Universidad de las Ciencias Informáticas

Grupo de Bioinformática

Título: Herramienta para la generación de código de aplicaciones WEB

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autores:

David Rodríguez Luque

Daniel Rodríguez Luque

Tutor:

Lic. Milena Ossorio Lamí

JULIO 2007

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

Autores:

David Rodríguez Luque

Daniel Rodríguez Luque

_____

_____

Tutora:

Lic. Milena Ossorio Lamí

_____

OPINIÓN DEL TUTOR DEL TRABAJO DE DIPLOMA

El tutor del presente Trabajo de Diploma considera que los estudiantes han encontrado soluciones originales en la implementación de la herramienta generadora de código. Para ello han tenido que dominar diversas tecnologías en las que se destacan PHP, Java, y el trabajo con el estándar XML. Además, han impartido cursos a otros estudiantes sobre varias de estas tecnologías lo que refleja el alto nivel de preparación y responsabilidad.

Ambos estudiantes han trabajado con independencia y dedicación y el volumen del trabajo justifica haber realizado el proyecto en equipo. Han mostrado poseer cualidades para el trabajo de investigación y para el desarrollo de software, características imprescindibles en la profesión.

Uno de los principales objetivos de la Universidad de las Ciencias Informáticas es la producción de software, en mucho de los casos, correspondientes con aplicaciones Web. El uso de herramientas cuyo objetivo sea la generación automática de módulos que en definitiva se repiten en la mayoría de las aplicaciones Web, redundaría en grandes beneficios en el ámbito económico para nuestra institución. Podemos mencionar aspectos como la disminución del tiempo de desarrollo con todos los beneficios que de esto se derivan, además contribuiría a que el software sea menos proclive a errores, con ello aumentaría la robustez del mismo y finalmente se traduciría en una mayor credibilidad por los clientes. Herramientas como estas son de probada utilidad en el mundo informático.

Por todo lo anteriormente expresado considero que los estudiantes están aptos para ejercer como Ingeniero Informático; y propongo que se le otorgue al Trabajo de Diploma la calificación de **5 puntos**.

Lic. Milena Ossorio Lamí

_____ _____

Firma Fecha

Agradecimientos:

A Flor María Luque Nuñez, Gustavo Lázaro Rodriguez Álvarez, Gustavo Rodríguez Luque y Saira Pons Perez por todo el apoyo que nos han brindado.

A nuestra tutora Milena Ossorio Lamí.

A los integrantes del grupo de Bioinformática por haber contribuido decisivamente en nuestra formación como profesionales.

Dedicatoria

A nuestros familiares y amigos.
A todas las personas que de una forma u otra han hecho posible el funcionamiento de la Universidad de la Ciencias Informáticas durante estos años.

RESUMEN

Las herramientas de generación de código están ocupando un papel primordial en la producción de software. Con ellas se disminuye el tiempo en que se desarrolla un producto y las posibilidades de errores en la elaboración del mismo. Además se aumenta la calidad del software creado y se facilita su mantenimiento. El funcionamiento de estas herramientas se basa en la obtención de un modelo y la transformación del mismo en código fuente útil en el desarrollo de una aplicación. El modelo describe las características de las entidades del problema en cuestión y qué tipo de código será generado. Generalmente se utilizan como modelos, un diseño de base de dato, un diagrama de clases o un modelo creado con la misma herramienta. El uso más difundido de los generadores es en la creación de capas como la de acceso a datos o las interfaces. Este trabajo esta orientado a la creación de una herramienta para la generación de la capa de acceso a datos para una aplicación WEB usando como lenguaje de programación PHP.

Palabras Claves:
Generación de código, herramienta, PHP.

INTRODUCCIÓN

Las empresas desarrolladoras de software sea cual sea su tamaño, capital o presencia en el mercado persiguen un objetivo común: “desarrollar software de calidad a un costo y en un tiempo adecuados”. Esto es un reto difícil de lograr si se tiene en cuenta que las demandas de los clientes en muchos casos son superiores a las capacidades productivas de las empresas y que la producción de software de forma industrial es solo un mito que pocas entidades pueden respaldar. La solución a este conflicto se ha buscado en todas las direcciones, se han perfeccionado los procesos y las metodologías de desarrollo, se trabaja fuertemente en capacitación de los recursos humanos y constantemente se desarrollan nuevas tecnologías y frameworks para agilizar el desarrollo.

Un espacio que no se ha quedado atrás es el mejoramiento de las herramientas de desarrollo. El uso de una herramienta adecuada en el proceso de desarrollo de software es considerado en muchos casos un factor clave para el éxito de un proyecto. Por sólo citar un ejemplo del valor que le atribuyen los desarrolladores a las herramientas, el costo de la herramienta CASE Rational Rose 2003 Enterprise Edition es aproximadamente de tres mil dólares. [14]

Una de las funcionalidades más apreciadas en una herramienta es la capacidad de generar código. Esto por lo general se logra partiendo de modelos UML o bien utilizando la estructura de una base de datos. Existe una infinidad de herramientas que permiten la generación de código, sobre todo el código de acceso a datos. Esto se debe a que la construcción manualmente del código de acceso a datos requiere de mucho tiempo y se convierte en una tarea tediosa por lo repetitiva que es. Aunque inicialmente cualquiera de las herramientas de generación de código existentes brinda facilidades a los desarrolladores, vale la pena aclarar que una mala elección puede significar la pérdida de gran cantidad de tiempo y esfuerzo por parte de los desarrolladores y que es difícil encontrar una que se adapte por completo a las necesidades específicas de estos.

En el caso particular de la creación de aplicaciones WEB, la selección de la herramienta de generación de código a utilizar es una tarea difícil. Las tecnologías que se emplean en el desarrollo de la WEB evolucionan muy rápido, impidiendo que las herramientas ganen el grado de madurez ideal. Las herramientas existentes no abarcan todos los aspectos del desarrollo de aplicaciones WEB, unas son favoritas para el diseño de interfaz, mientras que otras se usan sólo en la codificación u otros propósitos específicos.

Sin duda la opción más deseada es la de contar con una herramienta propia que se adecue a las necesidades concretas de un proyecto y que pueda ser modificada para corregir errores o adaptarse a los cambios de las tecnologías. Cabe mencionar que no todos los proyectos de desarrollo de software pueden darse el lujo de dedicarle recursos al desarrollo de una herramienta de generación de código y algunos tienen que conformarse con desarrollar plantillas de código a reutilizar. La construcción de un generador no es un proceso trivial. En el funcionamiento de un generador, aspectos claves como la carga del modelo de datos y los mecanismos de generación requieren de mucho esfuerzo por parte de los programadores y se cae en el riesgo de que una vez terminado no cumpla con las expectativas de los futuros usuarios.

En la Universidad se desarrollan muchas aplicaciones WEB programadas en PHP y no se cuenta con una herramienta de generación de código eficaz, por lo tanto, el desarrollo de éstas sería de gran utilidad. El problema radica en cómo construir un generador de código de acceso a datos para aplicaciones programadas en PHP. No se tiene experiencia en el desarrollo de este tipo de herramientas y las soluciones de terceros que se han utilizado han estado muy por debajo de lo que se esperaba.

Planteando formalmente el problema queda de la siguiente forma: ¿Cómo construir un generador de código que contribuya a la automatización del desarrollo de la capa de acceso a datos en aplicaciones programadas en PHP?

Siendo el objeto de estudio:”El proceso de generación automática de código” y el campo de acción:”Las herramientas de generación de código para aplicaciones programadas en PHP”.

Esta investigación se propone como objetivo desarrollar un generador de código de acceso a datos para aplicaciones programadas en PHP para lo cual se plantean los siguientes objetivos específicos:

- Obtener un levantamiento de los principales requisitos a considerar para una herramienta de generación de código de aplicaciones WEB de acuerdo a las necesidades de la Universidad.
- Diseñar una herramienta de generación de código.
- Implementar la herramienta de generación de código para PHP.

Las tareas específicas para cumplir con los objetivos propuestos son:

- Revisión de la literatura sobre la generación de código.
- Revisión de las herramientas de generación de código para PHP disponibles.
- Levantamiento de los requisitos de la herramienta de generación de código.
- Diseño de la herramienta de generación de código.
- Implementación de la herramienta de generación de código.
- Creación de la documentación del proceso de desarrollo de software.

Como antecedente a esta investigación se tiene la implementación de una herramienta de generación de capa de acceso a datos para PHP la cual se utilizó por algunos desarrolladores y sirvió de motivación a esta investigación. La mencionada herramienta fue implementada en 30 días por 2 programadores encargados de desarrollar la capa de acceso a datos de un proyecto y los resultados obtenidos con su uso evidenciaron cuanto se puede hacer en el campo de la generación automática de código.

Proyectos como el de “Ensayos Clínicos” o los “LIMS” pueden ser los principales beneficiados del resultado de esta investigación. Estos proyectos tienen como finalidad desarrollar sistemas grandes con un alto nivel de flexibilidad, los cuales van a requerir de un gasto considerable de tiempo y esfuerzo en su implementación.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA.

Introducción

En este capítulo se da una explicación de los métodos empleados para dar solución al problema planteado. Se fundamenta el uso de cada una de las tecnologías empleadas y se da una visión general de cuales son las principales herramientas para la generación de código existentes en la actualidad.

Acerca de la generación de código

¿Qué es un generador de código?

Un generador de código es una herramienta capaz de generar código de forma automática. Por lo general generan código en algún lenguaje de tercera generación, listo para compilar o interpretar. Estas herramientas hacen de forma automática el trabajo que a los programadores les tomaría mucho más tiempo lograr de forma manual. Para su trabajo los generadores parten de un modelo que puede ser un diagrama de clases o la estructura de una base de datos. El código generado va a depender del modelo que se le entregue y de los algoritmos y patrones que tenga implementado el generador. La mayoría de estas herramientas poseen plantillas de lenguajes scripts para representar los algoritmos y patrones implementados, las cuales el usuario puede modificar y adaptar según sus necesidades. [1]

Tipos de generadores

Según su interacción con el código ya generado se clasifican en: Activos, Pasivos

Los generadores activos son aquellos que permiten generar varias veces sobre el mismo código generado a partir de cambios en la entrada. Estos generadores definen espacios de código seguros donde el programador puede hacer los cambios que desee sin que éstos se pierdan en las sucesivas generaciones de código.

Los generadores pasivos generan el código una vez y no vuelven a tener interacción con él. Tienen la desventaja de que si se corrige un error en los mecanismos de generación o se cambia el diseño y se vuelve a generar se pierde todo lo que se codificó manualmente. [1]

Según la aproximación que usan para generar el código se clasifican en: Estructurales, de Comportamiento y Traductivos.

Aproximación Estructural:
Genera bloques de código (como interfaces de clases) desde modelos estáticos y relaciones entre objetos. Las primitivas de trabajo en estos modelos son clases, atributos, tipos y asociaciones. Algunas herramientas usan un motor de traducción y plantillas preexistentes (que pueden ser modificadas y adaptadas) para especificar correspondencias con un código fuente en particular. Escritas en un lenguaje de script, las plantillas guían la traducción de los modelos en estructuras de código, como cabeceras de clases o declaraciones de funciones. Los lenguajes de script permiten a los diseñadores seguir estándares de codificación, personalizar la arquitectura del código generado y crear plantillas nuevas para lenguajes no soportados. Los generadores suelen ofrecer opciones para definir qué objetos y lenguajes usar, en el proceso de generación.
La generación de código estructural es incompleta pero ahorra esfuerzo de codificación manual y proporciona un marco de trabajo inicial consistente con los modelos. Las plantillas de traducción aportan un modesto grado de reutilización. La mayoría de las aplicaciones son adecuadas para esta aproximación. [4]

Aproximación de comportamiento:
Generan código completo a partir de modelos de máquinas de estados y la especificación de acciones en un lenguaje de alto nivel. La especificación del comportamiento se basa en máquinas de estados aumentadas con especificación de acciones. Algunos métodos que modelan comportamiento con máquinas de estados, añaden código (como C++ o un lenguaje propietario) para representar las acciones que ocurren durante la transición de estado. Junto con modelos de estructuras de objetos y mecanismos de comunicación, esta técnica permite a las herramientas generar código para el modelo completo de la aplicación. Un beneficio de esta técnica es la capacidad para simular y verificar el comportamiento del sistema basado en modelos antes de que el código sea generado. En los diagramas de estados los usuarios especifican el código explícito para manejar transiciones de eventos. Los lenguajes empleados incluyen C++, C e incluso ensamblador. [1]

En contraste con la aproximación estructural, la codificación del comportamiento se ha hecho durante el modelado de objetos. En la aproximación de comportamiento, la programación se reduce a especificar manejadores de eventos (como las acciones de una máquina de estados) y objetos que son codificados a mano (por ejemplo, optimizados por motivos de rendimiento o eficiencia). Estas herramientas de generación ofrecen poco control al usuario sobre la arquitectura y no usan ingeniería inversa, debido a que todo el código proviene de los modelos. Esta aproximación fomenta un uso continuado de los modelos a lo largo de todo el ciclo de vida del sistema. Los desarrolladores deben adoptar una visión de máquina de estados de la funcionalidad del sistema además de una visión de la estructura de objetos del sistema. Una especificación completa de comportamiento es ejecutable y por tanto permite ser probada y depurada.

La calidad del código generado es buena debido a que los generadores han madurado en respuesta a la retroalimentación de los clientes. Las aplicaciones típicas de este enfoque están en la industria de las telecomunicaciones además de muchos otros tipos de sistemas que pueden ser modelados con máquinas de estados. [1]

Aproximación traductiva:

La aproximaciones traductivas se basan en que los modelos de aplicación y de arquitectura son independientes uno del otro. Un modelo de aplicación completo, con estructura de objetos, comportamiento y comunicaciones es creado usando el método de Análisis Orientado a Objetos.

Un modelo de arquitectura (un conjunto de patrones llamados plantillas o arquetipos) es desarrollado con una herramienta que soporte esta aproximación. Entonces, un motor de traducción genera el código para la aplicación de acuerdo con las reglas de correspondencia en la arquitectura. Las aproximaciones traductivas ofrecen una reutilización significativa debido a que la aplicación y el modelo de arquitectura son independientes.

A la hora de especificar un sistema, el sistema es particionado en dominios.

Un domino es modelado por un modelo de información de objetos, un modelo de estados para cada objeto y especificaciones de acciones para cada estado, con el objetivo de permitir la generación del código. La especificación de acciones se realiza en un lenguaje propietario de cada herramienta y no en un lenguaje de alto nivel. El modelo de aplicación es verificado por un mecanismo de simulación antes de la generación del código.

Un modelo de arquitectura es un conjunto completo de reglas de traducción que establecen una correspondencia de diagramas de objetos con un código fuente. La correspondencia debe ser completa, esto es, todo objeto usado en el modelo de aplicación es traducido. Normalmente las correspondencias gestionan concurrencia (por ejemplo: múltiples hilos de ejecución, multi-tarea, mono-tarea), manejo de eventos (colas, comunicación entre procesos o flujos de entrada/salida) y datos (estructuras, mecanismos de almacenamiento y persistencia). Cualquier lenguaje destino puede ser soportado con la aproximación traductiva.

Construir un modelo de arquitectura es un proceso de desarrollo en sí mismo. Las correspondencias de traducción son escritas en lenguajes de scripts propietarios. Aunque construir una arquitectura requiere un esfuerzo similar a construir un compilador dirigido por tablas, afortunadamente, hay algo de ayuda. Los fabricantes de herramientas proporcionan arquitecturas genéricas que los desarrolladores pueden modificar. Existen arquitecturas desarrolladas por terceros.

Los desarrolladores pueden reutilizar una arquitectura de otros productos en la misma plataforma para amortizar su esfuerzo. Los fabricantes están mejorando las herramientas de construcción de arquitecturas, las cuales pueden incluir librerías de plantillas para ayudar a la composición.

Dado un modelo de aplicación y una arquitectura, el modelo de traducción extrae los objetos identificados del repositorio de modelos, realiza sustituciones y genera código de acuerdo a los scripts de reglas de correspondencias. El desarrollador controla totalmente la generación del código en la aproximación traductiva y el código es potencialmente completo.

Con la aproximación traductiva, el modelo de aplicación se convierte en el principal artefacto de software, desplazando al código fuente. La combinación de arquitecturas y motor de traducción es análoga a un conjunto de compiladores para un lenguaje de alto nivel. [1]

Funcionamiento de un generador de código

Los generadores implementan las siguientes fases en este orden:

Carga:

La fase de carga consiste en la lectura desde un repositorio (puede ser un fichero binario, XML, una base de datos, o un diagrama UML) del modelo a traducir y crear una representación de éste en memoria. La representación de este modelo en memoria, no tiene porqué ser completa (cargar toda la información de modelo), ni tampoco seguir la misma estructura del modelo. Al contrario, la carga y las estructuras pueden ser adaptadas para cargar sólo la información necesaria y disponerla del modo que sea más conveniente para la tarea de traducción a realizar. [4]

Inferencia:

Si se han definido una serie de mecanismos de inferencia, que completan la información de modelado, éstos se ejecutan. Las estructuras del modelo en memoria son completadas y extendidas. Es necesario llevar a cabo este proceso antes de proceder a la generación propiamente dicha. El proceso es responsable de realizar precálculos útiles y de disponer adecuadamente la información para la fase posterior. [4]

Generación:

La última fase es la de generación. En función del código destino a producir, se recorren secuencialmente, y de modo anidado, los elementos de modelo en sucesivas pasadas. Por ejemplo: para cada clase, para cada servicio y para cada argumento. Como resultado de esta fase se obtiene el código generado. [4]

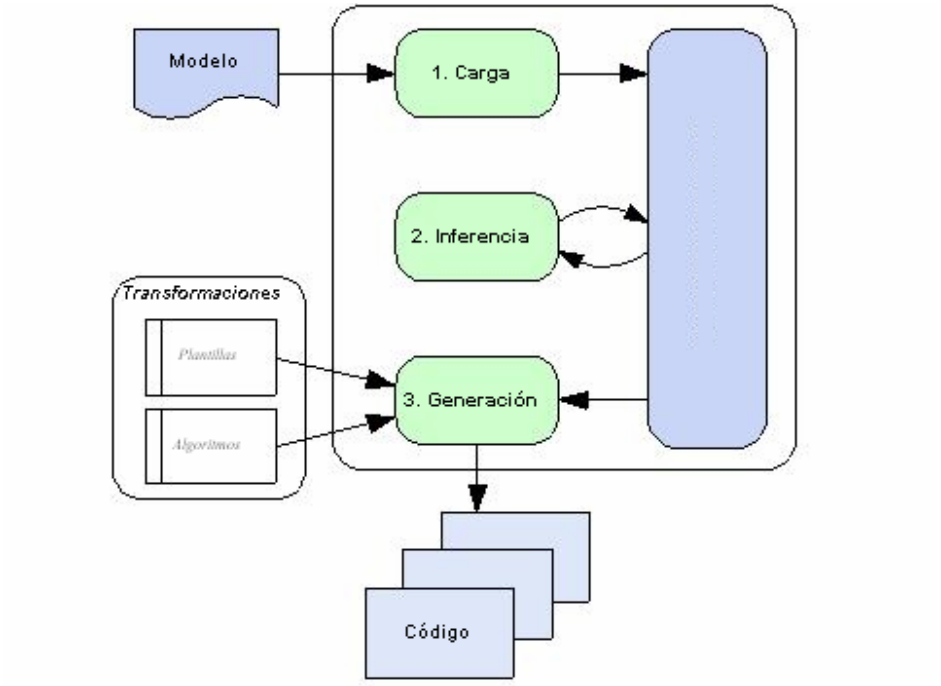

Fig 1. Funcionamiento de un generador.

Técnicas de generación:

Se reconocen cuatro técnicas generales utilizadas por los generadores de códigos:

Las técnicas son:

1. Clonación.
2. Concatenación de cadenas.
3. Plantillas.
4. Análisis de expresiones mediante gramáticas.

Clonación

Si el código destino a producir contiene ficheros que permanecen constantes independientemente del modelo a traducir, la traducción puede llevarse a cabo por medio de clonación, o copia directa del fichero origen al directorio de generación. Las librerías de funciones genéricas o ficheros binarios representando imágenes o iconos constantes pueden ser tratados de este modo: construidos una única vez y añadidos mediante un proceso de copia al producto final. [4]

Concatenación de cadenas

La concatenación de cadenas es un modo sencillo de ir construyendo el código destino desde un lenguaje de programación clásico. El nombre proviene del proceso de ir concatenando cadenas de texto que contienen todo el código a producir. Finalmente, la cadena es volcada a un fichero. Se dispone de toda la potencia del lenguaje de programación para determinar que código debe ser generado, permitiendo cálculos, sustituciones y procesados complejos. Sin embargo, entre las desventajas más sobresalientes, figura la necesidad de proteger los caracteres de control propios del lenguaje empleado para la generación. Por ejemplo: dentro de una cadena de texto en C o C++ no es posible usar el caracter comillas " (indicaría el final de la cadena) y para su empleo es necesario protegerlo con el caracter de escape \". Del mismo modo, deben protegerse otros caracteres como por ejemplo el retorno de carro (\n). El mayor inconveniente de estas protecciones radica en que dificulta la lectura del código objetivo dentro del código del generador.[4]

Plantillas

La generación mediante plantillas puede ser empleada cuando el código destino a producir tiene un patrón de repetición bien caracterizado, de modo que todos los ficheros generados son idénticos salvo los datos procedentes directamente del modelo a generar. En este caso puede definirse una plantilla genérica a partir de ejemplares del código objetivo. En esta plantilla, las dependencias del modelo son sustituidas por marcadores con nombre único. Aparejado a la plantilla, se puede definir un proceso de generación que, dado un elemento del modelo, la plantilla sea instanciada a código mediante un proceso de sustitución de cadenas: los marcadores son sustituidos por los datos del modelo correspondiente. [4]

Análisis de expresiones mediante gramáticas

Existen ocasiones donde las estrategias previas de generación se vuelven insuficientes. En particular, un caso muy claro es el del tratamiento de expresiones que siguen una gramática dada. Aquí las técnicas de compilación clásicas son las más adecuadas para producir el código necesario. La expresión puede ser convertida en una estructura con forma arbórea en memoria: Un analizador léxico, sintáctico y semántico realizan este trabajo. Después, un algoritmo puede recorrer dicho árbol que representa la expresión para analizarla y decidir el tipo de código a producir. El algoritmo incluso puede aplicar optimizaciones si el caso lo permite. [4]

Ventajas y desventajas de la generación de código.

Muchos autores coinciden en que la generación de código ofrece cuatro ventajas principales: Calidad, Consistencia, Productividad y Abstracción. [4][3]

Calidad: La calidad del código generado está en correspondencia con la calidad de las plantillas y del proceso que se usa para la generación. A medida que son detectados errores y es mejorado el código de las plantillas la calidad del código generado aumenta. Se pueden imponer reglas de estilo al código generado para aumentar su homogeneidad y legibilidad. Se puede generar la documentación del código así como comentarios que faciliten su mantenimiento. [4][3]

Consistencia: El código generado es extremadamente consistente. El nombre de las variables, métodos y clases es formado de la misma forma a lo largo de todo el código. La relación entre el modelo y el código generado permite que también haya consistencia entre la documentación y el código, la cual es muy difícil de mantener en un proceso de desarrollo tradicional. [4][3]

Productividad: Es fácil reconocer los beneficios de la generación de código respecto a la productividad. Se comienza con un diseño de entrada e instantáneamente se obtiene una implementación de salida que responde al diseño. Por otra parte estos beneficios son mucho más apreciados cuando regeneras el código debido a un cambio en el diseño y no pierdes todo el trabajo que ya está hecho. Por otra parte libera a los programadores del trabajo tedioso y repetitivo permitiéndoles enfocarse en los aspectos que sí requieren de toda su creatividad e inteligencia. [4][3]

Abstracción: Muchos generadores construyen el código basados en modelos abstractos. Por ejemplo, se puede generar una capa de acceso a datos, a través de un XML que represente las tablas, los atributos y sus relaciones. Entre las ventajas que esto brinda está la portabilidad a distintas plataformas ya que elevando el nivel de abstracción no se cae en especificaciones únicas de una plataforma, sino que permite centrarse en el modelado de los datos o del negocio. Incluso si surgiera una nueva tecnología sólo habría que cambiar el generador y volver a generar la aplicación para que la implemente. [4][3]

Desventajas de la generación de código:

Esfuerzo extra en educación: Se necesita educar a los desarrolladores en las ventajas que ofrece el generador y de qué forma deben emplearlo. [1]

Mantenimiento: cuando se usa un generador hay que darle mantenimiento constantemente. Si es desarrollado por terceros se tiene que estar al tanto de las versiones nuevas que salen y de los errores que se le van detectando. En cualquier caso hay que dedicarle esfuerzos a resolver los errores que pueda traer el generador y a agregarle las nuevas funcionalidades que van surgiendo en el mercado. Si es un generador poco usado se corre además el riesgo de que sus desarrolladores dejen de darle soporte y que por tanto en poco tiempo se vuelva obsoleto. Por otra parte el grado de sofisticación de estas herramientas dificulta mucho su mantenimiento. [3]

Dominios reducidos. La generación de código tradicionalmente se ha aplicado a dominios muy específicos y bien conocidos donde es posible anticiparse a la variabilidad de problemas que pueden encontrarse en ese dominio. Los dominios o áreas de aplicación demasiado grandes o fuera de ámbito no se benefician de la aplicación de técnicas de generación de código. [1]

Resistencia por parte de los desarrolladores al uso de generadores: Hay programadores convencionales que ven la generación de código como una amenaza para su trabajo. Ante lo cual, toman una postura defensiva o de rechazo. Otros afirman que el uso de generadores va contra las buenas prácticas de diseño y critican mucho la idea de copiar y pegar sobre plantillas. [1]

Estudio de las Herramientas de generación de código existente

Existe una variedad de herramientas para la generación de código tanto comercial como libre. A continuación se muestran las de mayores prestaciones y presencia en el mercado.

TierDeveloper:

Es una herramienta propietaria que permite la creación de la capa de acceso a datos para tecnología .NET. Soporta cuatro gestores de base de datos (SQL Server, Oracle, IBM Db2, Microsoft Access). Permite obtener un diagrama de objetos a partir de la estructura de la base de datos. Permite definir un diagrama propio de objeto y cómo se almacenará cada clase en la base de datos. Genera una clase por cada tabla de la base de datos y una clase para manejar toda la capa de acceso a datos. Genera una interfaz WEB para comprobar que funcionan las clases generadas. Entre sus desventajas está su elevado costo y que sólo soporta tecnologías propietarias. [18]

TierDeveloper Ver 5.6	**Precio (USD)**
Licencia por un año de uso para un desarrollador	$795
Licencia por tiempo indefinido para un desarrollador	$1495
Licencia por tiempo indefinido para tres desarrolladores Licencia por tiempo indefinido para cinco desarrolladores Licencia por tiempo indefinido para diez desarrolladores	$3995 $5995 $9995

Tabla 1. Precio de licencias de TierDeveloper.

CodeCharge Studio:

Es una de las herramientas de mayores prestaciones disponibles en el mercado. Está orientada principalmente a la creación de aplicaciones WEB aunque puede ser utilizada para crear aplicaciones de escritorio. Sus principales características son:

1. Soporta varios gestores de base de datos (Microsoft SQL Server, Oracle, DB2, MYSQL y Microsoft Access.).
2. Brinda facilidades para la creación de reportes.
3. Implementa mecanismo de cache para los reportes.
4. Disponible en varios idiomas.
5. Permite la creación de interfaces de usuarios.
6. Soporta hojas de estilo CSS.
7. Soporta integración con SubVersion.
8. Soporta múltiples lenguajes de programación (PHP/JAVA/PERL/ASP).

En cuanto a su interacción con el código se clasifica en Activo y utiliza una aproximación estructural. Su principal desventaja es que no establece una división por capas, en la aplicación que genera por lo que se hace difícil la comprensión del código por parte de los desarrolladores. Es una herramienta propietaria y su costo es elevado. [19]

Licencia	**Características**	**Precio (USD)**
CodeCharge Studio 3.1		
Perpetua	Incluye Soporte y actualizaciones gratis.	$499 por usuario.
No Perpetua	30 días de soporte y actualizaciones gratis.	$279 al año por usuario.
CodeCharge Studio Personal Edition 3.1		
Perpetua	30 días de soporte y actualizaciones gratis.	$199 por usuario.
No Perpetua	30 días de soporte y actualizaciones gratis.	$139 al año por usuario.
Consultas y accesoria técnica		
	Asistencia técnica personalizada	$60 por hora.

Tabla 2. Precio por licencia de CodeCharge.

Visual Paradigm

Es una herramienta Visual para el modelado UML. Está diseñada para una amplia gama de usuarios entre los que se incluyen ingenieros de software, analistas de sistemas, analistas de negocio, arquitectos y desarrolladores. Está orientada a la creación de diseños usando el paradigma de programación orientada a objetos. Visual Paradigm incluye una herramienta llamada Visual Architect que permite la generación de código para el manejo de la base de datos. Con esta herramienta se puede generar código para los lenguajes PHP, JAVA y C# y para los gestores de base de datos DB2, Informix, SQL Server, MYSQL, Oracle y PostgreSql. Entre sus limitaciones está que el código generado hace uso de librerías propias, suprimiendo al desarrollador la posibilidad de escoger qué librería usar para acceder a los datos. El código generado no es de fácil comprensión para los desarrolladores. Es una herramienta propietaria que tiene un costo elevado. [20]

Licencia	Precio (USD)
Enterprise	$1399
Professional	$699
Standard	$299
Modeler	$99

Tabla 3.Precio por licencia de Visual Paradigm.

PHP Object Generator

PHP Object Generator, (POG) es un generador libre para aplicaciones WEB realizadas con PHP. Permite la generación de los métodos elementales (select, insert, delete, update) en forma de funciones. Es compatible con PHP4/PHP5. Brinda soporte para la codificación de caracteres. Es una aplicación de interfaz WEB. Según su interacción con el código se clasifica en pasivo y utiliza una aproximación estructural para generar código. Tiene como desventaja que sólo genera código en forma de funciones. [21]

PHPMyEdit

Herramienta orientada al manejo de bases de datos MYSQL desde PHP. Permite la creación de formularios para el manejo de la base de datos. Genera funciones para la manipulación de tablas. Permite el paginado de las consultas. Obtiene la estructura de la tabla directamente de la base de datos. Es libre y de código abierto. En cuanto a su interacción con el código se clasifica en pasivo y utiliza una aproximación estructural para generar código. Es un software libre. Tiene como desventaja que el código que genera no es de fácil comprensión por los desarrolladores porque no establece una división por capas del mismo. [3]

Codejay

Es una herramienta diseñada para ayudar a los desarrolladores en la creación de aplicaciones WEB. Permite el trabajo con varias plataformas e incorpora mecanismos de administración para la aplicación. Soporta varios sistemas gestores de base de datos (SQL Server, MS Access, MYSQL). Soporta varios lenguajes de programación (ASP, PHP). Permite la creación automática de reportes WEB. Inserta código JAVAscript para la validación dentro de los campos del formulario. En cuanto a su interacción con el código se clasifica en activo y utiliza una aproximación estructural para generar código. Tiene como desventaja que es un software propietario. [22]

Clarion/PHP templates

Permite la generación de aplicaciones WEB. Soporta la edición de los procedimientos para refinarlos. Posee un conjunto de plantillas conocidas como meta-base que contienen las funcionalidades más comunes a implementar en un software. Facilita la creación de asistentes manteniendo el estado de los objetos. Soporta varios gestores de base de datos. Implementa todas las capas de la aplicación. En cuanto a su interacción con el código se clasifica en pasivo y utiliza una aproximación estructural para generar código. Tiene como desventaja que es un software propietario. [23]

PHPGEN

Es un generador de archivos PHP para facilitar el trabajo y ahorrar tiempo de los desarrolladores. El funcionamiento es simple: Dada una tabla en una base de datos MYSQL, PHPGEN genera los archivos PHP suficientes para seleccionar, eliminar, generar un listado y modificar registros en la tabla. Esto permite un ahorro sustancial de tiempo para el programador, ya que éste ahora, no tendrá que programar los archivos generados y se ocupará de asuntos más específicos. Sólo trabaja con el gestor de base de datos MYSQL. Brinda compatibilidad para PHP4/PHP5. Es libre y de código abierto. En cuanto a su interacción con el código se clasifica en pasivo y utiliza una aproximación estructural para generar código. [24]

PHPGem

Es un generador de código libre que soporta como lenguaje de programación PHP y como gestores de base de datos a MYSQL, PostgreSQL, Sybase, SQL Server e Informix. Permite generar funciones para el manejo de la base de datos y la creación de interfaces HTML para la presentación de los mismos. Tiene como desventaja que la instalación se hace compleja ya que es necesario crear un archivo de configuración por cada tabla de la base de datos. Además no hace una división por capas del código, lo que dificulta el mantenimiento del mismo. En cuanto a su interacción con el código, se clasifica en pasivo y utiliza una aproximación estructural para generar código. [3]

Selección de las herramientas y tecnologías a utilizar.

PHP.

PHP (Personal Home Page) es un lenguaje de programación usado generalmente para la creación de contenido para sitios WEB. Es un lenguaje interpretado, multiplataforma y libre por lo que se ha convertido en uno de los mas difundidos en toda Internet. Últimamente también se puede usar para la creación de otro tipo de programas incluyendo aplicaciones con interfaz gráfica usando la biblioteca GTK+. (*PHP*). [5]

Ventajas.

- Es un lenguaje multiplataforma.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- No requiere de grandes recursos del sistema para su funcionamiento.
- Capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad, se destaca su conectividad con MYSQL
- Capacidad de expandir su potencial utilizando la enorme cantidad de módulos (llamados ext's o extensiones).
- Posee una amplia documentación en su página oficial entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un archivo de ayuda.
- Permite el paradigma de Programación Orientada a Objetos.
- Cuenta con una biblioteca nativa de funciones sumamente amplia.
- La sencillez de su sintaxis facilita su aprendizaje por lo que se ha difundido mucho en Internet.

Desventajas

- No está totalmente desarrollado en lo referente a la Programación Orientada a Objetos.
- No brinda gran rendimiento en aplicación de complejidad como el tratamiento de imágenes o de cálculos intensivos.

JAVA

Es un lenguaje de programación orientado a objetos desarrollado por Sun Microsystems a principios de los años 1990. Tiene una sintaxis parecida a la empleada por C++. Es multiplataforma. Es completamente libre. Es un lenguaje maduro que cuenta con un gran número de librerías disponibles. Se cuenta con

variedad de ambientes de desarrollo (Netbeans, Eclipse, J-Builder,…). Puede emplearse lo mismo para el desarrollo de aplicaciones de escritorio que para aplicaciones WEB. [6]

XML

Lenguaje extensible de marcas (Extensible Markup Language) es un lenguaje creado por W3C (World Wide WEB Consortium) para describir la estructura de los datos a almacenar en un documento. La sencillez de su estructura y la facilidad de su uso lo han convertido en uno de los estándares más difundidos en Internet. Se dice que a partir de su creación se ha cambiado la forma en que se hacen aplicaciones. La inmensa mayoría de los lenguajes de programación han implementado interfaces para trabajar con este nuevo estándar. [7][8]

XSD

Es un tipo de documento XML que tiene como objetivo describir la estructura y las restricciones de los contenidos de los documentos XML. Fue desarrollado por el World W3C (Wide WEB Consortium) y alcanzó el nivel de recomendación en mayo de 2001. Su principal ventaja es que permite comprobar si un documento cumple con una estructura dada. [9]

XSLT

Es un lenguaje para la trasformación de documentos XML en otros documentos. Es una recomendación de la W3C (World Wide WEB Consortium) de noviembre de 1999. Fue creado con el objetivo de separar el contenido de la presentación en la WEB permitiendo que dispositivos de diferentes características mostraran el mismo contenido. Su funcionamiento se basa en la búsqueda de elementos de un documento XML que coincidan con patrones definidos en XPAHT y la transformación de estos elementos en otro documento. [10]

Apache

Hoy en día es el servidor WEB más utilizado del mundo, encontrándose muy por encima de sus competidores, tanto gratuitos como comerciales. Es un software de código abierto que funciona sobre cualquier plataforma. Tiene capacidad para servir páginas tanto de contenido estático como de contenido dinámico. [11]

Ventajas

El servidor es mucho más flexible en tiempo de ejecución porque pueden añadirse módulos mediante comandos de configuración como LoadModule en httpd.conf en lugar de tener que hacerlo con las opciones de configuración al compilar. Por ejemplo, de esta manera uno puede ejecutar diferentes instancias del servidor con una única instalación de Apache.

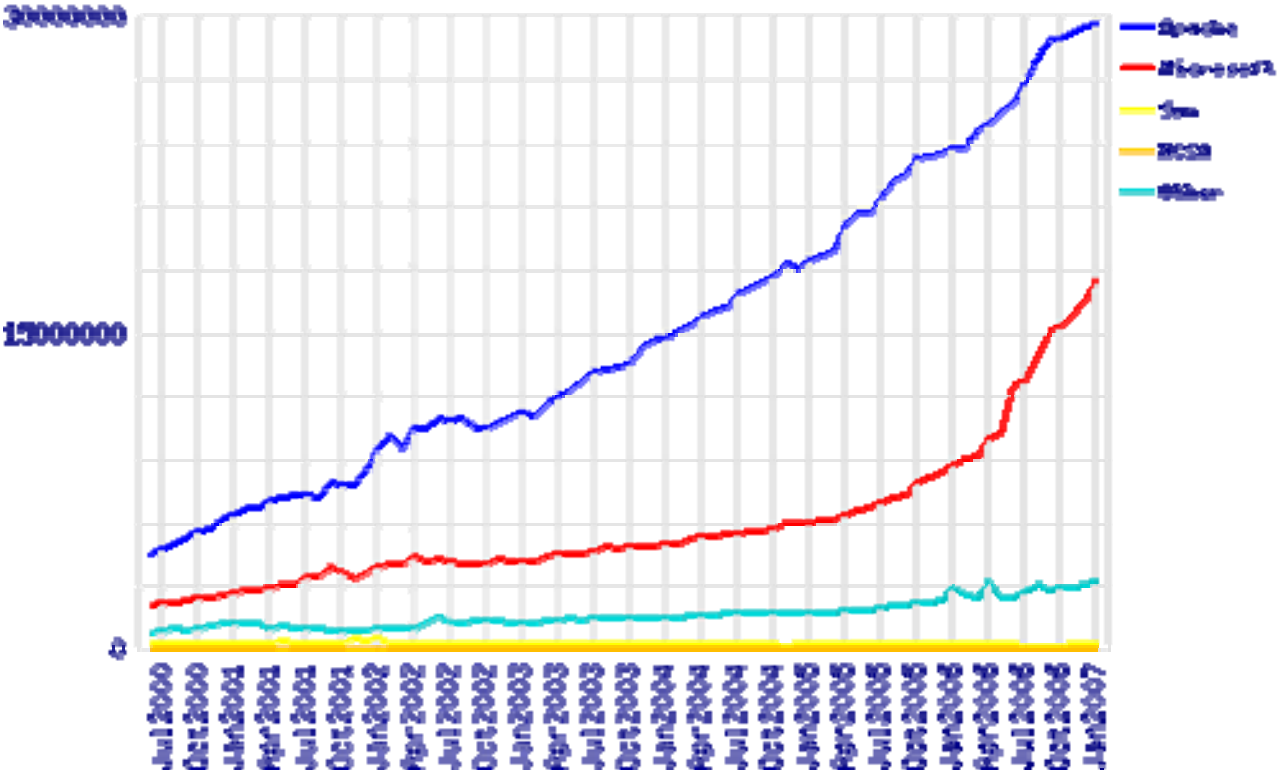

Fig 2. Número de servidores en Internet que utilizan cada tipo de sistema. [12]

MYSQL

MYSQL se ha convertido en uno de los más populares gestores de base de datos por su estabilidad, velocidad y rendimiento. Es usado por más de 10 millones de empresas en Internet y el hecho de que sea libre ha contribuido grandemente a su masificación. [13]

Ventajas

- Es software libre.
- MYSQL funciona sobre más de 20 plataformas.
- Es un gestor multi-usuario
- Seguridad: ofrece un sistema de contraseñas y privilegios seguro mediante verificación basada en el host y el tráfico de contraseñas está encriptado al conectarse a un servidor.
- Soporta gran cantidad de datos. MYSQL Server tiene bases de datos de hasta 50 millones de registros.
- Se permiten hasta 64 índices por tabla (32 antes de MYSQL 4.1.2). Cada índice puede consistir desde 1 hasta 16 columnas.
- Los clientes se conectan al servidor MYSQL usando sockets TCP/IP en cualquier plataforma.

RUP

El Proceso Unificado de Desarrollo de Software RUP (Rational Unified Process) es un proceso de desarrollo de software que junto al Lenguaje Unificado de Modelado UML constituyen la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. RUP es en realidad un refinamiento realizado por Rational Software del más genérico Proceso Unificado. [2]

Ventajas

- Es guiado por casos de uso.
- Es un proceso iterativo incremental.
- Es centrado en la arquitectura.
- Utiliza UML para el modelado visual.

Rational Rose:

- Es una herramienta visual para el análisis y diseño de aplicaciones.
- Utiliza UML como lenguaje de modelado.
- Abarca todas las etapas del desarrollo de software.
- Se integra con Eclipse.
- Soporta ingeniería inversa para los lenguajes C++ y JAVA.
- Genera código para los lenguajes JAVA/J2EE™, C++, Ada, ANSI C++, CORBA, Visual Basic y Visual C++.
- Genera documentación a partir de los modelos creados.
- Domina el mercado de herramientas para el análisis y diseño orientado a objetos.
- Es una herramienta propietaria y tiene un costo de $2955 por usuario. [14]

Conclusiones

Después de haber hecho un estudio de las principales herramientas existentes se decidió desarrollar una herramienta de generación de código, ya que ninguna de las disponibles responde a las necesidades del desarrollo de software en la Universidad. Una parte de ellas son propietarias y con elevados precios en sus licencias de desarrollo. Ninguna brinda la posibilidad de generar servicios WEB. El código que generan no siempre es de fácil compresión por parte de los desarrolladores. No todas cuentan con la posibilidad de definir nuevos mecanismos de generación de código y pocas brindan la posibilidad de generar interfaces para probar el buen funcionamiento del código generado. Inicialmente se hará un generador de capa de acceso a datos pasivo utilizando una aproximación estructural. Este generador tomará como entrada un modelo en XML de la base de datos y a través de plantillas en XSLT se obtendrá como salida las clases necesarias para el acceso a la base de datos.

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

Descripción general del sistema.

Funcionamiento del Generador:

El uso de XSLT facilita el proceso de transformación de un modelo a código. Este lenguaje permite transformar un documento XML en otro tipo de documento. De esta forma a partir de un modelo XML que contenga la información de una tabla de la base de datos y una plantilla en XSLT para realizar las transformaciones se obtiene como resultado código en PHP para acceder a los datos de la tabla. A continuación se muestra un diagrama de su funcionamiento.

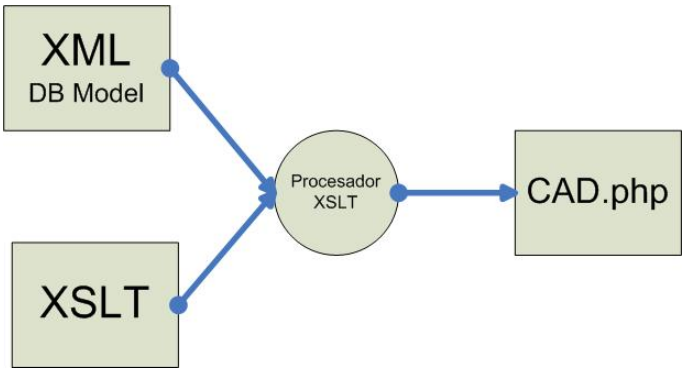

Fig 3. Transformación XSLT.

Resuelto el problema de la transformación de un modelo XML a código en PHP falta definir cómo el generador llevará a cabo las transformaciones y qué modelos usar para obtener el resultado deseado. Para esto se crean patrones que guían el proceso de transformación. Un patrón contiene la información de qué modelos XML y qué plantillas XSLT se deben usar para obtener un resultado deseado. El patrón es un archivo XML el cual cumple con un esquema definido. Básicamente en un patrón hay modelos y transformaciones. Cada modelo tiene un nombre y un tipo. El nombre de un modelo debe ser único ya que sirve para identificarlo dentro del patrón. Las transformaciones pueden ser de distintos tipos y usar uno o varios modelos.

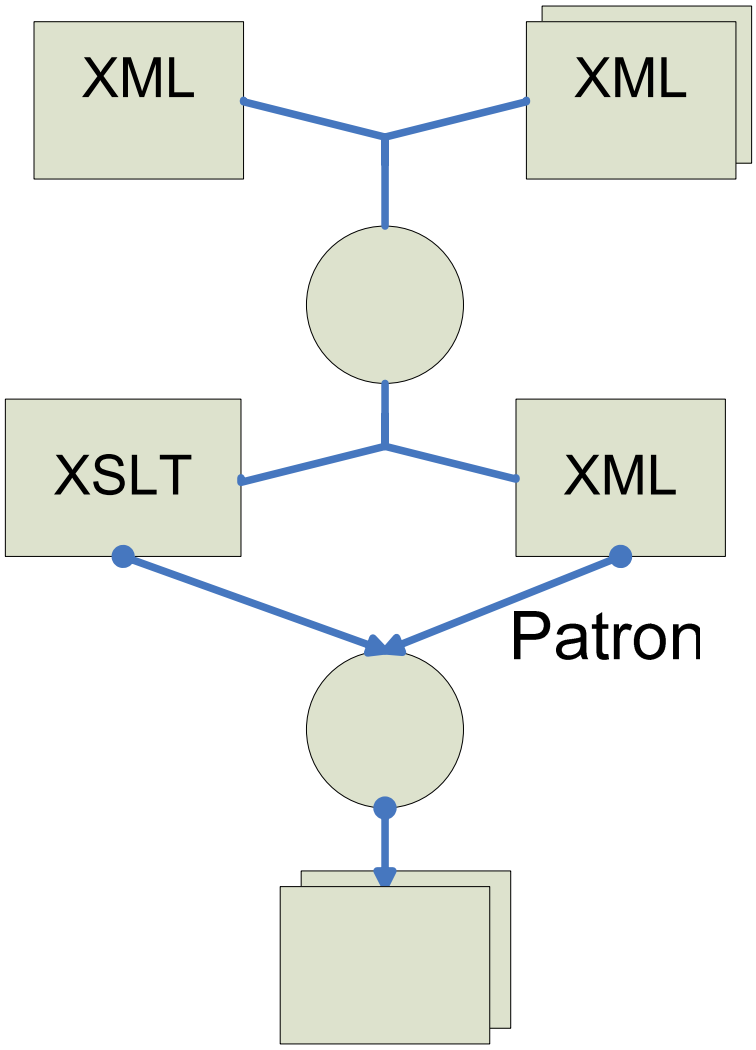

Fig 4. Funcionamiento del generador.

Procesador de Patrones

Procesador XSLT

Por ejemplo si se tiene la estructura de una tabla en un modelo XML y se quiere generar el código de acceso a esta tabla, se puede definir un patrón que reciba un modelo con la estructura de la tabla y realice una transformación usando un XSLT dado. El patrón pudiera quedar así:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<patron>
  <name>CAD_TABLA</name>
  <descripciom> genera el acceso a una tabla</descripciom>
  <lenguage>PHP</lenguage>
  <modelos>
     <modelo>
      <name>Tabla</name>
      <tipo>FILE</tipo>
    </modelo>
  </modelos>
  <transformaciones>
    <transformacion>
      <name>CAD</name>
      <modelo_name> Tabla </modelo_name>
      <xslt>cad.xsl</xslt>
      <filename>
        <value>CAD</value>
        <ext>.PHP</ext>
      </filename>
    </transformacion>
  </transformaciones>
</patron>
```

El patrón tiene un nombre, una descripción y el lenguaje para el que está definido. El patrón no contiene los datos del modelo, el sólo le sirve de guía al generador indicándole que debe recibir un modelo que está en un archivo (tipo FILE) y realizar una transformación XSLT utilizando “cad.xsl” poniendo el resultado en el archivo “CAD.PHP”.

El uso de patrones que guíen el proceso de generación requiere de un procesador capaz de interpretar estos patrones y realizar las transformaciones correspondientes. En los patrones se pueden definir cinco tipos de transformaciones que combinadas permiten guiar al proceso de generación hasta obtener el resultado deseado.

Las transformaciones definidas son:

- transformación.
- foreach.
- xpathOp.
- addOp.
- copyfile.

Transformación: utiliza un modelo XML y una plantilla XSLT para realizar una transformación utilizando un procesador XSLT. El resultado se obtiene en un archivo o en una variable según se especifique. Las variables son valores intermedios en forma de nuevos modelos que utiliza el patrón para realizar el proceso de generación.

La estructura en XML de una transformación cuenta con un nombre (<name>), modelo a transformar (<modelo_name>), plantilla XSLT a usar (<xslt>) y un indicador que señala si se pondrá el resultado en un archivo (<filename>) o en una variable (<var>).

```
<transformacion>
    <name>Entidad</name>
    <modelo_name>ariable_nombre</modelo_name>
    <xslt>file.xslt</xslt>
    <filename>
      <value>valor</value>
      <modelo_name>modelo_nombre</modelo_name>
      <xpath>/root/elemento</xpath>
      <ext>.PHP</ext>
    </filename>
 </transformacion>
```

Si el resultado va para un archivo el nombre del archivo se obtendrá concatenando una valor fijo (<value>) con el resultado de una consulta XPATH (<xpath>) sobre un modelo (<modelo_name>) y poniéndole la extensión (<ext>) indicada. Si uno de esto elementos no se encuentra no se incluye en el nombre del archivo. Esto permite que dicho nombre, pueda ser obtenido de diversas formas según el propósito del patrón. Si el resultado se pone en una variable se indicará el nombre (<name>) y quedará como se muestra a continuación:

```
    <transformacion>
  <name>Entidad</name>
  <modelo_name>ariable_nombre</modelo_name>
  <xslt>file.xslt</xslt>
  <var>
    <name>valor</name>
  </var>
 </transformacion>
```

Foreach: permite iterar por un conjunto de elementos resultados de una consulta XPATH. La variable de control de ciclo (<var>) almacena cada uno de los elementos por los cuales se está iterando. Estos elementos son obtenidos aplicándole una consulta XPATH (<xpath>) a un modelo (<modelo_name>). En el interior del foreach se pueden anidar transformaciones de todos los tipos y estas transformaciones pueden usar cualquiera de los modelos existentes o la variable de control de ciclo.

```
<foreach>
 <var>variable_nombre</var>
 <modelo_name>modelo_nombre_ciclo</modelo_name>
 <xpath>/root/elemento</xpath>
 <transformacion>
   <name>Entidad</name>
   <modelo_name>ariable_nombre</modelo_name>
   <xslt>file.xslt</xslt>
   <filename>
     <value>valor</value>
     <ext>.PHP</ext>
     <modelo_name>modelo_nombre</modelo_name>
     <xpath>/root/elemento</xpath>
   </filename>
 </transformacion>
</foreach>
```

XpathOp: realiza una consulta XPATH (<xpath>) a un modelo (<modelo_name>) y la asigna a una variable (<var>). Es útil para seleccionar parte de un modelo antes de realizar una transformación.

```
<xpathOp>
  <var>variable_nombre</var>
  <modelo_name>modelo_nombre</modelo_name>
  <xpath>/root/elemento</xpath>
</xpathOp>
```

AddOp: une el modelo A (<modelo_namea>) y el modelo B (<modelo_nameb>) en un solo modelo y el resultado lo asigna a la variable (<var>). Es útil cuando se tiene la información en varios modelos y se quieren unir para hacerle una transformación.

```
<addOp>
  <var>variable_nombre</var>
  <modelo_namea>modelo_nombre</modelo_namea>
  <modelo_nameb>modelo_nombre</modelo_nameb>
</addOp>
```

Copyfile: permite copiar un archivo de entrada (<file>) a un destino especificado (<out>). Es útil cuando se necesitan archivos como imágenes o librerías para el buen funcionamiento del código generado.

```
<copyfile>
  <file>imagenes\\fondo.jpg</file>
  <out>imagenes\\fondo.jpg</out>
</copyfile>
```

La decisión de utilizar patrones que guíen el proceso de generación de código permite que un generador se adapte a muchas situaciones. El uso de esta alternativa separa el desarrollo del generador del desarrollo de los patrones de generación de código aumentando así la flexibilidad del generador.

Modelo de dominio:

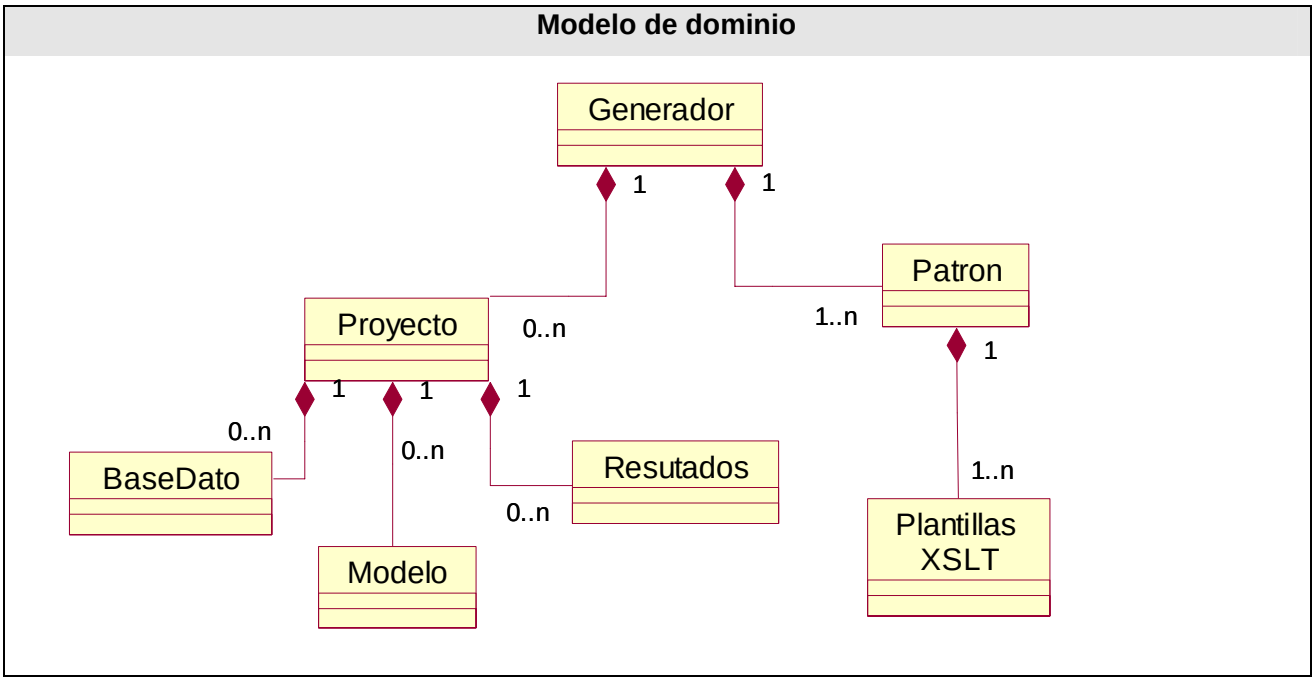

Fig 5. Modelo de dominio.

Requisitos funcionales:

El sistema debe:

RF-1. Crear nuevos proyectos suministrando un nombre y una carpeta para su ubicación.

RF-2. Abrir un proyecto suministrando el nombre y la carpeta en que se encuentra.

RF-3 Guardar un proyecto.

RF-4. Mostrar los patrones de generación existentes.

RF-5. Agregar un nuevo patrón de generación a partir del nombre y la carpeta en que se encuentra.

RF-6. Eliminar un patrón de generación suministrando el nombre.

RF-7.Registrar bases de datos suministrando el servidor, el nombre de la base de datos, el usuario y la contraseña.

RF-8.Mostrar las bases de datos registradas.

RF-9.Guardar en el proyecto las bases de datos registradas.

RF-10.Comprobar conexión a la base de datos.

RF-11.Obtener un nuevo modelo con la estructura de la base de datos suministrando el nombre del modelo.

RF-12.Guardar en el proyecto los modelos creados.

RF-13.Mostrar modelos existentes.

RF-14.Definir un módulo a partir de un modelo de base de datos suministrando el nombre y las tablas que lo forman.

RF-15.Generar la capa de acceso a datos.

RF-16.Generar servicios WEB a partir de un modelo de base de datos.

RF-17.Generar interfaces que permitan interactuar con la base de datos.

RF-18.Mostrar los resultados de la generación.

RF-19.Guardar resultados generados en el proyecto.

Requisitos no funcionales:

Software

RNF1- Para su funcionamiento debe estar instalada la maquina virtual de JAVA.

Interfaz

RNF2- El sistema debe tener una interfaz amigable.

RNF3- El sistema debe ser una aplicación escritorio.

Diseño e implementación.

RNF4- El sistema será implementado en JAVA.

RNF5- El sistema usará como gestor de base de datos MYSQL.

Portabilidad

RNF6- El sistema debe ser multiplataforma y funcionar en Windows y Linux.

Usabilidad

RNF7- El sistema debe tener documentación de ayuda al usuario.

RNF8- El código generado debe ser fácil de entender y de modificar.

Actor del Sistema

Actor	Descripción
Desarrollador	Es quien usará la herramienta de generación en el desarrollo de un proyecto.

Tabla 4. Actores del sistema.

Diagrama de Casos de Uso del Sistema:

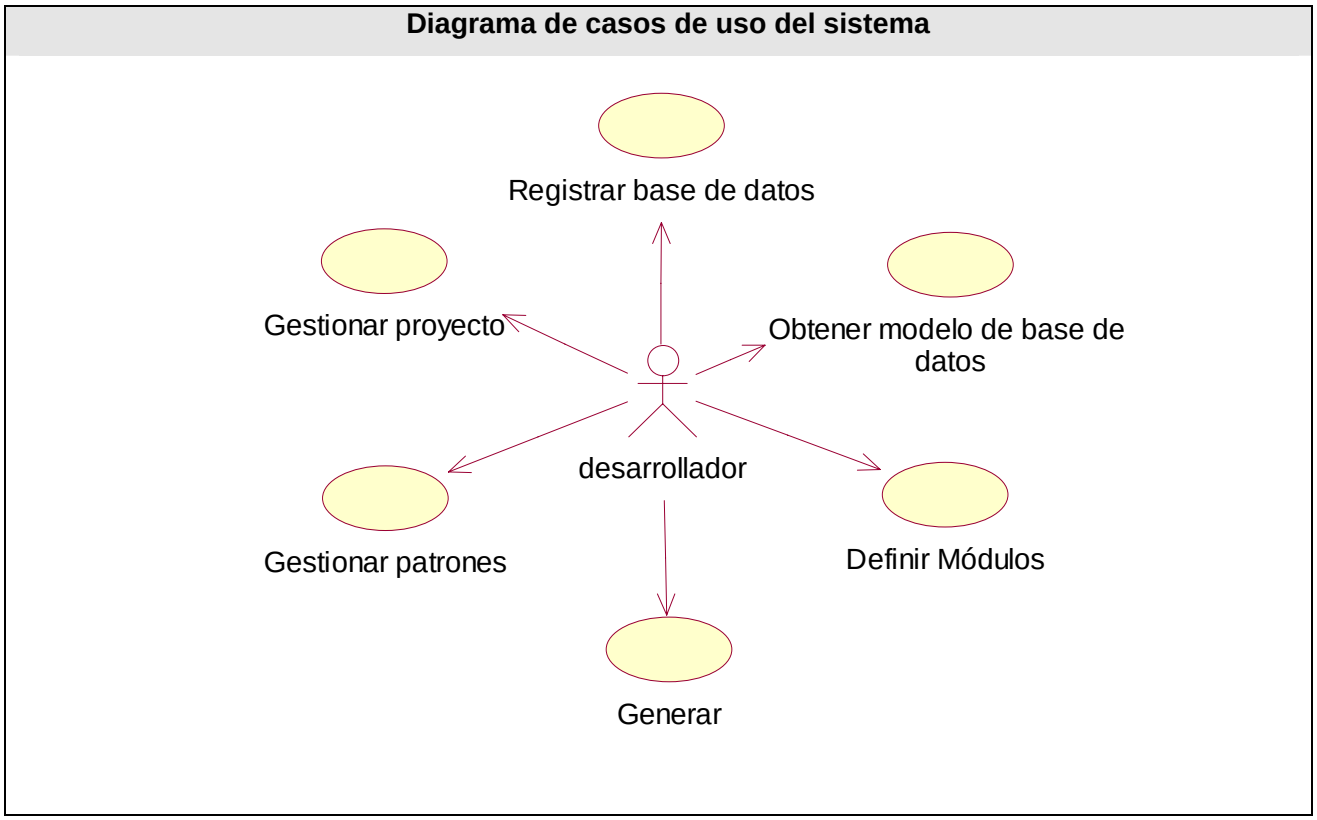

Fig 6. Diagrama de casos de uso del sistema.

Resultado de priorizar los casos de uso.
Críticos:

- Gestionar Proyecto.
- Registrar base de datos.
- Obtener modelo de base de datos.
- Definir módulos.
- Generar.

Secundarios:

- Gestionar patrones.

Descripción de los Casos de Uso del sistema.

Caso de Uso:	Gestionar Proyecto	
Actores:	Desarrollador	
Resumen:	Permite al desarrollador crear, abrir y guardar proyectos.	
Referencia:	RF-1, RF2, RF3,	
Flujo Normal de Eventos		
Acción del Actor		**Respuesta del Sistema**
1. El desarrollador selecciona crear nuevo proyecto.		2. El sistema solicita que introduzca el nombre y la carpeta en la que desea crearlo.
3. El desarrollador introduce los datos al sistema.		4. El sistema crea el proyecto en la carpeta seleccionada finalizando así el caso de uso.
Sección: abrir proyecto.		
Acción del Actor		**Respuesta del Sistema**
1. El desarrollador selecciona abrir un proyecto.		2. El sistema solicita que introduzca el nombre y la carpeta en que está el proyecto.
3. El desarrollador introduce los datos al sistema.		5. El sistema abre el proyecto y finaliza así el caso de uso.
Sección: guardar proyecto.		
Acción del Actor		**Respuesta del Sistema**
1. El desarrollador selecciona guardar un proyecto.		2. El sistema guarda el proyecto en la carpeta que fue creado y finaliza así el caso de uso.
Prioridad:	Crítico	

Tabla 5. Descripción del Caso de uso “Gestionar Proyecto”.

Caso de Uso:	Registrar base de datos	
Actores:	Desarrollador	
Resumen:	El desarrollador registra una base de datos introduciendo el servidor en que está, el nombre, el usuario y la contraseña. Una vez terminado el proceso la base de datos estará disponible para trabajar.	
Referencia:	RF7, RF-9, RF-10	
Precondiciones:	Que se haya creado un proyecto y esté abierto.	
Flujo Normal de Eventos		
Acción del Actor		**Respuesta del Sistema**
1. El desarrollador introduce el servidor, el nombre, el usuario y la contraseña con la que desea registrar una base de datos.		2. El sistema comprueba la conexión a la base de datos.
		3. El sistema guarda en el proyecto los datos de la base de datos registrada y finaliza el caso de uso.
Poscondiciones:	Queda una base de datos registrada y lista para usarse.	
Prioridad:	Crítico	

Tabla 6. Descripción del Caso de uso “Registrar Base de datos”.

Caso de Uso:	Obtener modelo de base de datos	
Actores:	Desarrollador	
Resumen:	El desarrollador obtiene un modelo que representa la estructura de la base de datos.	
Referencia:	RF-8, RF-12, RF-11	
Precondiciones:	Que ya esté registrada alguna base de datos.	
Flujo Normal de Eventos		
Acción del Actor		**Respuesta del Sistema**
1. El desarrollador solicita las bases de datos registradas.		2. El sistema muestra las bases de datos que están registradas
3. El desarrollador selecciona la base de datos de la que va a obtener el modelo e introduce el nombre.		3. El sistema obtiene el modelo y lo guarda en el proyecto terminando así el caso de uso.
Poscondiciones:	Se obtiene un modelo de base de datos.	
Prioridad:	Crítico	

Tabla 7. Descripción del Caso de uso “Registrar Base de datos”.

Caso de Uso:	Definir módulos	
Actores:	Desarrollador	
Resumen:	Permite al desarrollador dividir una base de datos en módulos para trabajar con ellos por separado.	
Referencia:	RF-13, RF-14.	
Precondiciones:	Que ya se haya cargado el modelo de la base de datos	
Flujo Normal de Eventos		
Acción del Actor		**Respuesta del Sistema**
1. El desarrollador solicita los modelos de base de datos disponibles		2. El sistema muestra los modelos de base de datos que se han cargado.
3. El desarrollador selecciona un modelo, introduce el nombre del módulo que desea crear y selecciona las tablas que pertenecen a este módulo.		4. El sistema almacena la información correspondiente al módulo y termina el caso de uso.
Poscondiciones:	Se creará un módulo.	
Prioridad:	Crítico	

Tabla 8. Descripción del Caso de uso “Definir módulos”.

Caso de Uso:	Gestionar patrones	
Actores:	Desarrollador	
Resumen:	Permite al desarrollador agregar o eliminar los patrones utilizados en el proceso de generación.	
Referencia:	RF-4, RF-5, RF-6	
Flujo Normal de Eventos		
Acción del Actor		**Respuesta del Sistema**
1. El desarrollador selecciona agregar patrón.		2. El sistema le solicita la carpeta y el nombre del archivo que contiene el patrón.
3. El desarrollador introduce los datos.		4. El sistema agrega el patrón y termina el caso de uso.
Sección: eliminar patrón.		
Acción del Actor		**Respuesta del Sistema**
1. El desarrollador selecciona eliminar un patrón.		2. El sistema muestra los patrones que se han agregado.
3. El desarrollador selecciona el patrón que desea eliminar.		2. El sistema elimina el patrón y termina el caso de uso.
Prioridad:	Secundario	

Tabla 9. Descripción del Caso de uso “Gestionar Patrones”.

Caso de Uso:	Generar	
Actores:	Desarrollador	
Resumen:	Permite al desarrollador generar código a partir de un modelo de una base de datos.	
Referencia:	RF-15, RF-16, RF-17, RF-18, RF-19	
Precondiciones:	Que ya se haya cargado un modelo de la base de datos.	
Flujo Normal de Eventos		
Acción del Actor		**Respuesta del Sistema**
1. El desarrollador solicita los modelos de base de datos disponibles		2. El sistema muestra los modelos de base de datos que se han obtenido.
3. El desarrollador selecciona el modelo y el patrón con que desea generar código.		4. El sistema genera el código, guarda el resultado de la generación en el proyecto y lo muestra finalizando así el caso de uso.
Poscondiciones:	Debe generarse el código.	
Prioridad:	Crítico	

Tabla 10. Descripción del Caso de uso “Genera”.

Conclusiones

Una vez capturados los requisitos del sistema se puede proceder a la fase de análisis y diseño. Los casos de uso identificados en el flujo de trabajo de requisitos guiarán el proceso de desarrollo a lo largo de los flujos de trabajo de análisis, diseño, implementación y prueba. El venidero diseño tendrá que dar respuesta a todos los requisitos del sistema, tanto funcionales como no funcionales.

CAPÍTULO 3: ANÁLISIS Y DISEÑO DEL SISTEMA

Introducción

En este capítulo se expone cómo se traducen los requisitos a un diseño que describe cómo implementar el sistema. Para esto se crea por cada caso de uso, un diagrama de clases del análisis. Se crean los diagramas de colaboración para cada escenario dentro de los casos de uso y se hace el diagrama de clases del diseño distribuyendo estas clases en paquetes.

Modelo de análisis:

En la construcción del modelo de análisis se identifican las clases que describen la realización de los casos de uso, los atributos y las relaciones entre ellas. Con esta información se construye el diagrama de clases del análisis.

Diagramas de clases del análisis.

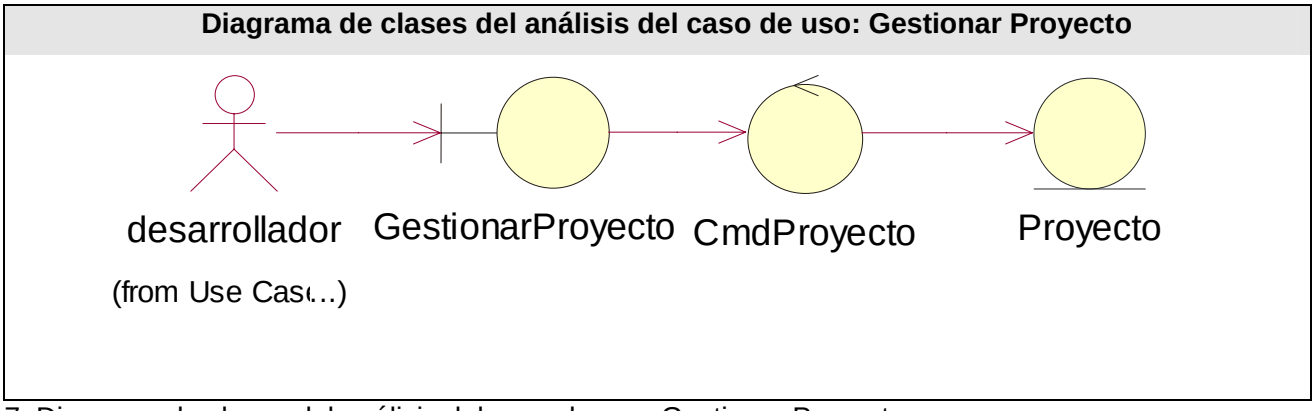

Fig 7. Diagrama de clases del análisis del caso de uso: Gestionar Proyecto.

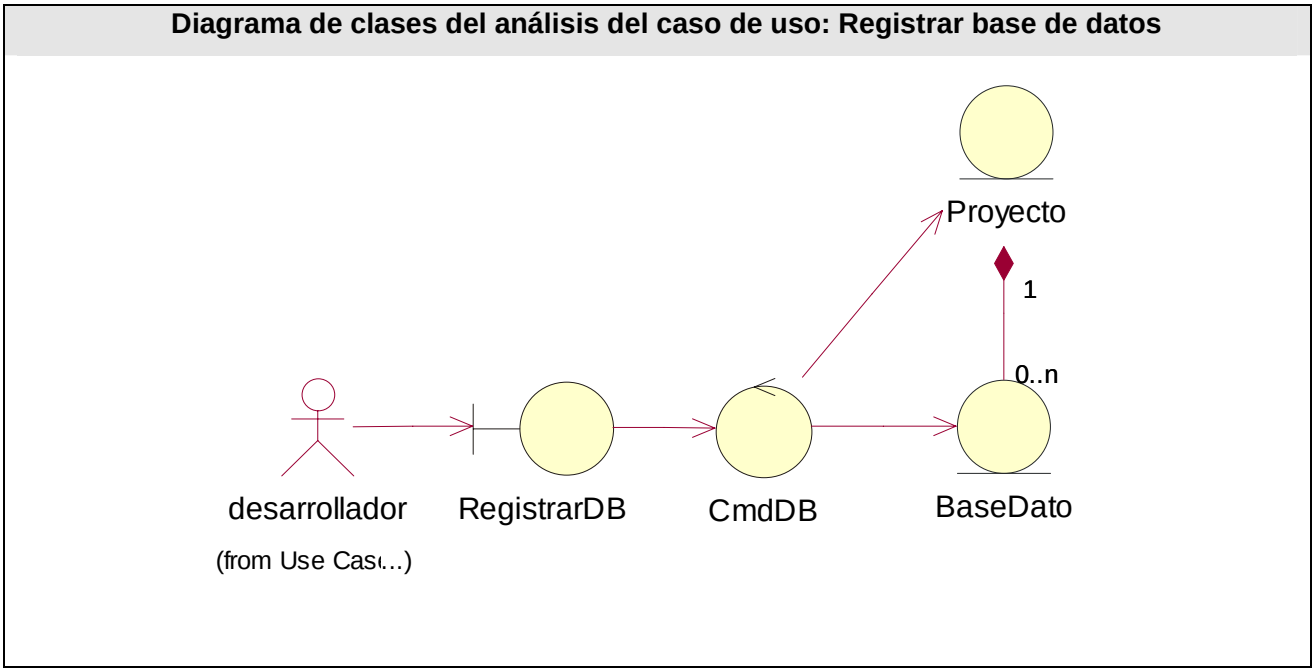

Fig 8. Diagrama de clases del análisis del caso de uso: Registrar base de datos.

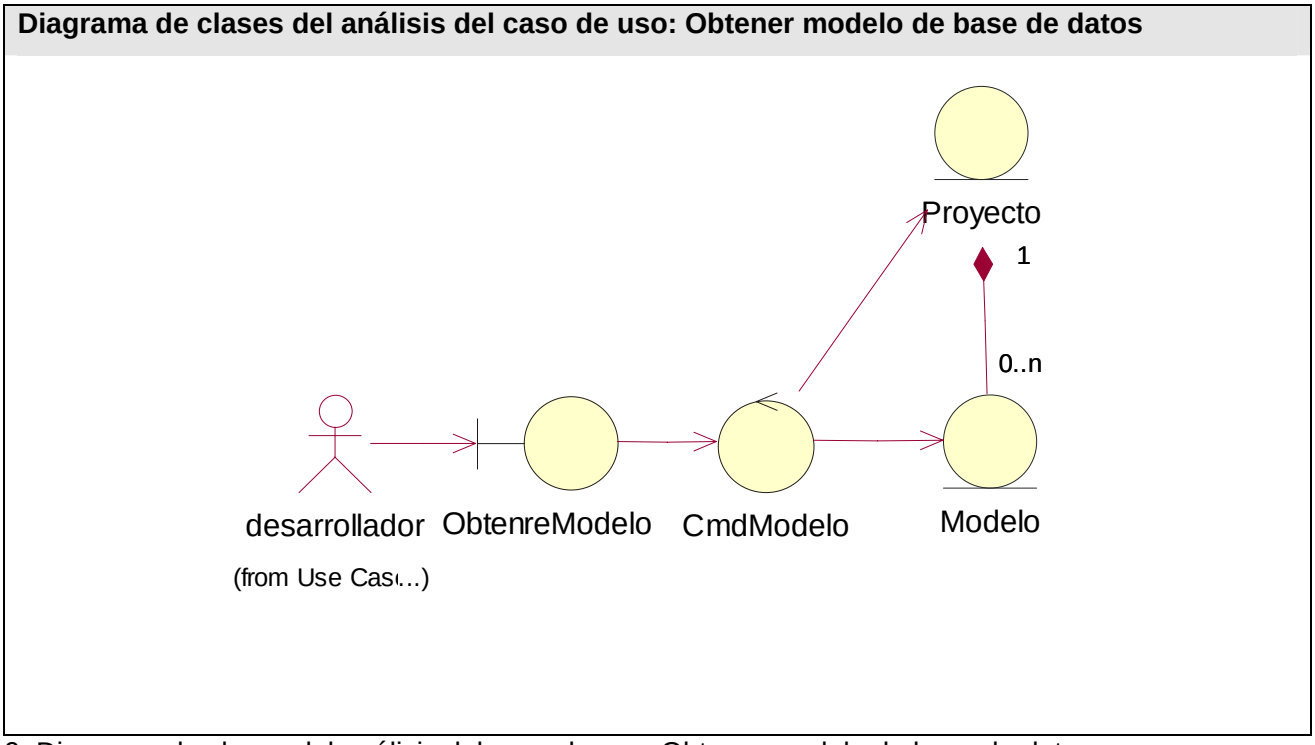

Fig 9. Diagrama de clases del análisis del caso de uso: Obtener modelo de base de datos.

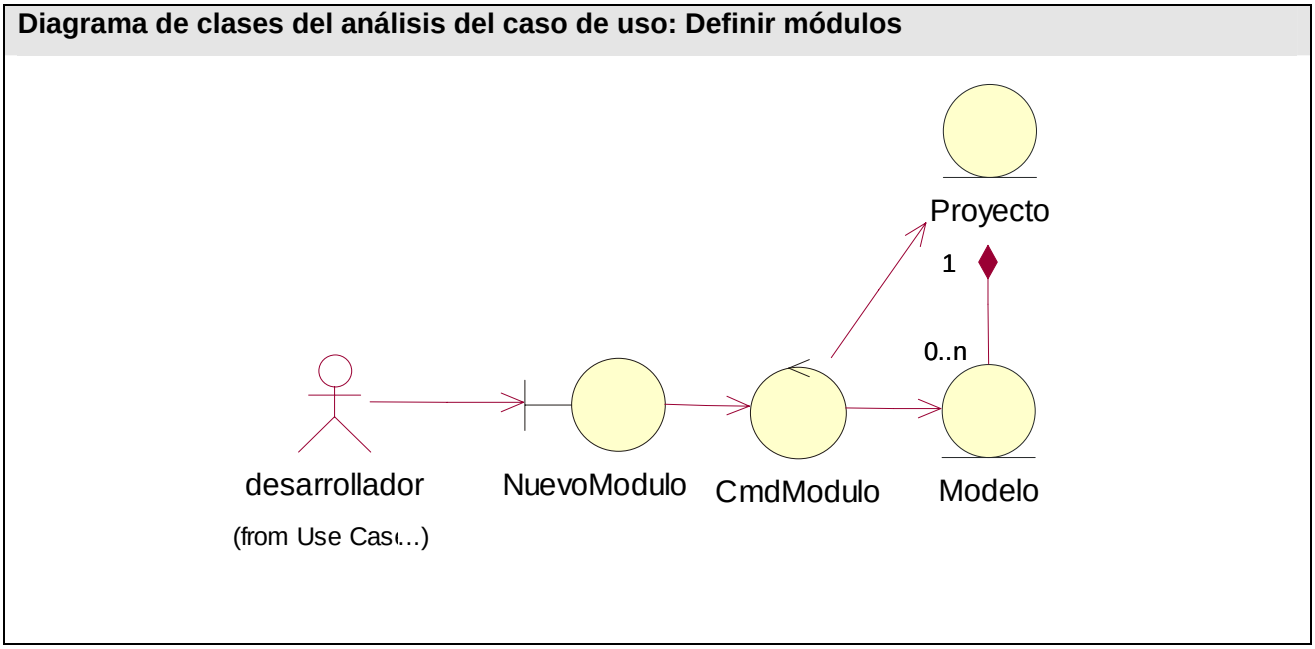

Fig 10. Diagrama de clases del análisis del caso de uso: Definir módulos.

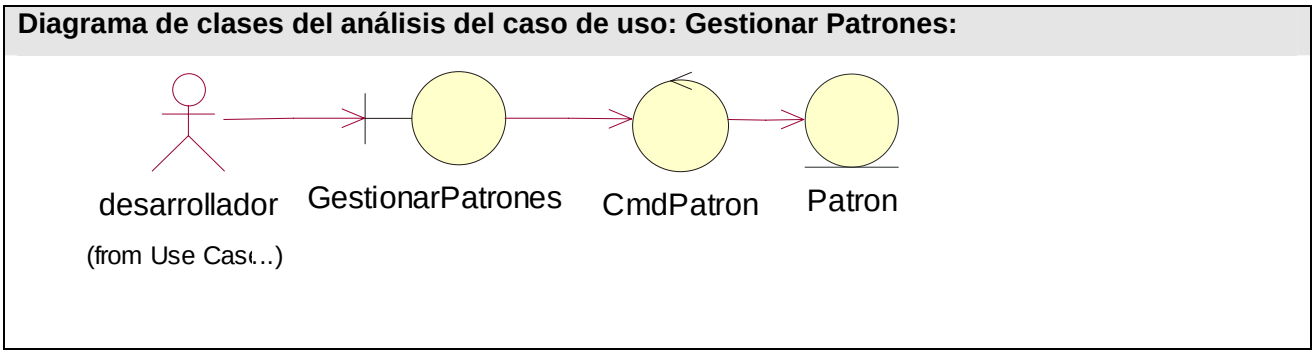

Fig 11. Diagrama de clases del análisis del caso de uso: Gestionar Patrones.

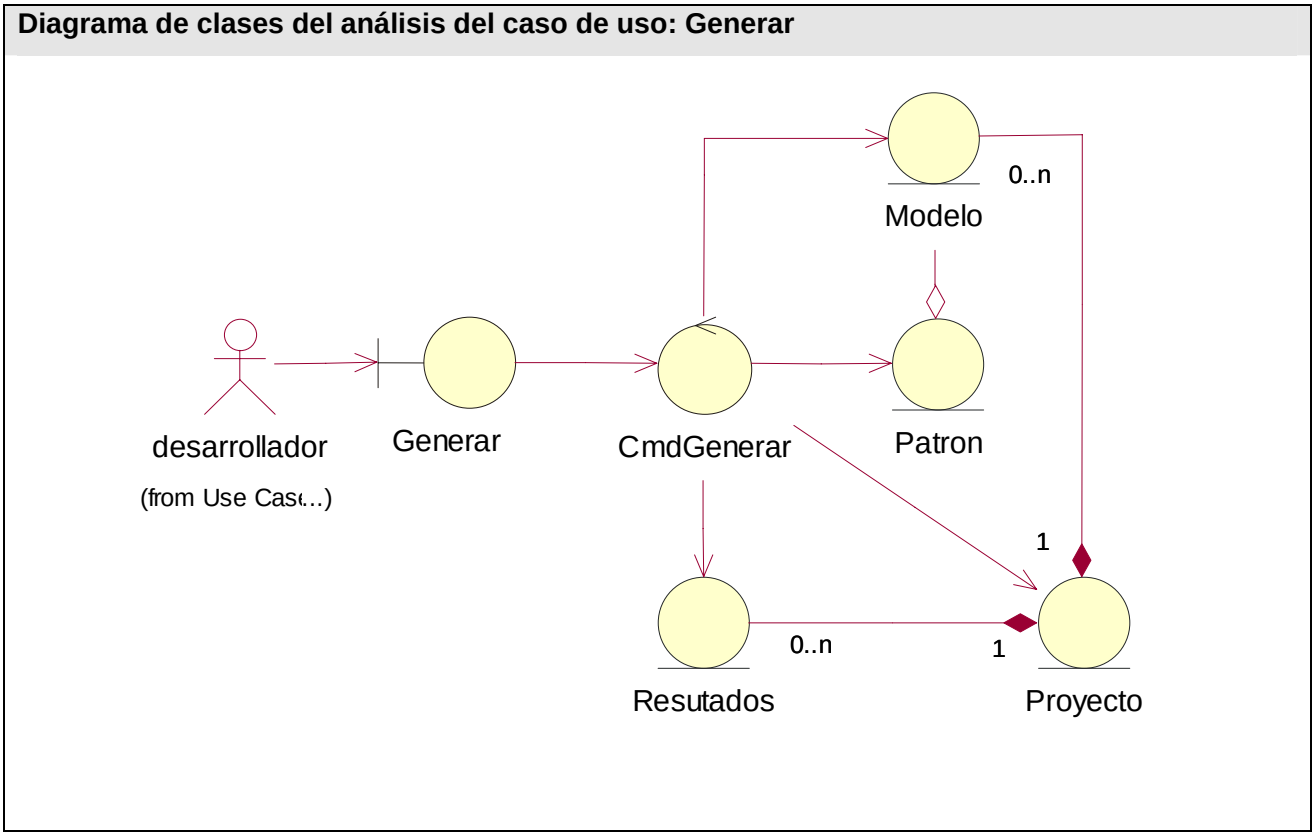

Fig 12. Diagrama de clases del análisis del caso de uso: Generar.

Diagramas de interacción.

Diagrama de secuencia del escenario “nuevo proyecto” del caso de uso: Gestionar Proyecto.

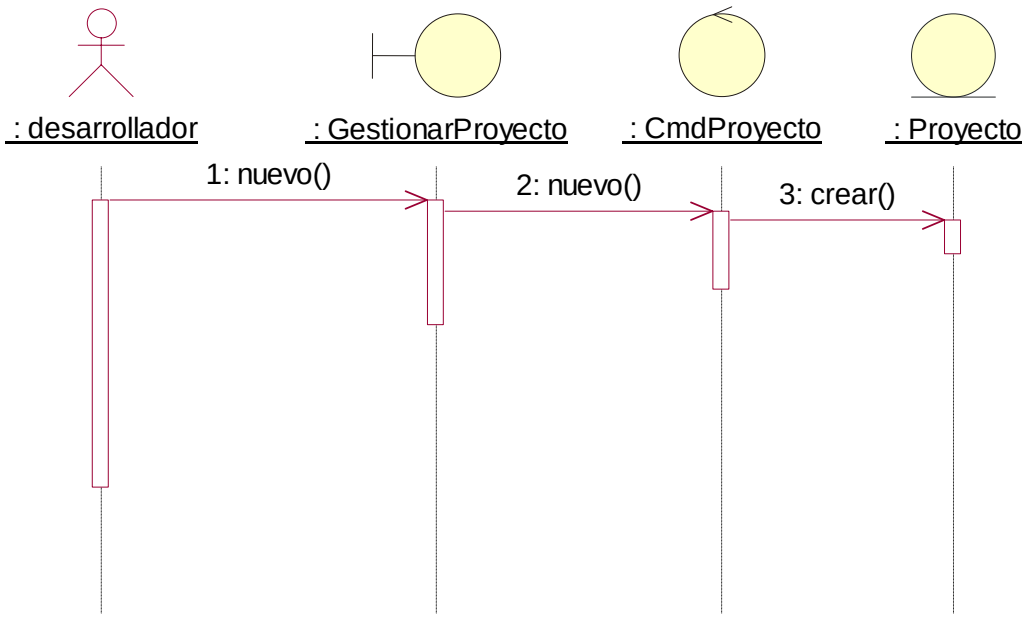

Fig 13. Diagrama de secuencia del escenario “nuevo proyecto” del caso de uso: Gestionar Proyecto.

Diagrama de secuencia del escenario “abrir proyecto” del caso de uso: Gestionar Proyecto.

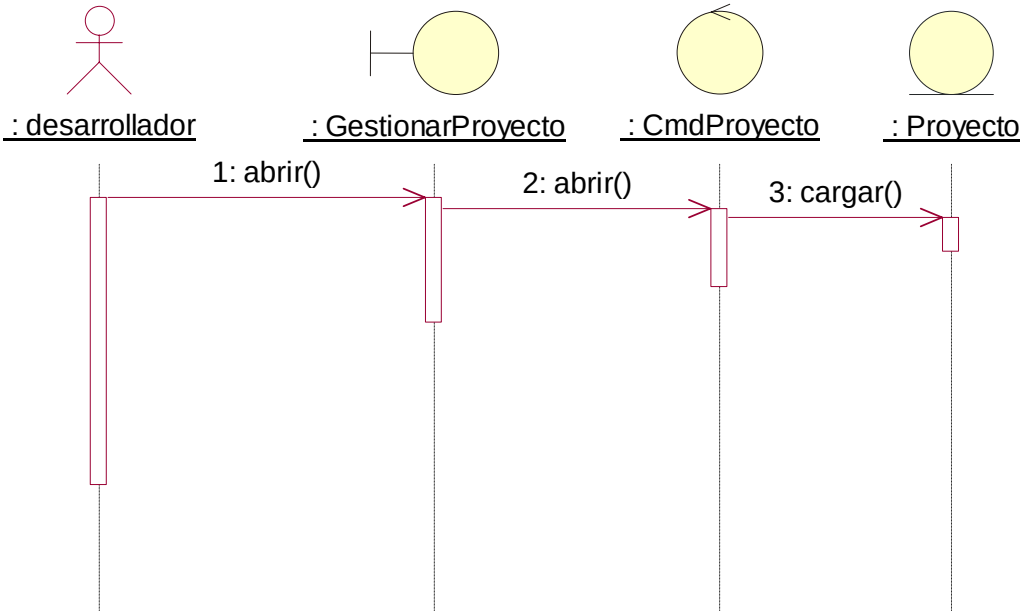

Fig 14. Diagrama de secuencia del escenario “nuevo proyecto” del caso de uso: Gestionar Proyecto.

Diagrama de secuencia del escenario “guardar proyecto” del caso de uso: Gestionar Proyecto.

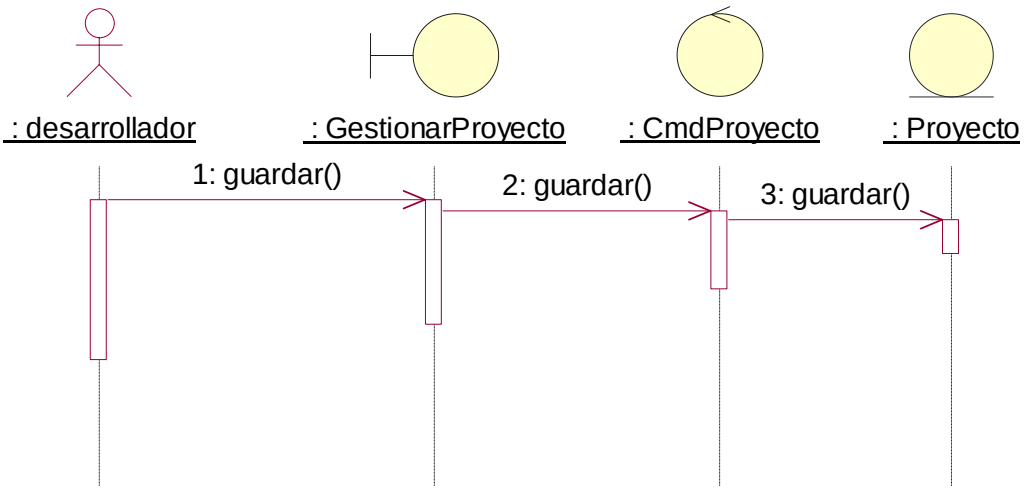

Fig 15. Diagrama de secuencia del escenario “guardar proyecto” del caso de uso: Gestionar Proyecto.

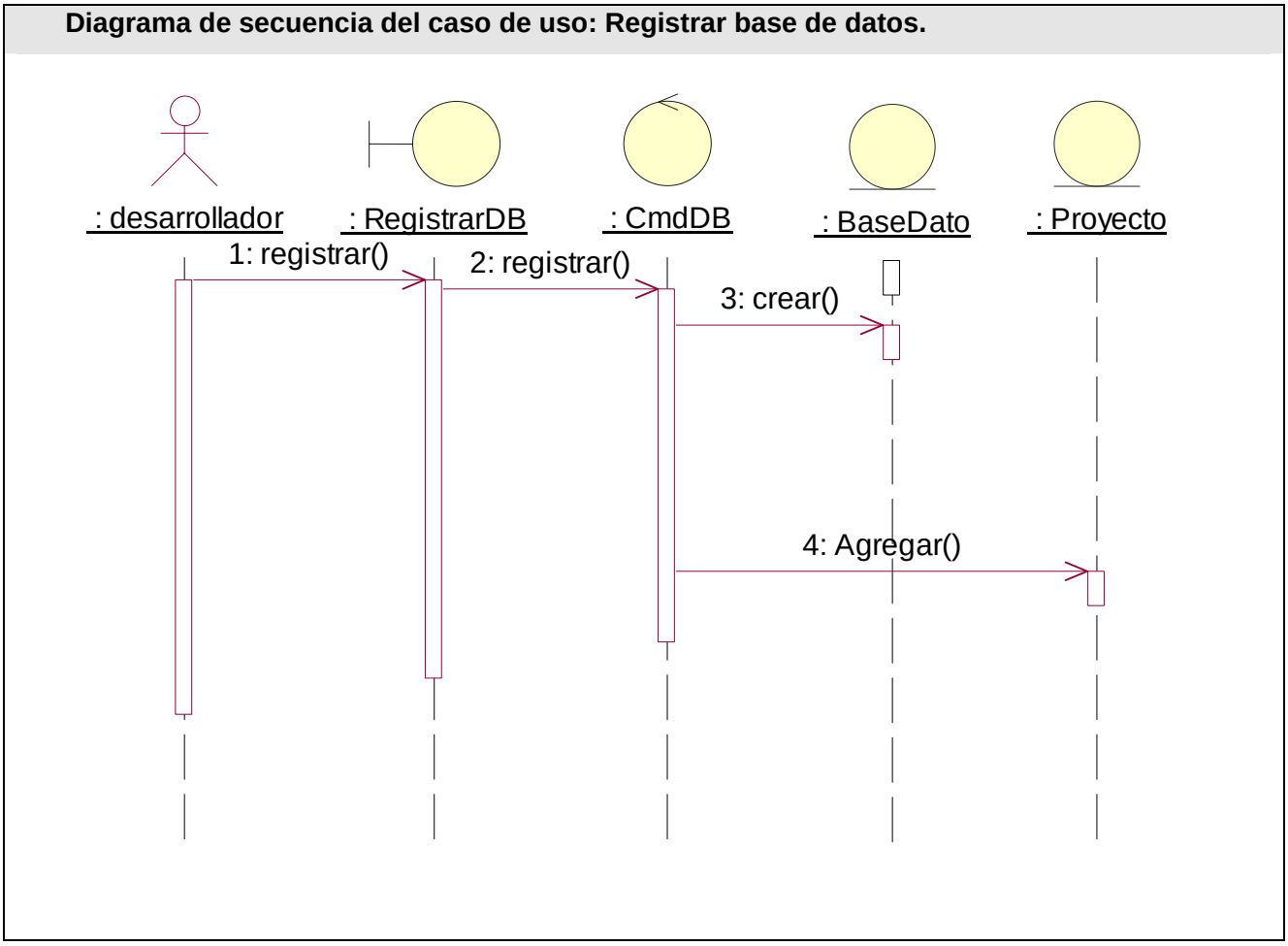

Fig 16. Diagrama de secuencia del caso de uso: Registrar base de datos.

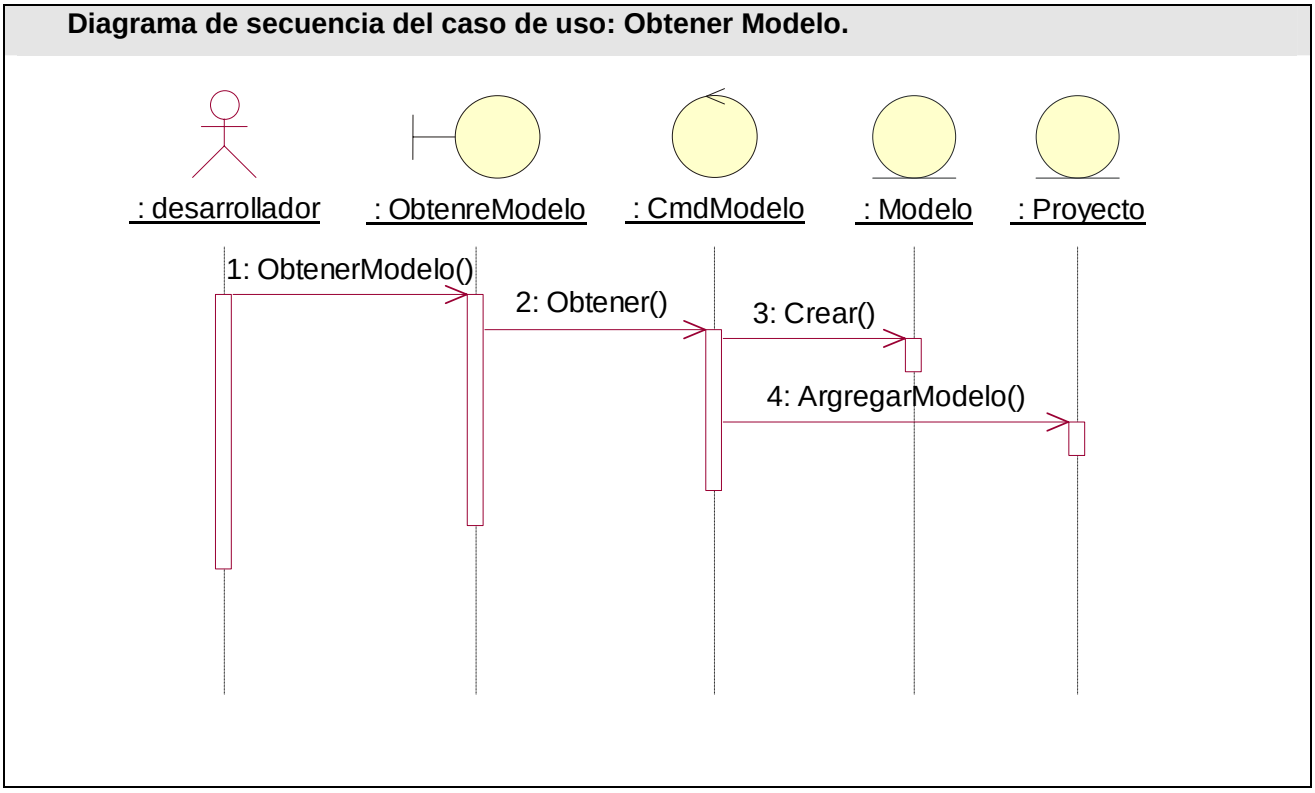

Fig 17. Diagrama de secuencia del caso de uso: Obtener Modelo.

Diagrama de secuencia del caso de uso: Definir Módulo.

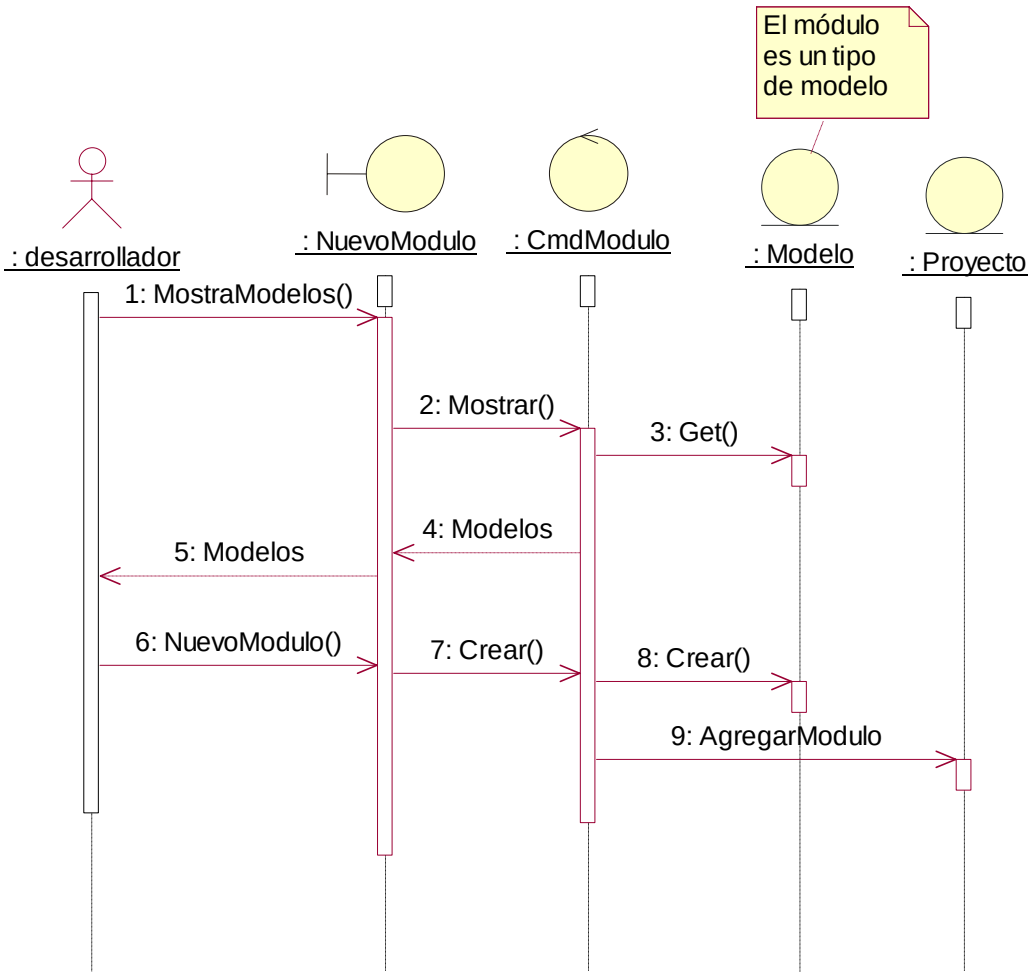

Fig 18. Diagrama de secuencia del caso de uso: Definir Módulo.

Diagrama de secuencia del escenario “agregar patrón” del caso de uso: Gestionar Patrones.

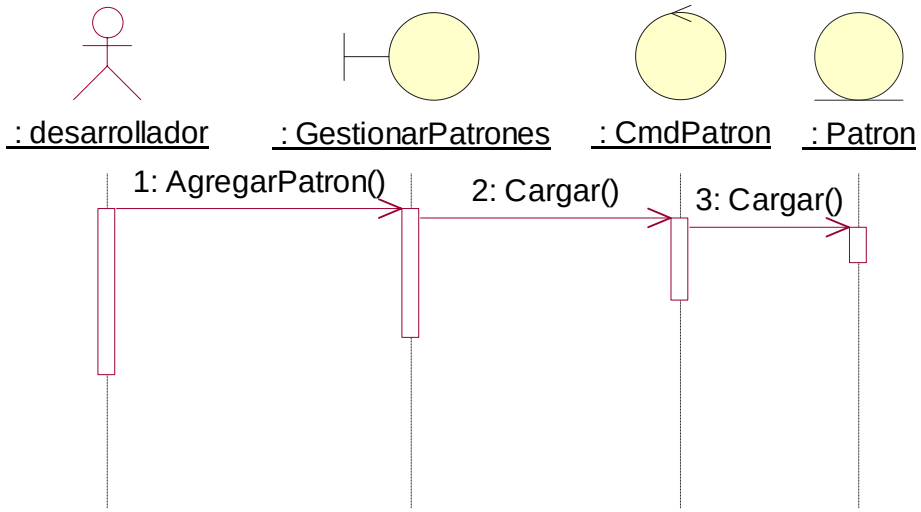

Fig 19. Diagrama de secuencia del escenario “agregar patrón” del caso de uso: Gestionar Patrones.

Diagrama de secuencia del escenario “eliminar patrón” del caso de uso: Gestionar Patrones.

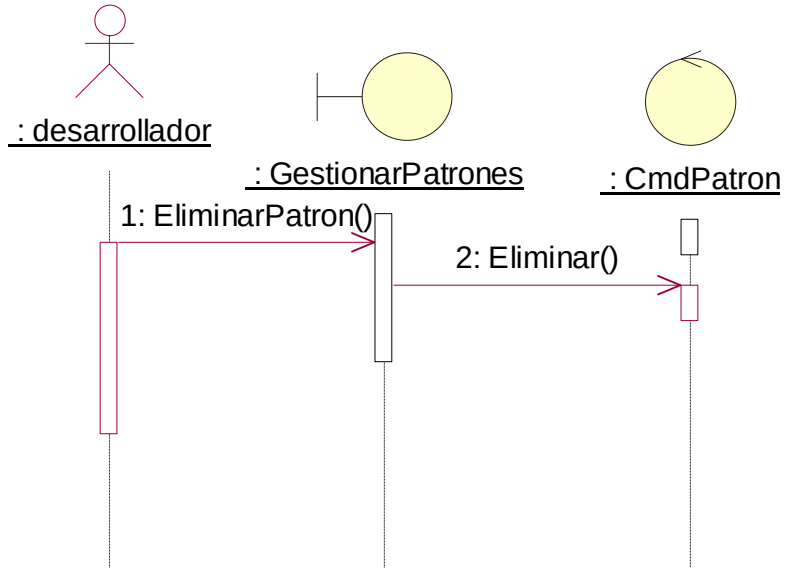

Fig 20. Diagrama de secuencia del escenario “eliminar patrón” del caso de uso: Gestionar Patrones.

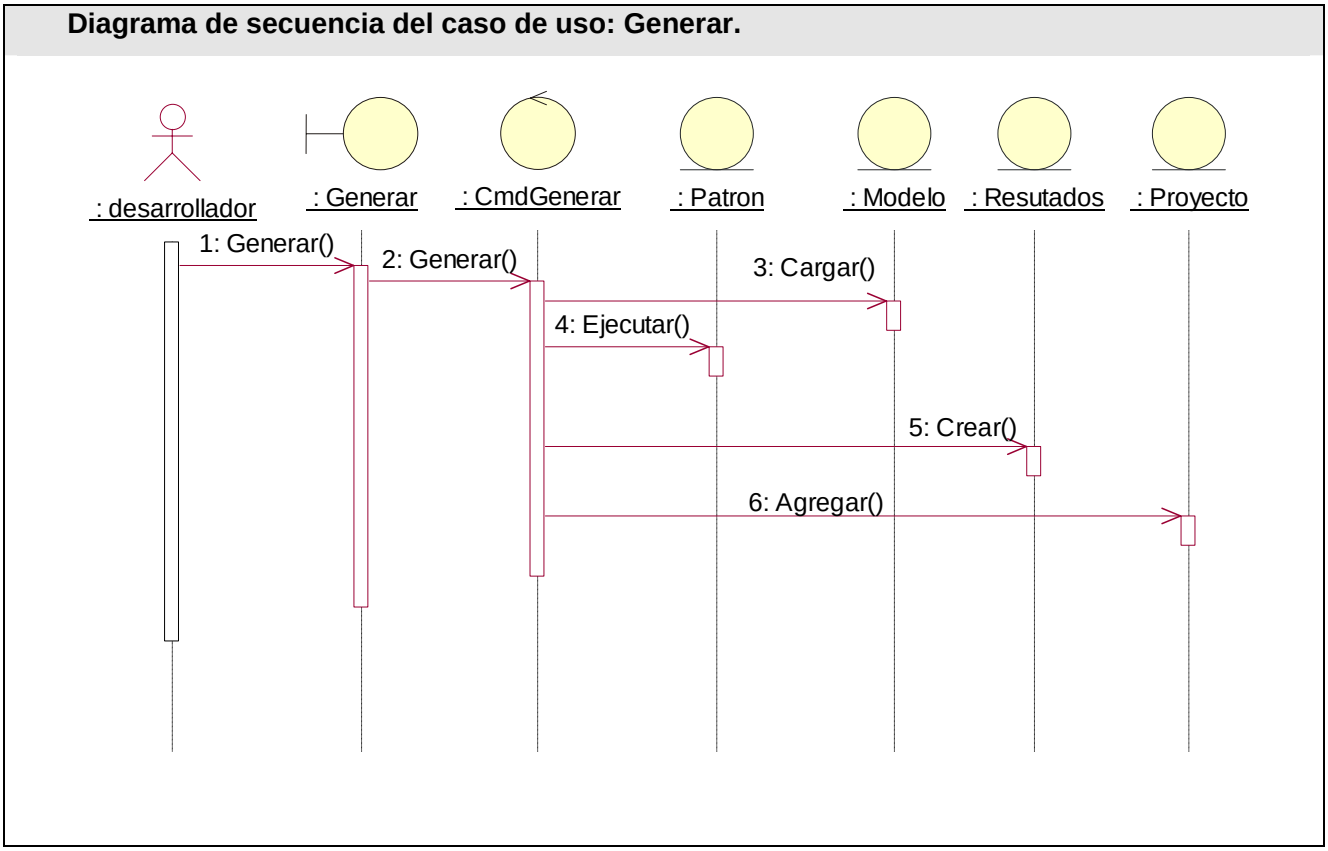

Fig 21. Diagrama de secuencia del caso de uso: Generar.

Modelo de Diseño

En este modelo se crea un diseño que da respuesta a los requisitos del software teniendo en cuenta el leguaje de programación a utilizar, los componentes y las tecnologías empleadas. Con este modelo se tiene un punto de partida para comenzar la implementación de software. [2]

Patrones de diseño empleados

Singleton

El patrón de diseño singleton (instancia única) está diseñado para restringir la creación de objetos pertenecientes a una clase. Su intención consiste en garantizar que una clase sólo tenga una instancia y proporcionar un punto de acceso global a ella.

El patrón singleton se implementa creando en la clase un método que crea una instancia del objeto sólo si todavía no existe alguna. Para asegurar que la clase no puede ser instanciada nuevamente se regula la construcción de los objetos. En muchos lenguajes esto se logra restringiendo el alcance del constructor a través de atributos como protegido o privado.

Las situaciones más habituales de aplicación de este patrón son aquellas en las que dicha clase controla el acceso a un recurso físico único (como puede ser el ratón o un archivo abierto en modo exclusivo) o cuando cierto tipo de datos debe estar disponible para todos los demás objetos de la aplicación. [16]

El patrón singleton provee una única instancia global gracias a que:

- La propia clase es responsable de crear la única instancia.
- Permite el acceso global a dicha instancia mediante un método de clase.
- Declara el constructor de clase como privado para que no sea instanciable directamente.

Command

Intención

Este patrón permite solicitar una operación a un objeto sin conocer realmente el contenido de esta operación, ni el receptor real de la misma. Para ello se encapsula la petición como un objeto, con lo que además se facilita la parametrización de los métodos. [17]

Propósito

- Encapsula un mensaje como un objeto, con lo que permite gestionar colas o registro de mensaje y deshacer operaciones.
- Soportar para restaurar el estado a partir de un momento dado.
- Ofrecer una interfaz común que permita invocar las acciones de forma uniforme y extender el sistema con nuevas acciones de forma más sencilla.

Motivo

- El concepto de "orden" puede ser ambiguo y complejo en los sistemas actuales y al mismo tiempo muy extendido: intérpretes de órdenes del sistema operativo, lenguajes de macros de paquetes ofimáticos, gestores de bases de datos, protocolos de servidores de Internet, etc.
- Este patrón presenta una forma sencilla y versátil de implementar un sistema basado en comandos facilitándose su uso y ampliación.

Aplicaciones

- Facilitar la parametrización de las acciones a realizar.
- Independizar el momento de petición del de ejecución.
- Soportar el "deshacer".
- Desarrollar sistemas utilizando órdenes de alto nivel que se construyen con operaciones sencillas (primitivas).

Paquetes del diseño.

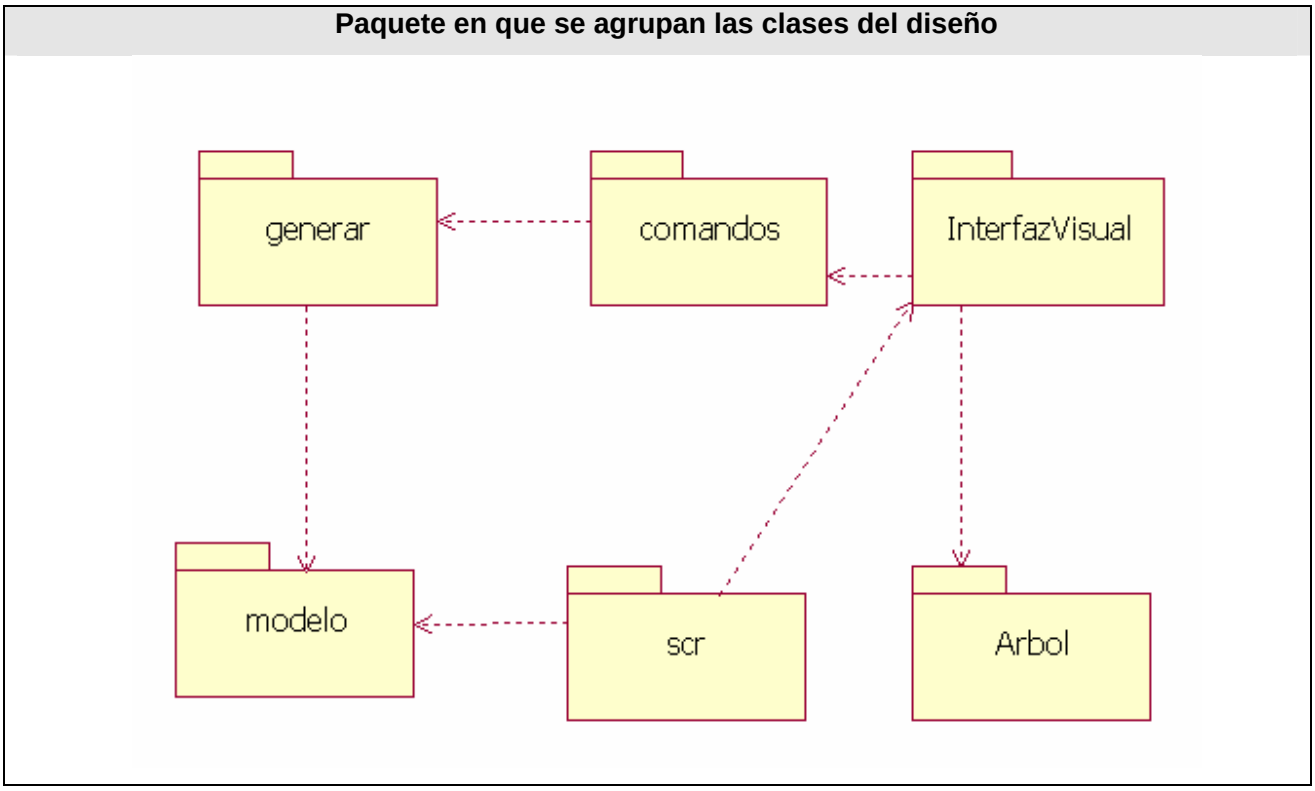

Fig 22. Paquete en que se agrupan las clases del diseño.

Diagrama de Clases del diseño.

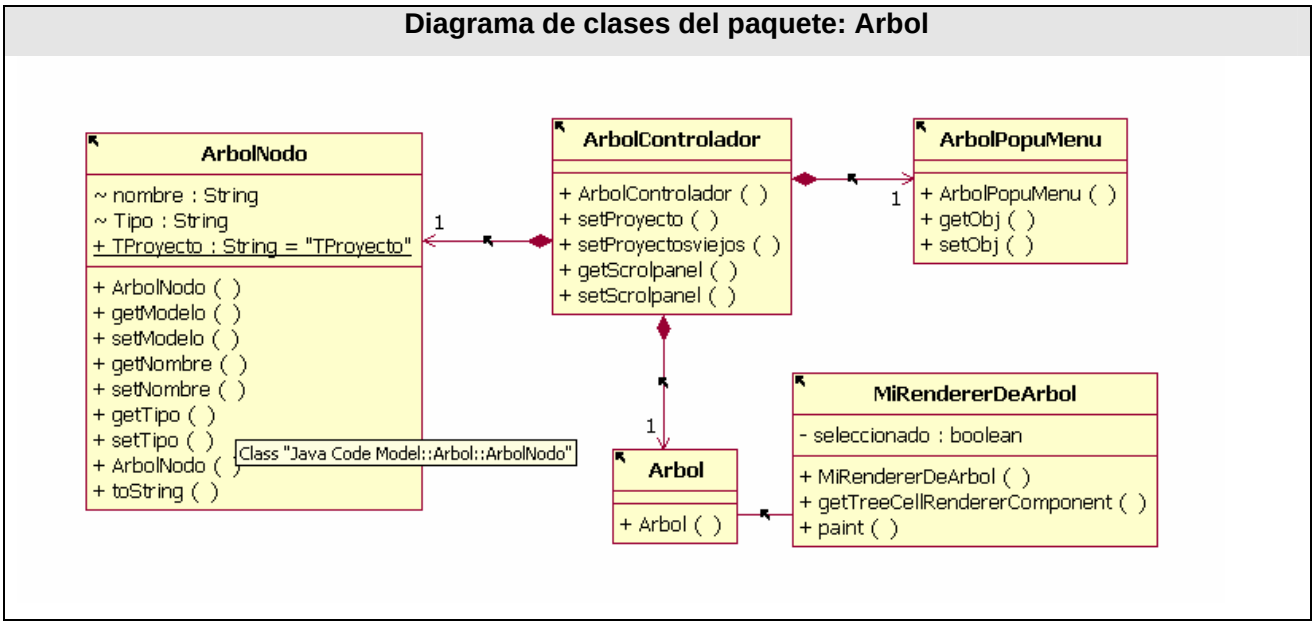

Fig 23. Diagrama de clases del paquete: Arbol.

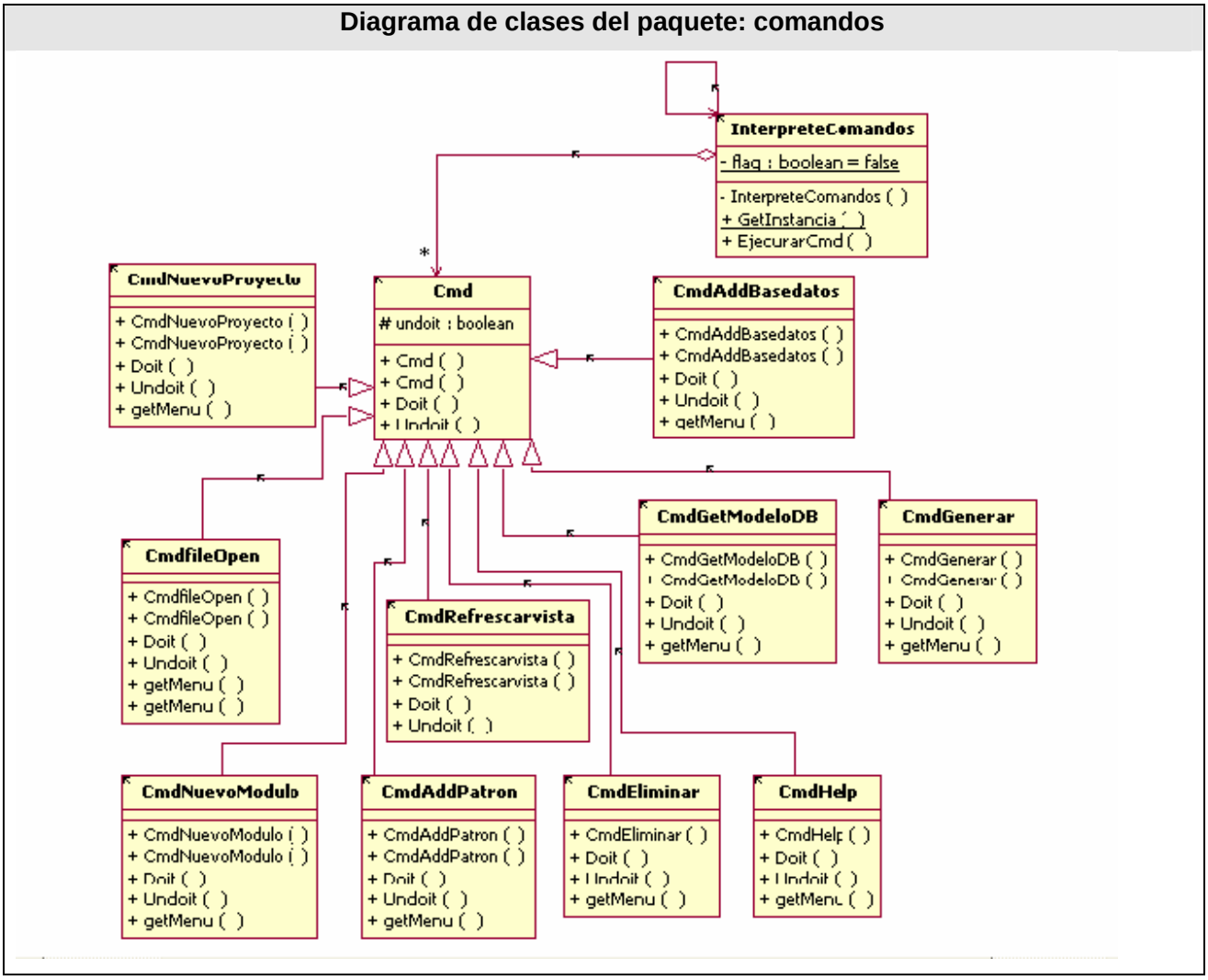

Fig 24. Diagrama de clases del paquete: comandos.

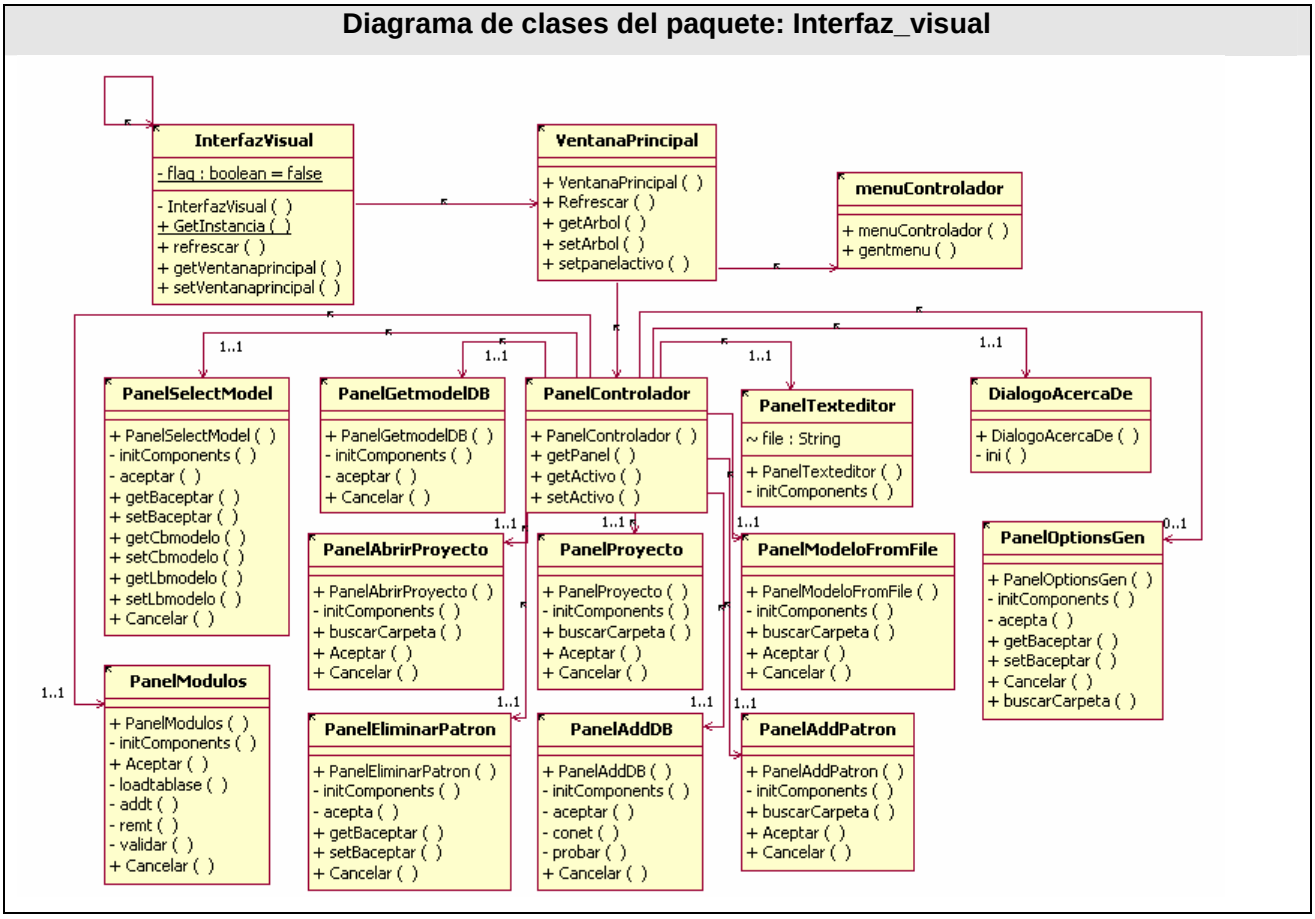

Fig 25. Diagrama de clases del paquete: Interfaz_visual.

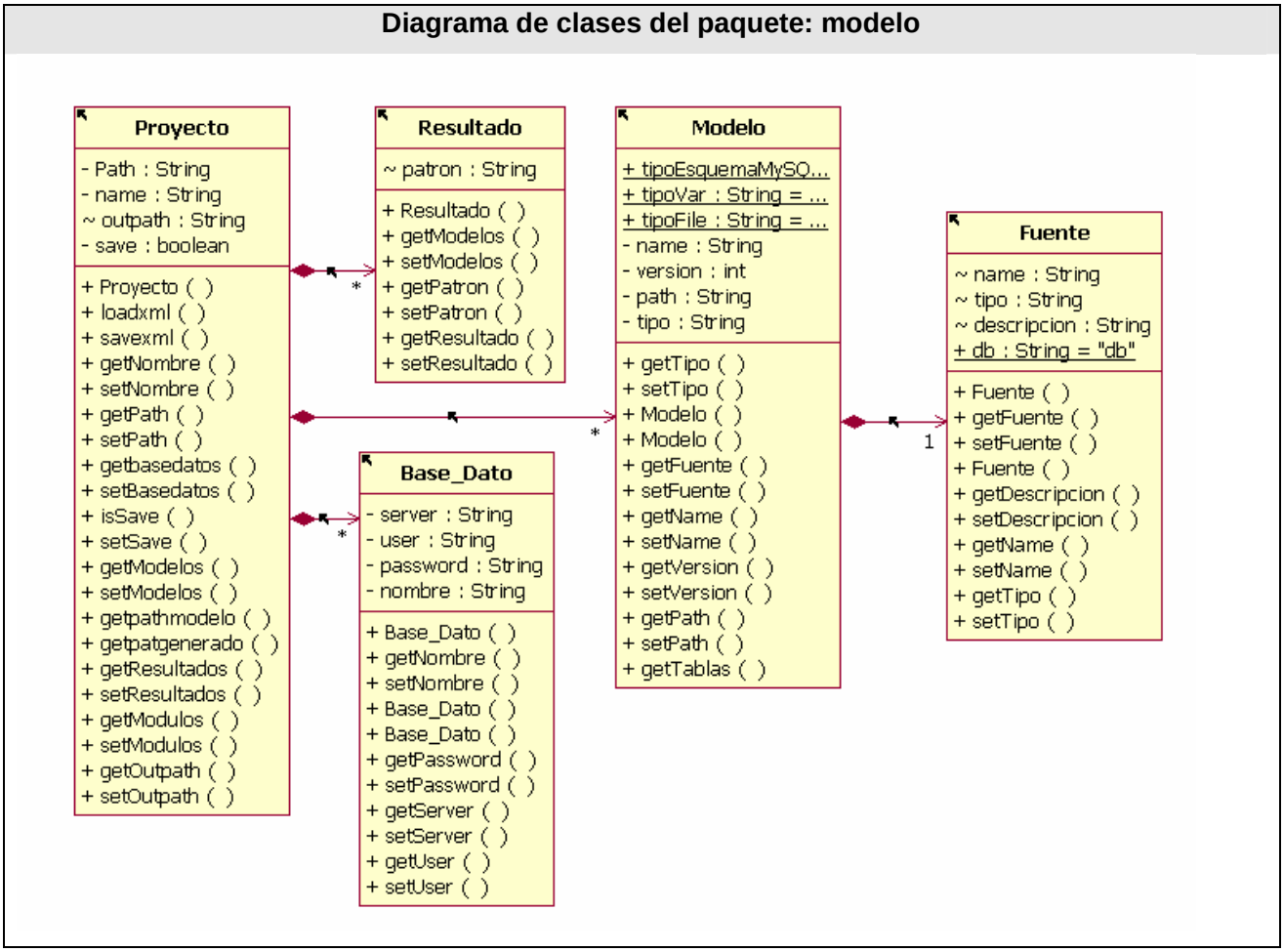

Fig 26. Diagrama de clases del paquete: modelo.

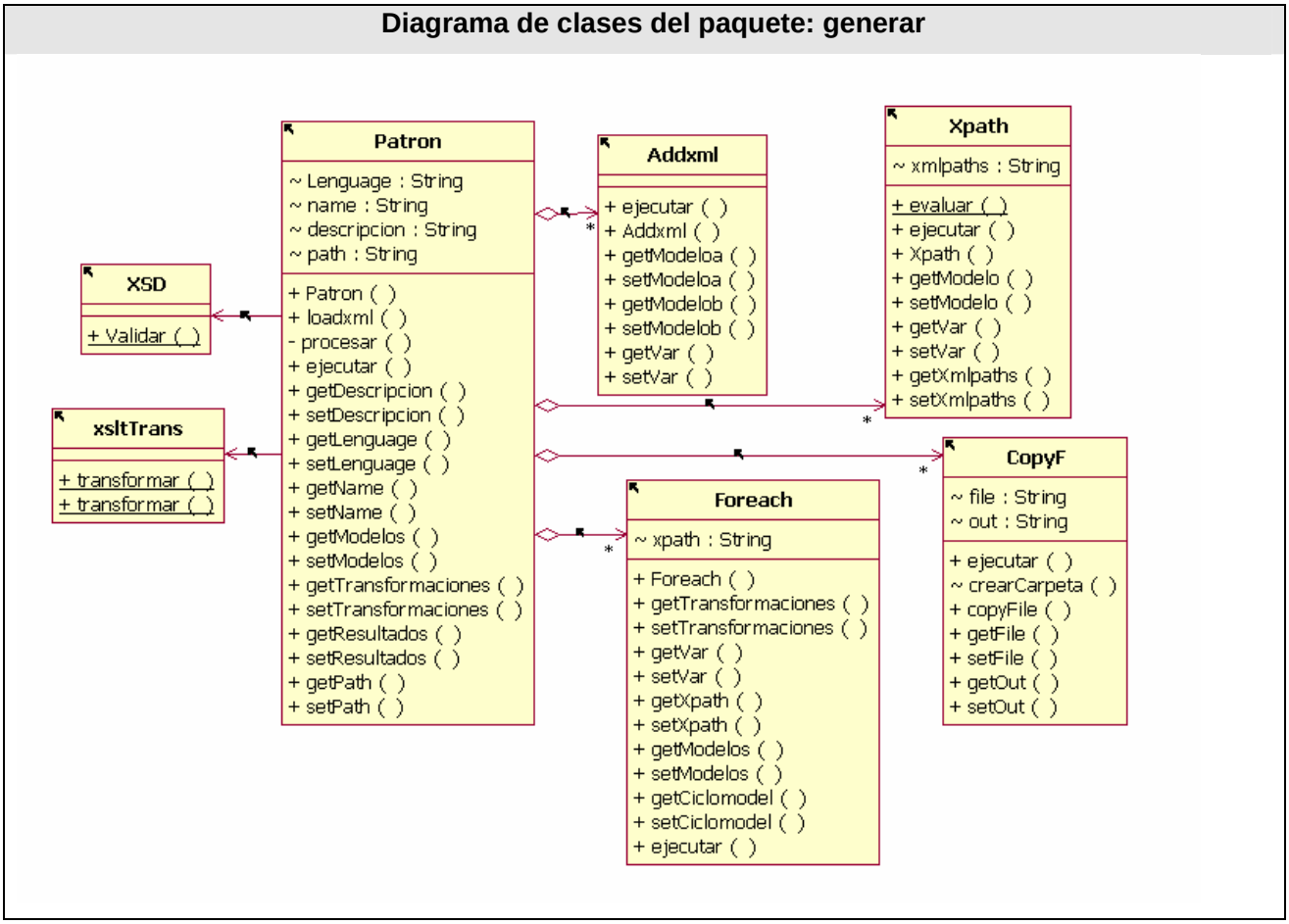

Fig 27. Diagrama de clases del paquete: generar.

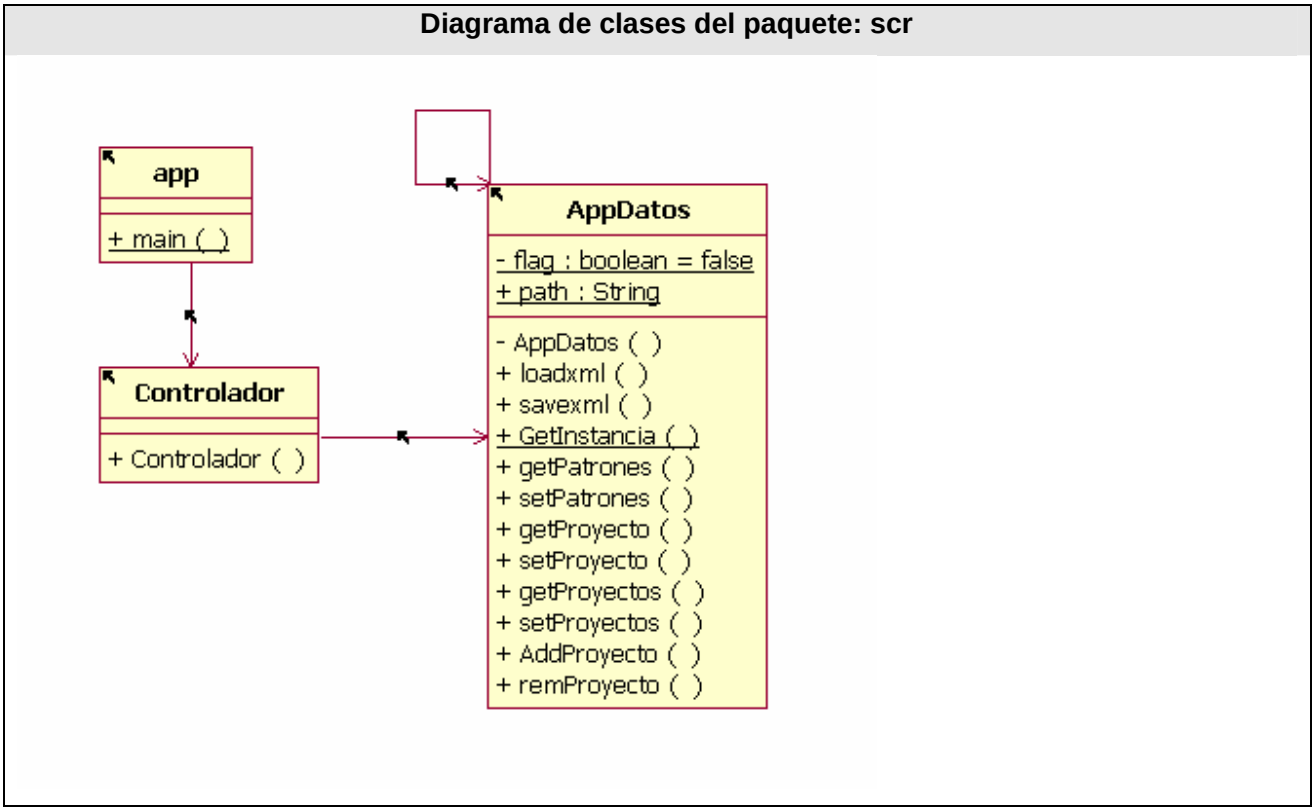

Fig 28. Diagrama de clases del paquete: scr.

Diagramas de interacción.

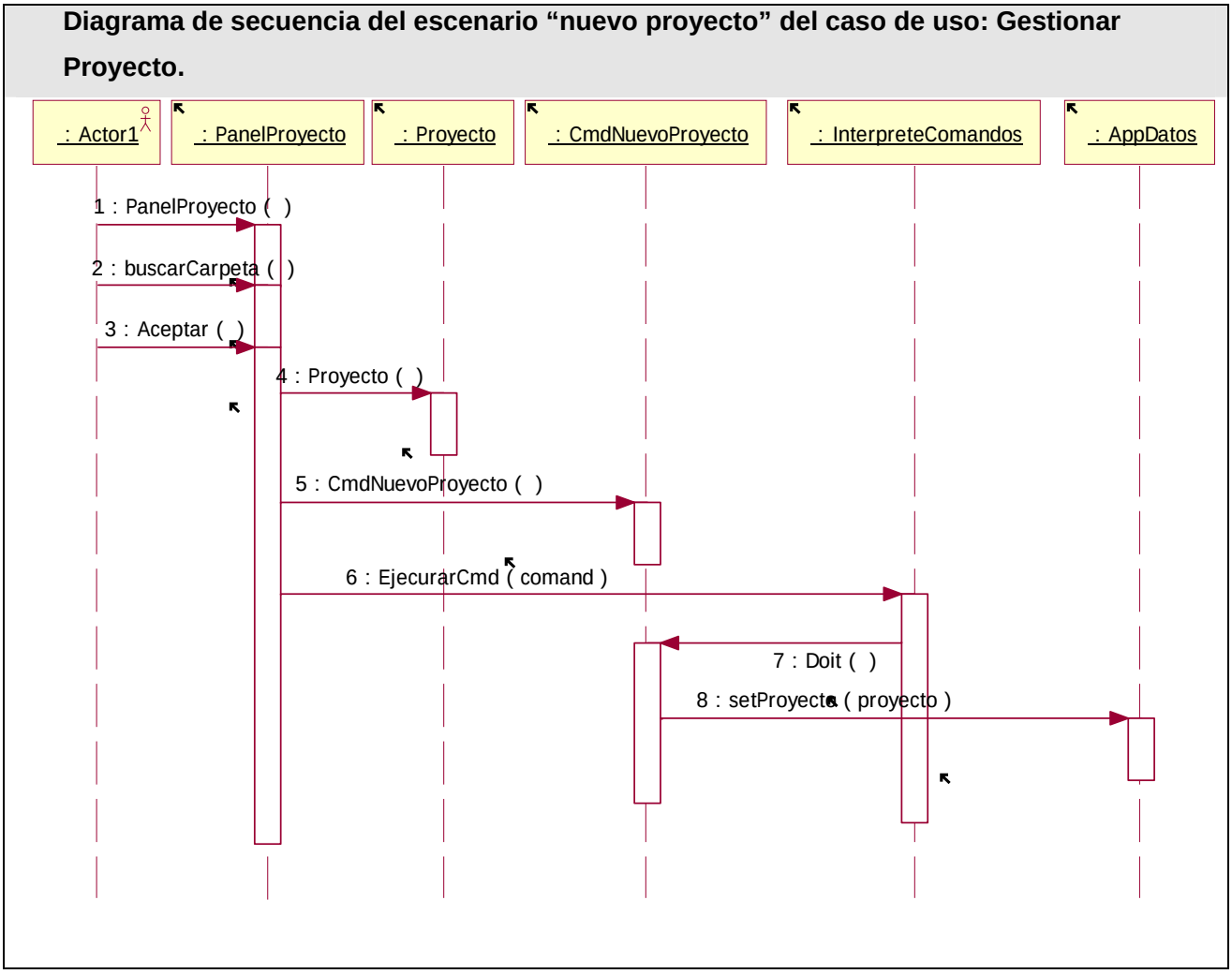

Fig 29. Diagrama de secuencia del escenario “nuevo proyecto” del caso de uso: Gestionar Proyecto.

Diagrama de secuencia del escenario “abrir proyecto” del caso de uso: Gestionar Proyecto.

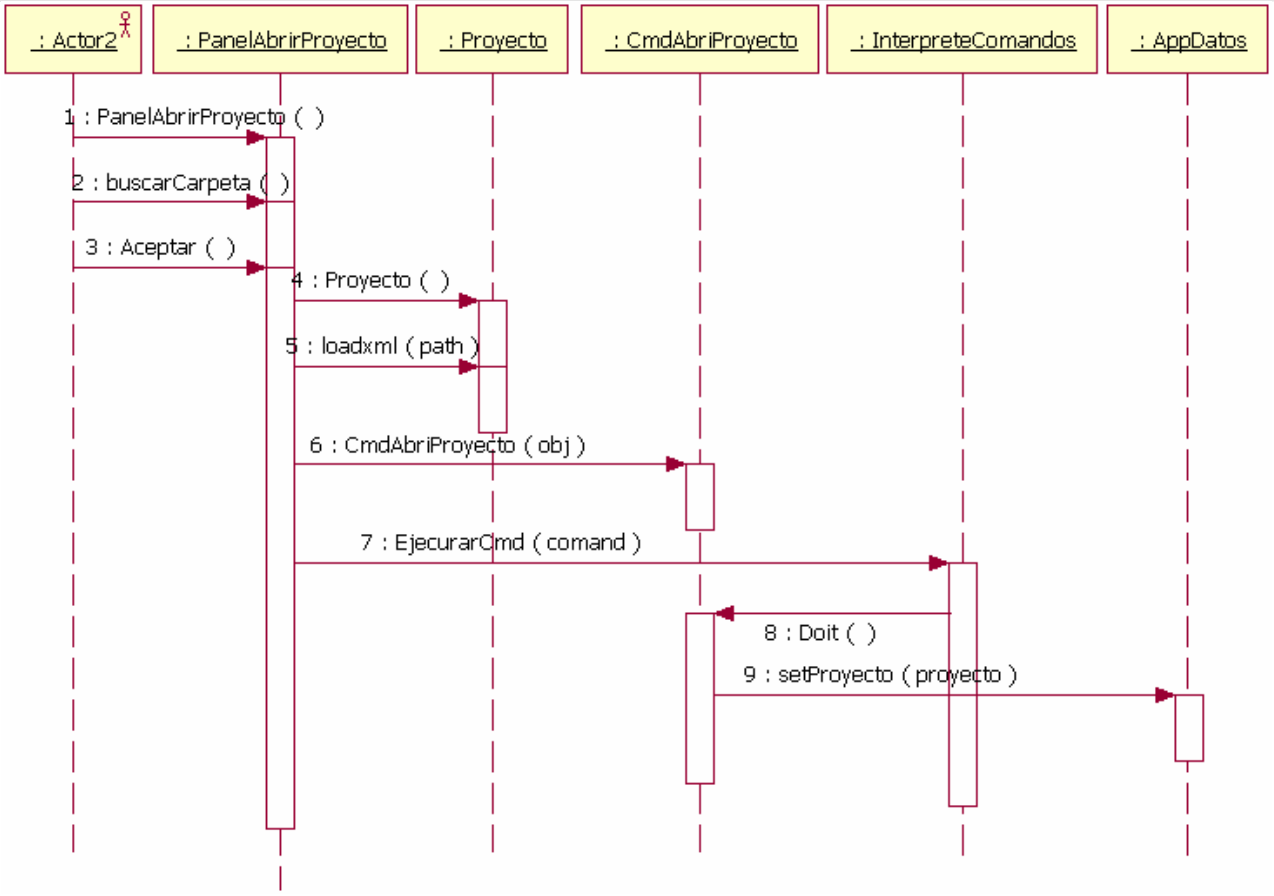

Fig 30. Diagrama de secuencia del escenario “abrir proyecto” del caso de uso: Gestionar Proyecto.

Diagrama de secuencia del escenario “guardar proyecto” del caso de uso: Gestionar Proyecto.

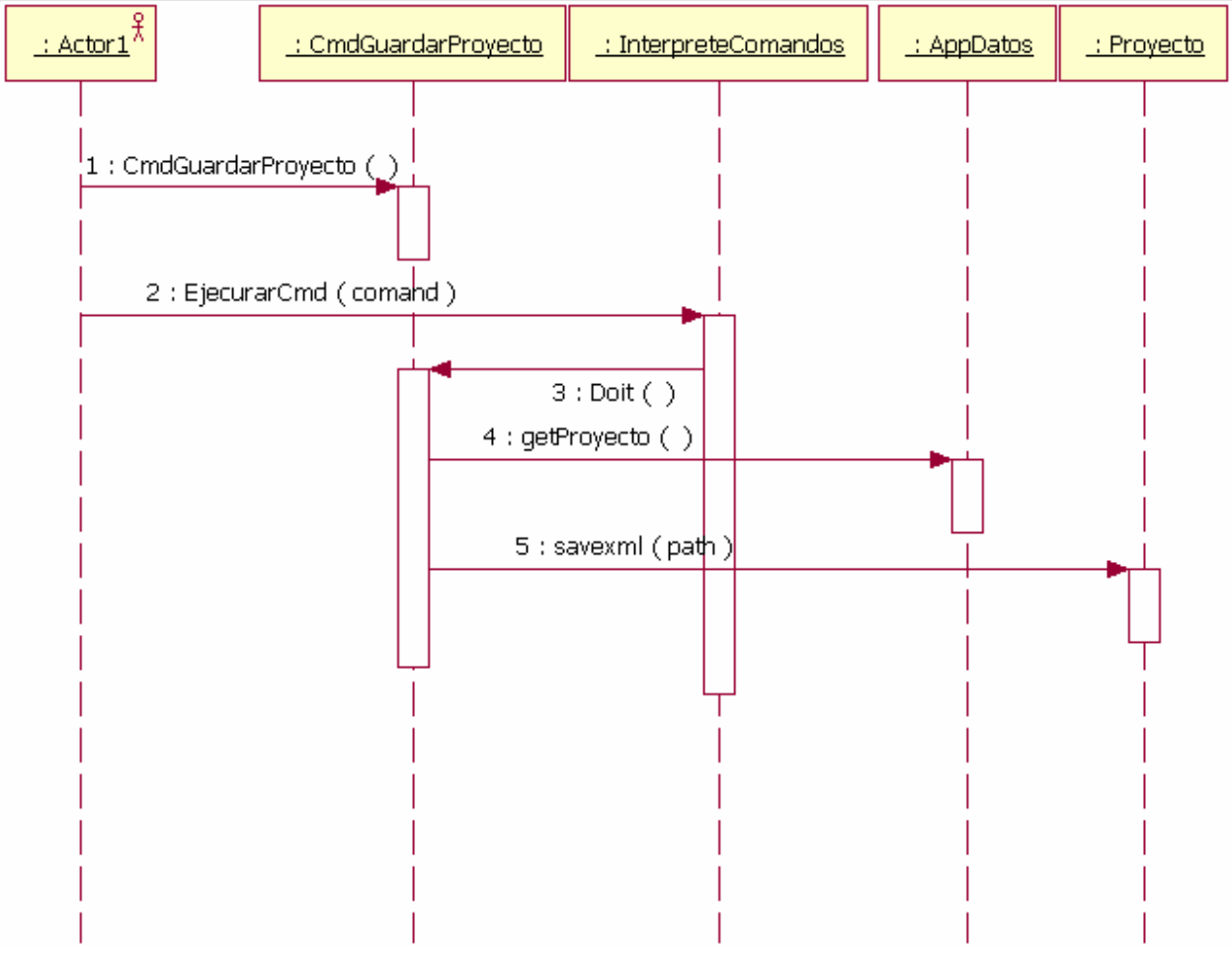

Fig 31. Diagrama de secuencia del escenario “guardar proyecto” del caso de uso: Gestionar Proyecto.

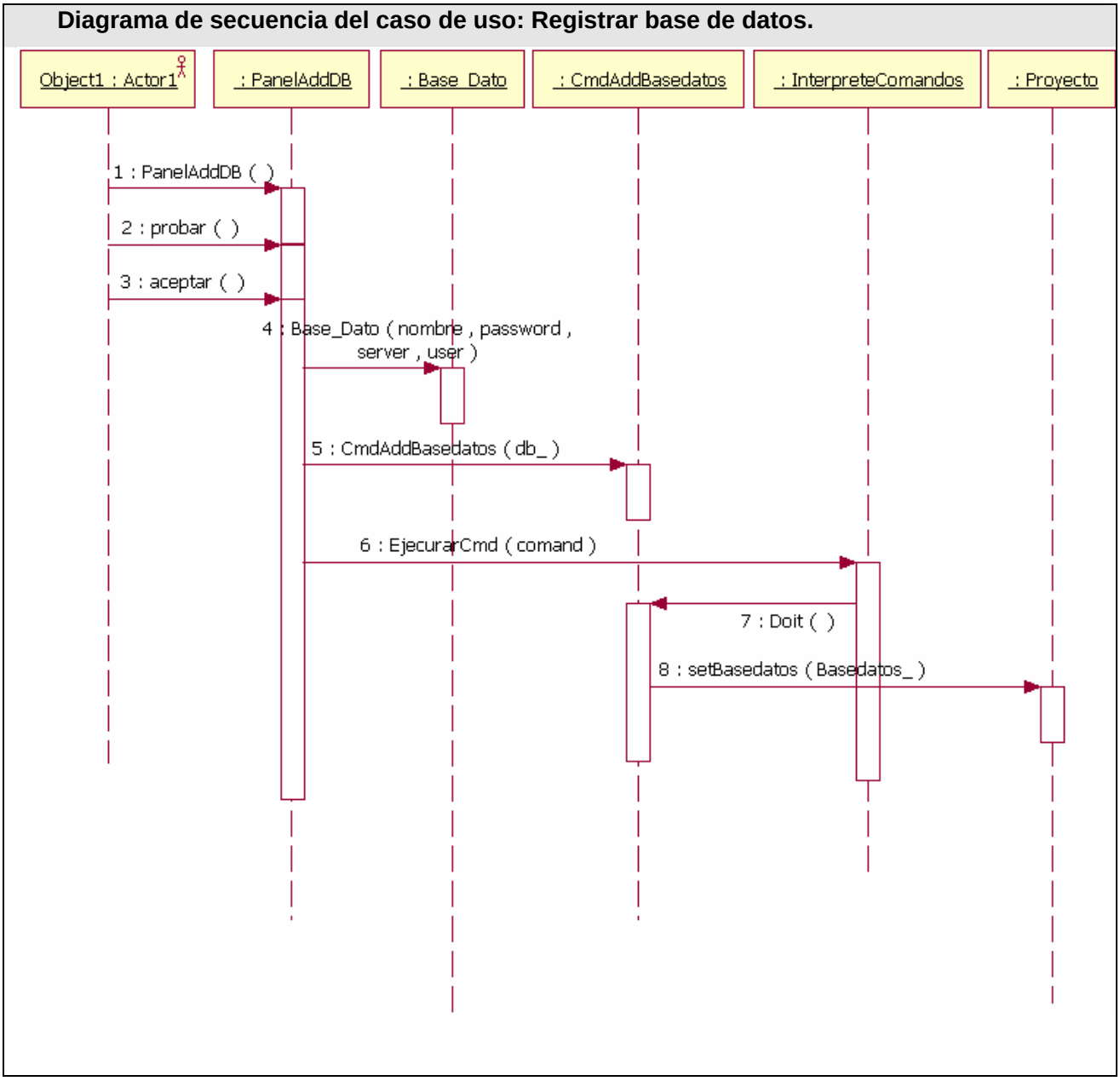

Fig 32. Diagrama de secuencia del caso de uso: Registrar base de datos.

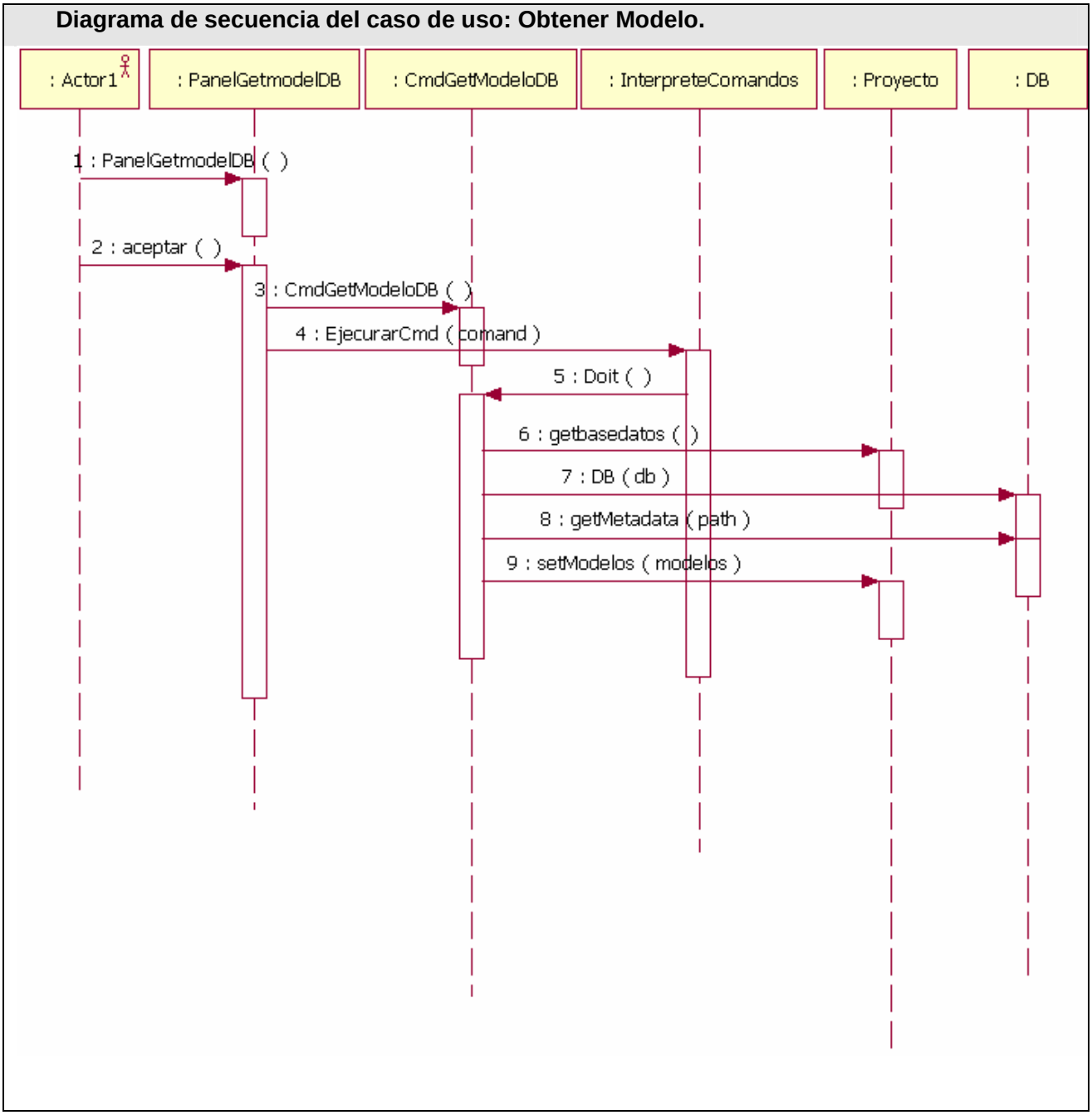

Fig 33. Diagrama de secuencia del caso de uso: Obtener Modelo.

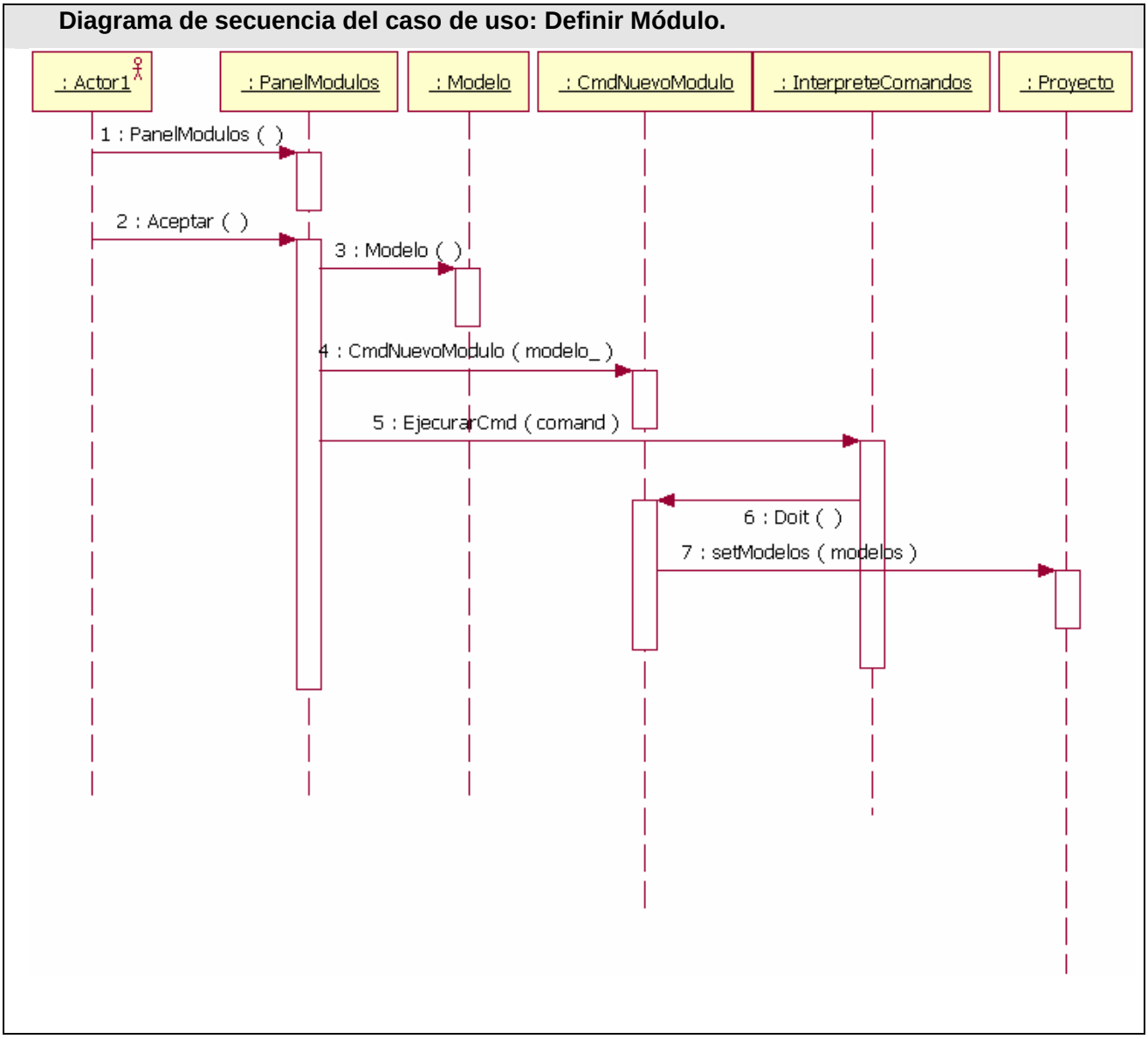

Fig 34. Diagrama de secuencia del caso de uso: Definir Módulo.

Diagrama de secuencia del escenario “agregar patrón” del caso de uso: Gestionar Patrones.

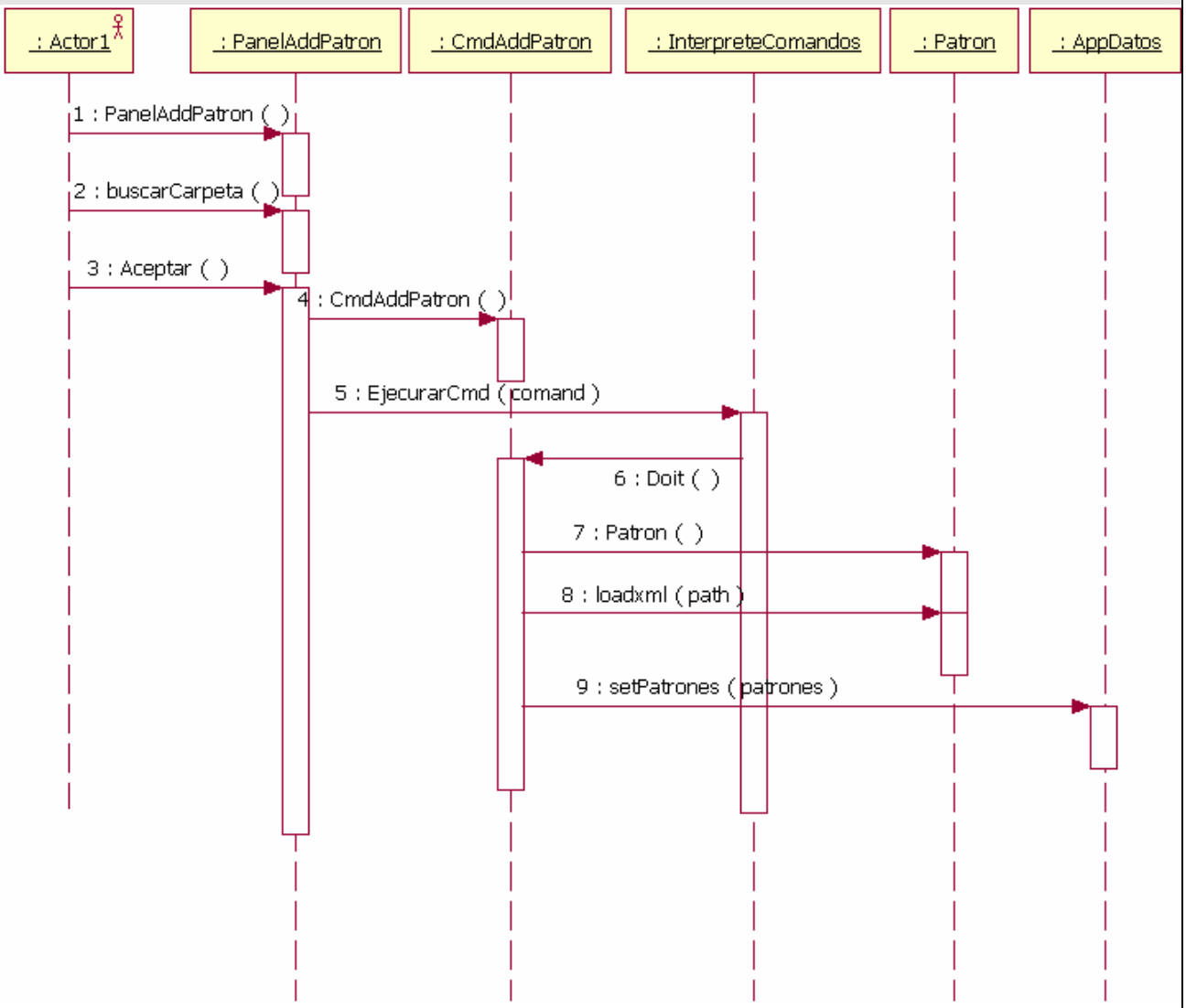

Fig 35. Diagrama de secuencia del escenario “agregar patrón” del caso de uso: Gestionar Patrones.

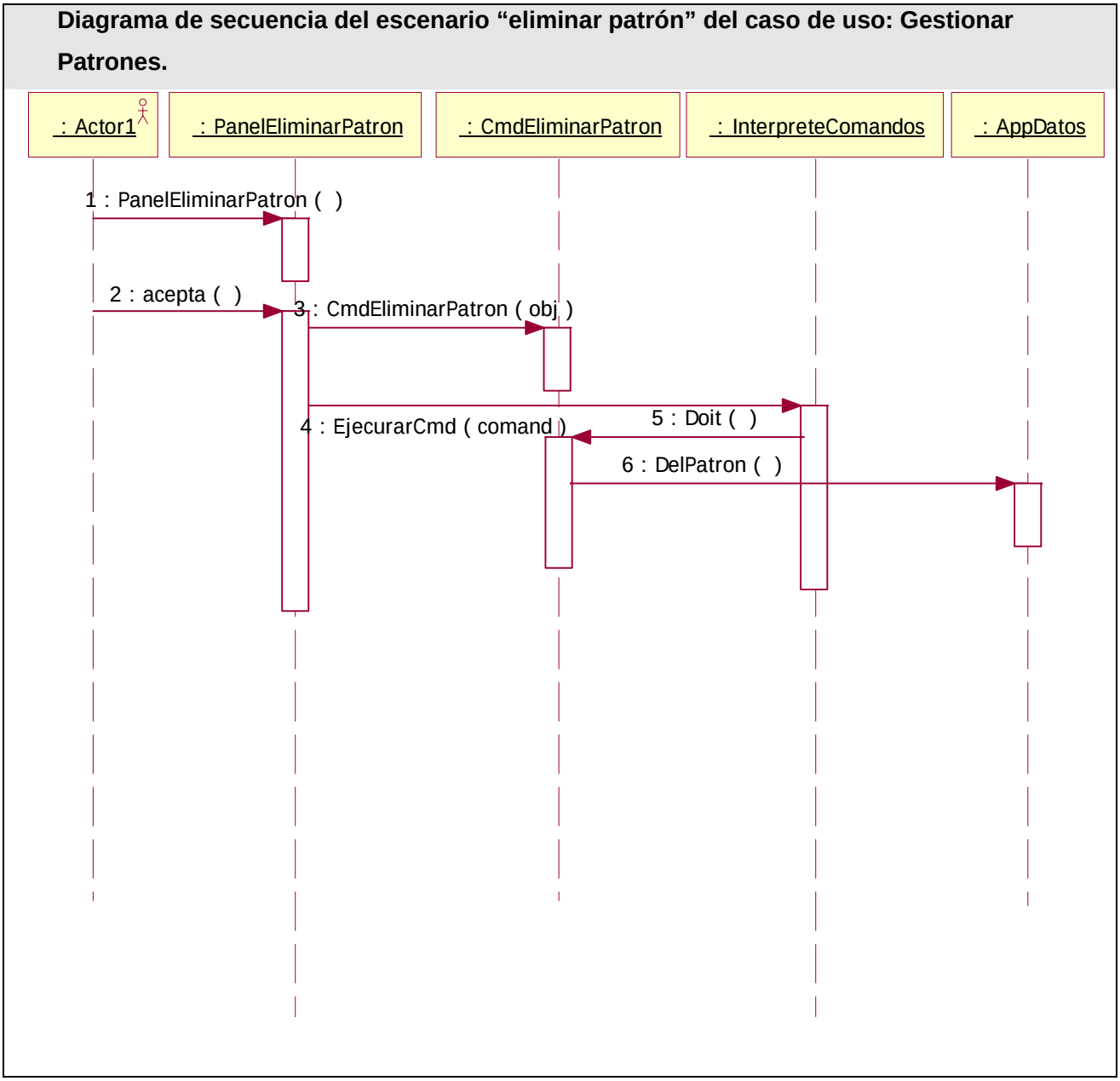

Fig 36. Diagrama de secuencia del escenario “eliminar patrón” del caso de uso: Gestionar Patrones.

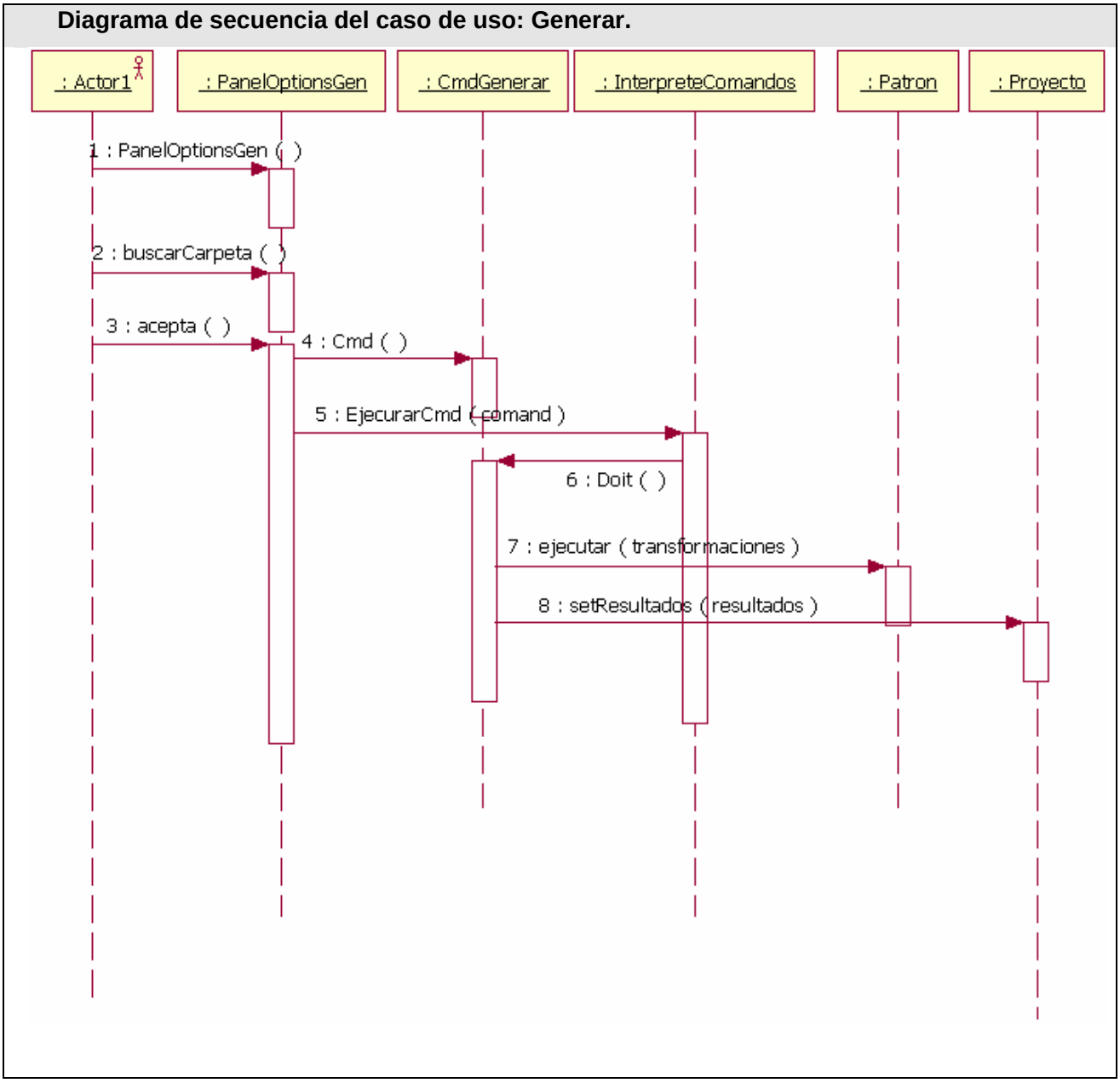

Fig 37. Diagrama de secuencia del caso de uso: Generar.

Descripción y diseño del código generado.

En la programación de aplicaciones WEB se hace conveniente dividir las aplicaciones por capas. Generalmente se divide en tres capas: la capa de presentación, que es la que se le muestra al usuario, la capa lógica del negocio, en la que se implementan todas las reglas del negocio y la capa de acceso a datos. La capa de acceso a datos contiene todas las funciones necesarias para acceder a la base de datos. Si se decide cambiar el sistema gestor de base de datos sólo hay que cambiar esta capa y la aplicación sigue funcionando tal y como era. El generador implementa esta capa en una clase que contiene las funciones más comunes para manejar la base de datos. [15]

Generación de la capa de Acceso a Datos.

Cuando se genera la capa de acceso a datos se crean tres archivos. El primero se llama Cad.PHP y contiene la definición de la clase Acceso_Datos. El segundo se llama config.PHP y contiene variables de configuración global como son: el nombre de la base de datos, el servidor, el usuario y la contraseña. El tercero se llama doc.html que contiene la documentación de todo el código generado. En la clase Acceso_Datos está definido para cada tabla funciones para insertar, actualizar, eliminar y seleccionar los datos de la misma. Si la tabla no contiene llave primaria se crearán sólo las funciones para insertar y seleccionar datos en ella. Para tratar los errores está definida una función que se llama on_error y recibe como argumento el mensaje de error. Esta función por defecto lo que hace es imprimir el mensaje de error pero el desarrollador la puede modificar según sus necesidades. La clase también cuenta con un constructor a través del cual se le pasa el nombre de la base de datos, el servidor, el usuario y la contraseña.

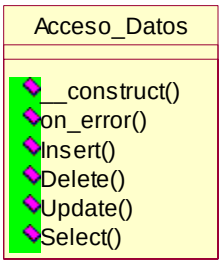

Fig 38. Clase Acceso_Datos.

Para el buen uso de esta clase se recomienda heredar de ella, de forma tal que las nuevas operaciones que definan los desarrolladores se vean afectadas en caso de que sea necesario volver a generar esta clase. A continuación se muestra un diagrama de cómo se recomienda emplear esta clase.

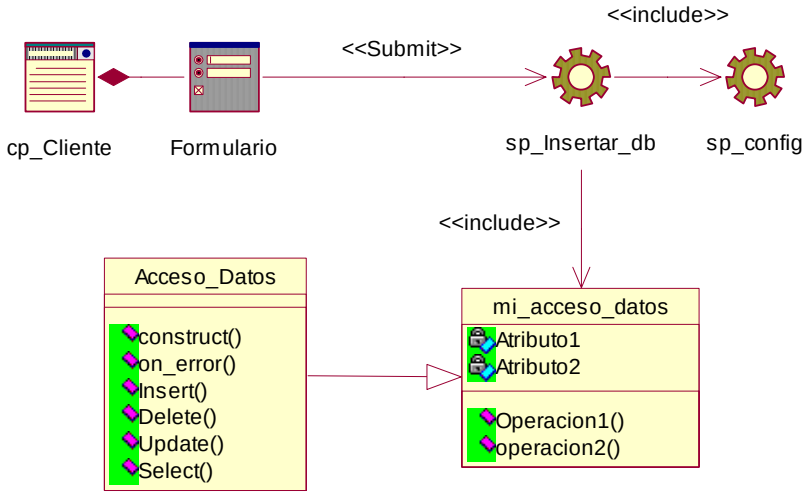

Fig 39. Diagrama de clases del código generado.

Generación de interfaces.

En la generación las interfaces, por cada tabla de la base de datos, se crean seis archivos. El primero de ellos es el archivo Insertar_Nombre_Tabla.html el cual contiene un formulario con los campos necesarios para insertar en la tabla. Este formulario le envía los datos usando el método POST a otro archivo llamado Insertar_ Nombre_Tabla.PHP el cual crea una instancia de un objeto de la clase Acceso_Datos y ejecuta el método Insertar_ Nombre_Tabla() insertando los datos en la base de datos. Otro de los archivos que se genera es el archivo Seleccionar_ Nombre_Tabla.PHP el cual haciendo uso de la clase Acceso_Datos selecciona los datos de una tabla y los muestra en la interfaz. Se crea también el archivo Modificar_ Nombre_Tabla.PHP el cual a partir de las llaves primarias de una tabla carga el contenido de una fila en un formulario para ser modificado. Este formulario manda los datos al archivo Modificar_bd_codigo_ Nombre_Tabla.PHP el cual se encarga de actualizar la fila en la base de datos. El último archivo que se crea es: Eliminar_ Nombre_Tabla.PHP el cual a partir del la llave primaria de una tabla elimina una fila de la misma. Además de estos archivos se crea el archivo index.PHP el cual contiene un menú para mostrar el contenido de cada una de las tablas.

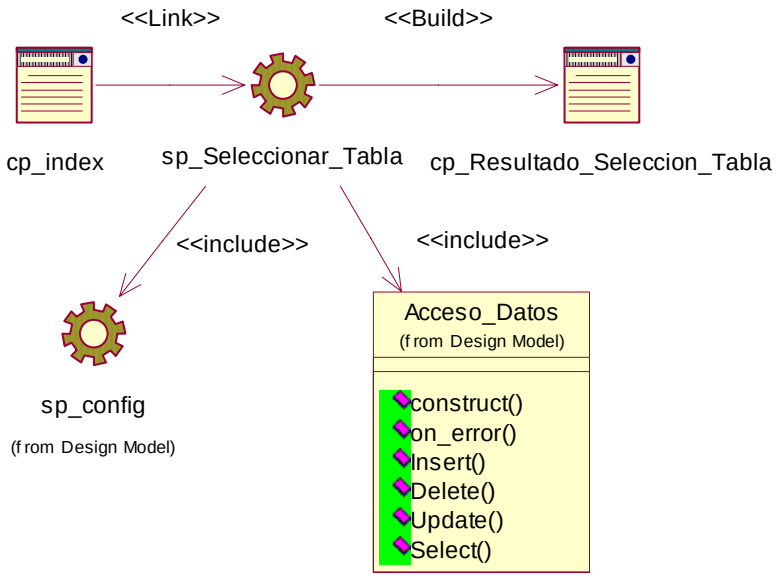

Fig 40. Diagrama de clases propuesto para seleccionar una tabla.

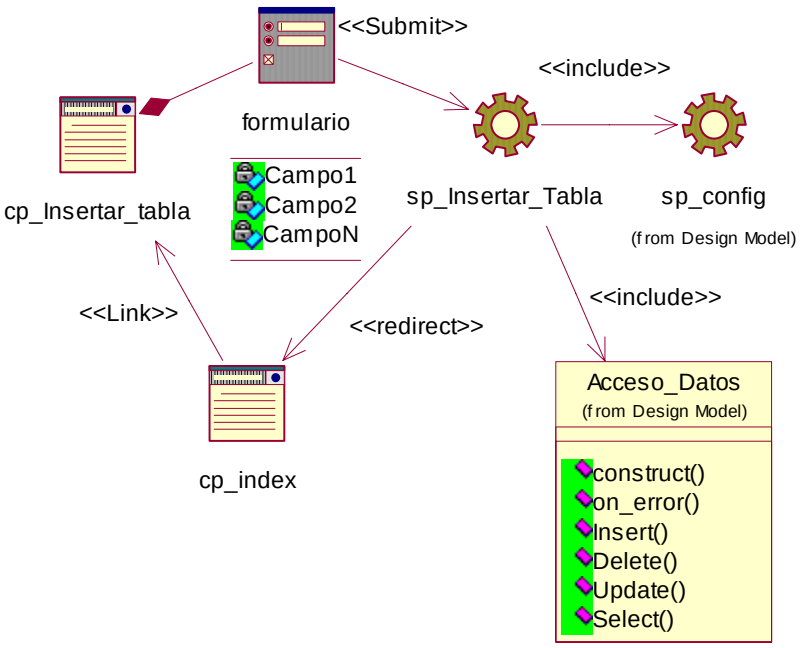

Fig 41. Diagrama de clases propuesto para Insertar datos en una tabla.

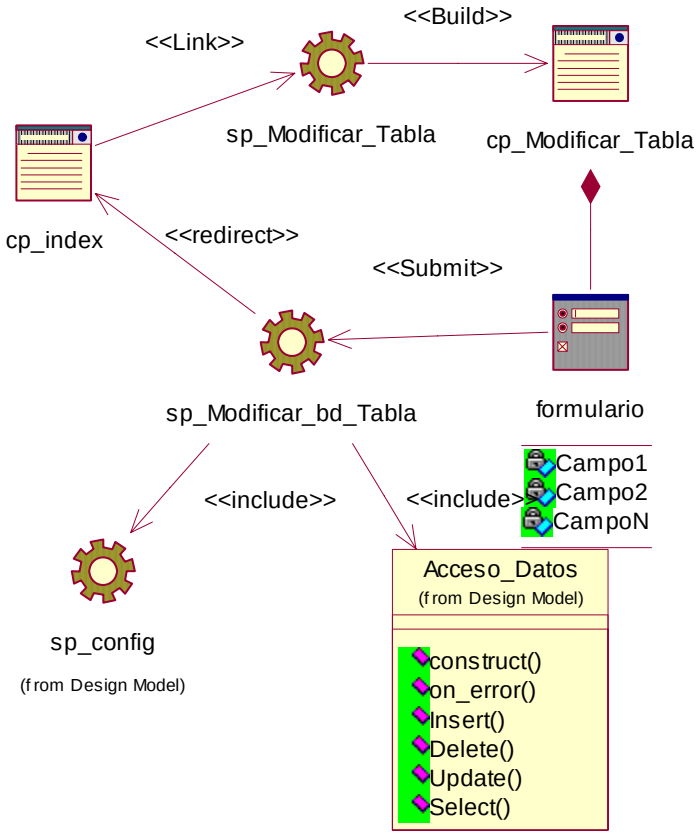

Fig 42. Diagrama de clases propuesto para modificar datos de una tabla.

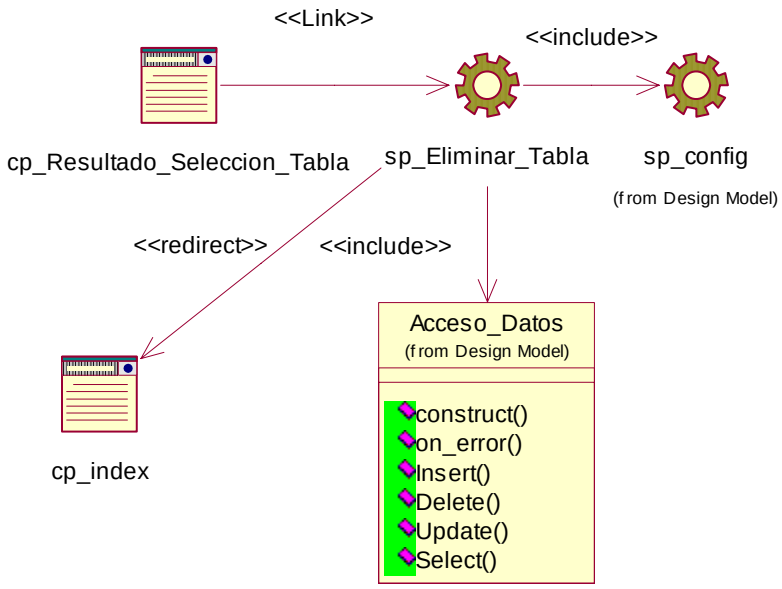

Fig 43. Diagrama de clases propuesto para eliminar datos de una tabla.

Generación de servicios WEB

En la generación de los servicios WEB se crean 6 archivos. El archivo wsdl.wsdl que contiene la descripción del servicio WEB. Para poder usar este código correctamente hay que modificar este archivo y poner en la etiqueta (<soap:address location="http://localhost/ WSDL/server.PHP" />) la dirección en la que se encuentra el archivo server.PHP. Se crea también el archivo server.PHP que es el que procesará todas las peticiones como servidor del servicio WEB. El archivo function.PHP que contiene la definición de todas las funciones que serán exportadas por el servicio WEB. El archivo config.PHP que contiene la declaración de las variables globales para conectarse a la base de datos. El archivo wsdl_clien.PHP el cual sirve de interfaz cliente para el servicio WEB. El código que se genera necesita para su funcionamiento que se active la extensión del servidor PHP_soap. A continuación se muestra el diagrama de aplicación del servicio WEB.

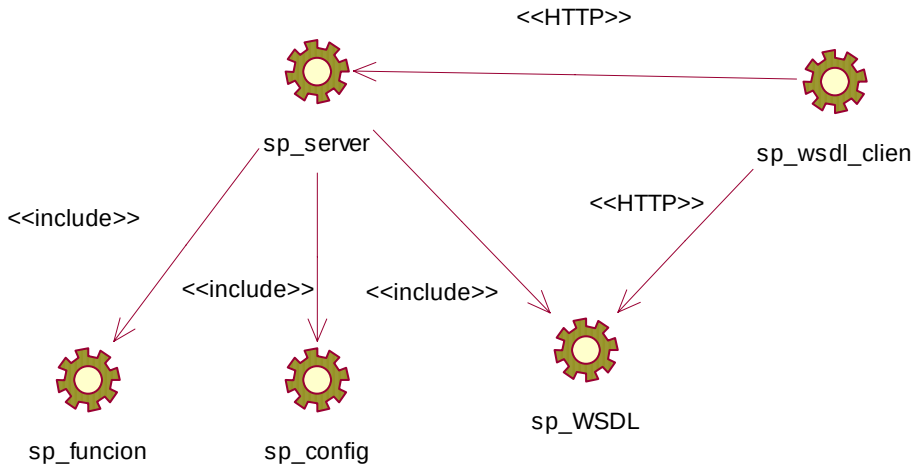

Fig 44. Diagrama de aplicación del servicio WEB.

Diagrama de despliegue.

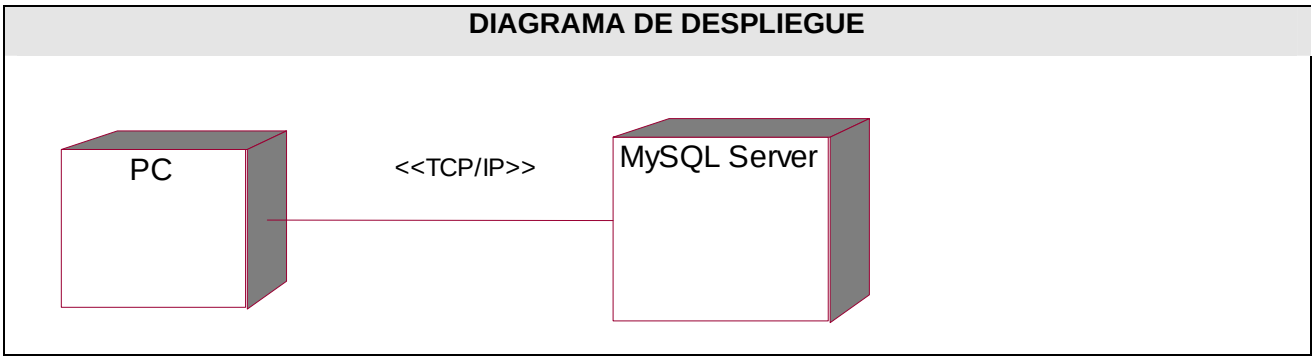

Fig 45. Diagrama de despliegue.

Conclusiones

Con la creación de este modelo de diseño se cuenta con la base necesaria para pasar al flujo de trabajo de implementación. Es oportuno mencionar que de la calidad del diseño realizado va a depender el éxito en la implementación. El uso de patrones de diseño ayudó a aplicar soluciones estandarizadas a problemas típicos del diseño aumentado así la calidad del Diseño.

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA

Diagramas de componentes.

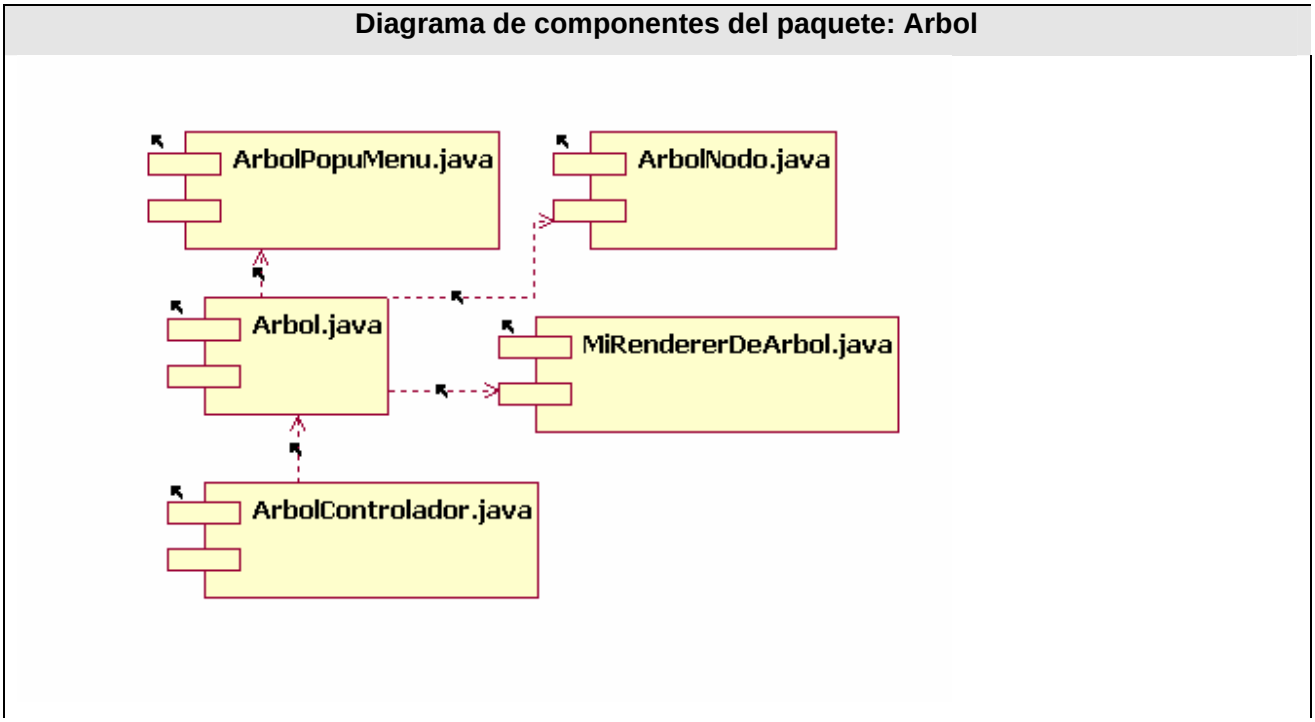

Fig 46. Diagrama de componentes del paquete: Arbol.

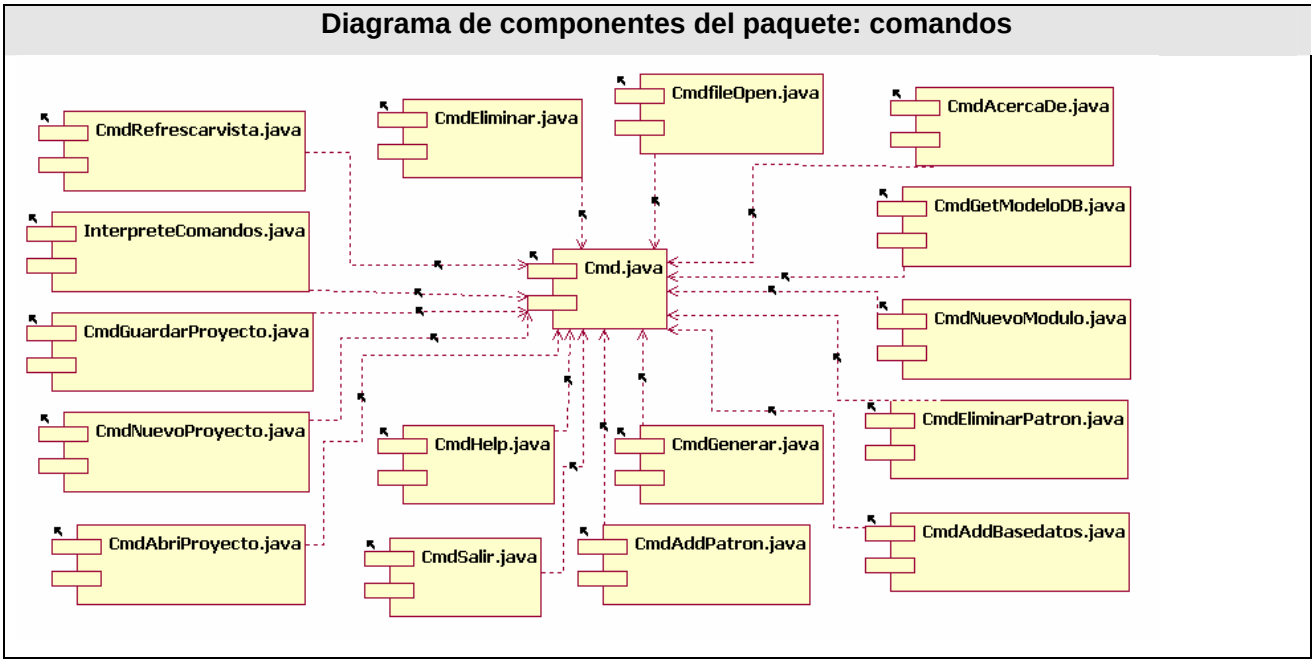

Fig 47. Diagrama de componentes del paquete: comandos.

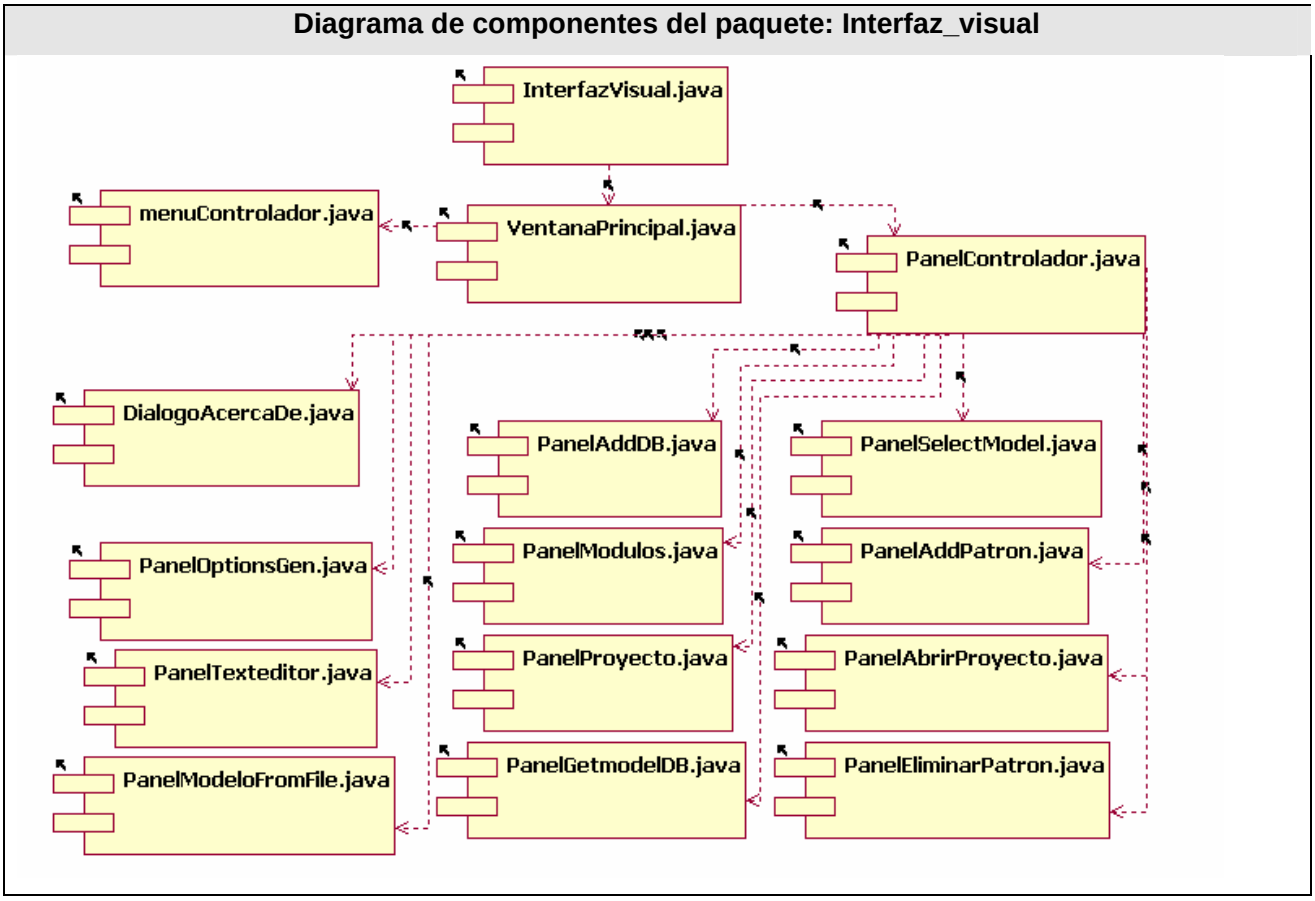

Fig 48. Diagrama de componentes del paquete: Interfaz_visual.

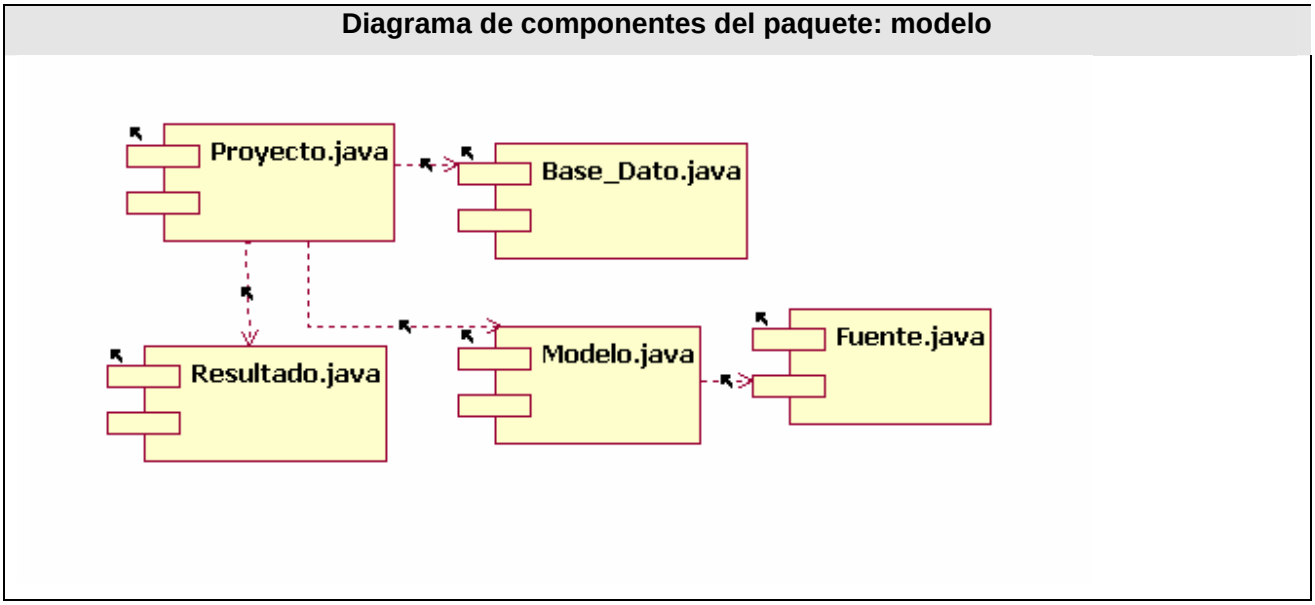

Fig 49. Diagrama de componentes del paquete: modelo.

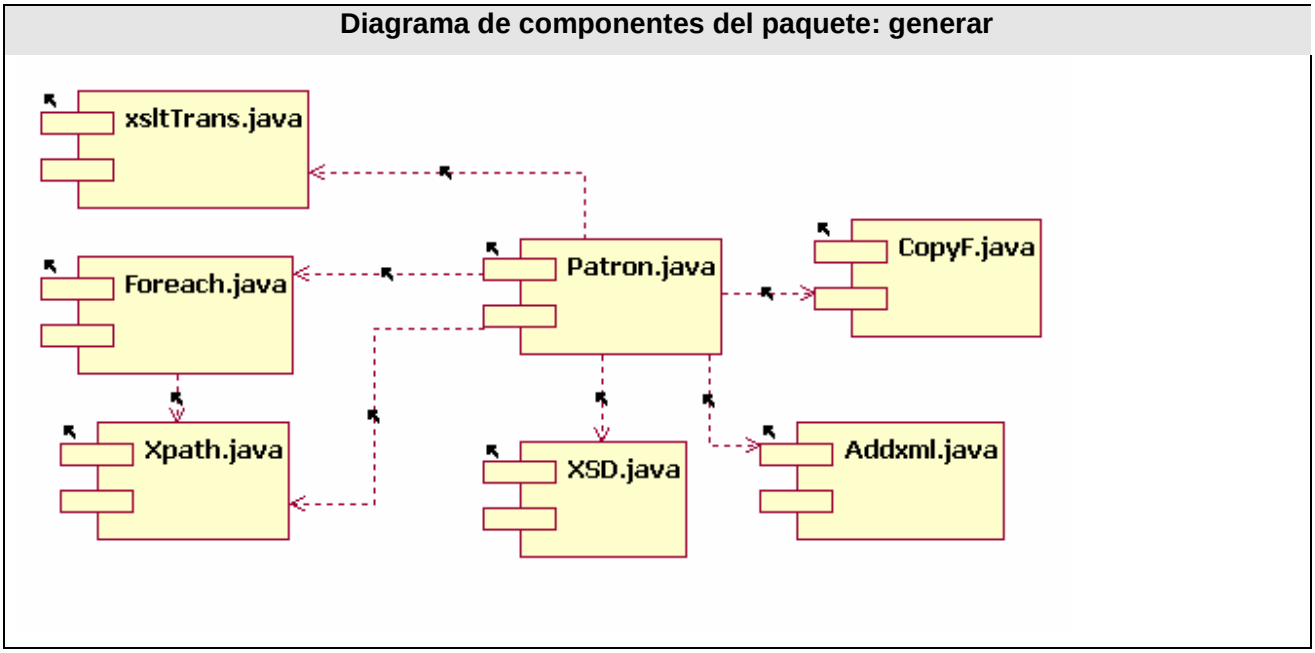

Fig 50. Diagrama de componentes del paquete: generar.

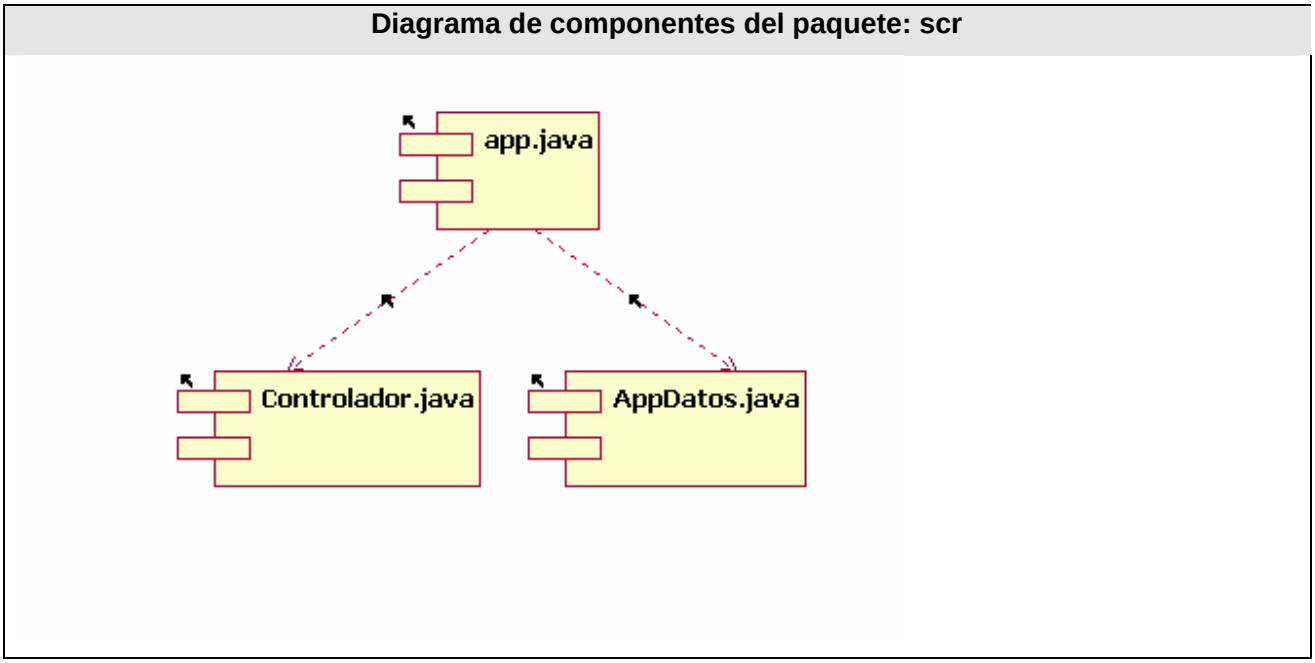

Fig 51. Diagrama de componentes del paquete: scr.

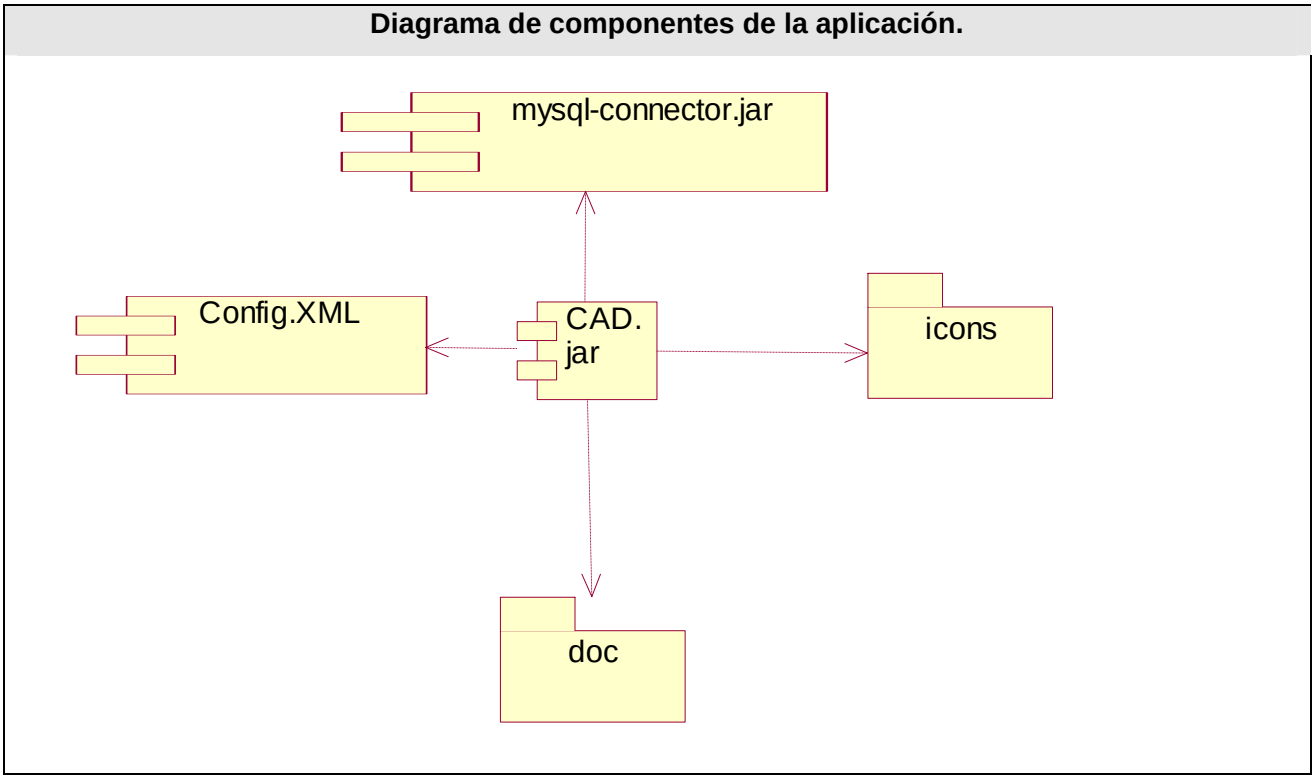

Fig 52. Diagrama de componentes de la aplicación.

Modelo de prueba

Modelo de prueba.

En la creación de software se requiere gran esfuerzo mental por parte de los desarrolladores por lo que la posibilidad de cometer errores es muy alta. No se puede asegurar que un producto es ciento por ciento fiable ni que cumplirán al máximo con las expectativas de los clientes. El método con que se cuenta para garantizar la calidad y el buen funcionamiento de un producto es la realización de pruebas. Las pruebas no confirman la ausencia de errores en el software, solo brindan una medida de cómo responderá el mismo ante algunas situaciones determinadas. [25]

Al sistema se le aplicaron pruebas de aceptación con el objetivo de verificar las funcionalidades del sistema. Para esto se utilizo el método de caja negra usando la técnica de partición de equivalencia. A continuación se muestran los casos de prueba empleados.

Casos de prueba Crear nuevo proyecto		
Entrada	**Resultados**	**Condiciones**
Nombre: Proy_1 Dirección destino : C:\proyectos	En la carpeta C:\proyectos se crea un archivo XML para almacenar la información del proyecto.	No debe existir un proyecto con ese nombre y la dirección de la carpeta debe ser valida.
Nombre: Proy_1 Dirección destino : k:\\proyectos	El sistema muestra un mensaje de error porque la dirección de la carpeta no es validad.	No es válida la dirección k:\\proyectos.
Nombre: Proy_1 Dirección destino : C:\proyectos_2	El sistema muestra un mensaje de error porque ya existe un proyecto con ese nombre.	Que exista un proyecto con el nombre Proy_1.

Tabla 11: Casos de prueba Crear nuevo proyecto.

Casos de prueba abrir proyecto		
Entrada	**Resultados**	**Condiciones**
Nombre: Proy_1 Dirección : C:\proyectos	Se habré el proyecto y se muestran los modelos y los resultados que le pertenecen.	Que exista un proyecto de nombre Proy_1 en la carpeta C:\proyectos
Nombre: Proy_5 Dirección : C:\proyectos	Se muestra un mensaje de error por no poder abrir el proyecto.	Que no exista el proyecto Proy_5 en la carpeta C:\proyectos

Tabla 12: Casos de prueba abrir proyecto

Casos de prueba guardar proyecto		
Entrada	**Resultados**	**Condiciones**
Se selecciona guardar proyecto.	Se guarda el proyecto.	Debe haber un proyecto abierto.

Tabla 13: Casos de prueba guardar proyecto

Casos de prueba registrar base de datos		
Entrada	**Resultados**	**Condiciones**
Servidor: db_server Base de datos: db_prueba Usuario: root Contraseña: root	Se registra esta base de datos en el proyecto.	Debe haberse creado un proyecto y existir la base de datos db_prueba en el servidor db_server con el usuario root y la contraseña root.
Servidor: db_server Base de datos: db_otra Usuario: root Contraseña: root	Se muestra un mensaje de error de conexión a la base de datos	No debe existir una base de datos con el nombre db_otra

Tabla 14: Casos de prueba registrar base de datos.

Casos de prueba Obtener Modelo		
Entrada	**Resultados**	**Condiciones**
Nombre: Modelo_1 Base de datos: db_prueba	Se obtiene el modelo de la base de datos en un archivo XML	Debe estar registrada la base de datos db_prueba en el proyecto actual.

Tabla 15: Casos de prueba Obtener Modelo.

Casos de prueba Definir Módulo		
Entrada	**Resultados**	**Condiciones**
Nombre: Modulo_1 Tablas: tabla_1, tabla 2	Se crea un modulo que contiene la tabla tabla_1 y la tabla_2	Que la base de datos registrada contenga la tabla tabla_1 y la tabla_2.

Tabla 16: Casos de prueba Definir Módulo.

Casos de prueba Agregar Patrón		
Entrada	**Resultados**	**Condiciones**
Nombre: CAD Archivo: C:\proyectos\patrones\ Patron_CAD.xml	Se adiciona este patrón a la aplicación.	Que el patrón no aya sido adicionado anteriormente. Y que se encuentre almacenado en C:\proyectos\patrones\ Patron_CAD.xml

Tabla 17: Casos de prueba Agregar Patrón.

Casos de prueba Eliminar Patrón		
Entrada	**Resultados**	**Condiciones**
Nombre: CAD	Se elimina el patrón	Que exista un patrón con nombre CAD

Tabla 18: Casos de prueba Eliminar Patrón.

Casos de prueba Generar		
Entrada	**Resultados**	**Condiciones**
Patrón: CAD Modelo: Modelo_1	Se crea la capa de acceso a datos	Que exista el patrón CAD y el modelo Modelo_1

Tabla 19: Casos de prueba Generar.

CONCLUSIONES

La presente investigación cumplió con los objetivos propuestos obteniéndose los siguientes resultados:

- Se realizó un levantamiento de los requisitos que debe cumplir una herramienta de generaron de código.
- Se diseñó una herramienta de generación de código que cumple con los requisitos propuestos.
- Se implementó una herramienta de generación de código que cumple con las funcionalidades propuestas, la cual se encuentra en la fase de prueba.
- La herramienta es útil y fácil de usar, lo cual ha propiciado el uso de la misma en proyectos productivos y en la docencia.
- Se demostró de forma práctica el potencial que brinda el uso de XSLT en la generación de código.
- Se desarrollaron cuatro patrones de generación de código los cuales satisfacen en gran medida las expectativas de los desarrolladores de aplicaciones WEB en PHP.
- Se cuenta con toda la documentación del proyecto que permite que futuros desarrolladores continúen mejorando el generador.

Este trabajo se presentó en la Jornada Científica Estudiantil a nivel de Facultad y Universidad, obteniendo mención en la comisión de programación a nivel de Universidad. El trabajo realizado constituye un punto de partida más sólido para aquellos que deseen desarrollar una herramienta de este tipo en la Universidad de Ciencias Informáticas, la cual por sus características es un lugar propicio para el diseño y construcción de herramientas de desarrollo de software.

RECOMENDACIONES

- Se debe trabajar en la construcción de una herramienta que facilite la creación y edición de patrones de generación con el objetivo ampliar el uso de esta herramienta a otros lenguajes de programación.
- Se deben crear herramientas que obtengan modelos en XML de otros gestores de bases de datos así como de estructuras de archivos de texto plano para poder ampliar el campo de aplicación del generador.
- Se debe trabajar en la posibilidad de integrar la herramienta a varios IDE de desarrollo.
- Se debe fomentar una comunidad de usuarios de este tipo de herramientas para obtener retroalimentación acerca de su utilidad.
- Se debe impartir cursos de capacitación a los desarrolladores para que introduzcan más eficazmente esta herramienta en el desarrollo de sus proyectos.

Referencias Bibliográficas

1. HERRINGTON, J. *Code Generation In Action*. Manning, 2006.
2. Jacobson, I.; Booch, G. y Rumbaugh, J.; "*El Proceso Unificado de Desarrollo de software*". 2000. Addison-Wesley. ISBN 84-7829-036-2
3. Software Acumen Limited, *Code Generator Models*, [en línea], febrero del 2005[Consultado en enero del 2007], Disponible en http://www.codegeneration.net/tiki-index.PHP?page=ModelsIntroduction
4. MORENO, P. J. M., Ed. *Especificación de interfaz de usuario: De los requisitos a la generación de código*Valencia, Universidad de Valencia, 2003.
5. Dondo, Agustín,"*Por qué elegir PHP*" [en línea], Programación en castellano, 2006, [consultado enero 2007] Disponible en : http://www.programacion.net/PHP/articulo/porquePHP/
6. Eck J. David , "*Introduction to Programming Using JAVA*" [en línea],2006 [consultado febrero 2007] Disponible en : http://math.hws.edu/JAVAnotes/
7. Obasanjo,Dare "*Descripción de XML*" [en línea], Microsoft Corporation, 2005 [consultado enero 2007] Disponible en : http://www.microsoft.com/spanish/msdn/articulos/archivo/011003/voices/understxml.asp
8. W3C Communications Team, "*XML in 10 points*", World Wide WEB Consortium,2003, [consultado enero 2007] Disponible en : http://www.w3.org/XML/1999/XML-in-10-points.html
9. Thompson S. Henry; Beech, David ; Maloney, Murray; Mendelsohn, Noah,"*XML Schema*", World Wide WEB Consortium, 2004 [consultado enero 2007] Disponible en : http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/
10. Clark, James," *XSL Transformations*", World Wide WEB Consortium, 1999, [consultado enero 2007] , Disponible en : http://www.w3.org/TR/xslt
11. The Apache Software Foundation, "*How it works*", 2007 [consultado febrero 2007] Disponible en : http://www.apache.org/foundation/how-it-works.html
12. Bath, England. "*WEB Server Survey*", [en línea] [consultado febrero 2007] Disponible en : http://news.netcraft.com/archives/2007/05/01/may_2007_WEB_server_survey.html
13. Russell J.T. Dyer, "*MYSQL Tutorial*" [en línea], 2003, [consultado febrero 2007] Disponible en : http://www.unixreview.com/documents/s=8989/ur0409f/

14. IBM Corporation, "*IBM Rational Rose*" [en línea], 2005, [consultado enero 2007] Disponible en : http://www-306.ibm.com/software/rational/
15. Martín Moreno, Tomás, "*Programación en 3 capas*",[en línea], 2005, [consultado enero 2007] , Disponible en : http://www.elguille.info/colabora/NET2005/Tomasmm_3Capas.htm
16. Kurniawan, Budi , "*Using the Singleton Pattern*" , 2004, 2007 [consultado enero 2007] Disponible en : http://www.onJAVA.com/pub/a/onJAVA/2003/08/27/singleton.html
17. Lasater, Chris," *Design Patterns*" , Wordware Publishing Inc. [en línea], 2006 [consultado enero 2007] Disponible en : http://www.codeproject.com/books/DesignPatterns.asp
18. Alachisoft, "*Rapidly Develop .NET Applications with O/R Mapping*", [en línea], 2007 [consultado febrero 2007] Disponible en: http://www.alachisoft.com/tdev/index.html
19. YesSoftware, "*CodeCharge Studio Feature List*", [en línea], 2006 [consultado enero 2007] Disponible en : http://www.yessoftware.com/products/product_detail.PHP?product_id=1
20. Visual Paradigm, "*10 Reasons to Choose Visual Paradigm*", [en línea], 2006 [consultado enero 2007] Disponible en : http://www.visual-paradigm.com/aboutus/10reasons.jsp
21. *PHP Object Generator*, [en línea], 2006 [consultado enero 2007] Disponible en : http://www.PHPobjectgenerator.com/
22. *Codejay* , [en línea], 2006 [consultado enero 2007] Disponible en : http://www.codejay.com/
23. SoftVelocity, "*fastest way to build business applications*", [en línea], 2006[consultado enero 2007] Disponible en : http://www.softvelocity.com/clarion/c6.htm
24. Dondo, Agustin, "*PHPgen*", [en línea], 2006 [consultado febrero 2007] Disponible en : http://PHPgen.sourceforge.net/indexsp.PHP
25. PRESSMAN,Roger "*Ingeniería del Software. Un enfoque práctico*".2002. McGraw-Hill/Interamericana, ISBN: 978-84-481-0026-1

Bibliografía

1. HERRINGTON, J. *Code Generation In Action*. Manning, 2006.
2. Jacobson, I.; Booch, G. y Rumbaugh, J.; "*El Proceso Unificado de Desarrollo de software*". 2000. Addison-Wesley. ISBN 84-7829-036-2
3. Software Acumen Limited, "*Code Generator Models*", [en línea], febrero del 2005[Consultado en diciembre de 2006], Disponible en http://www.codegeneration.net/tiki-index.PHP?page=ModelsIntroduction
4. MORENO, P. J. M., Ed. "*Especificación de interfaz de usuario: De los requisitos a la generación de código*", Valencia, Universidad de Valencia, 2003. 402
5. Dondo, Agustín,"*Por qué elegir PHP*" [en línea], Programación en castellano, 2006, [consultado enero 2007] Disponible en : http://www.programacion.net/PHP/articulo/porquePHP/
6. Eck J. David , "*Introduction to Programming Using JAVA*" [en línea],2006 [consultado febrero 2007] Disponible en : http://math.hws.edu/JAVAnotes/
7. Obasanjo,Dare "*Descripción de XML*" [en línea], Microsoft Corporation, 2005 [consultado enero 2007] Disponible en : http://www.microsoft.com/spanish/msdn/articulos/archivo/011003/voices/understxml.asp
8. W3C Communications Team, "*XML in 10 points*", World Wide WEB Consortium,2003, [consultado marzo 2007] Disponible en : http://www.w3.org/XML/1999/XML-in-10-points.html
9. Thompson S. Henry; Beech, David ; Maloney, Murray; Mendelsohn, Noah, "*XML Schema*", World Wide WEB Consortium, 2004 [consultado marzo 2007] Disponible en : http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/
10. Clark, James," *XSL Transformations*", World Wide WEB Consortium, 1999, [consultado marzo 2007] , Disponible en : http://www.w3.org/TR/xslt
11. The Apache Software Foundation, "*How it works*", 2007 [consultado marzo 2007] Disponible en : http://www.apache.org/foundation/how-it-works.html
12. Bath, England. "*WEB Server Survey*", [en línea] [consultado enero 2007] Disponible en : http://news.netcraft.com/archives/2007/05/01/may_2007_WEB_server_survey.html
13. Russell J.T. Dyer, "*MYSQL Tutorial*" [en línea], 2003, [consultado enero 2007] Disponible en : http://www.unixreview.com/documents/s=8989/ur0409f/

14. IBM Corporation, "*IBM Rational Rose*" [en línea], 2005, [consultado noviembre 2006] Disponible en : http://www-306.ibm.com/software/rational/
15. Martín Moreno, Tomás, "*Programación en 3 capas*",[en línea], 2005, [consultado diciembre 2006] , Disponible en : http://www.elguille.info/colabora/NET2005/Tomasmm_3Capas.htm
16. Kurniawan, Budi , "*Using the Singleton Pattern*" , 2004, 2007 [consultado enero 2007] Disponible en : http://www.onJAVA.com/pub/a/onJAVA/2003/08/27/singleton.html
17. Lasater, Chris," Design Patterns" , Wordware Publishing Inc. 2006 [consultado enero 2007] Disponible en : http://www.codeproject.com/books/DesignPatterns.asp
18. LARMAN, Craig, "*UML y Patrones*". 2000. Prentice Hall, ISBN-84-205-3438-2
19. Booch, G.: Rumbaugh, J. y Jacobson, I.; "*El Lenguaje Unificado de Modelado*". 2000. Addison-Wesley. **ISBN** 020189551X
20. Sturm, Jack, "*Desarrollo de soluciones XML*", 2001, McGraw Hill, **ISBN**: 8448131363
21. Gallego Vázquez, José Antonio, "*Desarrollo WEB con PHP y MYSQL*" , 2003, ANAYA MULTIMEDIA , **ISBN** 84-415-1525-5
22. Montero Ayala, Ramón, "*Fundamentos de programación en XML*", 2001, McGraw Hill, **ISBN** 84-481-2894-x
23. Kalishev, Kirill, onBoard Journal, "*Applying Code Generation Approach in Fabrique*", [en línea] 2005, [consultado en marzo del 2007] , Disponible en: http://www.onboard.jetbrains.com/is1/articles/04/10/fabr/
24. Gärtner, Jacob, Cost Journal, "*Code Generator Schemes Aid Safety-Critical Code Development*" [en línea] 2005,[Consultado febrero 2007], Disponible en: http://www.cotsjournalonline.com/home/article.PHP?id=100298
25. Palasí Lallana, Vicent Ramon, "*Motores de Persistencia*" [en línea], Programación en castellano, 2006 [consultado enero 2007] Disponible en : http://www.programacion.net/articulo/joa_persistencia/
26. Bray, Tim; Paoli, Jean; Maler, Eve, "*Extensible Markup Language*", World Wide WEB Consortium, 2006, [consultado enero 2007] Disponible en : http://www.w3.org/TR/xml/
27. Bondre,Prathit, "*Efficient Programming Techniques*" , OASIS, 2005 [consultado enero 2007] Disponible en : http://www.xml.org/xml/xslt_efficient_programming_techniques.pdf

28. Alachisoft, "*Rapidly Develop .NET Applications with O/R Mapping*", [en línea], 2007 [consultado enero 2007] Disponible en: http://www.alachisoft.com/tdev/index.html
29. YesSoftware, "*CodeCharge Studio Feature List*", [en línea], 2006 [consultado enero 2007] Disponible en : http://www.yessoftware.com/products/product_detail.PHP?product_id=1
30. Visual Paradigm, "*10 Reasons to Choose Visual Paradigm*", [en línea], 2006 [consultado enero 2007] Disponible en : http://www.visual-paradigm.com/aboutus/10reasons.jsp
31. "*PHP Object Generator*", [en línea], 2006 [consultado enero 2007] Disponible en : http://www.PHPobjectgenerator.com/
32. "*Codejay*" , [en línea], 2006 [consultado enero 2007] Disponible en : http://www.codejay.com/
33. SoftVelocity, "F*astest way to build business applications*", [en línea], 2006[consultado enero 2007] Disponible en : http://www.softvelocity.com/clarion/c6.htm
34. Dondo, Agustin, "*PHPgen*", [en línea], 2006 [consultado enero 2007] Disponible en : http://PHPgen.sourceforge.net/indexsp.PHP

GLOSARIO

ASP

Páginas activas del servidor (Active Server Pages) es un lenguaje de programación WEB del lado del servidor.

Bioinformática:

Es la aplicación de los ordenadores y los métodos informáticos en el análisis de datos experimentales y simulación de los sistemas biológicos. Una de las principales aplicaciones de la bioinformática es la simulación, la minería de datos y el análisis de los datos obtenidos en un estudio.

C++

Lenguaje de programación de alto nivel que implementa el paradigma de programación orientada a objetos.

CASE:

Ingeniería de Software Asistida por Computadoras (Computer Aided Software Engineering). Es el uso de la asistencia de software para organizar y controlar el desarrollo de software.

Clase

Declaración o abstracción de un objeto cuando se programa usando el paradigma de orientación a objetos.

Código:

Cifras, clave. En informática se utiliza para referirse a un conjunto de instrucciones en un lenguaje de programación.

Frameworks:

Estructura predefinida para la creación de aplicaciones. Puede estar formado por un conjunto de librerías y clases o por una arquitectura que facilita el desarrollo de software.

HTML

Lenguaje hipertexto de marcas (HyperText Markup Language). Es el lenguaje más usado para la creación de contenidos en Internet.

IDES

Ambiente integrado de desarrollo (Integrated development environment). Es un conjunto de software que permite el desarrollo de aplicaciones.

Ingeniería inversa

Es la obtención de información y características técnicas a partir de un producto ya creado.

Instanciar

Crear una instancia de una clase. Reservar dinámicamente el espacio de memoria necesario para almacenar sus campos y el resto de los datos que permitirán su operación mediante una llamada al constructor de la clase

JAVA

Lenguaje de programación de alto nivel orientado a objetos y multiplataforma

JAVAscript

Lenguaje interpretado utilizado en la programación WEB del lado del cliente.

LIMS

Sistema de manejo de información de laboratorio (Laboratory Information Management System). Es un software destinado al manejo y gestión de la información y los procesos dentro de un laboratorio.

Modelo

Representación abstracta de la realidad. Diagramas que representan la estructura de un sistema dado.

Multiplataforma

Es un término usado para referirse a los programas, sistemas operativos o lenguajes de programación que puedan funcionar en diversas plataformas como Windows o Linux.

.NET

Plataforma de desarrollo propuesta por Microsoft. Integra lenguajes de programación y frameworks para el desarrollo de aplicaciones.

Patrones

Normas de comportamiento, características que identifican una situación.
En informática un patrón es una solución a un problema de diseño no trivial que es efectiva y reutilizable. Es posible aplicar a diferentes problemas de diseño en distintas circunstancias.

PHP:

Lenguaje de programación usado para la creación de aplicaciones para sitios WEB.

Scripts

Escrito, nota. En informática se refiere a un fragmento de código que realiza una funcionalidad determinada.

Sintaxis

Disciplina lingüística que estudia la forma que se combinan las palabras. En informática se refiere a la forma en que se definen las instrucciones en un lenguaje de programación.

Software:

Término genérico que designa al conjunto de programas que posibilitan realizar una tarea específica en un ordenador.

SQL

Lenguaje de Consulta Estructurado (Structured Query Language). Es un lenguaje destinado a la definición y manipulación de base de datos relacionales.

UML

Lenguaje Unificado de Modelado (Unified Modeling Language). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software.

W3C

World Wide WEB Consortium. Consorcio internacional que produce estándares para la WEB.

WEB:

Tela de araña, tejido, red. Sistema de redes mundiales interconectados entre si. Conjunto de documentos enlazados entre sí y distribuidos por todo el mundo.

Wizard

Fragmento de un programa que sirve de guía al usuario durante una operación.

XML

Lenguaje de marcas extensible (eXtensible Markup Language).

XPAHT

XML Path Language. Lenguaje para escribir expresiones que seleccionan elementos en un documento XML.

XSD

Lenguaje utilizado para describir la estructura y las restricciones de los contenidos de los documentos XML.

XSLT

Estándar de la organización W3C que presenta una forma de transformar documentos XML.

ANEXOS

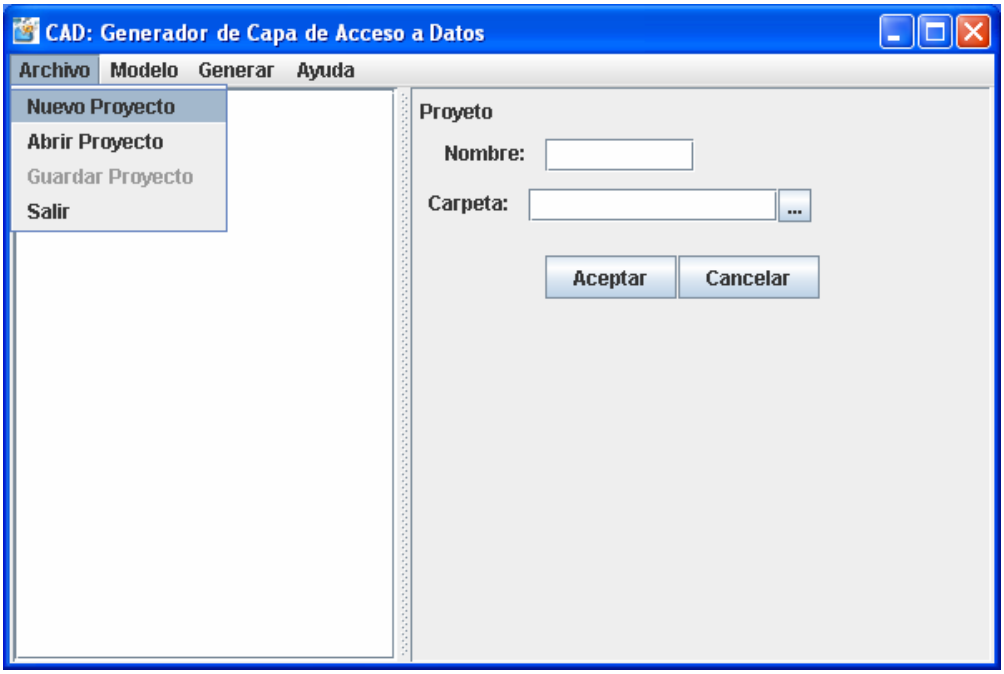

Fig 53: Interfaz Nuevo proyecto.

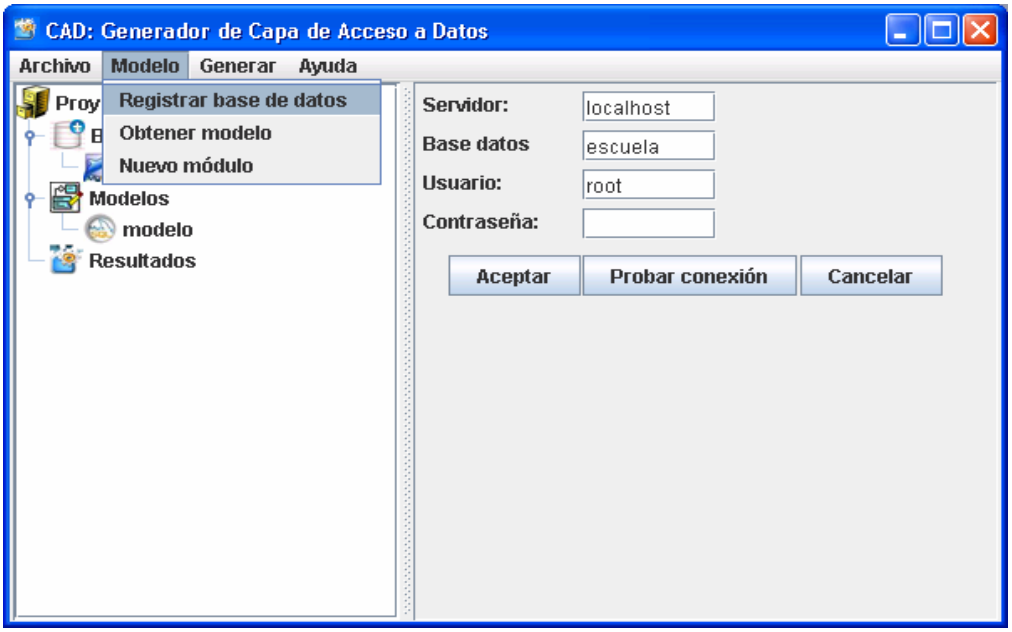

Fig 54: Interfaz Registrar base de datos.

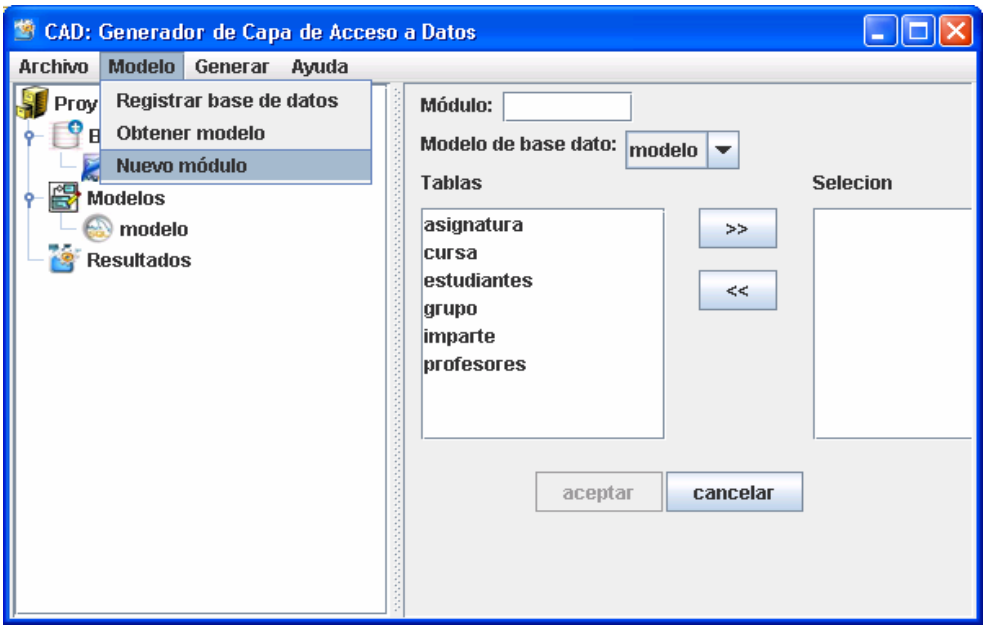

Fig 55: Interfaz Nuevo módulo.

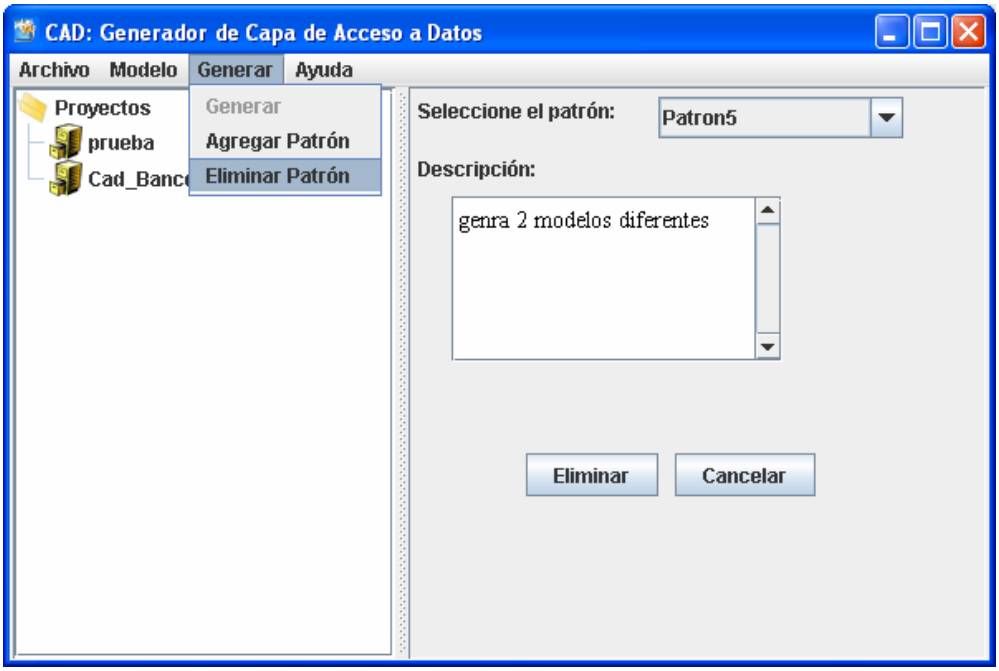

Fig 56: Interfaz Eliminar patrón.

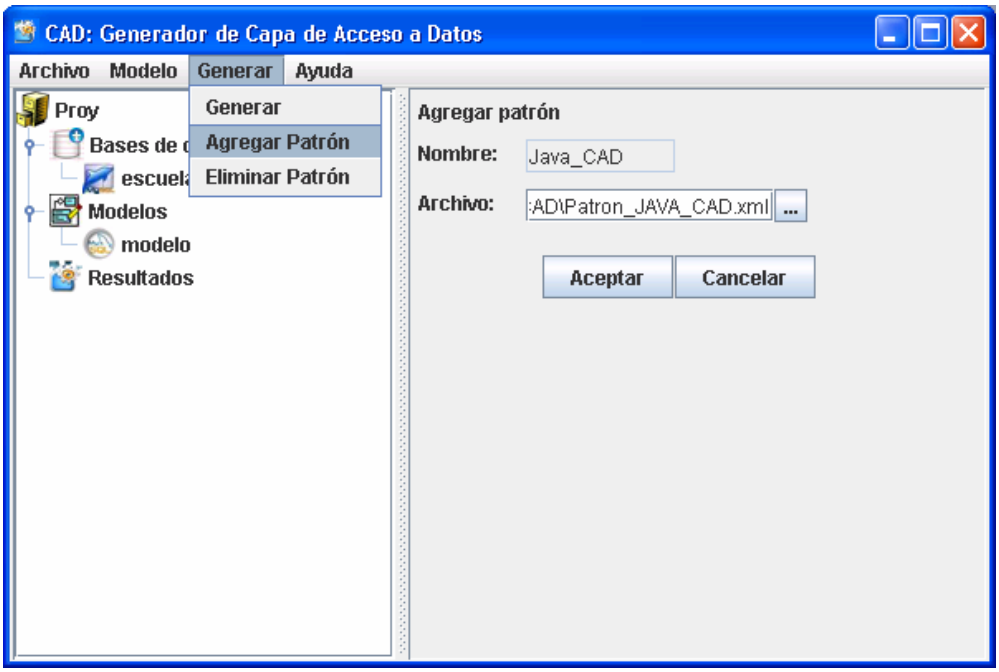

Fig 57: Interfaz Agregar patrón.

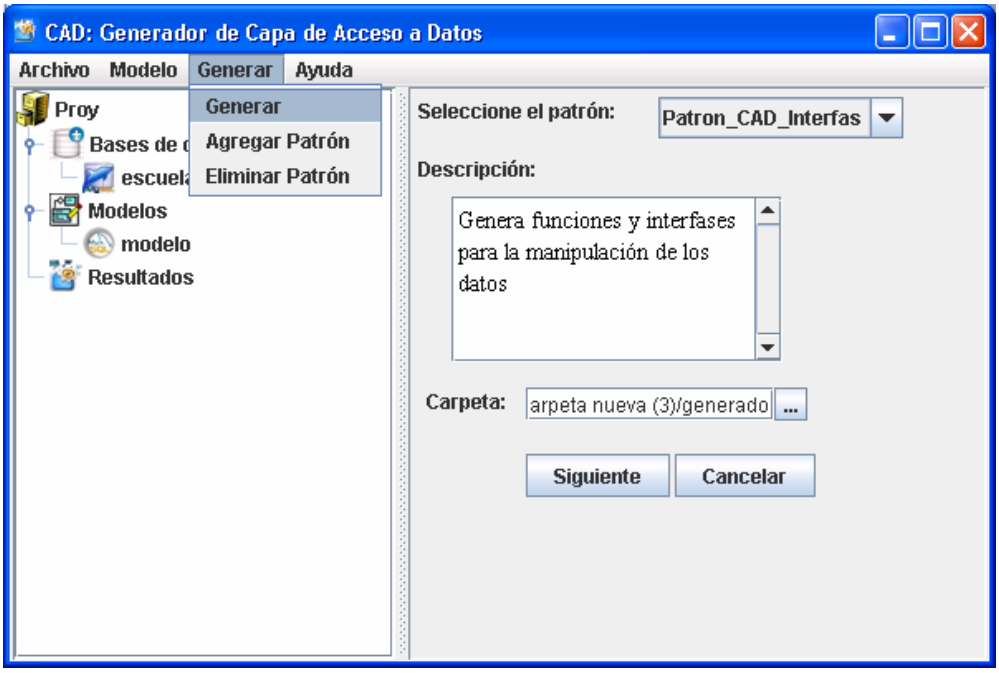

Fig 58: Interfaz Generar.