop y web, si el desarrollador configuró que se cree la capa de vista también. La finalidad es demostrar el uso.



ProductID	BusinessEntityID	AverageLeadTime	StandardPrice	LastReceiptCost	LastReceiptDate	MinOrderQty	MaxOrderQty	OnOrderQty	UnitMeasureCode	ModifiedDate
1	1580	17	47.8700	50.2635	29/08/2011	1	5	3	CS	29/08/2011
2	1688	19	39.9200	41.9160	29/08/2011	1	5	3	CTN	29/08/2011
4	1650	17	54.3100	57.0255	29/08/2011	1	5	0	CTN	29/08/2011
317	1578	19	28.1700	29.5785	29/08/2011	100	1000	300	EA	29/08/2011
317	1678	17	25.7700	27.0585	25/08/2011	100	1000	0	EA	25/08/2011
318	1578	19	34.3800	36.0990	29/08/2011	100	1000	0	EA	29/08/2011
318	1678	17	31.9800	33.5790	25/08/2011	100	1000	300	EA	25/08/2011
319	1556	19	44.2100	46.4205	29/08/2011	100	1000	300	EA	29/08/2011
319	1578	19	46.2700	48.5835	29/08/2011	100	1000	0	EA	29/08/2011
319	1678	17	43.8700	46.0635	25/08/2011	100	1000	0	EA	25/08/2011
320	1514	19	47.2800	49.6440	29/08/2011	1	5	3	CAN	29/08/2011
320	1602	17	45.2100	47.4705	19/08/2011	1	5	0	CAN	19/08/2011
320	1604	17	43.2100	45.3705	25/08/2011	1	5	0	CAN	25/08/2011
321	1514	19	42.2100	44.3205	29/08/2011	1	5	0	CAN	29/08/2011
321	1602	17	40.7600	42.7980	19/08/2011	1	5	0	CAN	19/08/2011
321	1604	17	38.5600	40.4880	25/08/2011	1	5	0	CAN	25/08/2011
322	1514	19	27.3300	28.6965	24/08/2011	20	100	0	DZ	24/08/2011
322	1602	17	24.2100	25.4205	25/08/2011	20	100	0	DZ	25/08/2011
322	1604	17	25.1100	26.3655	25/08/2011	20	100	0	DZ	25/08/2011
323	1698	17	47.8700	50.2635	25/08/2011	1	5	0	CS	25/08/2011
325	1590	19	0.2000	0.2100	29/08/2011	500	2000	0	EA	29/08/2011
326	1590	19	0.2000	0.2100	29/08/2011	500	2000	0	EA	29/08/2011
332	1608	19	10.3000	10.8150	29/08/2011	100	1000	0	EA	29/08/2011
341	1598	15	39.2100	41.1705	26/08/2011	1	5	3	CTN	26/08/2011
341	1614	19	45.3100	47.5755	28/08/2011	1	5	0	CTN	28/08/2011
341	1622	19	45.2100	47.4705	22/08/2011	1	5	0	CTN	22/08/2011
342	1598	15	37.2100	39.0705	26/08/2011	1	5	3	CTN	26/08/2011
342	1614	19	43.3100	45.4755	28/08/2011	1	5	0	CTN	28/08/2011
342	1622	19	43.2100	45.3705	22/08/2011	1	5	0	CTN	22/08/2011
343	1598	15	41.2800	43.3440	26/08/2011	1	5	0	CTN	26/08/2011

n) Para codificar, se debe contar con las referencias de los proyectos en los formularios donde va usar los métodos creados por el framework