SÍMPLEX MEDIA - SOFTWARE MULTIMEDIAL DE APOYO AL ESTUDIO DEL MÉTODO SÍMPLEX

SANDRA YAKELINE ASCUNTAR URBANO
JAMES RODRIGO CASTILLO QUIÑONES
RONALD JONATAN CORREA MONAGA

UNIVERSIDAD DE NARIÑO
FACULTAD DE INGENIERÍA
PROGRAMA DE INGENIERÍA DE SISTEMAS
SAN JUAN DE PASTO
2009

SÍMPLEX MEDIA - SOFTWARE MULTIMEDIAL DE APOYO AL ESTUDIO DEL MÉTODO SÍMPLEX

SANDRA YAKELINE ASCUNTAR URBANO
JAMES RODRIGO CASTILLO QUIÑONES
RONALD JONATAN CORREA MONAGA

Proyecto de Grado presentado como requisito parcial para optar al título de Ingeniero de Sistemas

Ing. Esp. LUIS VICENTE CHAMORRO MARCILLO.
Director

Ing. Esp. JESÚS INSUASTY PORTILLA.
Codirector

UNIVERSIDAD DE NARIÑO
FACULTAD DE INGENIERÍA
PROGRAMA DE INGENIERÍA DE SISTEMAS
SAN JUAN DE PASTO
2009

Las ideas y conclusiones aportadas en la tesis de grado son responsabilidad exclusiva de los autores. Artículo 1 del acuerdo Nº 324 de octubre 12 de 1966, emanado del honorable Consejo Directivo de la Universidad de Nariño.

Nota de aceptación

_____
_____
_____
_____
_____
_____

_____
Firma del presidente del jurado

_____
Firma del Jurado

_____
Firma del Jurado

_____
Firma del Director del proyecto

San Juan de Pasto, 6 de febrero de 2009

DEDICATORIA

Dedico este trabajo a Dios por estar conmigo en cada momento de mi vida y ser mi guía, a mi papá por ser mi amigo incondicional a mis hermanos Francisco y Mariluz por su apoyo, a mi tía por su ayuda incondicional, y a toda mi familia por estar ahí cuando los necesité.

Sandra Yakeline Ascuntar Urbano

Gracias a Jehová Dios por darme consistencia y sabiduría, gracias a mis padres por apoyarme y comprenderme cuando más lo necesitaba.

James Rodrigo Castillo Quiñones

Dedicado a aquel de quien proviene toda dádiva buena y todo don perfecto, Jehová, por ser mi fuente de energía y fortaleza constantes, y a todos aquellos que con su apoyo y ayuda contribuyeron a realizar este proyecto.

Ronald Jonatan Correa Monaga

AGRADECIMIENTOS

Agradecemos a Dios, por estar presente en cada paso dado durante el transcurso de nuestra carrera y por permitirnos alcanzar una meta más en nuestras vidas.

A nuestros familiares por apoyarnos de manera incondicional y permanente.

A los Docentes Vicente Chamorro y Jesús Insuasty por su tiempo y asistencia continua durante el desarrollo del proyecto.

Al Docente Oscar Revelo por su colaboración oportuna y desinteresada.

A nuestros amigos, por ser cómplices en este proceso que se convirtió en un reto alcanzable.

A nuestros profesores y compañeros por compartir con nosotros sus conocimientos y experiencias que sirvieron para formarnos.

Y a todas aquellas personas que de una u otra manera colaboraron para que este proyecto se lleve a cabo.

GLOSARIO

- **Algoritmo Simplex Primal:** El Algoritmo Símplex Primal es un algoritmo matemático que busca una solución inicial factible para luego buscar sistemáticamente otras soluciones básicas que tengan el potencial de mejorar el valor de la función objetivo identificando puntos de esquina de forma algebraica.

- **Coeficiente**: Cuando se multiplican dos factores, cualquiera de los dos puede ser llamado coeficiente del otro factor. Por ejemplo, en el producto 5a el factor 5 es coeficiente del factor "a", lo que indica que "a" debe multiplicarse 5 veces.

- **Desigualdad**: Una Desigualdad es una expresión matemática que consta de dos miembros o partes los cuales se encuentran asociados por los signos <, >, >= y <= (cuando el signo de la asociación es el "=" deja de ser una desigualdad y se convierte en una Igualdad). Estos miembros pueden ser, por ejemplo: una sumatoria de varios números reales, o sencillamente: un número real.

- **Ecuación**: Una ecuación es una igualdad en la que puede encontrarse una o varias incógnitas o variables, y solo se verifica o es verdadera para determinados valores que satisfagan la igualdad.

- **Ecuación Lineal:** Una Ecuación Lineal es un tipo de ecuación en la cual sus variables no están elevadas a una potencia mayor a 1.

- **Ejercicios Explicados**: Problemas tratados dentro del contexto de cada temática. Estos se encuentran registrados en el sistema, y se muestra al usuario su planteamiento, solución y explicación.

- **Estructura de una ecuación:** La estructura de una ecuación o de una inecuación está compuesta por dos miembros separados por un signo que puede ser =, <, >, <= o >=. Cada uno de los miembros a su vez está compuesto por términos.

- **Forma estándar de programación lineal:** Es la reorganización de los datos del problema en un formato predefinido, el cual se usa como punto de partida en cualquiera de los algoritmos del Método Símplex.

- **Frase**: Palabra u oración que el usuario introduce en el aplicativo para que el sistema busque los materiales que traten acerca de dicho tema.

- **Función Objetivo**: La función objetivo es una expresión matemática que sirve de referencia o meta, a la cual debe apuntar el desarrollo del algoritmo, los valores que tomen sus Variables deben satisfacer las Restricciones del problema.

- **Igualdad:** Una igualdad es una expresión matemática que consta de dos miembros asociados por el símbolo "=".

- **Incógnita:** Una incógnita es un Valor desconocido dentro de una ecuación o dentro de una inecuación.

- **Inecuación:** Una inecuación es una desigualdad en la que se encuentran incógnitas o variables con valores desconocidos.

- **Material**: Hace referencia a una de las temáticas tratadas dentro del marco teórico, lo que incluye: temas, ejercicios y/o términos. Cuando se hace referencia a la explicación de un material se quiere decir que se explicará un tema, un ejercicio o un término. Estos materiales están desarrollados en páginas XAML y pueden tener animaciones asociadas y es posible indicar o establecer si el sistema leerá al usuario la información.

- **Matriz**: Una Matriz puede definirse como una colección de datos organizados en una tabla dividida en filas y columnas. Cada uno de los datos posee su posición respectiva y representan algo en concreto dependiendo del problema, dependiendo del sentido o concepción que se le asigne a cada columna y a cada fila.

- **Método de Gauss:** Es un método que se utiliza para dar solución a un Sistemas de Ecuaciones Lineales, y consiste en ir transformando el sistema de inicial en otro equivalente el cual proporciona el resultado final.

- **Método de Gauss Jordan:** Es un método que se utiliza para dar solución a un Sistemas de Ecuaciones Lineales, y consiste en ir transformando el sistema de inicial en otro equivalente el cual proporciona el resultado final.

- **Método Gráfico:** El método Gráfico es un método matemático, el cual hace uso del plano cartesiano para dar solución a un Sistemas de Ecuaciones Lineales o una serie de restricciones expresadas mediante inecuaciones lineales.

- **Método Símplex**: Es un modelo de programación Lineal que sirve para expresar de forma algebraica problemas de la vida real que necesiten una solución factible. Estos problemas deben tener ciertas características (Conjunto de Restricciones, función Objetivo y Variables), pues de lo contrario no satisfarán el modelo.

- **Método Símplex Primal:** El Método Símplex Primal, cuenta con una serie de algoritmos como el Algoritmo Símplex Primal, Técnica M, Dos Fases, etc., diseñados para localizar la solución óptima entre las soluciones factibles que se presenten durante el desarrollo un problema de Programación Lineal.

- **Métodos de Eliminación:** Un método de eliminación se utiliza para resolver Sistemas de Ecuaciones Lineales. Entre los más conocidos se encuentra: Eliminación por Sustitución de Variables, Eliminación por Igualación y Método de Reducción.

- **Optimalidad:** Valor que adquiere la función objetivo, tras cada iteración con el Método Símplex.

- **Penalización:** "Castigo" o "Precio" que es necesario pagar a fin de compensar el "favor" o la "aprobación" que se ha otorgado al darse el "permiso" de incluir las variables de holgura, superávit o artificiales que facilitarán el desarrollo del ejercicio, y consiste en modificar la "Función Objetivo" agregando tantos

elementos en la Función Objetivo como el número de variables introducidas en las restricciones.

- **Pivote:** El pivote es el elemento que se escoge de una matriz y se toma como referencia para realizar operaciones de renglón dentro de ella.

- **Programación Lineal:** La programación lineal ofrece una serie de técnicas y algoritmos matemáticos para optimizar (maximizar o minimizar) una función lineal, denominada Función Objetivo, sujeta a una serie de restricciones expresadas mediante *inecuaciones lineales*.

- **Propiedades de las Ecuaciones:** Las Propiedades de las Ecuaciones son una serie de reglas que deben tenerse en cuenta para la solución de las ecuaciones.

- **Punto de Esquina:** Un punto ubicable en el plano cartesiano, el cual hace parte del conjunto de posibles soluciones que puede tener el problema, cuenta además con la particularidad de encontrarse en la intersección de dos rectas y puede convertirse en el valor máximo o mínimo de una Función Objetivo. Aunque se habla de punto de esquina generalmente cuando se está trabajando con el Método Gráfico, es la base del Método Símplex.

- **Raíz:** La raíz es el valor que al ser reemplazado en una ecuación hace que el resultado sea una igualdad.

- **Restricción:** Ecuación Lineal utilizada para expresar de forma matemática características de un problema de la vida real, la cual se encuentra inmersa dentro de un sistema de ecuaciones, a partir del cual se inicia el proceso de optimización.

- **Restricciones**: Son una serie de restricciones o desigualdades que representan los límites de disponibilidad y utilización de los recursos (representados por variables) que forman parte de la función objetivo.

- **Sentencia Numérica:** Cuando se usa la frase "*sentencia numérica*" hace referencia a una relación general que podría ser de igualdad o de desigualdad (Ecuación o Inecuación).

- **Sistema Lineal:** Un sistema de ecuaciones es lineal si está formados por 2 o más ecuaciones de primer grado, con dos o más incógnitas, que deben ser resueltas en forma simultánea.

- **Solución Básica Factible o Solución Factible:** Dependiendo del objetivo del problema (maximizar o minimizar), la solución factible representa los valores que deben tomar las variables a fin de que satisfagan de forma adecuada todas las restricciones del problema y proporcione un valor óptimo posible.

- **Solución Básica Óptima o Solución Óptima:** En el desarrollo del Algoritmo Simplex Primal se van encontrando sucesivamente soluciones básicas factibles, cada una de las cuales puede ser mejor que la anterior, hasta que finalmente encuentra una que no puede ser mejorada a esta se le denomina solución básica óptima.

- **Solución Inicial**: Es el primer resultado obtenido a partir de la Tabla Símplex Inicial en donde la solución está dada por las variables básicas.

- **Tabla Símplex:** La Tabla Símplex es una tabla en la cual se organizan los datos para realizar los procesos matemáticos, contiene las Restricciones y la Función Objetivo y está organizada de tal forma que se pueda distinguir la solución básica factible, las variables básicas, y las no básicas, además de las variables de decisión.

- **Término (en SÍMPLEX MEDIA)**: Hace referencia a una palabra o frase que se busca dentro de la base teórica mediante una serie de vínculos que llevan de un lugar a otro con el fin de apoyar la comprensión de algunos puntos importantes en determinado tema.

- **Término (en Sentencias Numéricas):** Se designa como término a cada una de las cantidades que se encuentra conectada una a otra por el signo “+” o “-”. Los términos pueden ser valores específicos, incógnitas o productos de números e incógnitas.

- **Usuario**: Es la persona que manipula el sistema, a quien se le proporciona toda la información y quien brinda todos los datos necesarios para la ejecución del

aplicativo. No solo representa a estudiantes universitarios, sino también a otras personas que hagan uso del sistema

- **Valores que Satisfacen una Ecuación:** Son una serie de valores que pueden tomar las variables con los cuales la ecuación se convierte en una expresión verdadera. Véase Raíz.

- **Variable:** Una variable es un símbolo que representa un valor no especificado, que puede adquirir o ser sustituido por un valor cualquiera.

- **Variable Artificial:** La finalidad de estas variables es puramente matemática, facilitan crear la Matriz Identidad, con lo que se hace posible hallar la primera solución básica del problema.

- **Variable de Decisión:** Es una variable que permite el análisis de los datos al finalizar el proceso mediante cualquier algoritmo.

- **Variable de Holgura:** Es una variable que se adiciona durante el paso de los datos, del problema original a la forma estándar y se hace únicamente cuando las Desigualdades son de la forma "<=", haciendo alusión a lo que le falta al miembro izquierdo de la Inecuación para ser igual al derecho. Esto se hace con el fin de facilitar el proceso matemático y analítico.

- **Variable Superávit:** Es una variable que se adiciona durante el paso de los datos, del problema original a la forma estándar y se hace únicamente cuando las desigualdades son de la forma >=, con el fin de restar la cantidad que se encuentra en exceso en el miembro izquierdo de la inecuación y hacer que la expresión matemática se convierta en una ecuación.

- **Variables Básicas:** Son las variables que representan de la Tabla Símplex a las que se les ha asignado un valor específico y por tanto afectan la Función Objetivo.

- **Variables No Básicas:** Es aquella variable que se encuentra en la tabla Símplex, pero a la cual no se le ha asignado ningún valor en la columna solución.

RESUMEN

Este documento contiene el análisis y diseño del proyecto de grado "SÍMPLEX MEDIA - Software Multimedial de Apoyo al Estudio del Método Símplex", el cual nace para servir como ayuda didáctica y facilitar el proceso de enseñanza y aprendizaje del método símplex, los algoritmos que lo componen y fundamentos teóricos necesarios para su comprensión.

Esta herramienta, consta de tres módulos; *Información Teórica*, *Ejercicios Prácticos* y *Solución de Ejercicios*, los cuales interactúan para facilitar al estudiante, la comprensión del método símplex.

Módulo de Información Teórica: en este módulo se encuentra contenida toda la base teórica, que servirá de apoyo para el estudio del método símplex. Se analizan temas adicionales como matrices y sistemas de ecuaciones lineales. En lo referente a la programación lineal, se estudian: el Método Símplex Primal con sus algoritmos (Algoritmo Símplex Primal, Algoritmo de la Técnica M, Algoritmo de Dos Fases) y el Método Símplex Dual.

Módulo de Ejercicios Prácticos: a través de este módulo el sistema presenta una serie de ejercicios explicados de los diferentes algoritmos que compone el Método Símplex Primal. Cada ejercicio consta de una solución guiada, que va desde la formulación y planteamiento del problema, hasta la interpretación de los resultados.

Módulo de Solución de Ejercicios: este módulo, se subdivide para dar solución a tres tipos de ejercicios diferentes.

El primero resuelve ejercicios del método símplex, haciendo uso de los algoritmos, Símplex Primal, Técnica M, Dos Fases y Símplex Dual. En este sub módulo, para cualquiera de los algoritmos, el software resuelve los problemas de forma directa o paso a paso, explicando lo que ocurre en sus iteraciones.

En el segundo sub módulo, es posible resolver sistemas de ecuaciones lineales, ya sea dando la solución de forma directa o paso a paso, haciendo uso de los algoritmos de Gauss y Gauss Jordan.

Y un tercer sub módulo, realiza operaciones con matrices, como suma, resta y multiplicación, o combinaciones de estas.

ABSTRACT

This document contains the analysis and design "SÍMPLEX MEDIA - Software Multimedial to Support the Study of the Símplex Method" project, which born to serve like didactic help and facilitate the teaching learning process, the algorithms that compose it and theoretical foundations to its compression.

This tool, has three modules; theoretical information, practical exercises solution, and exercises solution, which interact to facilitate to student, símplex method comprehension.

Theoretical Information Module: contained the whole theoretical base that will serve as support for method símplex study.

Additional topics are analyzed as and lineal equations systems. Regarding the lineal programming is studied: the Yearling Símplex Method with their algorithms (Algorithm Yearling Símplex, The Technical M, and Algorithm of Two Phases) and Dual Símplex Method.

Practical Exercises Module: through this module the system presents a series of explained exercises to different algorithms that composes the Yearling Símplex Method. Each exercise consists of a guided solution that goes from the formulation and the problem position, until the results interpretation.

Exercises Solution Module: this module is subdivided to give solution to three types of different exercises.

The first one solves exercises of símplex method, making use of algorithms, símplex primal, technical m, two phases and símplex dual. In this sub module, to anyone of algorithms, the software solves the problems in a direct way or step by step, explaining what happens in its iterations.

In second sub module, is possible to solve lineal equations systems, either giving the solution in a direct way or step by step, making use of Gauss and Gauss Jordan algorithms.

In a third sub module, carries out operations with wombs, like it adds, it subtracts and multiplication, or combinations of these.

CONTENIDO

LISTA DE TABLAS

LISTA DE ILUSTRACIONES

1 INTRODUCCIÓN

El giro que el avance de la tecnología ha dado a la educación es muy amplio y ha generado nuevos planteamientos en cuanto a la metodología que esta debe adoptar. El uso de la tecnología ha permitido crear nuevas herramientas de apoyo para el aprendizaje y la enseñanza volviéndolas más dinámicas y atractivas.

Todos estos avances, tanto a nivel de hardware como de software, han permitido que en el campo de la educación se trabaje en la construcción de aplicaciones para que los estudiantes tengan a la mano herramientas que les ayuden a profundizar e incrementar sus conocimientos sobre diversos temas en el momento en que ellos lo necesiten, al tiempo que se convierten en nuevos soportes metodológicos para el docente.

Este proyecto abordará, dentro del marco de un Software Multimedial, el Método Símplex de la programación lineal cuyo impacto desde 1950 ha sido extraordinario. Se han escrito decenas de libros de texto sobre la materia y los artículos publicados que describen aplicaciones importantes, se cuentan ahora por cientos. De hecho, una proporción importante de todo el cálculo científico que se lleva a cabo en computadores, se dedica al uso de la programación lineal y a técnicas íntimamente relacionadas.

En este proyecto se pretende desarrollar una herramienta computacional multimedial, que sirva de apoyo al estudio del Método Símplex.

La herramienta consta de tres módulos en formato multimedial: Base teórica de los modelos, presentación de los modelos con ejercicios asistidos por el sistema y solucionador de problemas del Método Símplex.

En este documento, se presentan entre otros temas, la justificación y los objetivos del proyecto así como la metodología que se siguió para su desarrollo.

1.1 DESCRIPCIÓN

1.1.1 Planteamiento del problema. Actualmente la Universidad de Nariño cuenta con un muy limitado inventario de herramientas tecnológicas que faciliten el estudio y aprendizaje del Método Símplex. Aparte del docente, los estudiantes, sólo se apoyan en textos o en Internet donde muchos de los temas no son abarcados de forma sencilla. Existen aplicaciones que utilizan Programación Lineal para dar resultados, sin permitir ver el procedimiento de cómo se obtuvieron y se puede afirmar que no se cuenta con una herramienta que facilite de forma didáctica el estudio de estos temas.

El diario compartir con estudiantes que han utilizado esta metodología permite afirmar que los elementos didácticos tradicionales no han sido suficientes para algunos, y han tenido dificultades en alcanzar los logros académicos esperados, razón por la cual, el uso de estos algoritmos en la vida cotidiana es reducido.

1.1.2 Formulación del problema. ¿Cómo contribuir al estudio eficiente del Método Símplex con el apoyo de nuevas tecnologías informáticas?

1.1.3 Sistematización del problema

- ✓ ¿Cómo sistematizar la información existente del Método Símplex para adaptarla al desarrollo de una aplicación multimedial?

- ✓ ¿Cuál es la característica más relevante que se tiene en cuenta en la metodología empleada para el desarrollo de la Aplicación?

- ✓ ¿Cuál debe ser el diseño de la interfaz de usuario para que sea de fácil manejo, eficiente y agradable?

- ✓ ¿Cómo se incorporarán en el sistema las respuestas a las necesidades didácticas fundamentales?

1.1.4 Alcance y delimitación. El proyecto se centra en la explicación didáctica del Método Símplex y aquellos temas preliminares necesarios para su compresión.

La temática abarca: Modelo de dos variables y su Solución Gráfica, Método Símplex Primal, que contiene a su vez, el Algoritmo Símplex Primal, el Algoritmo de la Técnica M y el Algoritmo de las Dos Fases, y Método Símplex Dual. Los temas preliminares que facilitarán la compresión del Método Símplex son: conceptos y tipos de matrices, operaciones con matrices, ecuaciones e inecuaciones, solución de sistemas de ecuaciones lineales.

Aunque estos últimos temas no son el objetivo principal del proyecto, son necesarios para que el estudiante comprenda adecuadamente los fundamentos del Método Símplex y sus diferentes algoritmos.

El aplicativo guía al estudiante a través del aprendizaje de las temáticas incluyendo sus *fundamentos teóricos,* la *explicación de ejercicios del sistema* y la *solución de problemas planteados por el estudiante*.

El proyecto se divide en tres módulos que se entrelazan en su trabajo para un mismo propósito: enseñar de manera fácil, didáctica y entretenida toda la teoría del sistema. Los módulos son:

- *Módulo de Información Teórica:* este módulo proporciona toda la base teórica necesaria para la comprensión adecuada del Método Símplex. Mediante él, el usuario de SÍMPLEX MEDIA, tiene acceso a temas principales, temas secundarios y definiciones puntuales de cada uno de las áreas y conceptos descritos arriba. Está diseñado para presentar de forma escrita, una redacción simple basada en ejemplos reales y explicaciones exactas que han sido subdivididas en pasos concretos para facilitar su comprensión.

  Este módulo se apoya en el *Módulo de Ejercicios Prácticos* a fin de sustentarse con ejemplos reales. Además, los ejemplos utilizados como base de explicaciones pueden ser comprobados usando el tercer módulo, *Solución de Ejercicios*.

- *Módulo de Ejercicios Prácticos*: a través de este módulo el sistema presenta una serie de ejercicios con solución guiada y analiza el significado de cada resultado. Esto permitirá al usuario aplicar los conceptos teóricos en la vida real y aprender a plantear, modelar, estandarizar y solucionar problemas de programación lineal.

  En vista de que el objetivo primordial de este proyecto es ayudar al estudio del Método Símplex, éste módulo se centrará en explicar ejercicios sólo del dicho Método. Pese a ello, en la *Información Teórica*, se analizan paso a paso ejemplos adecuados que lograrán el mismo propósito. Tal como en el caso anterior, *Ejercicios Prácticos* se apoya con sus asociados. Por ejemplo: proporciona vínculos a definiciones y temas que se analizan en el primer módulo y, además, plantea problemas que el usuario podrá resolver usando *Solución de Ejercicios*.

- *Módulo de Solución de Ejercicios:* aquí se da la posibilidad de plantear ejercicios propios y de desarrollarlos siguiendo uno de tres posibles procedimientos: solución inmediata por parte del sistema, solución paso a paso por parte del sistema y solución paso a paso con explicaciones de cada uno de los pasos tras cada iteración.

  En este módulo se puede dar solución a tres tipos de problemas: "*Operaciones con Matrices*", tales como, suma, resta y multiplicación; solución a sistemas de *"Ecuaciones Lineales"* mediante los métodos de eliminación de Gauss y Gauss Jordan; y problemas de "Programación Lineal" que pueden ser solucionados mediante cuatro de los diferentes algoritmos que se analizan en SÍMPLEX MEDIA: Algoritmo Símplex Primal, Algoritmo de la Técnica M, Algoritmo de Dos Fases y Método Símplex Dual.

  La interconexión modular, en este caso, se presenta en el momento en que *Solución de Ejercicios* presenta la explicación de un ejercicio, brinda vínculos a definiciones, temas y ejercicios que pueden ser estudiados usando a los módulos *Información Teórica* y *Ejercicios Prácticos*.

Otra de las funcionalidades importantes de SÍMPLEX MEDIA es la facilidad que presenta para extraer de él la información que el usuario necesite. Es posible copiar el contenido de alguno de los materiales que se presenten para pegarlo en una aplicación procesadora de textos. También es posible realizar impresiones.

En cuanto a los ejercicios, el software le permite guardar y recuperar copias en formato digital, además de realizar la impresión del desarrollo paso a paso que proporciona el tercer módulo.

1.1.5 Antecedentes. Realmente, se conocen pocos antecedentes acerca de un proyecto de este tipo. Existen diversos aplicativos que realizan de forma adecuada algunas de las tareas que realiza SÍMPLEX MEDIA. Sin embargo, es muy difícil conseguir uno que convine el trabajo educativo con el poder de un Solucionador de Ejercicios. Por citar algunos ejemplos, en el ámbito internacional existen programas tales como:

- WinQSB (módulo: Linear and Integer Programing): sistema computacional desarrollado bajo ambiente Windows que permite el desarrollo de problemas de programación lineal (Maximización y Minimización) con interfaz gráfica amigable.

- FarmTracker: Sistema integrado que permite ingresar los datos productivos de un establecimiento, mejorando la toma de decisiones productivas y económicas.

- NRC Model Application - Swine: software de los modelos que predicen los niveles de nutrientes necesarios para alcanzar un cierto nivel productivo, bajo unas determinadas condiciones ambientales.

- UFFDA: User-Friendly Feed Formulation Program: software práctico de uso libre que calcula raciones de mínimo costo basado en sistemas de programación lineal, desarrollado bajo Pascal para el sistema DOS en la Universidad de Georgia.

Por otro lado, existen varios programas genéricos para la solución de los modelos de Programación Lineal como TORA y una aplicación de QSB, que permiten hallar la solución óptima de problemas de una forma rápida pero desafortunadamente con escasa o ninguna guía didáctica que facilite el aprendizaje del tema, y mucho menos, con fundamentación teórica de conceptos preliminares.

En el nivel regional, estudiantes de las diferentes universidades han desarrollado software educativo y multimedial en muchas áreas y líneas del conocimiento aunque ninguna enfocada al estudio del Método Símplex, con excepción del Proyecto de grado AQI-Diet, Software para el Balanceo de Dietas en Especies Icticas, presentado por el ingeniero

Hernán Javier López, que implementa varios de estos modelos para el cálculo de la porción de alimento necesaria para la nutrición adecuada de peces, pero que tampoco tiene enfoque didáctico[1].

1.1.6 Justificación. El estudio del Método Símplex, igual que el de otras áreas del conocimiento, exige el uso de herramientas de apoyo didáctico para su clara comprensión, por lo que los docentes deben recurrir a material sencillo y práctico para la explicación de la temática. Entre los materiales didácticos que se están utilizando se pueden mencionar, entre otros, los gráficos de libros, las diapositivas con animaciones y algunas aplicaciones de software.

Pensando en esta última forma de ayuda didáctica, en la Universidad de Nariño se han desarrollado algunos Software Multimediales como apoyo a algunas temáticas entre los que merecen especial mención el Software Multimedial para el Santuario de Flora y Fauna Galeras y el Software Multimedial sobre el proceso de educación ambiental participativo "laurelito protector" primera fase para el plan de investigación y fomento e industrialización del Laurel de cera "PIFIL".

Desafortunadamente, la Universidad de Nariño no cuenta con un Software Multimedial para el estudio del Método Símplex. Existen aplicativos prácticos, que solicitan datos y generan resultados, pero que no permiten la comprensión de la forma en que se obtienen aquellos resultados.

Con el fin de suplir dicha necesidad, este proyecto tiene como finalidad desarrollar un Software Multimedial para el estudio del Método Símplex que sea una herramienta computacional de apoyo al docente para agilizar y facilitar el estudio de este tema y sirva, además, como una ayuda didáctica y dinámica para el estudiante.

La investigación que se ha considerado realizar tiene como mira, hallar maneras novedosas de abordar el Método Símplex con el uso de esta herramienta multimedial. Se espera que los resultados sean de provecho para la comunidad educativa de la Universidad de Nariño y de los demás centros educativos que deseen implementar en su metodología de trabajo el uso de esta herramienta, en la medida en que mejoren los métodos y condiciones de trabajo y aprendizaje del Método Símplex.

[1] López, H. AQI-Diet, Software para el Balanceo de Dietas en Especies Icticas. 2006. Todo el Libro.

1.2 OBJETIVOS

1.2.1 Objetivo general. Desarrollar una herramienta Multimedial que sirva de apoyo para el estudio del Método Símplex.

1.2.2 Objetivos específicos

- ✓ Recolectar, analizar y clasificar información relacionada con el estudio del Método Símplex.

- ✓ Orientar el desarrollo del proyecto con fundamentos pedagógicos y didácticos que permitan el mejoramiento del aprendizaje.

- ✓ Construir un módulo de *Información Teórica* que proporcione al usuario de la aplicación, todas las explicaciones necesarias para entender los fundamentos, el proceso y la aplicación del Método Símplex.

- ✓ Construir un módulo de *Ejercicios Prácticos* que proporcione al usuario ejercicios reales resueltos con el Método Símplex, brindando una estructura que sirva de guía para modelar el problema, de la forma textual a la cuantitativa.

- ✓ Construir un módulo de *Solución de Ejercicios* que permita al usuario ingresar y resolver ejercicios del Método Símplex observando de forma detallada los pasos realizados y las debidas explicaciones teóricas de cada iteración.

- ✓ Construir un sistema con una interfaz gráfica de fácil manejo, eficiente y amigable al usuario, que sirva de apoyo en el estudio del Método Símplex.

- ✓ Realizar las pruebas de fallo del software resultante.

2 MARCO TEÓRICO

Se define la educación como: 'el proceso mediante el cual se transmiten conocimientos, valores, costumbres y formas de actuar'. La educación no sólo se produce a través de la palabra dada o a través de una clase magistral, está presente en todas las acciones, sentimientos y actitudes que se den, ya sea de forma directa o a través de algún medio.

El proceso de globalización y la expansión constante de la tecnología y la información en los últimos años, ha permitido que se piense en el software educativo como una nueva posibilidad para apoyar el conocimiento que se imparte en las aulas de clase. Se habla de apoyo, ya que es muy difícil que un software llegue a cubrir todas las necesidades y exigencias que deben tenerse en cuenta cuando se trata de explicar o enseñar la temática de un área en particular.

Sin embargo, si las ayudas son buenas y utilizadas de manera adecuada, se puede llegar a tener una respuesta positiva en el proceso enseñanza - aprendizaje en el que intervienen estudiantes y profesores.

A este respecto, Álvaro H. GALVIS PANQUEVA, Ricardo A. GÓMEZ CASTRO**,** y Olga MARIÑO DREWS en su trabajo *Ingeniería de Software Educativo con Modelaje Orientado a Objetos*, opinan que: "Usar la informática como apoyo a procesos de aprendizaje ha sido una inquietud que durante mucho tiempo ha sido investigada y probada por muchas personas. Su asimilación dentro de instituciones educativas, incluyendo el hogar, ha aumentado en los últimos años, con lo que la demanda por software educativo de alta calidad es cada vez mayor."[2]

SÍMPLEX MEDIA es una herramienta que se crea con la finalidad de ser un *apoyo* en la enseñanza y aprendizaje del Método Símplex. *SÍMPLEX MEDIA no es un software educativo, es un software de apoyo al estudio, y esto lo exime de realizar labores de evaluación al estudiante, en cuanto a los conocimientos adquiridos*.

[2] Pagina Web: http://www.c5.cl/ieinvestiga/actas/ribie98/184.html

La diferencia entre un *Software Educativo* y uno de *Apoyo*, radica en la Evaluación de los conocimientos que el estudiante adquiera. Tal como un libro de Física contiene ejercicios y preguntas que permiten al estudiante autoevaluarse, el *Software Educativo*, debe poseer un módulo capaz de ayudar al estudiante a comprobar sus conocimientos y evaluar su progreso. Este no es el caso de un *Software de Apoyo*, como lo es SÍMPLEX MEDIA.

Vale aclarar que no se pretende restar importancia a los conocimientos que se puedan adquirir en el aula de clase, sino más bien, entrar a reforzarlos mediante el contenido del software, el cual se enfoca en ejemplos prácticos de la vida cotidiana.

Dichos ejemplos facilitan la comprensión de la información teórica. Además, haciendo uso de su módulo Solución de Ejercicios, el usuario podrá plantear ejercicios y hacerle un seguimiento a su desarrollo, obteniendo explicaciones basadas en la misma información que presenta el software. Estas características hacen que SÍMPLEX MEDIA sea una excelente ayuda para la enseñanza y el aprendizaje del Método Símplex.

Para la realización de este proyecto se dispuso de una adecuada fuente de información que permitió explicar los conceptos teóricos que se examinan en el programa. Con el objeto de analizar, diseñar, modelar y documentar los resultados se utilizó la *Metodología Orientada a Objetos* (O.O), el *Lenguaje Unificado de Modelado* (UML) y el modelo del *Proceso Unificado* (UP). Para la construcción del aplicativo se recurrió a la jerarquía de clases de *Windows Presentation Foundation* (WPF) del *Framework 3.5* de *Microsoft .NET* haciendo uso de los lenguajes de programación *C#* y *XAML*.

En los ítems siguientes se abordan los conceptos teóricos necesarios para el desarrollo del proyecto, tanto la información que se explica al estudiante, como los procesos metodológicos, de desarrollo y codificación.

2.1 CONTENIDO TEÓRICO DE SÍMPLEX MEDIA

Al igual que en cualquier estructura arquitectónica, en un proceso educativo es importante cimentar bien las bases, para que el conocimiento se construya de manera adecuada, sin dejar dudas que puedan confundir al estudiante en los procesos que posteriormente se desarrollen.

Es por esta razón que en el software SÍMPLEX MEDIA, se tocan temas como Matrices, Ecuaciones Lineales y Sistemas de Ecuaciones Lineales; temas de importancia para la

comprensión de los algoritmos del Método Símplex de la Programación Lineal. Sin embargo, no son el fuerte del software, ya que la verdadera importancia de este radica en el aprovechamiento del método en sí, y no en los procesos matemáticos e iterativos que deben realizarse para lograr el resultado final.

A continuación se realiza una breve descripción de los temas principales que se analizan en el software.

2.1.1 Matrices. En la vida diaria es necesario enfrentar problemas en los que los cálculos matemáticos son imprescindibles y hay diferentes maneras para plantear y resolver un mismo problema. Una forma útil para representar y plantear un problema de la vida real es hacer uso de las matrices. Aunque en principio manejarlas y operar con ellas parezca un tanto difícil, son una herramienta importante que permite expresar, formular, modelar y hasta resolver, algunos problemas con gran facilidad.

Las matrices se han utilizado desde mucho tiempo atrás, existen algunos datos que llevan a pensar que se trabajó con ellas en culturas como la china. Prueba de esto, son algunos textos tomados de la obra *Los Nueve Capítulos*, en la cual se muestra ejemplos en donde se realiza la solución de problemas utilizando sistemas de ecuaciones lineales, que se basan en el uso de las matrices.

Las matrices como tal se conocieron en el siglo XIX; el primero en utilizar el término "*matriz"* fue *James Joseph Sylvester* en el año de 1850. La definición matemática que se realizó para ellas es la siguiente:

Se llama matriz de orden $m \times n$ a todo conjunto rectangular de elementos a_{mn} dispuestos en m líneas horizontales (filas) y n verticales (columnas) de la forma:

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \dots & a_{3n} \\ \dots & \dots & \dots & \dots & \dots \\ a_{m1} & a_{m2} & a_{m3} & \dots & a_{mn} \end{bmatrix}$$

Cada elemento dentro de la matriz se identifica con una letra minúscula (normalmente la misma del nombre) y su posición con dos subíndices, el primero, que identifica la posición del elemento en la fila, y el segundo, su posición en la columna.

Dependiendo de sus características las matrices pueden clasificarse en diferentes *tipos*: matriz fila, matriz columna, matriz cuadrada, matriz diagonal, matriz identidad, matriz triangular, matriz nula, matriz aumentada, matriz traspuesta etc. La utilidad de estas radica en la facilidad con la cual se pueden llegar a desarrollar operaciones, que sin la ayuda de ellas, pudiesen tornarse complicadas.

Dentro de las *operaciones* que pueden realizarse utilizando matrices están: la suma, la resta, la multiplicación de una matriz por un número y la multiplicación de dos matrices.

¿Qué relación tienen las matrices con SÍMPLEX MEDIA? Las matrices servirán para representar matemáticamente un conjunto de valores al cual se le conoce como *Conjunto de Restricciones*. Estas restricciones, que pueden ser ecuaciones o inecuaciones, deben visualizarse en matrices para poderlas manipular y resolver el problema planteado. Además, la matriz con el *conjunto de restricciones*, irá adquiriendo ciertas características que, dependiendo del *tipo de matriz* que sea, facilitan la comprensión de los resultados.

2.1.2 Ecuaciones e inecuaciones lineales. Este tipo de expresiones se utilizan generalmente cuando es necesario representar de forma matemática problemas cotidianos que llevan implícita alguna relación de comparación. A esta relación se le denomina igualdad.

Una igualdad es una expresión matemática que consta de dos miembros asociados por el símbolo "=". Un ejemplo de esto sería el siguiente enunciado: "Ana tiene 23 años y Sofía tiene 19. La suma de estas edades es *igual* a 42 años".

La igualdad que representa este enunciado es: $23 + 19 = 42$.

Al hablar de ecuaciones no se puede dejar de lado las inecuaciones, las cuales parten del mismo fundamento, solo que el signo a través del cual se asocian sus miembros puede ser: menor "<", mayor ">", menor igual "<=" o mayor igual ">=". En cualquiera de los casos el resultado será una expresión matemática que estará formada por dos miembros unidos por un signo de relación.

Tanto a las ecuaciones como a las inecuaciones se les han asignado una serie de propiedades, las cuales hay que tener en cuenta cuando se quiera trabajar con ellas.

Los gráficos suelen servir para modelar mucha de la información que se extrae del mundo real; de manera que llegan a ser útiles para hacer representaciones menos complejas de ideas abstractas. Uno de los casos más comunes es la estadística; la cual se apoya en una serie de formas para representar grandes cantidades de datos, con lo que se hace posible la fácil absorción de los conceptos y la adecuada aplicación de los procedimientos. De igual manera, en el caso de las ecuaciones y en el de las inecuaciones, los gráficos brindan mucha ayuda en la comprensión de los conceptos matemáticos y en la utilización de éstas en los diferentes métodos y algoritmos de solución a problemas de Programación Lineal y de otras áreas.

2.1.3 Sistemas lineales. Los sistemas lineales están formados por 2 o más ecuaciones o inecuaciones de primer grado con dos o más incógnitas, que deben ser resueltas en forma simultánea.

Están constituidos por coeficientes (a_{mn}), términos independientes (b_m) e incógnitas (x_n). En donde los coeficientes y los términos independientes son constantes. Lo anterior se puede representar así:

$$\begin{array}{rcrcrcrcr} a_{11}x_1 & + & a_{12}x_2 & + & \dots & a_{1n}x_n & = & b_1 \\ a_{21}x_1 & + & a_{22}x_2 & + & \dots & a_{2n}x_n & = & b_2 \\ \dots & & \dots & & \dots & \dots & \dots & \dots \\ a_{m1}x_1 & + & a_{m2}x_2 & + & \dots & a_{mn}x_n & = & b_m \end{array}$$

Los sistemas de ecuaciones pueden llegar a tener tantas ecuaciones y variables como sea necesario. Si se trata de un sistema de dos ecuaciones con dos incógnitas se resuelven de forma analítica usando métodos algebráicos como eliminación por igualación, eliminación por sustitución o método de reducción.

7x + 4y = 13
5x - 2y = 19

Pero, si se trata de ecuaciones con tres o más incógnitas existen varios métodos; los más conocidos son Gauss y Gauss Jordan.

¿Qué relación tienen los Sistemas Lineales con SÍMPLEX MEDIA? La solución de Sistemas Lineales mediante el método de Gauss Jordan es la base de las *Operaciones de Renglones*

(u Operaciones de Gauss Jordan) que se efectúan en las iteraciones de los algoritmos del Método Símplex.

2.1.4 Programación lineal. Para tener una idea de cómo nació la Programación Lineal, se hablará primero de la "Investigación de Operaciones". Ésta, tuvo sus orígenes mucho tiempo atrás. Sin embargo, su mayor crecimiento se logró a partir de la Segunda Guerra Mundial, época en la cual se utilizó para operaciones militares, que es de donde nace su nombre. Posteriormente su uso se extendió a un sinnúmero de áreas del conocimiento, para lo cual se dividió en ramas como: Programación Lineal, Programación No Lineal, Programación Dinámica, Simulación, Teoría de Colas, etc. En éstas, el objetivo es encontrar la mejor solución o curso de acción, a determinado problema que pueda presentarse en las actividades cotidianas.

Para poder llevarse a cabo un análisis mediante la investigación de operaciones, es necesario definir el problema, posteriormente; mediante los datos que arroje este, se construye un *Modelo*, al cual se da solución utilizando alguno de los algoritmos, según la rama a la que pertenezca. Existen diferentes tipos de modelos: el modelo matemático, el modelo de simulación, los modelos de la ciencia de la administración, etc.

Como puede verse el área de acción de la investigación de operaciones, es muy amplia, por esta razón el tema a tratar se centra particularmente en la *Programación Lineal*.

De ésta se puede decir que nació en 1939, cuando se concedió el premio Nobel a L. Kantorovich, y T. Koopmans, por el desarrollo de la teoría matemática llamada *"Programación Lineal"*.

En la frase Programación Lineal, la palabra *programación* se concibió como un sinónimo de *planeación*, la cual se sustentaría en procesos desarrollados utilizando funciones lineales, para obtener resultados óptimos en situaciones específicas.

Dicho de otra manera, la *programación Lineal*, es una técnica matemática que reúne una serie de métodos, que permiten resolver problemas en los cuales el objetivo fundamental es la optimización de una meta preestablecida, dependiendo de las condiciones que se presenten, esta técnica puede utilizarse en cualquiera de las áreas del conocimiento de manera similar, teniendo en cuenta una metodología que reúne los siguientes pasos:

- ***Definición del problema*:** este es un proceso de gran importancia, ya que afectará el posterior desarrollo del problema. Se debe hacer una descripción detallada de los datos y objetivos del problema.

- ***Formulación de un modelo matemático:*** esta es una aproximación matemática al problema que se está analizando, debe ser sencilla e incluir todas las consideraciones que permitan evaluar las alternativas de solución.

- ***Obtención de una solución a partir del modelo***: en primer lugar se deben estudiar las características del modelo matemático, para seleccionar el método con el cual se desarrollará el problema, que puede ser matemático, de simulación, etc. Una vez seleccionado el modelo, la solución se centra en encontrar los valores de las variables independientes, asociadas a los componentes controlables del sistema, con el fin de optimizar su efectividad teniendo en cuenta las restricciones planteadas.

- ***Prueba del modelo***: una vez encontrada la solución debe ser estudiada minuciosamente, con el fin de corregir las fallas que puedan presentarse.

- ***Validación del modelo***: esto se logra analizando la consistencia de las dimensiones de las unidades que se hayan empleado en el desarrollo del problema.

- ***Establecimiento de controles sobre la solución***: a este paso también se denomina "*Análisis de Sensibilidad*", y consiste en probar la solución óptima del problema, variando los parámetros en un rango de datos determinado, mientras no cambie la solución del problema.

- ***Implantación de la solución***: una vez hallados los resultados finales, la decisión, la forma en la cual se apliquen, depende directamente de los responsables del problema***.***

2.1.5 Método símplex. Existen acontecimientos en la historia que sorprenden por su magnitud. En el caso de las matemáticas, tal vez la aparición del Método Símplex sea uno de ellos. En un momento que pareciese que en cuestión de matemáticas ya estaba todo dicho, aparece un grupo de personas con una idea novedosa, la cual nace en medio de un momento crítico como es la guerra.

El autor del Método Símplex es George Dantzing, quien nació en Oregón en 1914, hijo de inmigrantes de origen ruso, estudió Matemáticas en la Universidad de Maryland. Poco después de doctorarse por la Universidad de Berkeley, en 1947, formuló el enunciado estándar de un problema general de Programación Lineal y desarrolló el Método Símplex. De hecho, estos estudios son una consecuencia de su trabajo como experto en métodos de planificación para las Fuerzas Aéreas estadounidenses, que resolvía utilizando calculadoras de mesa.

Stanley Grossman habla sobre el nacimiento del Método Símplex. En su artículo cita apartes de una entrevista realizada a George Dantzing, en la cual, menciona una de las últimas anotaciones sobre la descripción del proceso que le llevó al descubrimiento del Método Símplex: "Cuando formulé por primera vez un problema de programación lineal, lo hice sin una función objetivo. Estuve luchando por algún tiempo con la adición de reglas básicas para elegir entre las soluciones factibles a la que en algún sentido fuese óptima. Pero, pronto abandoné esa idea y la sustituí por la de una función objetivo a ser maximizada. El modelo que formulé no estaba hecho específicamente para fines militares. Podía aplicarse a toda clase de problemas de planeación económica lo mismo que a un problema de planeación industrial"[3].

La primera formulación del Método Símplex fue en el verano de 1947. El primer problema práctico que se resolvió con este método fue uno de nutrición.

Como lo explica Dantzing, después de la guerra este método se empezó utilizar en la industria y la economía. De esto han pasado más de 60 años y aún sigue vigente; esta es una muestra de cómo la investigación científica encabeza los fenómenos del progreso o estancamiento de un país.

[3] Grossman, Stanley. Aplicaciones del Algebra Lineal. Reseña sobre George Dantzing y la historia de la Programación Lineal.

Sin temor a equivocarse se puede decir que el Método Símplex es un eficiente método analítico para la solución de modelos de Programación Lineal. Al conocer los fundamentos del Método Símplex y la manera como opera, el estudiante está en capacidad de usarlo eficientemente obteniendo ventajas de sus potencialidades.

Por esta razón, se ha analizado un poco más el Método Símplex y los algoritmos que lo conforman. El Método Símplex, está compuesto por el *Método Símplex Primal* y el *Método Símplex Dual*. El *Método Símplex Primal* está compuesto, a su vez, por tres algoritmos: Símplex Primal, Técnica M y Dos fases.

Cada uno de estos se enfoca en optimizar un objetivo, el cual se encuentra sujeto a unas restricciones.

Estas restricciones y función objetivo se obtienen del planteamiento inicial del problema. Para poder iniciar los cálculos que llevarán a obtener un resultado, es necesario organizar estos datos a una *Forma Estándar de Programación Lineal.*

2.1.5.1 Forma estándar de programación lineal.

En el proceso para llevar un problema a la Forma Estándar de programación lineal, es necesario tener en cuenta que debe cumplir con tres propiedades:

1. Todas las restricciones deben estar expresadas en forma de ecuaciones con su lado derecho no negativo.
2. Todas las variables deben ser no negativas.
3. La "Función Objetivo" debe ser de Maximización o de Minimización y debe ser afectada por las variables introducidas en las restricciones.

Estas propiedades hacen que las ecuaciones lineales y función objetivo sufran algunas transformaciones a través de las cuales quedan listas para iniciar el proceso con los algoritmos, dicha transformación depende de cómo estén planteadas. Así, por ejemplo, si se trata de restricciones en las cuales el signo de relación es "<=", al aplicar la primera propiedad éstas quedan convertidas en igualdades.

La igualdad se obtiene al agregar del lado izquierdo de la desigualdad una variable denominada "Variable de Holgura" que representa el valor que le haría falta para ser igual al derecho. Con restricciones de tipo "<=" se suele hacer referencia al consumo máximo de un recurso. Por ello, simulando el hecho de completar el valor de la derecha, se

adiciona la “Variable de Holgura”, representada por la letra “**h**” seguida de un subíndice consecutivo que va desde 1 hasta el número de variables adicionadas al problema.

En esa misma propiedad, se establece que el lado derecho de las ecuaciones no debe ser negativo. En caso que una de las restricciones incumpla esto, debe ser multiplicada por -1. Con esta acción el signo de relación cambiará y según sea éste, se añadirá una “Variable de Holgura” o una de “Superávit”, como se analiza más adelante.

La segunda propiedad expresa que todas las variables deben ser no negativas. A fin de adecuar el modelo al cumplimiento de esta propiedad, es necesario especificar el hecho de que todas las variables, tanto las del problema, como las adicionadas, sean mayores o iguales a 0. Con el propósito de indicar esto será necesario añadir restricciones adicionales, en las cuales, como en el ejemplo que se muestra a continuación, todas las variables deben ser no negativas, esto es, que tomen valores positivos o cero.

Ejemplo:

$$x_1, x_2, x_3, \mathbf{h_1}, \mathbf{h_2}, \mathbf{h_3} \text{ y } \mathbf{h_4} >= 0.$$

La tercera propiedad hace referencia a que todas las variables que adicionaron en las restricciones deben estar presentes también en la “Función Objetivo”. Además, dicha función debe establecer si el problema es de maximización o minimización. A continuación se muestra un ejemplo en el cual se indica la función objetivo original de un problema y posteriormente la función objetivo del mismo problema pero con las variables que se le adicionaron a las restricciones.

Función objetivo original: $Z = x_1 + x_2 + x_3$ para Maximizar

Función objetivo final: $Z = x_1 + x_2 + x_3 + \mathbf{0h_1} + \mathbf{0h_2} + \mathbf{0h_3} + \mathbf{0h_4}$ para Maximizar

Sin importar el valor que adquieran las “Variables de Holgura”, al multiplicarlas por 0, no afectarán el resultado de la “Función Objetivo”.

Después de aplicadas las propiedades se inicia el proceso con el algoritmo que, para este caso, es el Símplex Primal.

Pero no siempre las restricciones serán de tipo “<=”. En caso de que sean de tipo “>=” o “=”, o una combinación de los dos tipos se adicionan otro tipo de variables.

Si se trata de ecuaciones en donde el signo de relación es ">=", las expresiones hacen referencia a lo obligatorio de consumir o utilizar, como mínimo, cierta cantidad de un producto o material, en este caso se adicionan las variables de "Superávit", las cuales generalmente se designan con la letra "**s**" y un subíndice dependiendo del número de variables de este tipo que se deban adicionar.

A diferencia de las "Variables de Holgura", con las "Variables de Superávit", en realidad, no se hace una adición, sino una sustracción. Estas variables deben ser restadas a las restricciones que tengan el singo de relación de tipo ">=".

Una vez que todas las ecuaciones queden convertidas en igualdades, es necesario pasar los datos a una matriz que estará dividida en tres partes, o tres matrices concatenadas de la siguiente manera: una primera matriz compuesta por los coeficientes de las variables x_i en todas las restricciones (éstas son las que conforman el planteamiento inicial del problema); luego una segunda matriz creada a partir de los coeficientes de las variables adicionales, es decir, las holguras y las superávit añadidos; y, por último, una tercera matriz de una sola columna compuesta con los valores que se encuentran del lado derecho de las igualdades.

La segunda matriz de la cual se está hablando debe convertirse en una *Matriz Identidad*, para que el problema pueda resolverse utilizando la lógica del método de eliminación de Gauss Jordan.

Debido a estas razones matemáticas, es necesario introducir otro tipo de variables que permitan completar la *Matriz Identidad*. Estas variables se denominan "Variables Artificiales" y se representan con la letra "**a**" seguidas de un subíndice consecutivo que irá desde 1 hasta el número de variables utilizadas y se adicionan de igual manera que las "Variables de Holgura".

Las "Variables Artificiales" deben ser adicionadas en las restricciones que, originalmente son de tipo "=" y las que sigan siendo de tipo ">=", en caso que hayan sido multiplicadas por -1.

Las "Variables Artificiales" también deben ser añadidas a la "Función Objetivo" (se restan a ella), pero no se multiplican por 0, y si el problema será resuelto utilizando la Técnica M, cada variable artificial de la "Función Objetivo" debe ser multiplicada por el coeficiente M.

Una vez organizados los datos en la forma estándar de programación lineal, están listos para ser procesados mediante el algoritmo del Método Símplex que le corresponda. Como anteriormente se dijo éste método está conformado por tres algoritmos. Los problemas en los cuales *todas* las restricciones sean de tipo "<=" pueden ser resueltos mediante el algoritmo Símplex Primal. Los problemas en los cuales las restricciones son de tipo ">=" e "=" o combinaciones de estos tipos se resuelven utilizando el algoritmo de la Técnica M o el algoritmo de Dos Fases, y algunos de ellos pueden ser solucionados utilizando el Algoritmo del Método Símplex Dual.

A continuación se mencionan los pasos de cada uno de los algoritmos.

2.1.5.2 Algoritmo símplex primal

Cada algoritmo tiene su propia forma de llegar a la solución del problema propuesto, el Símplex Primal, está compuesto de una serie de pasos que llevan a la solución del problema. Los pasos son:

1. Transformar el modelo en la Forma Estándar de Programación Lineal.
2. Pasar este modelo a la Tabla Símplex Inicial.
3. Encontrar la "Solución Inicial".
4. Verificar la Optimalidad, es decir, el valor que adquiere la función objetivo. Si los coeficientes de las "Variables de Decisión" de la fila Z de la "Tabla Símplex" son todos positivos, en el caso de maximización (o negativos en el caso de minimización), terminar la ejecución del proceso. En caso contrario, ir al paso 5.
5. Seleccionar la "Variable de Entrada" y la "Variable de Salida".
6. Seleccionar un "Elemento Pivote" para realizar operaciones aritméticas.
7. Realizar la "Sustitución de Variables".
8. Volver al paso 4.

2.1.5.3 Algoritmo de la técnica M

El Algoritmo de la Técnica M ó Método de la M, es el segundo de los tres algoritmos del Método Símplex Primal. Esta técnica permite solucionar problemas en los que no es aplicable el Algoritmo Simplex Primal, ya que aparecen restricciones que no son todas reducibles a la forma "<=" (Permanecen combinaciones de "<=", ">=" e "=").

Este algoritmo plantea el uso de un coeficiente adicional llamado M, del cual se deriva su nombre, que permitirá abordar estos problemas, pues solo usando el Algoritmo Símplex

Primal sería imposible. M es una cantidad “grande”, con respecto a los demás parámetros que aparecen en el problema.

Por ser parte del Método Símplex Primal, la técnica M sigue los mismos pasos que el Algoritmo Símplex Primal, con la diferencia de que la forma estándar varía debido al tipo de restricciones que están presentes en el modelo de programación lineal. Dicha variación no afecta los pasos siguientes del algoritmo, pero si cambia la manera de interpretar la Tabla Símplex. Los pasos para la Técnica M son:

1. Transformar el modelo en la Forma Estándar de la Técnica M.
   1.1. Convertir el modelo en la Forma Estándar de Programación Lineal.
   1.2. Representar matricialmente el modelo en su forma estándar.
   1.3. Llevar la Función Objetivo a la Forma Estándar de la Eliminación Gaussiana.
2. Pasar este modelo a la Tabla Símplex Inicial.
3. Encontrar la “Solución Inicial”.
4. Verificar la Optimalidad, es decir, el valor que adquiere la función objetivo. Si los coeficientes de las “Variables de Decisión” de la fila Z de la “Tabla Símplex” son todos positivos, en el caso de maximización (o negativos en el caso de minimización), terminar la ejecución del proceso.
5. Seleccionar la “Variable de Entrada” y la “Variable de Salida”.
6. Seleccionar un “Elemento Pivote” para realizar operaciones aritméticas.
7. Realizar la “Sustitución de Variables”.
8. Volver al paso 4.

2.1.5.4 Algoritmo de dos fases

Al usar la Técnica M para resolver problemas de Programación Lineal que tengan restricciones que no puedan ser modificables a “<=” sin que el lado derecho de ellas sea negativo, se suele presentar un error de redondeo en los cálculos generados por la computadora. Este error producido por el manejo conjunto de números grandes (los valores de la M) y números pequeños (el de las demás variables) afectará el resultado de la “Solución Optima” en cuanto a exactitud. El *Algoritmo de Dos Fases*, está diseñado para mitigar este problema, eliminando por completo la constante M, aún cuando se haga uso de “Variables Artificiales”.

Tal como en el caso de los demás algoritmos, éste cuenta con un listado de pasos que facilitan su ejecución, y debido a que pertenece a la serie de algoritmos del Método Símplex Primal, sus pasos son muy similares a los de la Técnica M y Símplex Primal. En este

caso los pasos son más numerosos debido a que se cuenta con dos fases o procedimientos.

Para la Primera Fase los pasos son:

1. Transformar el modelo en la "Forma Estándar de la Primera Fase".
   1.1. Convertir el modelo en la Forma Estándar de Programación Lineal.
   1.2. Formar una **Nueva** "Función Objetivo" de *Minimización* a partir de las "Variables Artificiales".
   1.3. Representar matricialmente el **Nuevo** modelo en su forma estándar.
   1.4. Llevar la **Nueva** Función Objetivo a la Forma Estándar de la Eliminación Gaussiana.
2. Pasar el **Nuevo** modelo a la Tabla Símplex Inicial.
3. Encontrar la "Solución Inicial".
4. Verificar la Optimalidad, es decir, el valor que adquiere la función objetivo. Si los coeficientes de las "Variables de Decisión" de la fila **R** de la "Tabla Símplex" son todos negativos y **R** es diferente de 0 terminar la ejecución del proceso de *todo el algoritmo*; si estos coeficientes son negativos y **R** es igual a 0 se pasa a la *Segunda Fase*.
5. Seleccionar la "Variable de Entrada" y la "Variable de Salida".
6. Seleccionar un "Elemento Pivote" para realizar operaciones aritméticas.
7. Realizar la "Sustitución de Variables".
8. Volver al paso 4.

En el caso de la Segunda Fase, los pasos son:

1. Transformar el modelo en la "Forma Estándar de la Segunda Fase".
   1.1. Eliminar las "Variables Artificiales" no Básicas del "Diagrama Terminal" de la Primera Fase.
   1.2. Crear un **Nuevo** modelo a partir del Diagrama Terminal de la Primera Fase y de la "Función Objetivo" del planteamiento original del problema.
   1.3. Representar matricialmente el **Nuevo** modelo en su forma estándar.
   1.4. Llevar la Función Objetivo a la Forma Estándar de la Eliminación Gaussiana.
2. Pasar este modelo a la Tabla Símplex Inicial.
3. Encontrar la "Solución Inicial".
4. Verificar la Optimalidad, es decir, el valor que adquiere la función objetivo. Si los coeficientes de las "Variables de Decisión" de la fila Z de la "Tabla Símplex" son

todos positivos, en el caso de maximización (o negativos en el caso de minimización), terminar la ejecución del proceso. En caso contrario, ir al paso 5.

5. Seleccionar la “Variable de Entrada” y la “Variable de Salida”.
6. Seleccionar un “Elemento Pivote” para realizar operaciones aritméticas.
7. Realizar la “Sustitución de Variables”.
8. Volver al paso 4.

2.1.5.5 Algoritmo del método símplex dual

El *Método Símplex Dual*, es un algoritmo rápido y eficiente que permite dar solución a problemas de Programación Lineal. Su objetivo es hallar una Solución Óptima Inicial, pero infactible, y partiendo de ésta, realizar las iteraciones necesarias hasta hallar una que sea factible.

El algoritmo que usa el Método Símplex Dual tiene la misma estructura que los algoritmos del Método Símplex Primal. Los pasos son:

1. Transformar el modelo en la Forma Estándar del Método Símplex Dual.
2. Pasar la forma estándar a la Tabla Símplex Inicial.
3. Encontrar la Solución Inicial.
4. Verificar la Factibilidad, es decir, el valor que adquieren las Variables Básicas. Cuando en la columna Solución de la Tabla Símplex todos los valores son no negativos el proceso termina. En caso contrario, ir al paso 5.
5. Seleccionar la Variable de Salida y la Variable de Entrada.
6. Seleccionar un Elemento Pivote para realizar operaciones aritméticas.
7. Realizar la Sustitución de Variables.
8. Regresar al paso 4.

La “Forma Estándar del Método Símplex Dual”, que se menciona en el primer paso, es similar a la Forma Estándar de Programación Lineal, pero con la excepción de que no usa Variables Artificiales, y multiplica por -1 las restricciones en las que se han agregado Variables de Superávit. En esta Forma Estándar no es un impedimento el que el lado derecho de las ecuaciones sea negativo, más bien, es un requisito el que al menos en una de ellas exista el negativo en el lado derecho.

Además, para poder utilizar el Método Símplex Dual, se deben cumplir con unas condiciones iniciales, estas son: ninguna de las restricciones del modelo inicial debe ser de tipo “=”; en el modelo inicial no todas las restricciones deben ser de tipo “<=”; y tras pasar

el modelo a la Forma Estándar del Método Símplex Dual, el lado derecho de al menos una de las ecuaciones debe ser negativo.

Como puede notarse en este documento los algoritmos están estructurados en una serie de 8 pasos, la mayoría de los cuales coinciden en los algoritmos, esto ayuda a que el estudiante vea de manera global al Método Símplex Primal, facilitando así la comprensión de su funcionamiento.

2.2 METODOLOGÍA PARA EL DESARROLLO DE SÍMPLEX MEDIA

SÍMPLEX MEDIA fue desarrollado siguiendo los estándares de modelado que plantea UP. Para comprender sus lineamientos es necesario conocer, manejar y poder utilizar los diagramas del Lenguaje Unificado de Modelado (UML) que, a su vez, se basa en el paradigma de programación Orientado a Objetos. Seguidamente se explicará en qué consisten cada una de estas herramientas del análisis, diseño y modelado de sistemas computacionales.

2.2.1 Análisis y diseño de sistemas usando la metodología orientada a objetos (O.O). Las metodologías de Ingeniería de Software tienen mecanismos para hacer análisis de necesidades y diseño de sistemas complejos, y han evolucionado con la tecnología en lo relacionado con el diseño computacional.

Así, por ejemplo, el análisis estructurado se basa en la descomposición en funciones o procesos. La metodología mas reciente (Orientada a objetos), se centra en el análisis usando una descomposición del sistema por objetos o conceptos.

De acuerdo a Joseph Schmuller, “Los objetos concretos y virtuales, están a nuestro alrededor, ellos conforman nuestro mundo. El software actual simula al mundo (o un segmento de él), y los programas, por lo general, imitan los objetos del mundo. Si comprende algunas cuestiones básicas de los objetos, entenderá cómo se deben mostrar estos en las representaciones de software.

Antes que nada, un objeto es la instancia de una clase (o categoría). Usted y yo, por ejemplo, somos instancias de la clase persona. Un objeto cuenta con una estructura, es decir atributos (propiedades) y acciones. Las acciones son todas las actividades que el

objeto es capaz de realizar. Los atributos y acciones, en conjunto, se conocen como características o rasgos".[4]

Schmuller define a los objetos no solo como aquello que tiene masa, sino también habla de la posibilidad de encontrar objetos virtuales, que no pueden tocarse, como por ejemplo una dirección de correo electrónico.

La metodología O.O. trae una serie de ventajas. Por ejemplo: el suministro de modelos similares a los del mundo real, la facilidad para manejar sistemas complejos y la reutilización de los objetos, permite además el desarrollo iterativo de las aplicaciones y también facilita la interoperabilidad de las mismas.

Según lo que menciona Schmuller un objeto cuenta con atributos y acciones.

Dicho de otro modo, los **Atributos** pueden considerarse como un estado, o conjunto de propiedades inherentes a él. Las acciones pueden catalogarse como un conjunto de operaciones o **Métodos**. Pero, además, los objetos tienen también, una **Identidad**, es decir, un identificador que los hará únicos.

Desde un punto de vista enfocado a la programación puede decirse que un objeto es una parte de software que cumple con ciertas características como:

- ***Herencia*:** una clase (clase derivada) puede adoptar los atributos y métodos de una clase superior (superclase), de esta manera, cuando se crea un objeto como instancia de una clase, éste tendrá todas las características tanto de la clase de la cual es instancia, como de la superclase de la cual ésta hereda.

- ***Polimorfismo***: hace referencia a como una operación puede tener el mismo nombre en una misma clase y, sin embargo, puede realizar diferentes procesos. A esto se le conoce como *Sobrecarga de Métodos*. La clave está en los valores de entrada que reciben dichas operaciones.

  El polimorfismo también se presenta en el momento en que una superclase define una operación y sus clases derivadas redefinen los procesos que se ejecutan dentro de ella. Así, al llamar a determinada operación redefinida, se ejecuta la

[4] Schmuller, Joseph. Aprendiendo UML en 24 horas, página: 39

redefinición y no la definición original en la superclase. Esto recibe el nombre de *Redefinición de Métodos*.

- ***Encapsulamiento***: esta característica de los objetos hace que se oculten sus funcionalidades internas, así solo se encargarán de realizar las acciones que le correspondan sin que el usuario se dé cuenta de qué es lo que pasa en su interior. esta característica hace que se protejan los datos de accesos indebidos, la forma de protegerlos es haciendo que los objetos puedan ser privados, protegidos o públicos.

- ***Abstracción***: existen diferentes mecanismos de abstracción, como por ejemplo composición, clasificación, agrupación y especialización.

Una Clase es la unidad básica que encapsula toda la información de un objeto. A través de ellas se puede modelar el entorno, o la categoría de un objeto. Todos los objetos se crean por instancia de las clases.

2.2.2 El lenguaje unificado de modelado (UML). Un modelo puede considerarse como la abstracción de un sistema del mundo real, este modelo se encarga de describir completamente los aspectos del sistema que son relevantes para un nivel de abstracción dado, teniendo en cuenta los detalles importantes del sistema real.

La necesidad de nuevas formas de diseño ha dado pie a la creación de nuevas herramientas que permitan a los diseñadores de sistemas trabajar de forma que los demás comprendan las ideas de sus modelos.

UML es la herramienta más utilizada para el modelado de sistemas. Nació en la empresa Rational Software Co, quien se puso en la tarea de crear una notación unificada en la qué basar la construcción de sus herramientas CASE. En el proceso de creación de UML han participado, empresas de gran peso en la industria como Microsoft, Hewlett-Packard, Oracle e IBM, así como grupos de analistas y desarrolladores; y su desarrollo fue encabezado por: Grady Booch, Ivar Jacobson y Jim Rumbaugh.

Uno de los objetivos principales de la creación de UML era posibilitar el intercambio de modelos entre las distintas herramientas CASE orientadas a objetos del mercado.
Pero, específicamente ¿Qué es UML?

Según Grady Booch, James Rumbaugh e Ivar Jacobson, "El lenguaje unificado de modelado (Unified Modelin Language, UML) es un lenguaje estándar para escribir planos de software. UML puede utilizarse para visualizar, especificar, construir, y documentar los artefactos de un sistema con gran cantidad de Software".[5]

UML combina notaciones provenientes del modelo orientado a objetos, el modelo de datos, el modelo de componentes y modelo de flujos de trabajo, y la principal característica es que está dirigido por casos de uso y se centra en la arquitectura.

Esta combinación da origen a tres clases de bloques de construcción que son:

- ***Elementos***: los elementos son abstracciones de cosas reales o ficticias como objetos o acciones.

- ***Relaciones***: formas de relacionar los *elementos* entre sí.

- ***Diagramas***: son colecciones de *elementos* con sus *relaciones*.

Profundizando un poco más en los diagramas. Un diagrama es la representación gráfica de un conjunto de elementos con sus relaciones, ofrece una vista del sistema a modelar. Para poder representar correctamente un sistema, UML ofrece una amplia variedad de diagramas para visualizar el sistema desde varias perspectivas.

Los diferentes tipos de diagramas que se utilizan en UML son:

- *Diagrama de Casos de uso*
- *Diagrama de clases*
- *Diagrama de Objetos*

Diagramas de Comportamiento

- *Diagrama de Estados*
- *Diagrama de Actividad*

[5] Booch, Grady; Rumbaugh, James; Jacobson, Ivar. El Lenguaje unificado de modelado, Presentación de UML. Capitulo 2.

Diagramas de Interacción

- *Diagrama de secuencia*
- *Diagrama de colaboración*

Diagramas de Implementación

- *Diagrama de Componentes*
- *Diagrama de Despliegue*

Diagramas: Los diagramas más utilizados en el modelamiento de sistemas como SÍMPLEX MEDIA son:

- ***Diagrama de Casos de Uso***: Muestra la relación entre los actores y los casos de uso del sistema. Representa la funcionalidad que ofrece el sistema en lo que se refiere a su interacción externa. Se define un caso de uso como cada interacción supuesta con el sistema a desarrollar, donde se representan los requisitos funcionales. Es decir, se está diciendo lo que tiene que hacer un sistema y cómo.

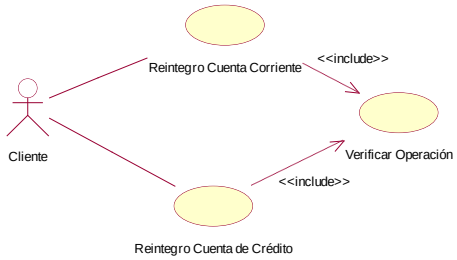


Ilustración 2-1 Representación gráfica de un Diagrama de Casos de Uso

- ***Diagrama de clases:*** Muestra un conjunto de clases, interfaces y sus relaciones, es el diagrama más común a la hora de escribir un diseño de los sistemas orientados a objetos.

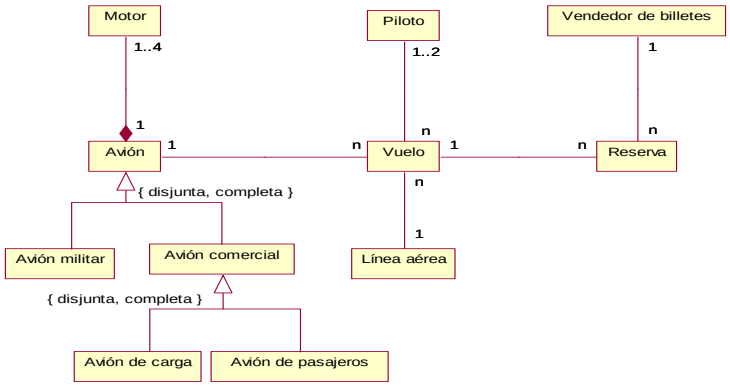


Ilustración 2-2 Representación gráfica de un Diagrama de clases

- ***Diagrama de secuencia:*** Se muestra en la interacción entre los actores y los objetos que componen un caso de uso o una operación.

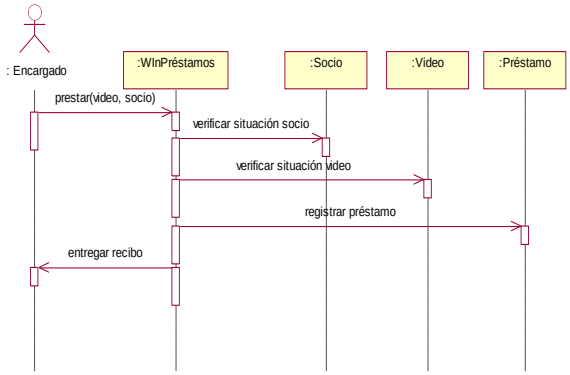


Ilustración 2-3 Representación gráfica de un Diagrama de secuencia

- ***Diagrama de colaboración:*** Es un diagrama de interacción que resulta de la organización estructural de los objetos que envían y reciben mensajes. Un diagrama de colaboración muestra un conjunto de objetos, enlaces entre ellos y los mensajes enviados y recibidos por ellos. Los objetos son normalmente instancias de clases, aunque también pueden representar instancias de otros elementos como colaboraciones, componentes y nodos. Los diagramas de colaboración se utilizan para describir la vista dinámica de un sistema.

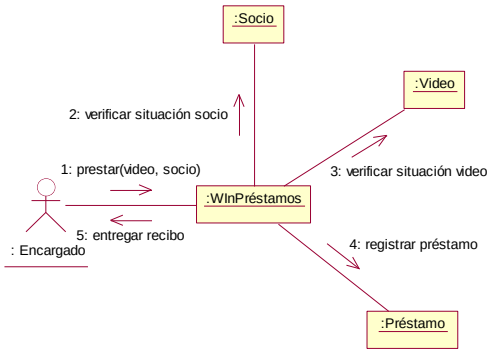


Ilustración 2-4 Representación gráfica de un Diagrama de colaboración

Los diagramas expresan gráficamente las partes de un modelo. Un modelo es una abstracción del sistema centrada en una estructura semántica determinada. Los modelos están compuestos por un conjunto de diagramas según lo establezca la metodología de modelamiento seleccionada. Por ejemplo, en el Proceso Unificado, el Modelo de Análisis cuenta con diagramas de clases y de colaboración.

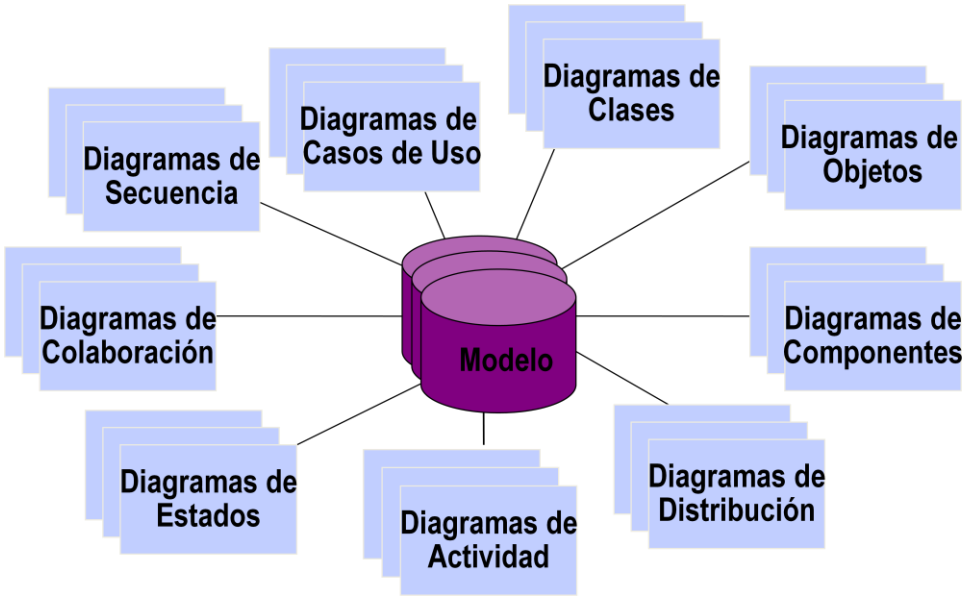


Ilustración 2-5 Representación de los diagramas que componen un Modelo

Todos estos diagramas deben organizarse en paquetes. Un ***paquete*** es un mecanismo de propósito general para organizar elementos en grupos; elementos estructurales, elementos de comportamiento, etc. Un paquete es puramente conceptual, es decir que

solo existe en el tiempo de desarrollo. Gráficamente un paquete se visualiza como una carpeta, incluyendo normalmente su nombre y en ocasiones su contenido.

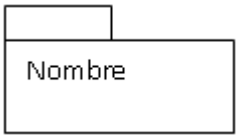


Ilustración 2-6 Representación gráfica de un Paquete

Relaciones: En UML existen cuatro tipos de relaciones: dependencia, asociación, generalización y realización.

- ***Dependencia***: es una relación semántica entre dos elementos. en la cual el cambio de un elemento puede afectar la semántica de otro, se representa gráficamente de la siguiente manera:

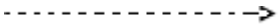


Ilustración 2-7 Representación gráfica de una relación tipo Dependencia

- ***Asociación***: es una relación estructural que describe cómo una clase está asociada a otra al tener en sus atributos objetos de dicha clase. Existe una relación de asociación especial denominada agregación.

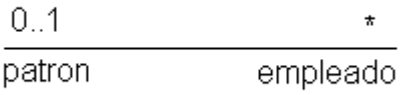


Ilustración 2-8 Representación gráfica de una relación tipo Asociación

- ***Agregación***: es una relación de asociación que sirve para indicar que la asociación es de composición, es decir, que un objeto está compuesto por otro. La diferencia de la asociación y la agregación es prácticamente conceptual y se hace evidente cuando el objeto componente no puede existir o funcionar completamente sin el objeto agregado.

  Por ejemplo, es posible que un objeto casa (de la clase Vivienda) esté compuesto por un objeto mueble (de la clase ComponentesDeLaVivienda) y por otro objeto

paredes (de la clase MuroEstructural). En el primer caso la relación sería de *asociación* debido a que una casa sigue siendo casa si no posee sillas o muebles; pero, la segunda relación sería de *agregación* dado que una casa sin paredes no tendría aspecto de casa sino de kiosco.

Ilustración 2-9Representación gráfica de una relación tipo Agregación

- ***Generalización***: es una relación de especialización en la cual los objetos del elemento generalizado pueden sustituir los objetos del elemento general (hijo - padre). De esta forma el hijo comparte la estructura y el comportamiento del padre.

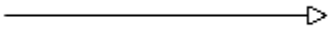


Ilustración 2-10 Representación gráfica de una relación tipo Generalización

- ***Realización***: es una relación semántica entre clasificadores, en donde un clasificador especifica un contrato que otro clasificador garantiza que cumplirá. Se pueden encontrar relaciones de realización entre interfaces y clases que las implementan, y entre casos de uso y colaboraciones que los realizan.

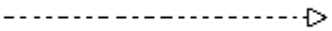


Ilustración 2-11 Representación gráfica de una relación tipo Realización

2.2.3 El modelo del proceso unificado (UP).

El Proceso Unificado (UP) se basa en la metodología Orientada a Objetos (O.O.) y utiliza, para representar los elementos del sistema, los procesos y los flujos de datos, los diagramas del UML.

Aunque UML es independiente del proceso de desarrollo, los creadores de UML propusieron su propia metodología de desarrollo denominada *Proceso Unificado de Desarrollo*.

Las características principales de UP son tres:

- **Dirigido por casos de uso:** Un caso de uso representa una pieza de funcionalidad en el sistema que le devuelve al usuario un resultado de valor. Los casos de uso sirven para capturar requerimientos funcionales.

- **Centrado en la arquitectura:** Los casos de uso son desarrollados en conjunto con la arquitectura del sistema y se desarrollan a medida que lo hace el ciclo de vida.

- **Iterativo e incremental:** La vida de un sistema se encuentra dividida en ciclos. Cada ciclo termina con un lanzamiento de diferentes modelos del producto y consiste de cuatro fases.

El UP se compone de 4 fases. Dichas Fases, son una especie de mini proyectos en las que se repiten, iteración tras iteración, los pasos normales de un ciclo de vida de desarrollo. Es decir, el Análisis, el Diseño, la Implementación y las Pruebas, entre otras. Se realizan en cada una de las iteraciones, una y otra vez hasta completar la fase actual y pasar a la siguiente. Las iteraciones repetitivas van proporcionando resultados inmediatos comparables al desarrollo incremental basado en prototipos.

Las fases sirven para alcanzar hitos o puntos de toma de decisiones en cuanto a continuar o no con el desarrollo del proyecto dependiendo de los resultados obtenidos. Las fases son:

- *Inicio*: Se realizará una adecuada delimitación del alcance del software. Se establecerán los temas, subtemas y ejercicios que abordan tanto el módulo de información teórica como el módulo de ejercicios prácticos. Igualmente se establecerá el alcance del módulo de ejercicios planteados por el usuario.

  En cuanto a los criterios de éxito, se determinará el resultado esperado al terminar cada una de las iteraciones.

- *Elaboración*: Se determinará el mayor número de requisitos específicos del aplicativo y se realizará un prototipo con los casos de uso más significativos.

  De esta manera se determinará de forma detallada el contenido de cada uno de los temas, subtemas y ejercicios, además de los respectivos requisitos del solucionador de ejercicios, entre ellos: la forma cómo el usuario se comunicará con

él, el debido procedimiento que se llevará a cabo tras cada iteración en el desarrollo de un ejercicio y la forma en que el sistema proporcionará los resultados obtenidos.

- *Construcción*: Se desarrollará cada uno de los módulos diseñados para el sistema. El módulo de información teórica deberá explicar cualquiera de los temas, el módulo de ejercicios prácticos debe estar en capacidad de explicar cualquiera de los ejercicios del sistema y el solucionador de ejercicios debe poder realizar sin ningún problema cualquier ejercicio que el usuario le plantee. De cada una de las operaciones se realizarán las pruebas respectivas.

- *Transición*: Se realizarán las pruebas finales del aplicativo y se procederá a la utilización del sistema en una prueba piloto. En ella se observará cómo el grupo de estudiantes responde al uso del aplicativo y se determinará si es necesario implementar un requisito adicional. Al final de esta fase se tendrá el producto terminado: SÍMPLEX MEDIA.

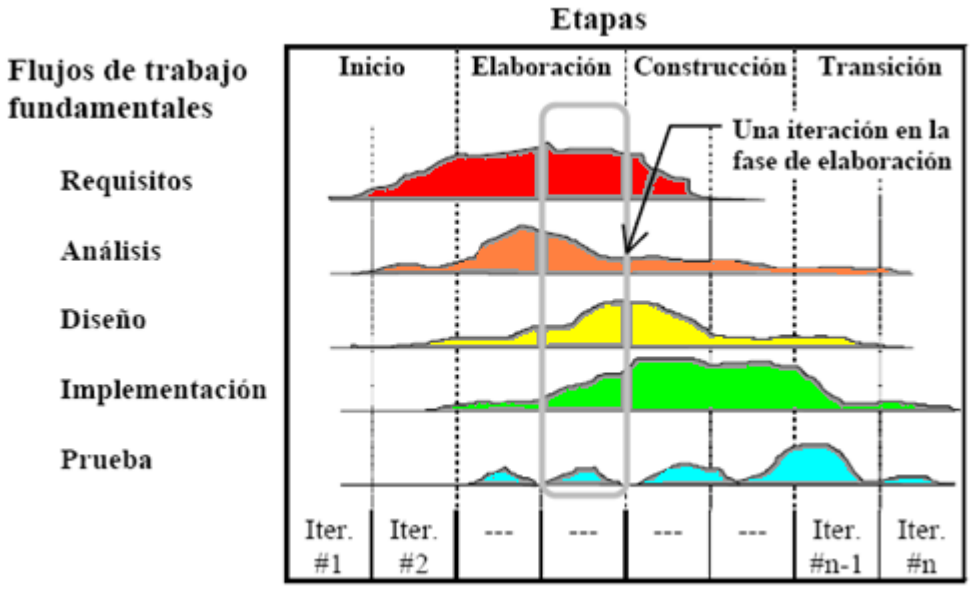


Ilustración 2-12 Representación de las Fases de UP

2.3 HERRAMIENTAS TECNOLÓGICAS PARA LA CONSTRUCCIÓN DE SÍMPLEX MEDIA

La construcción de SÍMPLEX MEDIA se hace posible gracias a que se cuenta con nuevas tecnologías para el desarrollo de software. Desde plataformas de trabajo robustas hasta entornos de desarrollo sencillos y fáciles de usar. La tecnología usada para el desarrollo del sistema es la de Microsoft .Net en su versión del Framework 3.5 que incorpora un nuevos aspectos de Windows Presentation Foundation.

A continuación se dará una descripción general acerca de estas tecnologías para el desarrollo de software altamente llamativos en aspectos visuales y poder de cálculo.

2.3.1 Microsoft .NET. Microsoft .NET es una plataforma sobre la cual se puede desarrollar y ejecutar aplicaciones. Pone además, una gran cantidad de herramientas y servicios que facilitan el desarrollo de aplicaciones, así como mecanismos robustos y seguros.

Entre los componentes de la plataforma .Net se encuentran:

- Un entorno de ejecución de aplicaciones (Runtime), el cual permite la ejecución de las aplicaciones. Es algo así como la máquina virtual de java (JVM).

- Un variado conjunto de bibliotecas de funcionalidades y controles reutilizables (.exe y .dll) que pueden ser utilizados por cualquier tipo de aplicaciones.

- Un conjunto de lenguajes de programación, que pueden interactuar entre sí. En este proyecto solo se usa C# .NET (y en unión con el WPF, el XAML).

- Una serie de utilidades y herramientas para simplificar las tareas que se derivan de la programación.

- Una completa documentación y guías de arquitectura.

2.3.2 Framework 3.5. El .NET Framework (Marco de Trabajo) es el componente principal de la plataforma .Net y es necesario para desarrollar y ejecutar aplicaciones en entornos de prueba o producción.

El .NET Framework incluye un entorno de ejecución (Runtime) y una biblioteca de funcionalidades (Class Library), y tiene tres variantes principales:

- .NET Framework Redistributable Package.
- .NET Framework SDK.
- .NET Compact Framework.

El .NET Framework puede ser instalado en cualquier sistema operativo de la familia Windows superior a Windows 98. Pero, puede encontrarse instalado en las versiones de Windows superiores a Windows 2003 Server y Windows XP SP2.

Microsoft .NET Framework, tiene como objetivo lograr una forma sencilla para el desarrollo de software reduciendo los puntos vulnerables y logrando así mayor seguridad en los programas.

Microsoft ha desarrollado varias versiones del .NET Framework. Este proyecto se ha desarrollado utilizando la versión 3.5 que incorpora nuevas características adecuadas para el desarrollo multimedial de SÍMPLEX MEDIA.

La versión 3.5 incorpora nuevas características y mejora muchas otras. Entre aquellas novedades figuran: ampliaciones en la compatibilidad con aplicaciones móviles distribuidas, mejoras en ASP .NET que permiten la implementación de páginas web con *Microsoft AJAX Library*, nuevas clases para el *Common Language Runtime (CLR)*, *Criptografía*, *Windows Communication Foundation (WCF)*, *Windows Presentation Foundation (WPF)*, *Windows Workflow Foundation (WF)*, compatibilidad con Windows Vista y WPF para los Formularios de *Windows Forms*, *Language-Integrated Query (LINQ)*, entre otras, y establece a XAML como un canal de comunicación entre todos los productos .NET.

2.3.3 Windows presentation foundation (WPF). Microsoft, define a Windows Presentation Foundation (WPF) como "modelo de programación unificado con el que generar experiencias de cliente inteligentes de Windows, en las que se incorpora la interfaz de usuario, multimedia y documentos.

La plataforma de desarrollo WPF se ha generado sobre un sistema de programación básico, que se extiende para admitir un amplio conjunto de características de desarrollo

de aplicaciones que incluyen el propio modelo de aplicación, recursos, controles, gráficos, diseño, enlace de datos, documentos y seguridad."[6]

"En .NET Framework 3.5, Windows Presentation Foundation contiene modificaciones y mejoras en numerosas áreas, entre las que se incluyen el control de versiones, el modelo de la aplicación, el enlace de datos, los controles, los documentos, las anotaciones y los elementos de la interfaz de usuario 3D"[7].

WPF es una de las novedosas tecnologías de Microsoft, la cual enriquece el desarrollo de interfaces ampliando las mejores características de aplicaciones Windows; también tiene en cuenta las aplicaciones web.

En cuanto a su plataforma de desarrollo ofrece una gran potencialidad, que permite el desarrollo de aplicaciones con apariencias llamativas y funcionales que proporcionan una mayor interacción con el usuario, ya que permite manipular fácilmente animación video, audio, documentos, gráficos 3D.

Dentro de la jerarquía de clases del WPF se han usado los siguientes espacios de nombres (namespace):

- `System.Windows`: Proporciona varias importantes clases de elementos base de Windows Presentation Foundation (WPF), varias clases que admiten el sistema de propiedades y la lógica de eventos de WPF.

- `System.Windows.Input`: Proporciona los tipos para la compatibilidad del sistema de entrada de WPF. Se incluyen las clases de abstracción de dispositivo del mouse, del teclado y de los dispositivos de lápiz.

- `System.Windows.Media`: Proporciona los tipos que habilitan la integración de elementos multimedia enriquecidos, incluidos los dibujos, el texto y el contenido de audio o vídeo en aplicaciones de WPF.

[6] Sitio web: http://msdn.microsoft.com/es-es/library/ms745659.aspx

[7] Documentación de Microsoft Visual Studio 2008. Manual del programador de .NET Framework Lo nuevo de .NET Framework versión 3.5.

- `System.Windows.Media.Animation`: Proporciona tipos que permiten la funcionalidad de animación de propiedades, también incluye escalas de tiempo, guiones gráficos y fotogramas clave.

- `System.Windows.Documents`: Contiene tipos que admiten la creación de *Documentos Dinámicos* (FlowDocument).

2.3.3.1 Documentos dinámicos (FlowDocumnet)

En lugar de establecerse en un diseño predefinido, este tipo de documentos ajusta y recoloca dinámicamente su contenido dependiendo del tamaño de la ventana, la resolución del dispositivo y las preferencias opcionales del usuario. Además, ofrecen características avanzadas de documentos, como la paginación y las columnas.

También, los documentos dinámicos tienen varias características integradas que incluyen la búsqueda, modos de presentación que optimizan la legibilidad y la capacidad de cambiar el tamaño y la apariencia de las fuentes. Este tipo de documentos son óptimos para su uso cuando la facilidad de lectura constituye el principal escenario de consumo del documento.

En SÍMPLEX MEDIA se usan estos documentos a la hora de presentar los temas de los módulos Información Teórica y Ejercicios Prácticos. Con esto se brinda al usuario una variada gama de posibilidades en el momento de examinar la información del sistema.

2.3.3.2 Líneas de tiempo (Storyboard)

WPF brinda un conjunto de características en cuanto a gráficos y diseño que permite crear interfaces de usuario y documentos muy atractivos. En este caso las animaciones hacen que una interfaz de usuario sea más vistosa y práctica.

Una animación o línea de tiempo es una ilusión que se crea mediante el cambio rápido entre una serie de imágenes ligeramente diferentes. El cerebro humano asimila este conjunto de imágenes como una sola escena cambiante.

WPF posee un sistema para el control de tiempo que se expone a través del código administrado y *Lenguaje de Marcado de Aplicaciones Extensible* (XAML), y que está completamente integrado en su marco de trabajo (Framework). Las animaciones de WPF permiten animar controles y otros objetos gráficos de forma sencilla.

Sin usar WPF, es necesario preocuparse por la forma en que se ejecutarán las animaciones, es decir, administrar los temporizadores y demás controladores necesarios.

Sin embargo, WPF controla todo el trabajo de administración del sistema de temporización y de actualización de la pantalla. Todo esto se produce en segundo plano.

Por lo tanto, la ventaja de usar WPF es que él proporciona clases de control de tiempo que permiten al programador concentrarse en los efectos que se desea observar y no en la manera de conseguirlos.

Por medio de estas poderosas herramientas se ha dado a SÍMPLEX MEDIA un amigable entorno que resulta atractivo, entretenido y fácil de usar.

También ha sido posible crear animaciones que guían al estudiante a través del estudio del Método Símplex haciendo que el trabajo del aprendizaje sea una grata experiencia.

2.3.4 Lenguaje de marcado de aplicaciones extensible (XAML). XAML es un Lenguaje de Marcado similar a HTML, pero es para Aplicaciones y es Extensible a código administrado, de ahí su nombre. El Lenguaje de marcado para la programación declarativa de aplicaciones (Extensible Application Markup Language) simplifica la creación de una interfaz de usuario cuando se hace uso del modelo de programación de WPF.

El propio XAML es un concepto de lenguaje más amplio que WPF y abarca distintas funcionalidades.

Es posible crear elementos visibles de la interfaz de usuario en el marcado declarativo XAML y, a continuación, separar la definición de la interfaz de usuario de la lógica en tiempo de ejecución utilizando archivos de código subyacente, que se unen al marcado mediante definiciones de clases parciales.

La utilización de código administrado (C#) y código declarativo (XAML) de forma combinada es muy útil, dado que el código declarativo no permite el control de flujo, esto es, la ejecución secuencial a la que está acostumbrado todo programador. Pero con dicho código se simplifica de manera increíble la construcción de la interfaz de usuario. Con esto se habilita el código simplificado y el acceso a la depuración para los objetos que se crean en XAML.

3 MODELO DE DOMINIO

El modelo de dominio permitirá al lector introducirse en lo que se pretende conseguir con el desarrollo de SÍMPLEX MEDIA.

Se proporcionará un listado de las características más sobresalientes que se consideraron al inicio del desarrollo.

Además, se contará con una vista de las clases del dominio, es decir, los conceptos más importantes que se pueden extraer del listado de características y, que posteriormente, servirán de base para las clases definitivas del sistema.

3.1 REQUERIMIENTOS DEL SISTEMA

Para conocer los requerimientos en cuanto a funciones y operaciones de SÍMPLEX MEDIA se ha creado un listado de características y un listado de reglas. Cada característica (o conjunto de características) servirá para crear los casos de uso que son la base del análisis O.O.

3.1.1 Listado de características. Los requisitos son capacidades o condiciones que deben conformar el sistema que se desarrolla.

Para registrar los requisitos de este proyecto, se utilizo un listado de características de alto nivel, esto significa que los requisitos están considerados dentro de un contexto amplio de uso del sistema, orientados a obtener un resultado de valor o cumplir con un objetivo del usuario.

El siguiente es el listado de características que se tuvieron en cuenta para este proyecto, agrupadas según el módulo en el que operan.

3.1.1.1 Características generales del sistema y de los tres módulos

Tabla 3.1-1 Característica 1 de Símplex Media

ID	CARACTERÍSTICA 1
Descripción	El sistema presentará una página de bienvenida donde se visualizan algunas opciones de acceso rápido: vínculos de los archivos de ejercicios revisados recientemente, vínculos a los materiales estudiados recientemente y vínculos a diferentes secciones de la ayuda del sistema. Además, tres opciones que lo conduzcan de forma directa a cada uno de los módulos del sistema.
Estado	Incorporada

Tabla 3.1-2 Característica 2 de Símplex Media

ID	CARACTERÍSTICA 2
Descripción	Se permitirá al usuario realizar búsquedas de términos, temas y ejercicios. Los resultados aparecerán en un listado en forma de vínculos.
Estado	Incorporada

Tabla 3.1-3 Característica 3 de Símplex Media

ID	CARACTERÍSTICA 3
Descripción	Para mayor comodidad visual, se permitirá al usuario modificar el tamaño de la interfaz a Modo Pantalla Completa.
Estado	Incorporada

Tabla 3.1-4 Característica 4 de Símplex Media

ID	CARACTERÍSTICA 4
Descripción	Se podrá acceder a todas las funcionalidades del sistema a través de menús y se podrán configurar algunas opciones del sistema.
Estado	Incorporada

Tabla 3.1-5 Característica 5 de Símplex Media

ID	CARACTERÍSTICA 5
Descripción	La teoría de cada uno de los temas y subtemas, así como la explicación de términos o conceptos, los ejercicios del sistema y el desarrollo paso a paso de ejercicios planteados por el usuario, estarán soportadas en páginas XAML.
Estado	Incorporada

Tabla 3.1-6 Característica 6 de Símplex Media

ID	CARACTERÍSTICA 6
Descripción	La página de la explicación que se esté visualizando proporciona vínculos que conduzcan a otros materiales relacionados o que sirven para dar una definición ha determinado término. Estos vínculos adicionales aparecerán en un listado por fuera del contenido del material.
Estado	Incorporada

Tabla 3.1-7 Característica 7 de Símplex Media

ID	CARACTERÍSTICA 7
Descripción	Cada uno de los materiales contará con vínculos a otros materiales que expliquen conceptos que suelen ser desconocidos. Estos vínculos aparecerán en el contenido mismo del material.
Estado	Incorporada

Tabla 3.1-8 Característica 8 de Símplex Media

ID	CARACTERÍSTICA 8
Descripción	El sistema proporciona la opción de reproducir un archivo de sonido que contenga determinada explicación en caso que el material que se esté visualizando tenga habilitada esta funcionalidad.
Estado	Incorporada

Tabla 3.1-9 Característica 9 de Símplex Media

ID	CARACTERÍSTICA 9
Descripción	En caso de que el material que se esté mostrando tenga asociada una animación con la explicación de su contenido, el sistema proporcionará una opción que permita reproducir dicha animación.
Estado	Incorporada

Tabla 3.1-10 Característica 10 de Símplex Media

ID	CARACTERÍSTICA 10
Descripción	El sistema da la opción de reproducir, pausar, adelantar, retroceder y detener la explicación que se esté proporcionando.
Estado	Incorporada

Tabla 3.1-11 Característica 11 de Símplex Media

ID	CARACTERÍSTICA 11
Descripción	El sistema permite al usuario realizar impresiones, tanto de los materiales que se estén visualizando, como de los desarrollos de ejercicios planteados por el usuario.
Estado	Incorporada

Tabla 3.1-12 Característica 12 de Símplex Media

ID	CARACTERÍSTICA 12
Descripción	Los tres módulos se comunicarán entre sí. Los temas y subtemas del módulo de Información Teórica proporcionará vínculos a ejercicios explicados del módulo de Ejercicios Prácticos. Éste, a su vez, tendrá vínculos de ejercicios sin solucionar que pueden ser desarrollados en el módulo de Solución de Ejercicios, quien, en la explicación de los pasos de un problema, proporcionará vínculos a materiales de Información Teórica.
Estado	Incorporada

Tabla 3.1-13 Característica 13 de Símplex Media

ID	CARACTERÍSTICA 13
Descripción	Cada vez que se abra un material en el visualizador, el sistema, guardará, en un registro de entradas, la dirección de este.
Estado	Incorporada

Tabla 3.1-14 Característica 14 de Símplex Media

ID	CARACTERÍSTICA 14
Descripción	Cada vez que el usuario cree o abra un ejercicio, el sistema actualizará el registro de los ejercicios consultados recientemente.
Estado	Incorporada

Tabla 3.1-15 Característica 15 de Símplex Media

ID	CARACTERÍSTICA 15
Descripción	Al iniciar el uso de cada módulo se presentará un listado con los materiales o ejercicios recientes alusivos a dicho módulo.
Estado	Incorporada

3.1.1.2 Características relacionadas con el módulo información teórica

Tabla 3.1-16 Característica 16 de Símplex Media

ID	CARACTERÍSTICA 16
Descripción	El sistema proporciona al usuario temas preliminares como: conceptos de los diferentes tipos de matrices, manejo y operaciones con matrices, ecuaciones lineales y eliminaciones con Gauss y Gauss Jordán.
Estado	Incorporada

Tabla 3.1-17 Característica 17 de Símplex Media

ID	CARACTERÍSTICA 17
Descripción	Los temas principales serán los relacionados con el Método Símplex: modelo de dos variables, solución gráfica, forma estándar de

	programación lineal, Métodos Símplex Primal y Símplex Dual.
Estado	Incorporada

Tabla 3.1-18 Característica 18 de Símplex Media

ID	CARACTERÍSTICA 18
Descripción	El sistema permite que el usuario seleccione subtemas de cada uno de los temas principales.
Estado	Incorporada

Tabla 3.1-19 Característica 19 de Símplex Media

ID	CARACTERÍSTICA 19
Descripción	El sistema proporciona la opción de revisar ejemplos adicionales durante la explicación de determinado tema.
Estado	Incorporada

3.1.1.3 Características relacionadas con el módulo ejercicios prácticos

Tabla 3.1-20 Característica 20 de Símplex Media

ID	CARACTERÍSTICA 20
Descripción	El sistema presenta un listado de problemas agrupados de acuerdo al tema, para que el usuario seleccione uno de ellos.
Estado	Incorporada

Tabla 3.1-21 Característica 21 de Símplex Media

ID	CARACTERÍSTICA 21
Descripción	Los ejercicios del sistema, ya desarrollados, tratarán de temas relacionados con: el algoritmo Símplex Primal, la Técnica M, el algoritmo de las Dos Fases y el Método Símplex Dual.
Estado	Incorporada

Tabla 3.1-22 Característica 22 de Símplex Media

ID	CARACTERÍSTICA 22
Descripción	El sistema explica paso a paso cada iteración del desarrollo del ejercicio seleccionado por el usuario, sustentando de forma adecuada cada procedimiento.
Estado	Incorporada

Tabla 3.1-23 Característica 23 de Símplex Media

ID	CARACTERÍSTICA 23
Descripción	El sistema explica los datos obtenidos del desarrollo de un ejercicio.
Estado	Incorporada

Tabla 3.1-24 Característica 24 de Símplex Media

ID	CARACTERÍSTICA 24
Descripción	El sistema presentará vínculos para resolver ejercicios con el módulo Solución de Ejercicios.
Estado	Incorporada

3.1.1.4 Características relacionadas con el modulo solución de ejercicios

Tabla 3.1-25 Característica 25 de Símplex Media

ID	CARACTERÍSTICA 25
Descripción	El sistema presentará una descripción de la forma de plantear, desarrollar e interpretar los resultados de los diferentes tipos de ejercicios.
Estado	Incorporada

Tabla 3.1-26 Característica 26 de Símplex Media

ID	CARACTERÍSTICA 26
Descripción	El sistema resolverá diferentes tipos de ejercicios: operaciones con matrices, solución a sistemas de ecuaciones lineales y problemas de programación lineal.
Estado	Incorporada

Tabla 3.1-27 Característica 27 de Símplex Media

ID	CARACTERÍSTICA 27
Descripción	El sistema proporcionará al usuario una manera de seleccionar el tipo de ejercicio que desea desarrollar.
Estado	Incorporada

Tabla 3.1-28 Característica 28 de Símplex Media

ID	CARACTERÍSTICA 28
Descripción	El sistema da la opción de resolver el ejercicio dando una respuesta de forma directa o proporcionando el desarrollo paso a paso.
Estado	Incorporada

Tabla 3.1-29 Característica 29 de Símplex Media

ID	CARACTERÍSTICA 29
Descripción	El sistema da la opción de resolver el ejercicio con o sin la explicación del procedimiento que se lleva a cabo tras cada paso y tras cada iteración.
Estado	Incorporada

Tabla 3.1-30 Característica 30 de Símplex Media

ID	CARACTERÍSTICA 30
Descripción	El sistema permite al usuario plantear un ejercicio (ingresar datos) y desarrollarlo de forma directa.
Estado	Incorporada

Tabla 3.1-31 Característica 31 de Símplex Media

ID	CARACTERÍSTICA 31
Descripción	El sistema permite al usuario plantear un ejercicio y desarrollarlo dando una explicación paso a paso del procedimiento que se lleva a cabo tras cada iteración.
Estado	Incorporada

Tabla 3.1-32 Característica 32 de Símplex Media

ID	CARACTERÍSTICA 32
Descripción	El sistema permite al usuario guardar el ejercicio en cualquier momento del desarrollo, en formato digital.
Estado	Incorporada

Tabla 3.1-33 Característica 33 de Símplex Media

ID	CARACTERÍSTICA 33
Descripción	El sistema permite la apertura de ejercicios guardados por el usuario.
Estado	Incorporada

Tabla 3.1-34 Característica 34 de Símplex Media

ID	CARACTERÍSTICA 34
Descripción	El sistema permite la modificación del planteamiento del ejercicio que el usuario haya abierto.
Estado	Incorporada

Tabla 3.1-35 Característica 35 de Símplex Media

ID	CARACTERÍSTICA 35
Descripción	El sistema permite la impresión del desarrollo de un ejercicio.
Estado	Incorporada

Tabla 3.1-36 Característica 36 de Símplex Media

ID	CARACTERÍSTICA 36
Descripción	El sistema permitirá seleccionar un planteamiento de un ejercicio almacenado en el sistema a fin de resolverlo utilizando las características anteriores.
Estado	Incorporada

Tabla 3.1-37 Característica 37 de Símplex Media

ID	CARACTERÍSTICA 37
Descripción	El sistema permitirá generar un planteamiento de forma aleatoria de problemas de programación lineal y de ecuaciones lineales
Estado	Incorporada

Tabla 3.1-38 Característica 38 de Símplex Media

ID	CARACTERÍSTICA 38
Descripción	La explicación paso a paso del desarrollo de los ejercicios estará habilitada para programación lineal y ecuaciones lineales, y será un documento en una página XAML.
Estado	Incorporada

3.1.2 Listado de reglas

Las reglas del dominio especifican las medidas o políticas, de cómo se van a desarrollar las características del sistema o que se tendrá presente como base para su desarrollo. Son políticas, leyes, entre otras que influyen sobre los requisitos de la aplicación.

Estas son las reglas que se consideraron para el desarrollo del proyecto.

Tabla 3.1-39 Regla 1 de Símplex Media

ID	REGLA 1
Descripción	Los usuarios que accedan al material del aplicativo han de tener conocimientos básicos en manejo de matrices y solución de sistemas de ecuaciones lineales. Por lo tanto, las explicaciones de dichas áreas no serán tan detalladas.

Tabla 3.1-40 Regla 2 de Símplex Media

ID	REGLA 2
Descripción	Dado que el sistema dará explicaciones mediante ayuda de multimedia es necesario que la computadora donde se ejecute el aplicativo posea los requerimientos mínimos en cuanto a hardware.

Tabla 3.1-41 Regla 3 de Símplex Media

ID	REGLA 3
Descripción	Para la explicación audible de los temas que la posean, el equipo deberá contar con un adecuado sistema de sonido, y para las explicaciones visuales es necesario que cuente con un monitor a color de resolución mínima de 1024 por 768 pixeles.

Tabla 3.1-42 Regla 4 de Símplex Media

ID	REGLA 4
Descripción	Para que el sistema pueda gestionar sus configuraciones y los ejercicios y materiales recientes del usuario, creará archivos planos y los guardará en disco. Por ello el usuario deberá garantizar al aplicativo adecuados permisos de lectura y escritura de archivos en el disco en el que esté instalado.

Tabla 3.1-43 Regla 5 de Símplex Media

ID	REGLA 5
Descripción	Al analizar la solución de ejercicios, explicados paso a paso, el usuario, debe tener a mano herramientas educativas (como calculadora, cuadernos, lápices, etc.) a fin de verificar o comprobar la información que el sistema brinda.

Tabla 3.1-44 Regla 6 de Símplex Media

ID	REGLA 6
Descripción	Durante el uso del aplicativo el usuario debe estar apoyado con material bibliográfico, esto le ayudará a asimilar de forma más adecuada la información proporcionada por el sistema.

Tabla 3.1-45 Regla 7 de Símplex Media

ID	REGLA 7
Descripción	Se agruparan los ejercicios y explicaciones de acuerdo a los temas y sub temas.

Tabla 3.1-46 Regla 8 de Símplex Media

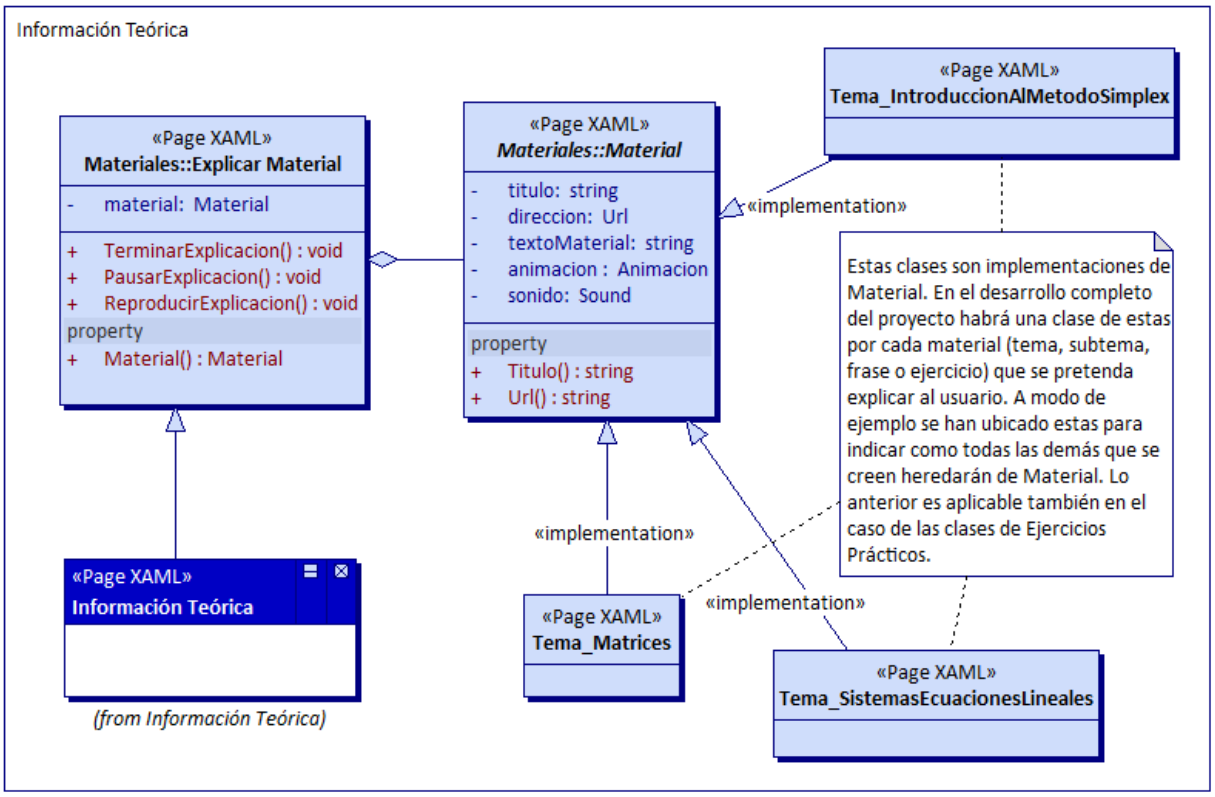
ID	REGLA 8
Descripción	En vista del carácter didáctico de este software, se limitará el ingreso de grandes cantidades de datos en el momento de plantear ejercicios para que el sistema los resuelva.

3.2 CLASES DEL DOMINIO

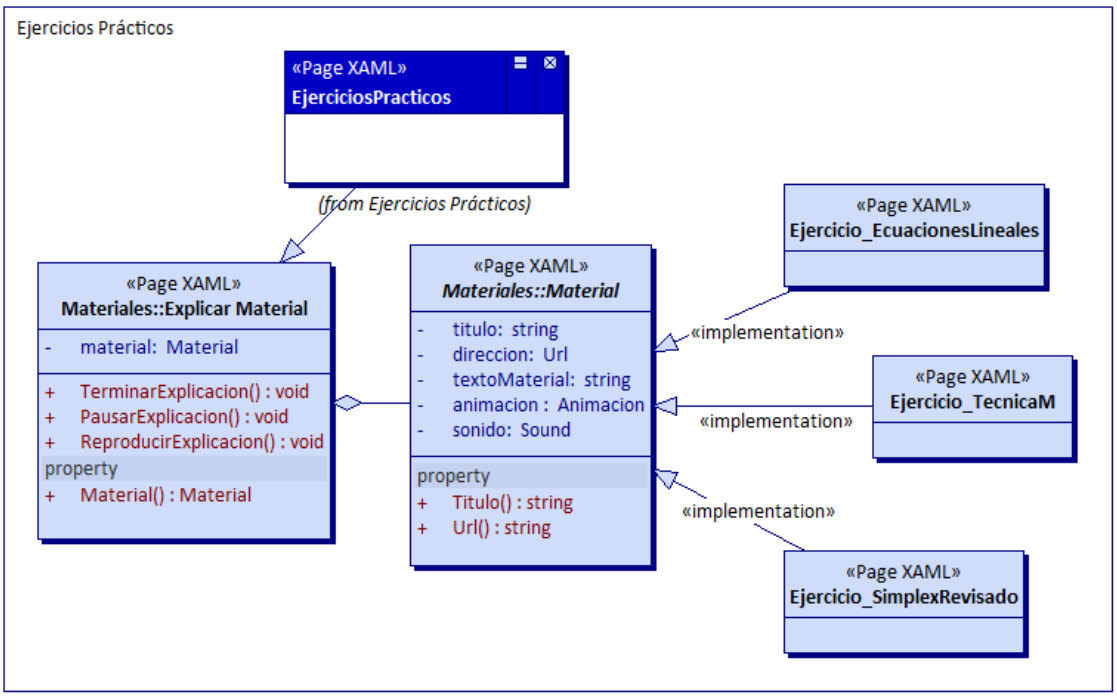
Las clases del dominio no son clases del código fuente del aplicativo, son conceptos clave que servirán como fuente de partida para analizar y diseñar las clases codificables definitivas. Se han organizado en paquetes (o contenedores que agrupan conceptos que se relacionan fuertemente) y se han realizado diagramas que muestran las relaciones entre ellos.

3.2.1 Diagramas de clases

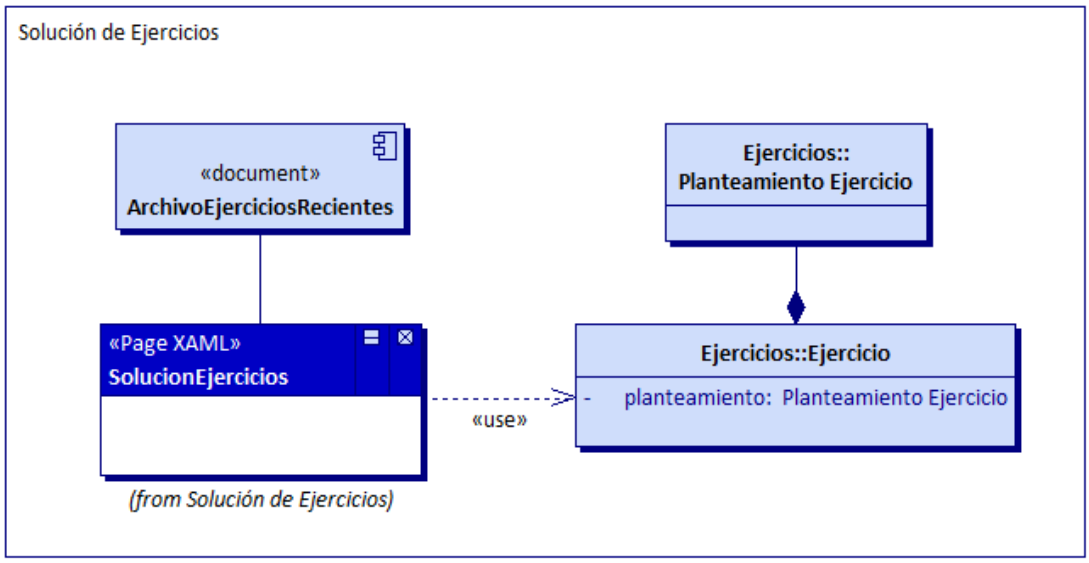
3.2.1.1 Diagrama: Módulo información teórica



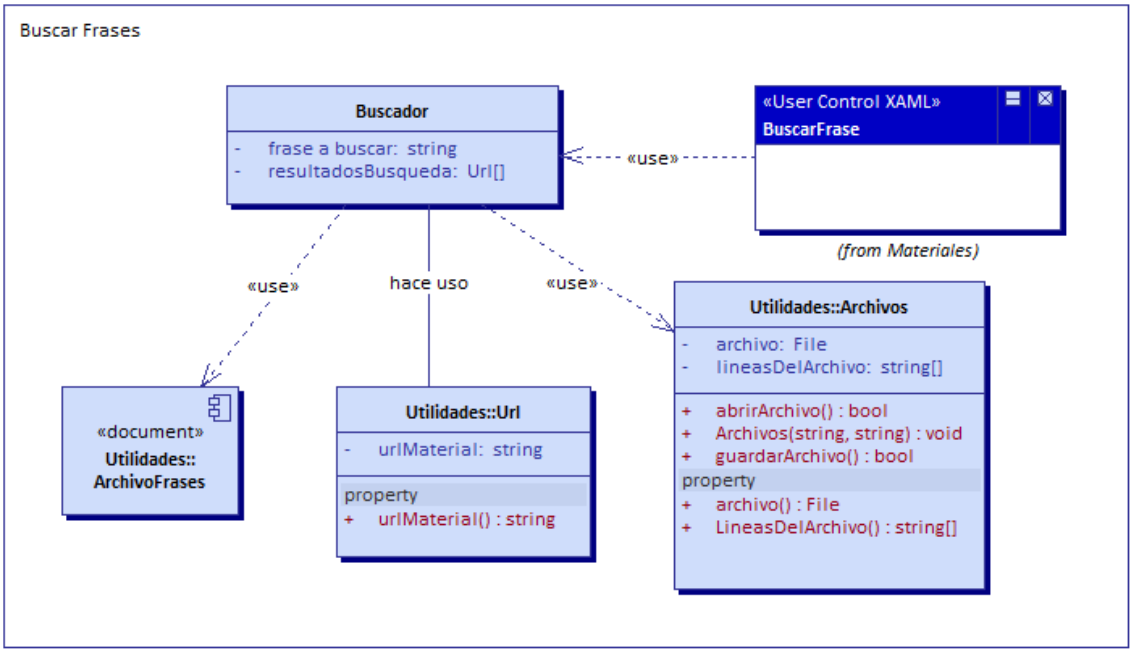
3.2.1.2 Diagrama: Módulo ejercicios prácticos



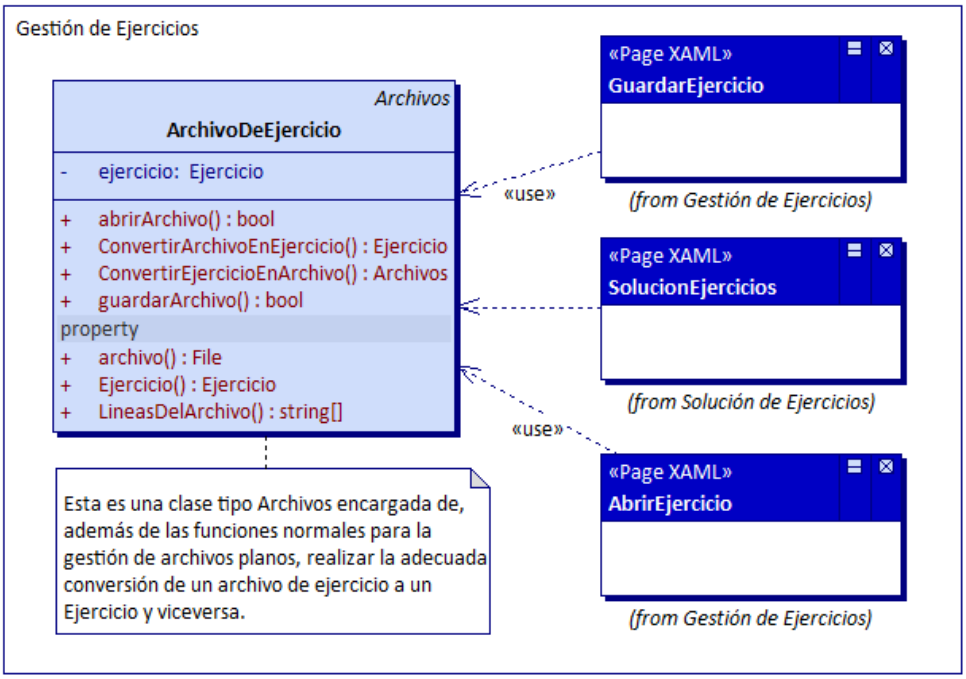
3.2.1.3 Diagrama: Módulo solución de ejercicios



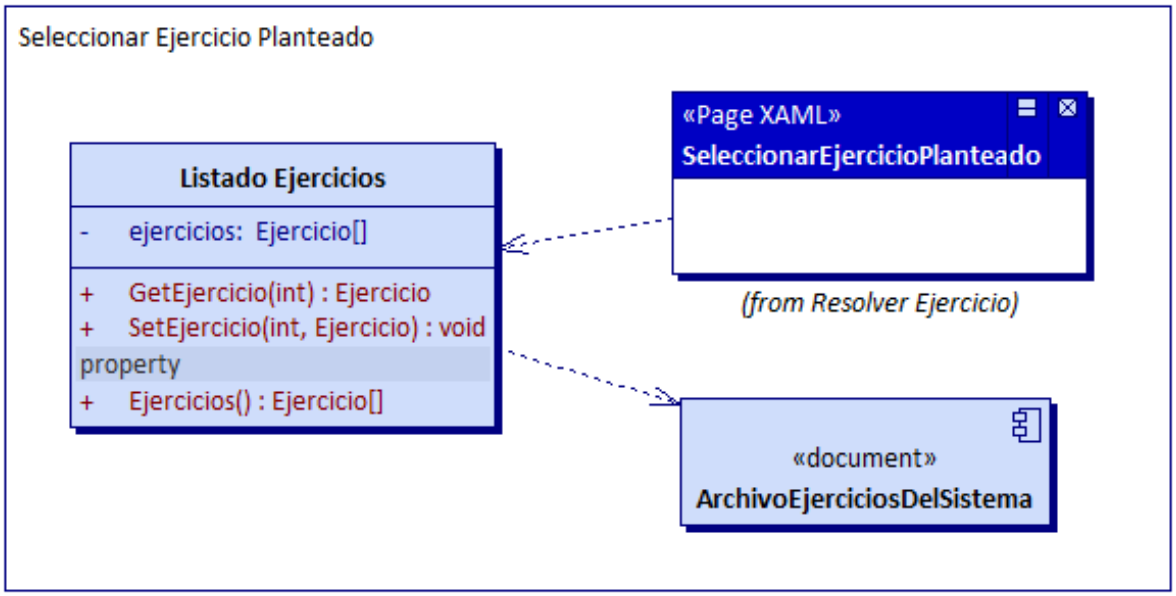
3.2.1.4 Diagrama: buscar frases



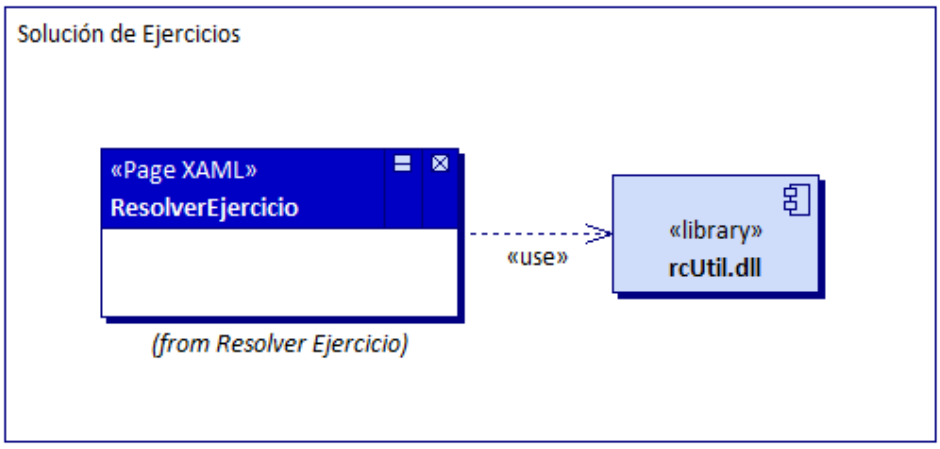
3.2.1.5 Diagrama: Gestión de ejercicios



3.2.1.6 Diagrama: Seleccionar ejercicios del sistema

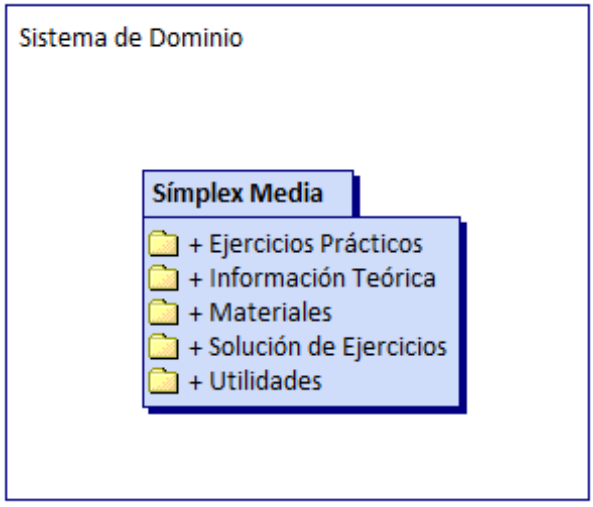


3.2.1.7 Diagrama: Resolver ejercicio

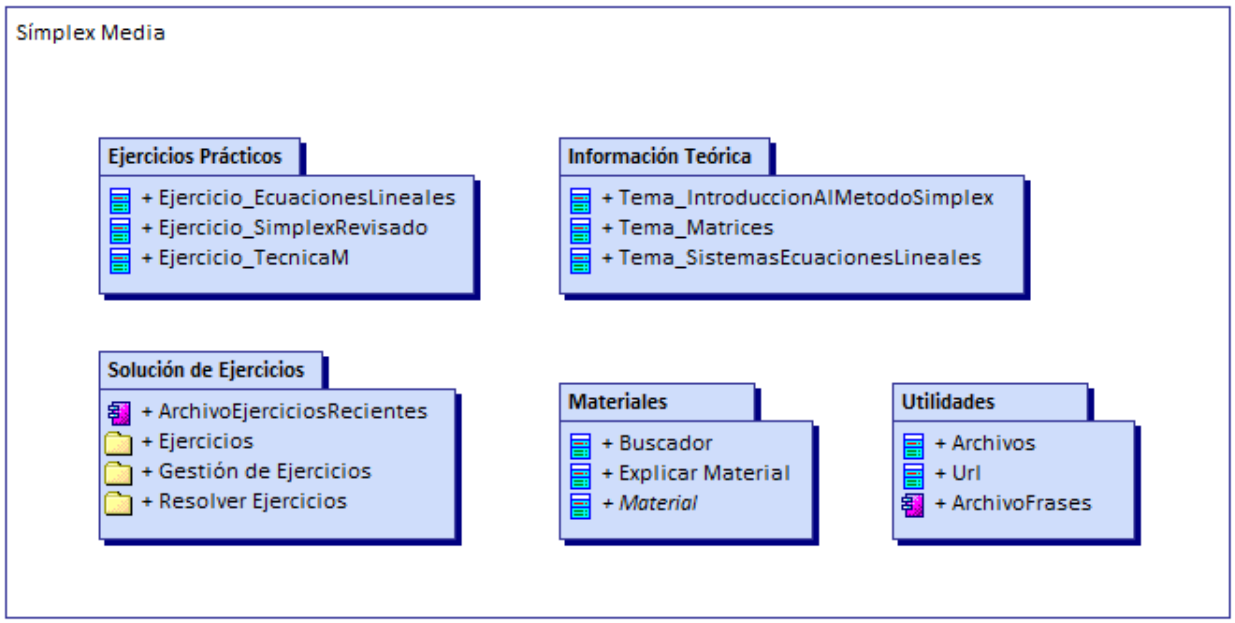


3.2.2 Diagrama de paquetes

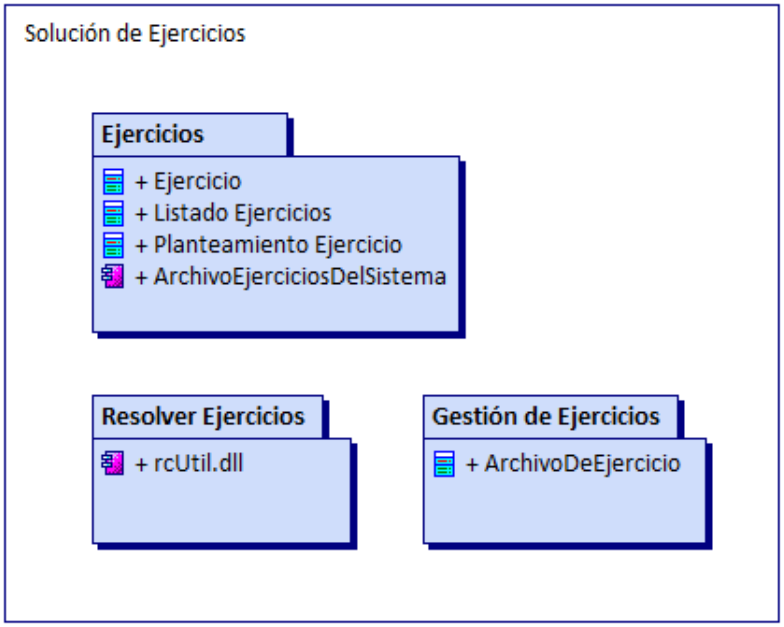
3.2.2.1 Paquete general: Sistema de dominio



3.2.2.2 Paquete: Símplex Media



3.2.2.3 Paquete: Solución de ejercicios



4 MODELO DE CASOS DE USO

Mediante este modelo se presentan los diferentes escenarios (o caminos de uso) del sistema. Partiendo de ellos, el analista de sistemas puede identificar de forma más clara, la mayoría de los objetos involucrados en realizar determinada tarea en comunicación con los diferentes roles que adquieren los usuarios del sistema.

4.1 ACTORES DE SÍMPLEX MEDIA

En UML un Caso de Uso es una descripción de un conjunto de acciones secuenciales que se llevan a cabo durante la comunicación de un Actor con el Sistema.

Un Actor es un rol (comportamiento específico) que adquiere el usuario del sistema en determinado momento. SÍMPLEX MEDIA posee solo un actor. Este actor será conocido como Usuario, y será el encargado de comunicarse con el sistema para realizar las todas las actividades posibles.

Sin embargo, en el modelamiento del sistema se identificó que existen algunas operaciones internas (que necesitaban ser descritas) que no las inicia directamente el Usuario, sino el mismo sistema. Por ello se ha llamado como segundo actor al Sistema.

A continuación se halla una descripción de estos actores:

Nombre:	Usuario
Descripción	
Es el actor principal del sistema, a quien se le proporciona toda la información y quien brinda todos los datos necesarios para la ejecución del aplicativo. SÍMPLEX MEDIA puede ser utilizado por docentes y estudiantes cuyos estudios estén relacionados con el contenido teórico analizado en el software. Sin embargo, en el momento de adquirir información del sistema, cualquiera de los anteriores se convierte en un *Usuario*, pues el sistema no clasifica la información dependiendo del perfil o nivel educativo que posea quien lo utilice.	
Objetivos	
Inicia sesión, realiza solicitudes de información, inicia la reproducción auditiva o visual de la explicación de un material, plantea problemas, gestiona archivos de ejercicios.	

Nombre:	Sistema
Descripción	
El aplicativo mismo se encarga de iniciar una serie de casos de uso, por ello es necesario representarlo como un actor independiente.	
Objetivos	
Inicia el trámite de algunos ejercicios, gestiona la consulta o profundización de algunos términos y/o temas, maneja eventos multimediales.	

Estos actores se encargarán de iniciar y operar los diferentes Casos de Uso del sistema.

4.2 CASOS DE USO EXPANDIDOS

4.2.1 Iniciar uso de SÍMPLEX MEDIA

Caso de uso	Iniciar Uso de SÍMPLEX MEDIA
Actor principal	**Usuario**
Precondiciones	✍ **Se debe haber ejecutado el aplicativo.**
Poscondiciones	✍ **Se ejecuta la opción que el Usuario haya seleccionado.**
Flujo básico	
1. El sistema muestra las opciones que conducirán al usuario a cada uno de los tres módulos. 2. El sistema muestra una pantalla de bienvenida o página de inicio con vínculos a diferentes secciones de la ayuda del sistema, con los archivos de ejercicios revisados recientemente y con un listado de los materiales estudiados recientemente. 3. El sistema muestra una caja de texto que permite ingresar una frase a buscar. 4. El sistema muestra un menú con diferentes opciones. 5. El sistema muestra una opción para cambiar a Modo Pantalla Completa, o en caso de encontrarse en ese modo, muestra una opción para cambiar a Modo Pantalla Normal. 6. El sistema muestra una opción que permita mostrar nuevamente la página de bienvenida en caso que esta haya sido ocultada para mostrar otro tipo de información. 7. El sistema muestra los listados de todos los materiales de los módulos. 8. Se espera que el usuario seleccione una de las opciones o que el usuario cierre el aplicativo.	
Flujos alternativos	
8a. En caso que el Usuario seleccione la opción del módulo de información teórica. 1. Inicia el caso de uso Iniciar Módulo Información Teórica. 2. Reinicia el caso de uso. 8b. En caso que el Usuario seleccione la opción del módulo de ejercicios prácticos. 1. Inicia el caso de uso Iniciar Módulo Ejercicios Prácticos. 2. Reinicia el caso de uso. 8c. En caso que el Usuario seleccione la opción del módulo de solución de ejercicios. 1. Inicia el caso de uso Iniciar Módulo Solución de Ejercicios. 2. Reinicia el caso de uso.	

8d. En caso que el Usuario seleccione un material estudiado recientemente.

1. Inicia el caso de uso Explicar Material.
3. Reinicia el caso de uso.

8e. En caso que el Usuario seleccione un material de uno de los listados de los módulos.

2. Inicia el caso de uso Explicar Material.
3. Reinicia el caso de uso.

8f. En caso que el Usuario seleccione un archivo de ejercicio revisado recientemente.

1. Inicia el caso de uso Recuperar Ejercicio.
2. Reinicia el caso de uso.

8g. En caso que el Usuario ingrese una frase a buscar.

1. Inicia el caso de uso Buscar Frase.
3. Reinicia el caso de uso.

8h. En caso que el Usuario seleccione una de las opciones del menú.

1. Inicia el caso de uso Ejecutar Opción del Menú.
4. Reinicia el caso de uso.

8i. En caso que el Usuario seleccione la opción de cambiar a (o de) *pantalla completa*.

1. Se realiza el cambio a Pantalla Completa o, si se encuentra en ese modo, se cambia a Pantalla Normal.
5. Reinicia el caso de uso.

8j. En caso que el Usuario seleccione la opción de mostrar la *página de inicio*.

1. Se visualiza la página de inicio y se cargan en ella los ejercicios y materiales recientes.
2. Reinicia el caso de uso.

8k. En caso que el Usuario seleccione la opción para Salir.

1. Termina la ejecución del aplicativo.
2. Termina el caso de uso.

4.2.2 Iniciar módulo información teórica

Caso de uso	Iniciar Módulo Información Teórica
Actor principal	Usuario
Precondiciones	✍ Se debe haber seleccionado el Módulo Información Teórica.
Poscondiciones	✍ Se inicia la explicación del tema que el Usuario ha seleccionado del listado.
Flujo básico	
1. El sistema carga el listado de temas generales y sus respectivos subtemas. 2. El sistema presenta un listado de los temas generales y sus respectivos subtemas. 3. El sistema espera que el usuario seleccione un tema para mostrar su explicación.	
Flujos alternativos	
3a. En caso que el Usuario seleccione uno de los temas de la lista. 1. Inicia el caso de uso Explicar Material. 2. Reinicia el caso de uso.	

4.2.3 Iniciar módulo ejercicios prácticos

Caso de uso	Iniciar Módulo Ejercicios Prácticos
Actor principal	Usuario
Precondiciones	✍ Se debe haber seleccionado el Módulo Ejercicios Prácticos.
Poscondiciones	✍ Se inicia la explicación del ejercicio que el Usuario ha seleccionado del listado.
Flujo básico	
1. El sistema carga el listado de los ejercicios explicados por el sistema. 2. El sistema presenta un listado de los ejercicios asociados a los temas generales y sus respectivos subtemas. 3. El sistema espera que el usuario seleccione un ejercicio para mostrar su explicación.	
Flujos alternativos	
3a. En caso que el Usuario seleccione uno de los ejercicios de la lista. 1. Inicia el caso de uso **Explicar Material.** 2. Reinicia el caso de uso.	

4.2.4 Iniciar módulo solución de ejercicios

Caso de uso	Iniciar Módulo Solución de Ejercicios
Actores	Usuario
Precondiciones	✍ Se debe haber seleccionado el Módulo Solución de Ejercicios.
Poscondiciones	✍ Se ejecuta la opción que el Usuario haya seleccionado.
Flujo básico	
1. El sistema presenta al usuario una página de bienvenida con una explicación de lo que se puede hacer en el módulo y las opciones para crear o abrir un ejercicio. 2. El sistema presenta un listado de los ejercicios abiertos recientemente. 3. El sistema espera que el usuario seleccione una opción: crear un ejercicio, abrir uno desde una ubicación específica, abrir uno del listado de ejercicios recientes o examinar uno de los temas con explicación de los tipos de ejercicios que se pueden desarrollar.	
Flujos alternativos	
3a. En caso que el usuario seleccione la opción de crear un *Nuevo Ejercicio*. 1. Inicia el caso de uso **Nuevo Ejercicio.** 2. Si el usuario ha planteado un ejercicio, termina este caso de uso, en caso contrario, se reinicia. 3b. En caso que el Usuario seleccione la opción *Abrir Ejercicio*. 1. Inicia el caso de uso **Abrir Ejercicio.** 2. Si el usuario ha abierto un ejercicio, termina este caso de uso, en caso contrario, se reinicia. 3c. En caso que el Usuario seleccione uno de los nombres de los *Ejercicios Recientes*. 1. Inicia el caso de uso **Recuperar Ejercicio.** 2. Termina el caso de uso. 3d. En caso que el Usuario seleccione uno de los temas de la lista. 1. Inicia el caso de uso **Explicar Material.** 2. Reinicia el caso de uso.	

4.2.5 Ejecutar opción del menú

Caso de uso	Ejecutar Opción del Menú
Actor principal	Sistema
Precondiciones	✍ Se debe haber seleccionado una opción del menú.
Poscondiciones	✍ Se ejecuta la opción seleccionada.
Flujo básico	
1. El sistema ejecuta la opción seleccionada.	
Flujos alternativos	
1a. En caso que la opción seleccionada sea *Abrir Ejercicio*. 1. Inicia el caso de uso **Abrir Ejercicio.** 2. Termina el caso de uso. 1b. En caso que la opción seleccionada sea *Nuevo Ejercicio*. 1. Inicia el caso de uso **Nuevo Ejercicio.** 2. Termina el caso de uso. 1c. En caso que la opción seleccionada sea *Guardar Ejercicio*. 1. Inicia el caso de uso **Guardar Ejercicio.** 2. Termina el caso de uso. …	
Observaciones	
Aunque, en los flujos alternativos, solo se mencionen tres opciones del menú, en realidad existen muchas. Por lo tanto, lo que se quiere especificar es que según sea la opción del menú que se seleccione, el sistema iniciará el caso de uso correspondiente.	

4.2.6 Explicar material

Caso de uso	Explicar Material
Actores	Sistema, Usuario
Precondiciones	✍ Se debe haber escogido previamente cual es el material (tema, ejercicio o términos) sobre el que se desee mayor explicación.
Poscondiciones	✍ El sistema presenta la explicación de dicho material o en caso que no exista el archivo, un mensaje de error.
Flujo básico	
1. Realizar una búsqueda, en el sistema de archivos, de la dirección del archivo que	

contiene el material.

2. Se verifica la existencia del archivo.
3. El sistema muestra la información del material.
4. El sistema analiza si el material tiene asociados sonido y animación, y en caso afirmativo muestra opciones que permitan al usuario acceder a esas funcionalidades.
5. Si el material tiene sonido y/o animación el sistema permanece alerta a la espera de reproducir alguno de estos por orden del usuario.

Flujos alternativos

2a. En caso que el archivo no esté en la dirección especificada por el sistema

1. El sistema lanza un *Mensaje de Error* informando que el archivo ha sido borrado o no existe.
2. Termina el caso de uso.

5a. Si el material tiene asociado un sonido y el usuario desea escucharlo:

1. Inicia el caso de uso **Reproducir Sonido.**
2. Reinicia el caso de uso desde el paso 5.

5b. Si el material tiene asociado una animación y el usuario desea observarla:

1. Inicia el caso de uso **Reproducir Animación.**
2. Reinicia el caso de uso desde el paso 5.

Observaciones

Un material hace referencia a cualquier cosa que el sistema utilice como fuente de información para realizar una explicación. Bien sea un tema, un término o un ejercicio. Los materiales están compuestos por una serie de vínculos a otros materiales y texto que servirán para explicar de forma didáctica la información deseada. En algunos casos poseen sonido y/o animación.

Cuando se habla de “sonido”, se refiere a un archivo de audio con una lectura parcial del contenido del material, y cuando se habla de “animación”, se trata de una especie de video acompañada de audio que se reproducirá en una ventana independiente.

4.2.7 Buscar frase

Caso de uso	Buscar Frase
Actor principal	Sistema
Precondiciones	✍ Se debe haber indicado previamente una frase sobre el que se desee mayor información.
Poscondiciones	✍ El sistema presenta información referente a esta frase.
Flujo básico	
1. El sistema realiza una búsqueda de la frase en una lista. 2. Se identifican cuales son los temas, ejercicios o términos donde aparece. 3. Se presenta al usuario un listado con los materiales encontrados a fin de que seleccione el de su preferencia. 4. Se espera que el usuario seleccione uno de estos materiales para que sea conducido a su explicación.	
Flujos alternativos	
2a. En caso que la frase no esté dentro de su contenido teórico 1. El sistema lanza un *Mensaje de Error* informando que la frase no existe. 2. Termina el caso de uso. 4a. Si el usuario selecciona uno de los materiales de la lista: 1. Inicia el caso de uso **Explicar Material**. 2. Termina el caso de uso.	
Observaciones	
Las frases que se buscarán en el marco teórico de SÍMPLEX MEDIA y en los ejercicios pueden ser: una palabra o expresión a la que se le pueda dar un significado, en cuyo caso se mostrará el material correspondiente; el título de un tema o ejercicio del sistema; una palabra clave que sirva para hacer referencia a determinado tema o ejercicio. Estas frases son ingresadas por el usuario.	

4.2.8 Reproducir sonido

Caso de uso	Reproducir Sonido
Actor principal	Usuario, Sistema
Precondiciones	✍ El usuario debe haber seleccionado la opción de reproducir el sonido asociado al material activo.
Poscondiciones	✍ Se reproduce el sonido especificado.
Flujo básico	
1. El sistema cambiará a modo Pantalla Completa (en caso que no se encuentre en ese modo) y presentará al usuario las opciones para controlar la reproducción del sonido: reproducir, pausar, detener, retroceder y adelantar. 2. Se inicia la reproducción del sonido. 3. Se espera a que el usuario seleccione una de las opciones.	
Flujos alternativos	
3a. En caso que el usuario seleccione la opción *Reproducir*: 1. El sistema reproduce el sonido. 2. Termina el caso de uso. 3b. En caso que el usuario seleccione la opción *Pausar*: 1. El sistema pausa el sonido. 2. Termina el caso de uso. 3c. En caso que el usuario seleccione la opción *Detener*: 1. El sistema detiene el sonido. 2. Termina el caso de uso. 3d. En caso que el usuario seleccione la opción *Retroceder*: 1. El sistema retrocede el sonido. 2. Termina el caso de uso. 3e. En caso que el usuario seleccione la opción *Adelantar*: 1. El sistema adelanta el sonido. 2. Termina el caso de uso.	
Observaciones	
Las opciones de retroceder y adelantar pueden usarse a través de un deslizador de tiempo que indique en qué segundo se desea continuar con la reproducción. Es posible (según la configuración que establezca el usuario) que la reproducción se efectúe a través de un dibujo animado (o agente) que simulará estar hablando con el usuario, en ese caso, no se habilitarán las opciones para retroceder y adelantar la	

reproducción.

4.2.9 Reproducir animación

Caso de uso	Reproducir Animación
Actor principal	Usuario, Sistema
Precondiciones	✍ El usuario debe haber seleccionado la opción de reproducir la animación asociada al material activo.
Poscondiciones	✍ Se reproduce el sonido especificado.
Flujo básico	
1. El sistema abrirá una ventana independiente en la que se ejecutará la animación. 2. El sistema presentará al usuario las opciones para controlar la reproducción de la animación: reproducir, pausar, detener, retroceder y adelantar. 3. Se inicia la reproducción de la animación. 4. Se espera a que el usuario seleccione una de las opciones.	
Flujos alternativos	
4a. En caso que el usuario seleccione la opción *Reproducir*: 1. El sistema reproduce la animación. 2. Termina el caso de uso. 4b. En caso que el usuario seleccione la opción *Pausar*: 1. El sistema pausa la animación. 2. Termina el caso de uso. 4c. En caso que el usuario seleccione la opción *Detener*: 1. El sistema detiene la animación. 2. Termina el caso de uso. 4d. En caso que el usuario seleccione la opción *Retroceder*: 1. El sistema retrocede la animación. 2. Termina el caso de uso. 4e. En caso que el usuario seleccione la opción *Adelantar*: 1. El sistema adelanta la animación. 2. Termina el caso de uso.	
Observaciones	
Las opciones de retroceder y adelantar pueden usarse a través de un deslizador de tiempo que indique en qué segundo se desea continuar con la reproducción. Es posible (según la configuración que establezca el usuario) que la reproducción del	

sonido de la animación se realice a través de un dibujo animado (o agente) que simulará estar hablando con el usuario, en ese caso, no se habilitarán las opciones para retroceder y adelantar la reproducción.

4.2.10 Nuevo ejercicio

Caso de uso	Nuevo Ejercicio
Actores	Usuario
Precondiciones	✍ Se debe haber seleccionado la opción de Crear un Nuevo Ejercicio.
Poscondiciones	✍ Se presenta la solución del ejercicio planteado por el usuario.
Flujo básico	
1. El sistema presenta al usuario los diferentes tipos de ejercicios que se pueden desarrollar con SÍMPLEX MEDIA. 2. Se espera a que el usuario seleccione un tipo de ejercicio y ordene su creación, o que cancele la operación. 3. El sistema crea un problema predeterminado según el tipo de ejercicio que el usuario haya seleccionado. 4. Se abre un *visualizador de ejercicios* y se muestra el ejercicio planteado en pantalla. 5. Inicia el caso de uso **Gestionar Ejercicio.**	
Flujos alternativos	
2a. En caso que el usuario no desee crear un nuevo ejercicio. 1. Termina el caso de uso.	
Observaciones	
El *visualizador de ejercicios* es una página que presentará el planteamiento del ejercicio dando la posibilidad de modificarlo. Además mostrará las opciones de configuración de desarrollo que permitirán al usuario especificar cómo desea que se le proporcione la solución del ejercicio, y presentará las opciones de gestión del ejercicio, como, Resolver, Crear Nuevo, Abrir, Guardar y Cerrar. En caso que el usuario resuelva el ejercicio, la solución se mostrará en esa misma página.	

4.2.11 Gestionar ejercicio

Caso de uso	Gestionar Ejercicio
Actor principal	Sistema
Precondiciones	✍ Se debe tener abierto un ejercicio
Poscondiciones	✍ Se ejecuta la opción que el Usuario haya seleccionado.
Flujo básico	
1. El sistema muestra las opciones que conducirán al usuario a resolver, guardar, modificar o imprimir el ejercicio. 2. Se espera que el usuario seleccione una de las opciones.	
Flujos alternativos	
3a. En caso que el Usuario seleccione la opción de resolver el ejercicio. 1. Inicia el caso de uso **Resolver Ejercicio.** 2. Reinicia el caso de uso. 3b. En caso que el Usuario seleccione la opción de guardar el ejercicio. 1. Inicia el caso de uso **Guardar Ejercicio.** 2. Reinicia el caso de uso. 3c. En caso que el Usuario seleccione la opción de modificar el ejercicio. 1. Inicia el caso de uso **Modificar Planteamiento del Ejercicio.** 2. Reinicia el caso de uso. 3d. En caso que el Usuario seleccione la opción de imprimir el ejercicio. 1. Inicia el caso de uso **Imprimir.** 2. Reinicia el caso de uso. 3e. En caso que el Usuario seleccione la opción Cancelar. 1. Termina el caso de uso.	

4.2.12 Resolver ejercicio

Caso de uso	Resolver Ejercicio
Actor principal	Sistema
Precondiciones	✍ Debe haber un ejercicio abierto en el sistema.
Poscondiciones	✍ El sistema presentará la solución del ejercicio.
Flujo básico	
1. Analizar el planteamiento del problema ingresado por el usuario, en busca de algún posible error en el planteamiento y de las configuraciones realizadas por el usuario que indica el tipo de respuesta que desea. 2. El sistema resuelve el ejercicio. 3. El sistema proporciona los resultados deseados según la configuración realizada por el usuario. 4. El sistema permanece alerta ante cualquier solicitud del usuario en cuanto a más información.	
Flujos alternativos	
1a. En caso que el ejercicio esté mal planteado: 1. El sistema muestra un *Mensaje de Error* informando el problema**.** 2. Termina el caso de uso. 2a. En caso que el ejercicio no se pueda resolver: 1. El sistema muestra un *Mensaje de Error* informando la razón**.** 2. Termina el caso de uso. 4a. En caso que el usuario desee informarse acerca de una palabra. 1. Inicia el caso de uso **Buscar Frase.** 2. Se continúa con el proceso de desarrollo del ejercicio. 4b. En caso que el usuario desee informarse acerca de un tema. 1. Inicia el caso de uso **Explicar Material.** 2. Se continúa con el proceso de desarrollo del ejercicio.	
Observaciones	
Cuando se selecciona Resolver Ejercicio con Explicaciones, lo que el sistema hará es proporcionar una explicación adecuada del por qué de cada paso. Si se desea conocer lo que representa o significa cada resultado obtenido, en necesario recurrir al módulo de Ejercicios Prácticos y estudiar los ejercicios analizados por el sistema.	

4.2.13 Guardar ejercicio

Caso de uso	Guardar Ejercicio
Actor principal	Usuario
Precondiciones	✍ Se debe tener un ejercicio planteado en pantalla o se puede tener abierto un archivo de algún ejercicio.
Poscondiciones	✍ El sistema guardará el archivo en el lugar que el usuario indique.
Flujo básico	
1. El sistema permite al usuario especificar el lugar donde desea guardar el archivo de ejercicio. 2. El sistema guardará el archivo en el lugar que el usuario indique.	

4.2.14 Modificar planteamiento del ejercicio

Caso de uso	Modificar Planteamiento del Ejercicio
Actor principal	Usuario
Precondiciones	✍ Se debe tener abierto un archivo de algún ejercicio.
Poscondiciones	✍ El sistema modifica planteamiento y la configuración de cómo se va a desarrollar el ejercicio.
Flujo básico	
1. El sistema presenta en una ventana el planteamiento del ejercicio abierto. 2. El sistema presenta en dicha ventana las opciones para modificar el planteamiento de forma manual, para seleccionar uno del sistema o para que éste lo genere de forma aleatoria. 3. El sistema espera a que el usuario modifique el planteamiento del ejercicio y esté de acuerdo con la modificación realizada. 4. El sistema presenta en pantalla el ejercicio ya modificado.	
Flujos alternativos	
3a. En caso que el usuario seleccione la opción de modificar el planteamiento del ejercicio de forma manual. 1. Inicia el caso de uso **Realizar Planteamiento del Ejercicio.** 2. Reinicia el caso de uso en el paso 3. 3b. En caso que el usuario seleccione la opción de seleccionar un planteamiento del sistema: 1. Inicia el caso de uso **Seleccionar Planteamiento.** 2. Reinicia el caso de uso en el paso 3.	

3c. En caso que el usuario seleccione la opción de para que el sistema genere un planteamiento aleatoriamente: 1. Inicia el caso de uso **Generar Planteamiento.** 2. Reinicia el caso de uso en el paso 3.

4.2.15 Realizar planteamiento del ejercicio

Caso de uso	Realizar Planteamiento del Ejercicio
Actor principal	Usuario
Precondiciones	✍ El usuario debe haber decidido ingresar el planteamiento del ejercicio.
Poscondiciones	✍ El planteamiento del ejercicio queda terminado.
Flujo básico	
1. El sistema muestra al usuario los campos para que los datos del planteamiento del problema. 2. El usuario selecciona la opción si desea o no ingresar el planteamiento. 3. El sistema valida el planteamiento del problema ingresado por el usuario.	
Flujos alternativos	
2a. En caso de que el usuario no desee ingresar el planteamiento del problema. 1. Termina el caso de uso. 3a. En caso que el problema esté mal planteado 1. El sistema muestra un *Mensaje de Error* informando que el ejercicio está mal planteado. 2. Reinicia el caso de uso.	

4.2.16 Seleccionar planteamiento

Caso de uso	Seleccionar Planteamiento
Actor principal	Usuario
Precondiciones	✍ El usuario debe haber seleccionado la opción de seleccionar el planteamiento de un ejercicio almacenado en el sistema.
Poscondiciones	✍ El planteamiento del ejercicio queda terminado.
	Flujo básico
1. El sistema presenta al usuario un listado de ejercicios a fin de que el seleccione uno de estos. 2. Se espera a que el usuario seleccione uno de los planteamientos.	

4.2.17 Generar planteamiento

Caso de uso	Generar Planteamiento
Actor principal	Usuario
Precondiciones	✍ El usuario debe haber seleccionado la opción de Generar Planteamiento de un ejercicio generado por el sistema.
Poscondiciones	✍ El planteamiento del ejercicio queda terminado.
	Flujo básico
1. El sistema genera de forma aleatoria el planteamiento de un ejercicio, y lo muestra en pantalla.	

4.2.18 Abrir ejercicio

Caso de uso	Abrir Ejercicio
Actor principal	**Usuario**
Precondiciones	✍ **El usuario debe haber seleccionado la opción Abrir Ejercicio.**
Poscondiciones	✍ **El sistema muestra el ejercicio.**
	Flujo básico
1. El sistema permite al usuario especificar la dirección del archivo de ejercicio que desea abrir. 2. Inicia el caso de uso **Recuperar Ejercicio**.	

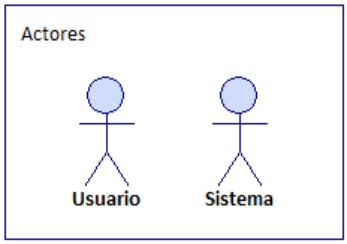
4.2.19 Recuperar ejercicio

Caso de uso	Recuperar Ejercicio
Actor principal	**Sistema**
Precondiciones	✍ **El usuario debe haber indicado el ejercicio que desea abrir.**
Poscondiciones	✍ **El sistema muestra el ejercicio.**
Flujo básico	
1. El sistema abre el ejercicio indicado. 2. El sistema analiza el archivo para comprobar si posee el formato correcto de archivos de ejercicio emitidos por SÍMPLEX MEDIA. 3. El sistema abre el archivo. 4. Inicia el caso de uso *Gestionar Ejercicio*.	
Flujos alternativos	
2a. En caso que el archivo de ejercicio seleccionado por el usuario no posea un formato correcto. 1. El sistema muestra un *Mensaje de Error* informando que dicho archivo no puede ser abierto. 2. Termina el caso de uso.	

4.2.20 Imprimir

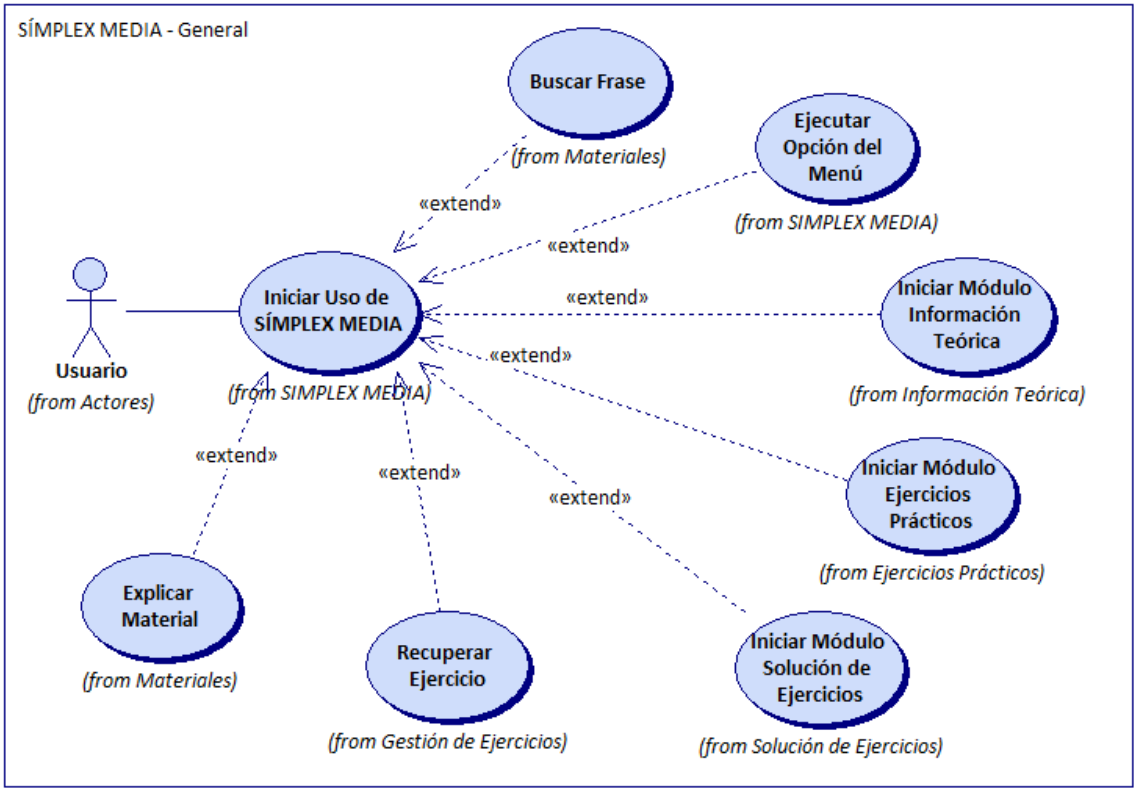
Caso de uso	Imprimir
Actor principal	Sistema
Precondiciones	✍ Se debe tener abierto un archivo de algún ejercicio o de algún material.
Poscondiciones	✍ El sistema imprime el archivo o material.
Flujo básico	
1. Se muestra una ventana de configuración de impresión. 2. Si el usuario acepta realizar la impresión, el sistema imprime el archivo o material.	

4.3 DIAGRAMA DE ACTORES

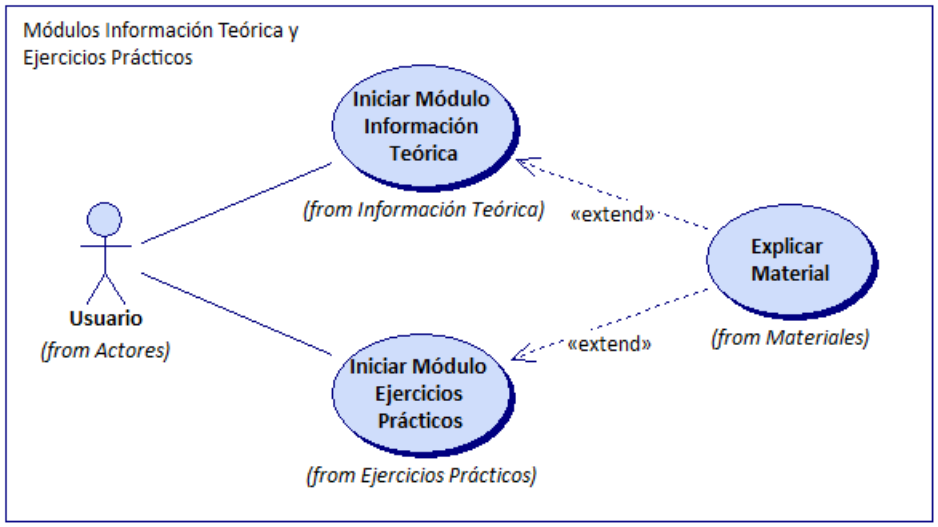


4.4 DIAGRAMAS DE CASOS DE USO

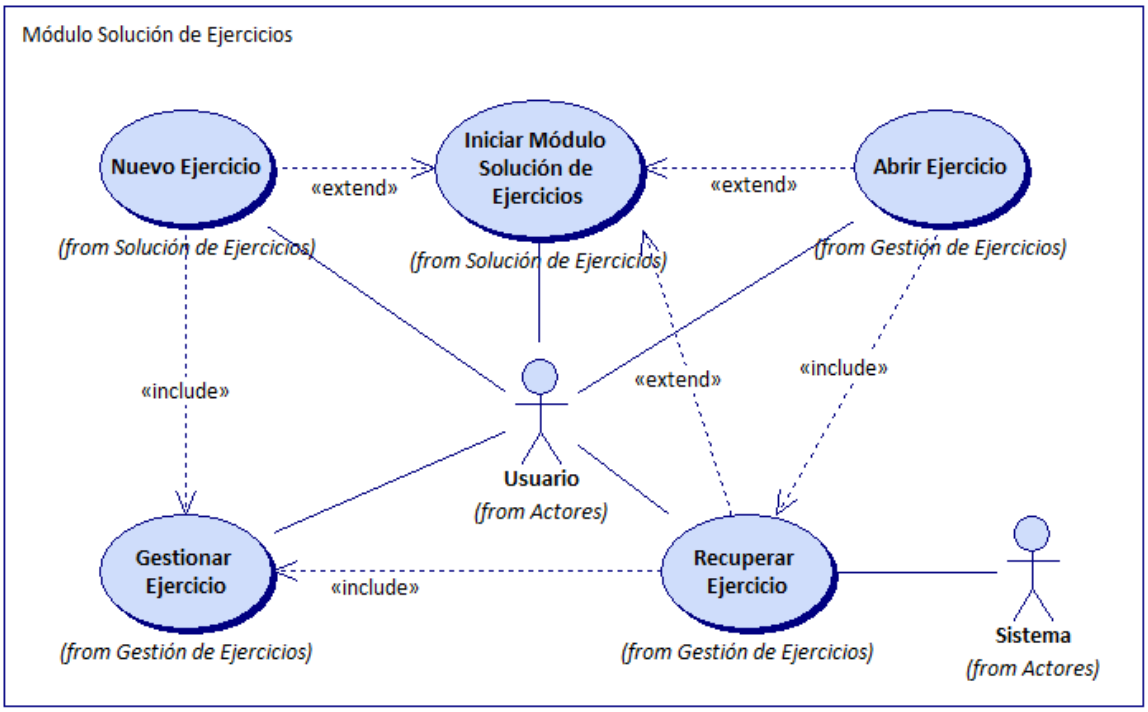
4.4.1 Diagrama: Uso general del sistema



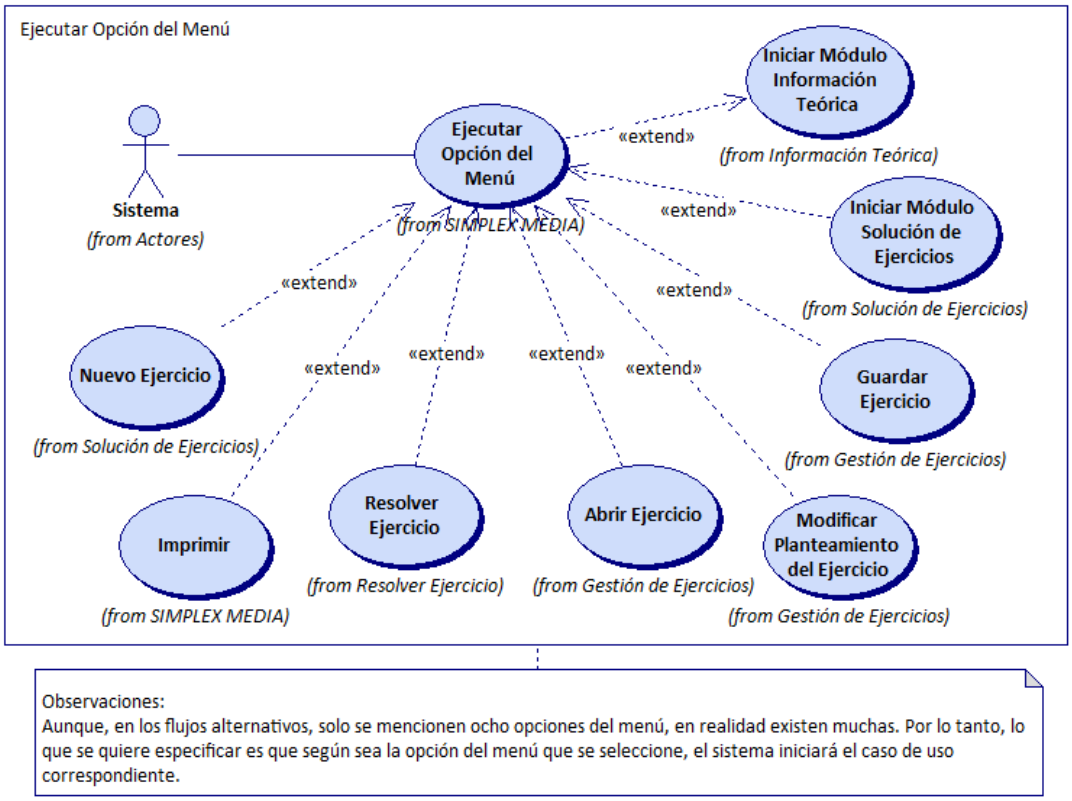
4.4.2 Diagrama: Uso de los módulos información teórica y ejercicios prácticos



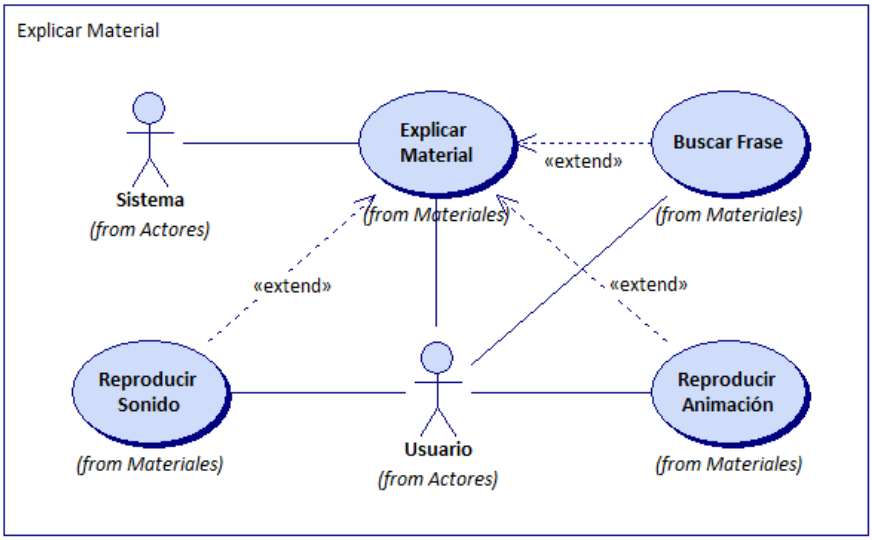
4.4.3 Diagrama: Uso del módulo solución de ejercicios



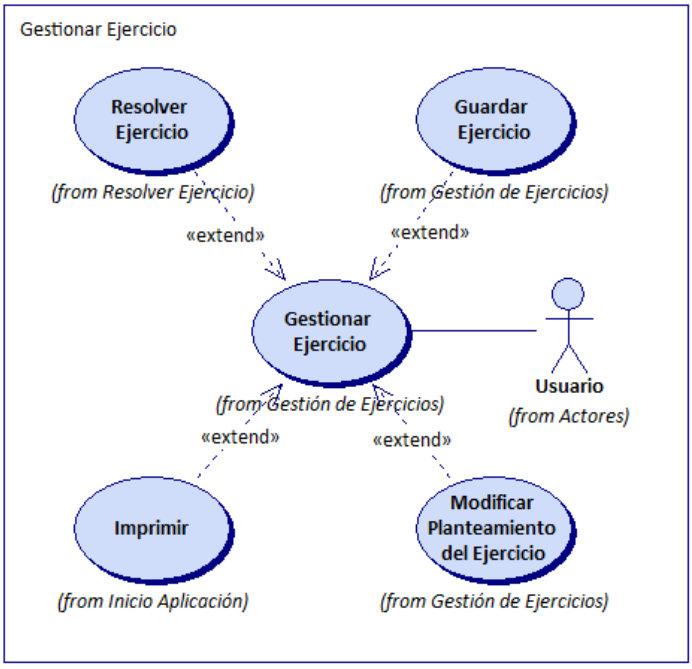
4.4.4 Diagrama: Ejecutar opción del menú



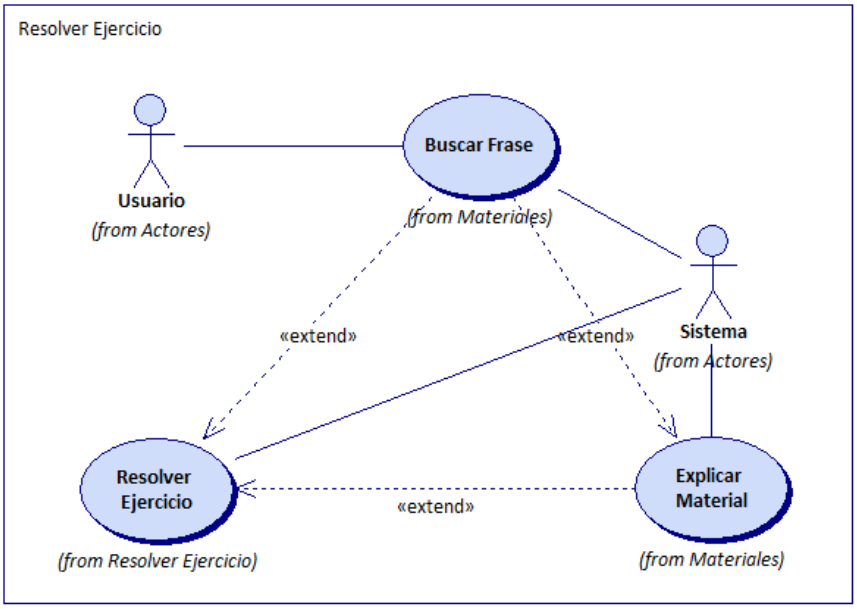
4.4.5 Diagrama: Explicar material



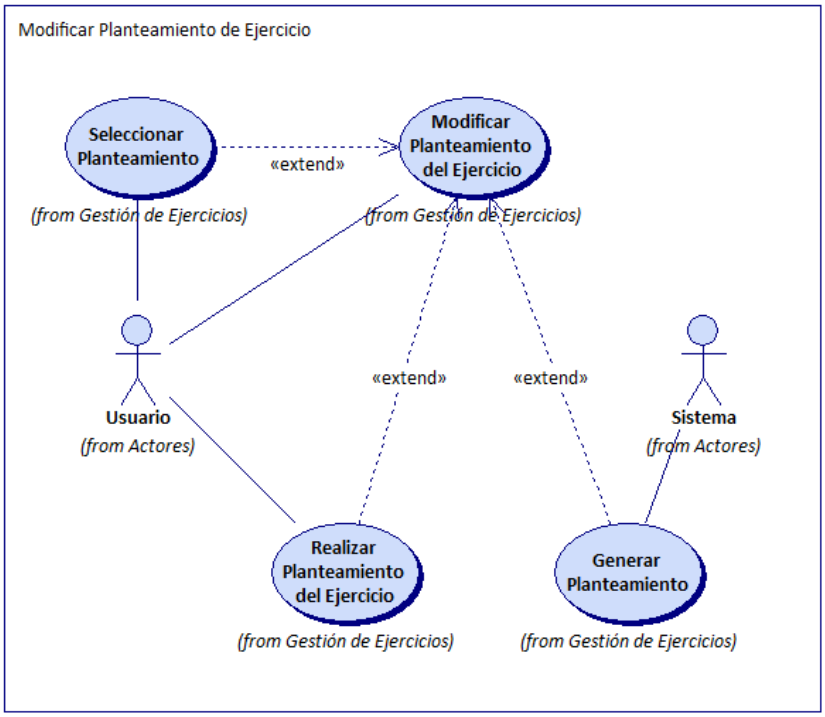
4.4.6 Diagrama: Gestionar ejercicio



4.4.7 Diagrama: Resolver ejercicio



4.4.8 Diagrama: Modificar planteamiento de ejercicio

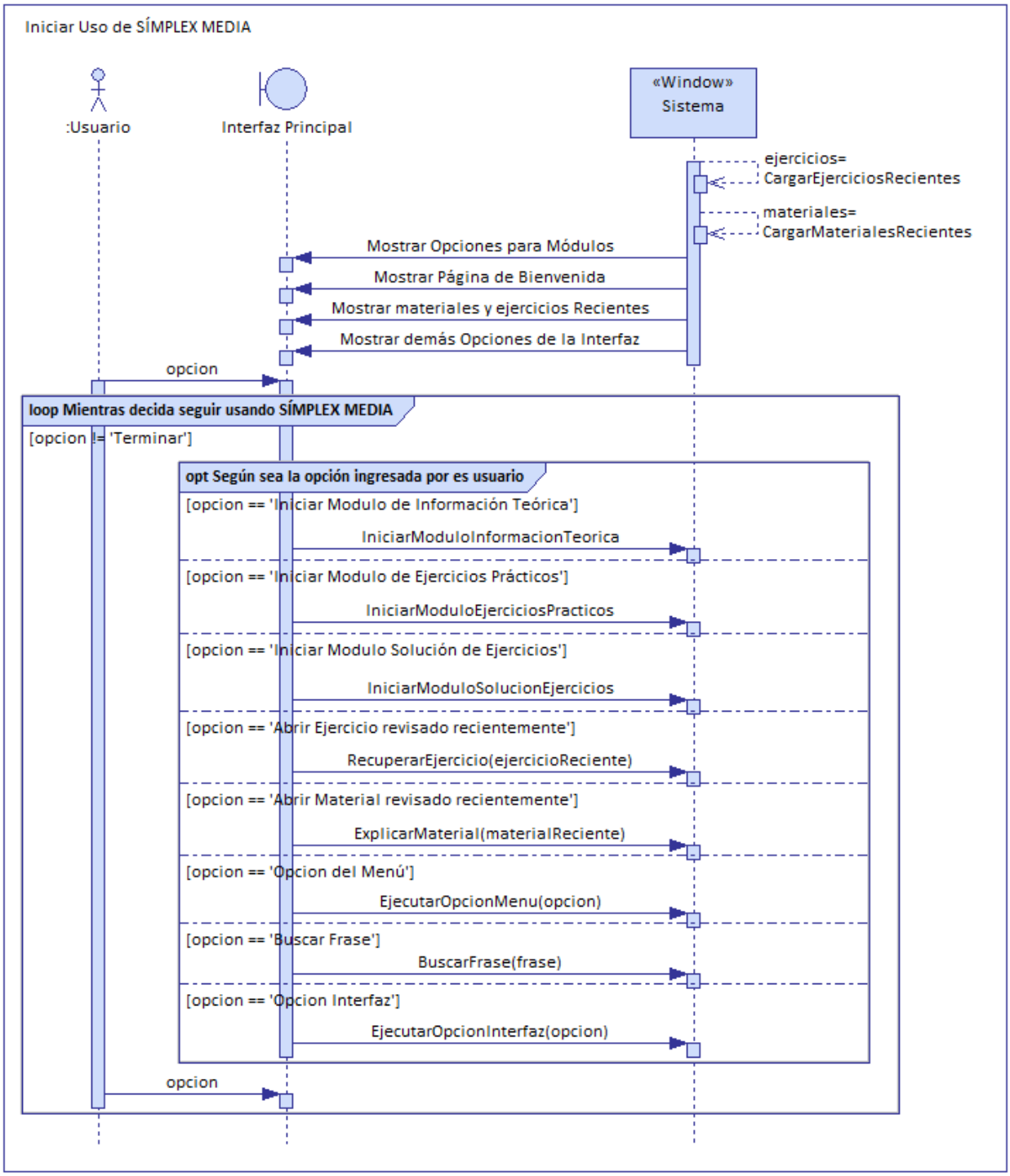


4.5 DIAGRAMAS DE SECUENCIA DEL SISTEMA

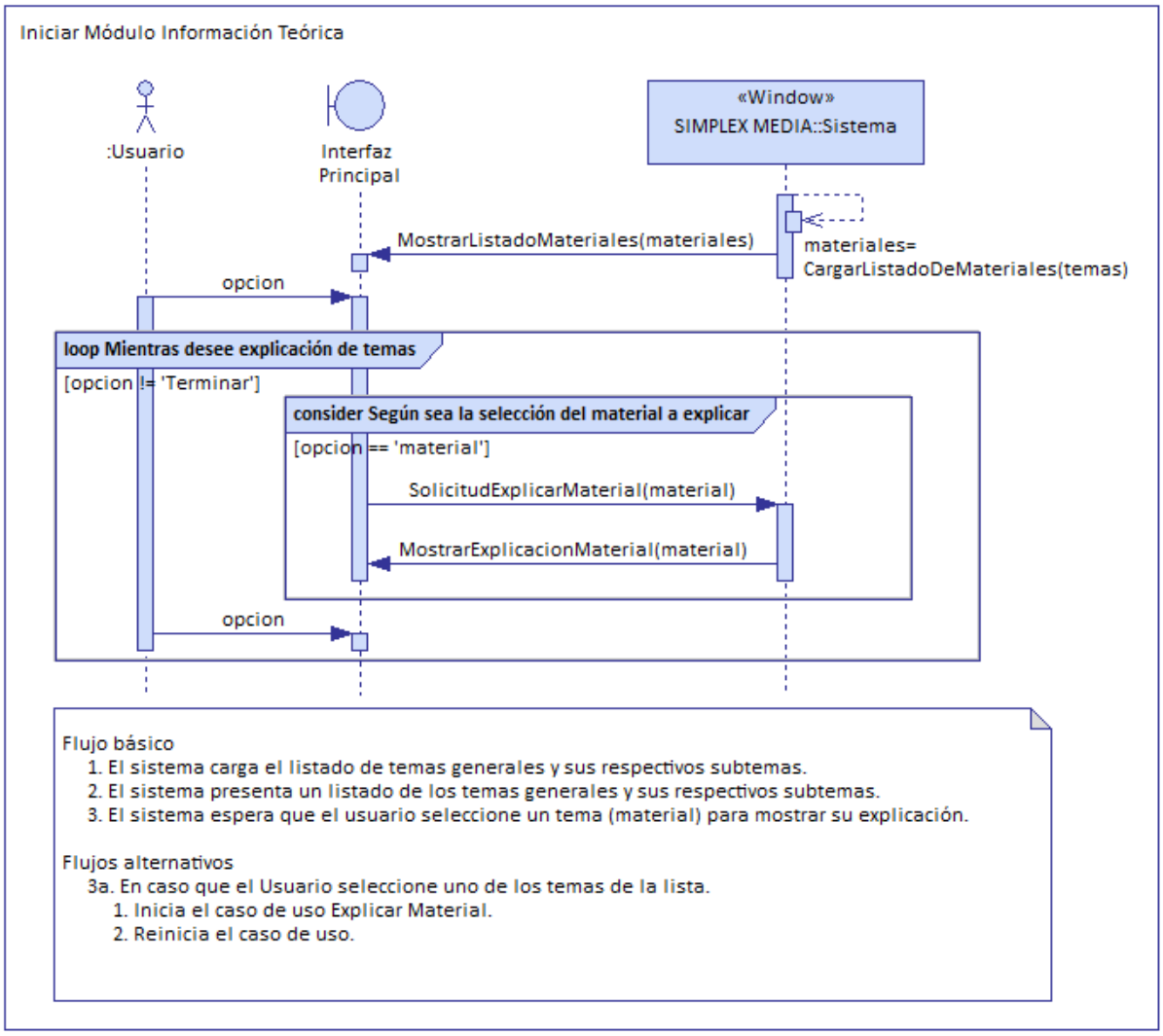
Los siguientes diagramas de secuencia del sistema se han creado con el objetivo de visualizar el flujo de comunicación entre el actor de cada caso de uso y la sección del sistema encargada de responder sus mensajes. Con estos diagramas se pueden conocer las secuencias lógicas de las iteraciones de forma detallada y ordenadas cronológicamente. Cada diagrama de secuencia es la representación de cada uno de los casos de uso descritos en los puntos anteriores.

En dichos diagramas la Clase Sistema representa el núcleo de SÍMPLEX MEDIA. Dicho núcleo hace referencia a aquella clase encargada de realizar determinada tarea. Por ejemplo, en caso que en un diagrama el usuario solicite Guardar un Ejercicio, se establecerá comunicación con Sistema para hacerlo; pero, en realidad, la comunicación se establece es con aquella clase encargada de la gestión de archivos de ejercicios. En el modelo de análisis se describen todas las clases del análisis.

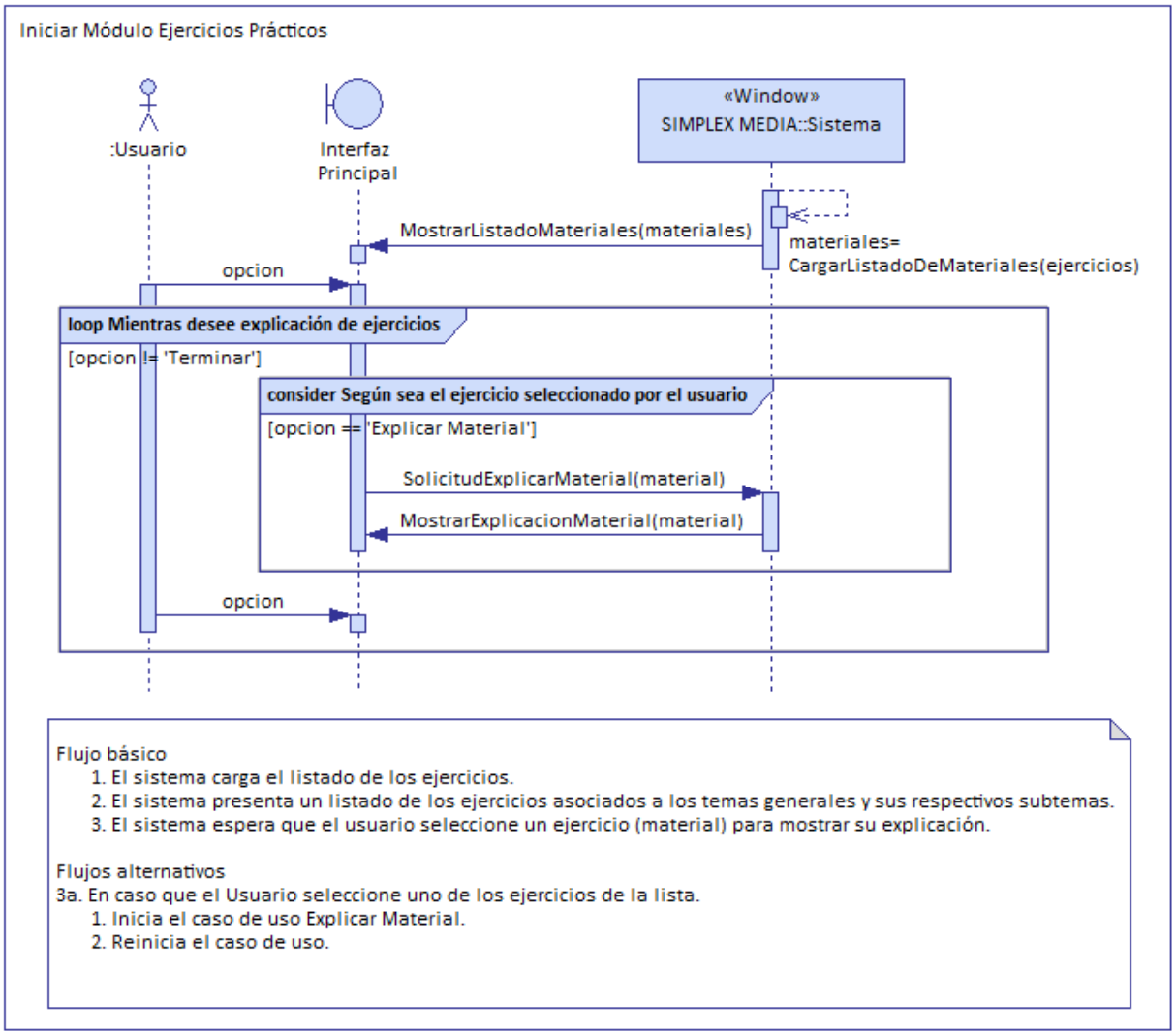
4.5.1 Iniciar uso de SÍMPLEX MEDIA



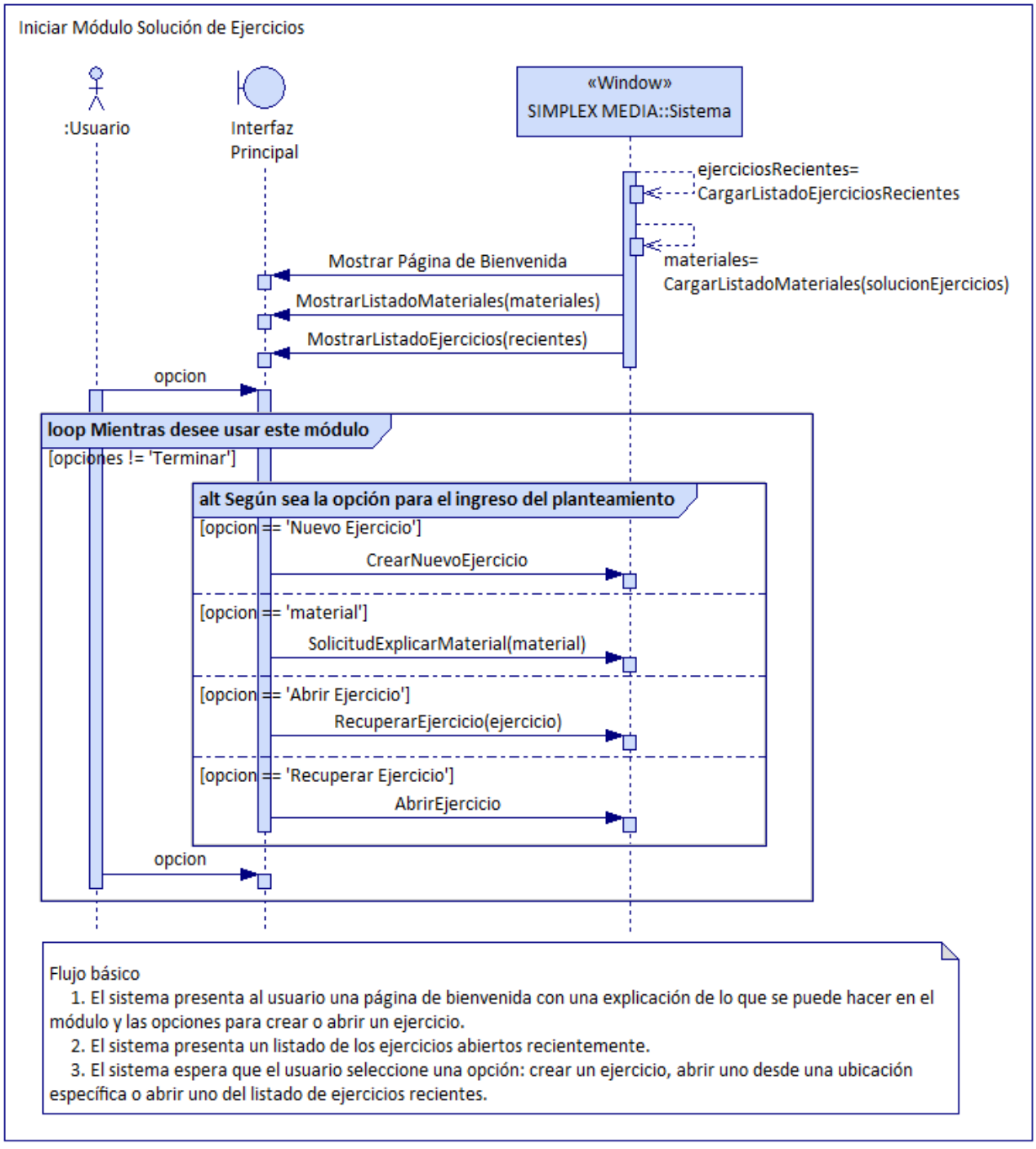
4.5.2 Iniciar módulo información teórica



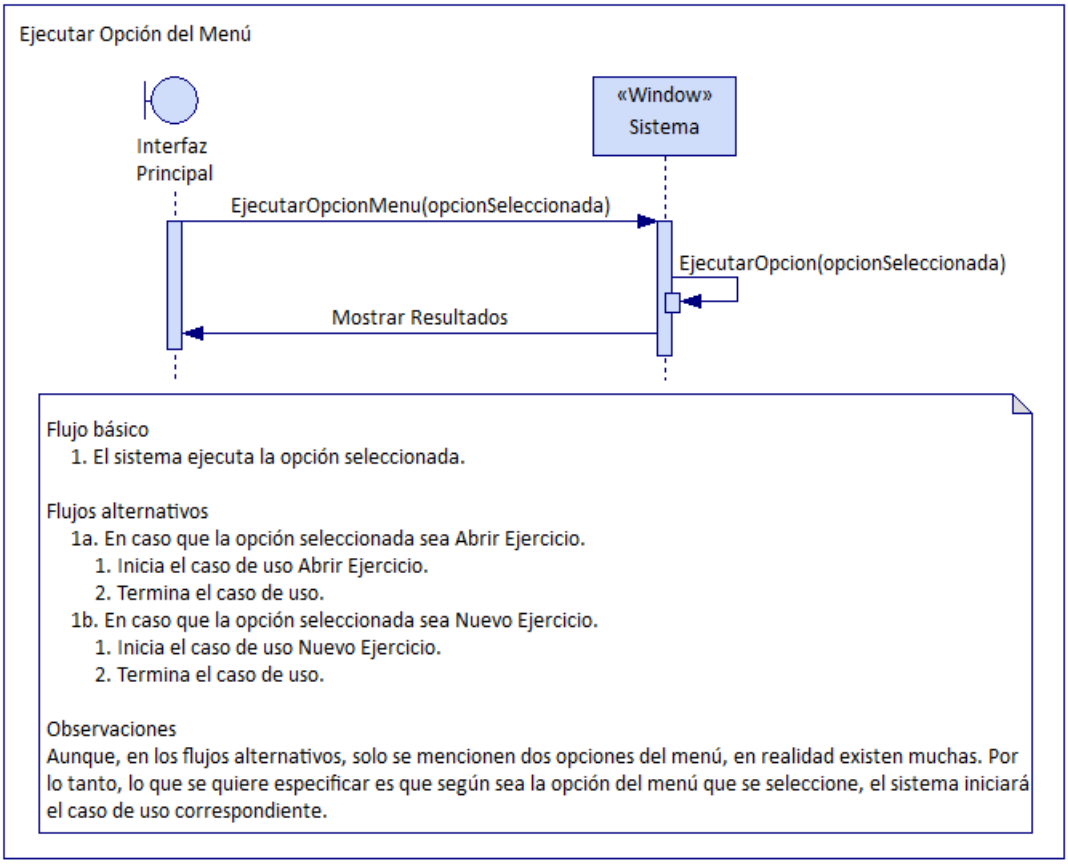
4.5.3 Iniciar módulo ejercicios prácticos



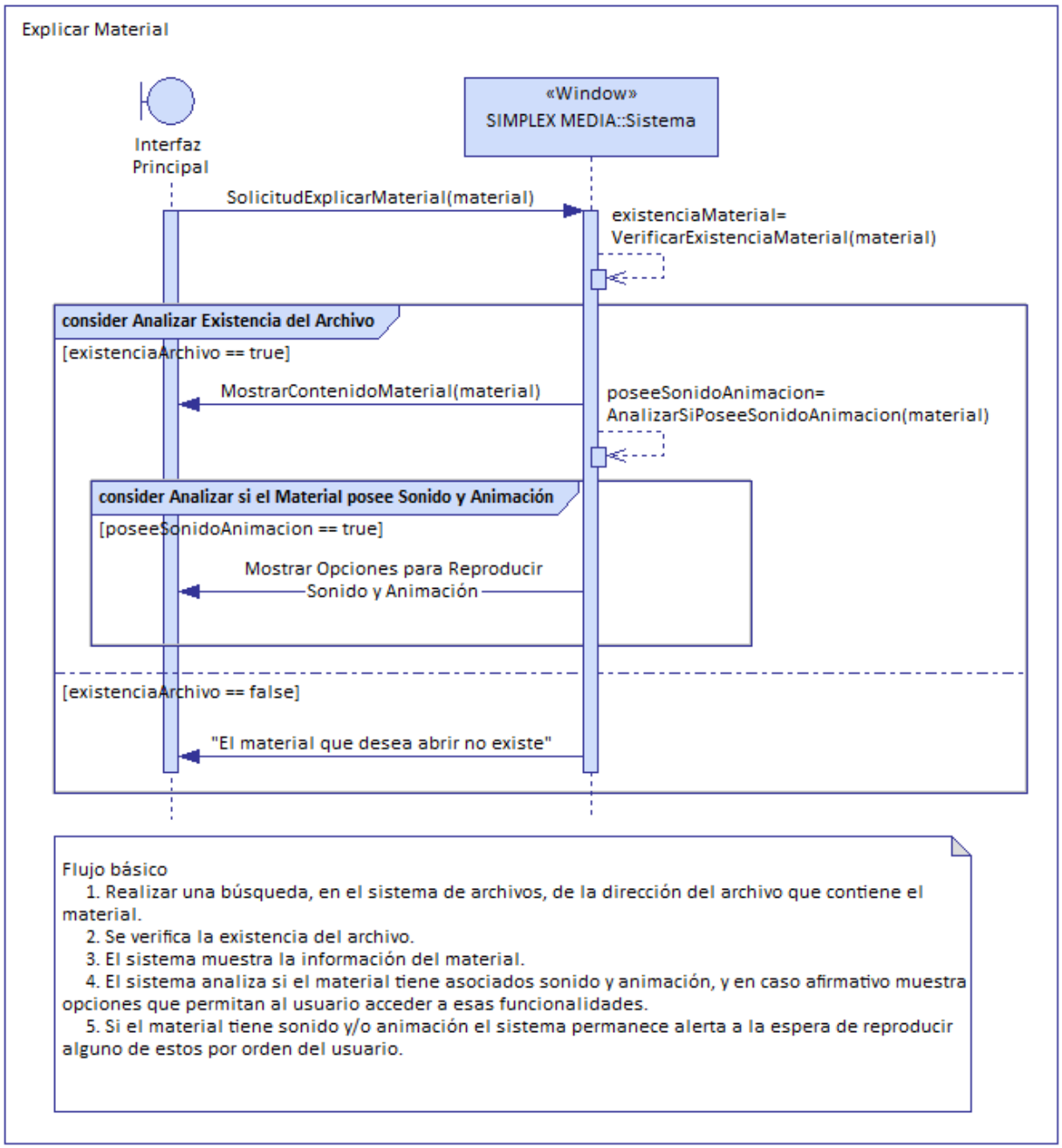
4.5.4 Iniciar módulo solución de ejercicios



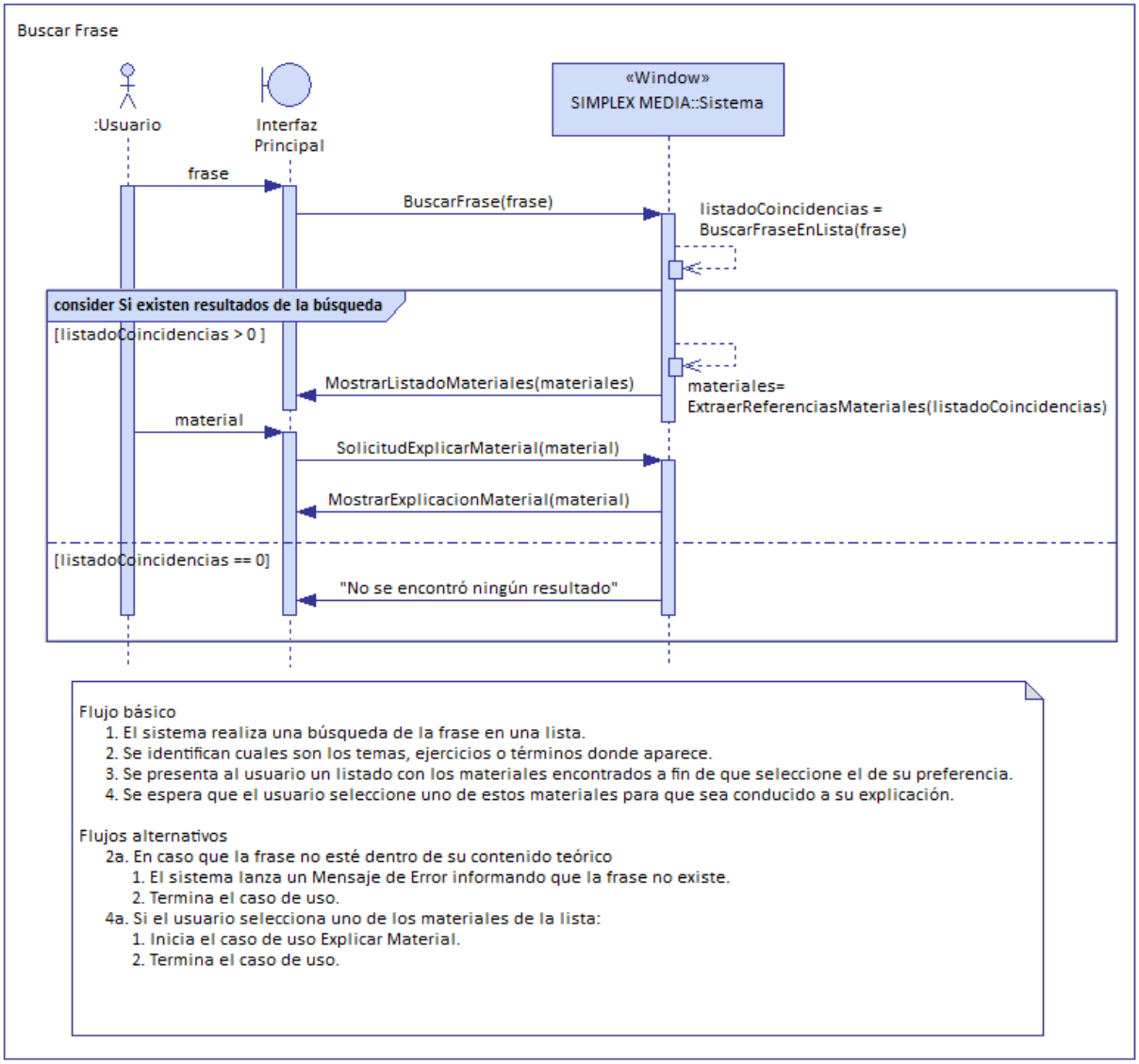
4.5.5 Ejecutar opción del menú



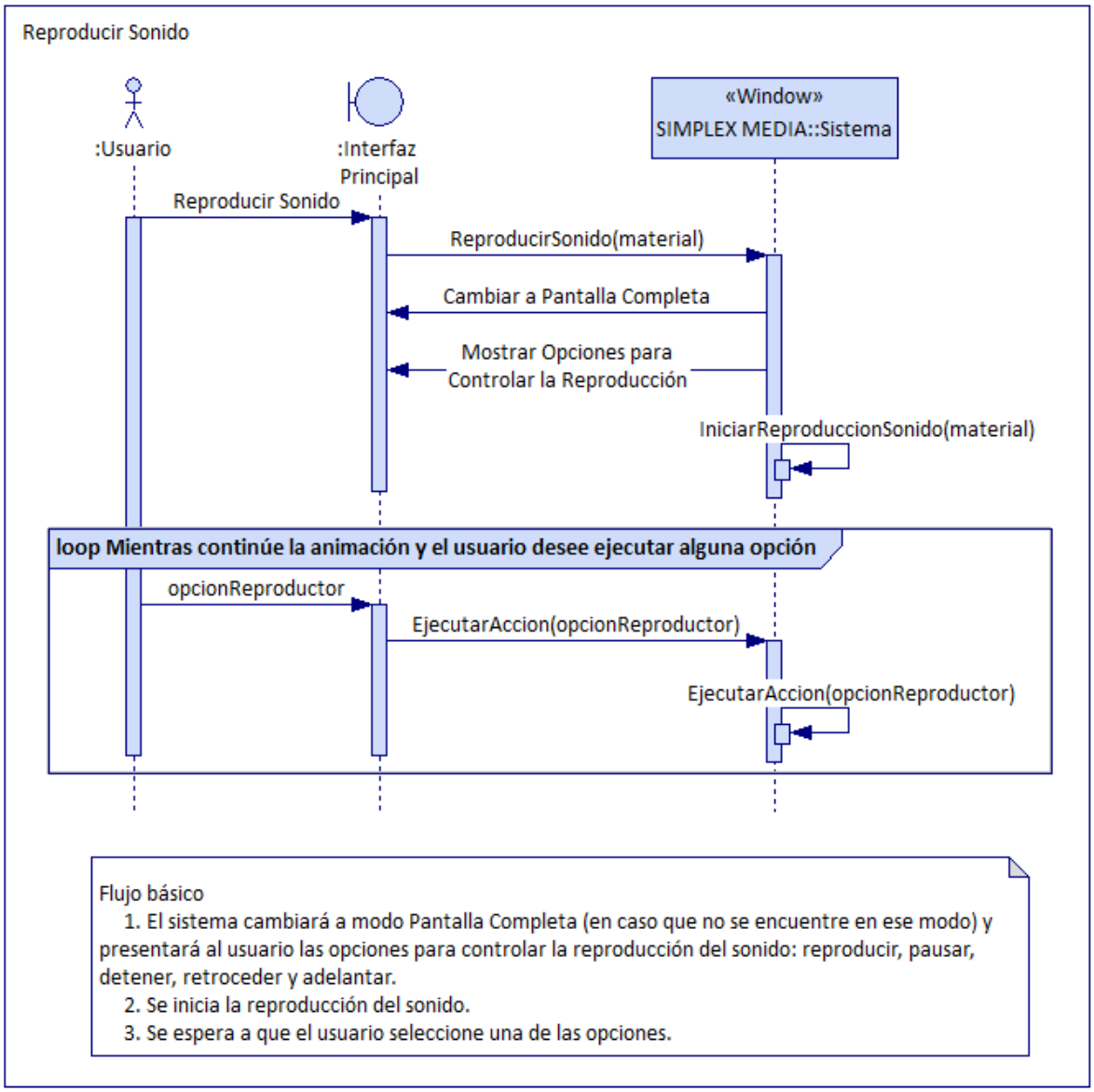
4.5.6 Explicar material



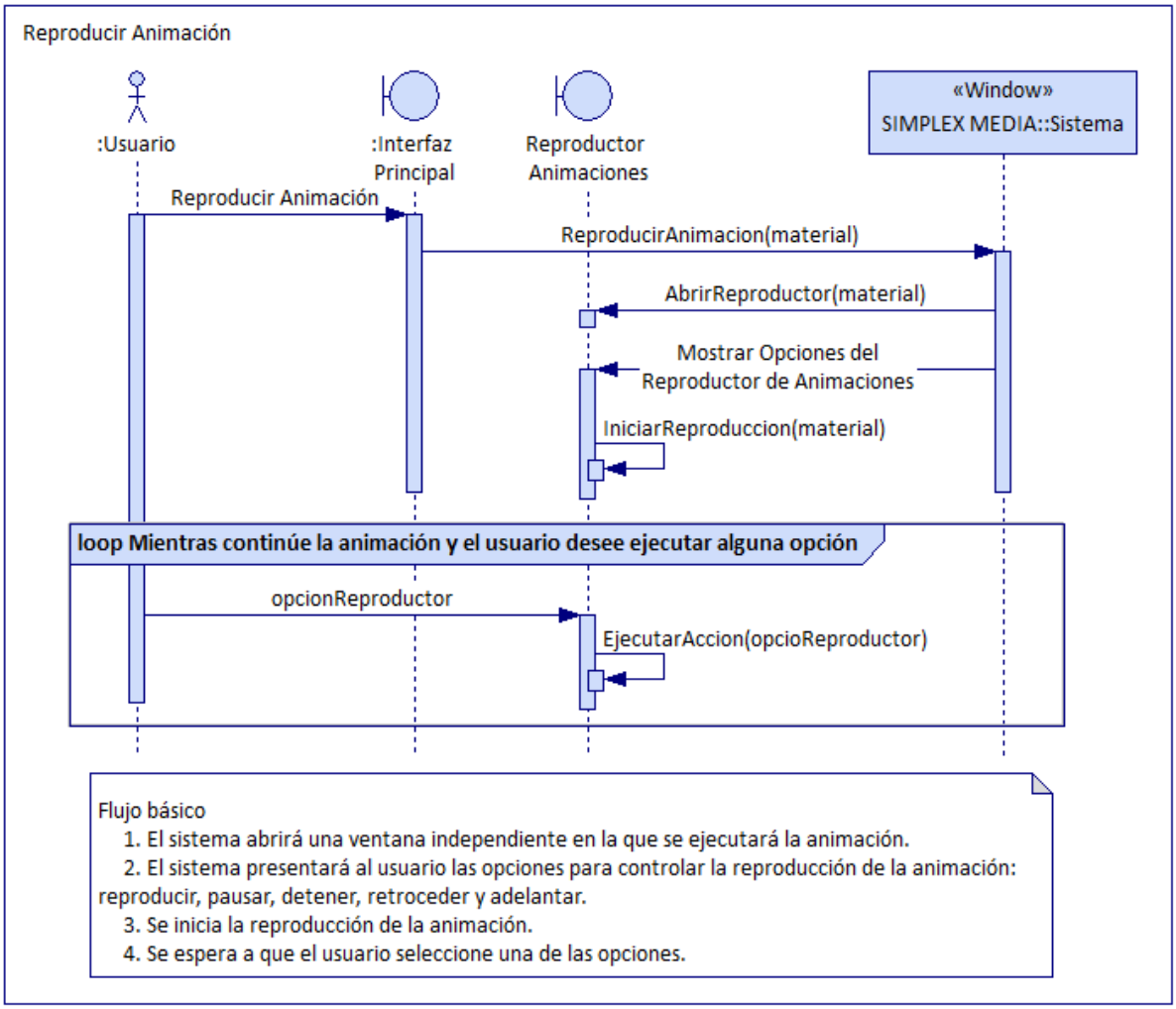
4.5.7 Buscar frase



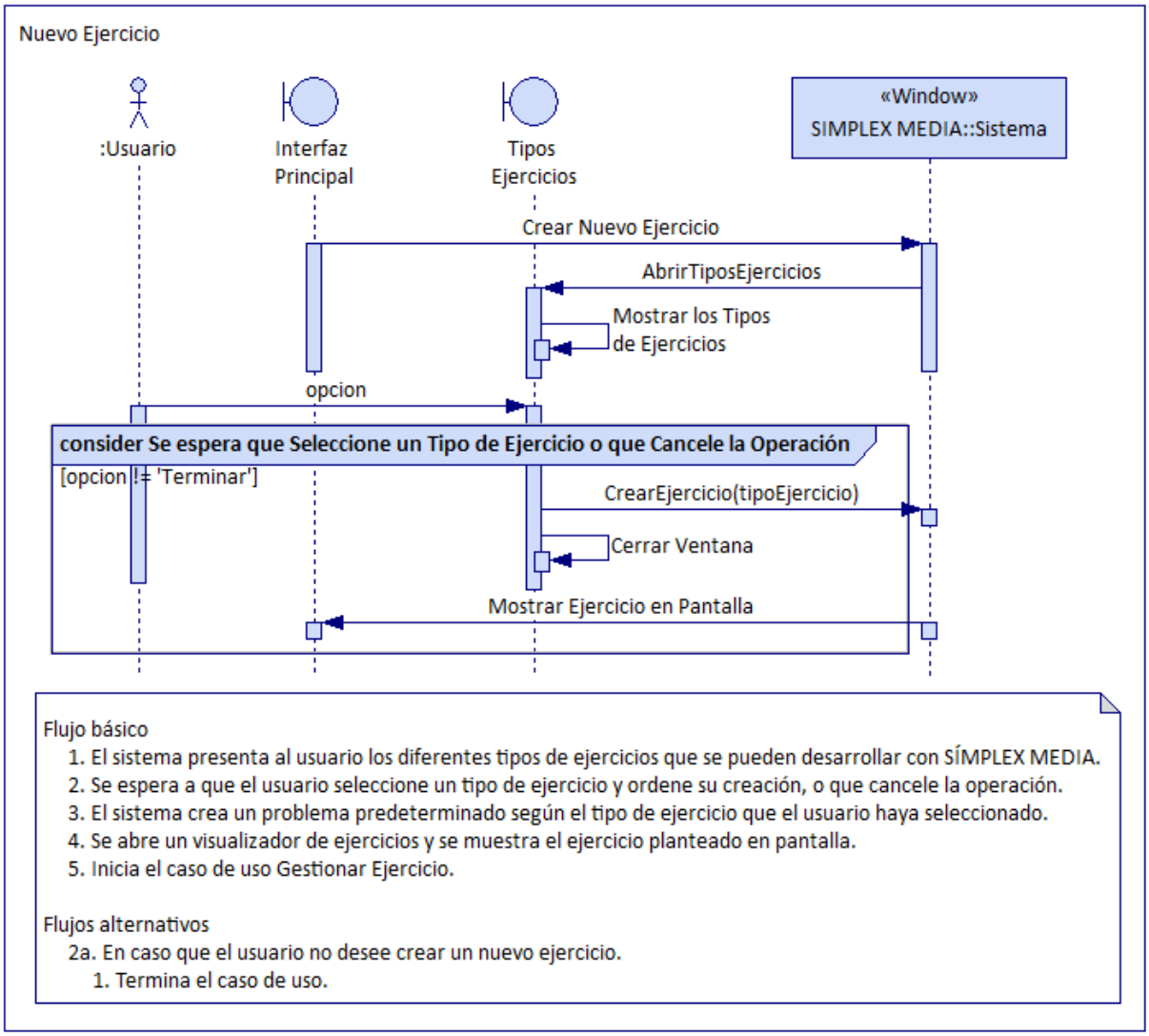
4.5.8 Reproducir sonido



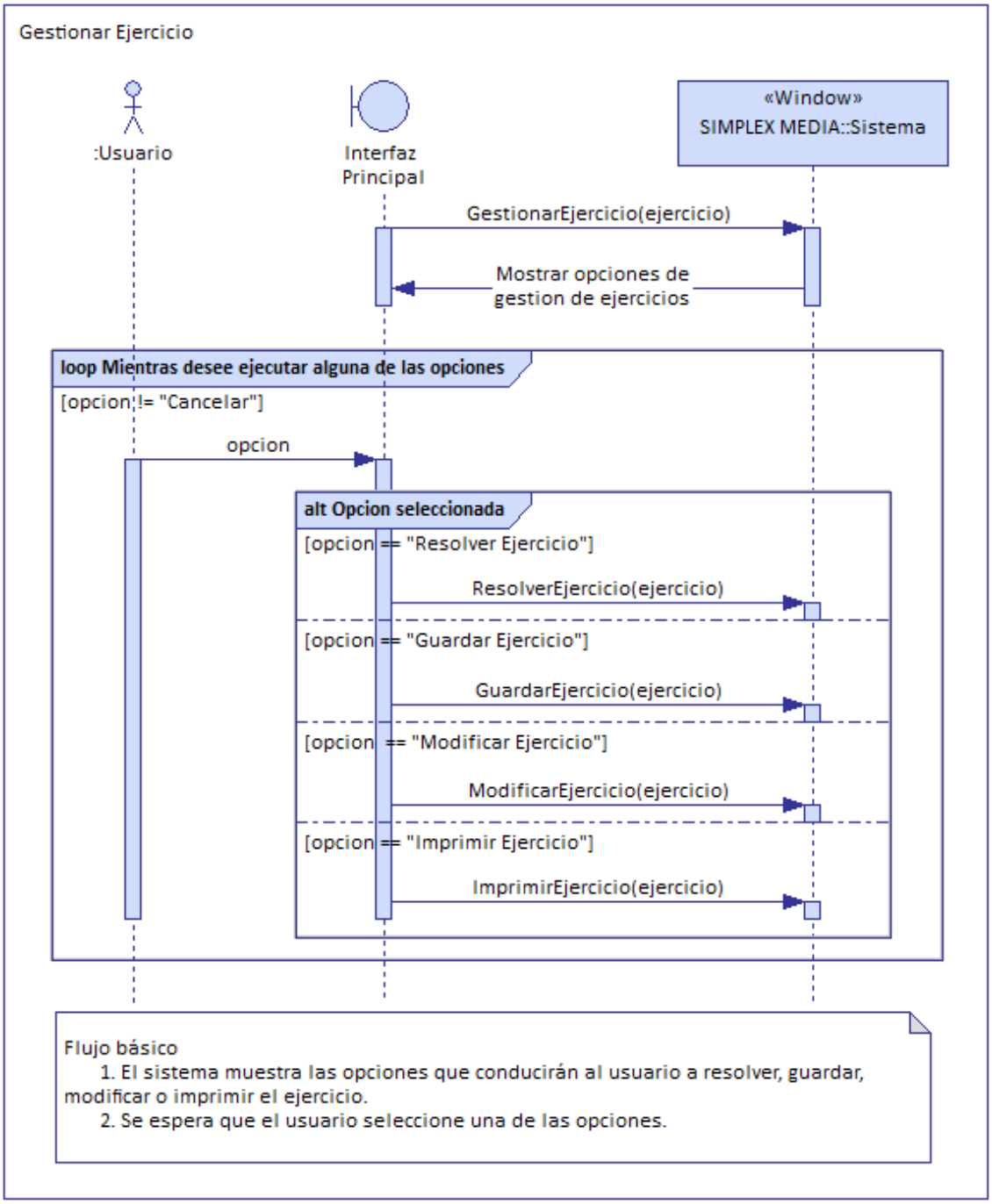
4.5.9 Reproducir animación



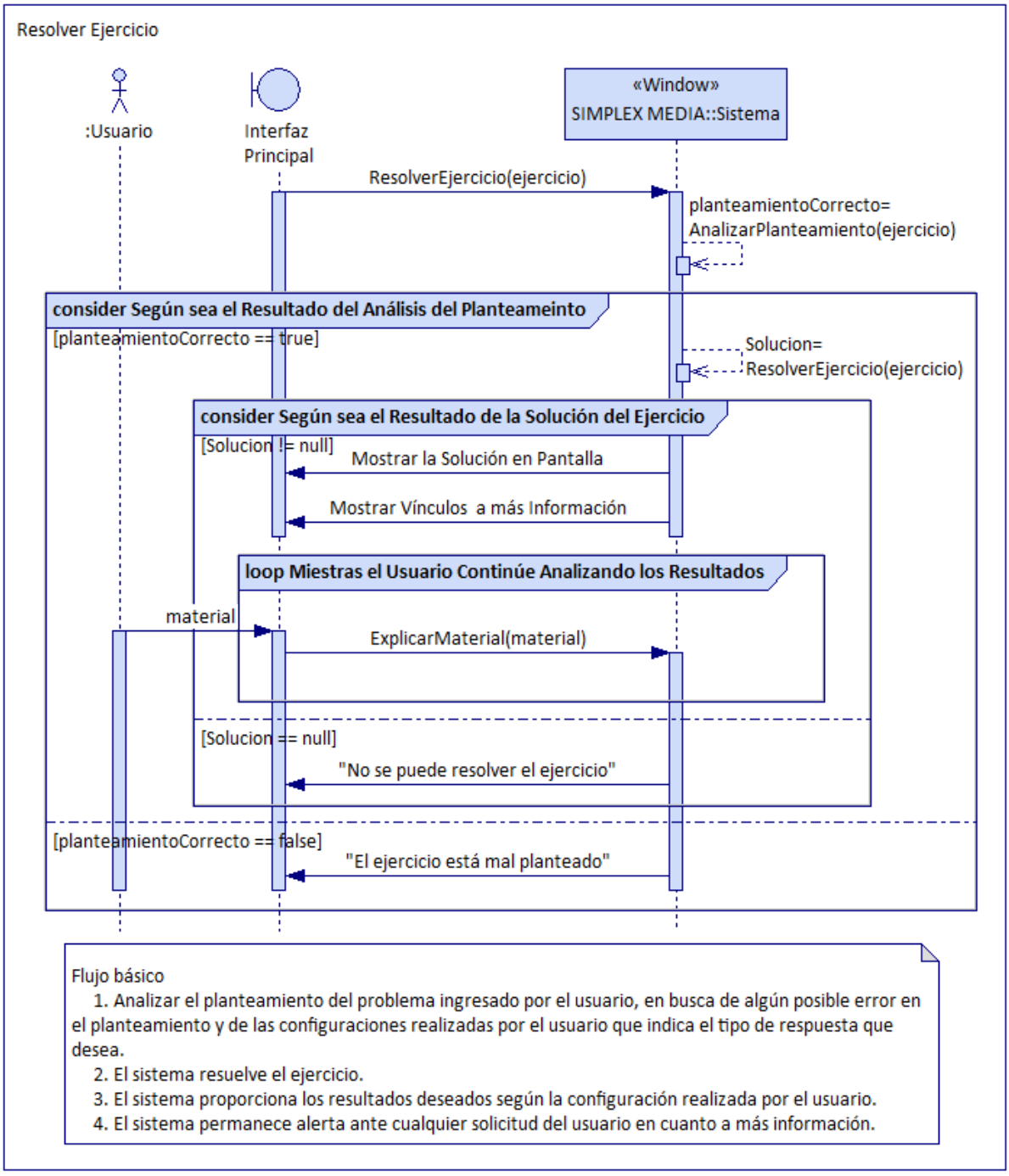
4.5.10 Nuevo ejercicio



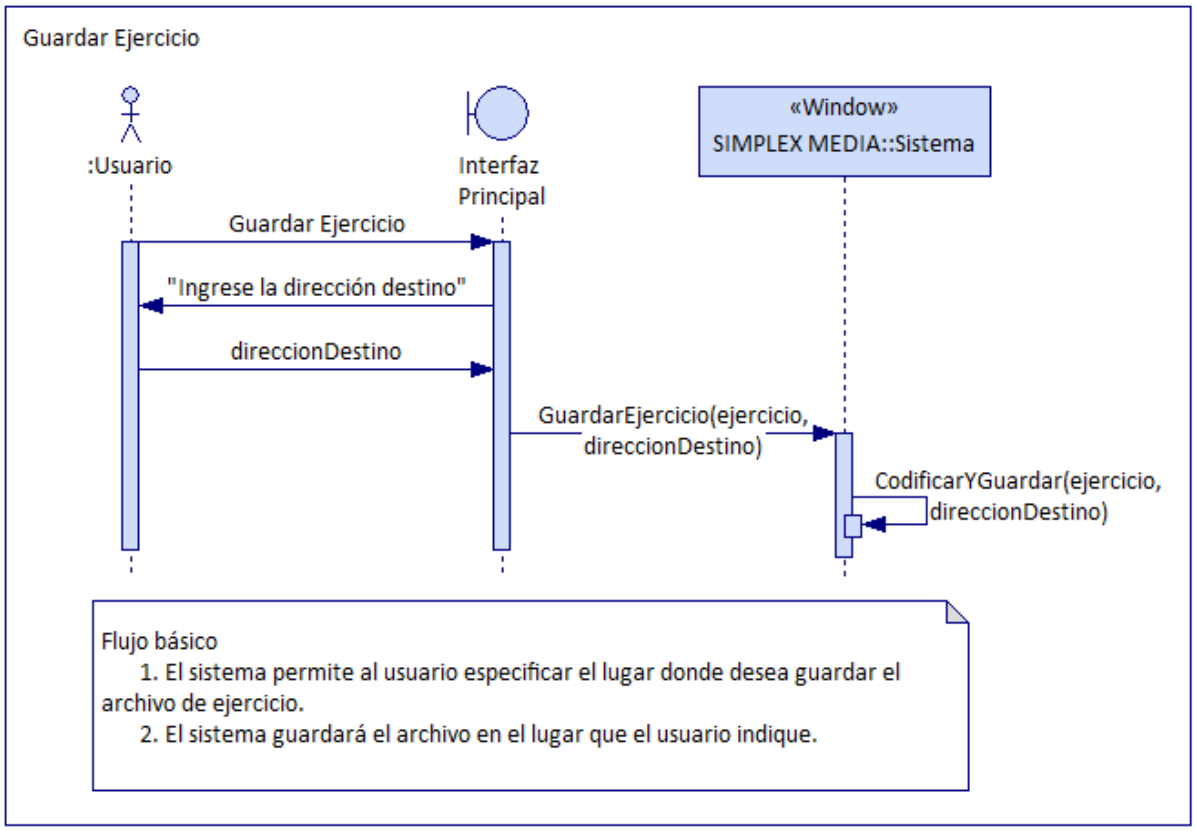
4.5.11 Gestionar ejercicio



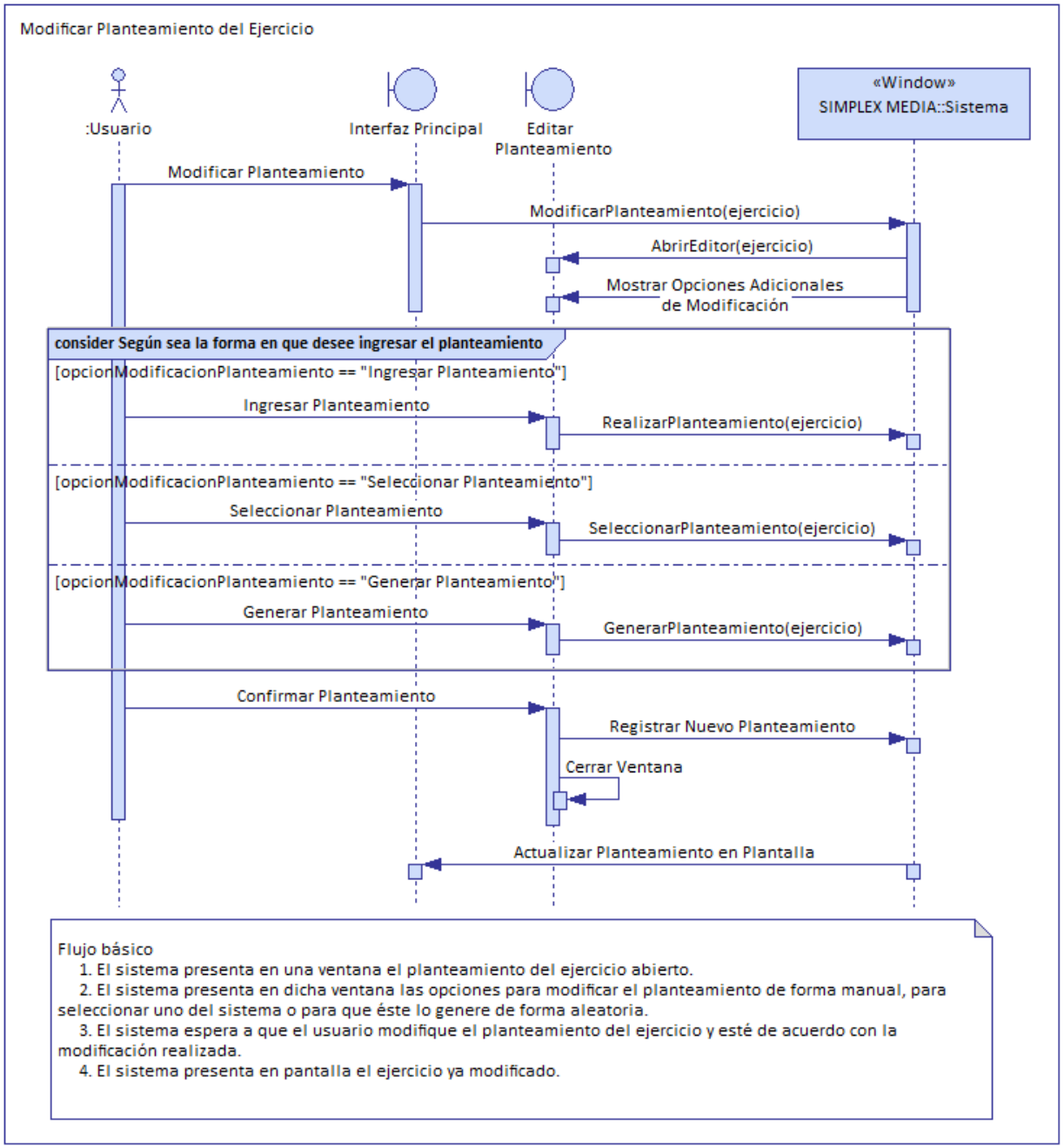
4.5.12 Resolver ejercicio



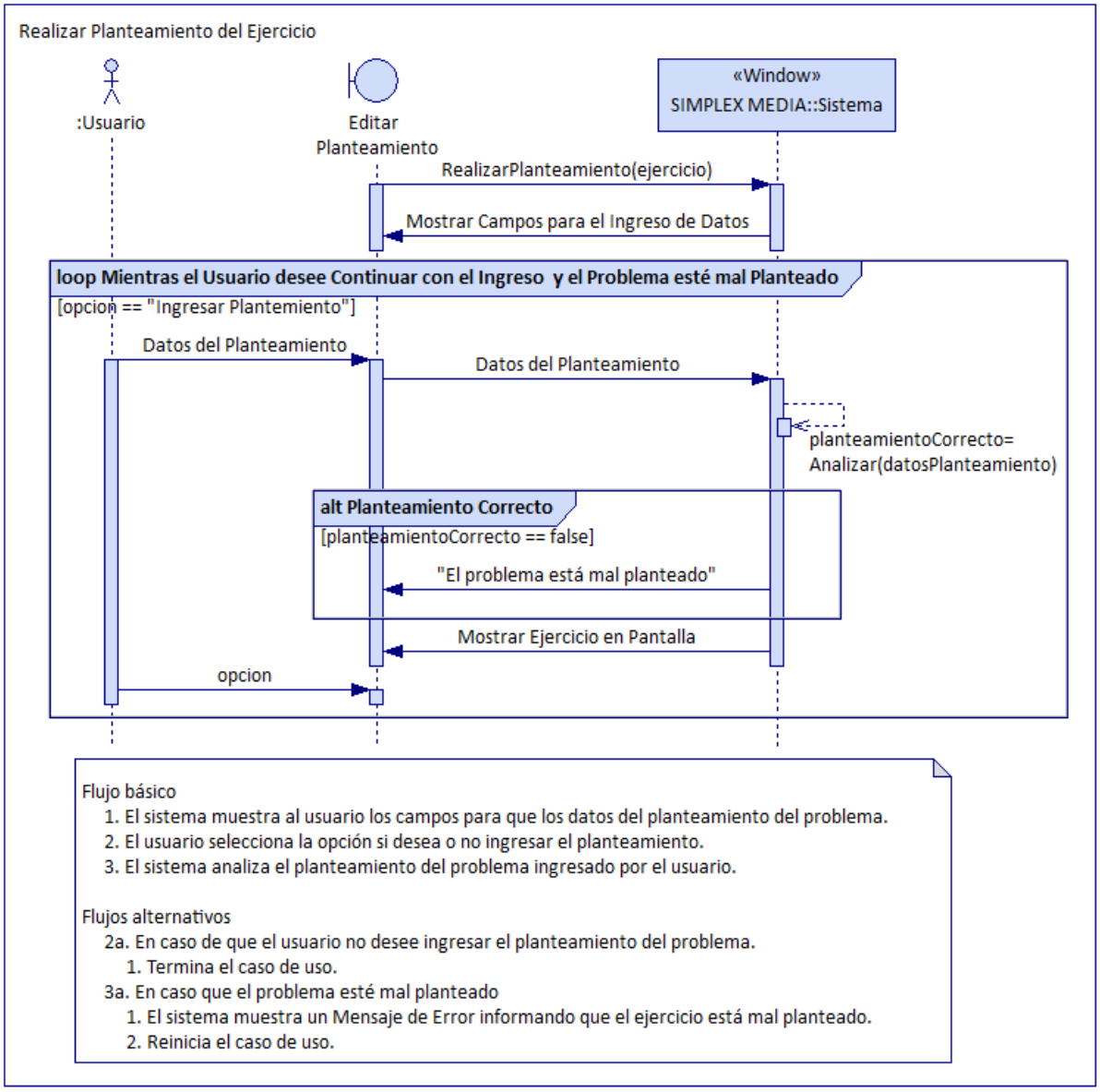
4.5.13 Guardar ejercicio



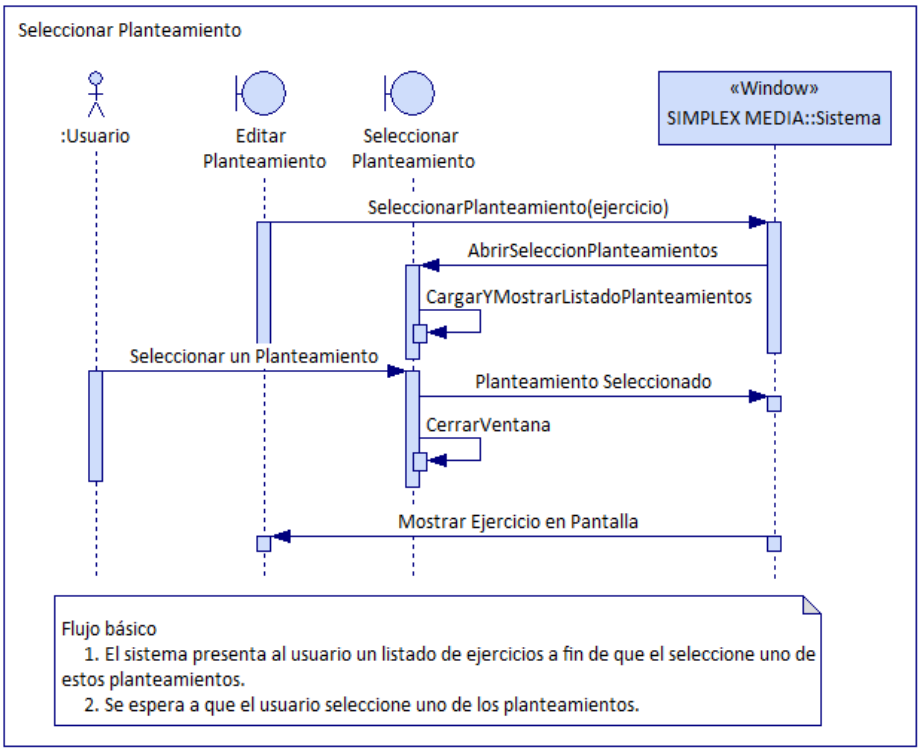
4.5.14 Modificar planteamiento del ejercicio



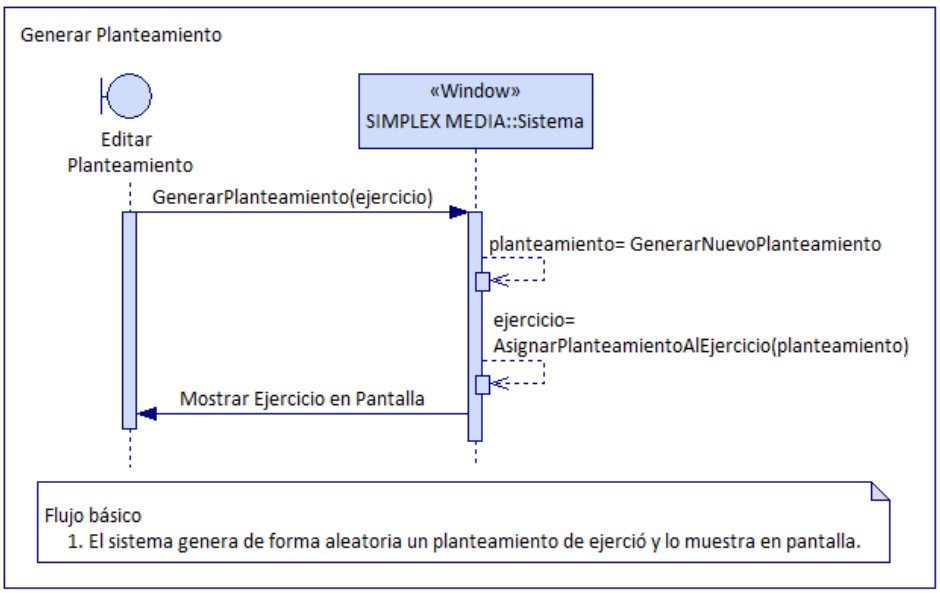
4.5.15 Realizar planteamiento del ejercicio



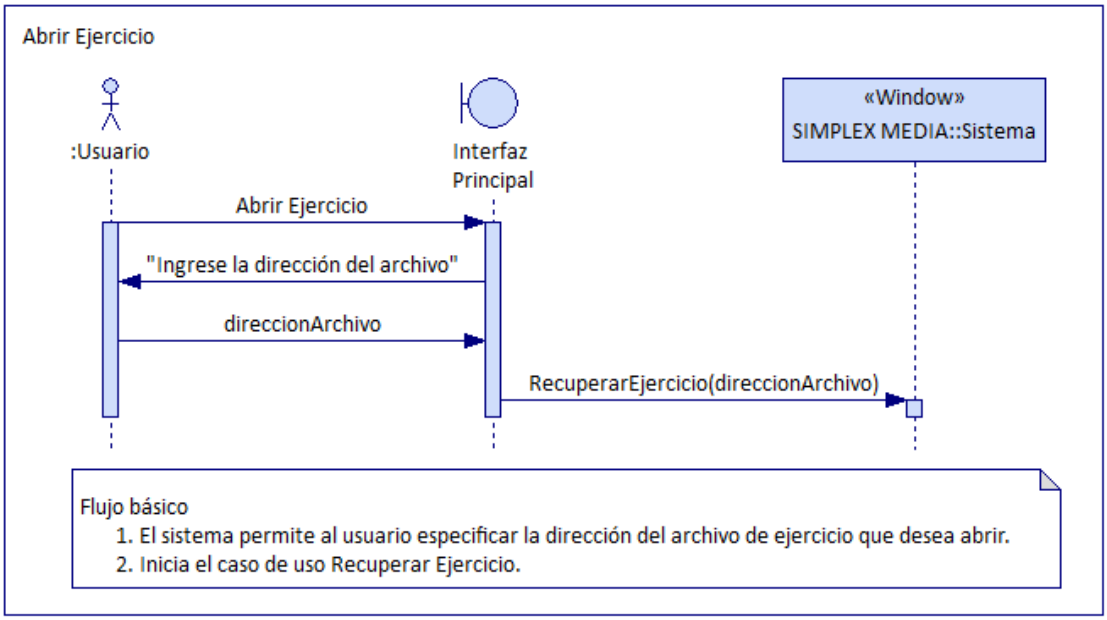
4.5.16 Seleccionar planteamiento



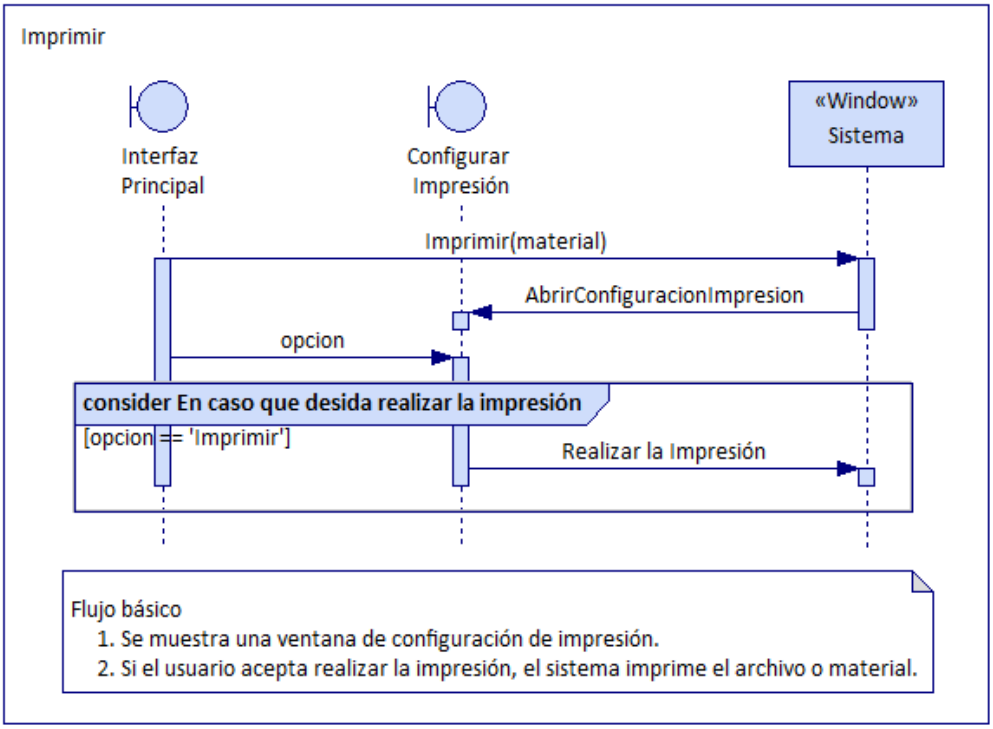
4.5.17 Generar planteamiento



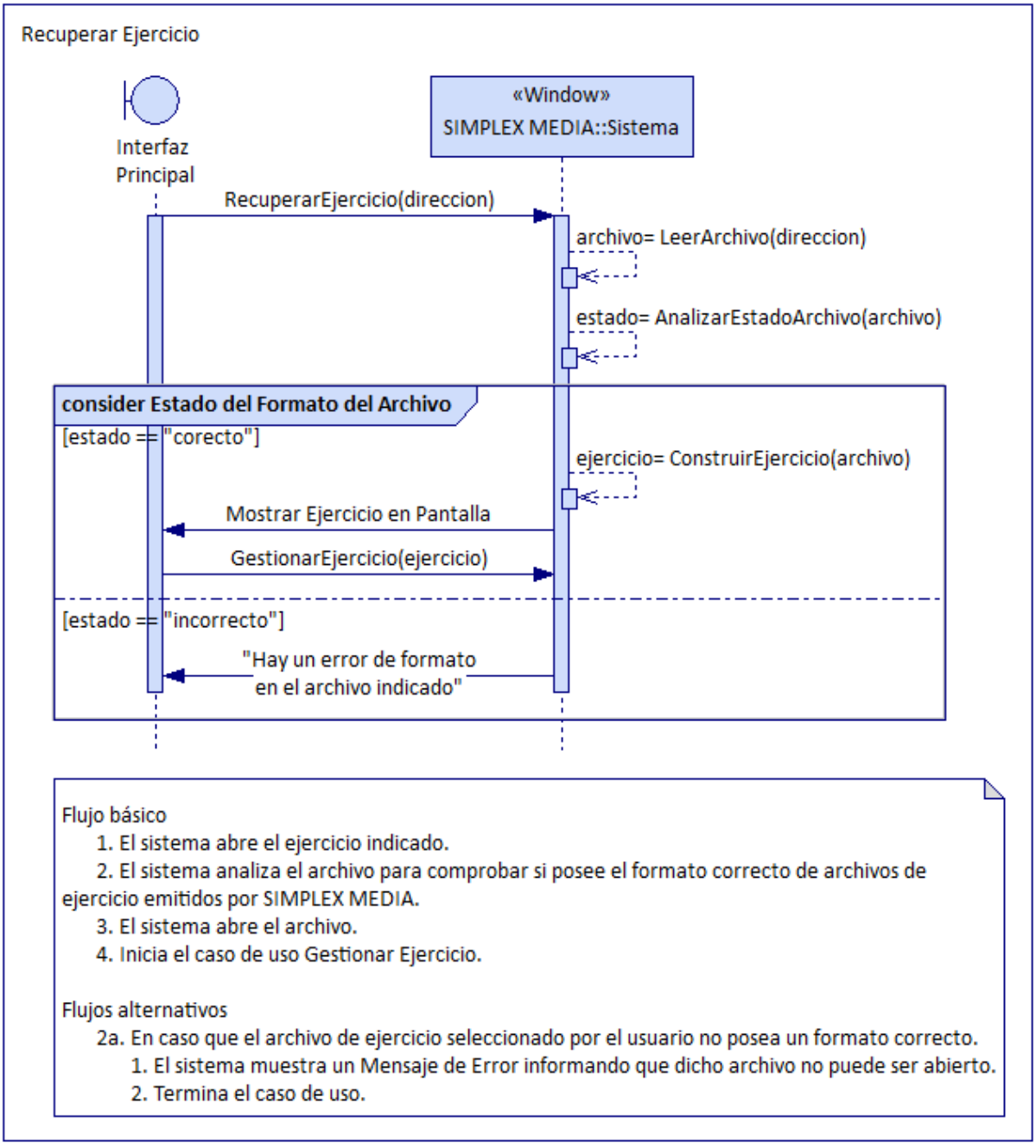
4.5.18 Abrir ejercicio



4.5.19 Imprimir



4.5.20 Recuperar ejercicio



Flujo básico

1. El sistema abre el ejercicio indicado.
2. El sistema analiza el archivo para comprobar si posee el formato correcto de archivos de ejercicio emitidos por SIMPLEX MEDIA.
3. El sistema abre el archivo.
4. Inicia el caso de uso Gestionar Ejercicio.

Flujos alternativos

2a. En caso que el archivo de ejercicio seleccionado por el usuario no posea un formato correcto.

1. El sistema muestra un Mensaje de Error informando que dicho archivo no puede ser abierto.
2. Termina el caso de uso.

4.6 PROTOTIPO DE INTERFAZ DE USUARIO

Los prototipos de interfaz de usuario permiten al analista y diseñador comprender de forma adecuada los casos de uso y los requisitos funcionales del sistema. También permiten especificar con más detalle cual es la comunicación de cada actor con cada parte del sistema. De esta manera se puede hallar mayor familiaridad con los casos de uso y disponer de una base para perfeccionar, en el diseño e implementación, la interfaz de usuario final.

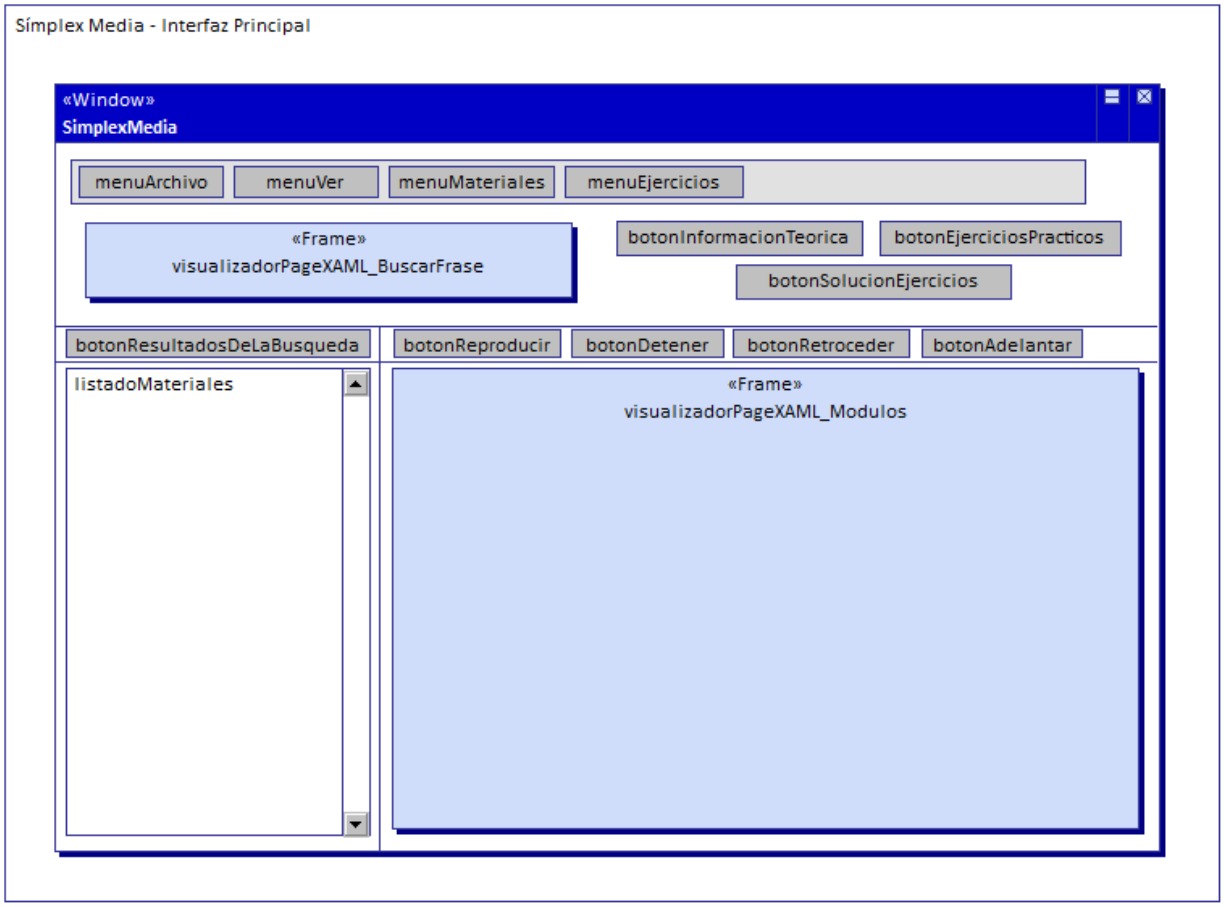
SÍMPLEX MEDIA usará Páginas XAML para mostrar la información de teoría al usuario. Estas páginas son parecidas a las páginas web, pero están hechas con código XAML y admiten código administrado.

En los diagramas de los prototipos de usuario se ha tenido en cuenta este hecho y se ha estereotipado las ventanas con una etiqueta “Page XAML” a aquellas clases de interfaz de usuario que sean de ese tipo. Además, hay ventanas estereotipadas con la palabra “Window” para especificar que se tratan de ventanas tipo Window de WPF y, finalmente, el estereotipo “User Control XAML” indica que se trata de un control de usuario definido bajo la tecnología de WPF.

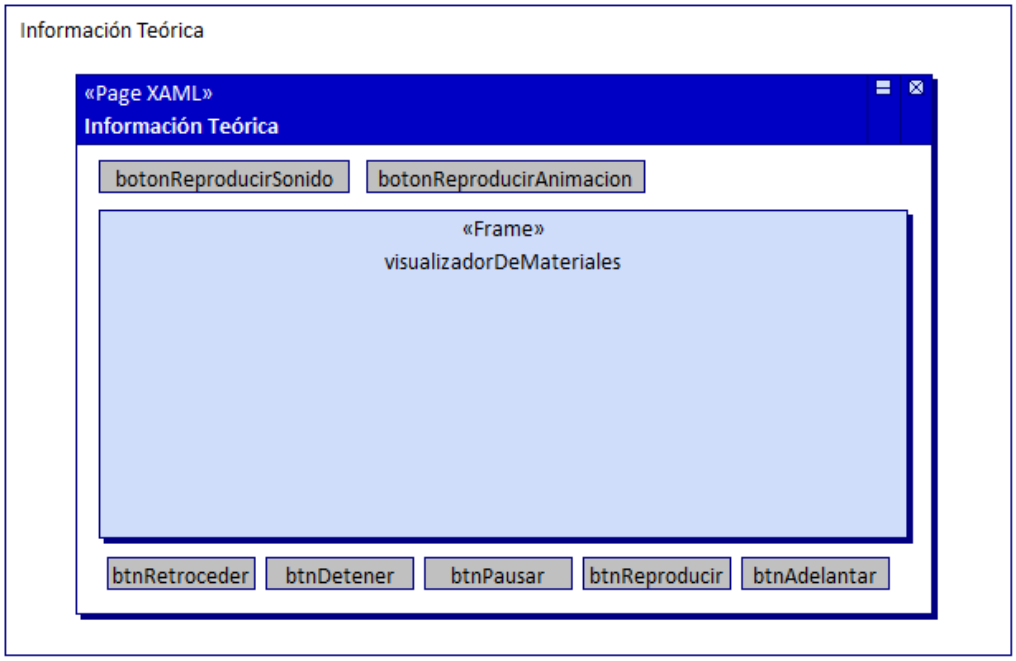
También se ha organizado en paquetes a aquellas que comparten características del sistema e interactúan mucho entre sí y se han creado diagramas que ilustran las relaciones entra las ventanas y páginas que componen la interfaz gráfica de usuario del programa.

4.6.1 Descripción de interfaces de usuario

4.6.1.1 Interfaz principal de SÍMPLEX MEDIA

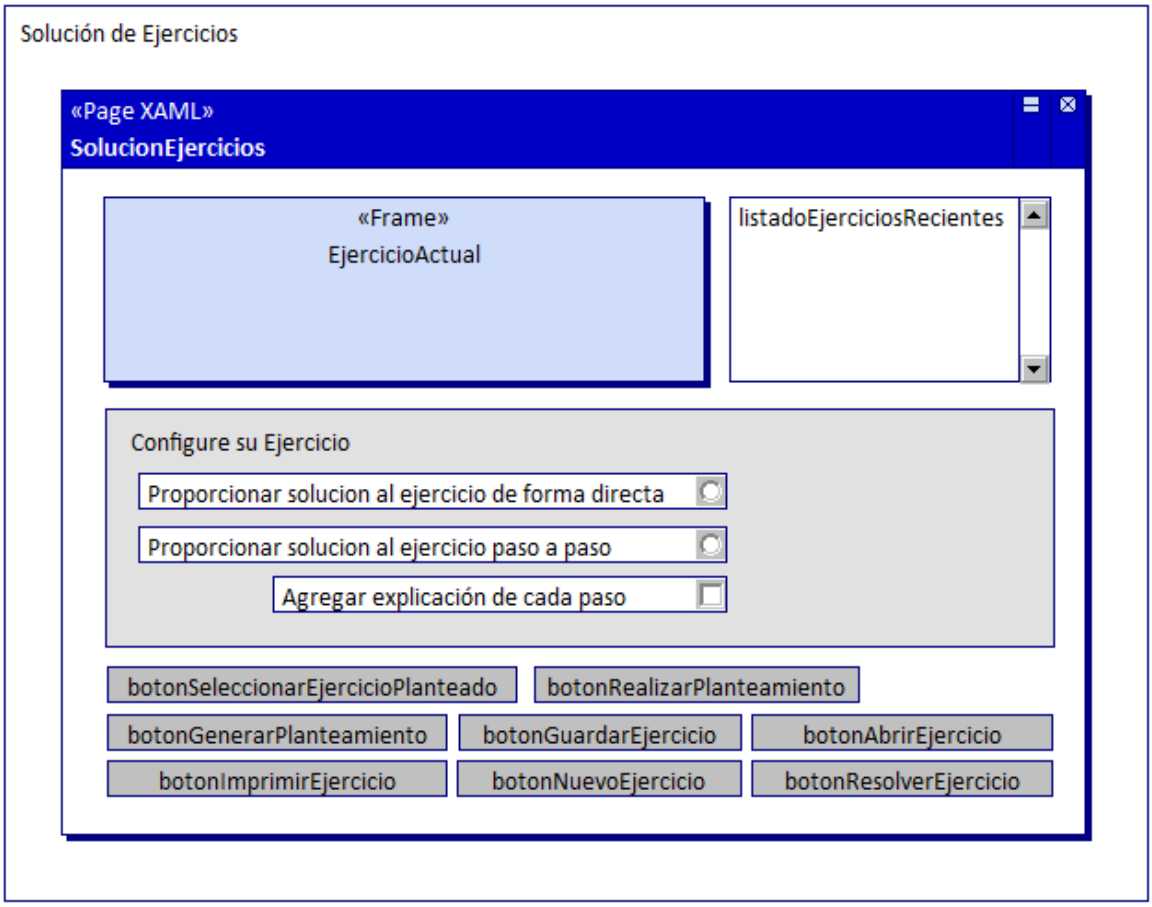


4.6.1.2 Información teórica

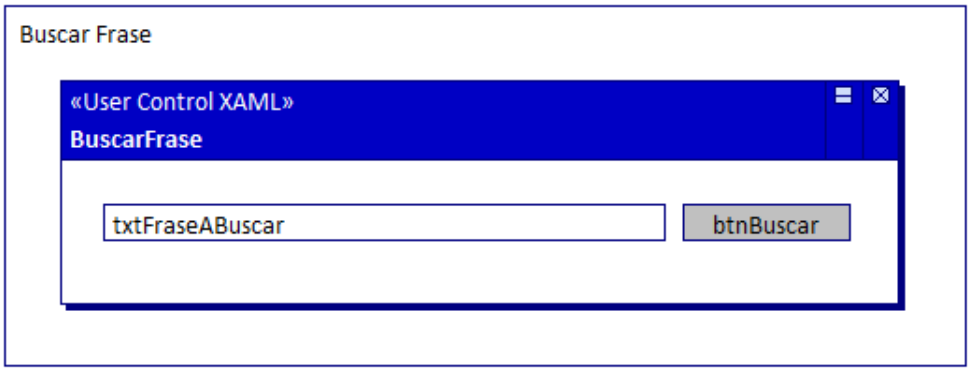


4.6.1.3 Ejercicios prácticos

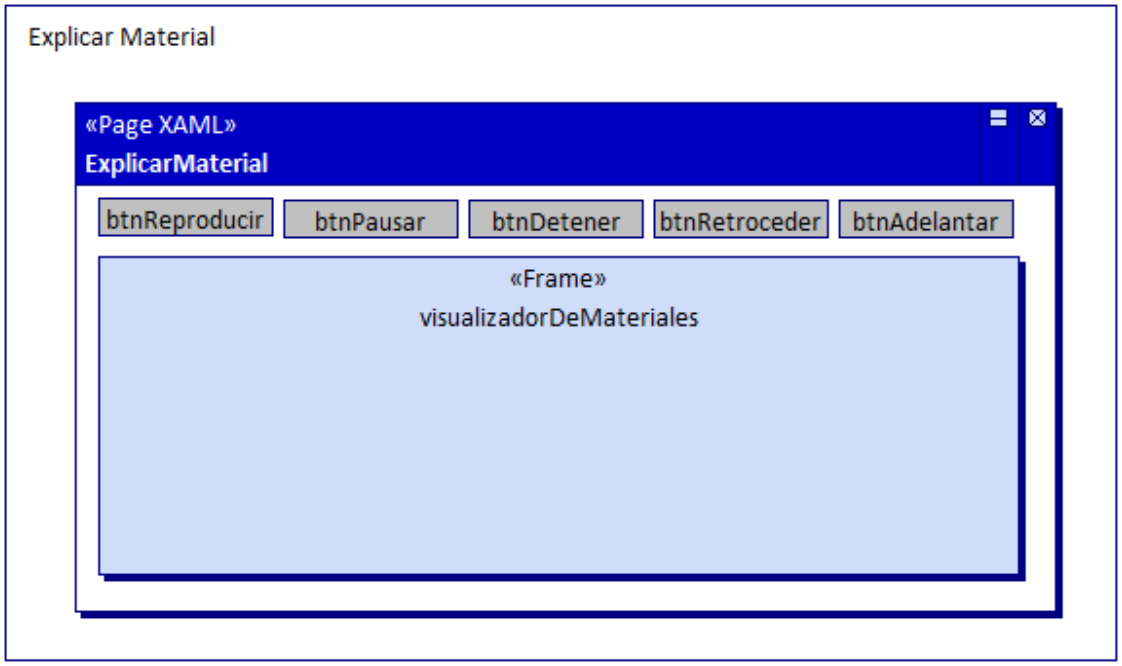
4.6.1.4 Solución de ejercicios



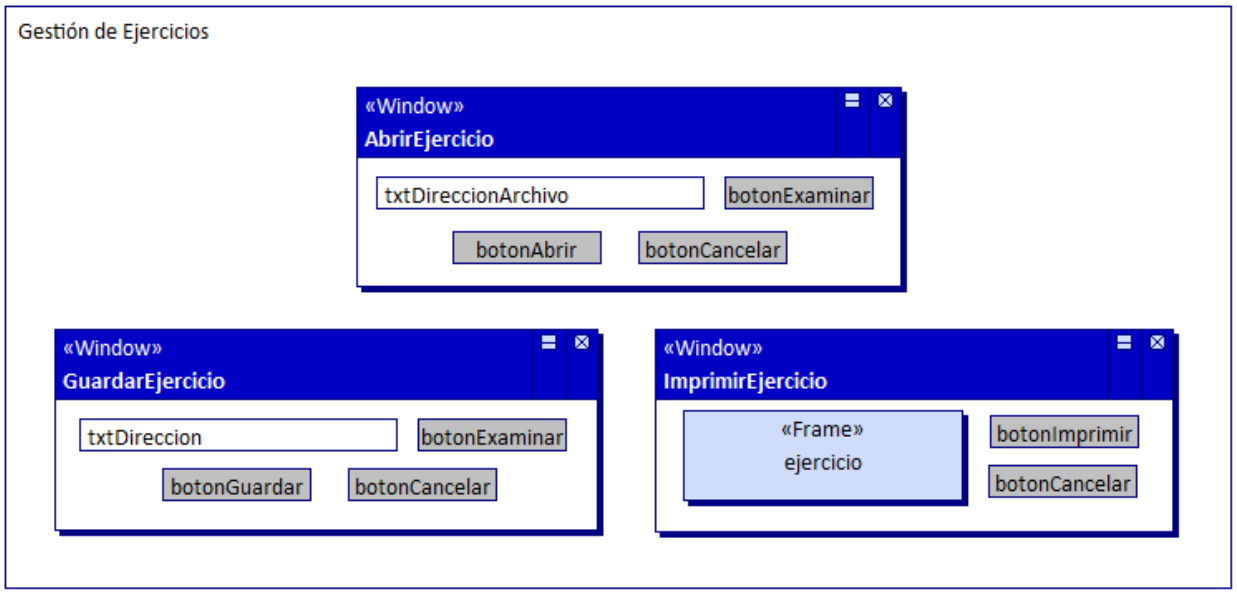
4.6.1.5 Buscar frases



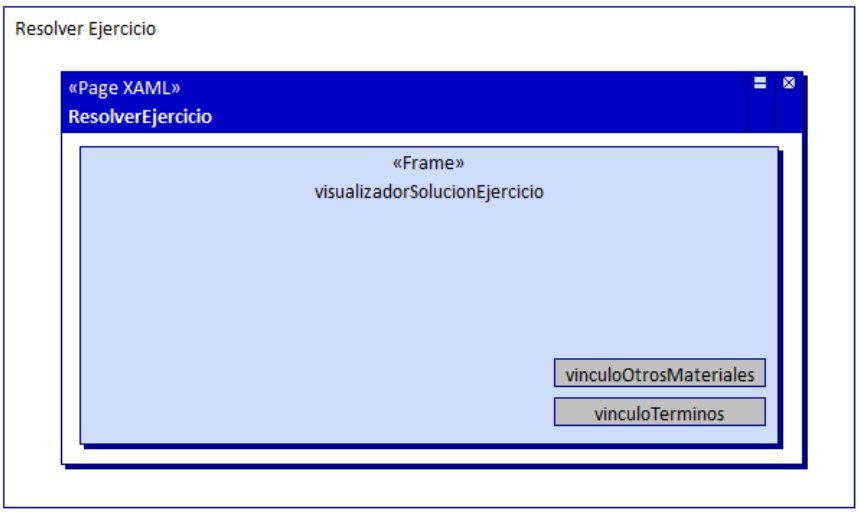
4.6.1.6 Explicar materiales



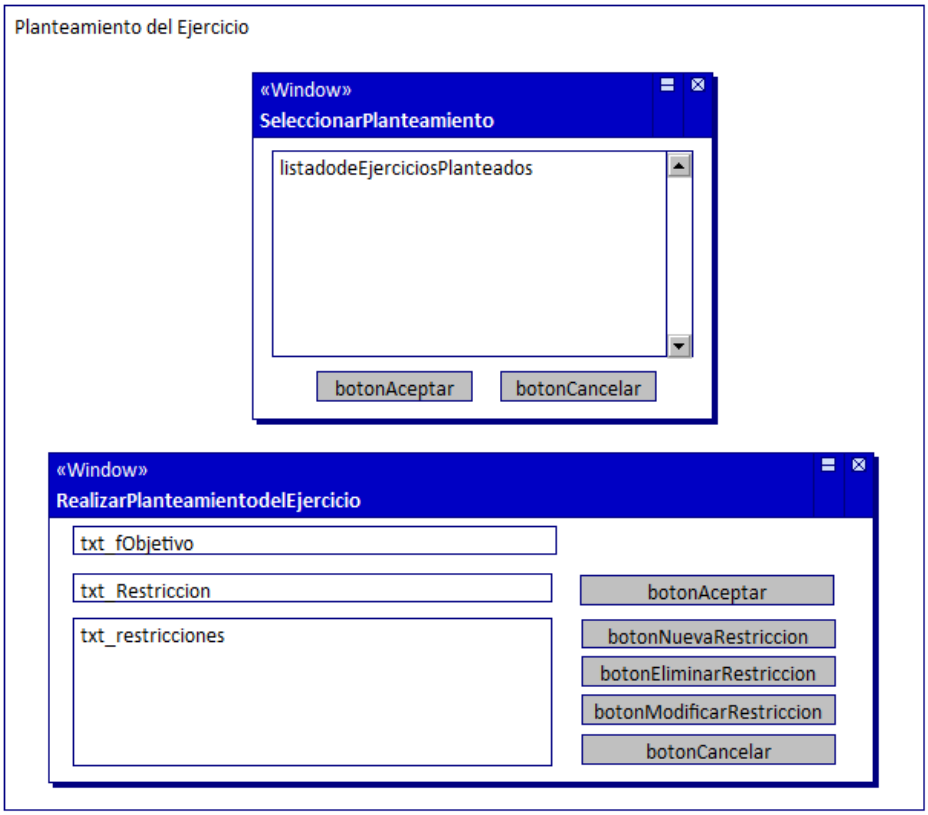
4.6.1.7 Gestión de ejercicios



4.6.1.8 Resolver ejercicio

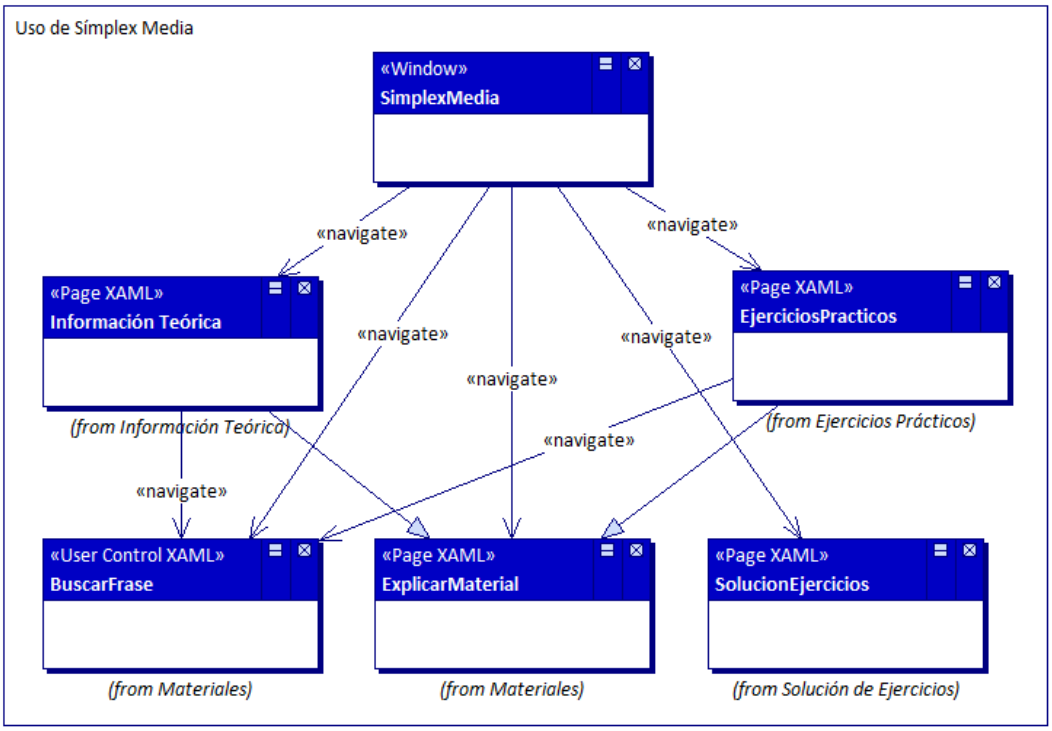


4.6.1.9 Planteamiento del ejercicios

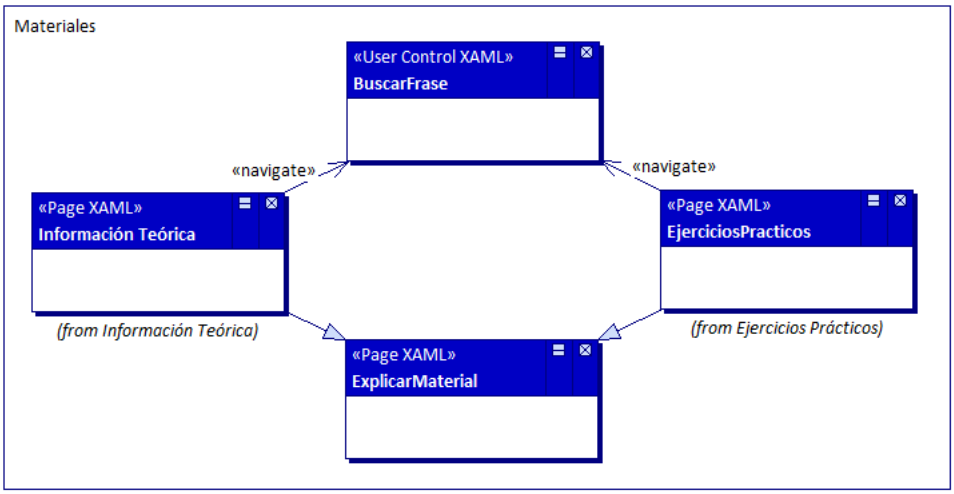


4.6.2 Relaciones de las interfaces de usuario

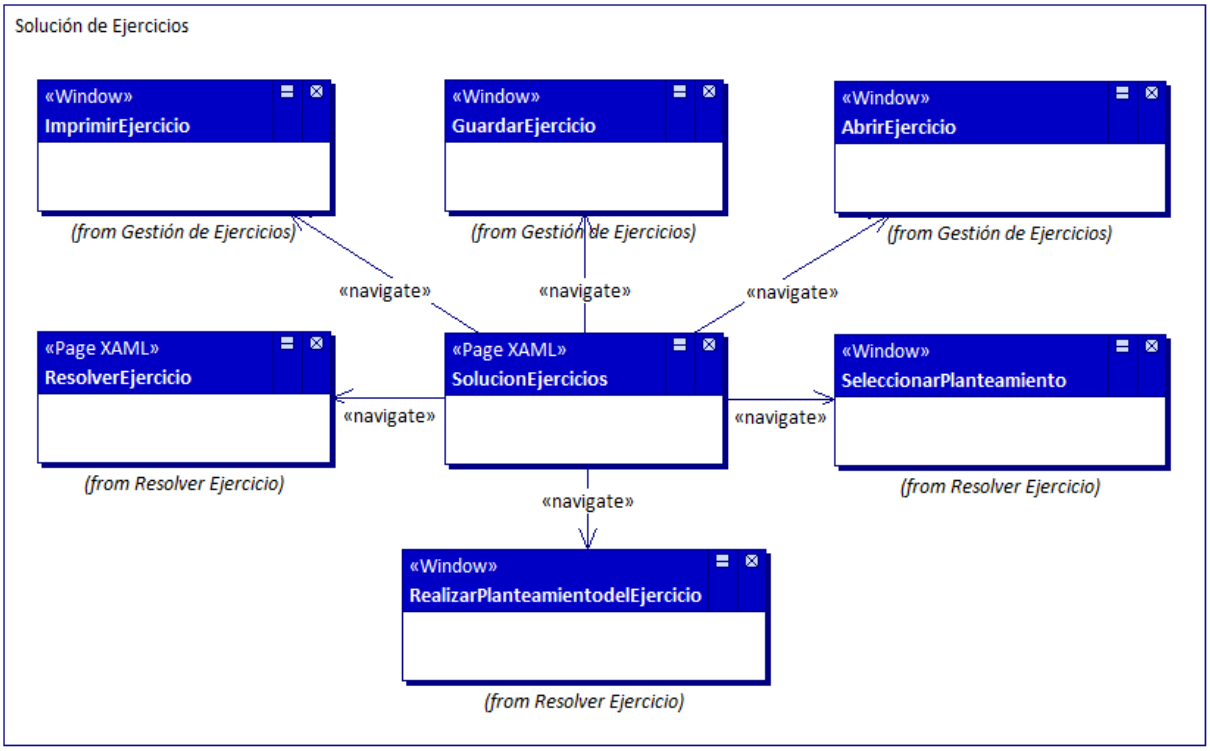
4.6.2.1 Diagrama: Uso de SÍMPLEX MEDIA



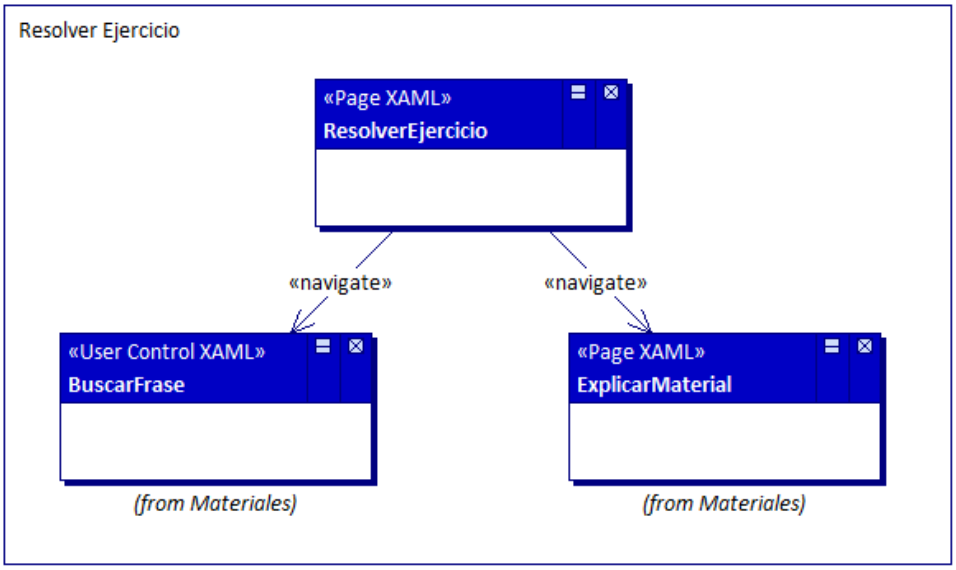
4.6.2.2 Diagrama: Materiales



4.6.2.3 Diagrama: Solución de ejercicios

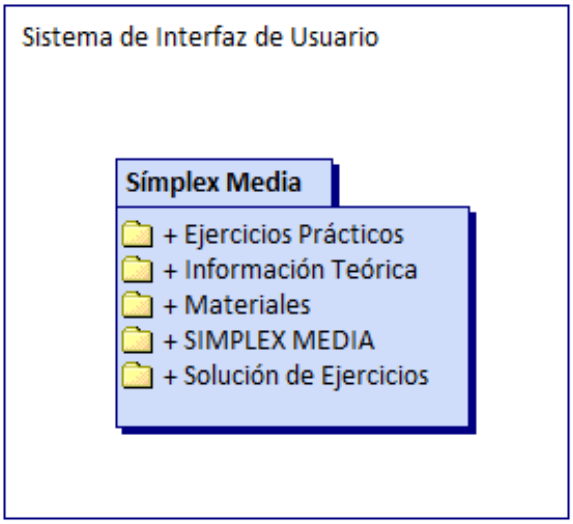


4.6.2.4 Diagrama: Resolver ejercicios

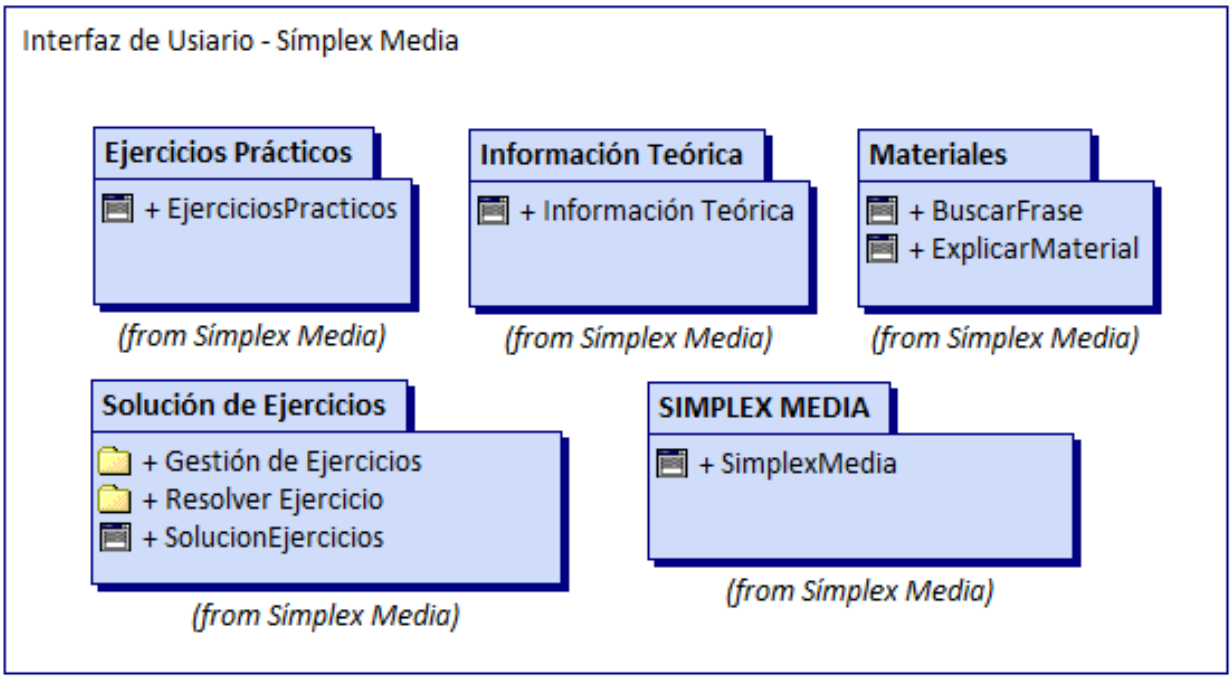


4.6.3 Paquetes de interfaz de usuario

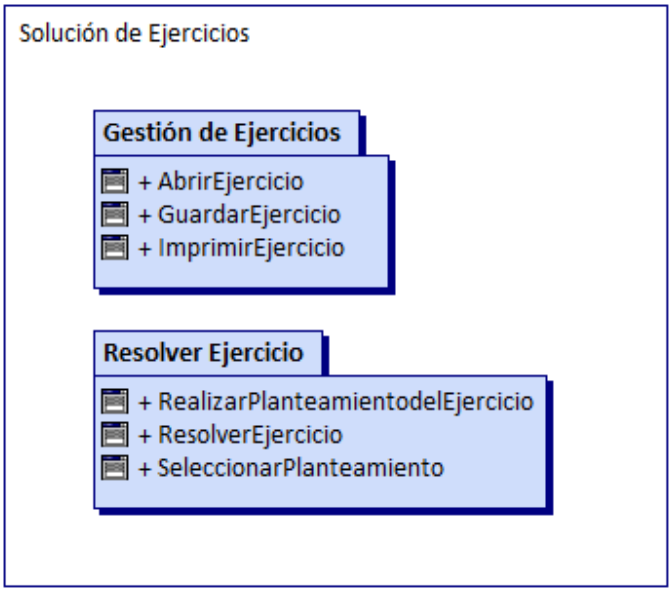
4.6.3.1 Paquete: Sistema de interfaz de usuario



4.6.3.2 Paquete: SÍMPLEX MEDIA



4.6.3.3 Paquete: Solución de ejercicios



5 MODELO DE ANALISIS

El contenido de este modelo se aproxima más a la construcción real de la aplicación. Se identifican muchas de las clases y sus relaciones a partir de los anteriores casos de uso. También, mediante las realizaciones de casos de uso, se logra describir los casos de uso en términos de objetos programables. Estas últimas servirán para determinar con mayor precisión las responsabilidades y atributos de las clases en la etapa del diseño.

5.1 CLASES DEL ANÁLISIS

Las clases del análisis no son las clases definitivas del proyecto, pero aportan un adecuado entendimiento acerca de la comunicación entre los objetos que interactúan en la realización de un caso de uso. Además, permiten al diseñador echar un vistazo en lo que sucede en la parte interna del sistema cuando se está llevando a cabo un caso de uso. Estas clases serán utilizadas para realizar los posteriores Diagramas de Colaboración.

Las clases del análisis han sido estereotipadas utilizando los tres estándares que provee UML para el análisis. Cada estereotipo sirve para que, en el momento de diseñar y desarrollar, se pueda distinguir el ámbito de cada clase.

Los tres estereotipos son: Clase de Interfaz, Clase de Control y Clase de Entidad.

- Clase de Interfaz:

Ilustración 5-1 Representación gráfica de una clase tipo Interfaz

Este tipo de clases son las que representan la interacción directa del usuario con el sistema. Puede tratarse de interfaces gráficas de usuario (GUI) que presenten o soliciten información al usuario o a otros sistemas.

- Clase de Control:

Ilustración 5-2 Representación gráfica de una clase tipo Control

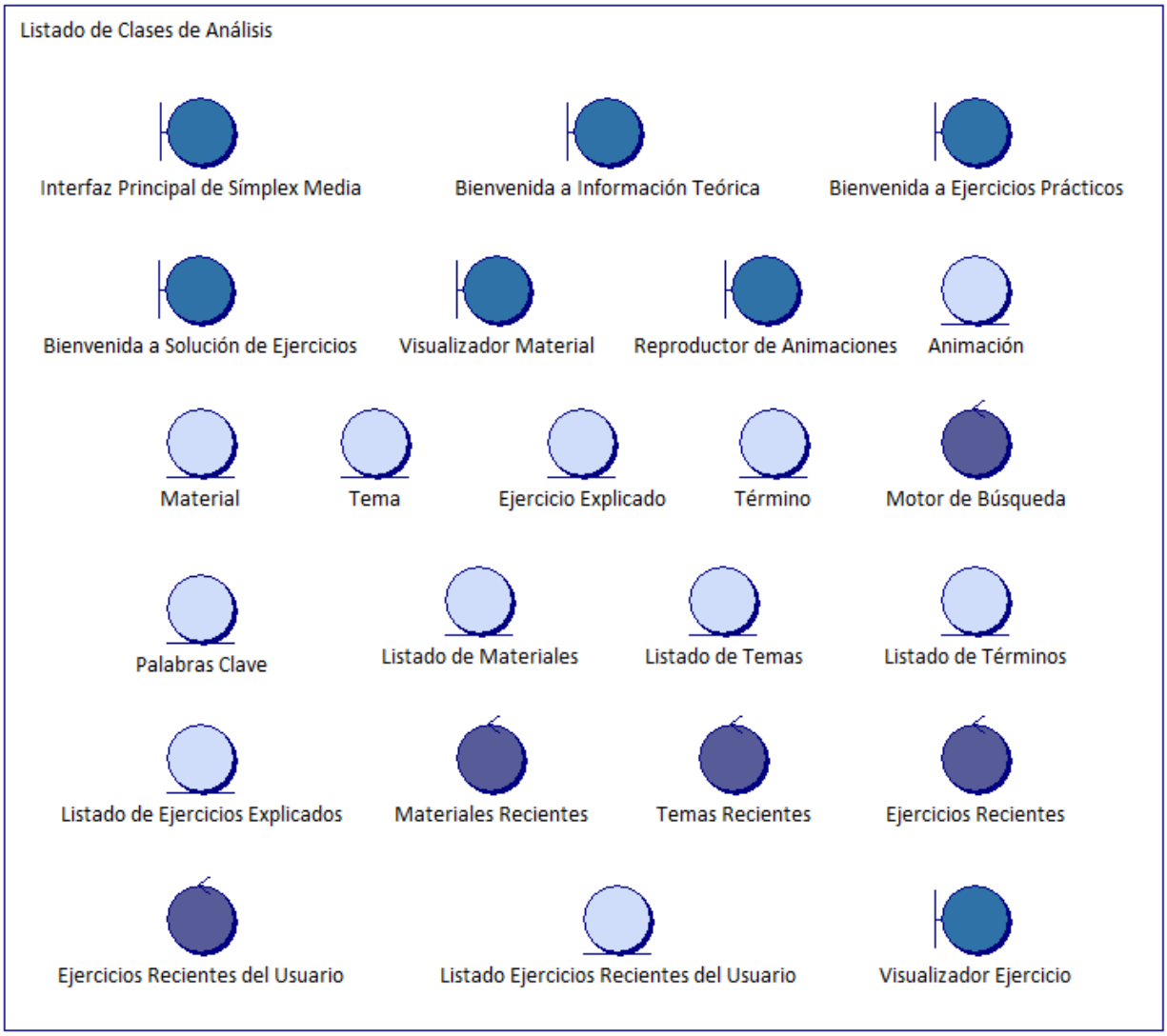
Utilizando las Clases de Control se modelan aquellas encargadas de ejecutar procesos internos como transacciones o control de otros objetos. Otros usos, que son los que se aplican a muchas de las clases de este proyecto, es el de cálculos complejos y aspectos dinámicos del sistema.

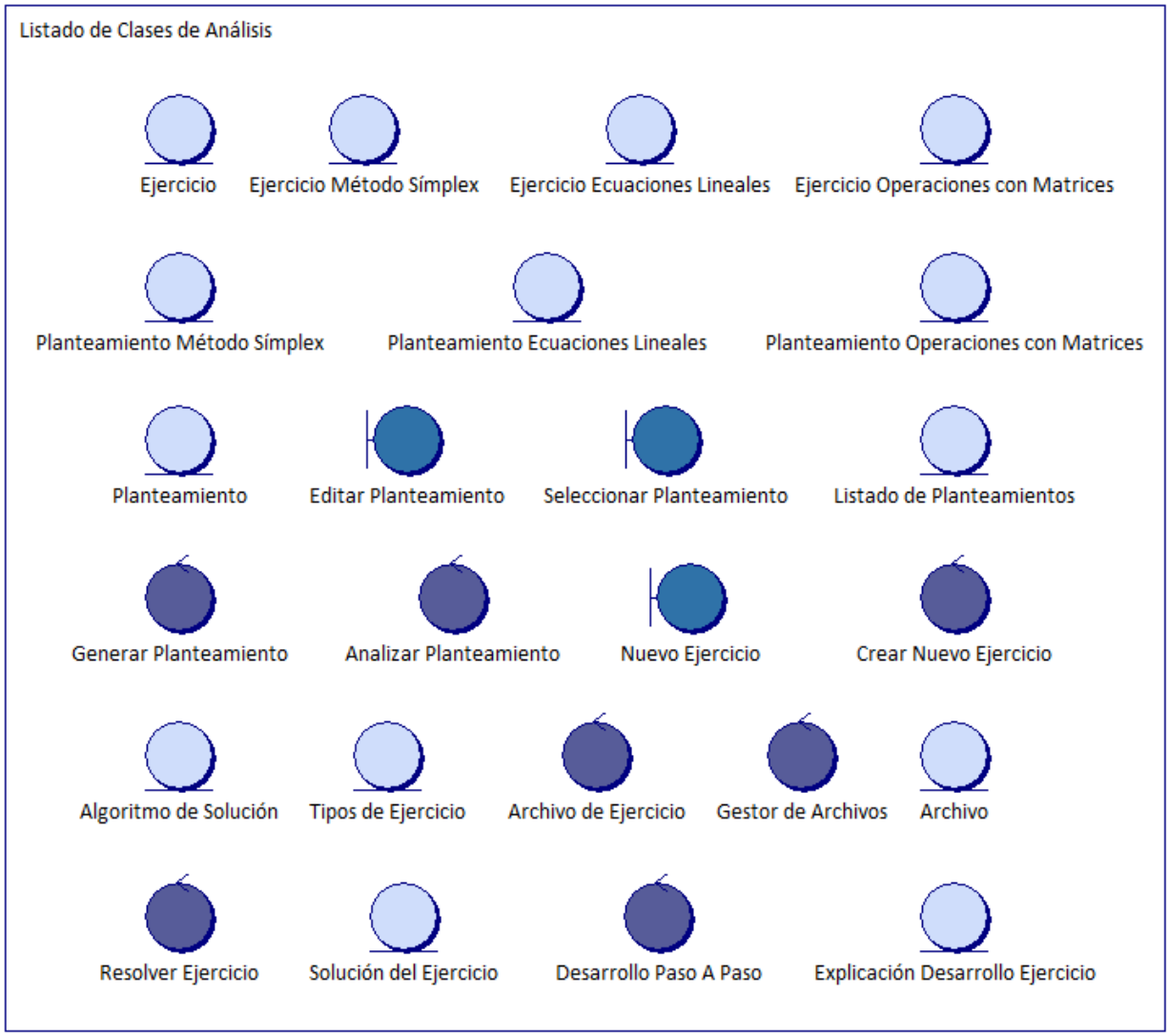
- Clase de Entidad:

Ilustración 5-3 Representación gráfica de una clase tipo Entidad

Para los casos de las clases que representan conjunto de datos, información persistente del sistema o entidades del mundo real, se utilizan las Clases de Entidad. Un ejemplo de este tipo de clases puede ser un listado de datos o un elemento del sistema, como un archivo o un ejercicio.

5.1.1 Listado de clases del análisis



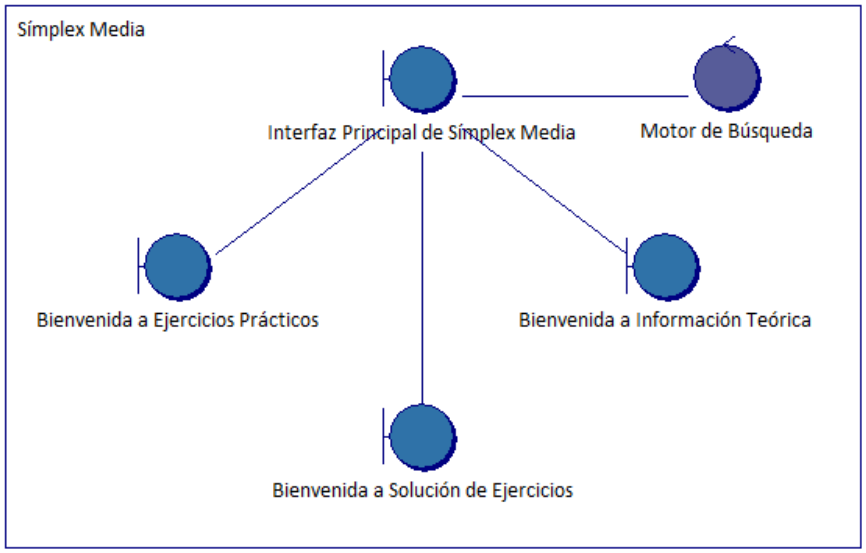


Para observar cómo se relacionan las clases se han ordenado y agrupado de acuerdo a su aparición en los anteriores casos de uso.

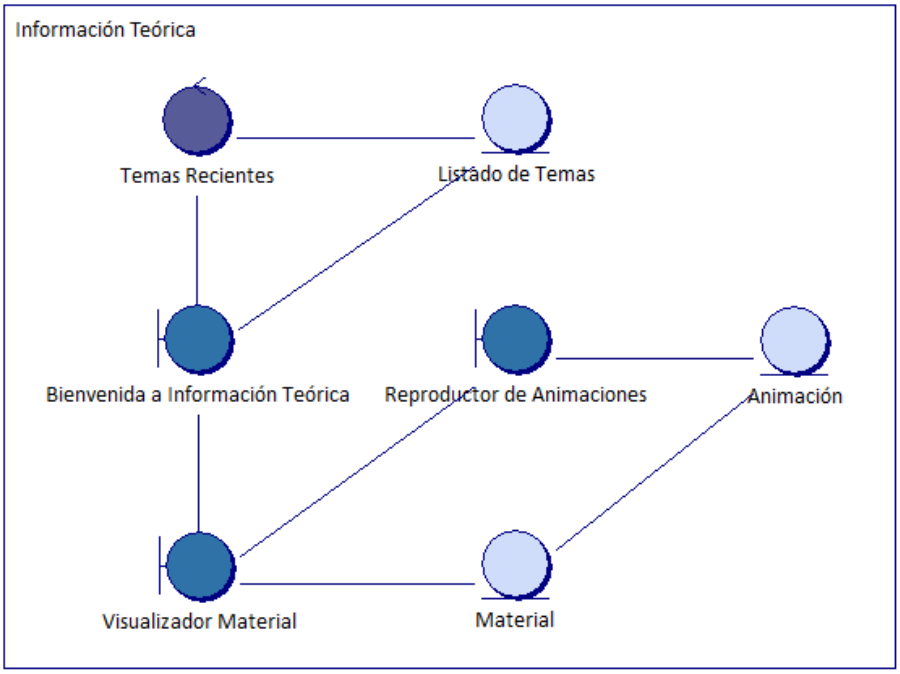
De casa agrupación se ha hecho un diagrama de clases que permite visualizar cada una de las secciones del sistema.

5.1.2 Diagramas de clases

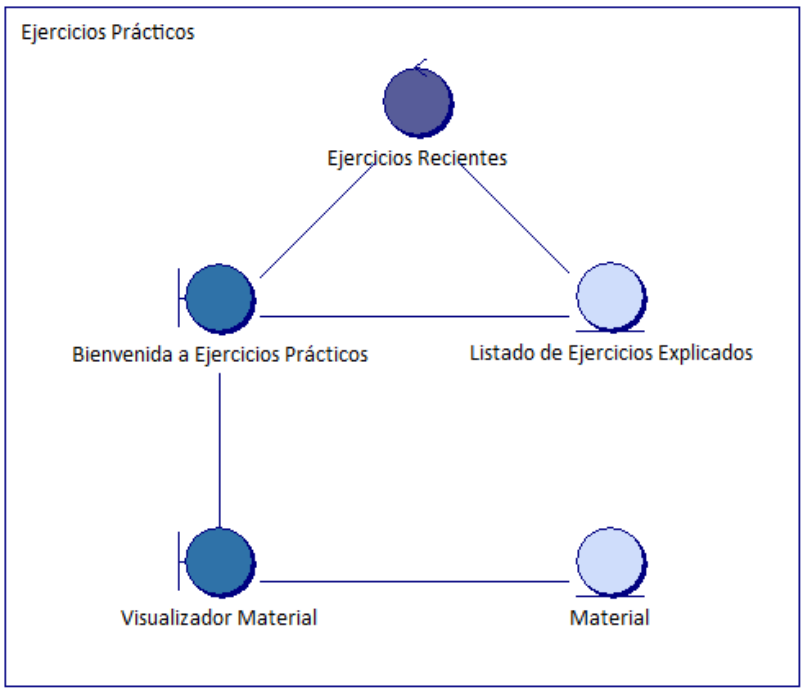
5.1.2.1 Diagrama: Uso de SÍMPLEX MEDIA



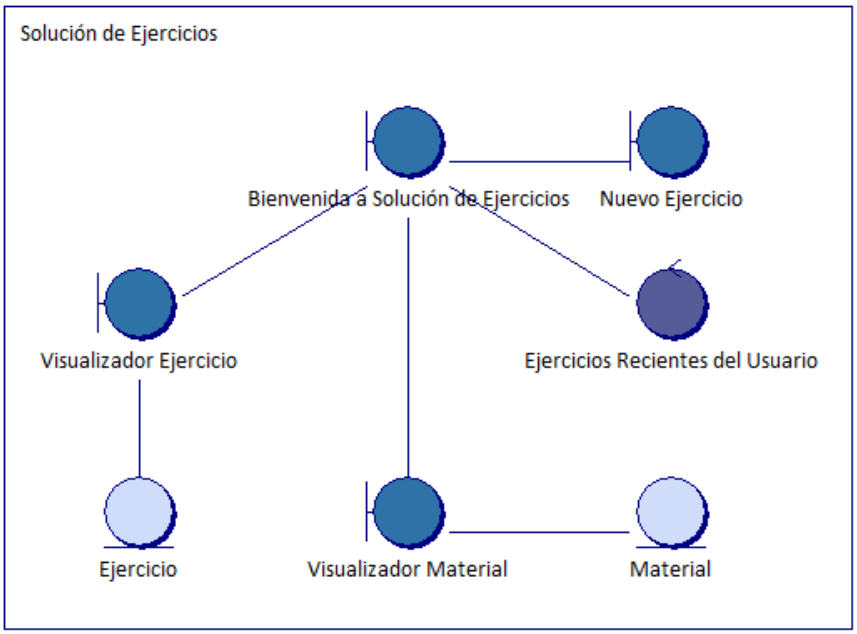
5.1.2.2 Diagrama: Módulo información teórica



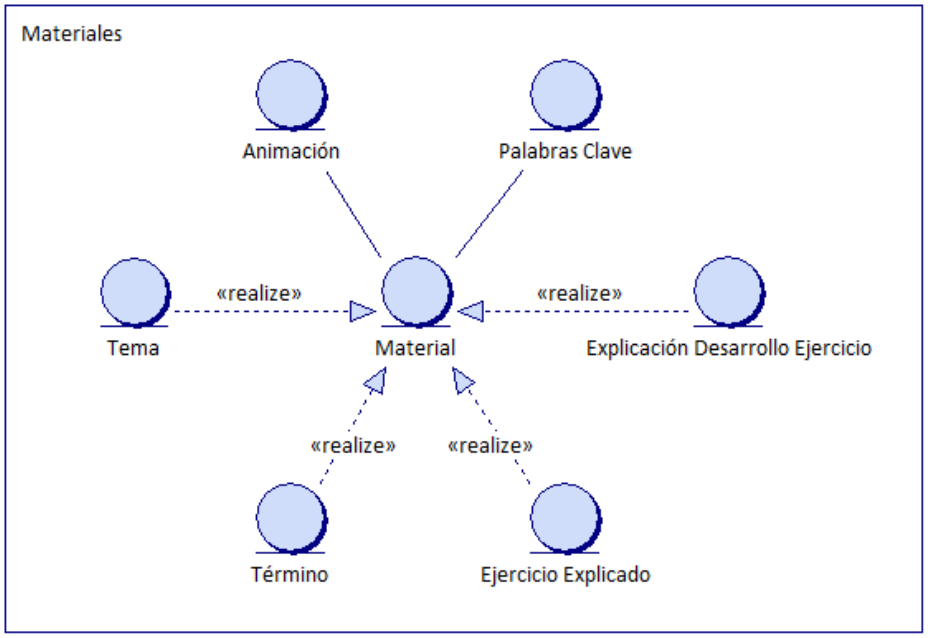
5.1.2.3 Diagrama: Módulo ejercicios prácticos



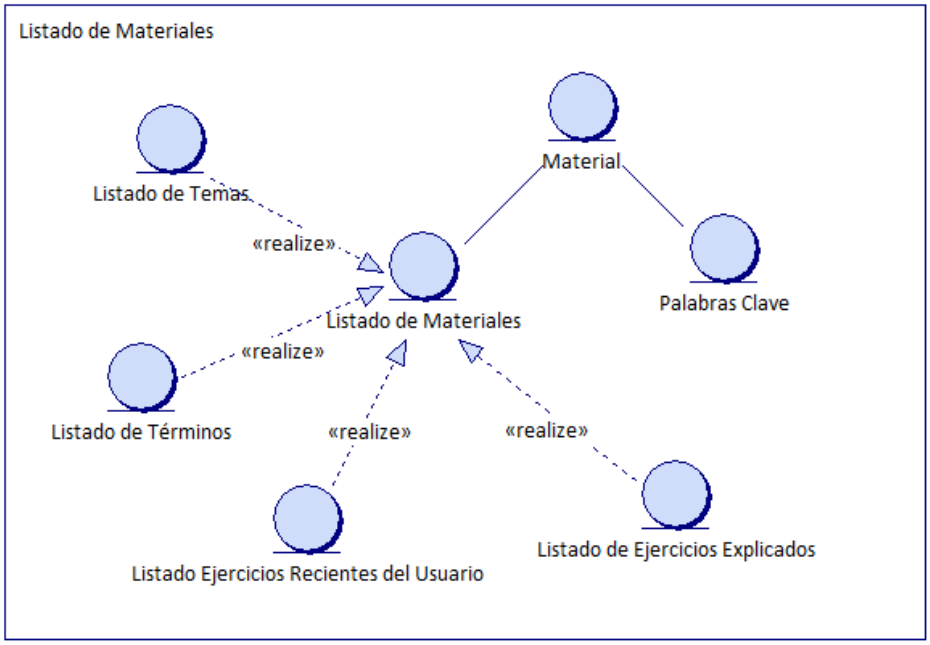
5.1.2.4 Diagrama: Solución de ejercicios



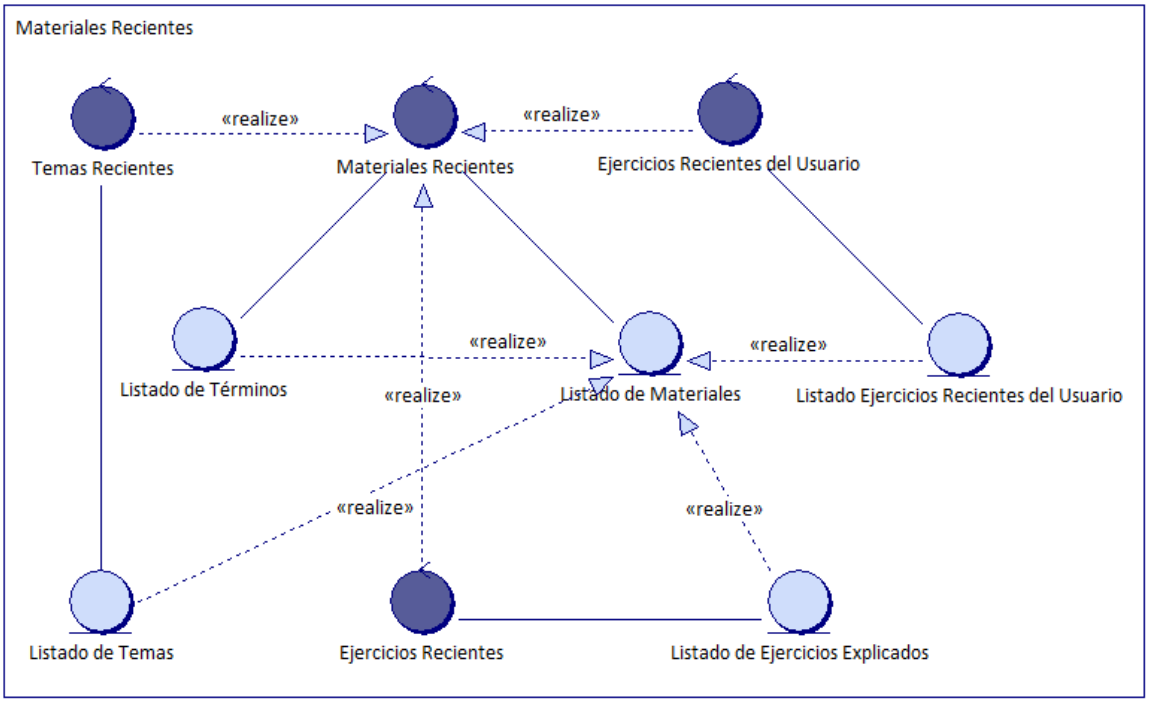
5.1.2.5 Diagrama: Materiales



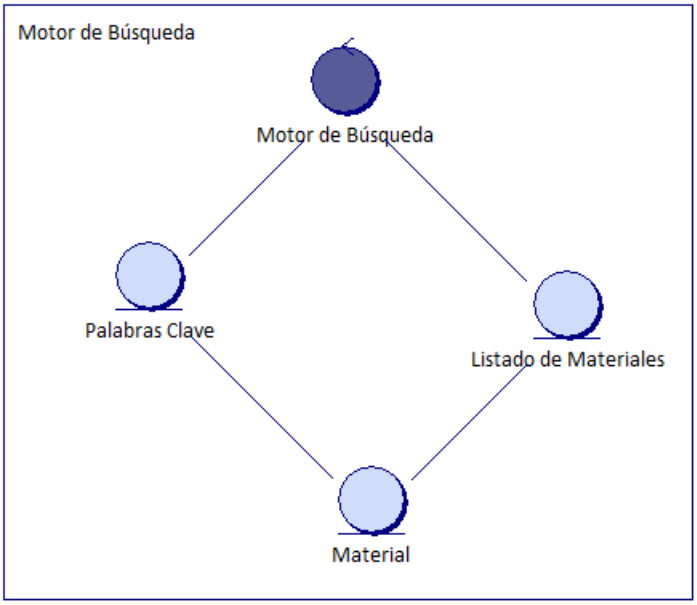
5.1.2.6 Diagrama: Listado de materiales



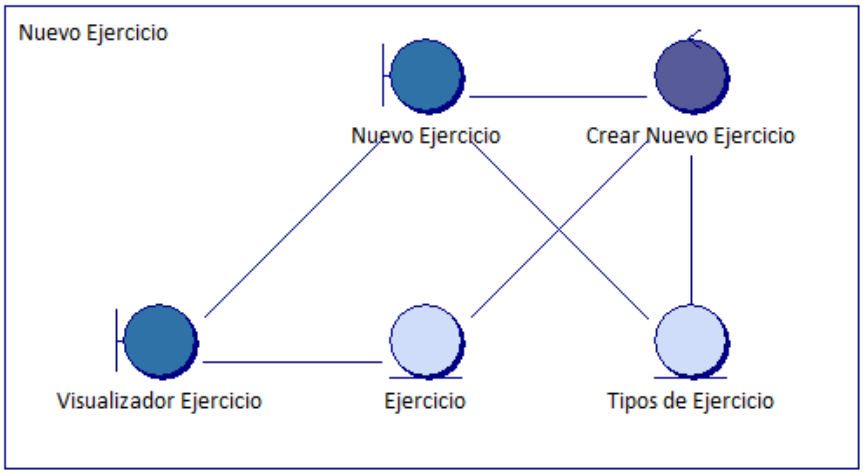
5.1.2.7 Diagrama: Materiales recientes



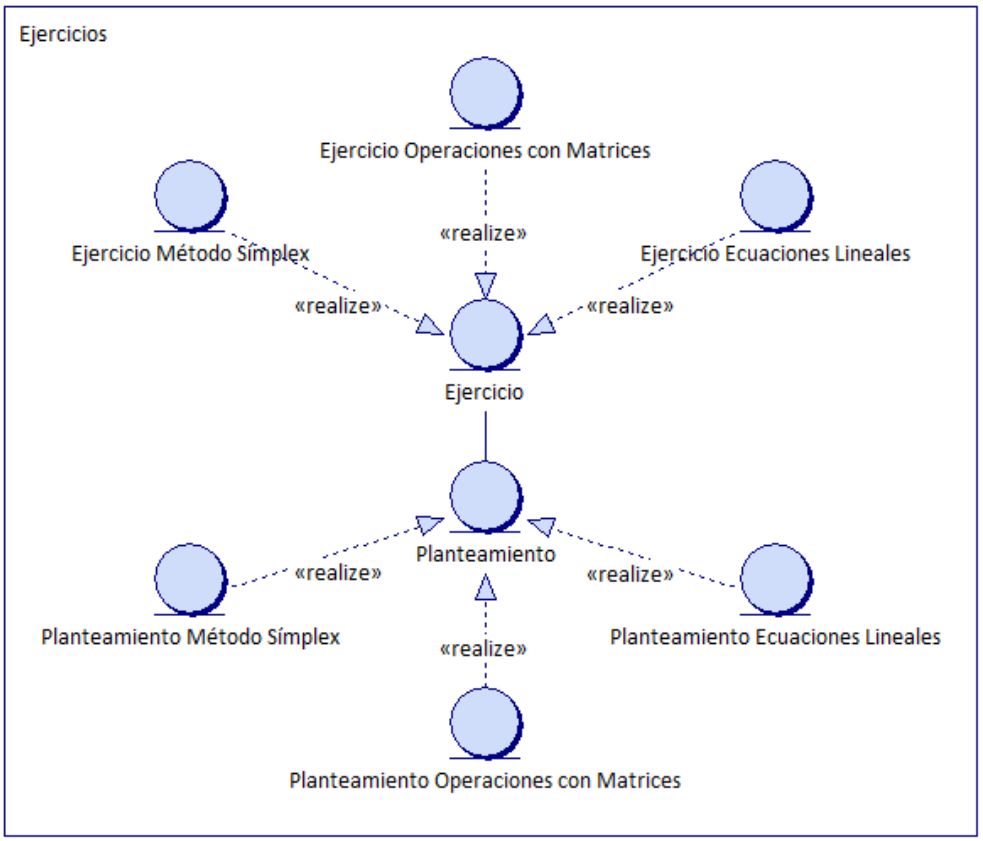
5.1.2.8 Diagrama: Motor de búsqueda



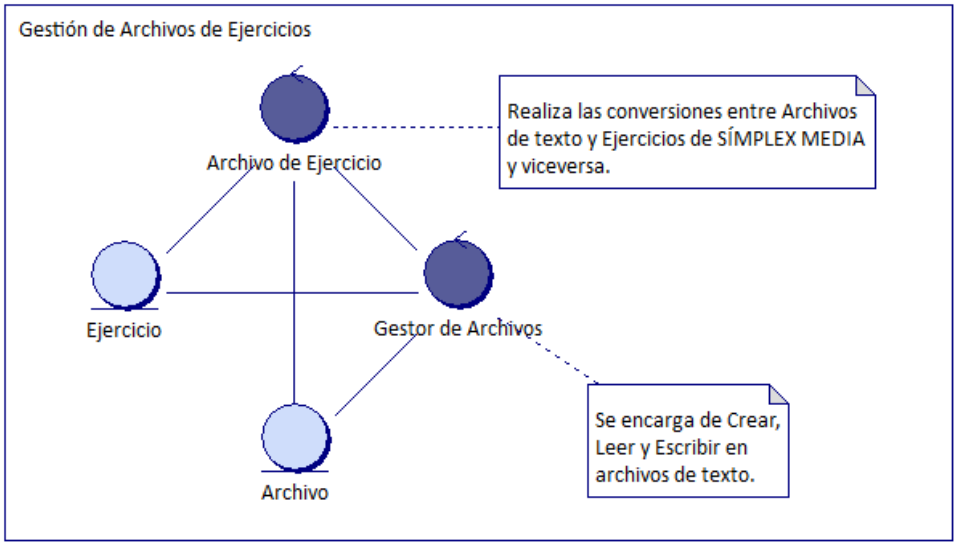
5.1.2.9 Diagrama: Nuevo ejercicio



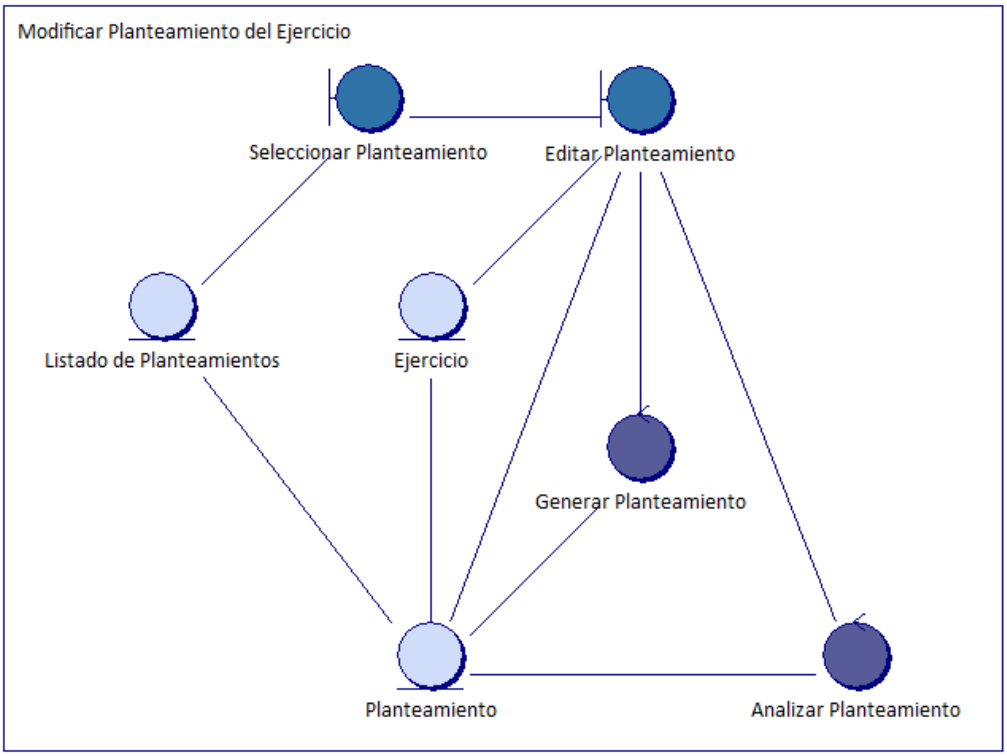
5.1.2.10 Diagrama: Ejercicios



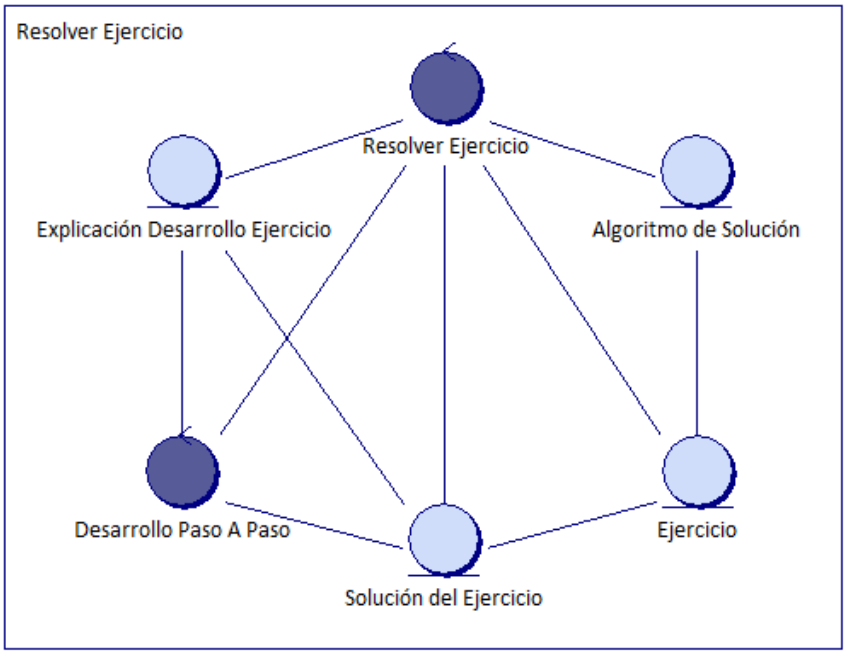
5.1.2.11 Diagrama: Gestión de archivos de ejercicios



5.1.2.12 Diagrama: Modificar planteamiento del ejercicio



5.1.2.13 Diagrama: Resolver ejercicio



5.2 REALIZACIONES DE CASOS DE USO - ANÁLISIS

Una *realización de caso de uso – análisis* es una Colaboración del modelo de análisis que se utiliza para describir el desarrollo de un caso de uso en términos de clases del análisis y mostrar cómo interactúan en él los objetos del análisis.

Una realización de caso de uso – análisis llega a ser un camino concreto de ejecución dentro un caso de uso. Por lo tanto, cada caso de uso estará referenciado por una o más colaboraciones dependiendo de los diferentes flujos que puedan seguirse en él.

Aunque no es necesario especificar el trazado que existe entre cada realización de caso de uso – análisis y cada caso de uso, en el análisis de este proyecto se ha realizado un diagrama que muestra dichas relaciones debido a que algunas de las realizaciones de caso de uso – análisis poseen otro nombre que identifica mejor el proceso que se lleva a cabo dentro de ellas.

Cada colaboración está compuesta por cuatro elementos adicionales que permiten contextualizar, al caso de uso que realizan, dentro de un grupo de clases y objetos específico. Los componentes son:

- *Diagrama de clases*: Con la ayuda de este diagrama el diseñador podrá abstraer los diferentes atributos y métodos de las clases que serán usadas en el diseño. Cada clase tiene participación en una o más realizaciones de casos de uso – análisis y de cada una de estas participaciones se puede extraer lo necesario para construir la clase del diseño.

- *Diagrama de Interacción*: Este diagrama permitirá mirar al interior del sistema en el momento que un actor o usuario inicia un caso de uso e interactúa en él. Lo que se pretende, no es mostrar la secuencia lógica de pasos en detalles cronológicos (eso se realiza en los diagramas de secuencia del sistema), sino lo que se desea, es identificar los requisitos y responsabilidades acaecidas sobre los objetos, por ello se usarán *Diagramas de Colaboración*.

- *Flujo de Sucesos – análisis*: La descripción de los sucesos que representan los diagramas de colaboración pueden ser difíciles de interpretar o leer. Debido a esto es necesaria una narración en sencilla (pero en términos de objetos) que describa

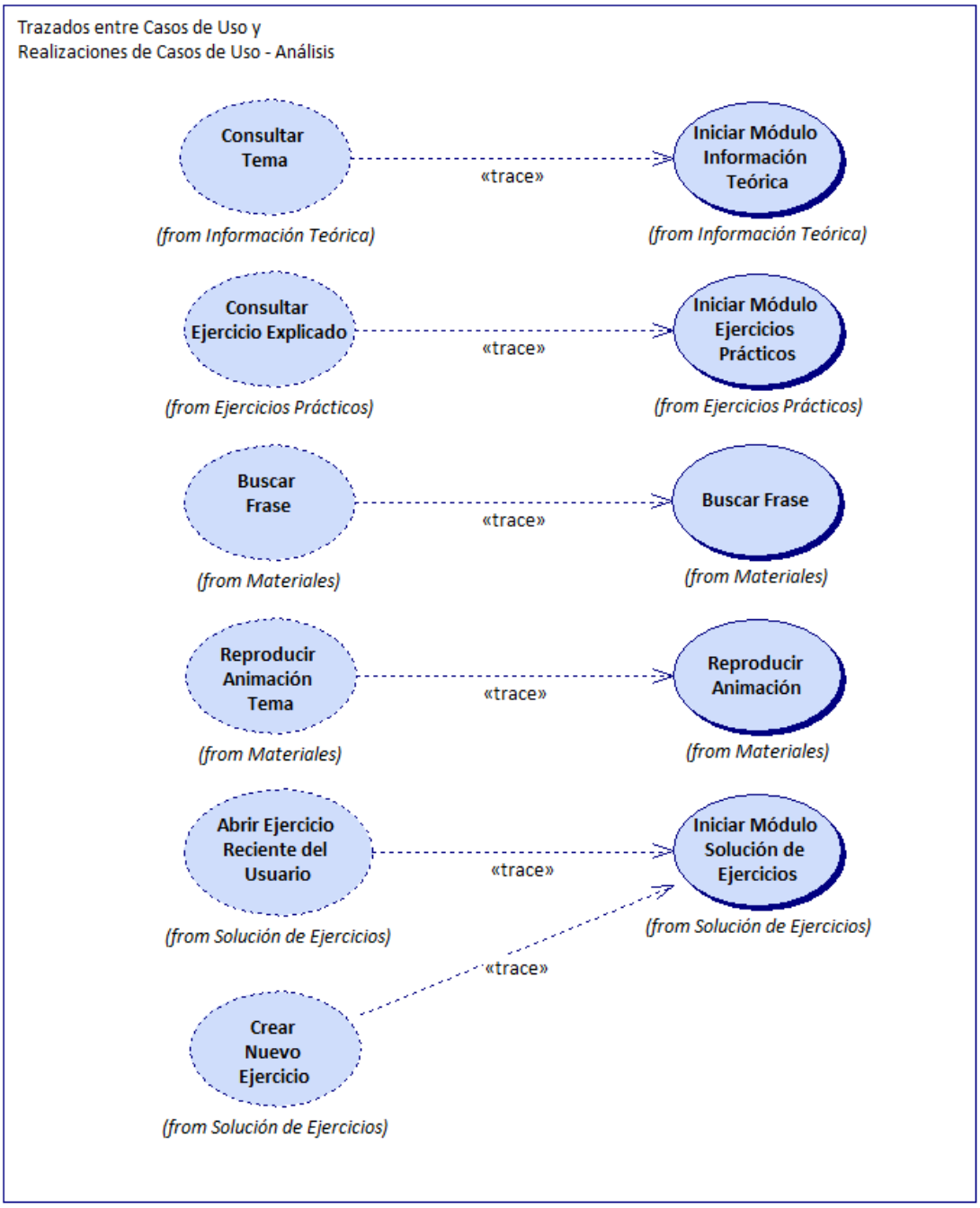
el diagrama. Esta narración del flujo de sucesos no hará mención de atributos, responsabilidades o asociaciones del objeto; estos suelen cambiar con el transcurso de las iteraciones. En este proyecto los diagramas suelen tener una descripción simplificada adjunta.

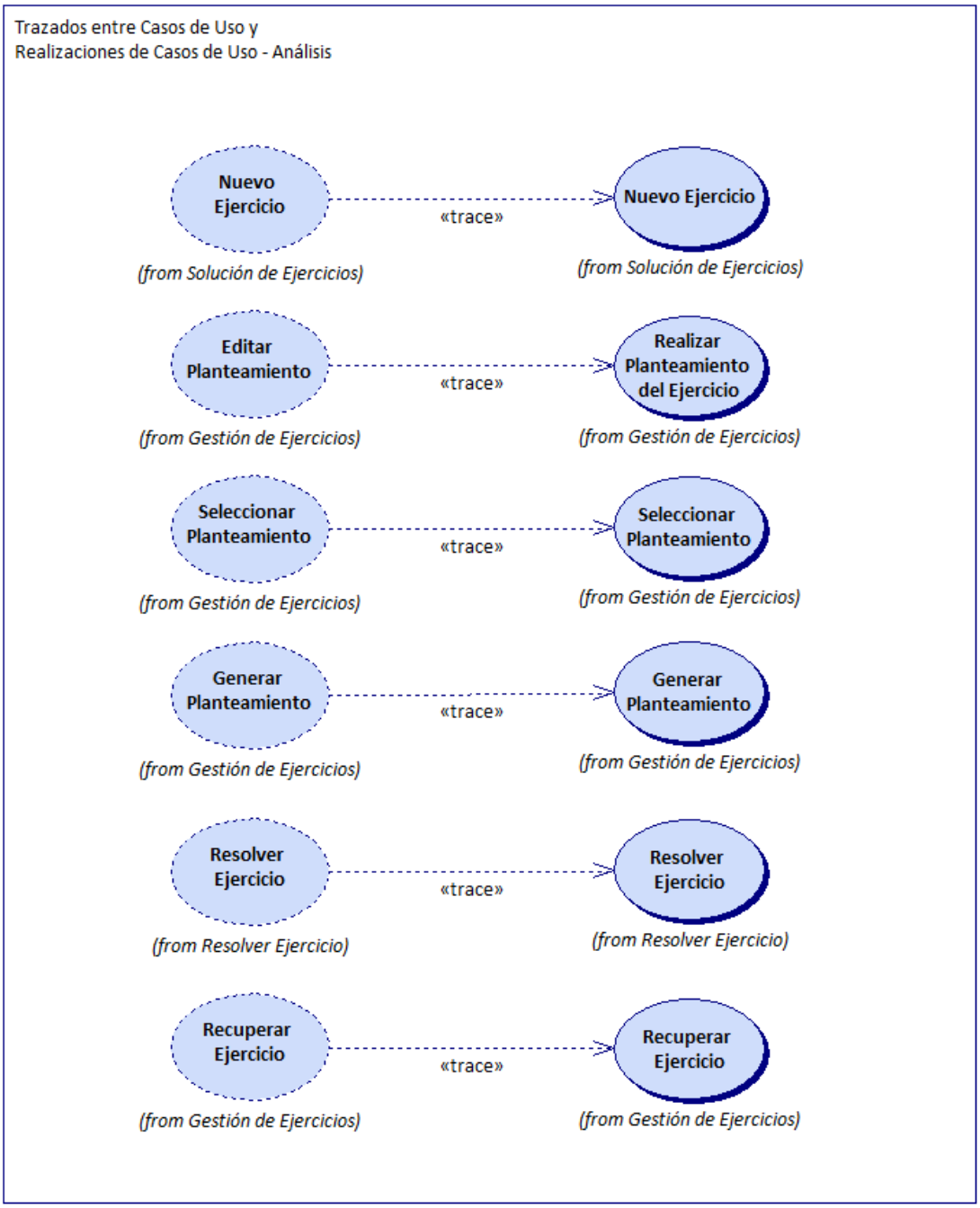
- *Requisitos especiales*: Estos son requisitos no funcionales sobre la realización de un caso de uso. Se hace, entonces, una descripción textual de dichos requisitos, que serán implementados en la fase del diseño. No todas las realizaciones de casos de uso – análisis poseen requisitos especiales; sólo se añaden cuando es necesario.

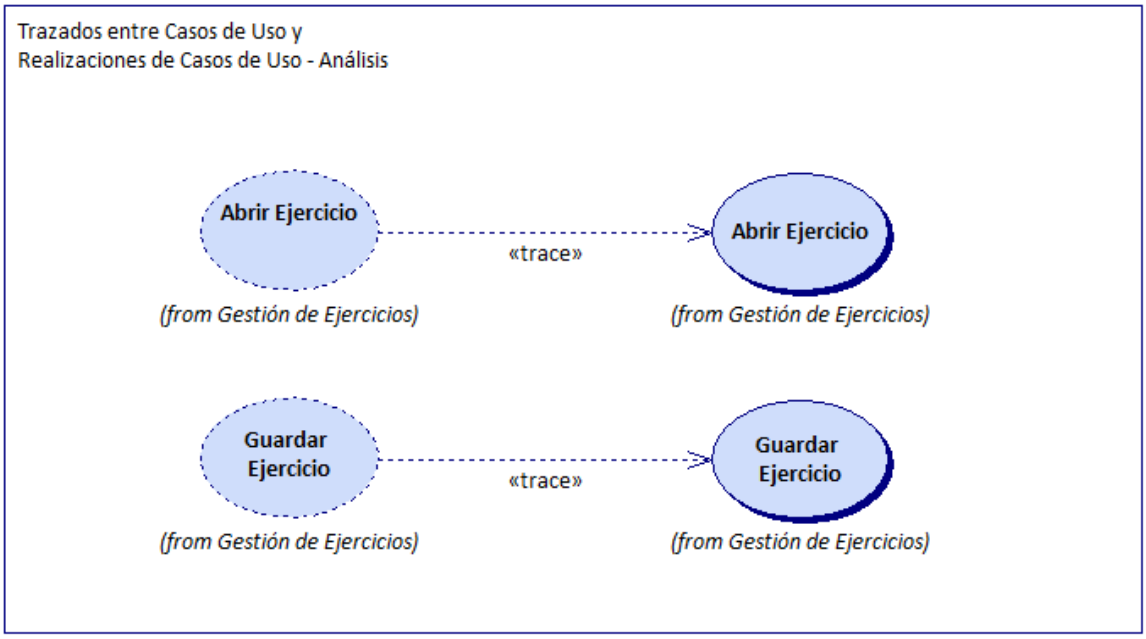
En esta sección del análisis se mostrará, primero, un listado de los trazados existentes entre cada realización de caso de uso – análisis y cada caso de uso, y posteriormente, los diagramas y la descripción de cada realización de caso de uso – análisis (colaboración).

Nota: No se han realizado colaboraciones por cada caso de uso, sólo se han modelado aquellas que sirvan para agrupar y relacionar un porcentaje sumamente alto de las clases del análisis.

5.2.1 Trazados entre realizaciones de casos de uso – análisis y casos de uso



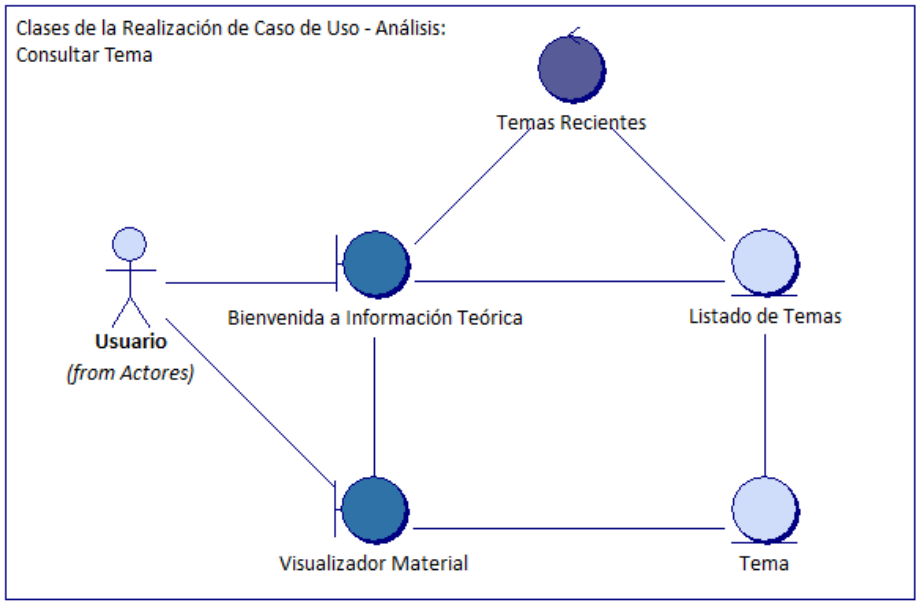

Trazados entre Casos de Uso y
Realizaciones de Casos de Uso - Análisis
Nuevo Ejercicio
«trace»
Nuevo Ejercicio
(from Solución de Ejercicios)
(from Solución de Ejercicios)
Editar Planteamiento
«trace»
Realizar Planteamiento del Ejercicio
(from Gestión de Ejercicios)
(from Gestión de Ejercicios)
Seleccionar Planteamiento
«trace»
Seleccionar Planteamiento
(from Gestión de Ejercicios)
(from Gestión de Ejercicios)
Generar Planteamiento
«trace»
Generar Planteamiento
(from Gestión de Ejercicios)
(from Gestión de Ejercicios)
Resolver Ejercicio
«trace»
Resolver Ejercicio
(from Resolver Ejercicio)
(from Resolver Ejercicio)
Recuperar Ejercicio
«trace»
Recuperar Ejercicio
(from Gestión de Ejercicios)
(from Gestión de Ejercicios)



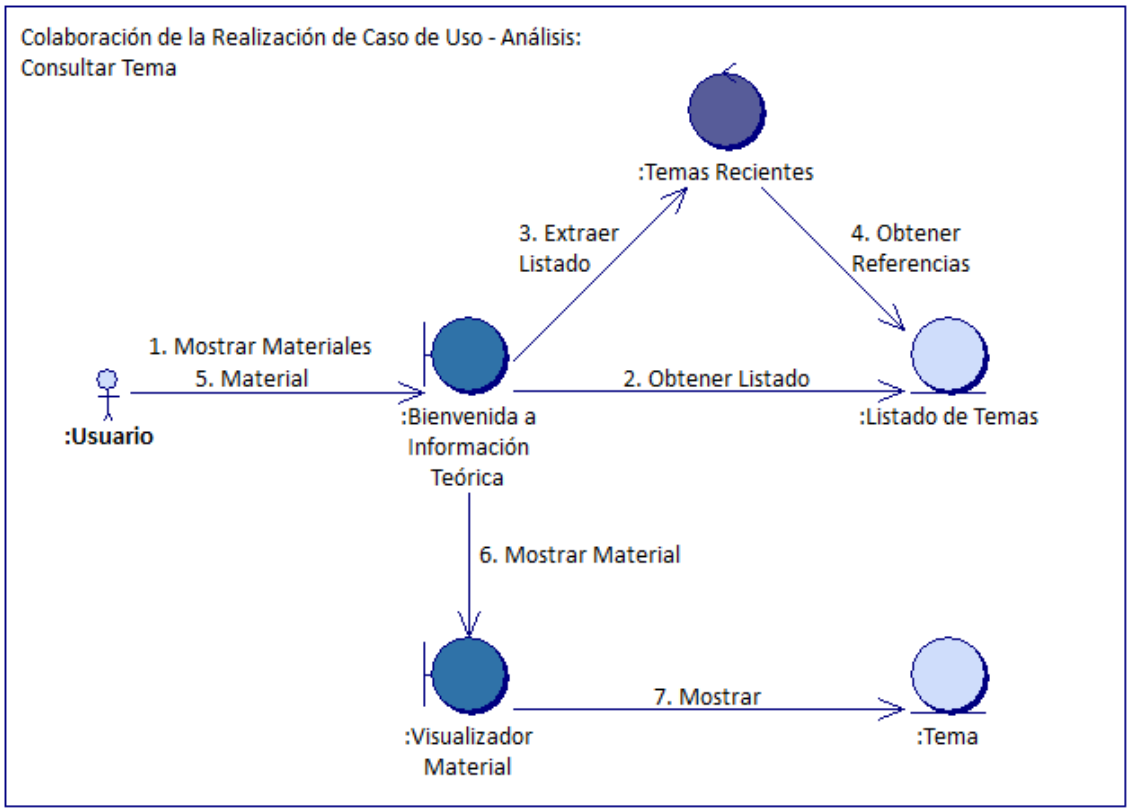
5.2.2 Colaboraciones

5.2.2.1 Consultar tema

5.2.2.1.1 Diagrama de clases



5.2.2.1.2 Diagrama de colaboración



5.2.2.1.3 Flujo de sucesos – análisis

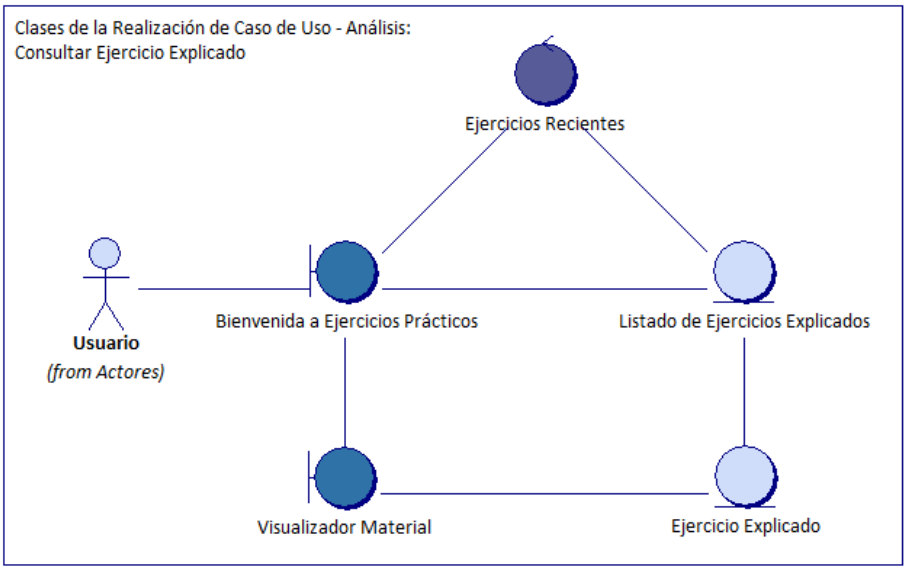
El Usuario abre al Módulo de Información Teórica y le solicita que muestre los materiales (temas del módulo y temas estudiados recientemente) disponibles (1) y éste carga y muestra en pantalla el listado de los temas que se analizan en el módulo (2) y el listado de los temas analizados recientemente (3, 4).

El usuario indica al módulo el material que desea que se visualice (5) y, entonces, se abre el Visualizador del Material proporcionándole la referencia o ruta del material (6) y él muestra el Material indicado (7).

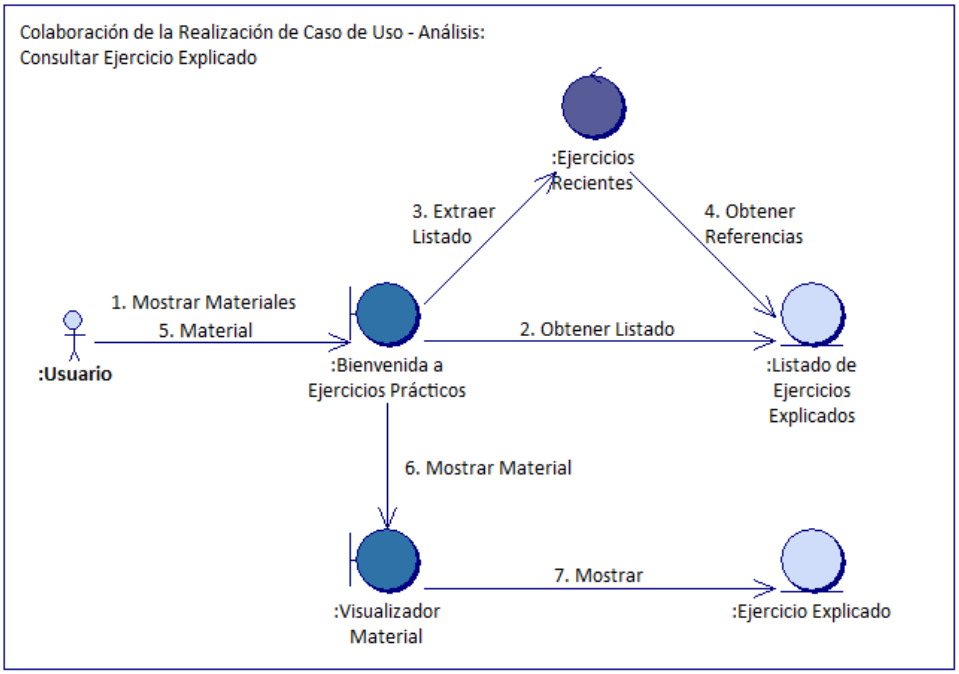
El material seleccionado por el usuario puede ser uno del listado de temas del módulo o del listado de temas recientes.

5.2.2.2 Consultar ejercicio explicado

5.2.2.2.1 Diagrama de clases



5.2.2.2.2 Diagrama de colaboración



5.2.2.2.3 Flujo de sucesos – análisis

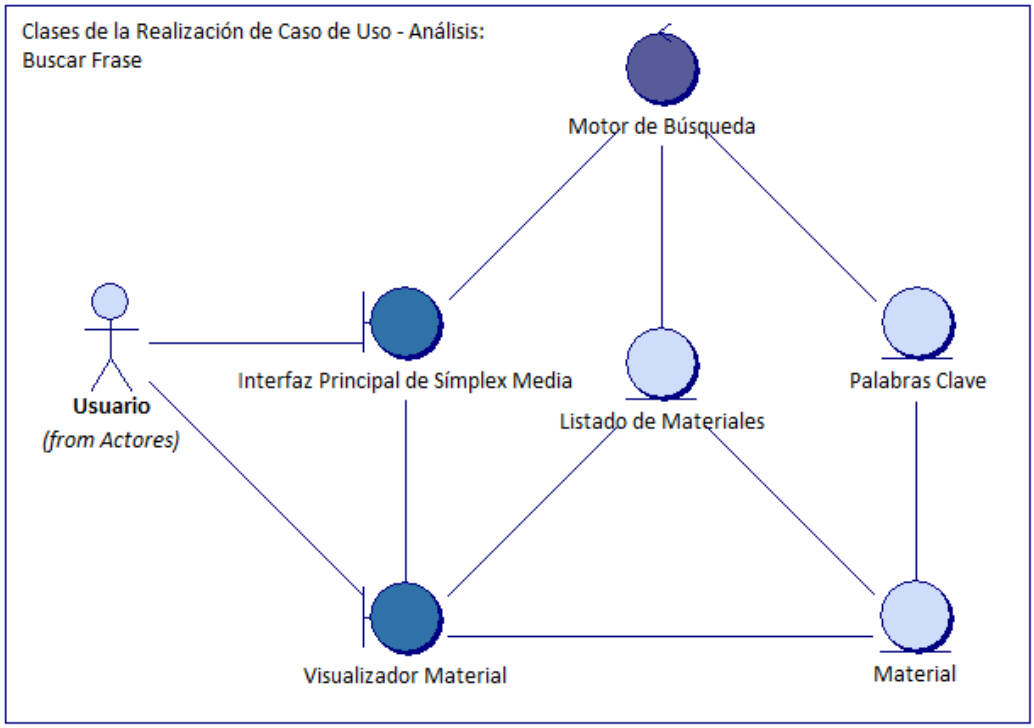
El Usuario abre al Módulo de Ejercicios Prácticos y le solicita que muestre los materiales (todos los ejercicios explicados del módulo y los ejercicios explicados que se hayan estudiado recientemente) disponibles (1) y éste carga y muestra en pantalla el listado de los ejercicios explicados que se analizan en el módulo (2) y el listado de los ejercicios explicados analizados recientemente (3, 4).

El usuario indica al módulo el material que desea que se visualice (5) y, entonces, se abre el Visualizador del Material proporcionándole la referencia o ruta del material (6) y se muestra el Material indicado (7).

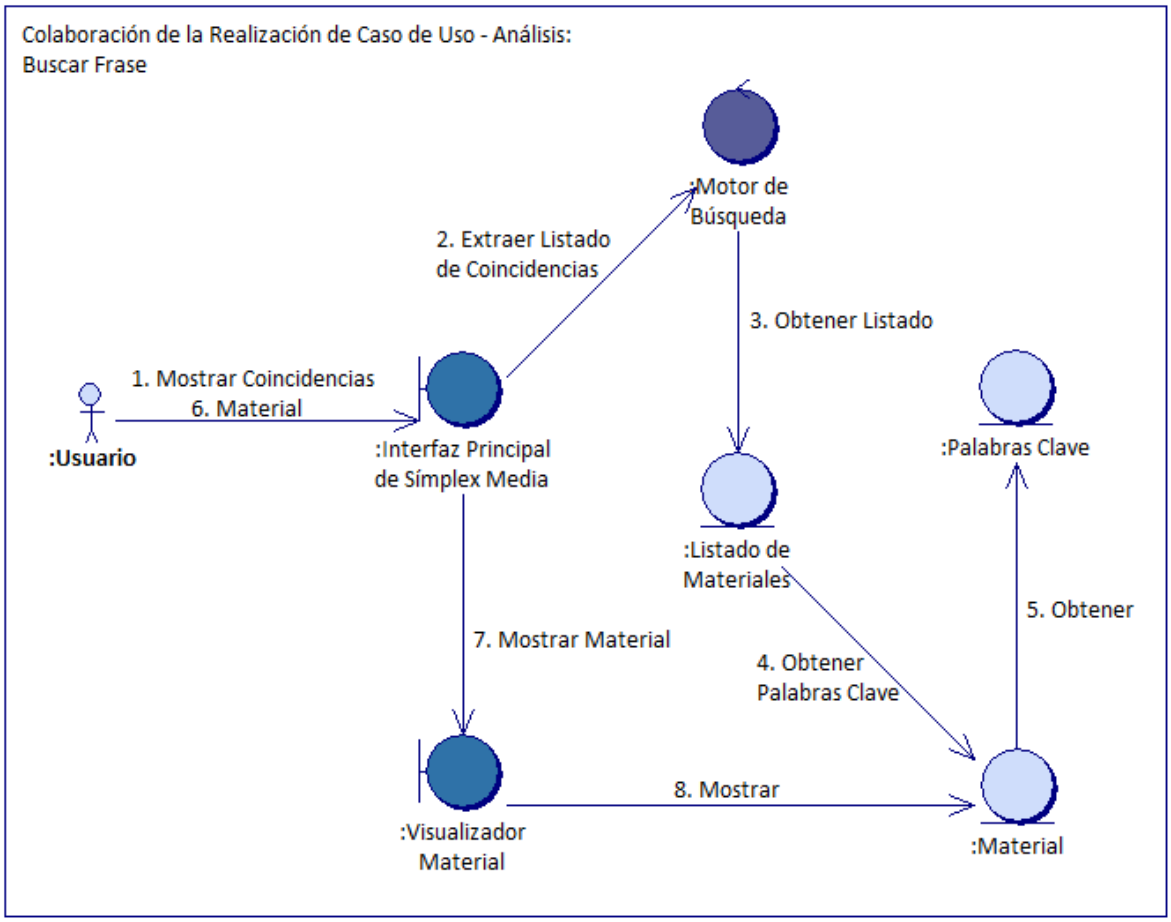
El material seleccionado por el usuario puede ser uno del listado de ejercicios explicados del módulo o del listado de ejercicios explicados recientes.

5.2.2.3 Buscar frase

5.2.2.3.1 Diagrama de clases



5.2.2.3.2 Diagrama de colaboración



5.2.2.3.3 Flujo de sucesos – análisis

El Usuario ingresa en la Interfaz Principal de Símplex Media la frase a buscar y ordena la búsqueda ordenando que se muestre un listado de materiales con coincidencias (1); éstas pueden ser temas, ejercicios explicados o términos. La Interfaz Principal de Símplex Media utiliza al Motor de Búsqueda para hallar las coincidencias (2).

El Motor de Búsqueda examina, en el Listado de Materiales, las Palabras Clave de cada Material para comprobar si hay coincidencias (3, 4, 5).

En caso de no haber resultado de materiales se muestra un mensaje que indique este hecho (no se muestra en el diagrama).

Si hay resultados se los muestra en pantalla para que el usuario seleccione el que desea analizar (6) y, entonces, se abre el Visualizador del Material proporcionándole la referencia o ruta del material (7) y se muestra el Material indicado (8).

5.2.2.3.4 Requisitos especiales

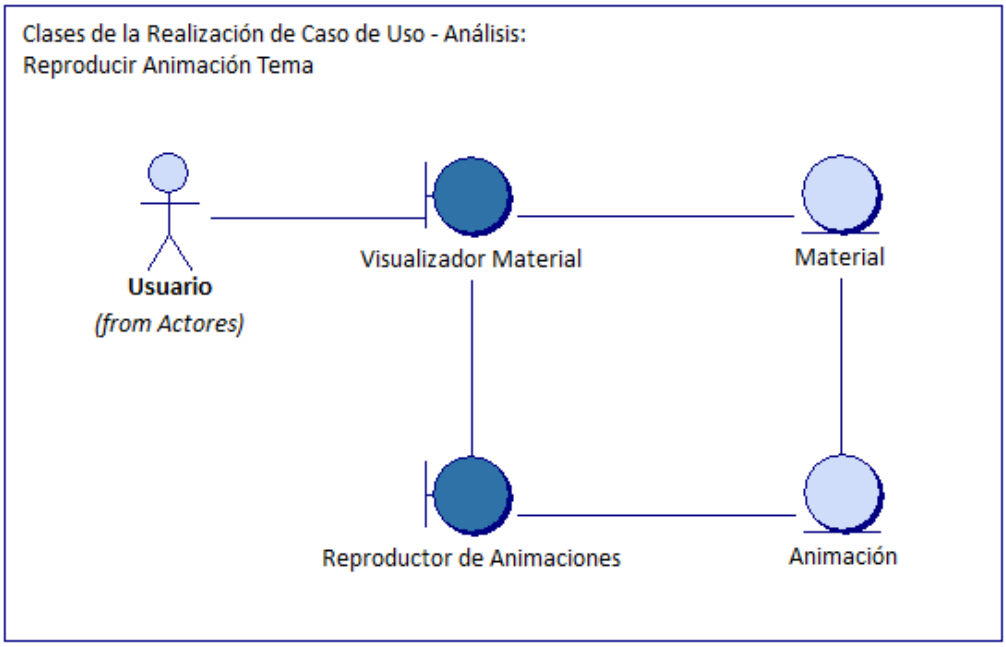
- Los materiales deben tener un listado de palabras claves, de tal manera que, el motor de búsqueda, examine el contenido de este listado y no el contenido completo del material. Esto ahorrará el tiempo de búsqueda.

- Al indicarle al buscador determinadas palabras clave, deben mostrarse todos los términos y todos los ejercicios explicados del sistema.

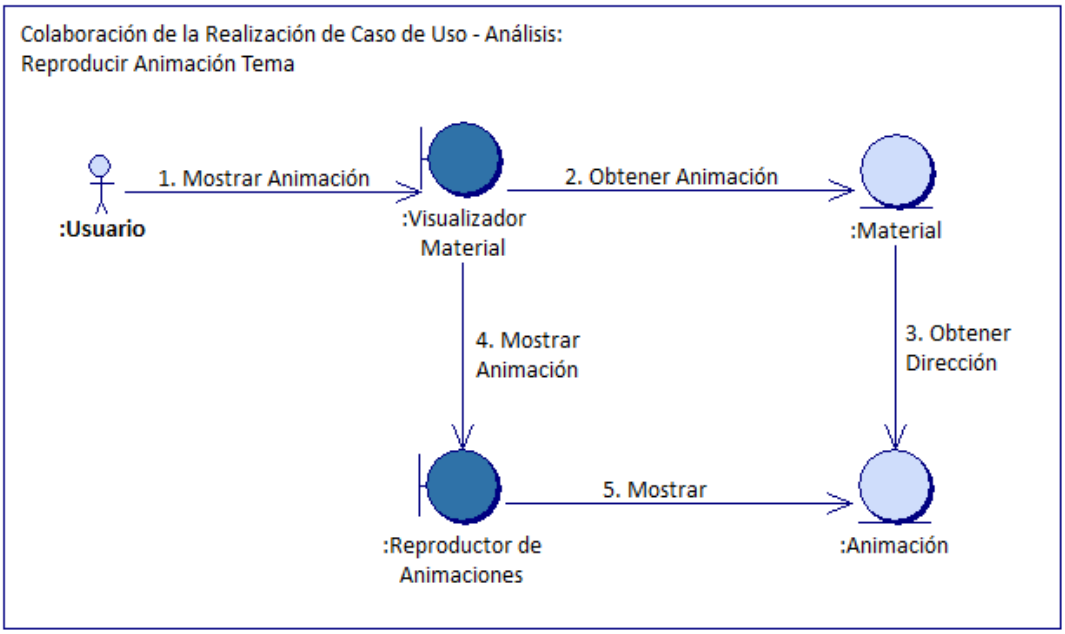
- Para buscar dentro del contenido de un material se debe utilizar la herramienta visual de los Documentos de Flujo (o *Documentos Dinámicos*).

5.2.2.4 Reproducir animación tema

5.2.2.4.1 Diagrama de clases



5.2.2.4.2 Diagrama de colaboración



5.2.2.4.3 Flujo de sucesos – análisis

El Usuario solicita al Visualizador Material que reproduzca la animación del material que se está visualizando (1). El Visualizador Material obtiene del Material, que se está mostrando, la ruta de la Animación que está asociada a él (2, 3).

Con la ruta de la Animación se abre el Reproductor de Animaciones (4) y se inicia la reproducción (5).

En caso que el Material presente no posea una Animación asociada a él, se presenta un mensaje al Usuario que lo indique (no se muestra en el diagrama).

5.2.2.4.4 Requisitos especiales

- Los materiales pueden tener una o más animaciones que expliquen todo o parte de su contenido.

- EL Reproductor de Animaciones debe tener una sección que muestre todas las animaciones asociadas a un material (en caso que este posea más de una).

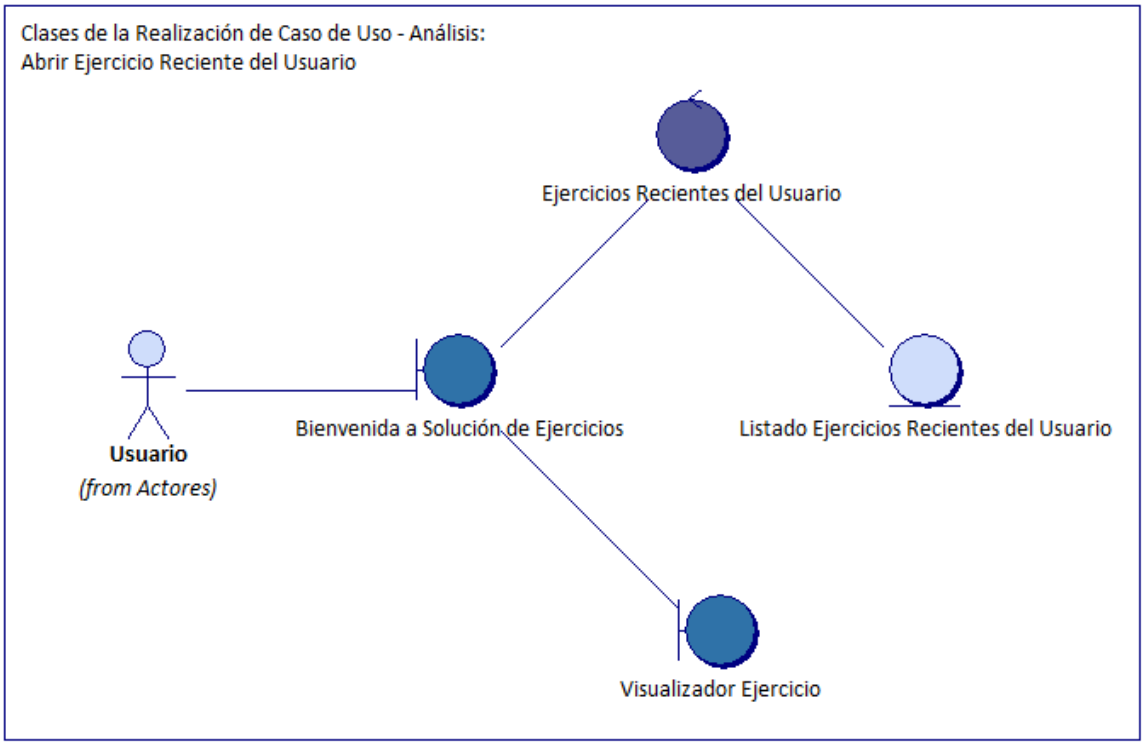
- El usuario podrá manipular la línea de reproducción para adelantar o retroceder la animación que se esté visualizando.

- La narración del contenido de la animación puede ser comentada por un dibujo animado (Agente de Microsoft). El usuario podrá especificar si desea esta funcionalidad o no.

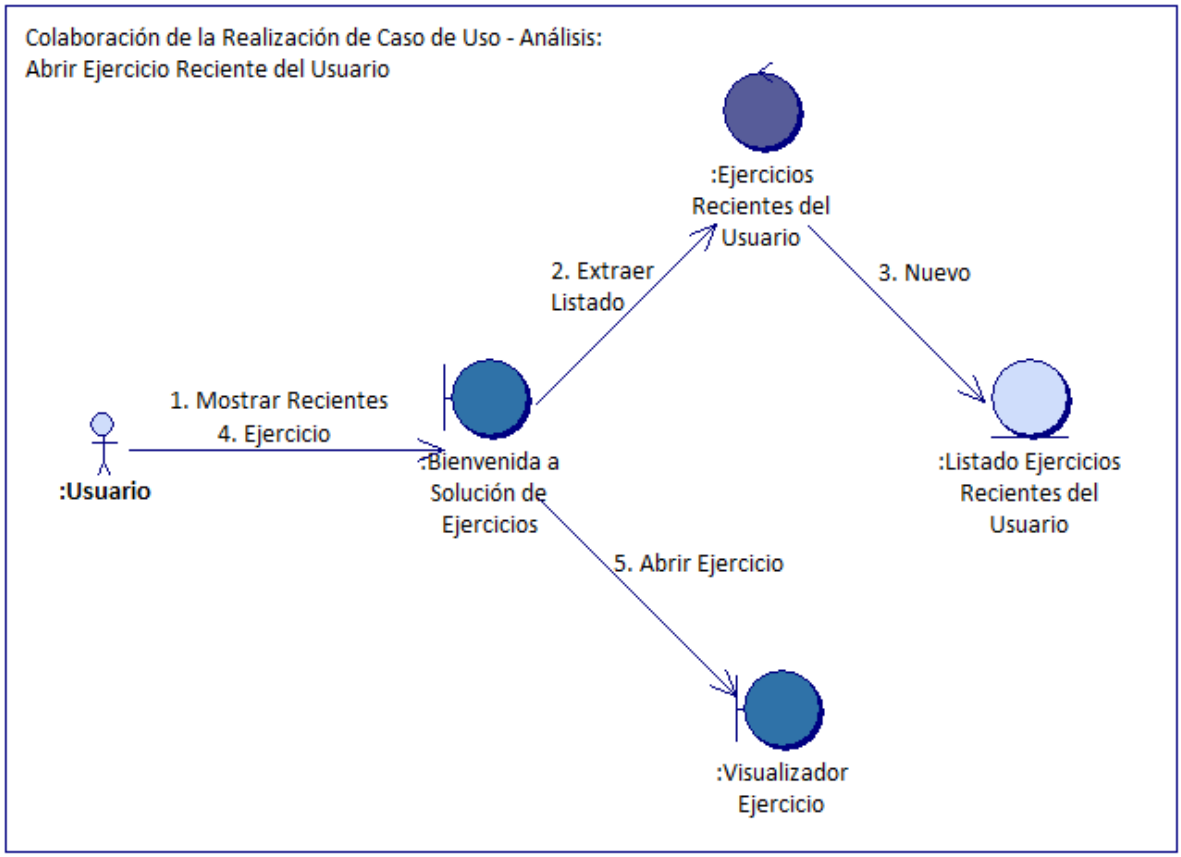
- En caso que la animación sea narrada por un Agente, no debe permitirse que se manipule la línea de reproducción.

5.2.2.5 Abrir ejercicio reciente del usuario

5.2.2.5.1 Diagrama de clases



5.2.2.5.2 Diagrama de colaboración



5.2.2.5.3 Flujo de sucesos – análisis

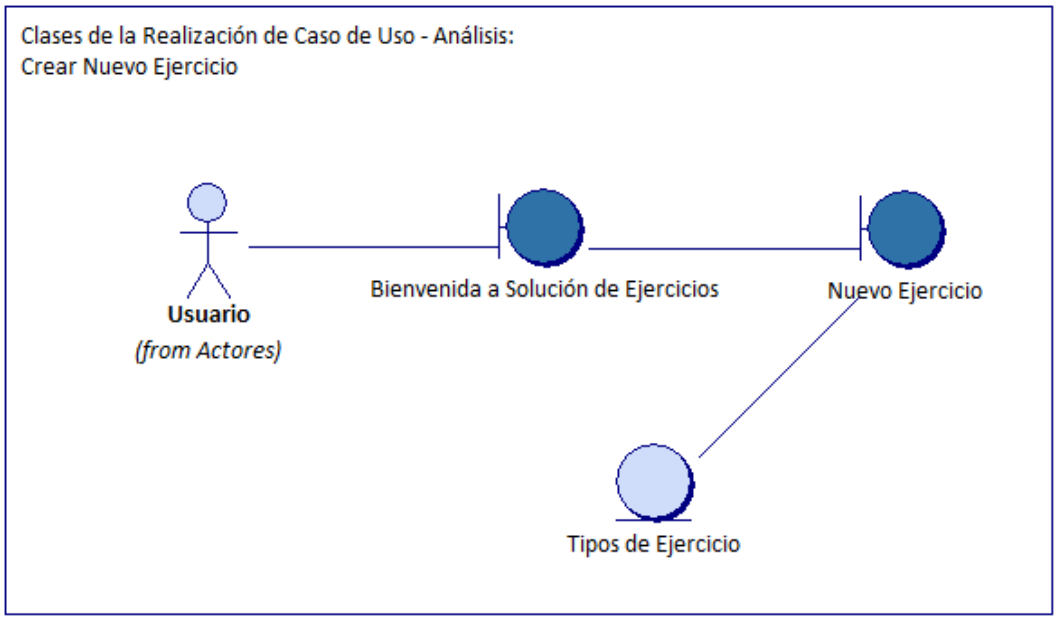
El Usuario abre al Módulo Solución de Ejercicios y le solicita que muestre el listado de ejercicios revisados recientemente (1) y éste, por medio de Ejercicios Recientes del Usuario (2), quién crea un nuevo Listado de Ejercicios Recientes del Usuario (3), carga y muestra en pantalla el listado de dichos ejercicios.

El usuario indica al módulo el ejercicio que desea abrir (4) y, entonces, se abre el Visualizador Ejercicio proporcionándole la ruta del ejercicio (5).

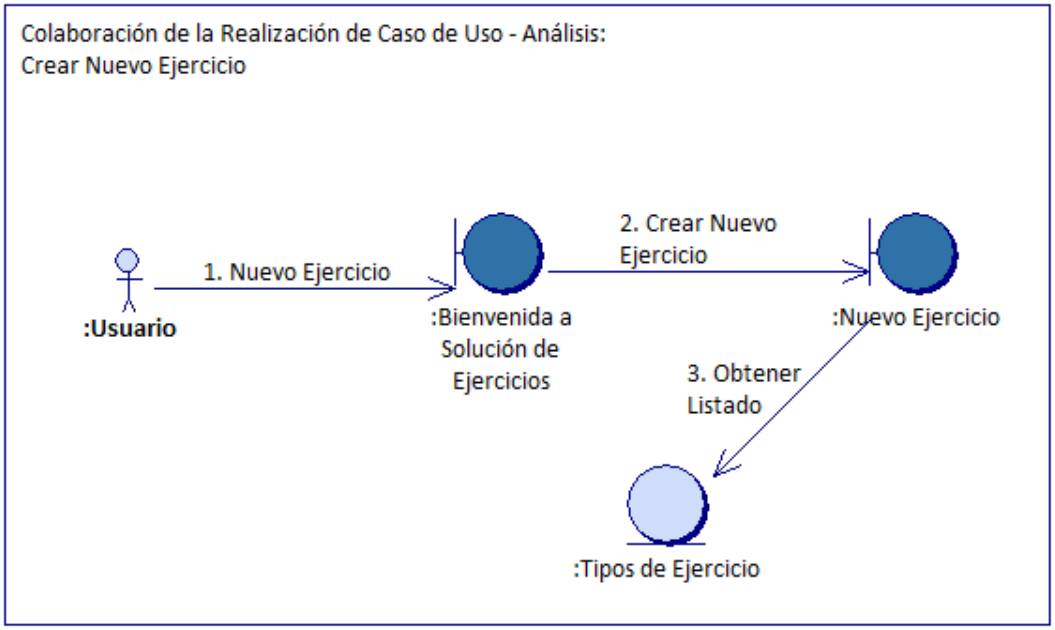
En la Colaboración *Recuperar Ejercicio* (Véase la sección 6.2.2.12) se describe el procedimiento de apertura de un archivo y su posterior visualización.

5.2.2.6 Crear nuevo ejercicio

5.2.2.6.1 Diagrama de clases



5.2.2.6.2 Diagrama de colaboración



5.2.2.6.3 Flujo de sucesos – análisis

El Usuario abre al Módulo Solución de Ejercicios e inicia la creación de un nuevo ejercicio (1). Se abre Nuevo Ejercicio (2) y se muestra un listado con los Tipos de Ejercicios disponibles en el sistema (3).

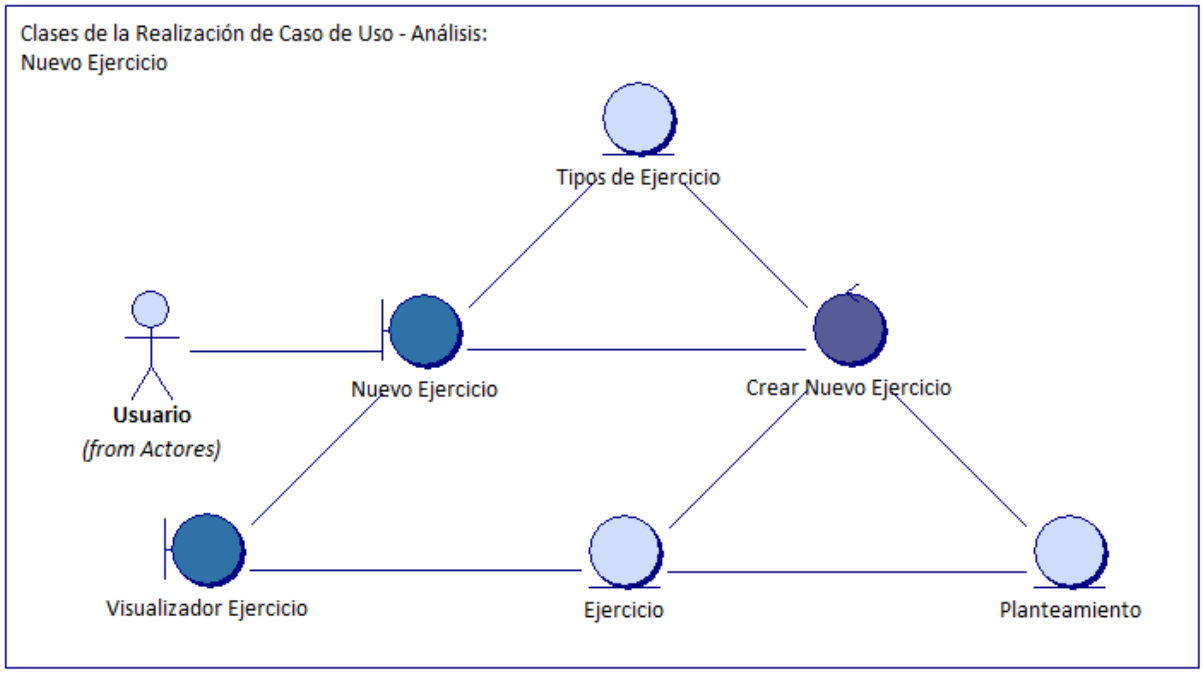
En la Colaboración *Nuevo Ejercicio* (Véase la sección 6.2.2.7) se describe el procedimiento de la creación de un Ejercicio.

5.2.2.6.4 Requisitos especiales

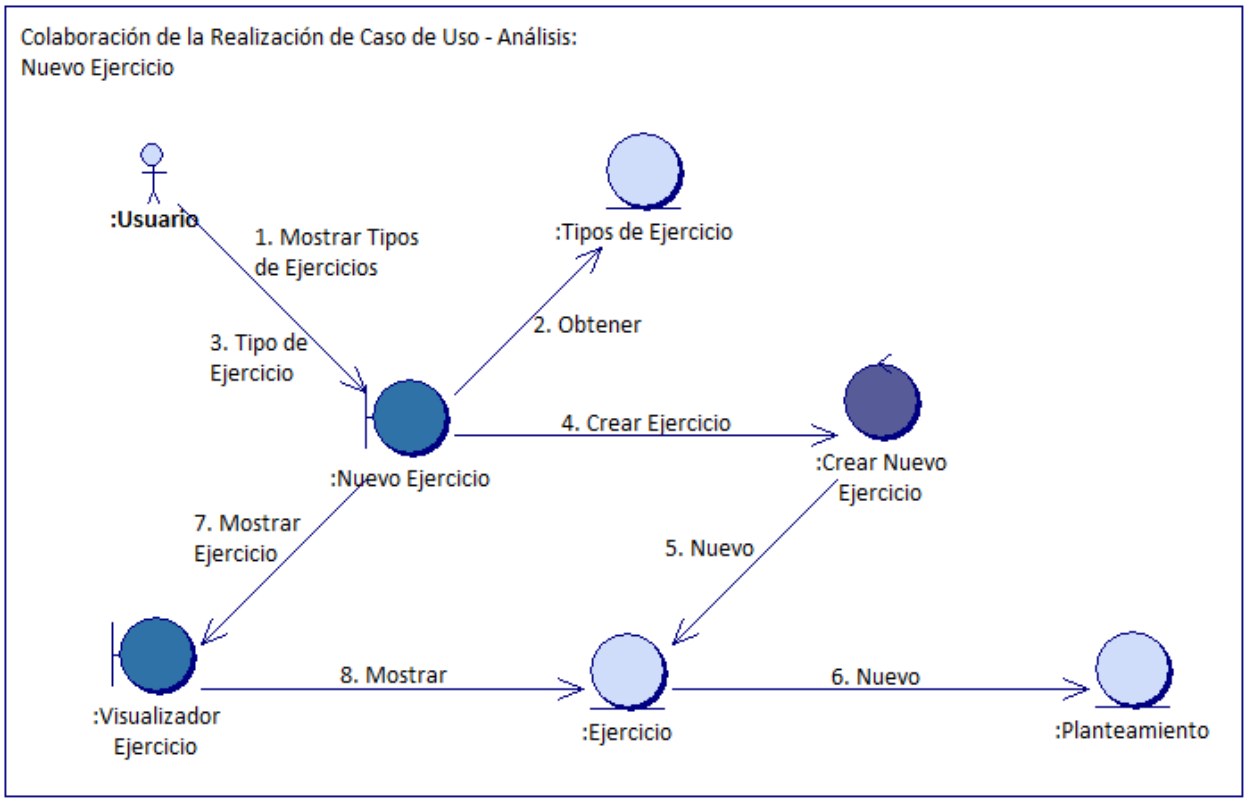
- El listado de ejercicios debe abarcar todos los algoritmos del Método Símplex Primal, el algoritmo del Método Símplex Dual, los métodos de eliminación de Gauss y Gauss Jordan y las operaciones con matrices.

5.2.2.7 Nuevo ejercicio

5.2.2.7.1 Diagrama de clases



5.2.2.7.2 Diagrama de colaboración



5.2.2.7.3 Flujo de sucesos – análisis

El Usuario solicita a Nuevo Ejercicio el listado de los Tipos de Ejercicios (1), éste los muestra (2) y el Usuario selecciona uno de ellos (3). Con el tipo de ejercicio se crea un nuevo Ejercicio con su Planteamiento a través de Crear Nuevo Ejercicio (4, 5, 6).

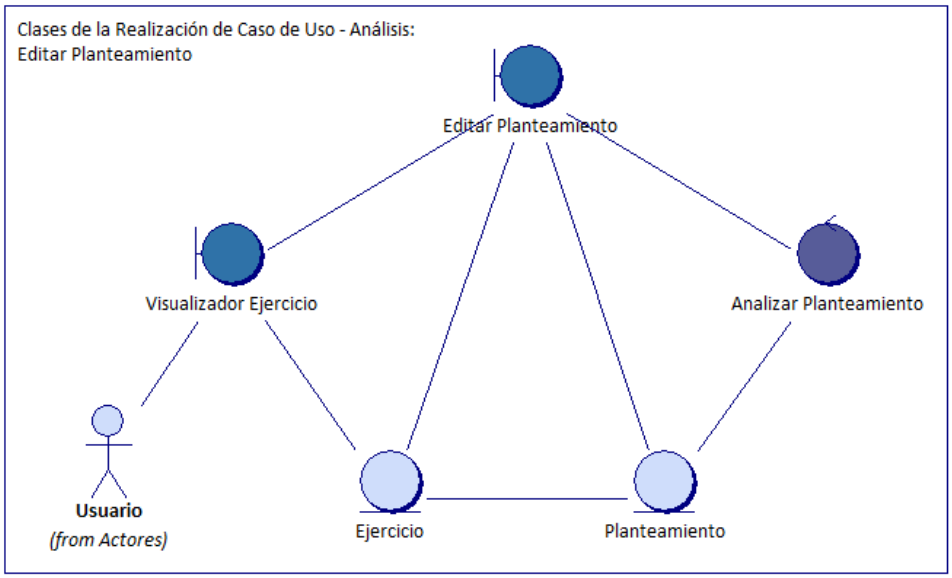
Se abre el Visualizador Ejercicio (7) y se muestra al Usuario el Ejercicio creado (8).

5.2.2.7.4 Requisitos especiales

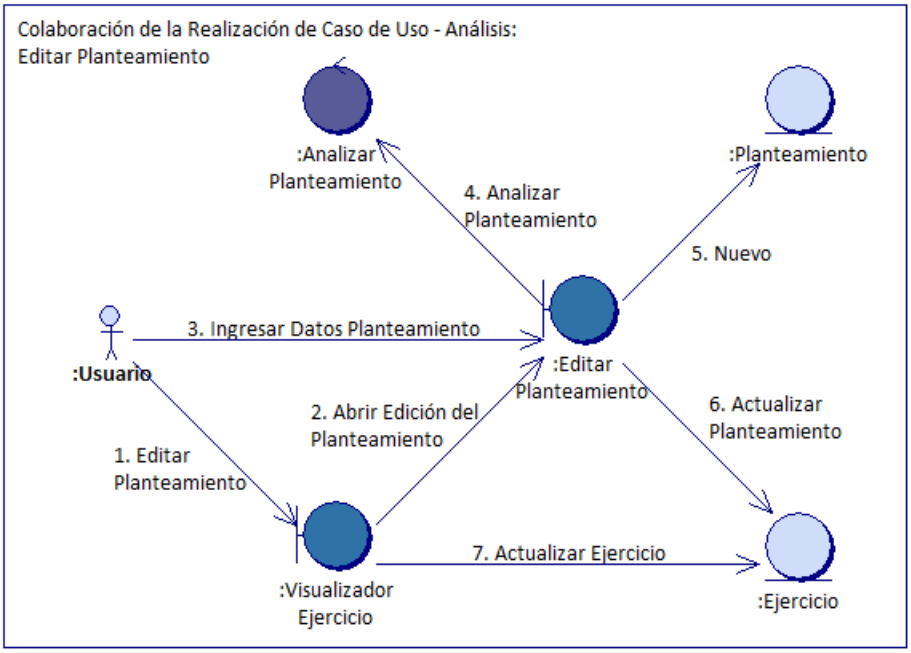
- Cada vez que se cree un nuevo ejercicio, el sistema debe proveer un planteamiento predeterminado (para cualquiera de los tipos de ejercicios) que aparecerá por defecto. El usuario podrá modificar dicho planteamiento.

5.2.2.8 Editar planteamiento

5.2.2.8.1 Diagrama de clases



5.2.2.8.2 Diagrama de colaboración



5.2.2.8.3 Flujo de sucesos – análisis

El Usuario le indica al Visualizador Ejercicio que desea modificar el planteamiento del ejercicio (1), éste abre la ventana de edición Editar Planteamiento (2) y el Usuario ingresa los nuevos datos del planteamiento (3).

El Editor Planteamiento usa a Analizar Planteamiento para comprobar que la información ingresada sea correcta (4) y en caso afirmativo se crea un nuevo Planteamiento con los datos ingresados (5) y se actualiza el planteamiento en el Ejercicio (6).

Finalmente, el Visualizador Ejercicio actualiza la información del ejercicio en pantalla (7).

5.2.2.8.4 Requisitos especiales

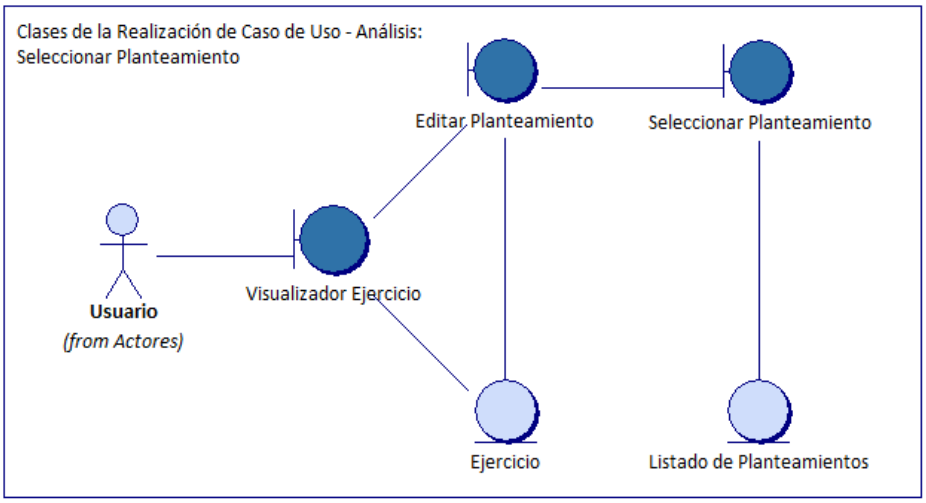
- En el momento de ingresar el planteamiento del ejercicio de forma manual, el sistema debe permitir al usuario, ingresar expresiones aritméticas y tener la capacidad de interpretarlas. Por ejemplo: será posible ingresar la expresión “5 + (8/3)”, y el sistema debe compilar la expresión y trabajar con su resultado, 7,666.

- Al ingresar planteamientos del Método Símplex, el sistema permitirá el ingreso de valores negativos en el lado derecho de las restricciones. En el paso del modelo a la forma estándar, el mismo sistema tratará el problema de dicha restricción.

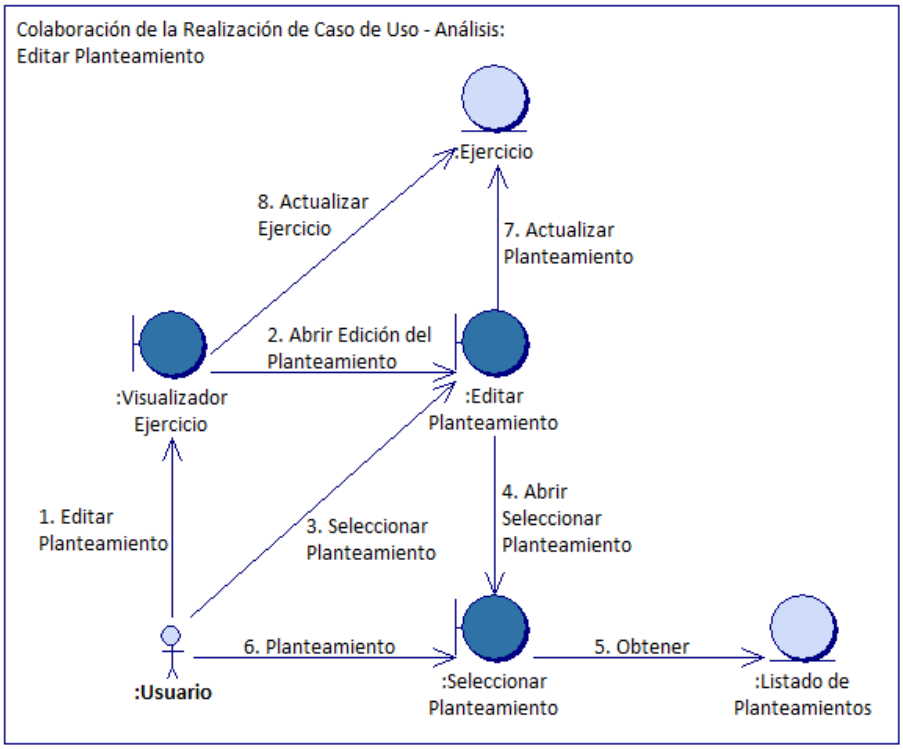
- Los planteamientos ingresados deben ser analizados pare determinar si el algoritmo seleccionado puede resolver un problema que contenga los signos de relación que el usuario ha especificado para cada restricción.

5.2.2.9 Seleccionar planteamiento

5.2.2.9.1 Diagrama de clases



5.2.2.9.2 Diagrama de colaboración



5.2.2.9.3 Flujo de sucesos – análisis

El Usuario le indica al Visualizador Ejercicio que desea modificar el planteamiento del ejercicio (1), éste abre la ventana de edición Editar Planteamiento (2) y el Usuario le indica a esta última que desea seleccionar un planteamiento del sistema (3).

El Editor Planteamiento abre a Seleccionar Planteamiento (4) y le muestra al usuario un listado de todos los planteamientos del sistema (5), y el Usuario selecciona el planteamiento deseado (6). Con el Planteamiento seleccionado se actualiza el Ejercicio (7).

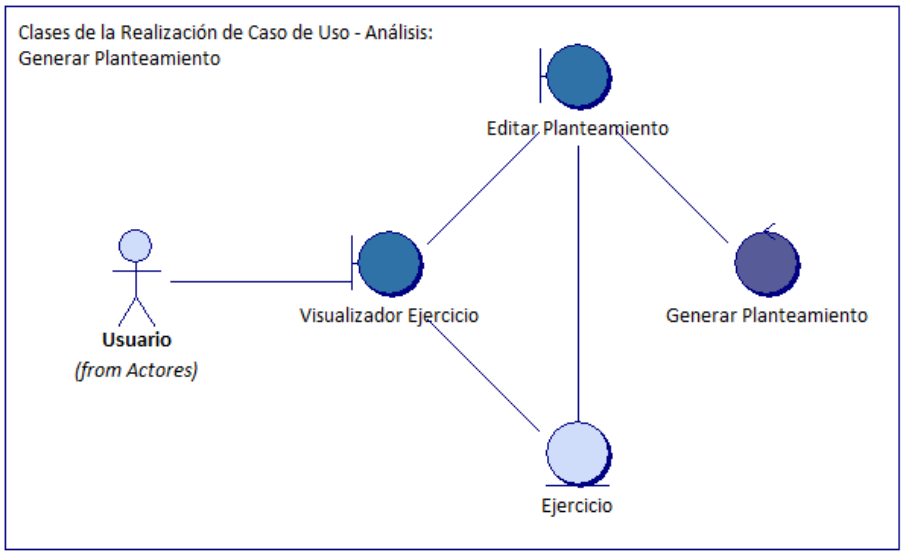
Finalmente, el Visualizador Ejercicio actualiza la información del ejercicio en pantalla (8).

5.2.2.9.4 Requisitos especiales

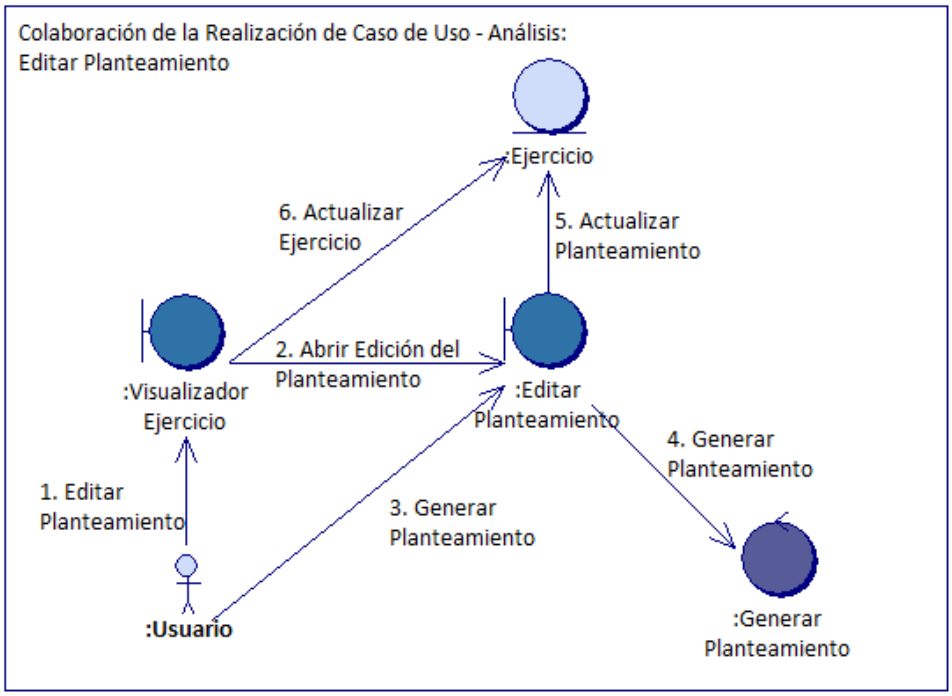
- Los planteamientos que se pueden seleccionar solo son para el Método Símplex.
-
- Los planteamientos que se muestren al usuario deben tener una descripción textual que le permita conocer qué representa cada uno de los valores.
-
- Los planteamientos se podrán observar en forma de un modelo de programación lineal y en forma de texto.

5.2.2.10 Generar planteamiento

5.2.2.10.1 Diagrama de clases



5.2.2.10.2 Diagrama de colaboración



5.2.2.10.3 Flujo de sucesos – análisis

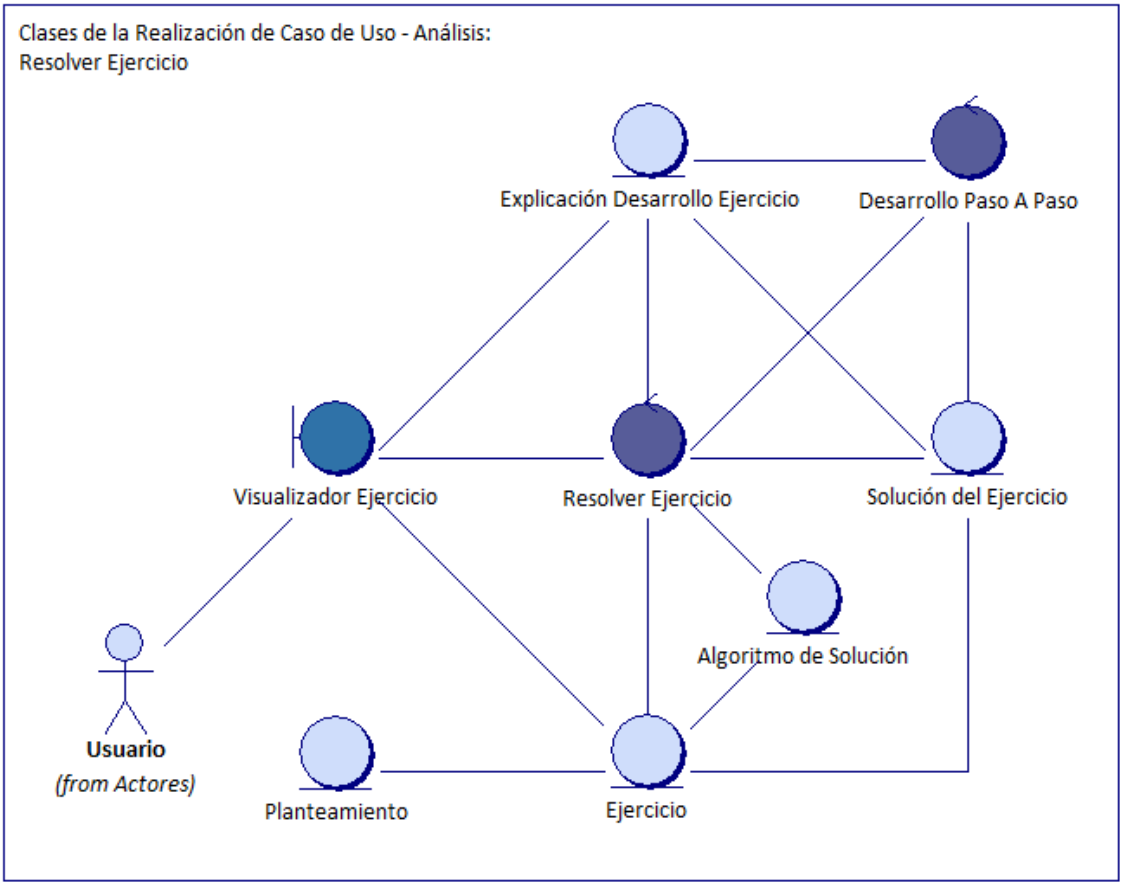
El Usuario le indica al Visualizador Ejercicio que desea modificar el planteamiento del ejercicio (1), éste abre la ventana de edición Editar Planteamiento (2) y el Usuario le indica a esta última que desea generar un planteamiento del sistema (3).

El Editor Planteamiento usa al Generador de Planteamientos (4) para crear un nuevo Planteamiento. Con el Planteamiento generado se actualiza el Ejercicio (5).

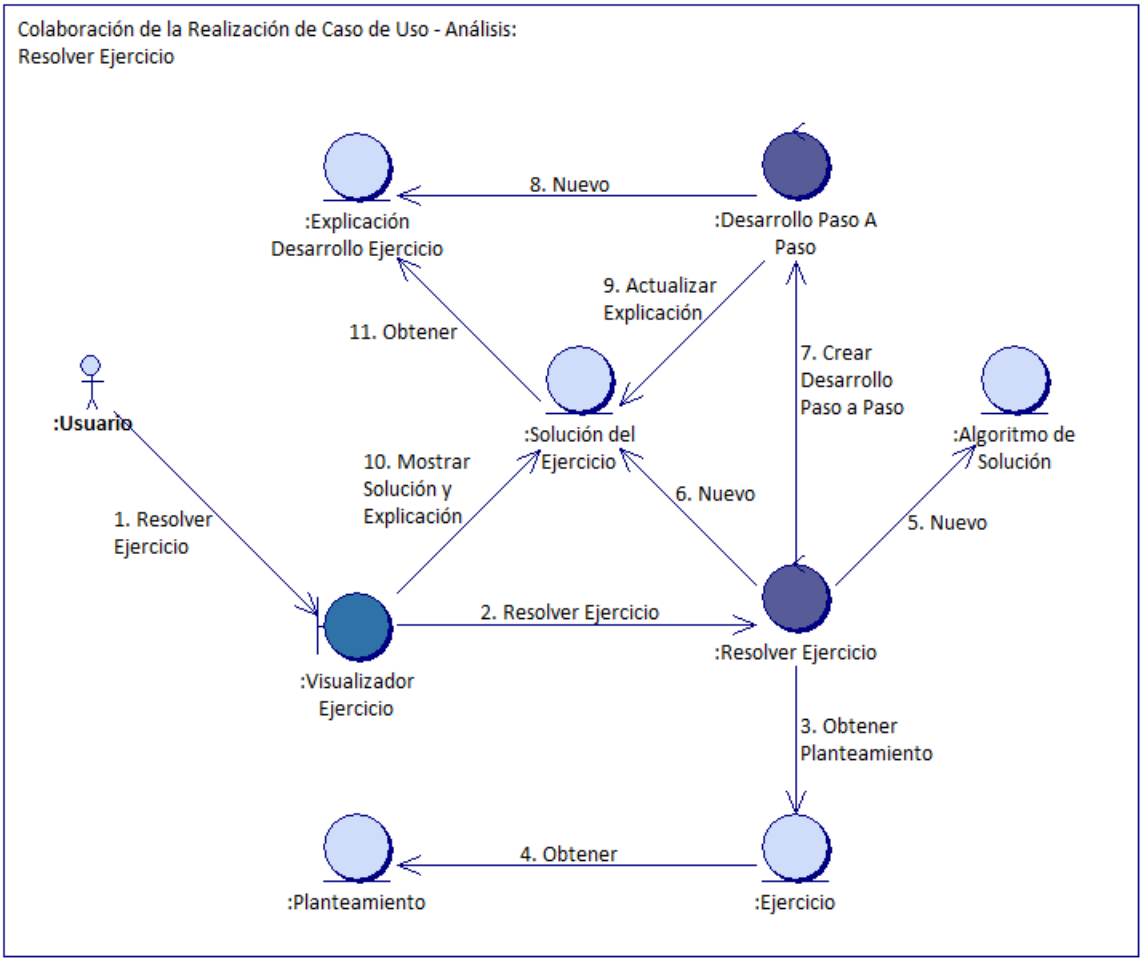
Finalmente, el Visualizador Ejercicio actualiza la información del ejercicio en pantalla (6).

5.2.2.11 Resolver ejercicio

5.2.2.11.1 Diagrama de clases



5.2.2.11.2 Diagrama de colaboración



5.2.2.11.3 Flujo de sucesos – análisis

El Usuario le indica al Visualizador Ejercicio que desea resolver el ejercicio (1) y éste, mediante Resolver Ejercicio, inicia el proceso de solución del ejercicio (2).

Resolver Ejercicio extrae el Planteamiento del Ejercicio (3, 4) y usando un Algoritmo de Solución (5) crea una nueva Solución del Ejercicio (6). Luego, en asociación con Desarrollo Paso a Paso (7), se crea la Explicación del Desarrollo del Ejercicio (8) y se vincula esa explicación a la Solución del Ejercicio (9).

El Visualizador Ejercicio muestra en pantalla la Solución del Ejercicio y su respectiva Explicación (10, 11).

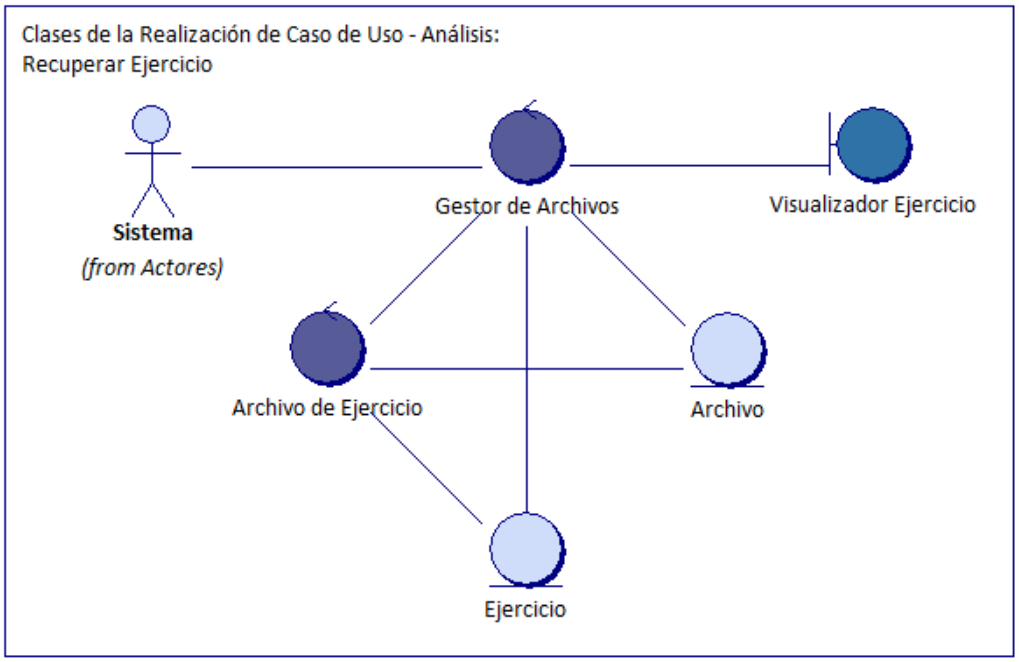
En este diagrama se ha tenido en cuenta que se desea mostrar el desarrollo paso a paso del ejercicio. En caso contrario, se omitirían los pasos que intervienen.

5.2.2.11.4 Requisitos especiales

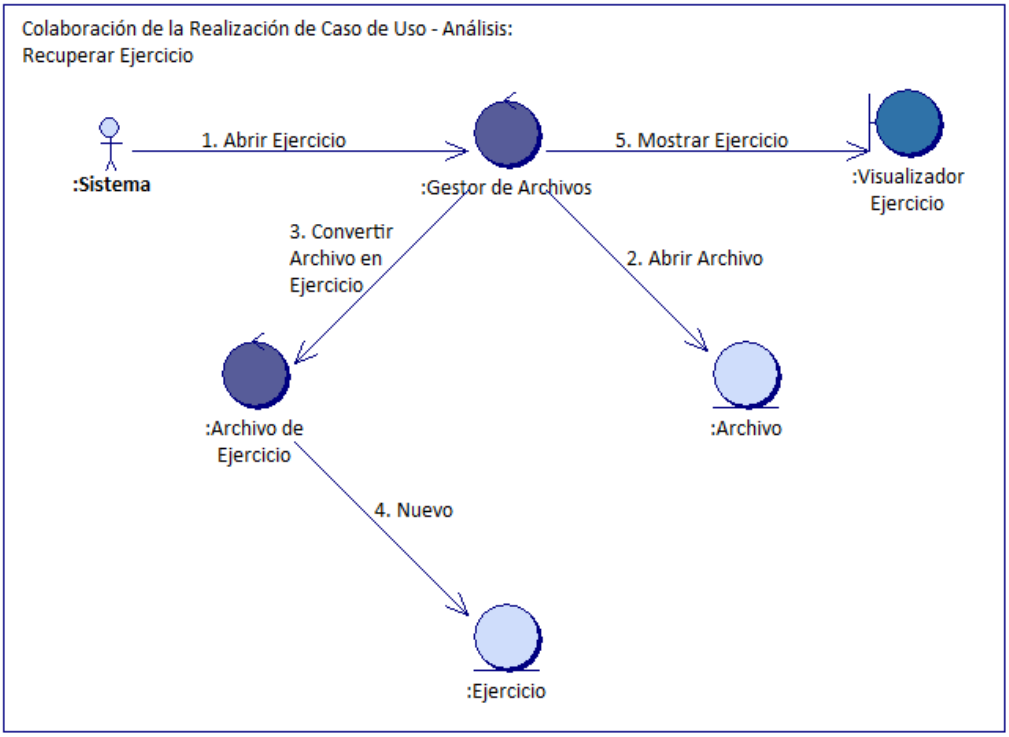
- Los algoritmos de solución serán utilizados desde su respectiva implementación realizada en el dll rcUtil.
-
- El contenido textual del desarrollo de un ejercicio contendrá vínculos a temas y términos que sustenten las acciones. La explicación de esos materiales deberán abrirse en una ventana independiente, que permita al usuario analizar el ejercicio y las explicaciones al mismo tiempo.

5.2.2.12 Recuperar ejercicio

5.2.2.12.1 Diagrama de clases



5.2.2.12.2 Diagrama de colaboración



5.2.2.12.3 Flujo de sucesos – análisis

Luego que el Usuario le ha indicado a alguna sección del sistema que desea Abrir un Archivo de Ejercicio, se hace uso del Gestor de Archivos para realizar la apertura del archivo de texto (1, 2). Luego, a través de Archivo de Ejercicio, se crea un nuevo Ejercicio con la información del archivo de texto (3, 4) y se lo muestra en el Visualizador de Ejercicios (5).

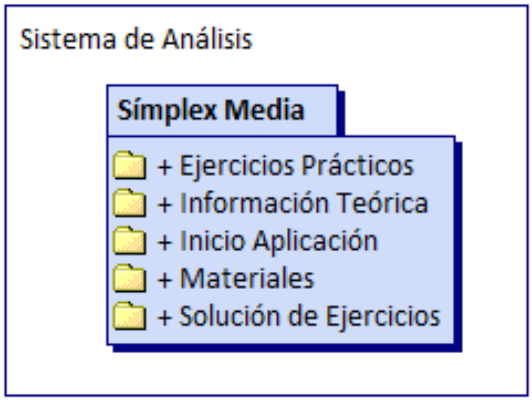
En caso que el archivo no pueda abrirse o no tenga el formato correcto se muestra el respectivo mensaje de error (No se muestra en el diagrama).

5.3 PAQUETES DEL ANÁLISIS

Con el objetivo de organizar y manejar de forma adecuada los artefactos productos del modelo de análisis, se han utilizado paquetes de análisis. En éstos se agruparán las Clases del Análisis, las Realizaciones de Casos de Uso – Análisis y otros Paquetes de Análisis en forma recursiva.

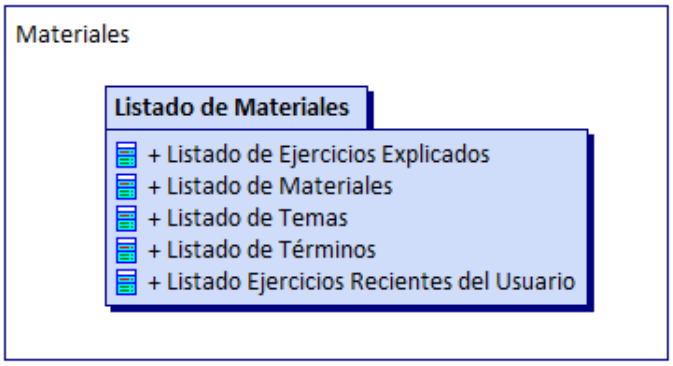
El paquete de análisis de más alto nivel del modelo es el paquete Sistema de Análisis. En este proyecto se identifica a este paquete usando el nombre: “Símplex Media”. Dentro de él se hallan los paquetes restantes.

5.3.1 Paquete: Sistema de análisis

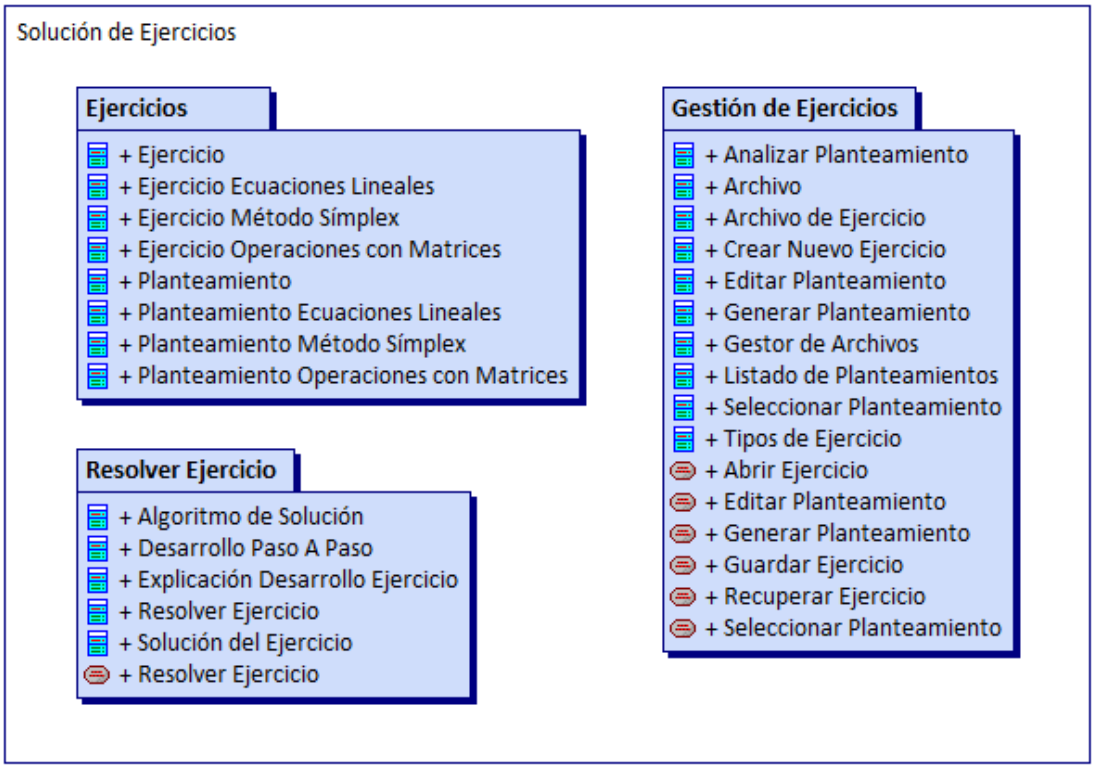


5.3.2 Paquete: SÍMPLEX MEDIA

5.3.3 Paquete: Materiales



5.3.4 Paquete: Solución de ejercicios



6 MODELO DE DISEÑO

En esta etapa del desarrollo se describe la realización física de los casos de uso. Las realizaciones de caso de uso – análisis llegan a ser refinadas en realizaciones de caso de uso – diseño. Las clases del análisis se completan, refinan (se eliminan las innecesarias y se complementa las que se conservan) y reorganizan para que lleguen a ser clases del diseño.

6.1 CLASES DEL DISEÑO

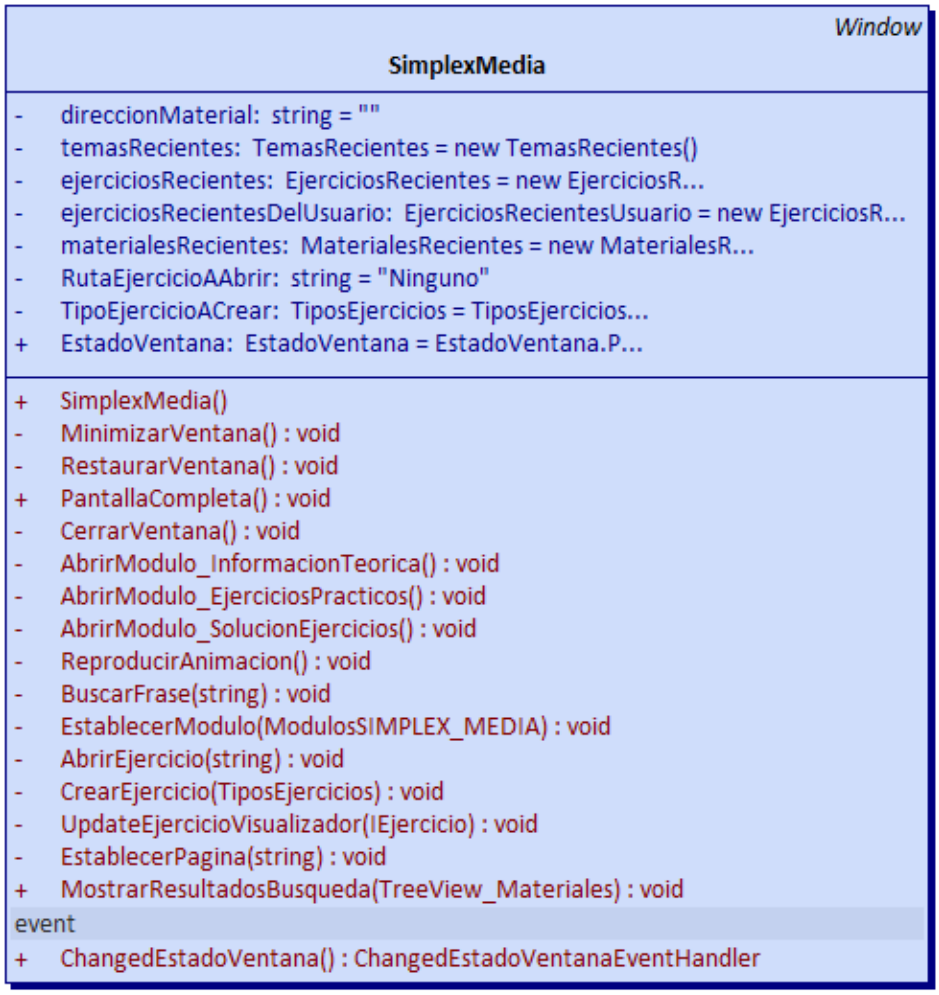
Las clases del diseño son las clases definitivas del proyecto. En los diagramas de clases se detallarán todos los atributos y métodos que aparecerán en el código fuente en la implementación.

Al describirlas se ha tenido en cuenta los lenguajes de programación que se usan para desarrollar SÍMPLEX MEDIA, C# .NET y XAML, por ello muchas de ellas estarán estereotipadas y definidas de usando palabras reservadas propias de los lenguajes.

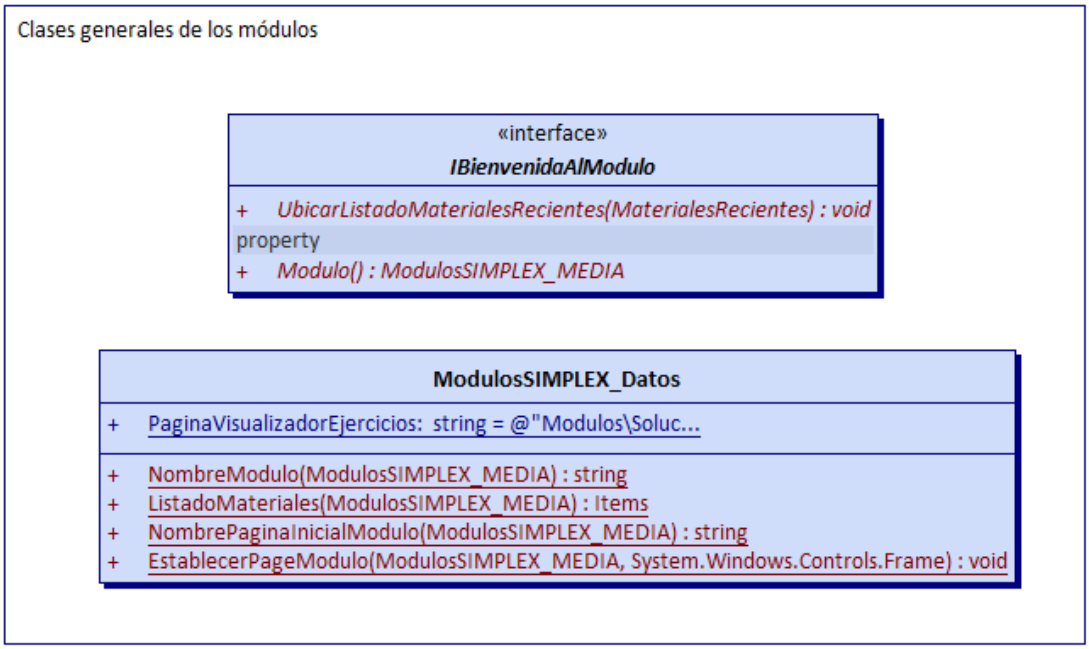
6.1.1 Listado de clases del diseño

6.1.1.1 Clase principal de SÍMPLEX MEDIA

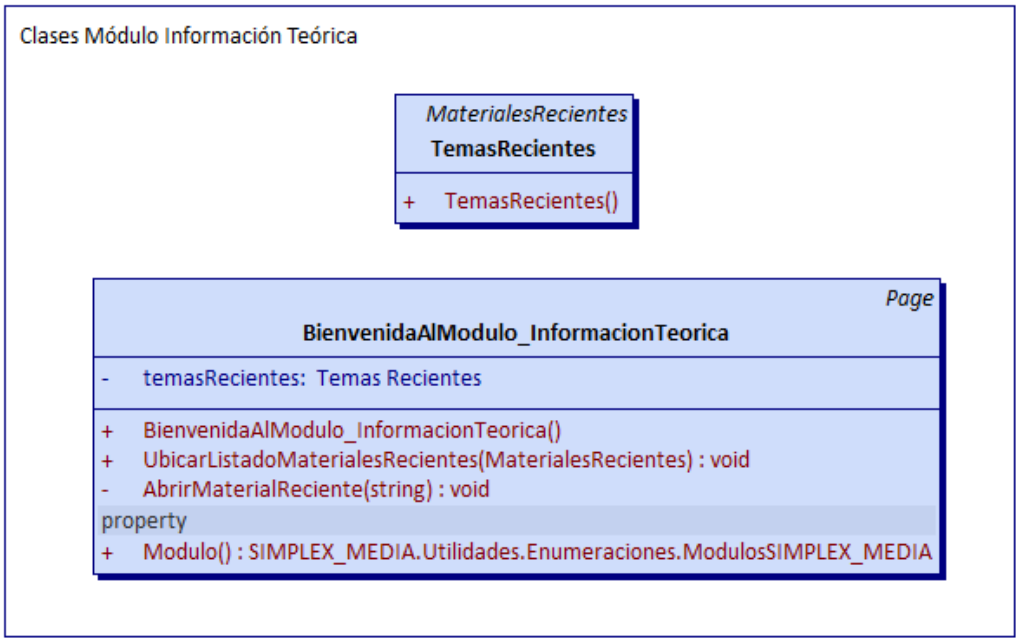
Clase de la Interfaz de Usuario Principal



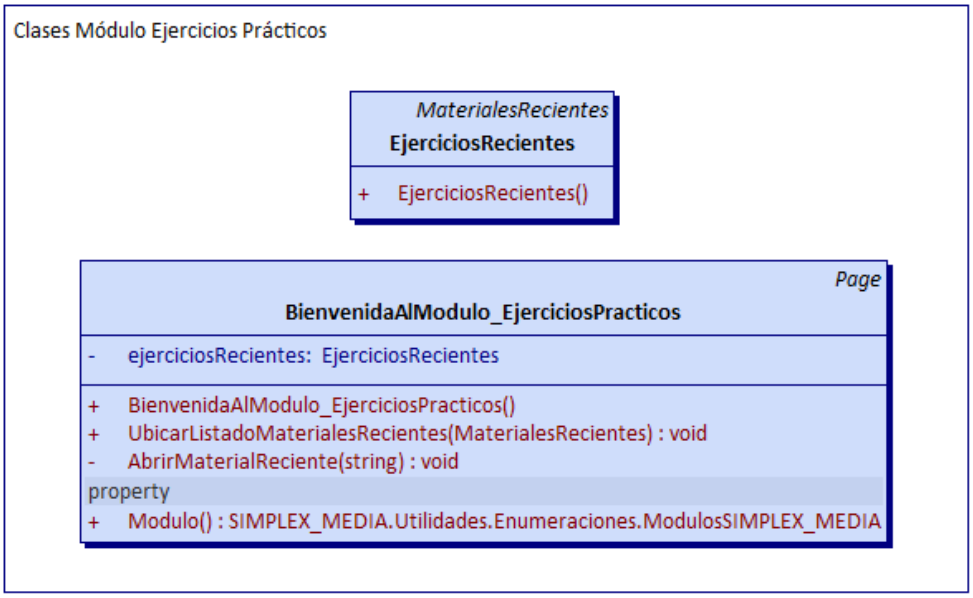
6.1.1.2 Clase generales de los módulos



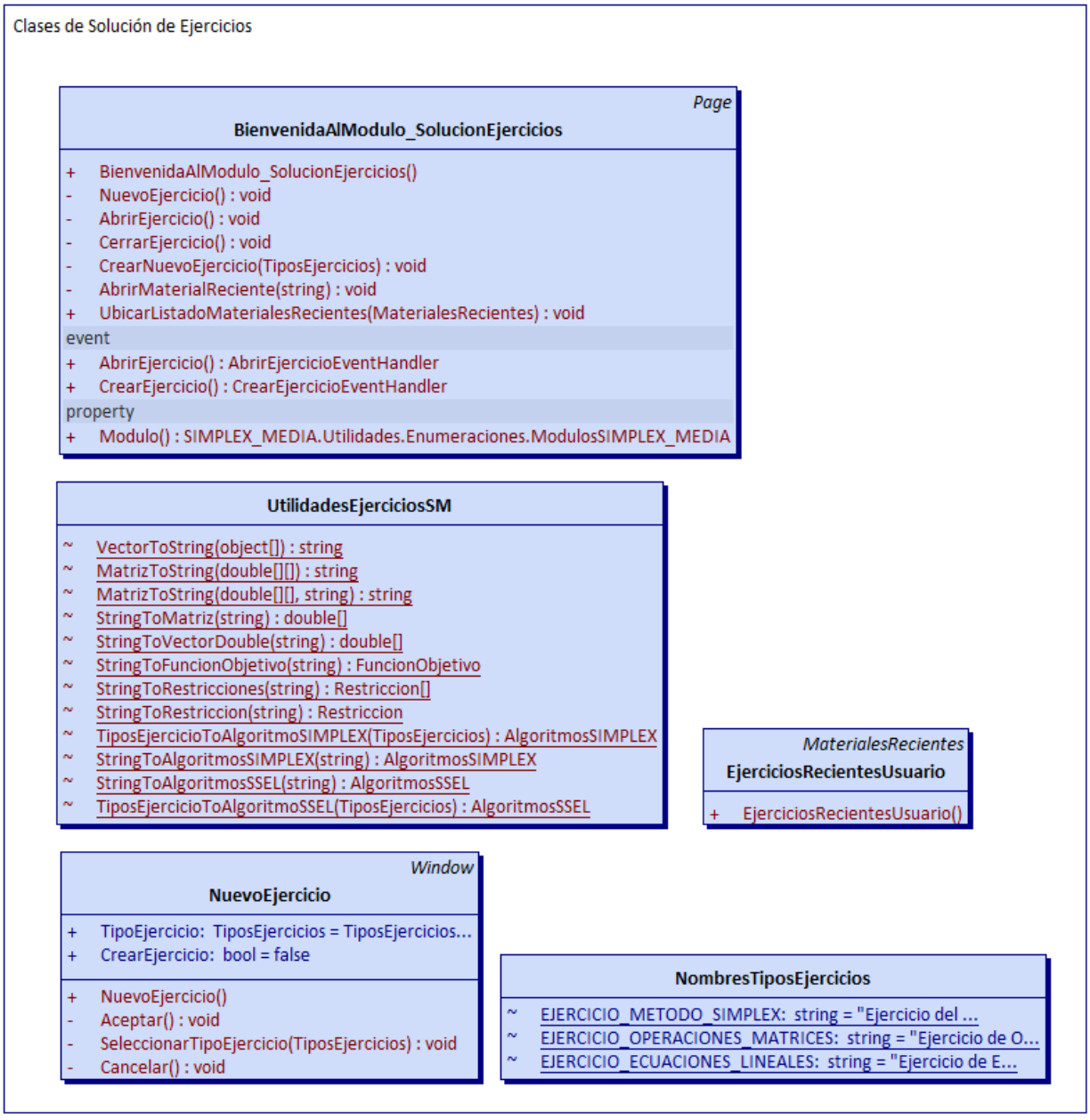
6.1.1.3 Módulo información teórica



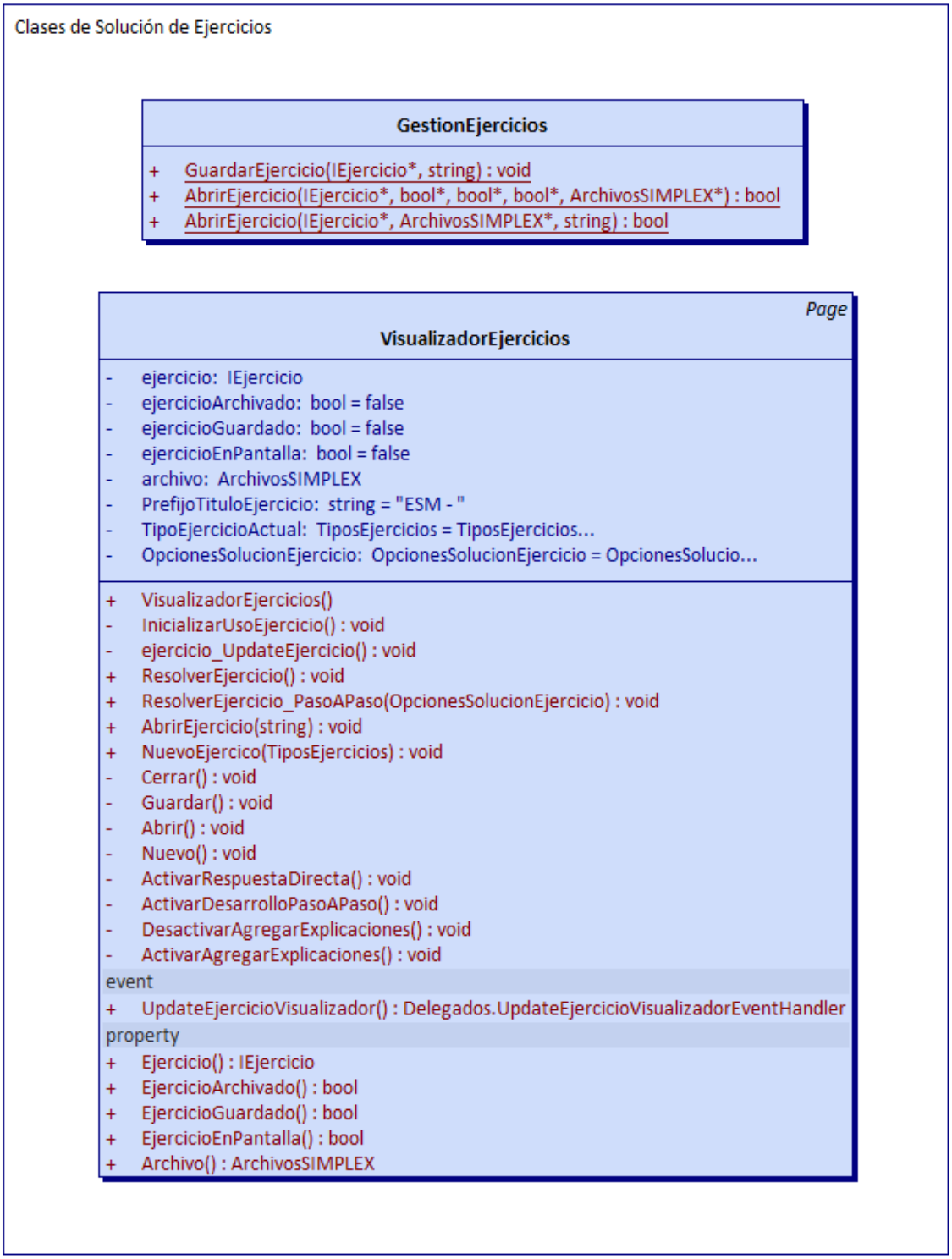
6.1.1.4 Módulo ejercicios prácticos

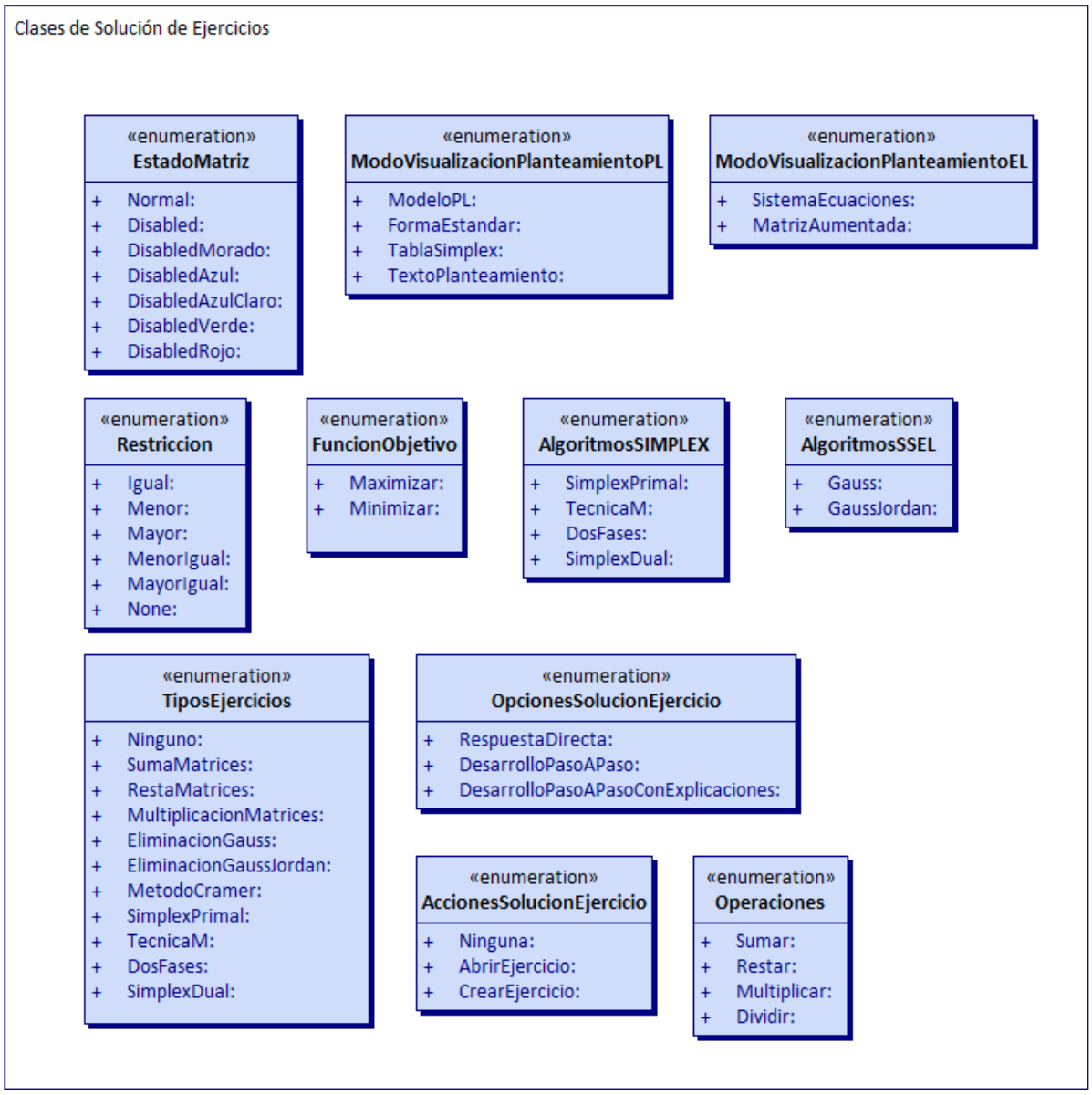


6.1.1.5 Módulo solución de ejercicios

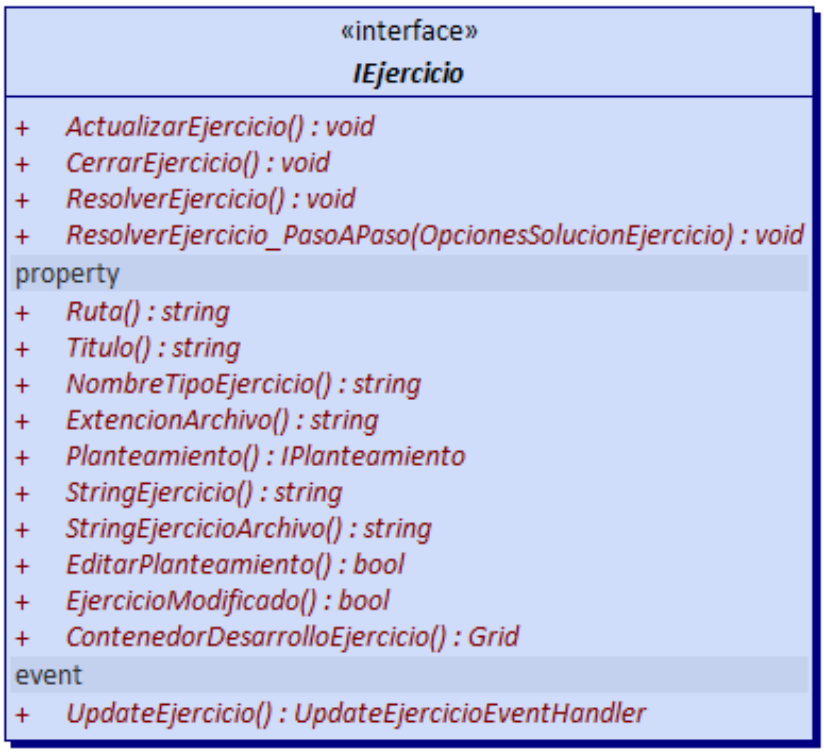


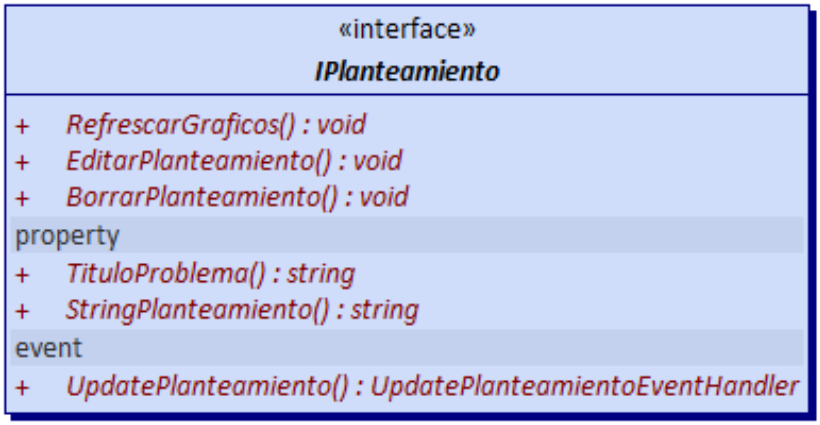
Clases de Solución de Ejercicios



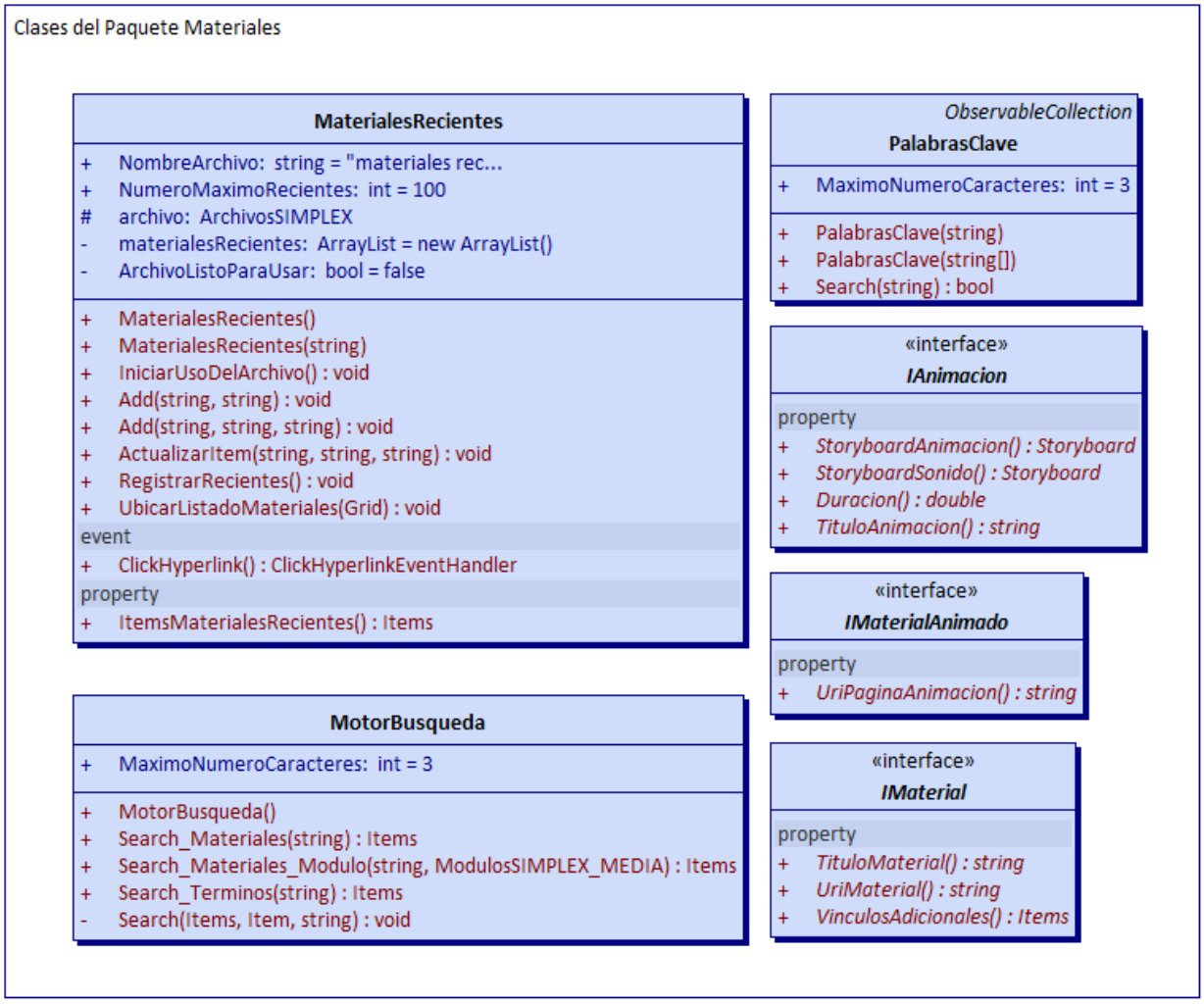

Clases de Solución de Ejercicios
«enumeration»
EstadoMatriz
+ Normal:
+ Disabled:
+ DisabledMorado:
+ DisabledAzul:
+ DisabledAzulClaro:
+ DisabledVerde:
+ DisabledRojo:
«enumeration»
ModoVisualizacionPlanteamientoPL
+ ModeloPL:
+ FormaEstandar:
+ TablaSimplex:
+ TextoPlanteamiento:
«enumeration»
ModoVisualizacionPlanteamientoEL
+ SistemaEcuaciones:
+ MatrizAumentada:
«enumeration»
Restriccion
+ Igual:
+ Menor:
+ Mayor:
+ MenorIgual:
+ MayorIgual:
+ None:
«enumeration»
FuncionObjetivo
+ Maximizar:
+ Minimizar:
«enumeration»
AlgoritmosSIMPLEX
+ SimplexPrimal:
+ TecnicaM:
+ DosFases:
+ SimplexDual:
«enumeration»
AlgoritmosSSEL
+ Gauss:
+ GaussJordan:
«enumeration»
TiposEjercicios
+ Ninguno:
+ SumaMatrices:
+ RestaMatrices:
+ MultiplicacionMatrices:
+ EliminacionGauss:
+ EliminacionGaussJordan:
+ MetodoCramer:
+ SimplexPrimal:
+ TecnicaM:
+ DosFases:
+ SimplexDual:
«enumeration»
OpcionesSolucionEjercicio
+ RespuestaDirecta:
+ DesarrolloPasoAPaso:
+ DesarrolloPasoAPasoConExplicaciones:
«enumeration»
AccionesSolucionEjercicio
+ Ninguna:
+ AbrirEjercicio:
+ CrearEjercicio:
«enumeration»
Operaciones
+ Sumar:
+ Restar:
+ Multiplicar:
+ Dividir:

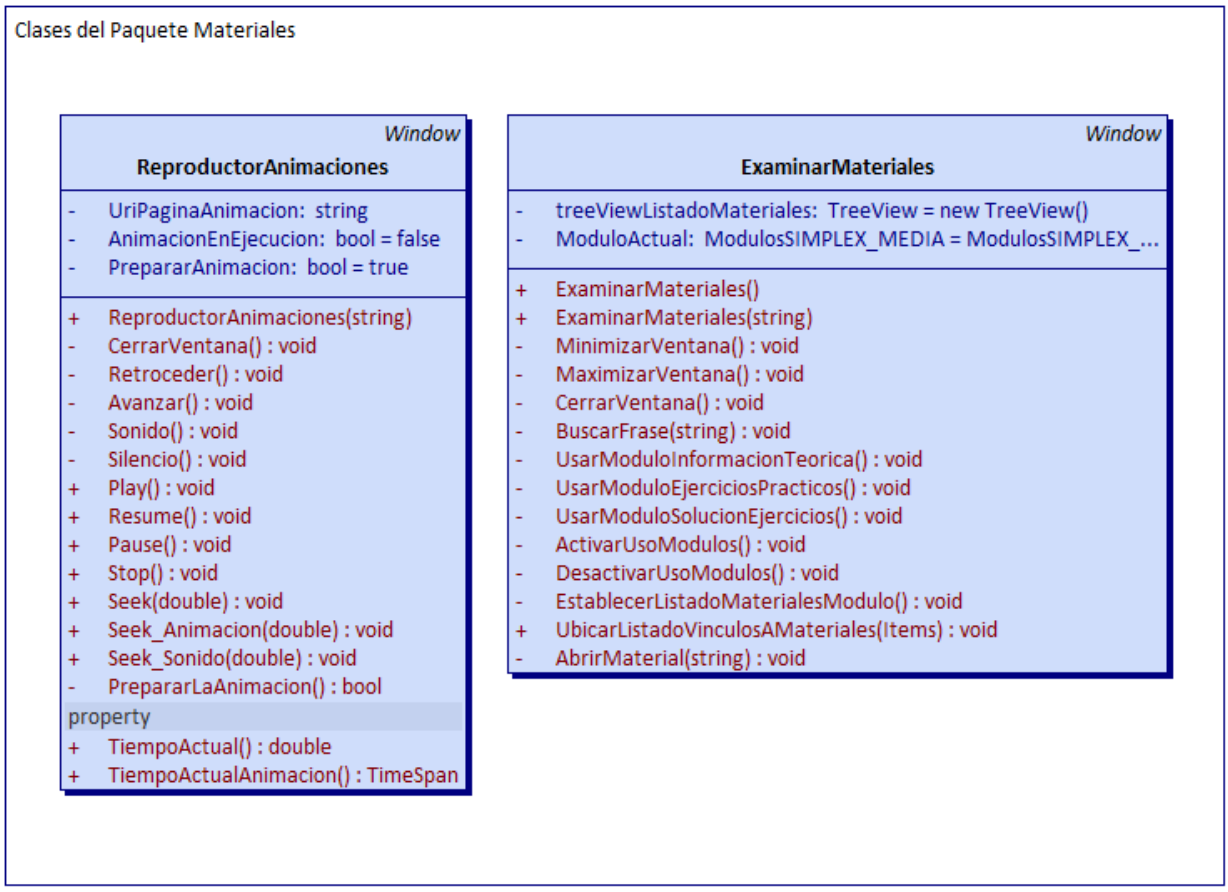
Interfaces de Solución de Ejercicios

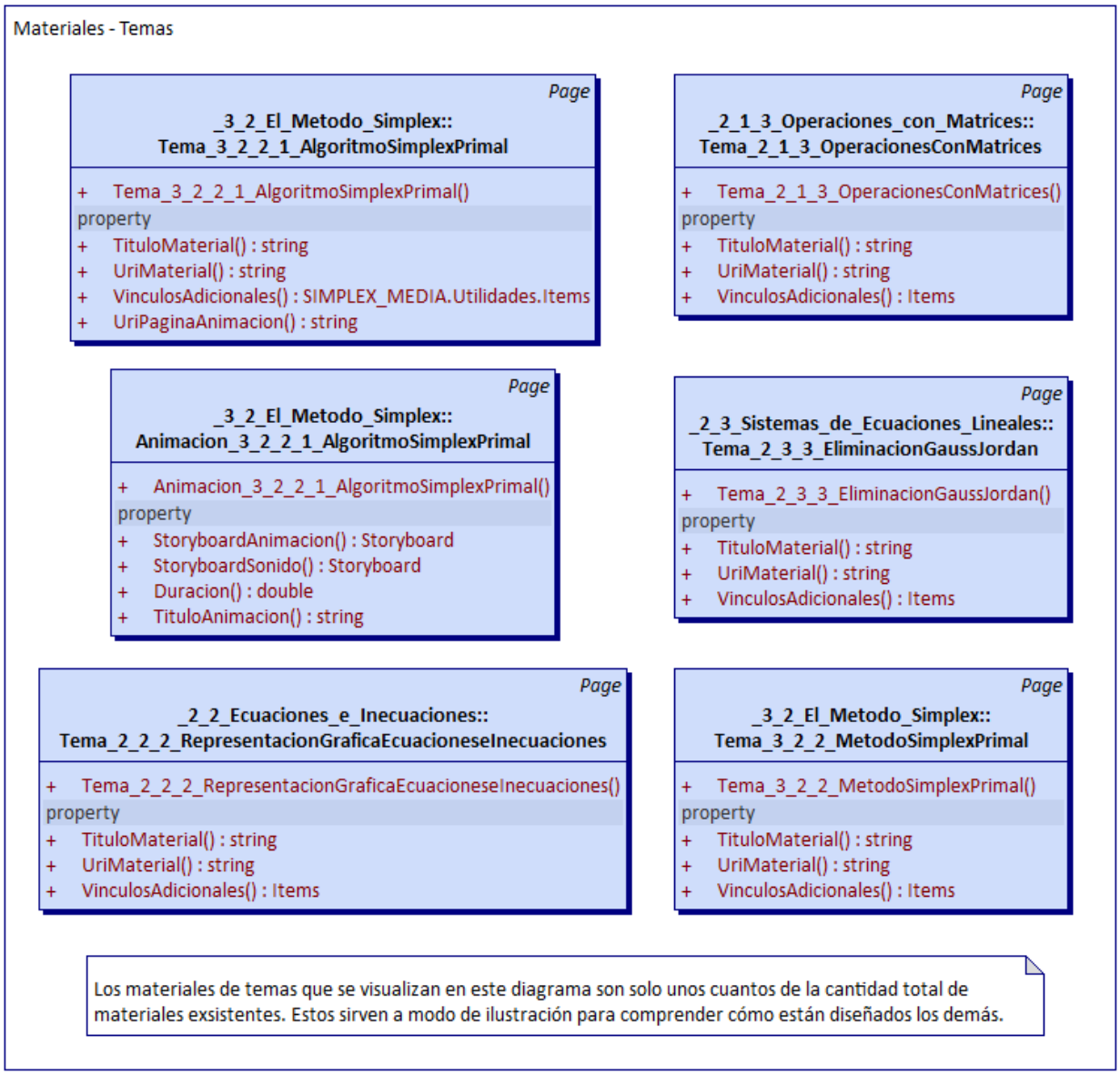


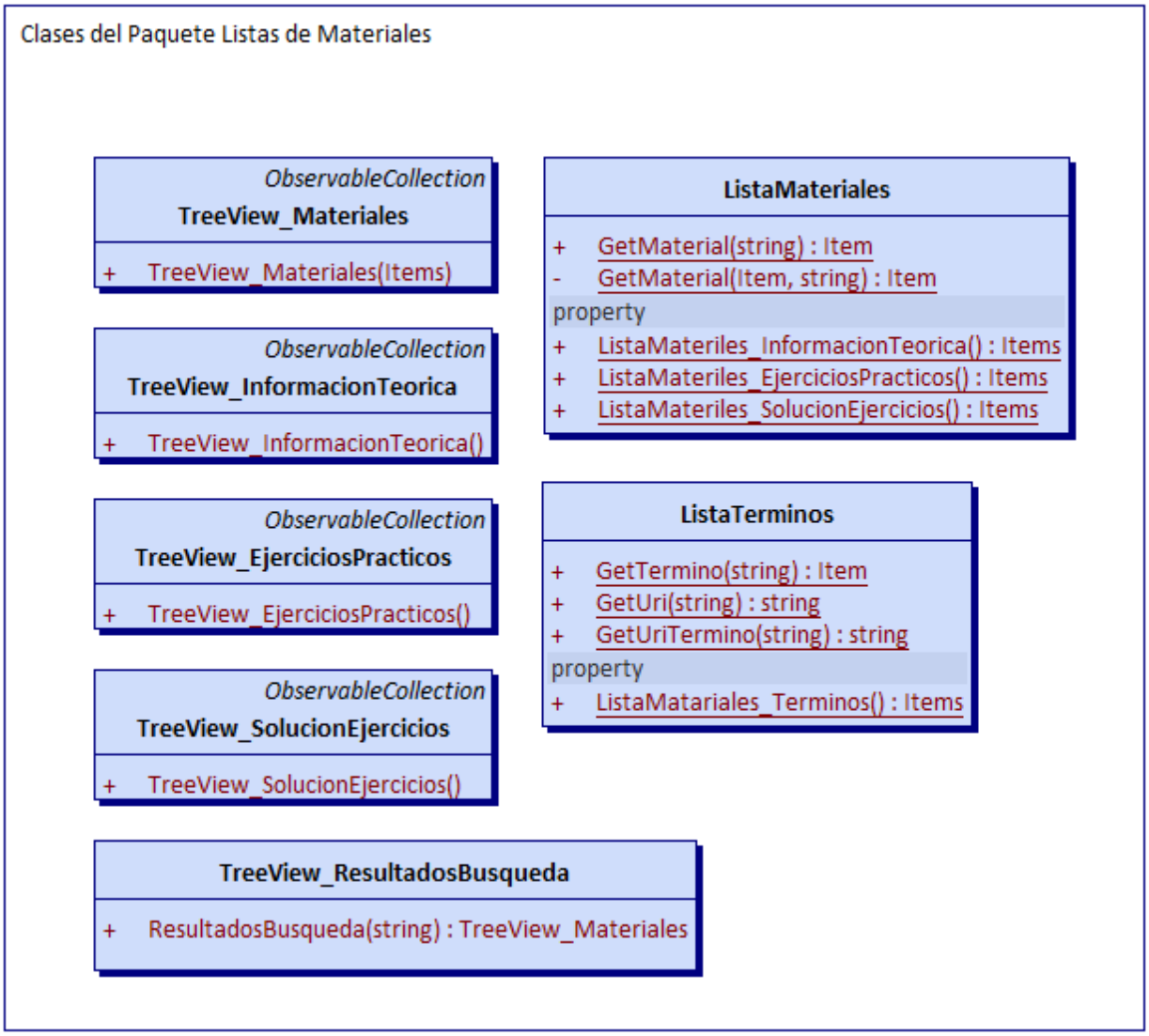


6.1.1.6 Materiales

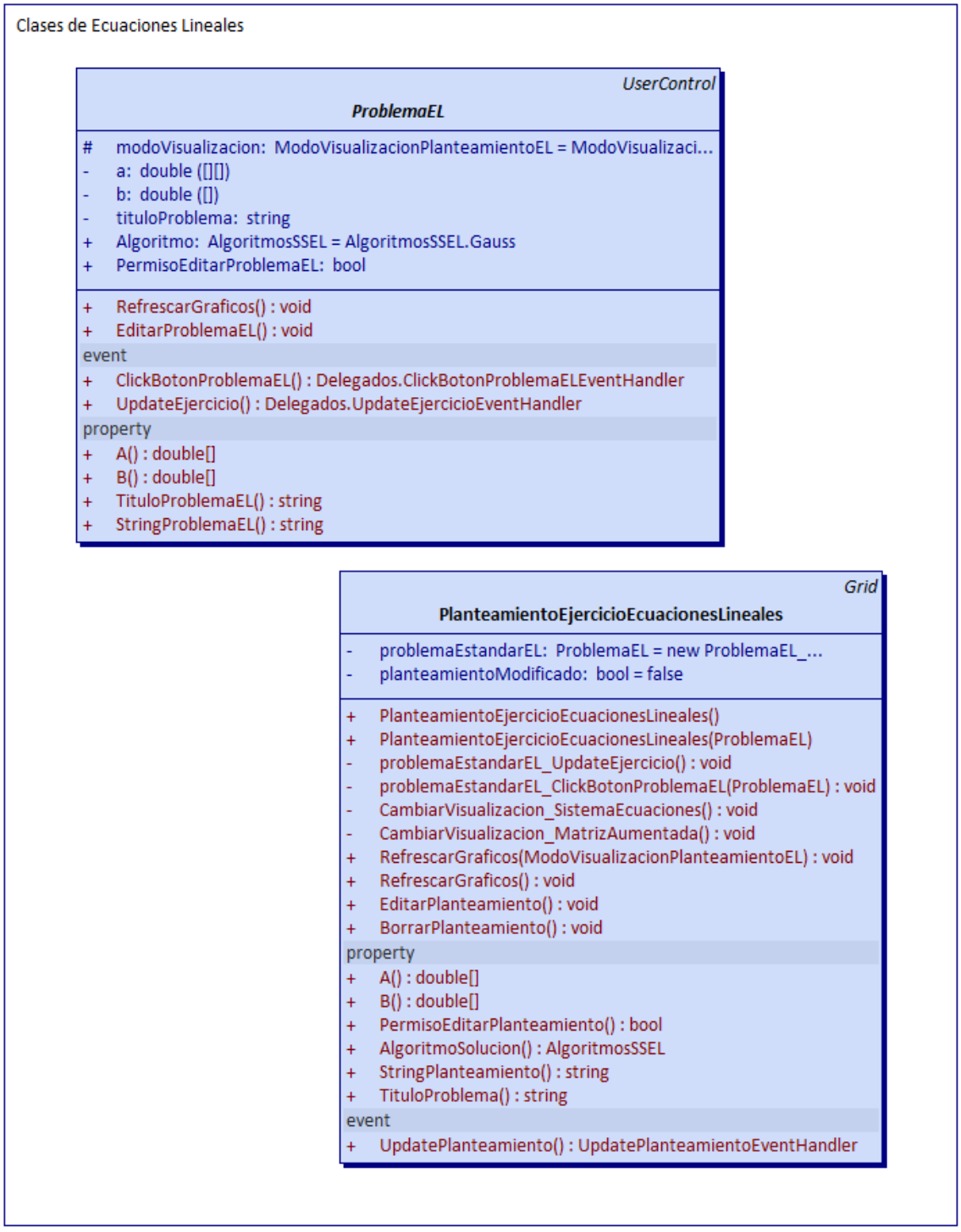



Clases del Paquete Materiales
Window
ReproductorAnimaciones
- UriPaginaAnimacion: string
- AnimacionEnEjecucion: bool = false
- PrepararAnimacion: bool = true
+ ReproductorAnimaciones(string)
- CerrarVentana() : void
- Retroceder() : void
- Avanzar() : void
- Sonido() : void
- Silencio() : void
+ Play() : void
+ Resume() : void
+ Pause() : void
+ Stop() : void
+ Seek(double) : void
+ Seek_Animacion(double) : void
+ Seek_Sonido(double) : void
- PrepararLaAnimacion() : bool
property
+ TiempoActual() : double
+ TiempoActualAnimacion() : TimeSpan
Window
ExaminarMateriales
- treeViewListadoMateriales: TreeView = new TreeView()
- ModuloActual: ModulosSIMPLEX_MEDIA = ModulosSIMPLEX_...
+ ExaminarMateriales()
+ ExaminarMateriales(string)
- MinimizarVentana() : void
- MaximizarVentana() : void
- CerrarVentana() : void
- BuscarFrase(string) : void
- UsarModuloInformacionTeorica() : void
- UsarModuloEjerciciosPracticos() : void
- UsarModuloSolucionEjercicios() : void
- ActivarUsoModulos() : void
- DesactivarUsoModulos() : void
- EstablecerListadoMaterialesModulo() : void
+ UbicarListadoVinculosAMateriales(Items) : void
- AbrirMaterial(string) : void

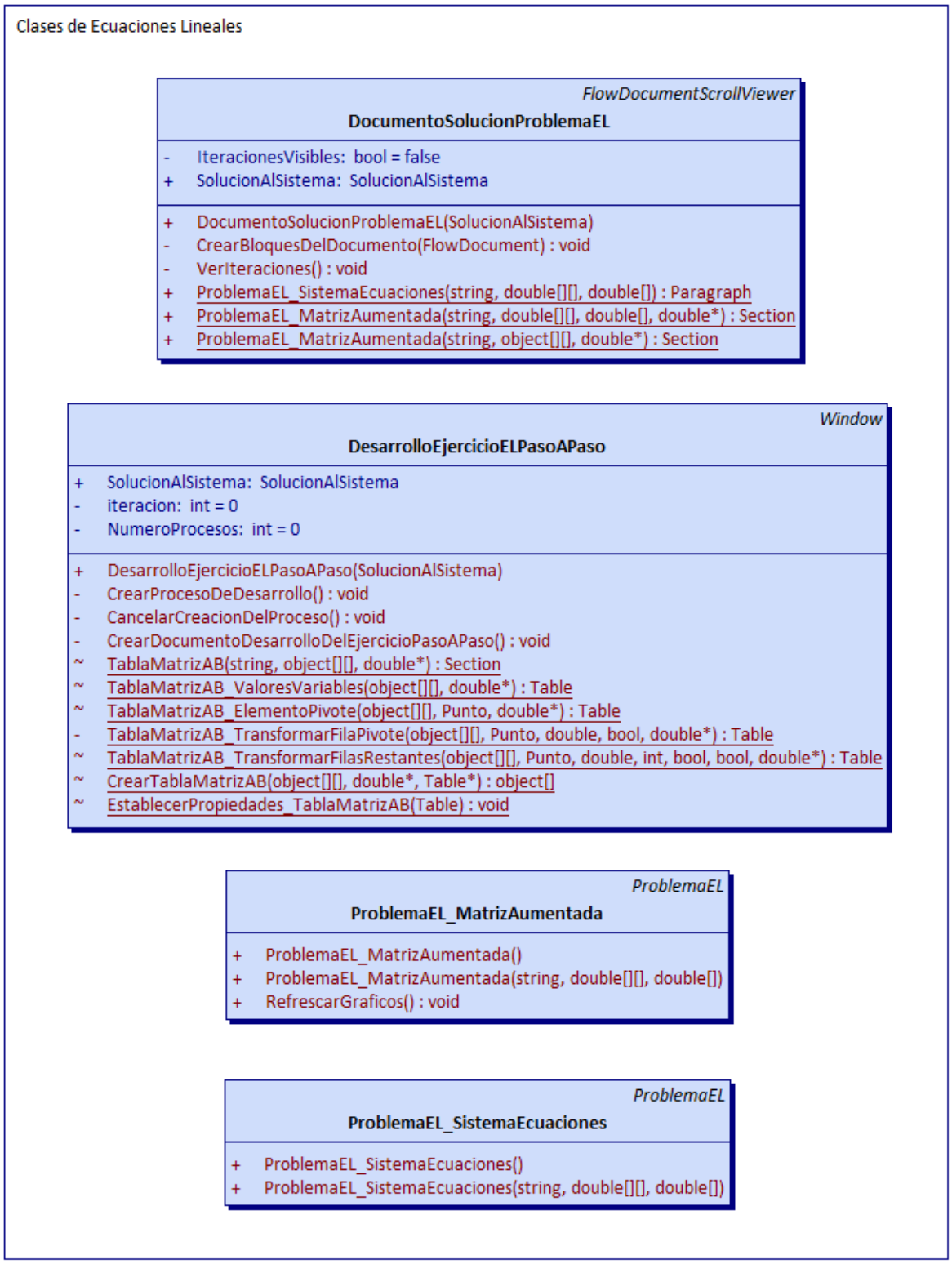

Materiales - Temas
Page
_3_2_El_Metodo_Simplex::
Tema_3_2_2_1_AlgoritmoSimplexPrimal
+ Tema_3_2_2_1_AlgoritmoSimplexPrimal()
property
+ TituloMaterial() : string
+ UriMaterial() : string
+ VinculosAdicionales() : SIMPLEX_MEDIA.Utilidades.Items
+ UriPaginaAnimacion() : string
Page
_2_1_3_Operaciones_con_Matrices::
Tema_2_1_3_OperacionesConMatrices
+ Tema_2_1_3_OperacionesConMatrices()
property
+ TituloMaterial() : string
+ UriMaterial() : string
+ VinculosAdicionales() : Items
Page
_3_2_El_Metodo_Simplex::
Animacion_3_2_2_1_AlgoritmoSimplexPrimal
+ Animacion_3_2_2_1_AlgoritmoSimplexPrimal()
property
+ StoryboardAnimacion() : Storyboard
+ StoryboardSonido() : Storyboard
+ Duracion() : double
+ TituloAnimacion() : string
Page
_2_3_Sistemas_de_Ecuaciones_Lineales::
Tema_2_3_3_EliminacionGaussJordan
+ Tema_2_3_3_EliminacionGaussJordan()
property
+ TituloMaterial() : string
+ UriMaterial() : string
+ VinculosAdicionales() : Items
Page
_2_2_Ecuaciones_e_Inecuaciones::
Tema_2_2_2_RepresentacionGraficaEcuacioneseInecuaciones
+ Tema_2_2_2_RepresentacionGraficaEcuacioneseInecuaciones()
property
+ TituloMaterial() : string
+ UriMaterial() : string
+ VinculosAdicionales() : Items
Page
_3_2_El_Metodo_Simplex::
Tema_3_2_2_MetodoSimplexPrimal
+ Tema_3_2_2_MetodoSimplexPrimal()
property
+ TituloMaterial() : string
+ UriMaterial() : string
+ VinculosAdicionales() : Items
Los materiales de temas que se visualizan en este diagrama son solo unos cuantos de la cantidad total de materiales exsistentes. Estos sirven a modo de ilustración para comprender cómo están diseñados los demás.


Clases del Paquete Listas de Materiales
ObservableCollection
TreeView_Materiales
+ TreeView_Materiales(Items)
ObservableCollection
TreeView_InformacionTeorica
+ TreeView_InformacionTeorica()
ObservableCollection
TreeView_EjerciciosPracticos
+ TreeView_EjerciciosPracticos()
ObservableCollection
TreeView_SolucionEjercicios
+ TreeView_SolucionEjercicios()
TreeView_ResultadosBusqueda
+ ResultadosBusqueda(string) : TreeView_Materiales
ListaMateriales
+ GetMaterial(string) : Item
- GetMaterial(Item, string) : Item
property
+ ListaMateriles_InformacionTeorica() : Items
+ ListaMateriles_EjerciciosPracticos() : Items
+ ListaMateriles_SolucionEjercicios() : Items
ListaTerminos
+ GetTermino(string) : Item
+ GetUri(string) : string
+ GetUriTermino(string) : string
property
+ ListaMatariales_Terminos() : Items

6.1.1.7 Ecuaciones lineales



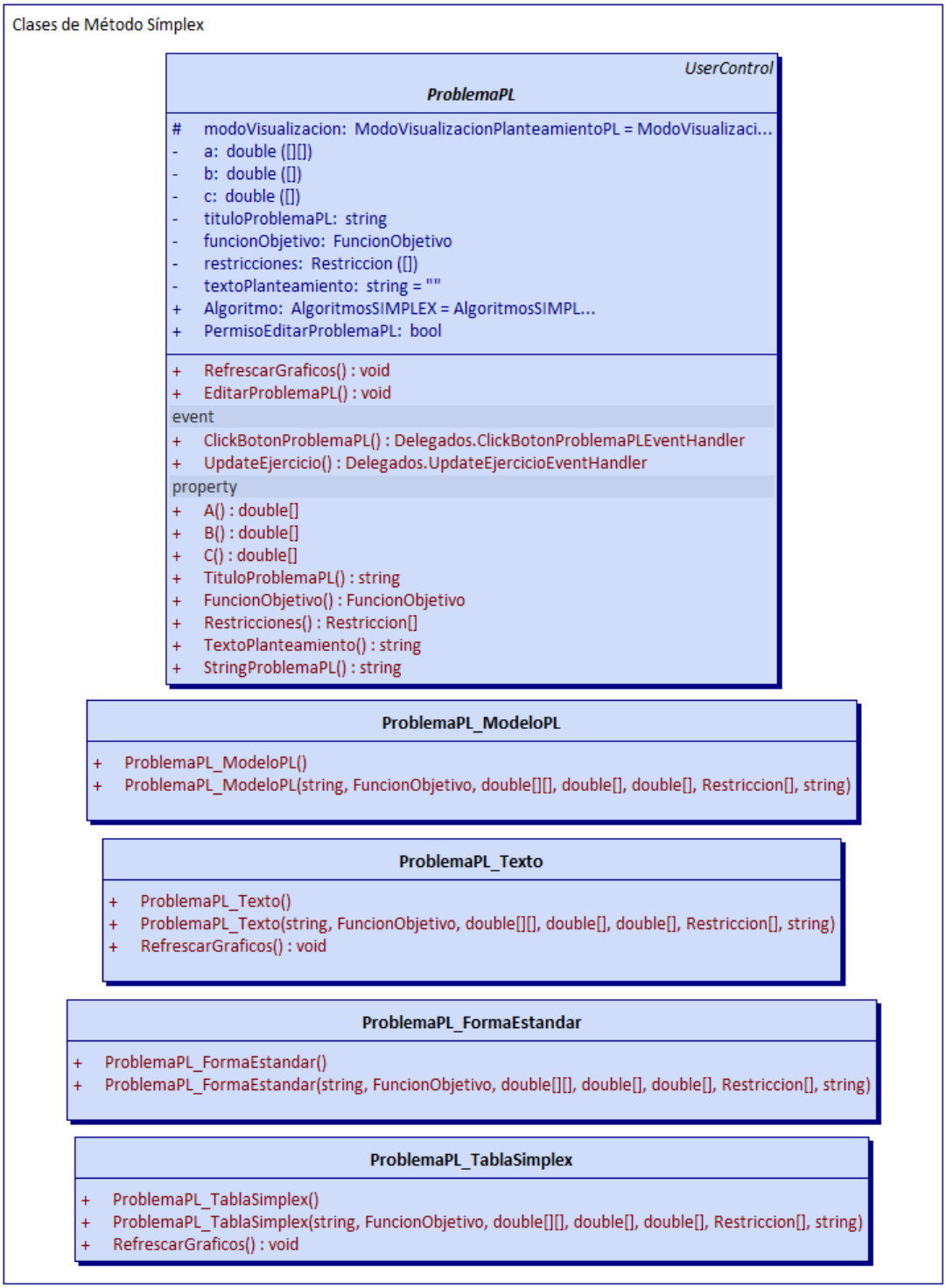

Clases de Ecuaciones Lineales
EjercicioEcuacionesLineales
+ contenedorDesarrolloEjercicio: Grid
- planteamientoEjercicio: IPlanteamiento
- rura_ejercicio: string = ""
- nombre_ejercicio: string = "Ejercicio Ecua...
- editarPlanteamiento: bool = true
- ejercicioModificado: bool = false
+ EjercicioEcuacionesLineales()
- planteamientoEjercicio_UpdatePlanteamiento(bool) : void
+ ResolverEjercicio() : void
+ ResolverEjercicio_PasoAPaso(OpcionesSolucionEjercicio) : void
+ CerrarEjercicio() : void
+ ActualizarEjercicio() : void
property
+ AlgoritmoSolucion() : AlgoritmosSSEL
+ Ruta() : string
+ Titulo() : string
+ NombreTipoEjercicio() : string
+ ExtencionArchivo() : string
+ Planteamiento() : IPlanteamiento
+ StringEjercicio() : string
+ StringEjercicioArchivo() : string
+ EditarPlanteamiento() : bool
+ EjercicioModificado() : bool
+ ContenedorDesarrolloEjercicio() : Grid
event
+ UpdateEjercicio() : UpdateEjercicioEventHandler
Window
EdicionProblemaEL
+ Algoritmo: AlgoritmosSSEL = AlgoritmosSSEL.Gauss
- ultimoTextoCelda: double = 0
+ TituloProblemaEL: string = "Problema EL"
- a: double ([][]) = new double[][]{...
- b: double ([]) = new double[] { ...
- problemaModificado: bool = false
+ EdicionProblemaEL()
+ EdicionProblemaEL(string, double[][], double[], AlgoritmosSSEL)
- ModificarTituloProblema(string) : void
- AdicionarColumna() : void
- RemoverUltimaColumna() : void
- AdicionarFila() : void
- RemoverUltimaFila() : void
- CambiarAlgoritmo(AlgoritmosSSEL) : void
- CerrarVentana() : void
+ ClonarProblemaEL(ProblemaEL) : bool
+ ModificarValorVariable(double, Variable) : void
+ RefrescarGraficos() : void
property
+ A() : double[]
+ B() : double[]


Clases de Ecuaciones Lineales
FlowDocumentScrollViewer
DocumentoSolucionProblemaEL
- IteracionesVisibles: bool = false
+ SolucionAlSistema: SolucionAlSistema
+ DocumentoSolucionProblemaEL(SolucionAlSistema)
- CrearBloquesDelDocumento(FlowDocument) : void
- VerIteraciones() : void
+ ProblemaEL_SistemaEcuaciones(string, double[][], double[]) : Paragraph
+ ProblemaEL_MatrizAumentada(string, double[][], double[], double*) : Section
+ ProblemaEL_MatrizAumentada(string, object[][], double*) : Section
Window
DesarrolloEjercicioELPasoAPaso
+ SolucionAlSistema: SolucionAlSistema
- iteracion: int = 0
- NumeroProcesos: int = 0
+ DesarrolloEjercicioELPasoAPaso(SolucionAlSistema)
- CrearProcesoDeDesarrollo() : void
- CancelarCreacionDelProceso() : void
- CrearDocumentoDesarrolloDelEjercicioPasoAPaso() : void
~ TablaMatrizAB(string, object[][], double*) : Section
~ TablaMatrizAB_ValoresVariables(object[][], double*) : Table
~ TablaMatrizAB_ElementoPivote(object[][], Punto, double*) : Table
- TablaMatrizAB_TransformarFilaPivote(object[][], Punto, double, bool, double*) : Table
~ TablaMatrizAB_TransformarFilasRestantes(object[][], Punto, double, int, bool, bool, double*) : Table
~ CrearTablaMatrizAB(object[][], double*, Table*) : object[]
~ EstablecerPropiedades_TablaMatrizAB(Table) : void
ProblemaEL
ProblemaEL_MatrizAumentada
+ ProblemaEL_MatrizAumentada()
+ ProblemaEL_MatrizAumentada(string, double[][], double[])
+ RefrescarGraficos() : void
ProblemaEL
ProblemaEL_SistemaEcuaciones
+ ProblemaEL_SistemaEcuaciones()
+ ProblemaEL_SistemaEcuaciones(string, double[][], double[])

6.1.1.8 Operaciones con matrices


Clases del Paquete Matrices
UserControl
BotonIgual_Matrices
+ BotonIgual_Matrices()
- EjecutarSolucionProblema() : void
event
+ ResolverEjercicioMatrices() : ResolverEjercicioMatricesEventHandler
EjercicioMatrices
+ contenedorDesarrolloEjercicio: Grid
- operacionInicial: Operaciones = Operaciones.Sumar
- matrizResultado: Matriz
- igual: BotonIgual_Matrices = new BotonIgual_...
- planteamientoEjercicio: PlanteamientoEjerciciosMatrices
- rura_ejercicio: string = ""
- nombre_ejercicio: string = "EjercicioMatrices"
- editarPlanteamiento: bool = true
- ejercicioModificado: bool = false
+ EjercicioMatrices()
- planteamientoEjercicio_UpdatePlanteamiento(bool) : void
- AdicionarMatriz() : void
- RemoverMatriz() : void
+ CerrarEjercicio() : void
+ ActualizarEjercicio() : void
+ ResolverEjercicio() : void
+ ResolverEjercicio_PasoAPaso(OpcionesSolucionEjercicio) : void
property
+ Igual() : BotonIgual_Matrices
+ Ruta() : string
+ Titulo() : string
+ NombreTipoEjercicio() : string
+ ExtencionArchivo() : string
+ Planteamiento() : IPlanteamiento
+ ContenedorDesarrolloEjercicio() : Grid
+ StringEjercicio() : string
+ StringEjercicioArchivo() : string
+ EditarPlanteamiento() : bool
+ EjercicioModificado() : bool
+ OperacionInicial() : Operaciones
event
+ UpdateEjercicio() : SIMPLEX_MEDIA.Delegados.UpdateEjercicioEventHandler

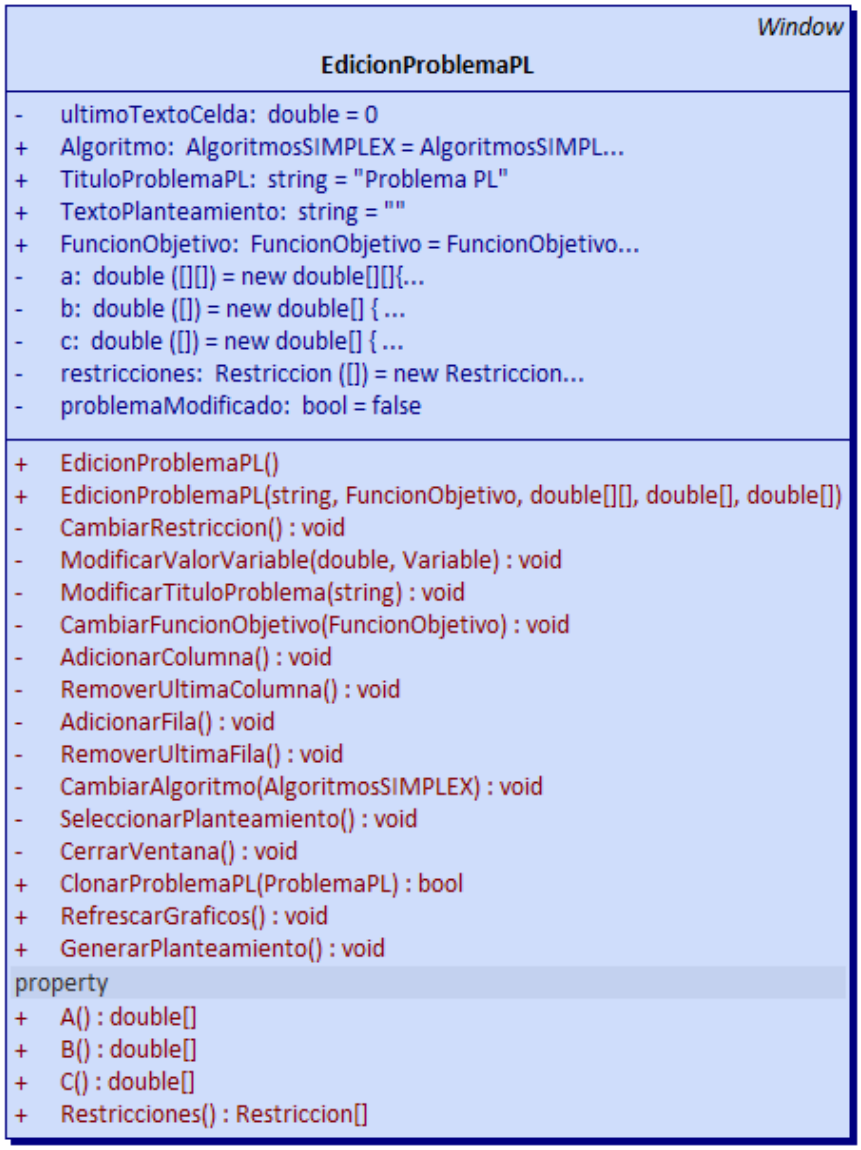
6.1.1.9 Método Símplex



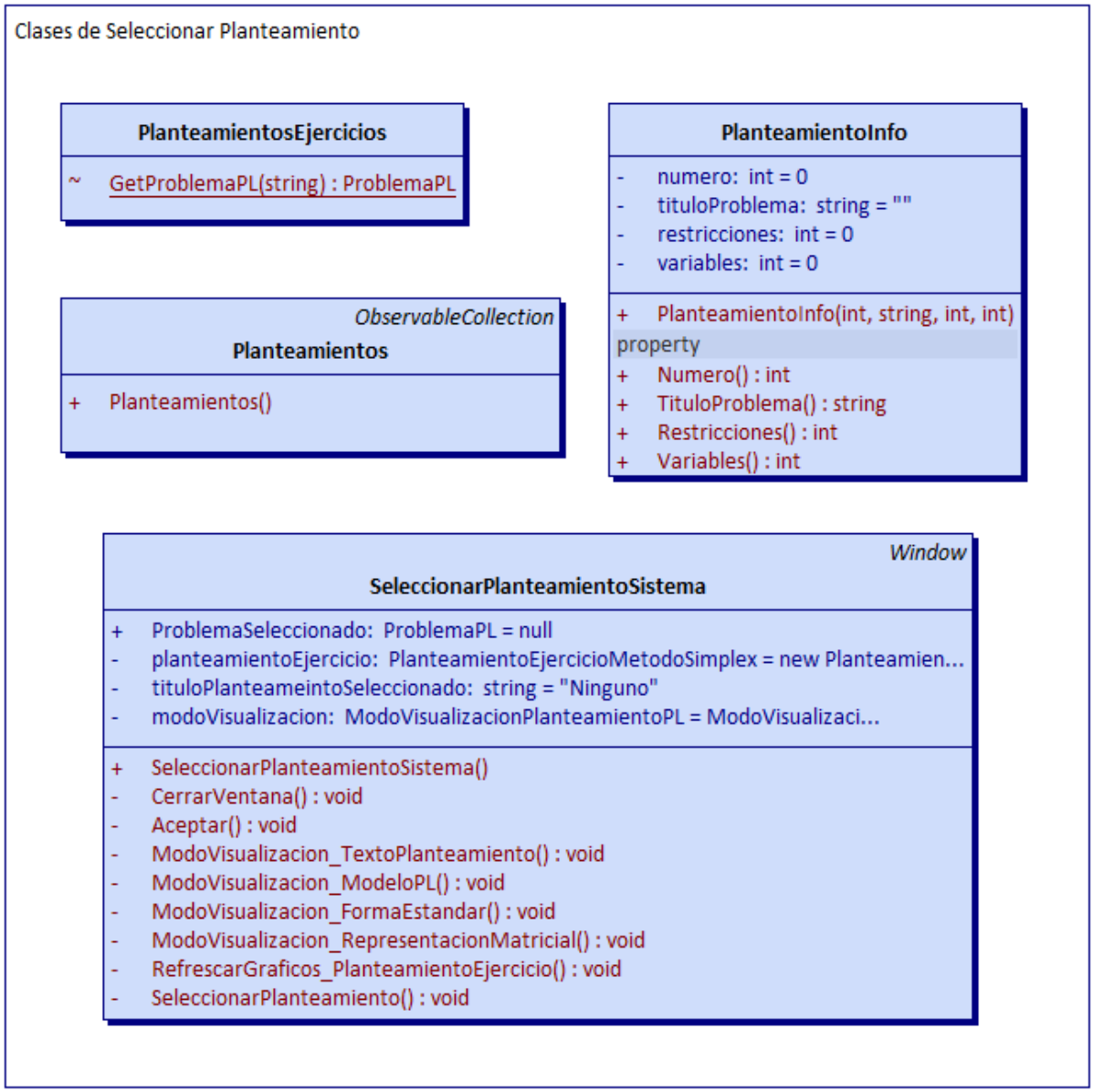

Clases de Método Símplex
EjercicioMetodoSimplex
+ contenedorDesarrolloEjercicio: Grid
- planteamientoEjercicio: IPlanteamiento
- rura_ejercicio: string = ""
- nombre_ejercicio: string = "Ejercicio Méto...
- editarPlanteamiento: bool = true
- ejercicioModificado: bool = false
+ EjercicioMetodoSimplex()
- planteamientoEjercicio_UpdatePlanteamiento(bool) : void
+ ResolverEjercicio() : void
+ ResolverEjercicio_PasoAPaso(OpcionesSolucionEjercicio) : void
+ CerrarEjercicio() : void
+ ActualizarEjercicio() : void
property
+ AlgoritmoSolucion() : AlgoritmosSIMPLEX
+ Ruta() : string
+ Titulo() : string
+ NombreTipoEjercicio() : string
+ ExtencionArchivo() : string
+ Planteamiento() : IPlanteamiento
+ StringEjercicio() : string
+ StringEjercicioArchivo() : string
+ EditarPlanteamiento() : bool
+ EjercicioModificado() : bool
+ ContenedorDesarrolloEjercicio() : Grid
event
+ UpdateEjercicio() : UpdateEjercicioEventHandler
Window
DesarrolloEjercicioPLPasoAPaso
+ SolucionFactible: SolucionFactible
+ ProblemaPL: ProblemaSimplex
- iteracion: int = 0
- NumeroProcesos: int = 0
+ DesarrolloEjercicioPLPasoAPaso(SolucionFactible)
- CrearProcesoDeDesarrollo() : void
- CancelarCreacionDelProceso() : void
- SeccionDesarrolloDelEjercicioPasoAPaso(int) : void
- CrearParrafoFuncionObjetivo(string, object[], string[], double[], bool, bool, bool) : Paragraph
- CrearParrafoVariablesSolucionInicial(string[], double[]) : Paragraph
~ TablaSimplex(string, object[][], Block, double*) : Section
~ TablaSimplex_ValoresVariables(object[][], double*) : Table
~ TablaSimplex_VariableEntrada(object[][], FuncionObjetivo, bool, bool*, Punto, double*) : Table
~ TablaSimplex_VariableSalida(object[][], Punto, double*) : Table
~ TablaSimplex_ElementoPivote(object[][], Punto, double*) : Table
~ TablaSimplex_TransformarFilaPivote(object[][], Punto, double, bool, double*) : Table
~ TablaSimplex_TransformarFilasRestantes(object[][], Punto, double, int, bool, bool, double*) : Table
~ TablaSimplex_MultiplicarFilaAiPorM(object[][], int, double*, bool) : Table
~ CrearTablaSimplex(object[][], double*, Table*) : object[]
~ EstablecerPropiedades_TablaSimplex(Table) : void

Clases de Método Símplex

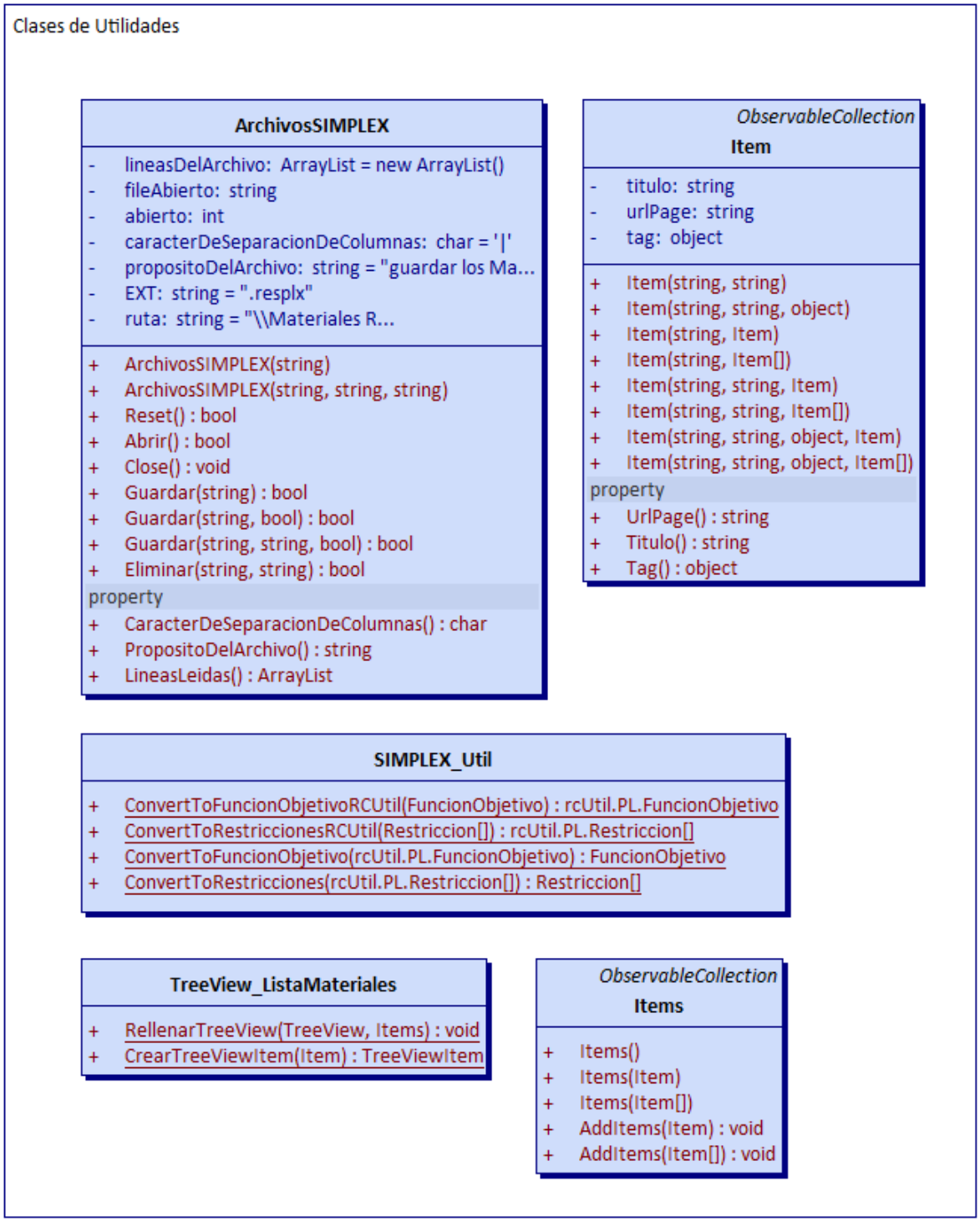
Clases de Método Símplex



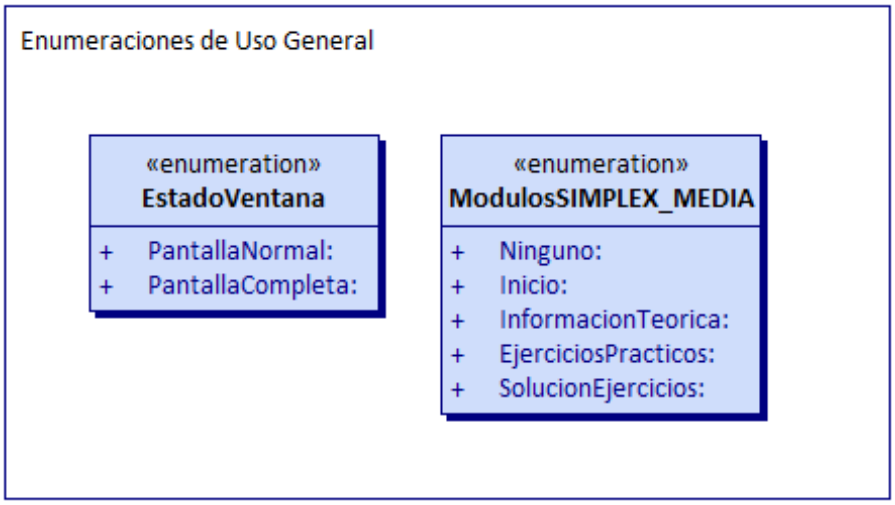
6.1.1.10 Seleccionar planteamiento



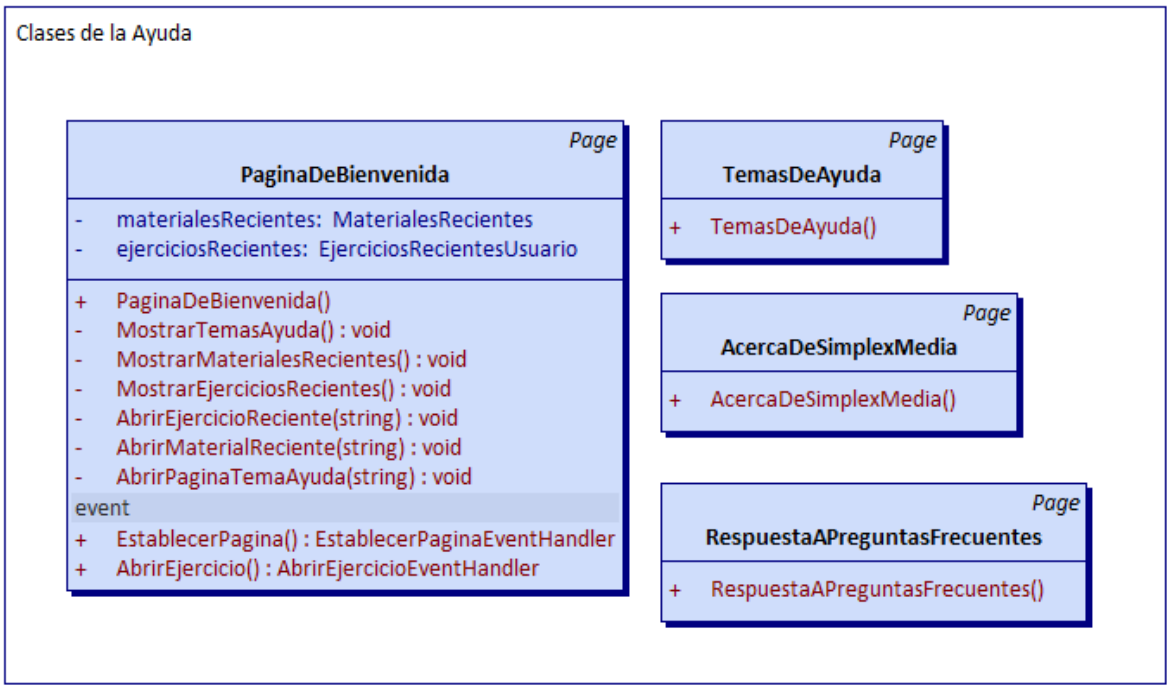
6.1.1.11 Utilidades



6.1.1.12 Enumeraciones generales

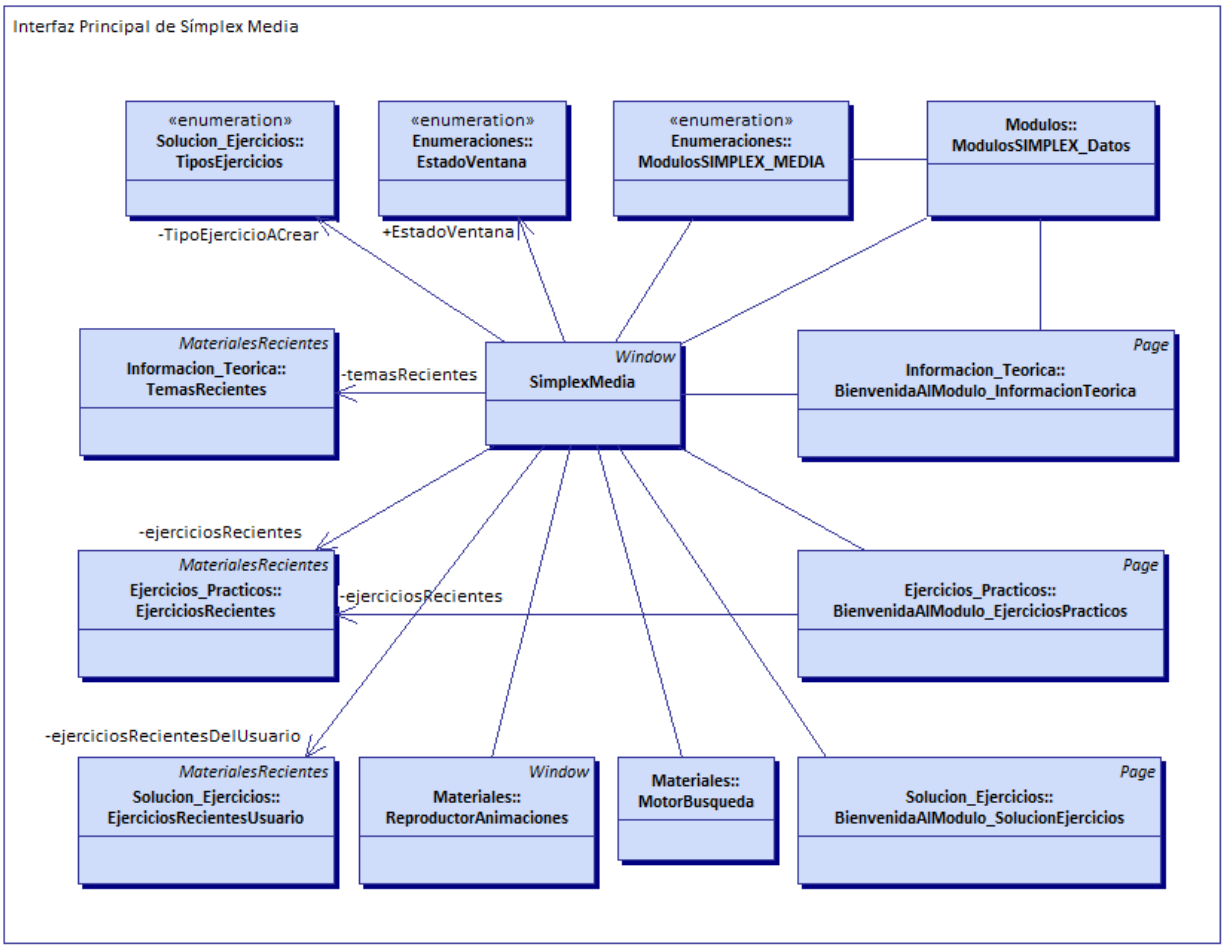


6.1.1.13 Página de bienvenida y ayuda

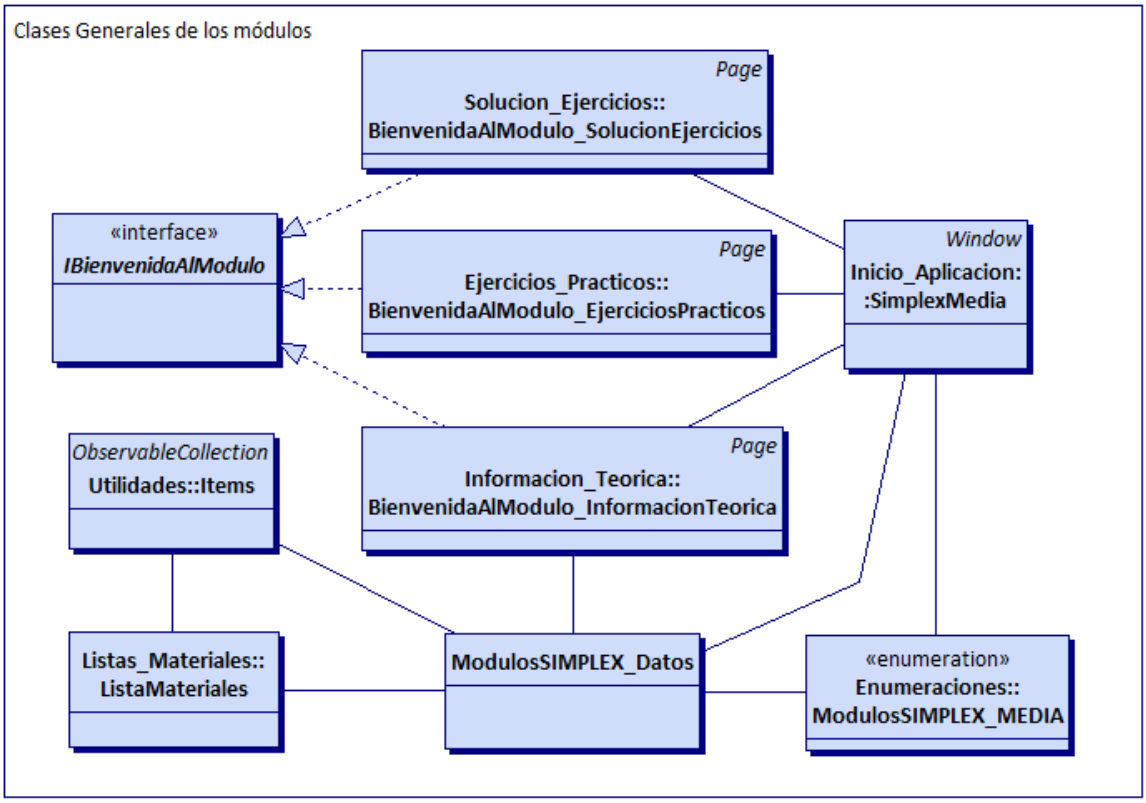


6.1.2 Diagramas de clases

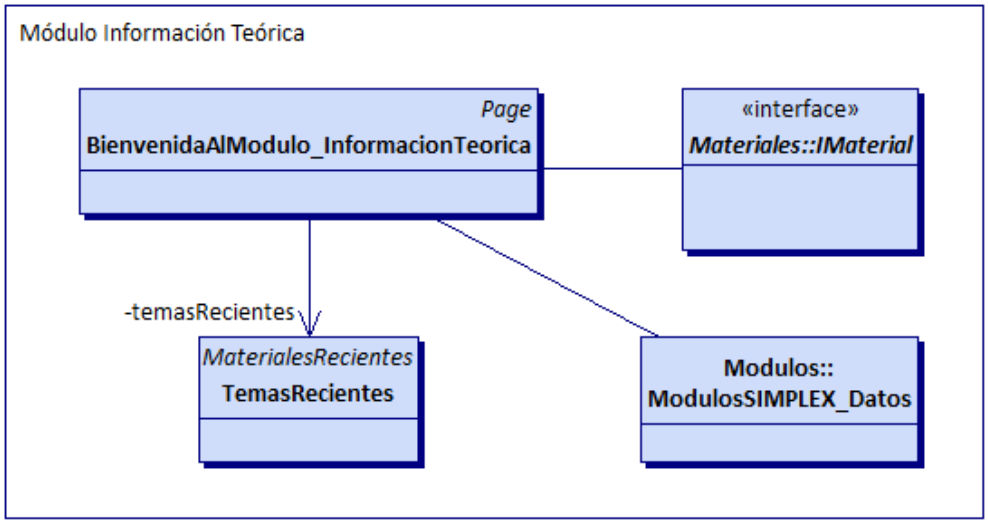
6.1.2.1 Diagrama: Interfaz principal de SÍMPLEX MEDIA



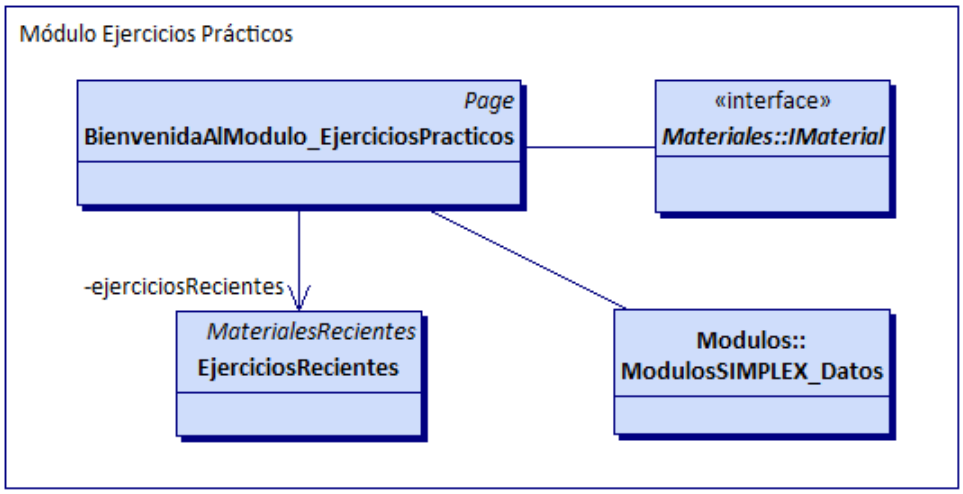
6.1.2.2 Diagrama: Interacción general de los módulos



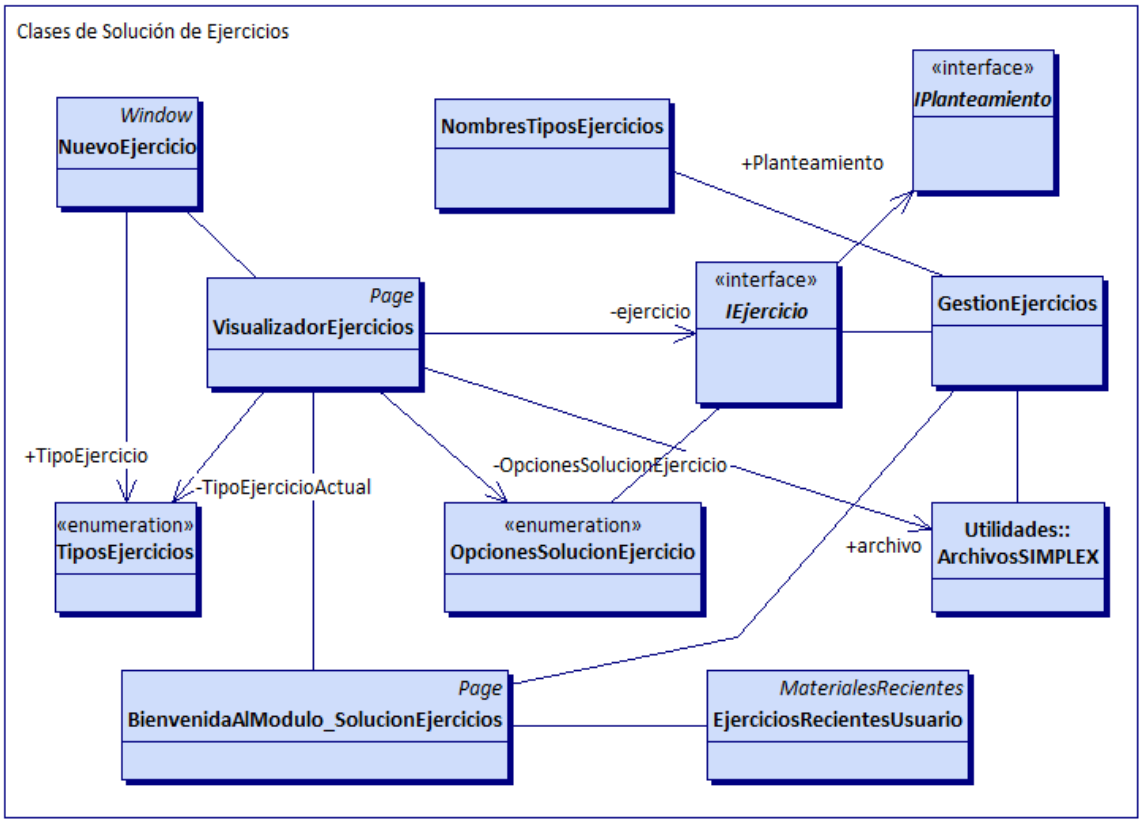
6.1.2.3 Diagrama: Información teórica

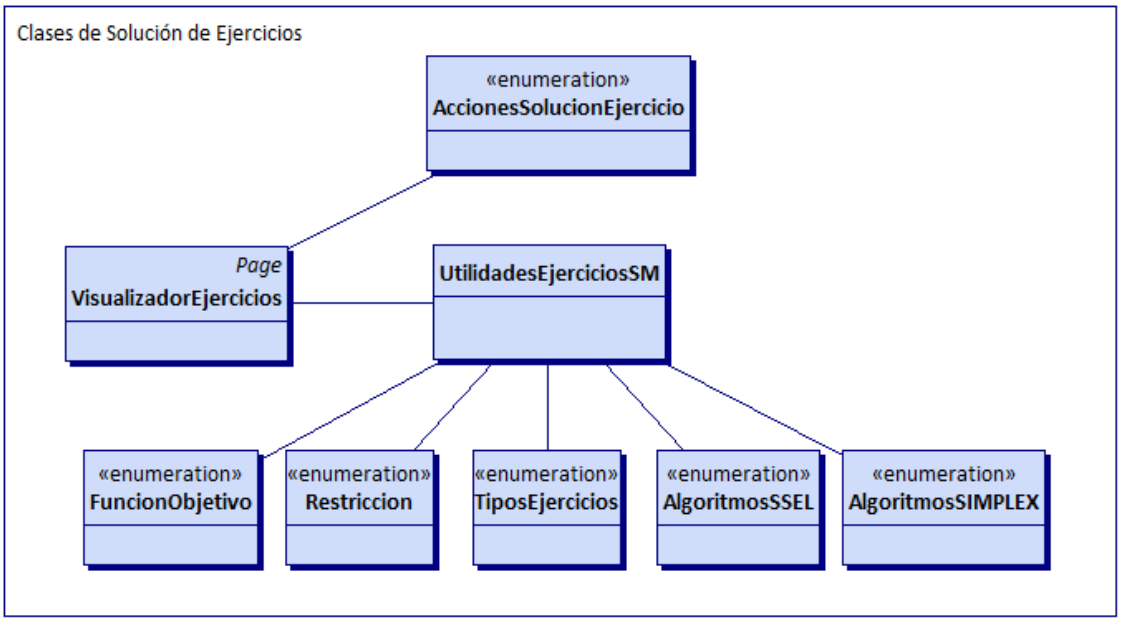


6.1.2.4 Diagrama: Ejercicios prácticos

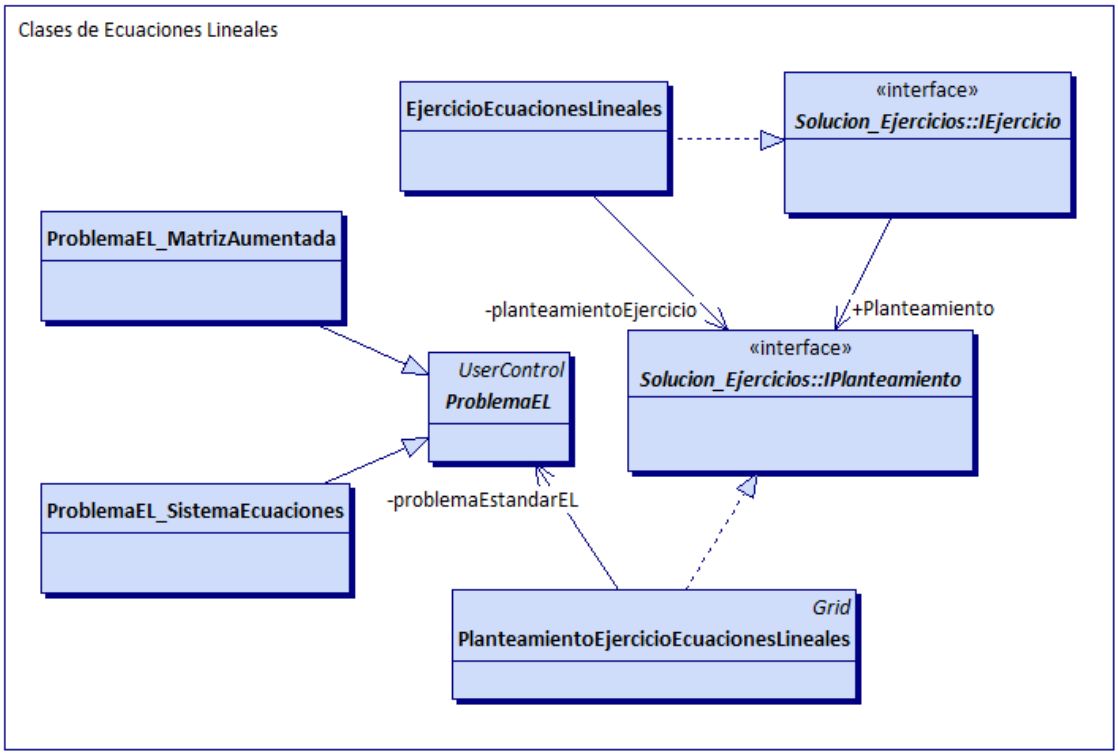


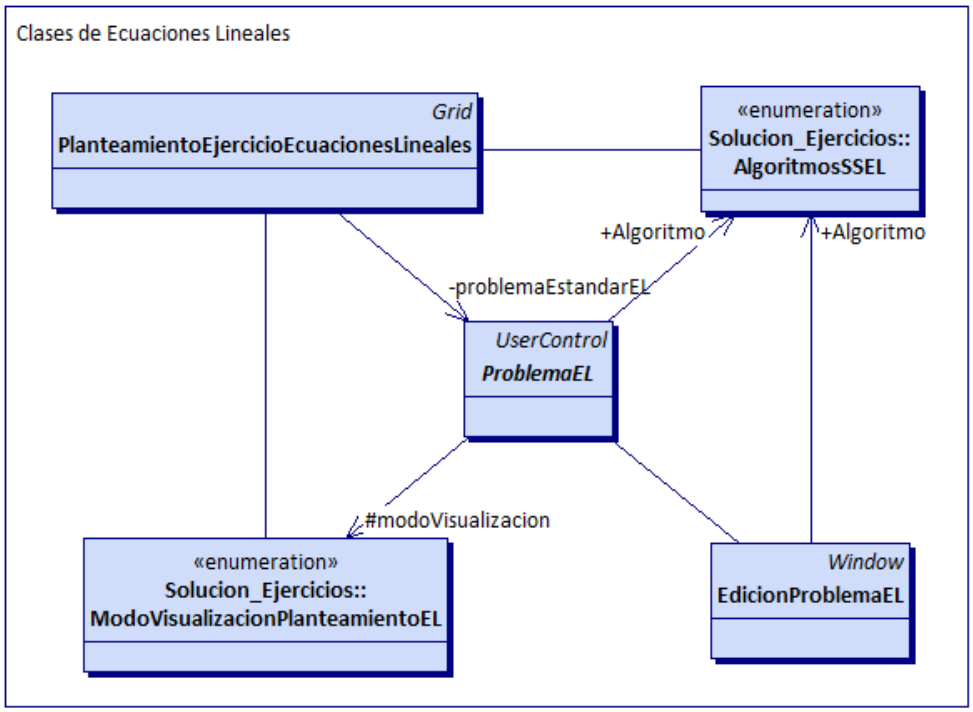
6.1.2.5 Diagrama: Solución de ejercicios

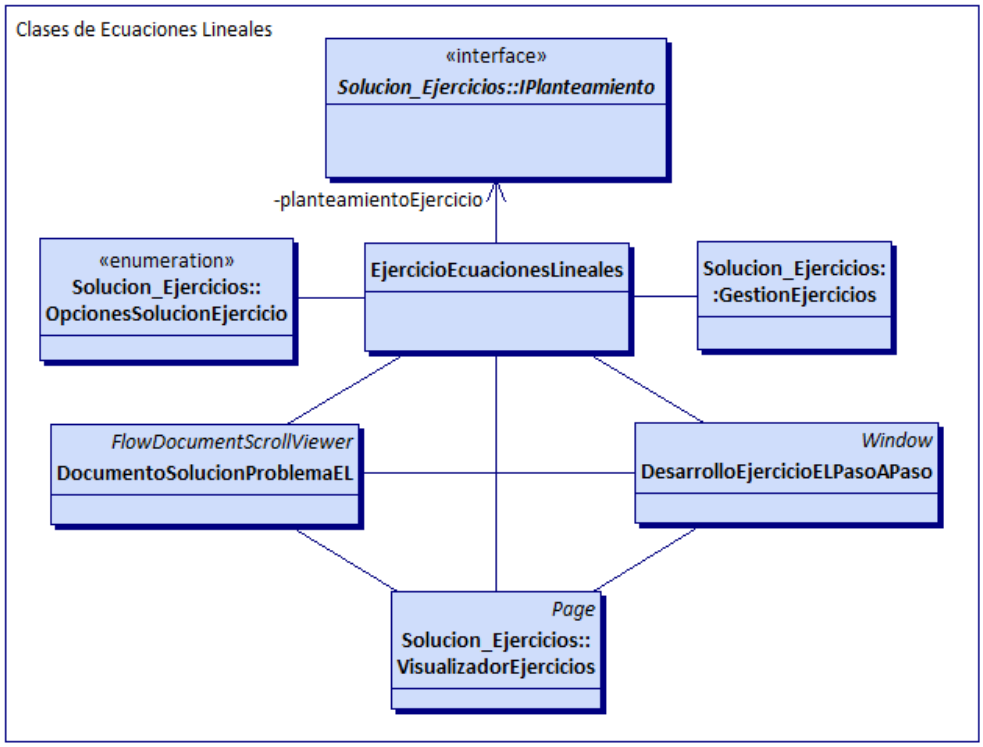




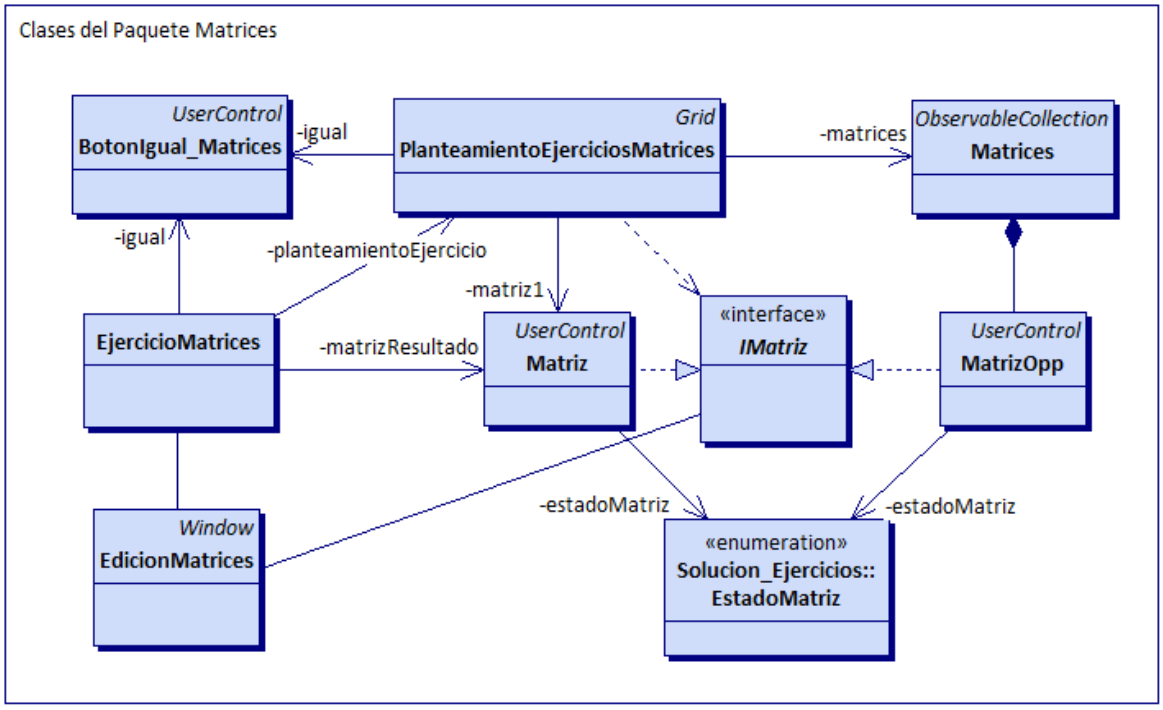
6.1.2.6 Diagrama: Ecuaciones lineales

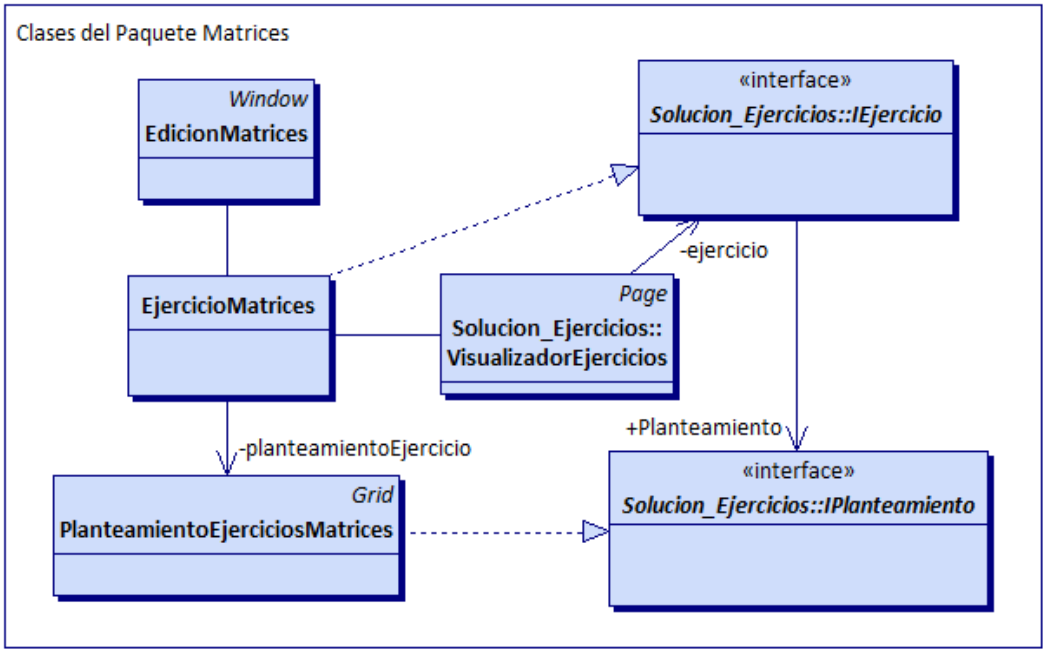



Clases de Ecuaciones Lineales
Grid
PlanteamientoEjercicioEcuacionesLineales
«enumeration»
Solucion_Ejercicios::
AlgoritmosSSEL
+Algoritmo
+Algoritmo
-problemaEstandarEL
UserControl
ProblemaEL
#modoVisualizacion
«enumeration»
Solucion_Ejercicios::
ModoVisualizacionPlanteamientoEL
Window
EdicionProblemaEL

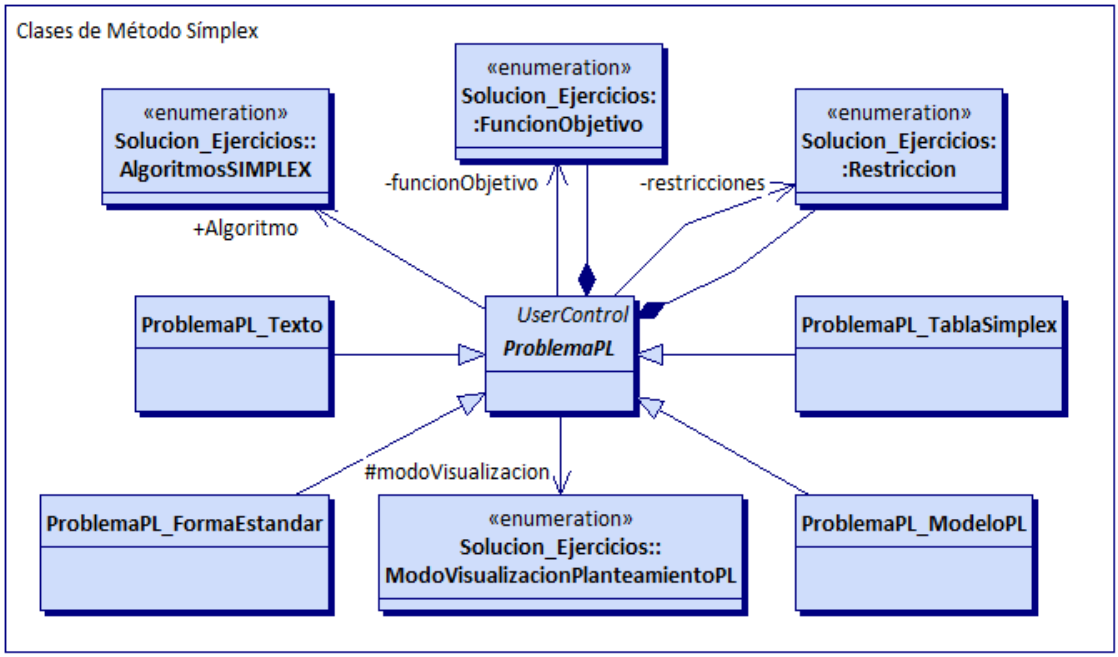

Clases de Ecuaciones Lineales
«interface»
Solucion_Ejercicios::IPlanteamiento
-planteamientoEjercicio
«enumeration»
Solucion_Ejercicios::
OpcionesSolucionEjercicio
EjercicioEcuacionesLineales
Solucion_Ejercicios:
:GestionEjercicios
FlowDocumentScrollViewer
DocumentoSolucionProblemaEL
Window
DesarrolloEjercicioELPasoAPaso
Page
Solucion_Ejercicios::
VisualizadorEjercicios

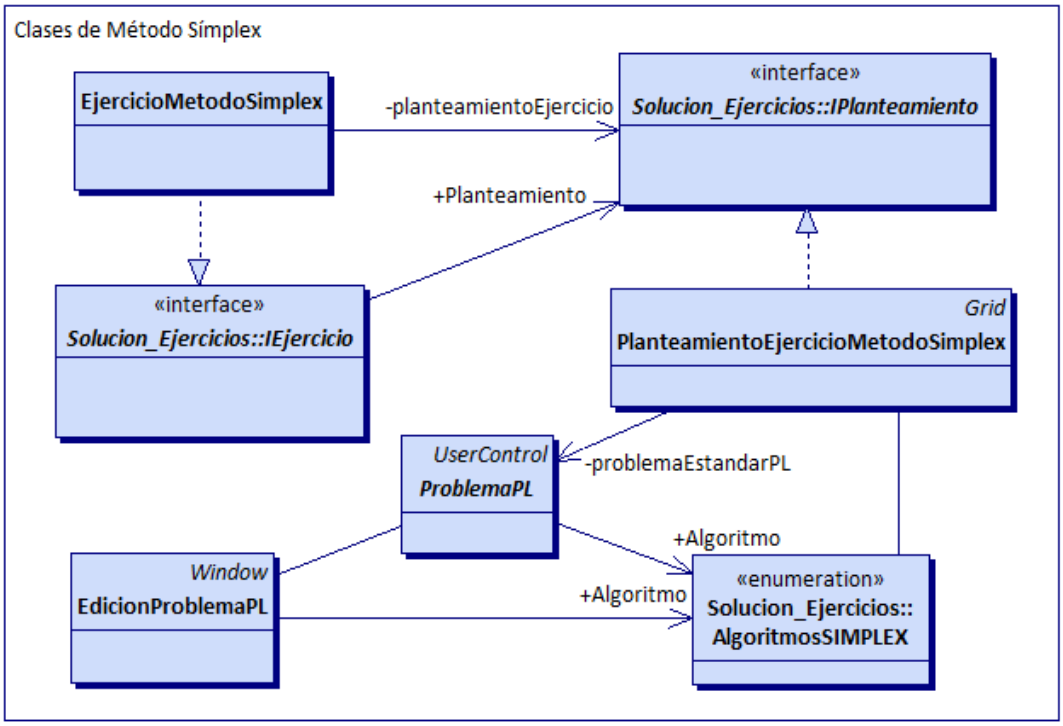
6.1.2.7 Diagrama: Matrices

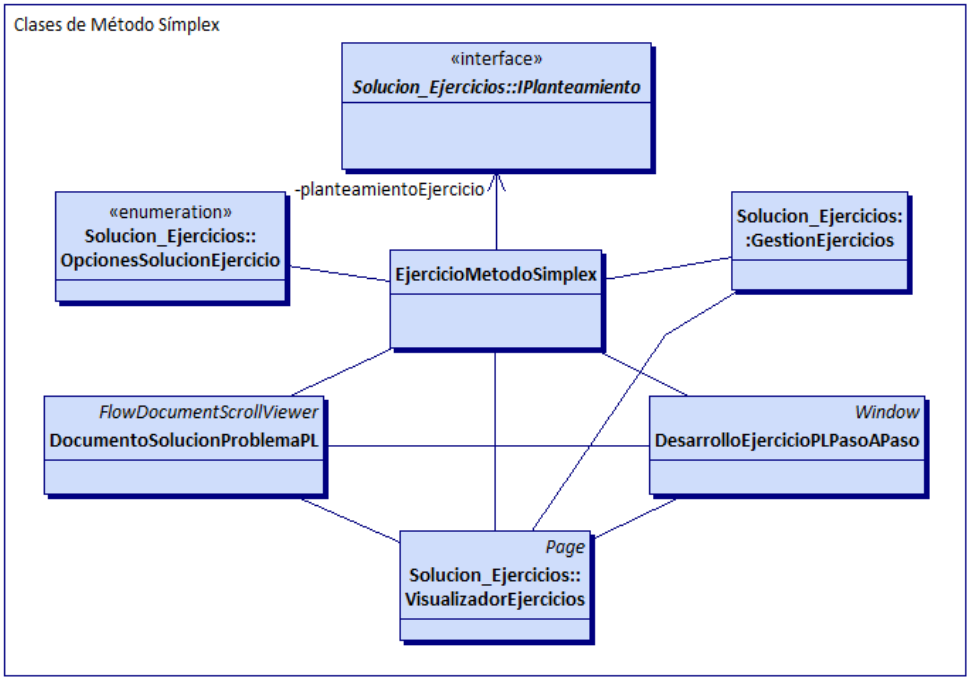




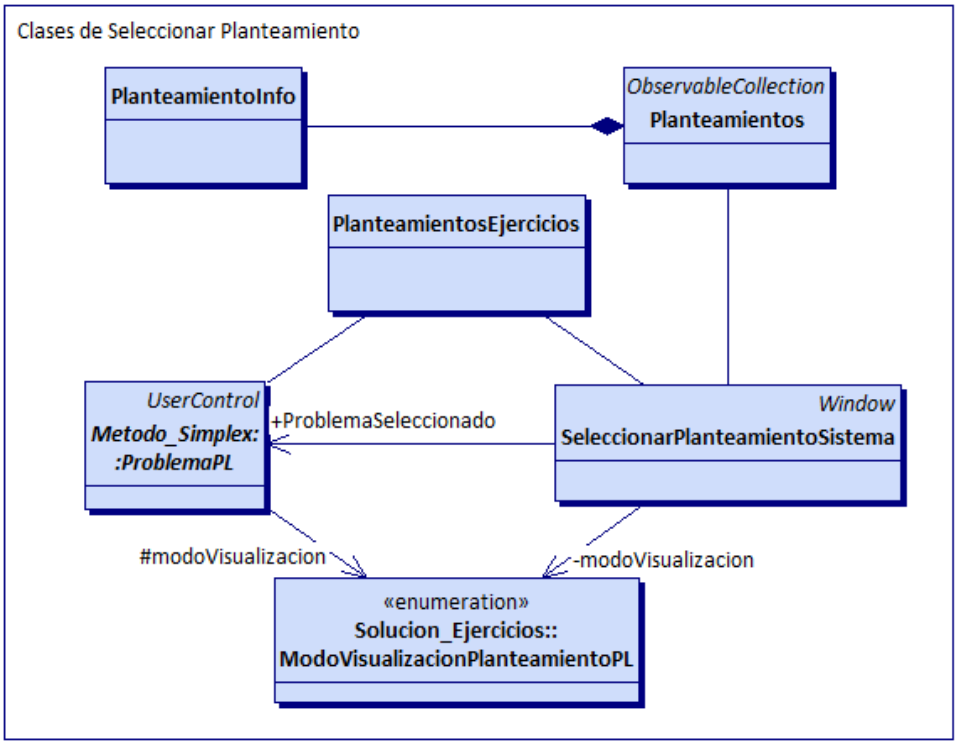
6.1.2.8 Diagrama: Método Símplex



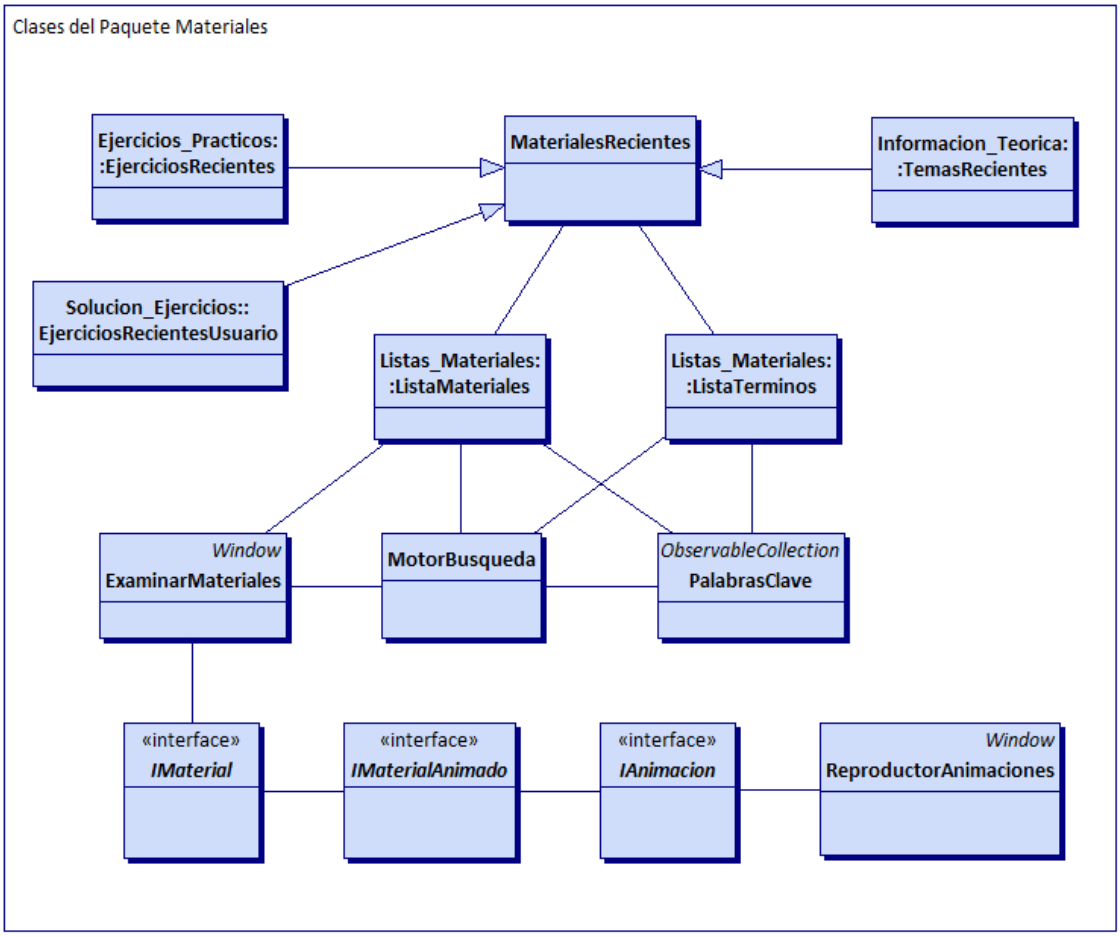




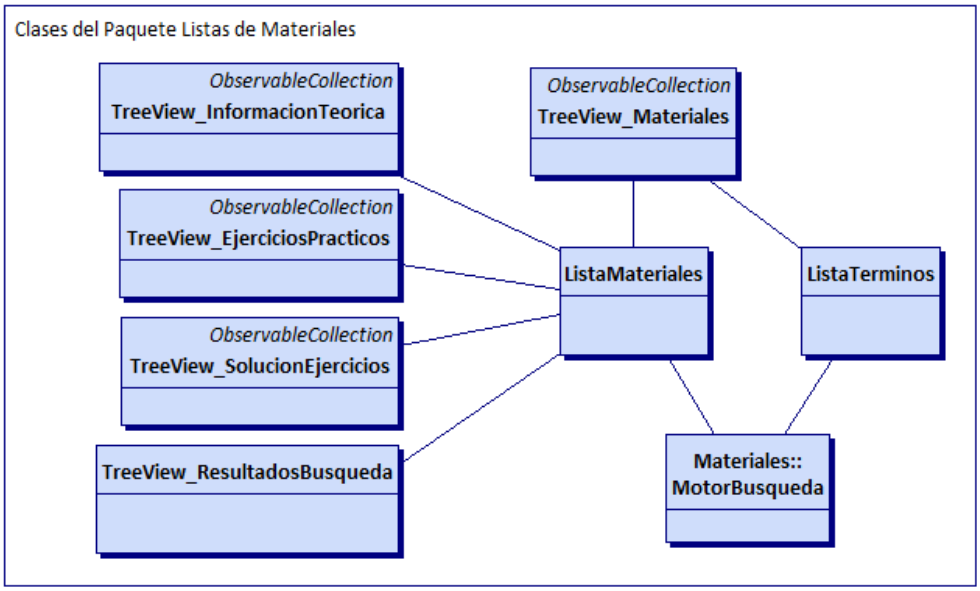
6.1.2.9 Diagrama: Seleccionar planteamiento



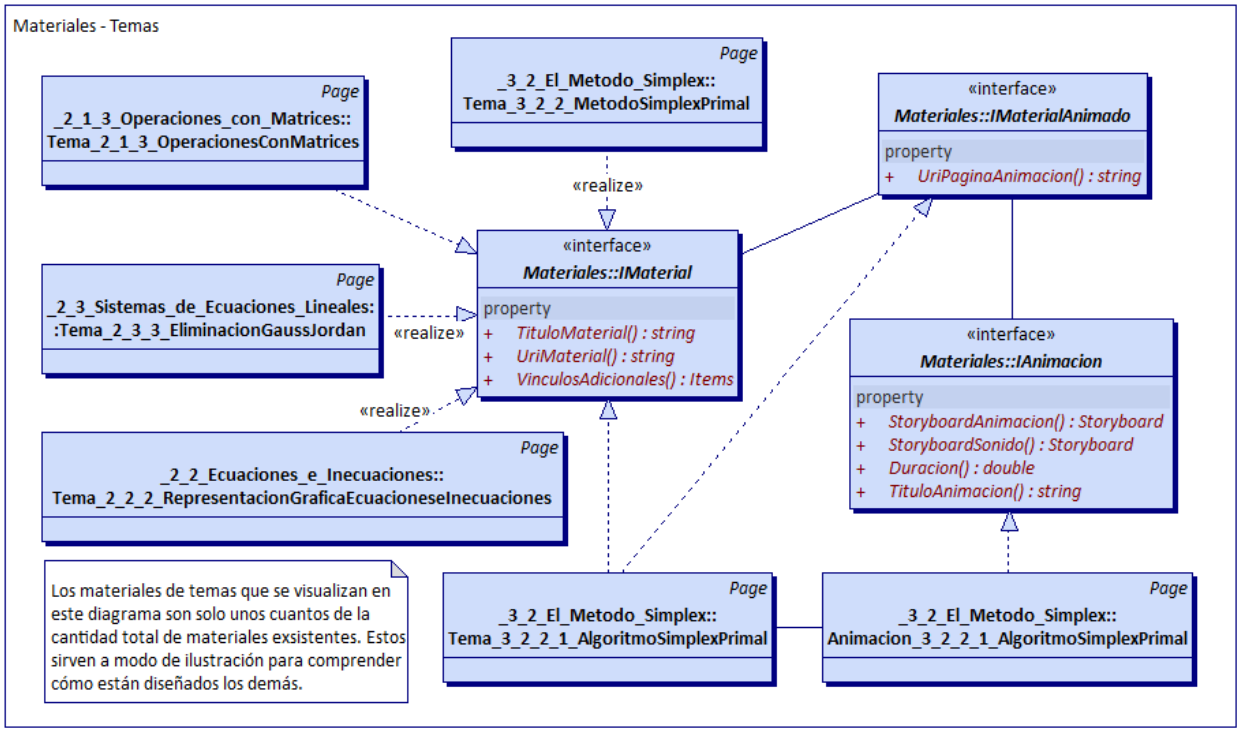
6.1.2.10 Diagrama: Materiales



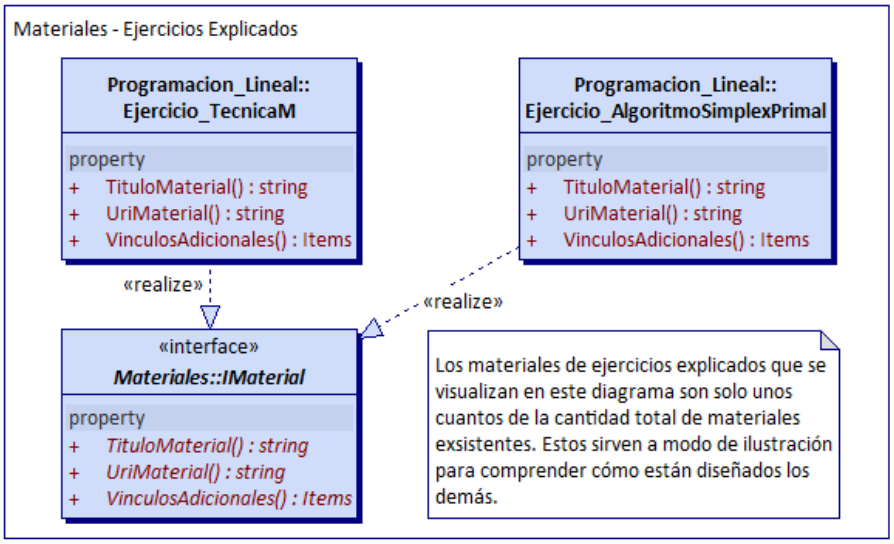
6.1.2.11 Diagrama: Listas de materiales



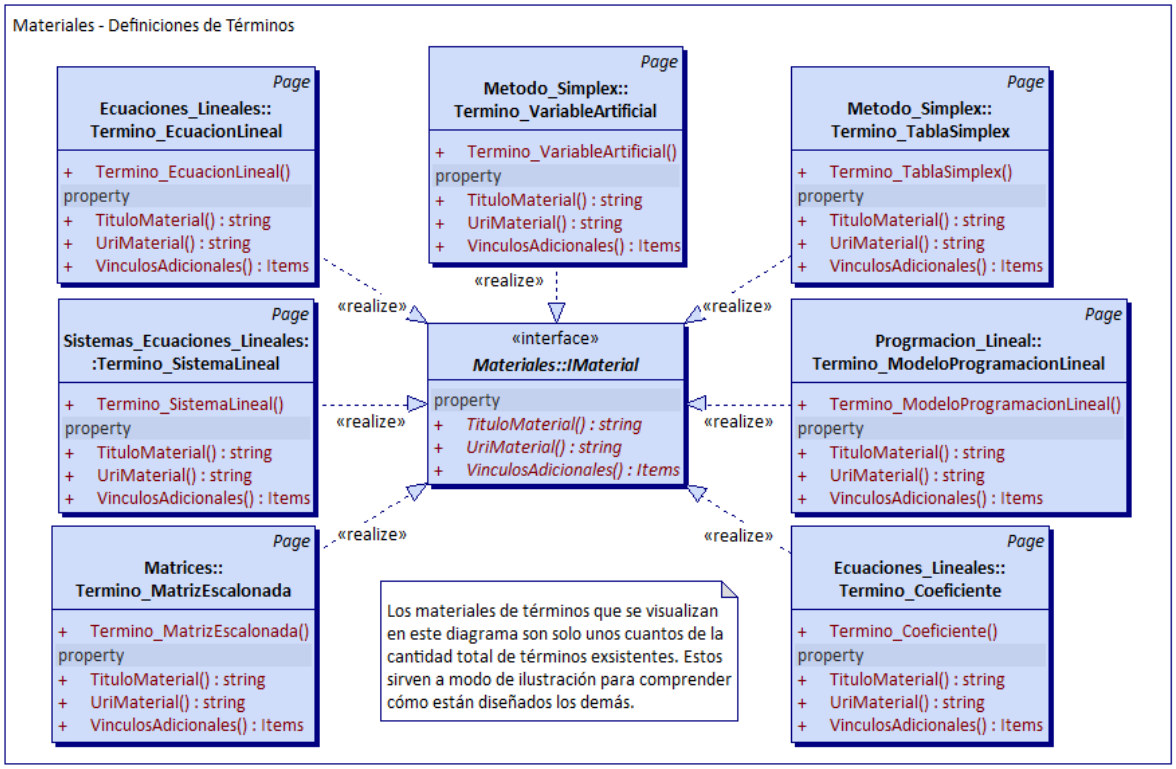
6.1.2.12 Diagrama: Materiales - Temas



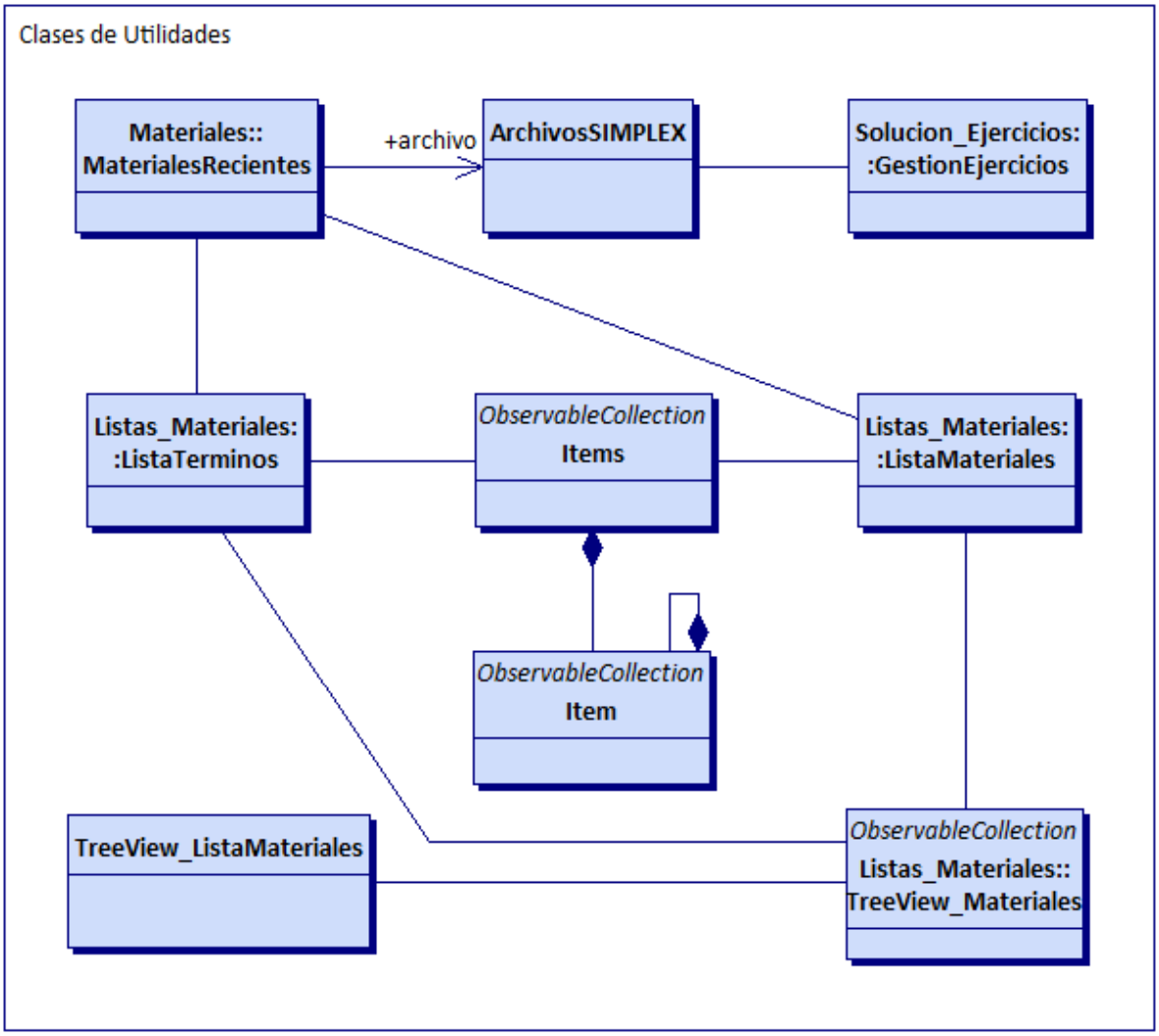
6.1.2.13 Diagrama: Materiales – Ejercicios explicados



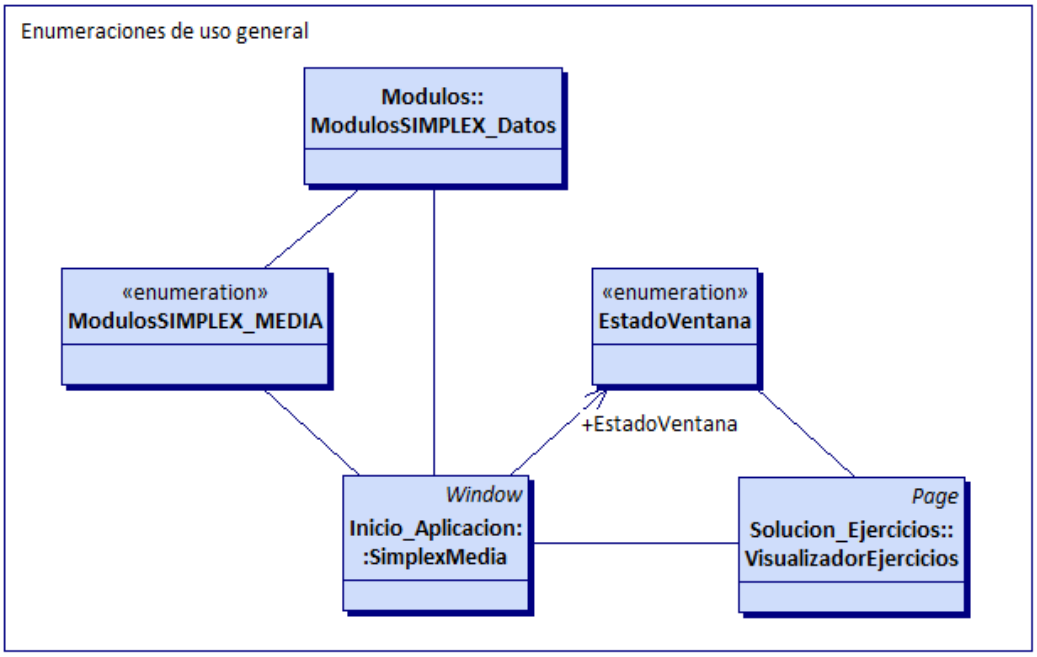
6.1.2.14 Diagrama: Materiales – Términos



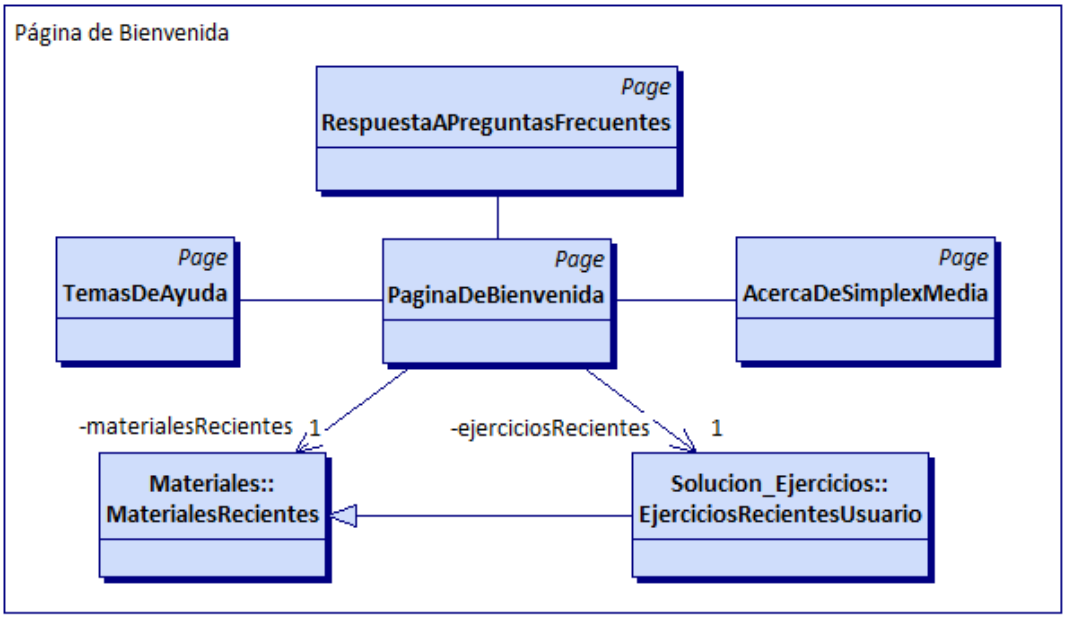
6.1.2.15 Diagrama: Utilidades



6.1.2.16 Diagrama: Enumeraciones generales



6.1.2.17 Diagrama: Página de bienvenida



6.2 REALIZACIONES DE CASOS DE USO – DISEÑO

Tal como en el caso de las *realizaciones de caso de uso – análisis*, las *realizaciones de caso de uso – diseño* son Colaboraciones del modelo de diseño utilizadas para describir cómo se realiza un caso de uso determinado, en términos de clases y objetos del diseño.

Una realización de caso de uso – diseño es una realización o implementación de una realización de caso de uso – análisis o de un caso de uso. De esta manera, tal como las realizaciones de caso de uso – análisis llegan a ser un camino de ejecución dentro de un caso de uso, las realizaciones de caso de uso – diseño son implementaciones de los diferentes caminos posibles que pueden tomarse dentro de una colaboración del análisis. Pero, es posible que un caso de uso no esté implementado por una realización de caso de uso – análisis y necesite ser implementado en el diseño, en ese caso, el trazado se efectúa de forma directa desde la realización de caso de uso – diseño hasta el caso de uso del modelo de casos de uso.

No es necesario especificar los trazados entre las realizaciones de caso de uso, pero, en este documento se presentan dichas relaciones para indicar cómo algunas colaboraciones del análisis son implementadas por varias del diseño.

Estas colaboraciones tienen la misma estructura que las del análisis, están compuestas por un diagrama de clases de diseño, un diagrama de interacción, un flujo de sucesos y un listado de requisitos de la implementación (opcional):

- *Diagrama de clases*: Con la ayuda de este diagrama el implementador puede abstraer los diferentes atributos y métodos de las clases que serán codificadas. Se pueden representar todas las relaciones de las clases que intervienen, de forma directa o indirecta, en la realización de un caso de uso. Sin embargo, no todas estas clases pueden estar en el diagrama de interacción, dado que algunas participan en operaciones muy internas que no suelen ser visibles, sino, solo en los Contratos de Operaciones.

- *Diagrama de Interacción*: Este diagrama sirve para observar lo que sucede en términos de código fuente en el interior del sistema cada vez que el actor envía mensajes al sistema y éste le responde. En las colaboraciones del análisis se usó Diagramas de Colaboración para detallar el la comunicación entre los objetos y los

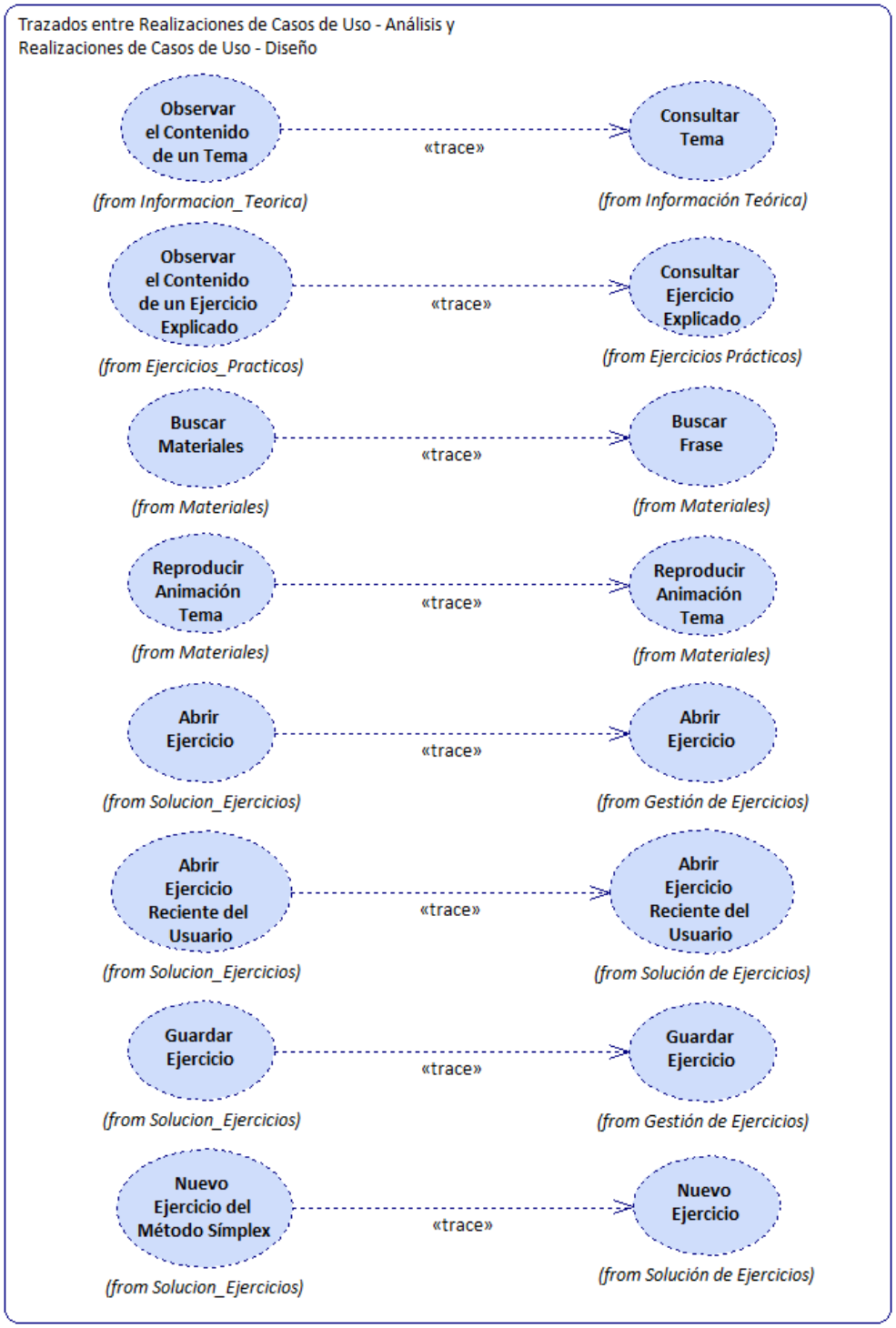
mensajes y responsabilidades que cada uno tiene. Ahora, lo que se desea determinar es la secuencia lógica y cronológica de dichos sucesos, por ello se usarán *Diagramas de Secuencia*. Estos diagramas son fácilmente traducidos a código fuente.

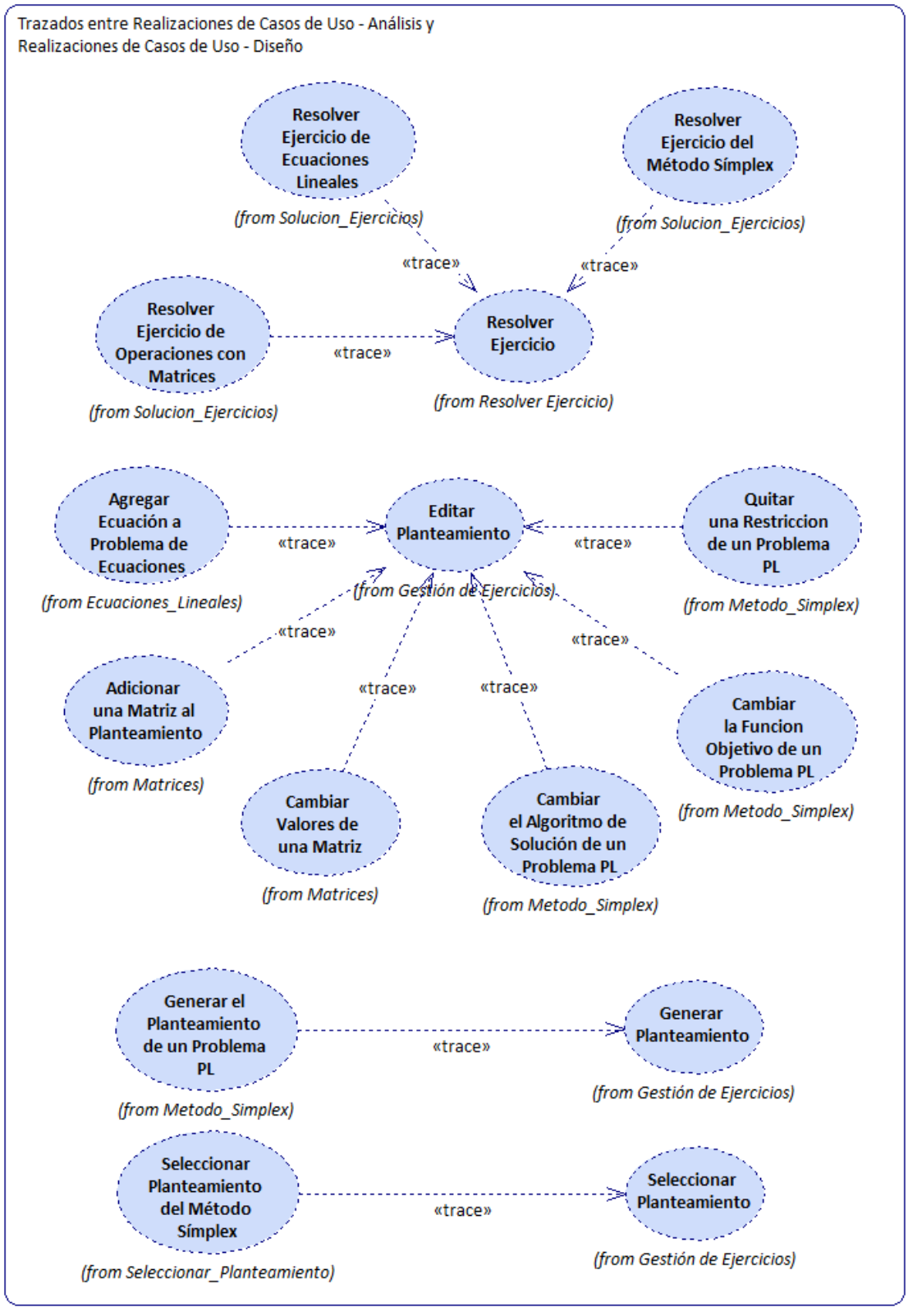
- *Flujo de Sucesos – diseño*: Con la descripción de los sucesos que se representan en los diagramas de colaboración se facilita el trabajo del implementador al momento de codificar las clases del diseño. Es necesario tener presente que la narración de un flujo de sucesos – diseño no debe hacer mención de los atributos y operaciones de los objetos. En caso que el diagrama posee Interfaces no se deben mencionar las operaciones de estas. Teniendo presente esto se evita, que tras la modificación repentina de un diagrama, tenga que modificarse el flujo de sucesos.

- *Requisitos de la implementación*: En la fase del diseño pueden aparecer requisitos no funcionales para el sistema. En esta sección se pueden detallar para luego ser implementadas en la fase siguiente.

Seguidamente se presentará un listado de los trazados existentes entre cada realización de caso de uso – diseño y realización de caso de uso – análisis. Además, se presentarán los diagramas y descripciones de las colaboraciones del diseño.

6.2.1 Trazados entre Realizaciones de casos de uso – análisis y Realizaciones de casos de uso – diseño

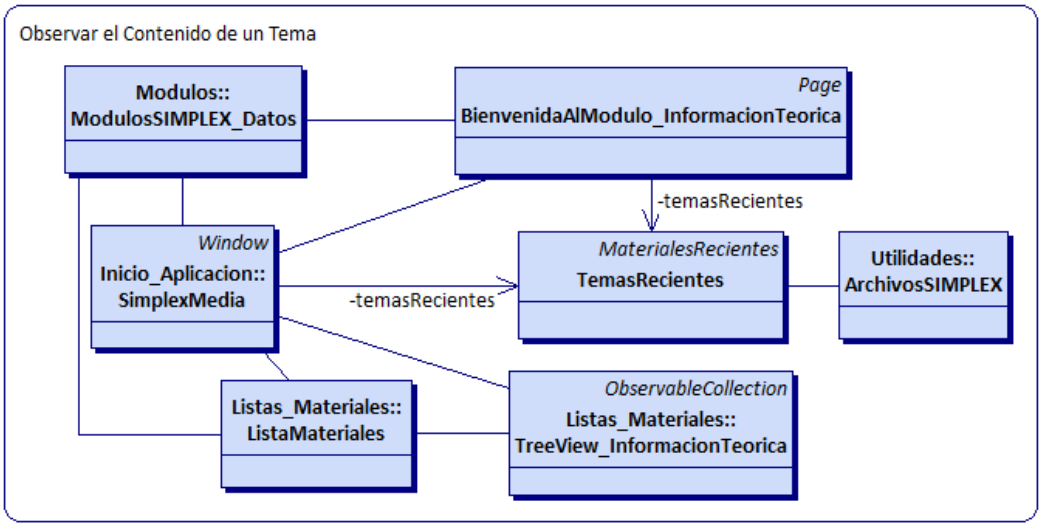



Trazados entre Realizaciones de Casos de Uso - Análisis y Realizaciones de Casos de Uso - Diseño
Resolver Ejercicio de Ecuaciones Lineales
(from Solucion_Ejercicios)
Resolver Ejercicio del Método Símplex
(from Solucion_Ejercicios)
«trace»
«trace»
Resolver Ejercicio de Operaciones con Matrices
(from Solucion_Ejercicios)
«trace»
Resolver Ejercicio
(from Resolver Ejercicio)
Agregar Ecuación a Problema de Ecuaciones
(from Ecuaciones_Lineales)
«trace»
Editar Planteamiento
(from Gestión de Ejercicios)
«trace»
Quitar una Restriccion de un Problema PL
(from Metodo_Simplex)
«trace»
«trace»
Adicionar una Matriz al Planteamiento
(from Matrices)
«trace»
«trace»
Cambiar la Funcion Objetivo de un Problema PL
(from Metodo_Simplex)
Cambiar Valores de una Matriz
(from Matrices)
Cambiar el Algoritmo de Solución de un Problema PL
(from Metodo_Simplex)
Generar el Planteamiento de un Problema PL
(from Metodo_Simplex)
«trace»
Generar Planteamiento
(from Gestión de Ejercicios)
Seleccionar Planteamiento del Método Símplex
(from Seleccionar_Planteamiento)
«trace»
Seleccionar Planteamiento
(from Gestión de Ejercicios)

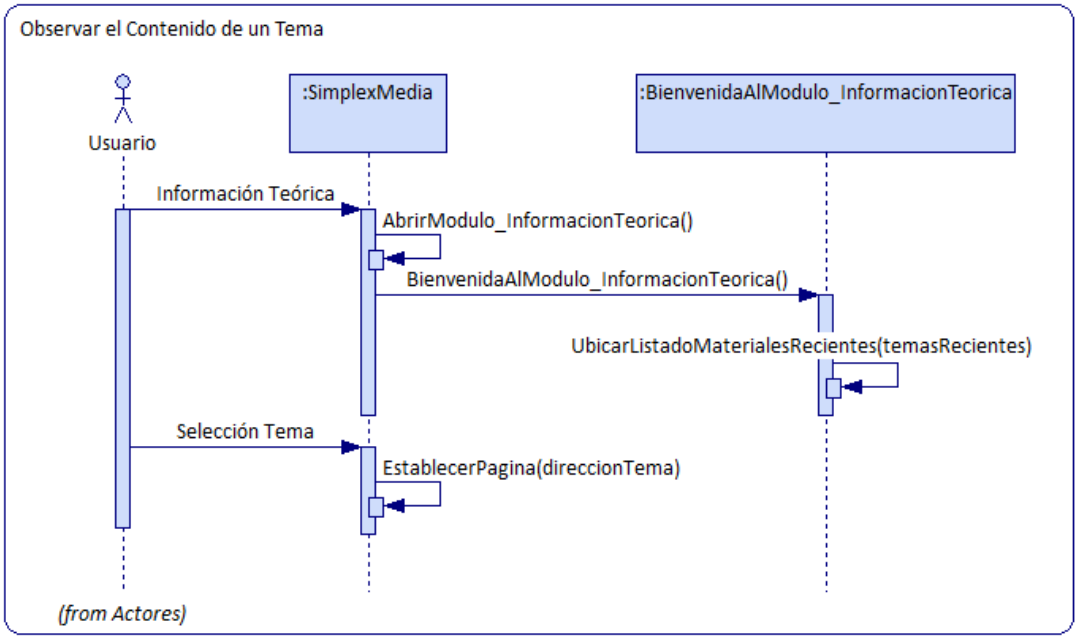
6.2.2 Colaboraciones

6.2.2.1 Observar el contenido de un tema

6.2.2.1.1 Diagrama de clases



6.2.2.1.2 Diagrama de secuencia



6.2.2.1.3 Flujo de sucesos – diseño

Después de haber abierto a SÍMPLEX MEDIA (y de haberse cargado el listado de temas en la interfaz principal de SÍMPLEX MEDIA), el Usuario da indica al sistema que desea abrir el módulo "Información Teórica". Acto seguido, el sistema abre el módulo y carga la página de Bienvenida a dicho módulo. Una vez abierta, la página de *BienvenidaAlModuloInformacionTeorica*, se ubican en pantalla el listado de Temas Recientes.

El Usuario selecciona un tema, del listado principal de temas o del listado de temas recientes, y se abre la página deseada.

Véase las operaciones:

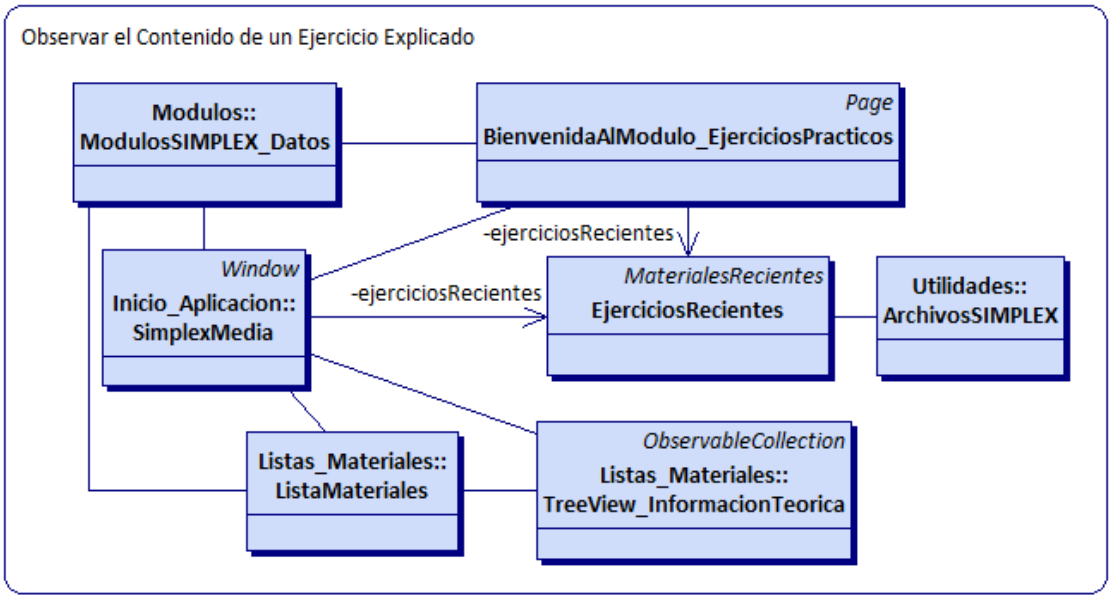
- AbrirModulo_InformacionTeorica() en la sección 7.3.1
- UbicarListadoMaterialesRecientes (materialesRecientes) en la sección 7.3.2

6.2.2.1.4 Requisitos de la implementación

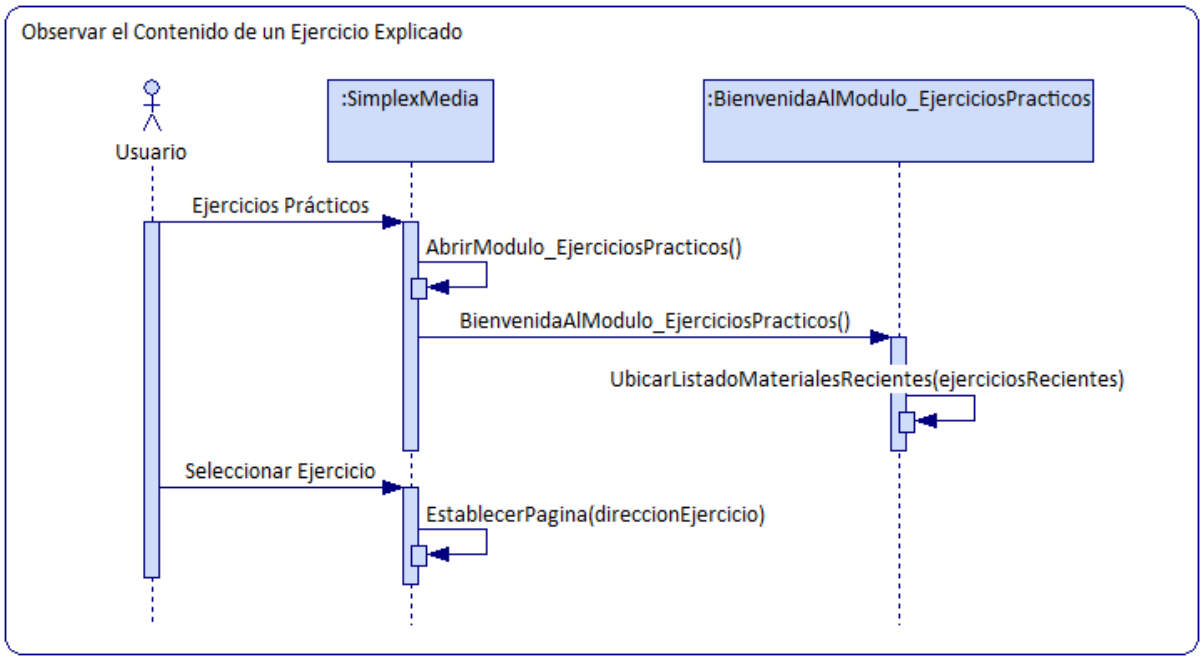
- SÍMPLEX MEDIA debe mostrar el listado de temas del módulo de Información Teórica en el momento que se abra la aplicación (aún sin haber seleccionado un módulo determinado).

6.2.2.2 Observar el Contenido de un Ejercicio Explicado

6.2.2.2.1 Diagrama de clases



6.2.2.2.2 Diagrama de secuencia



6.2.2.2.3 Flujo de sucesos – diseño

Después de haber abierto a SÍMPLEX MEDIA (y de haberse cargado el listado de temas en la interfaz principal de SÍMPLEX MEDIA), el Usuario da indica al sistema que desea abrir el módulo "Ejercicios Prácticos". Acto seguido, el sistema abre el módulo y carga la página de Bienvenida a dicho módulo. Una vez abierta, la página de *BienvenidaAlModuloEjerciciosPracticos*, se ubican en pantalla el listado de Ejercicios Explicados Recientes.

El Usuario selecciona un tema de ejercicio, del listado principal de ejercicios explicados o del listado de ejercicios explicados recientes, y se abre la página deseada.

Véase las operaciones:

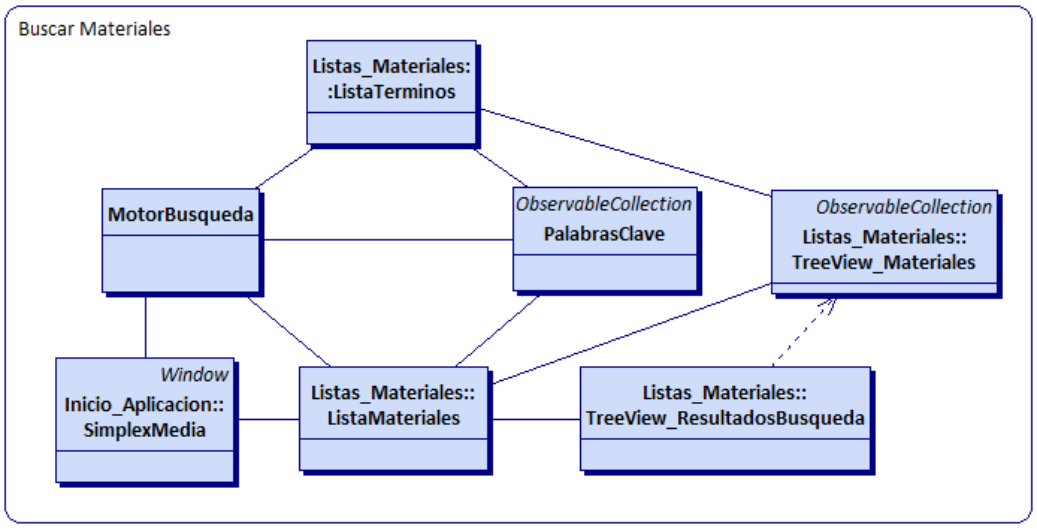
- AbrirModulo_InformacionTeorica() en la sección 7.3.1
- UbicarListadoMaterialesRecientes (materialesRecientes) en la sección 7.3.2

6.2.2.2.4 Requisitos de la implementación

- SÍMPLEX MEDIA debe mostrar el listado de ejercicios explicados del módulo de Ejercicios Prácticos en el momento que se abra la aplicación (aún sin haber seleccionado un módulo determinado).

6.2.2.3 Buscar materiales

6.2.2.3.1 Diagrama de clases



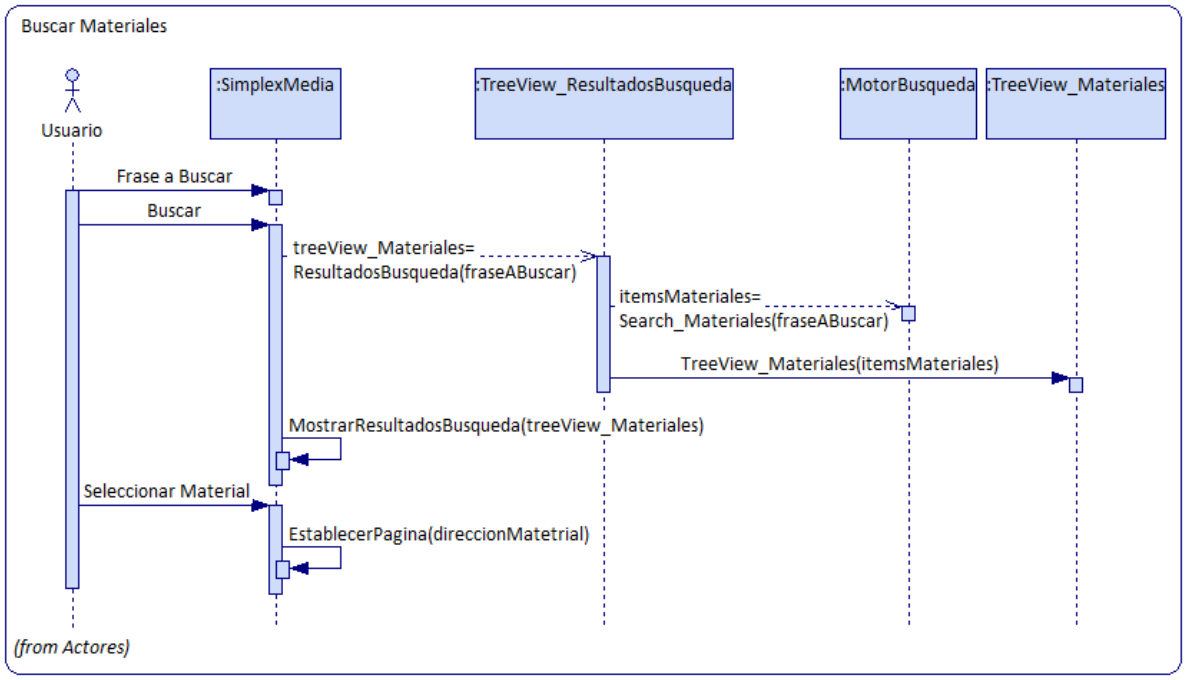
6.2.2.3.2 Flujo de sucesos – diseño

El Usuario ingresa al sistema la frase que desea a buscar y ordena la búsqueda. Se crea un árbol visual de resultados a través de *TreeView_ResultadosBusqueda*, quien invoca al *MotorBusqueda* para que le proporcione el listado de resultados y convertirlos en lista visual usando a *TreeView_Materiales*.

SimplexMedia muestra los resultados de la búsqueda en pantalla.

El usuario selecciona uno de los materiales del listado y el sistema lo muestra en pantalla.

6.2.2.3.3 Diagrama de secuencia

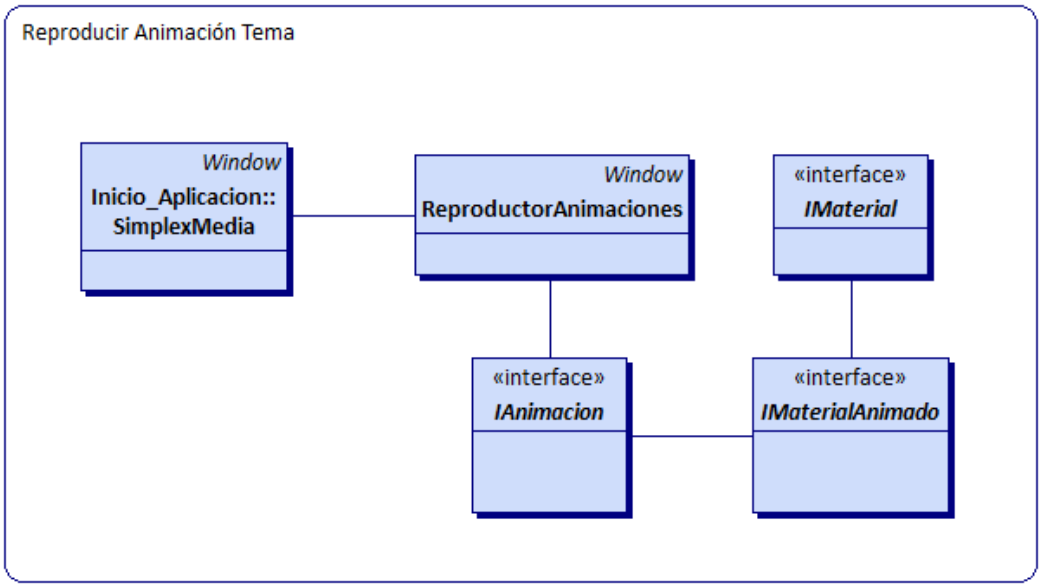


6.2.2.3.4 Requisitos de la implementación

- En los resultados de la búsqueda deben aparecer términos, temas y ejercicios explicados.

6.2.2.4 Reproducir animación tema

6.2.2.4.1 Diagrama de clases



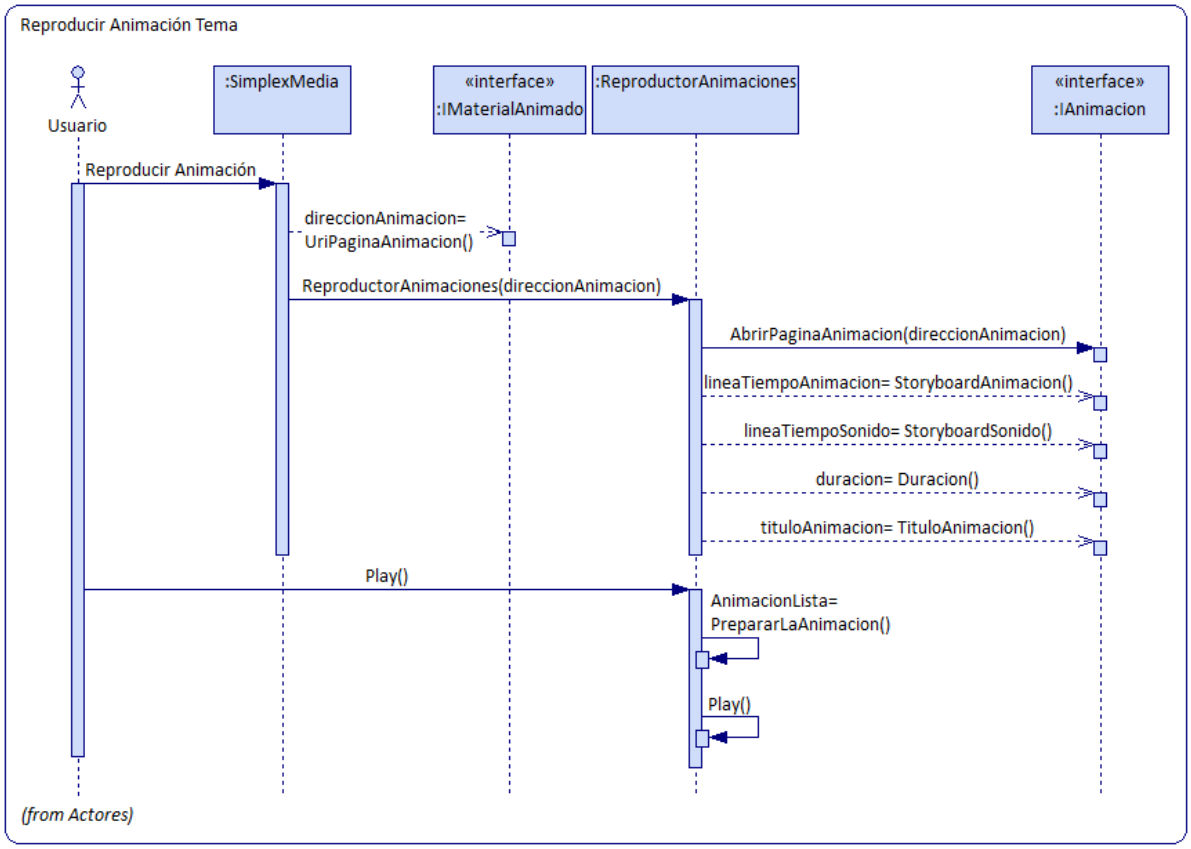
6.2.2.4.2 Flujo de sucesos – diseño

El Usuario indica a *SimplexMedia* que desea reproducir la animación del material que se está visualizando. El sistema extrae del material la dirección de la animación que está asociada a él.

Se abre al *ReproductorAnimaciones* y se le proporciona la dirección de la animación. *ReproductorAnimaciones* abre la animación, extrae las líneas de tiempo de la animación y del sonido, también extrae la duración y el título de la animación.

El Usuario ordena al *ReproductorAnimaciones* que inicie la reproducción de la animación. *ReproductorAnimaciones* prepara la animación e inicia la reproducción.

6.2.2.4.3 Diagrama de secuencia



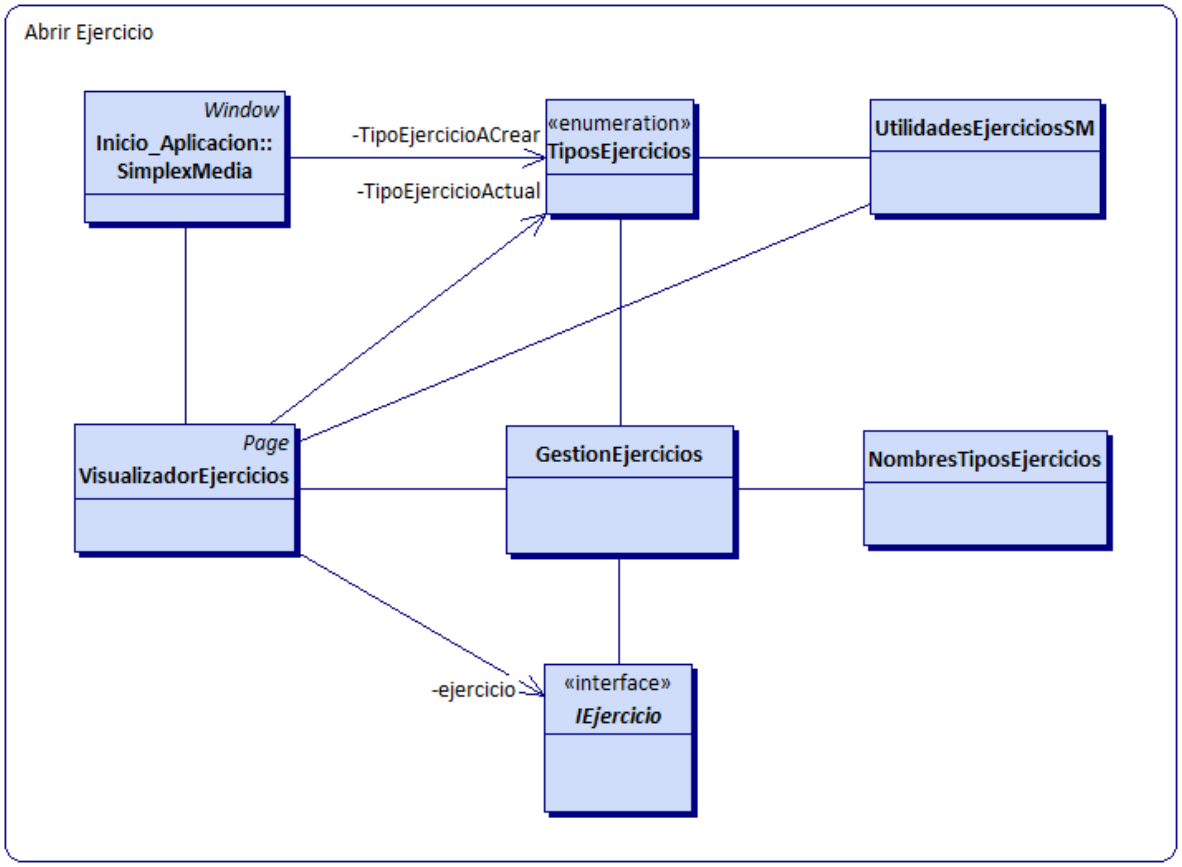
6.2.2.4.4 Requisitos de la implementación

- El ReproductorAnimaciones tendrá disponible un listado de adicional de animaciones (en caso que el material posea más de una) para que el usuario seleccione la de su preferencia.

- En caso que el material que se esté visualizando no posea una animación, el sistema debe desactivar la opción con la que el usuario le pueda indicar que quiere observarla.

6.2.2.5 Abrir ejercicio

6.2.2.5.1 Diagrama de clases



6.2.2.5.2 Flujo de sucesos – diseño

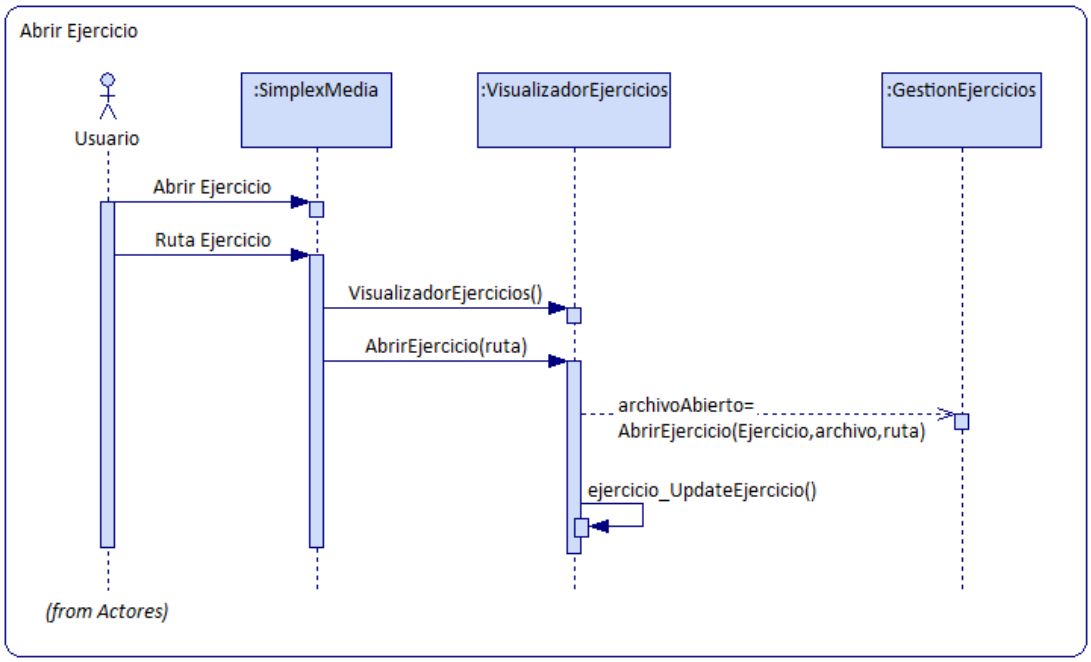
El Usuario le indica al sistema que desea abrir un ejercicio y le proporciona la ruta o dirección del archivo. *SimplexMedia* abre al *VisualizadorEjercicios* y se ordena abrir el ejercicio desde la ruta especificada.

Mediante el *GestorEjercicios* se realiza el proceso de apertura del archivo y se muestran los datos del planteamiento del ejercicio en el *VisualizadorEjercicio*.

Véase la operación:

- AbrirEjercicio(Ejercicio, archivo, ruta) en la sección 7.3.4

6.2.2.5.3 Diagrama de secuencia

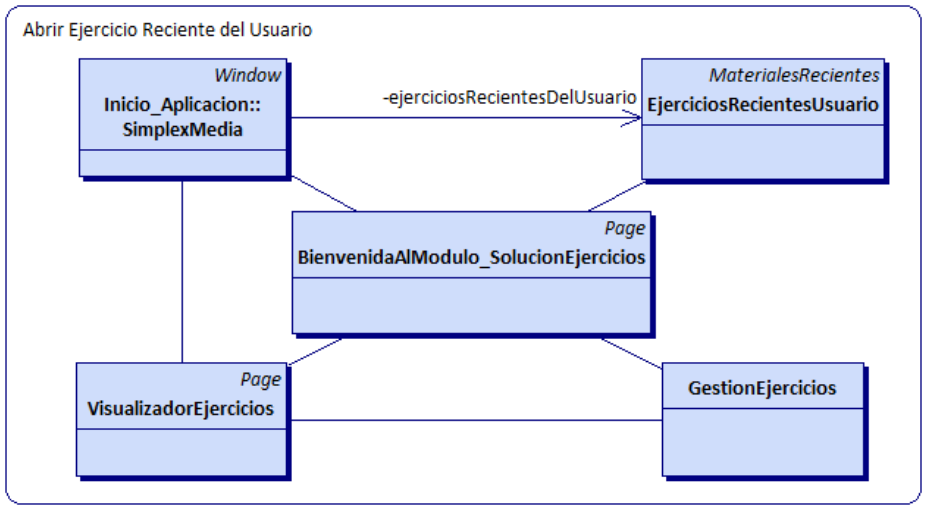


6.2.2.5.4 Requisitos de la implementación

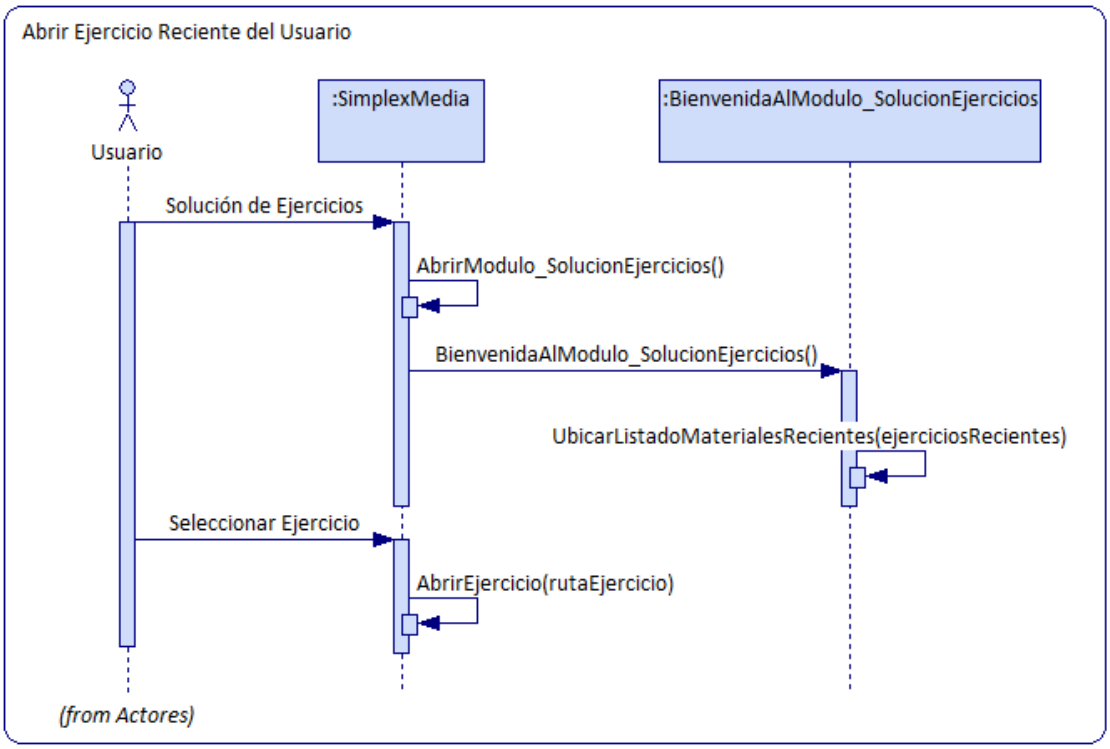
- En la operación *AbrirEjercicio(Ejercicio, archivo, ruta)* deben realizarse las validaciones necesarias al abrir el planteamiento.

6.2.2.6 Abrir ejercicio reciente del usuario

6.2.2.6.1 Diagrama de clases



6.2.2.6.2 Diagrama de secuencia



6.2.2.6.3 Flujo de sucesos – diseño

EL Usuario indica al sistema que desea abrir el módulo Solución de Ejercicios. SimplexMedia abre el módulo cargando la página de bienvenida *BienvenidaAlModulo_SolucionEjercicios* y se carga el listado de ejercicios recientes del usuario. El Usuario selecciona un ejercicio del listado y el sistema se encarga de abrirlo.

Véase:

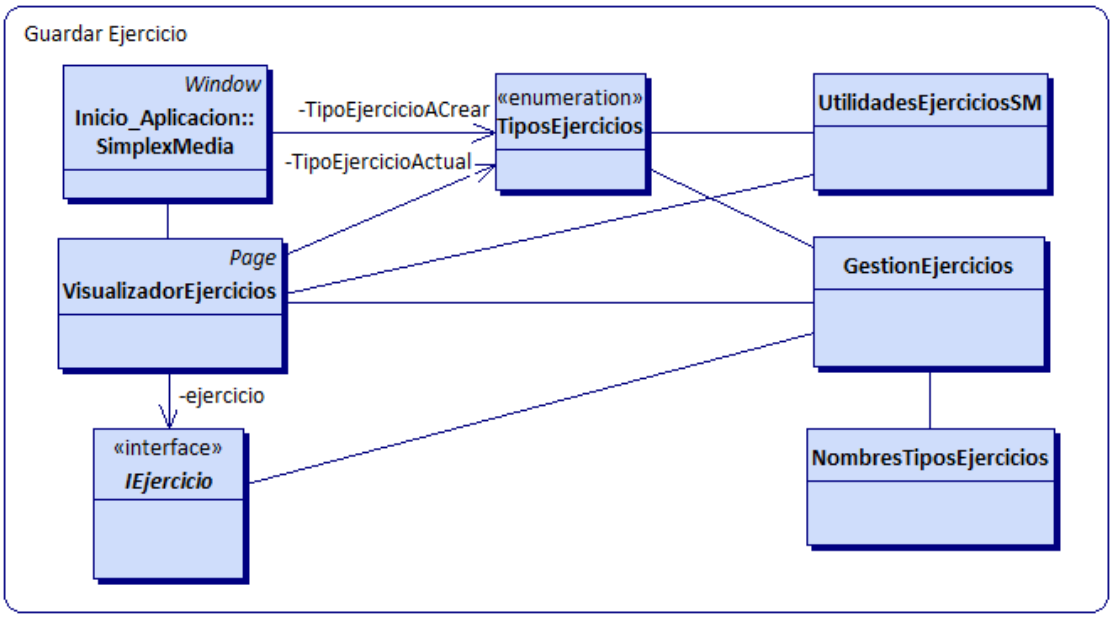
- La Colaboración "Abrir Ejercicio" de la sección 7.2.2.5.
- La Operación AbrirEjercicio(Ejercicio, archivo, ruta) en la sección 7.3.4

6.2.2.6.4 Requisitos de la implementación

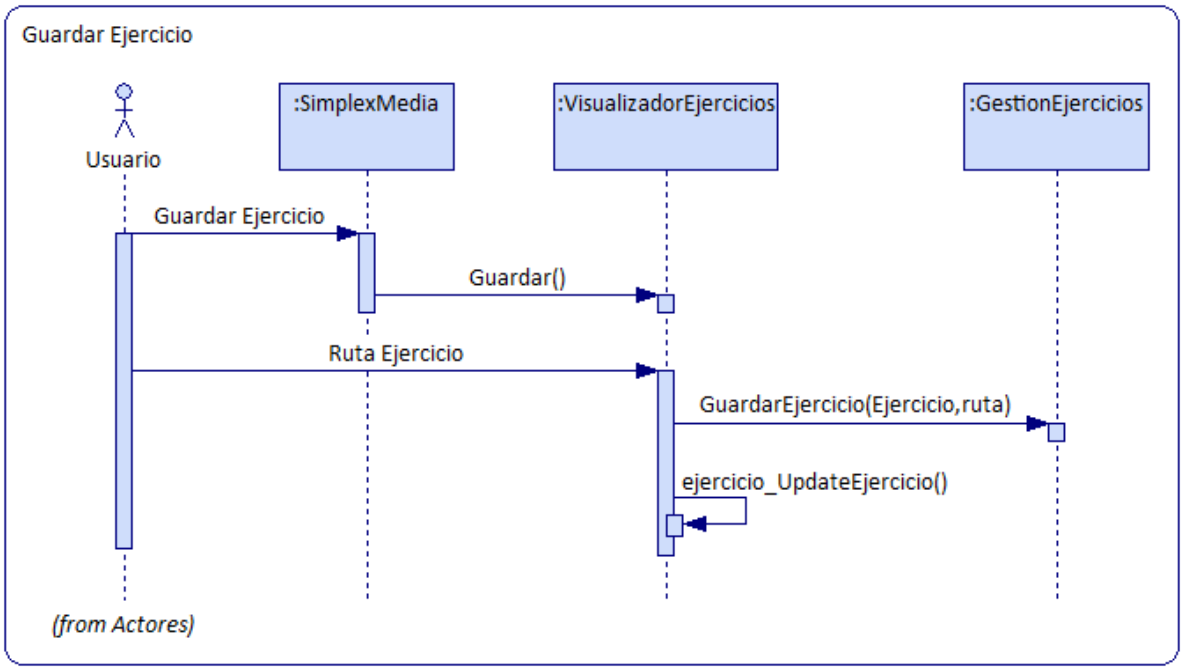
- *SimplexMedia* deberá mostrar los ejercicios recientes del usuario en el momento en que se abre (aún sin que se haya seleccionado un módulo específico).

6.2.2.7 Guardar ejercicio

6.2.2.7.1 Diagrama de clases



6.2.2.7.2 Diagrama de secuencia



6.2.2.7.3 Flujo de sucesos – diseño

El Usuario le indica al sistema que desea Guardar el ejercicio que está usando. El sistema para esa solicitud al *VisualizadorEjercicios*.

El Usuario le informa al *VisualizadorEjercicios* la ruta o dirección que desea para el archivo y se inicia el proceso de guardado del ejercicio.

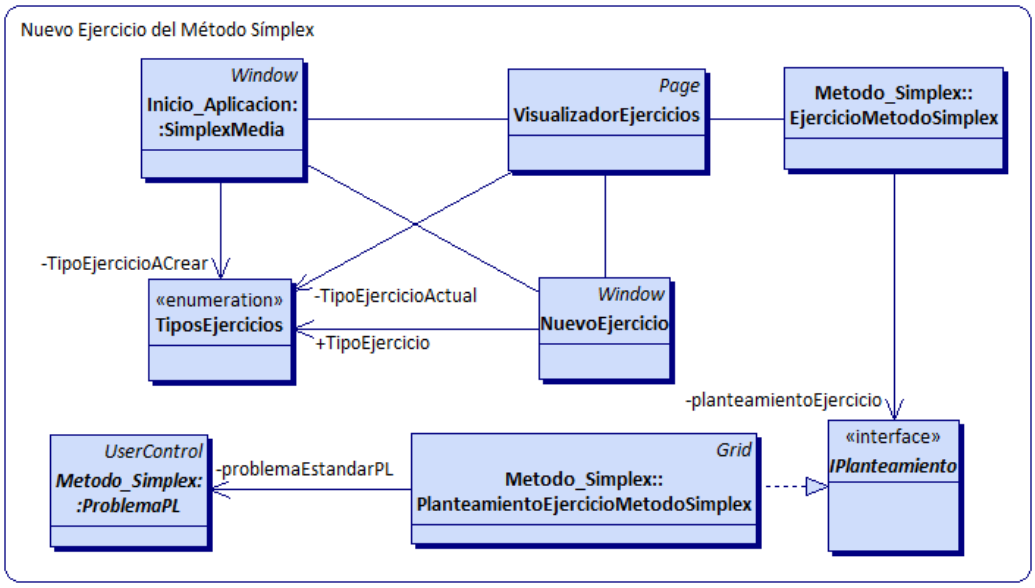
El *VisualizadorEjercicio* actualiza el ejercicio en pantalla.

Véase la operación:

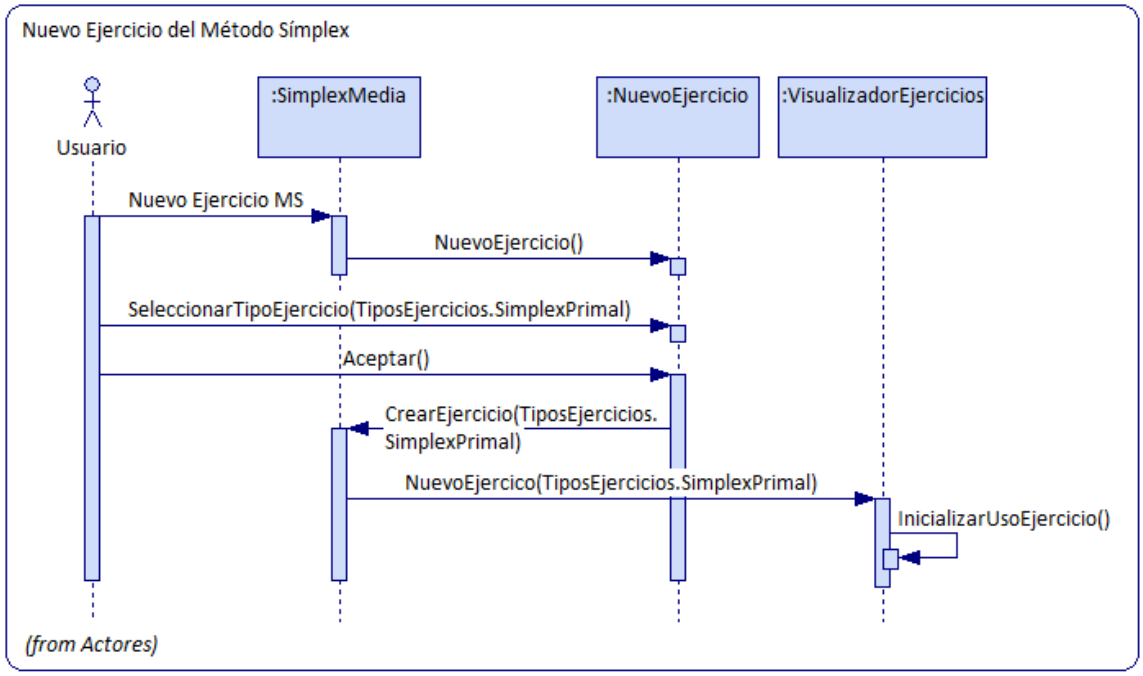
- GuardarEjercicio(Ejercicio, ruta) en la Sección 7.3.5

6.2.2.8 Nuevo ejercicio del método símplex

6.2.2.8.1 Diagrama de clases



6.2.2.8.2 Diagrama de secuencia



6.2.2.8.3 Flujo de sucesos – diseño

El Usuario indica al sistema que desea crear un nuevo ejercicio. El sistema abre a *NuevoEjercicio* y se muestran los diferentes tipos de ejercicios que puede resolver SÍMPLEX MEDIA.

El Usuario selecciona el tipo "Simplex Primal" y confirma su operación. *NuevoEjercicio* indica a *SimplexMedia* cual fue la decisión del Usuario.

Con la orden de crear un ejercicio del Algoritmo Símplex Primal, *SimplexMedia* abre el *VisualizadorEjercicios* y le pasa la solicitud del Usuario. Se crea el nuevo ejercicio y se inicia el uso de este.

Véase la operación:

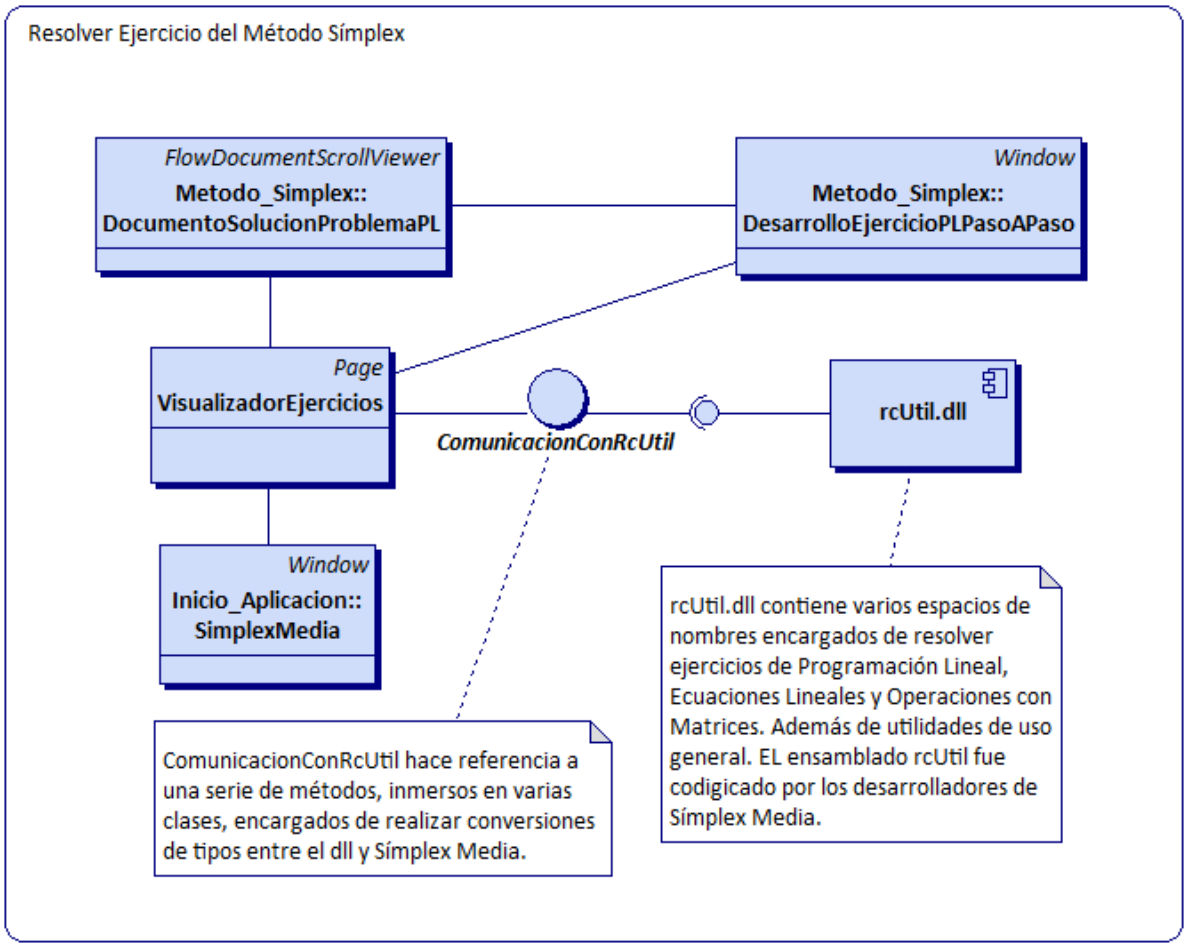
- NuevoEjercicio(TipoEjercicio) en la sección 7.3.3

6.2.2.8.4 Requisitos de la implementación

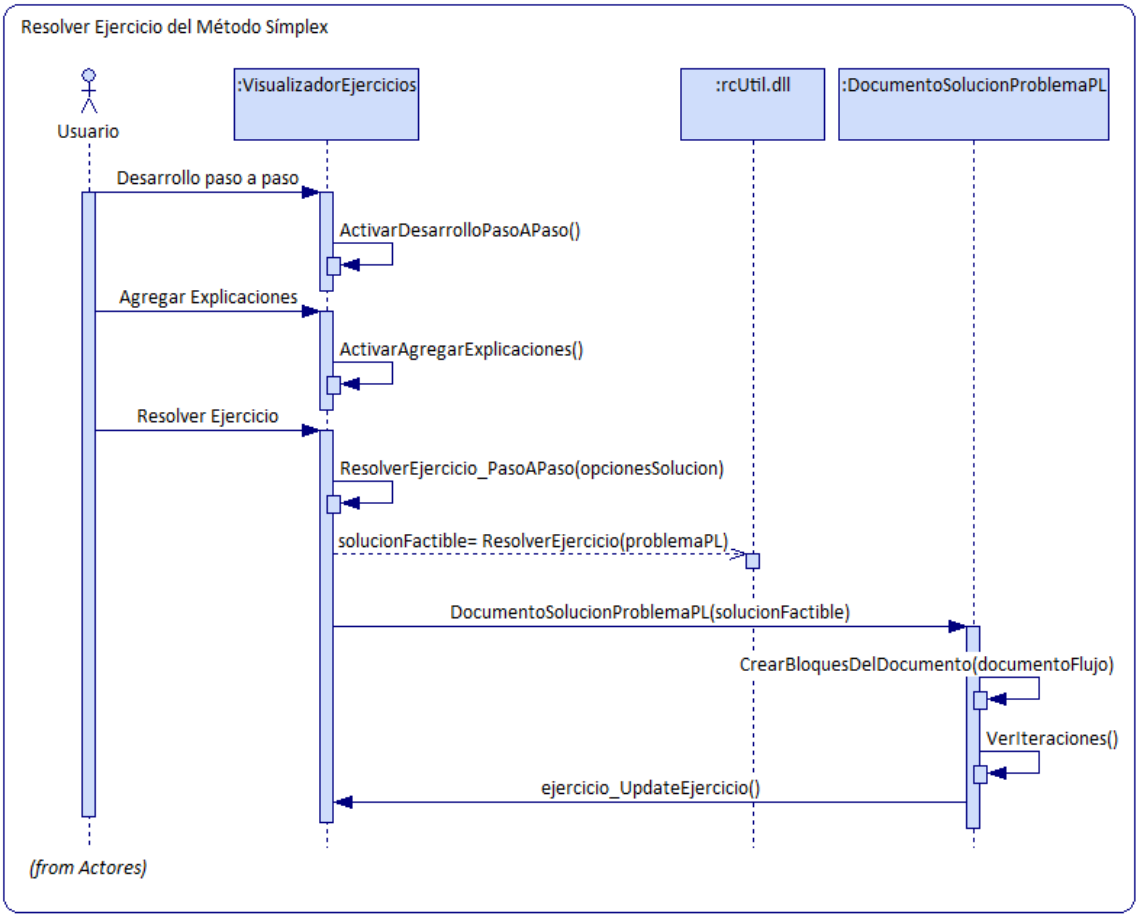
- El sistema creará el ejercicio con un planteamiento predeterminado para cada tipo de ejercicio.

6.2.2.9 Resolver ejercicio del método símplex

6.2.2.9.1 Diagrama de clases



6.2.2.9.2 Diagrama de secuencia



6.2.2.9.3 Flujo de sucesos – diseño

El Usuario le indica al *VisualizadorEjercicio* que desea que el ejercicio se desarrolle paso a paso. El *VisualizadorEjercicio* activa esta opción en la configuración del desarrollo. Posteriormente, el Usuario indica que desea obtener la explicación de las iteraciones y el *VisualizadorEjercicio* activa esta opción en la configuración del desarrollo del ejercicio.

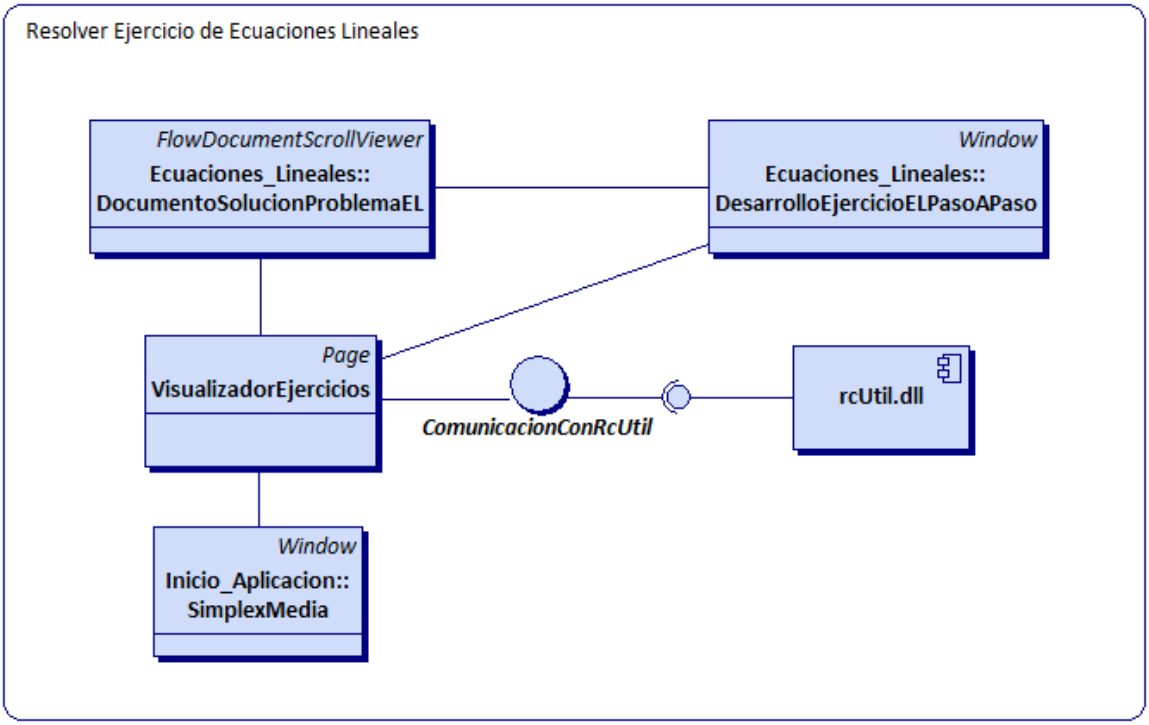
El Usuario le indica al *VisualizadorEjercicio* que desea resolver el ejercicio y se inicia la solución paso a paso. Se establece comunicación con el dll y se obtiene la solución factible del problema. A través de *DocumentoSolucionProblemaPL* se crea un documento detallado de la solución factible. Se crean los bloques de información y se visualizan las iteraciones. El contenido del desarrollo se muestra en pantalla.

6.2.2.9.4 Requisitos de la implementación

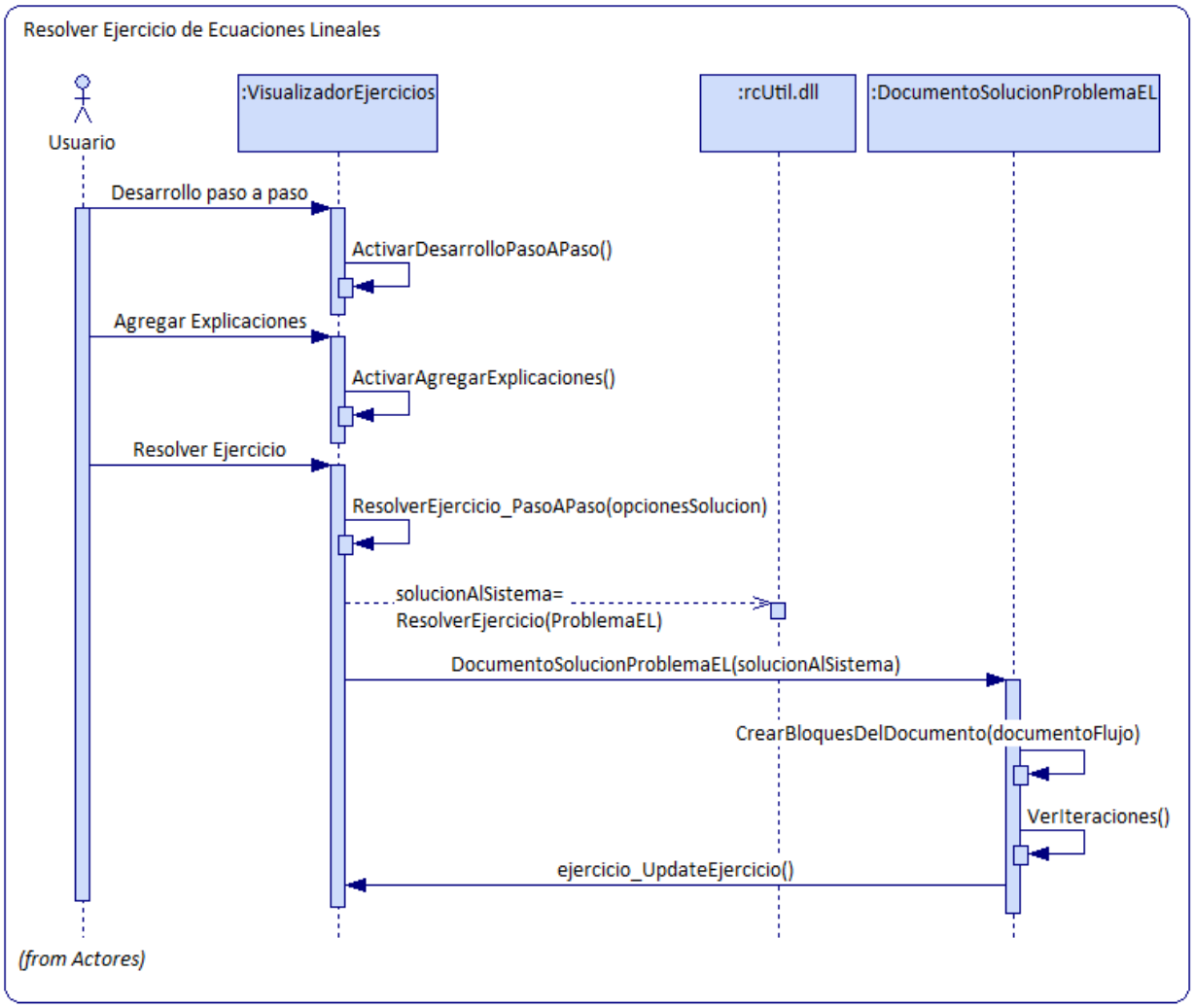
- El desarrollo del ejercicio tendrá vínculos a materiales que proporcionen las bases de los procedimientos.

6.2.2.10 Resolver ejercicio de ecuaciones lineales

6.2.2.10.1 Diagrama de clases



6.2.2.10.2 Diagrama de secuencia



6.2.2.10.3 Flujo de sucesos – diseño

El Usuario le indica al *VisualizadorEjercicio* que desea que el ejercicio se desarrolle paso a paso. El *VisualizadorEjercicio* activa esta opción en la configuración del desarrollo. Posteriormente, el Usuario indica que desea obtener la explicación de las iteraciones y el *VisualizadorEjercicio* activa esta opción en la configuración del desarrollo del ejercicio.

El Usuario le indica al *VisualizadorEjercicio* que desea resolver el ejercicio y se inicia la solución paso a paso. Se establece comunicación con el dll y se obtiene la solución al sistema del problema. A través de *DocumentoSolucionProblemaEL* se crea un documento

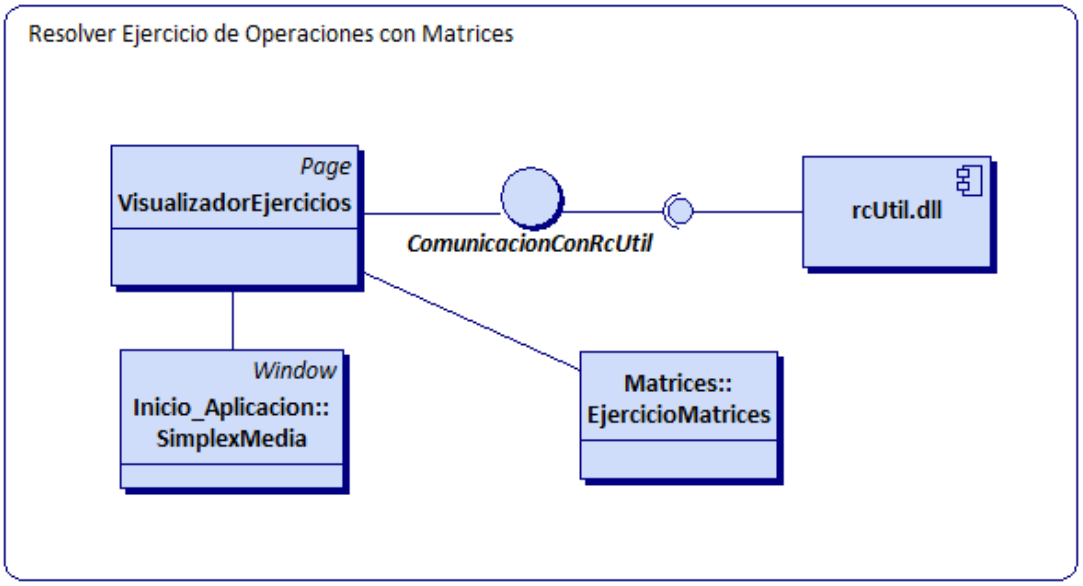
detallado de la solución al sistema. Se crean los bloques de información y se visualizan las iteraciones. El contenido del desarrollo se muestra en pantalla.

6.2.2.10.4 Requisitos de la implementación

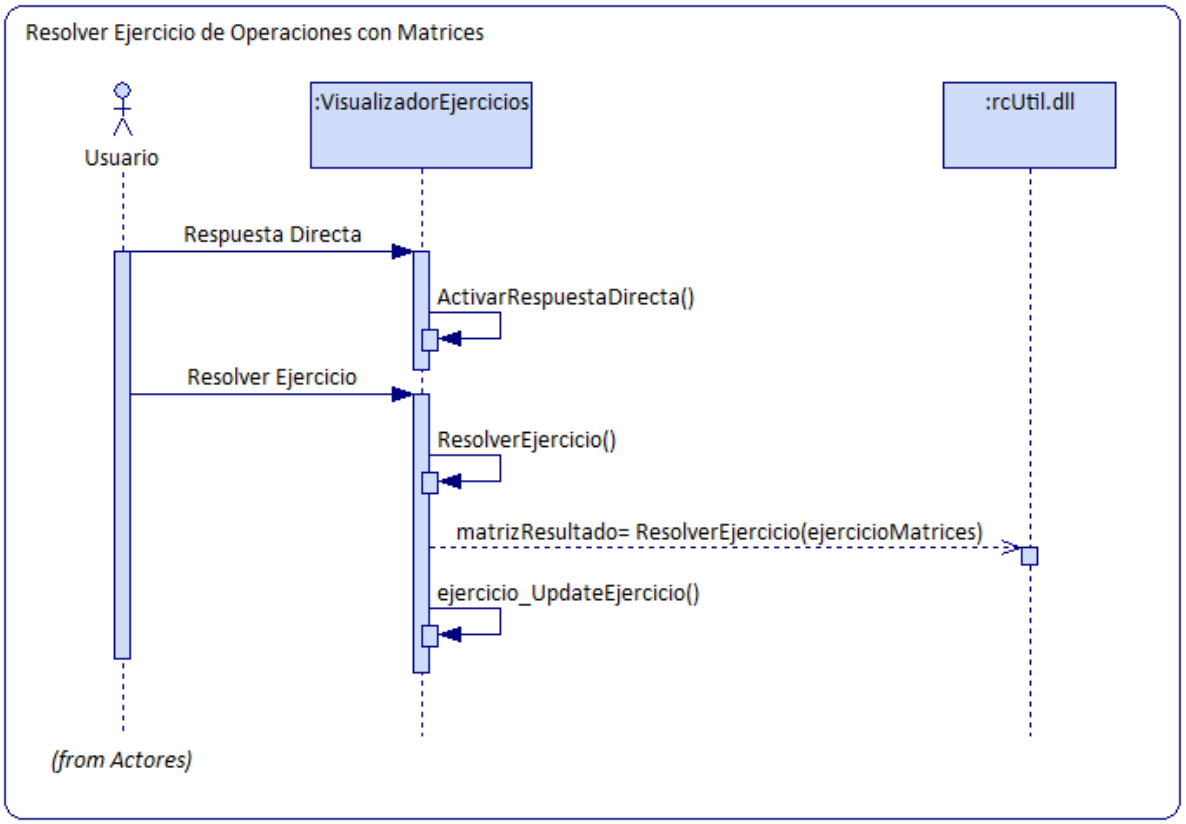
- El desarrollo del ejercicio tendrá vínculos a materiales que proporcionen las bases de los procedimientos.

6.2.2.11 Resolver ejercicio de operaciones con matrices

6.2.2.11.1 Diagrama de clases



6.2.2.11.2 Diagrama de secuencia



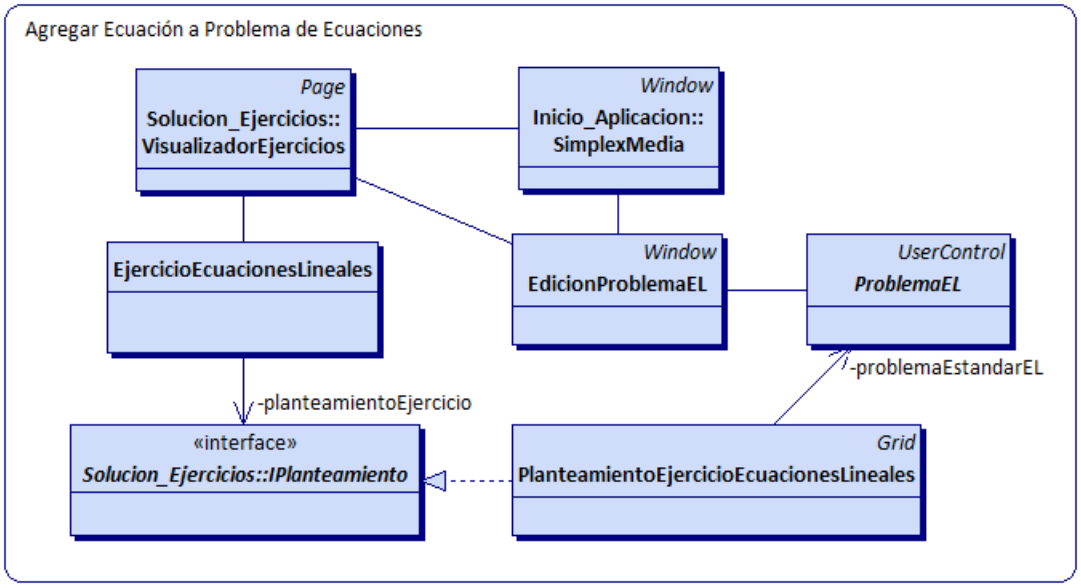
6.2.2.11.3 Flujo de sucesos – diseño

El Usuario le indica al *VisualizadorEjercicio* que desea que el ejercicio se desarrolle de forma directa. El *VisualizadorEjercicio* activa esta opción en la configuración del desarrollo.

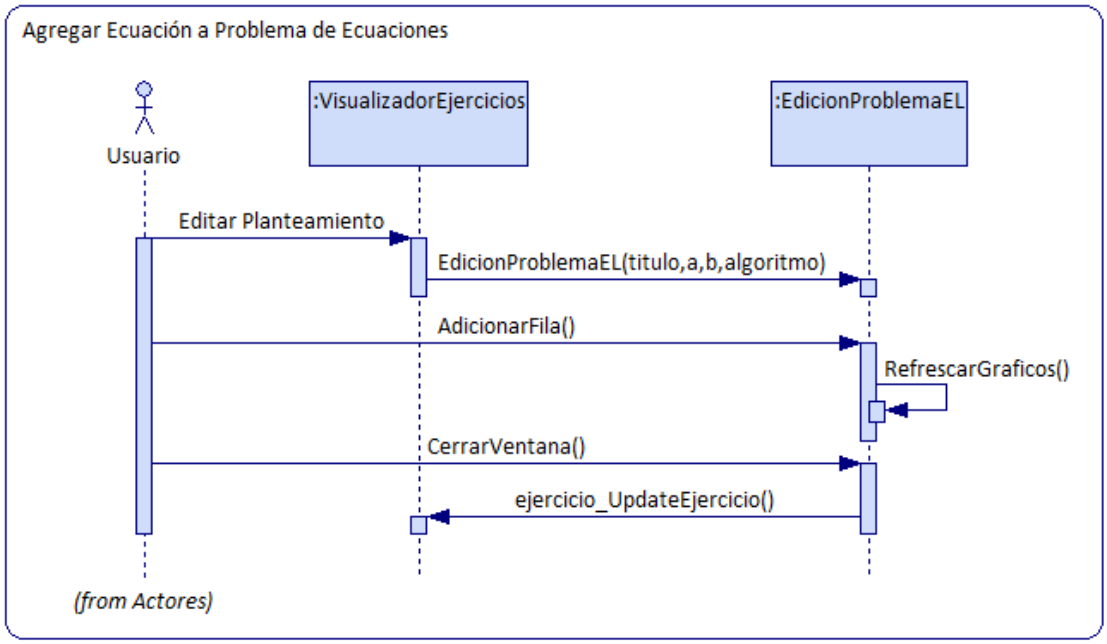
El Usuario le indica al *VisualizadorEjercicio* que desea resolver el ejercicio y se inicia la solución directa. Se establece comunicación con el dll y se obtiene la matriz resultado. Se muestra la matriz resultado en pantalla.

6.2.2.12 Agregar ecuación a problema de ecuaciones

6.2.2.12.1 Diagrama de clases



6.2.2.12.2 Diagrama de secuencia



6.2.2.12.3 Flujo de sucesos – diseño

El Usuario indica al *VisualizadorEjercicios* que desea Editar Planteamiento. El *VisualizadorEjercicios* abre la ventana de *EdicionProblemaPL* y le envía los datos del problema que se está visualizando.

El usuario le ordena a *EdicionProblemaPL* adicionar una fila (o ecuación). *EdicionProblemaPL* actualiza los gráficos con la nueva ecuación. El usuario cierra la ventana *EdicionProblemaPL* y *VisualizadorEjercicios* recibe el ejercicio modificado y lo actualiza en pantalla.

6.2.2.12.4 Requisitos de la implementación

- Cada que se adicione una ecuación al sistema de ecuaciones lineales, se debe crear una ecuación con valores predeterminados y con el mismo número de variables de las ecuaciones anteriores.

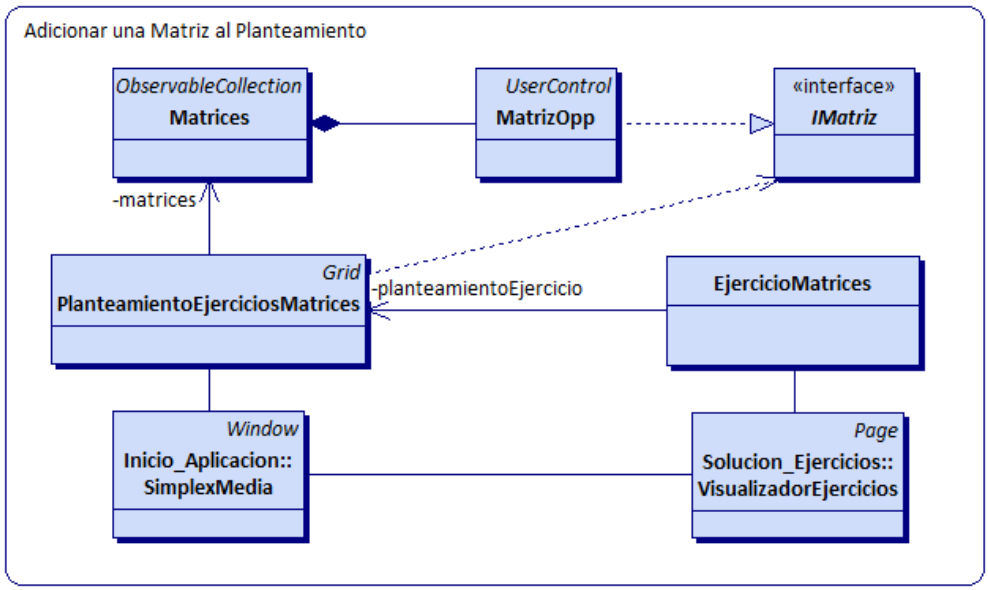
Nota: Aunque esta colaboración trata de cómo agregar una restricción a un ejercicio de Ecuaciones Lineales, también es posible quitar ecuaciones de la misma forma que se quita una restricción de un problema del Método Símplex. Y siguiendo la misma lógica se adicionan y quitan variables del sistema de ecuaciones.

También es posible modificar los valores de los coeficientes de las variables de las ecuaciones siguiendo el mismo lineamiento de las matrices.

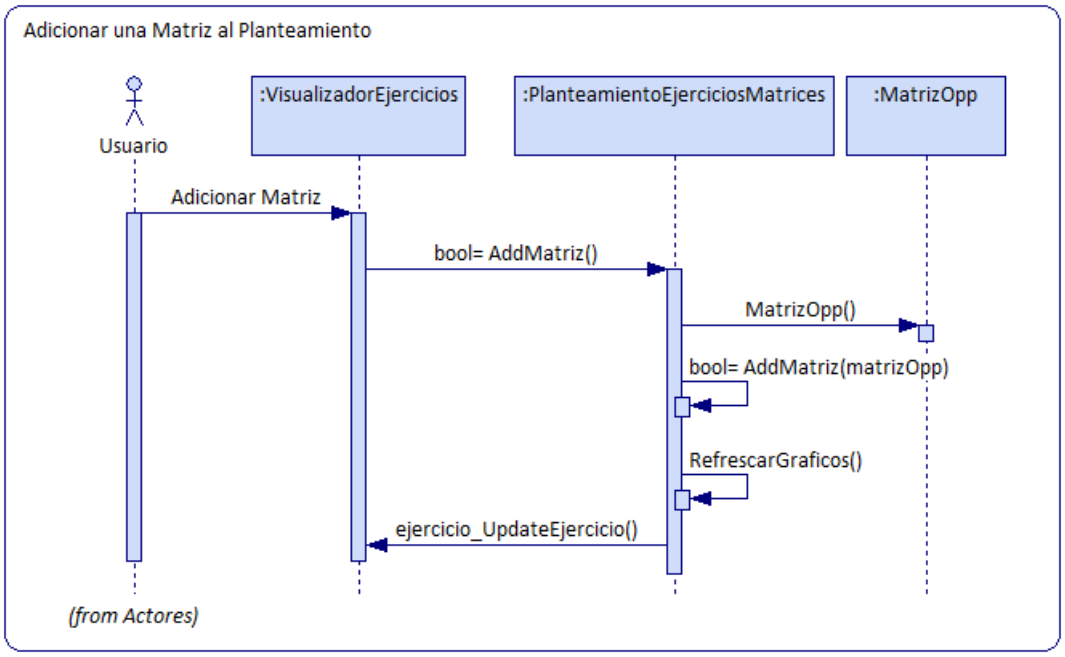
Véase la sección 7.2.2.17 acerca de Quitar Restricciones de un Problema PL.
Véase la sección 7.2.2.14 acerca de Cambiar los valores de una Matriz.

6.2.2.13 Adicionar una matriz al planteamiento

6.2.2.13.1 Diagrama de clases



6.2.2.13.2 Diagrama de secuencia



6.2.2.13.3 Flujo de sucesos – diseño

El Usuario le indica al *VisualizadorEjercicios* que desea adicionar una matriz al planteamiento del ejercicio de Operaciones con Matrices.

El *VisualizadorEjercicios* le indica a *PlanteamientoEjerciciosMatrices* que debe adicionar una matriz.

PlanteamientoEjerciciosMatrices crea una instancia de *MatrizOpp* y la adiciona al listado de matrices del planteamiento. Se refrescan los gráficos del planteamiento y del *VisualizadorEjercicios*.

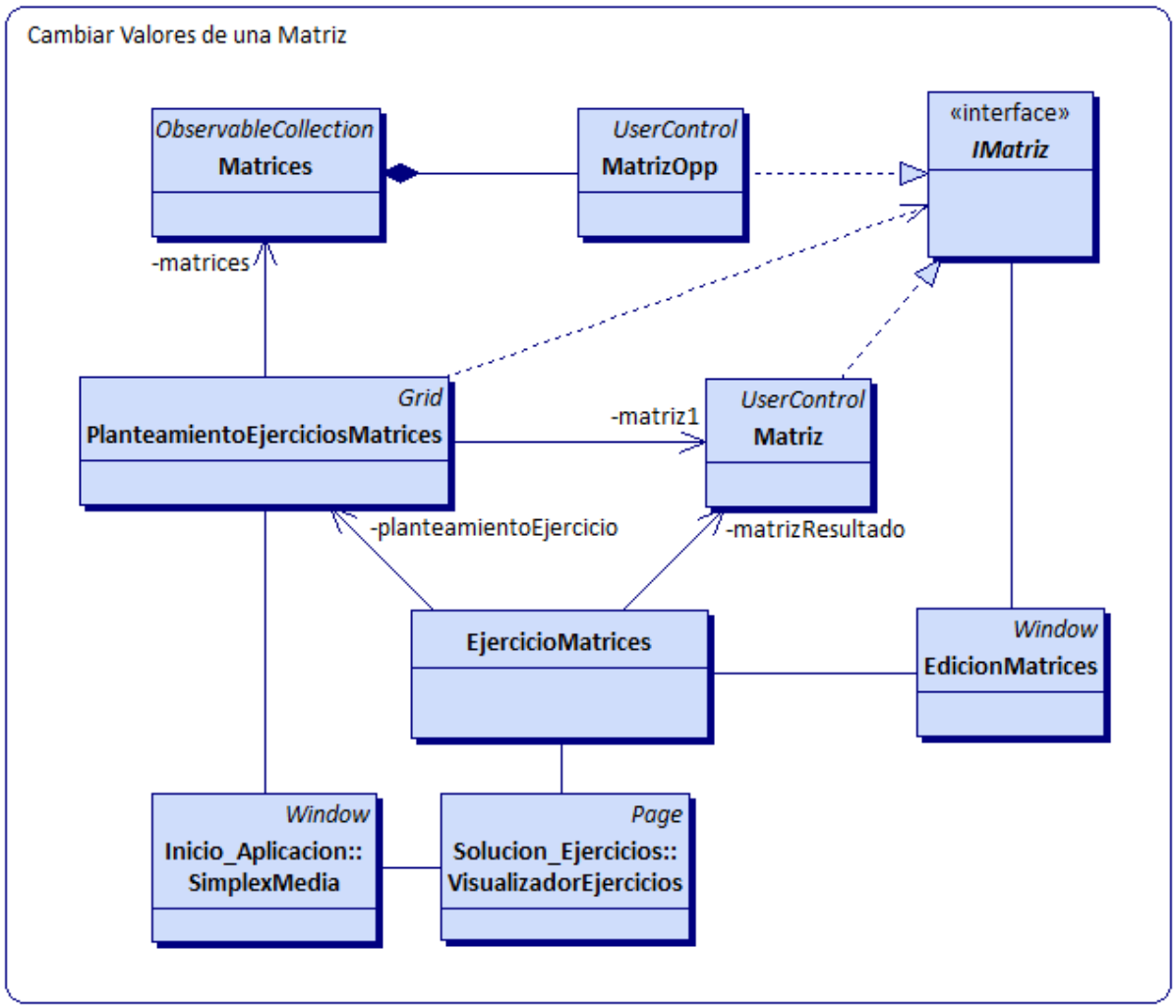
6.2.2.13.4 Requisitos de la implementación

- En la adición de nuevas matrices para un planteamiento de ejercicio con matrices, se deben crear matrices con un tamaño y valores predeterminados.

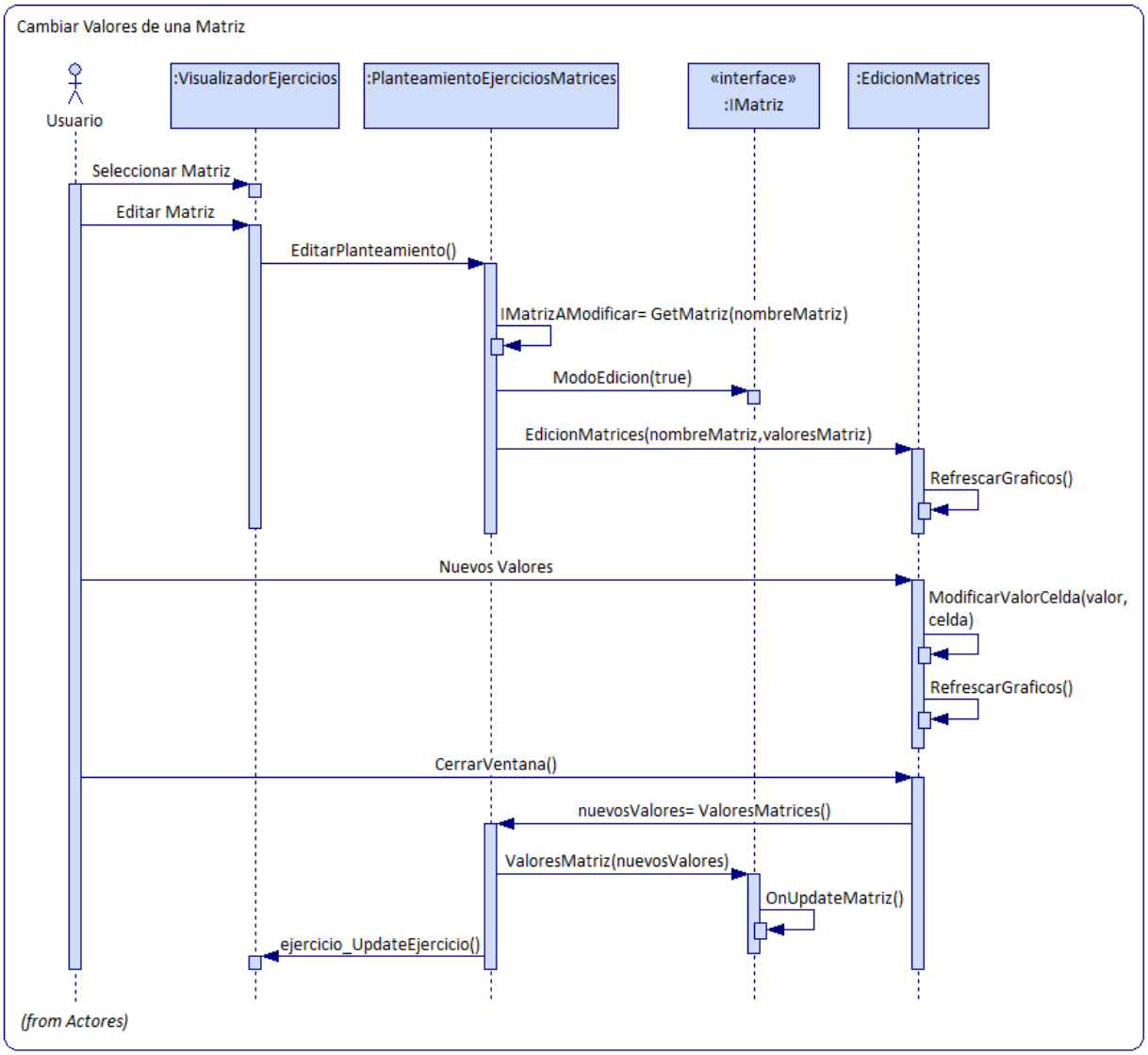
Nota: Tal como se puede adicionar una matriz a un planteamiento de un ejercicio de Operaciones con Matrices, será posible quitar una de las matrices que componen el planteamiento. El procedimiento será el mismo con la excepción de que no se creará una nueva matriz.

6.2.2.14 Cambiar valores de una matriz

6.2.2.14.1 Diagrama de clases



6.2.2.14.2 Diagrama de secuencia



6.2.2.14.3 Flujo de sucesos – diseño

El Usuario selecciona una matriz y le indica al sistema que desea editarla. *VisualizadorEjercicio* pasa esa solicitud a *PlanteamientoEjerciciosMatrices*.

PlanteamientoEjerciciosMatrices identifica cual es la matriz que ha de ser modificada y cambia el estado de esta *Matriz* a un estado de edición que sea visible al usuario. *PlanteamientoEjerciciosMatrices* abre a *EdicionMatrices* y le indica el nombre y datos de la

Matriz que de desea editar. *EdicionMatrices* realiza los gráficos respectivos y permite que el Usuario ingrese los nuevos valores.

Se modifican las celdas editadas por el Usuario. El Usuario cierra la ventada de edición y se pasan los nuevos valores a *PlanteamientoEjerciciosMatrices* para que el los registre en la *Matriz* que se está editando. Finalmente se actualizan los valores de la visualización del ejercicio.

6.2.2.14.4 Requisitos de la implementación

- *EdicionMatrices* debe contar con botones que permitan generar matrices con ciertos números especiales, como los que componen una Matriz Identidad.

- *EdicionMatrices* debe permitir el ingreso de expresiones aritméticas en el momento de cambiar los valores de una celda de la matriz. Por ejemplo: la suma, resta o multiplicación de varios valores , etc.

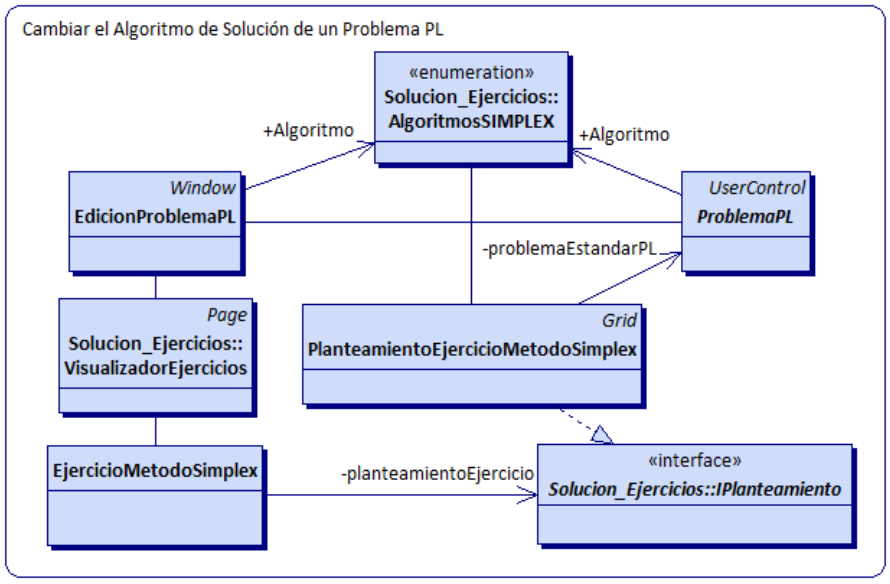
Nota: Tal como se procede en los casos de uso encargados de adicionar ecuaciones a un problema de Ecuaciones Lineales y quitar una restricción de un problema del Método Símplex, en las matrices será posible adicionar y quitar filas y/o columnas.

Véase la sección 7.2.2.12 acerca de Agregar una Ecuación a un Problema EL.
Véase la sección 7.2.2.17 acerca de Quitar Restricciones de un problema PL.

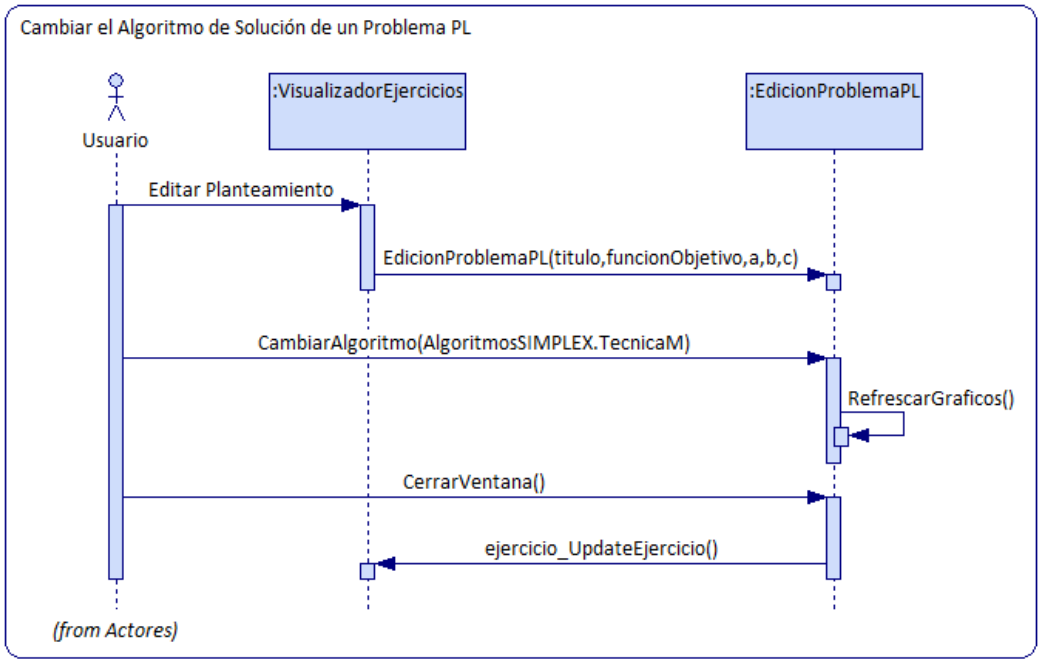
Esta misma lógica se usa para cambiar los valores de los coeficientes de las variables de problemas de Ecuaciones Lineales y de problemas del Método Símplex.

6.2.2.15 Cambiar el algoritmo de solución de un problema PL

6.2.2.15.1 Diagrama de clases



6.2.2.15.2 Diagrama de secuencia



6.2.2.15.3 Flujo de sucesos – diseño

El Usuario indica al *VisualizadorEjercicos* que desea editar el planteamiento de un problema de programación lineal. El *VisualizadorEjercicos* abre la ventana de *EdicionProblemaPL* y le envía los datos del problema actual.

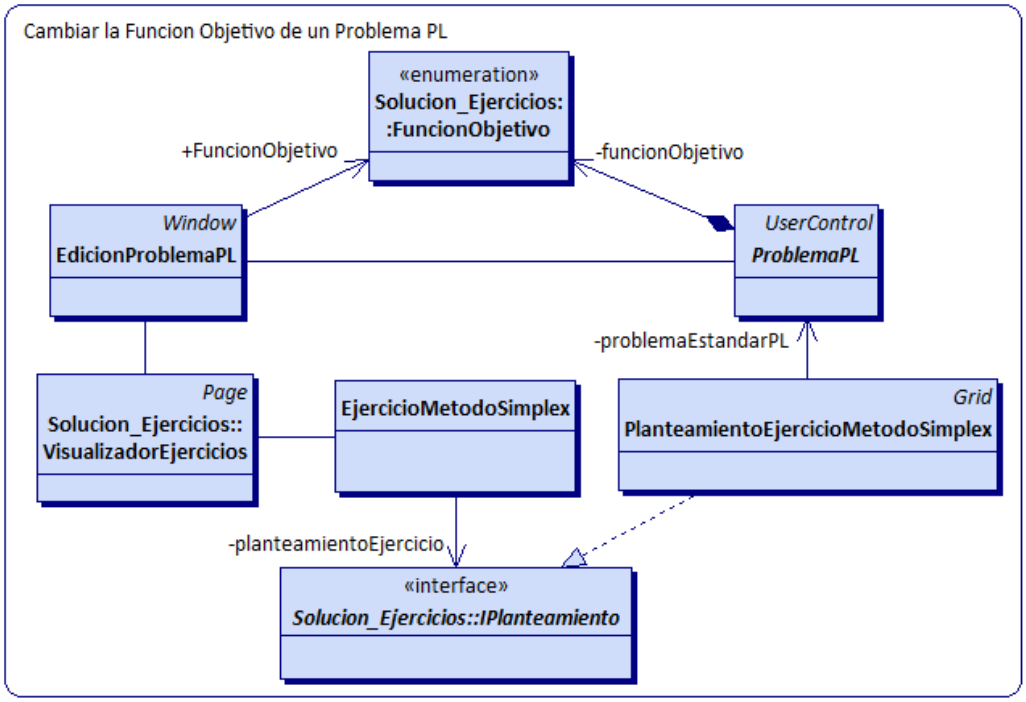
El Usuario indica a *EdicionProblemaPL* que desea cambiar el Algoritmo de Solución del problema. *EdicionProblemaPL* actualiza los gráficos con la información. El Usuario cierra la ventana *EdicionProblemaPL* y el *VisualizadorEjercicos* recibe el ejercicio modificado y lo actualiza en pantalla.

6.2.2.15.4 Requisitos de la implementación

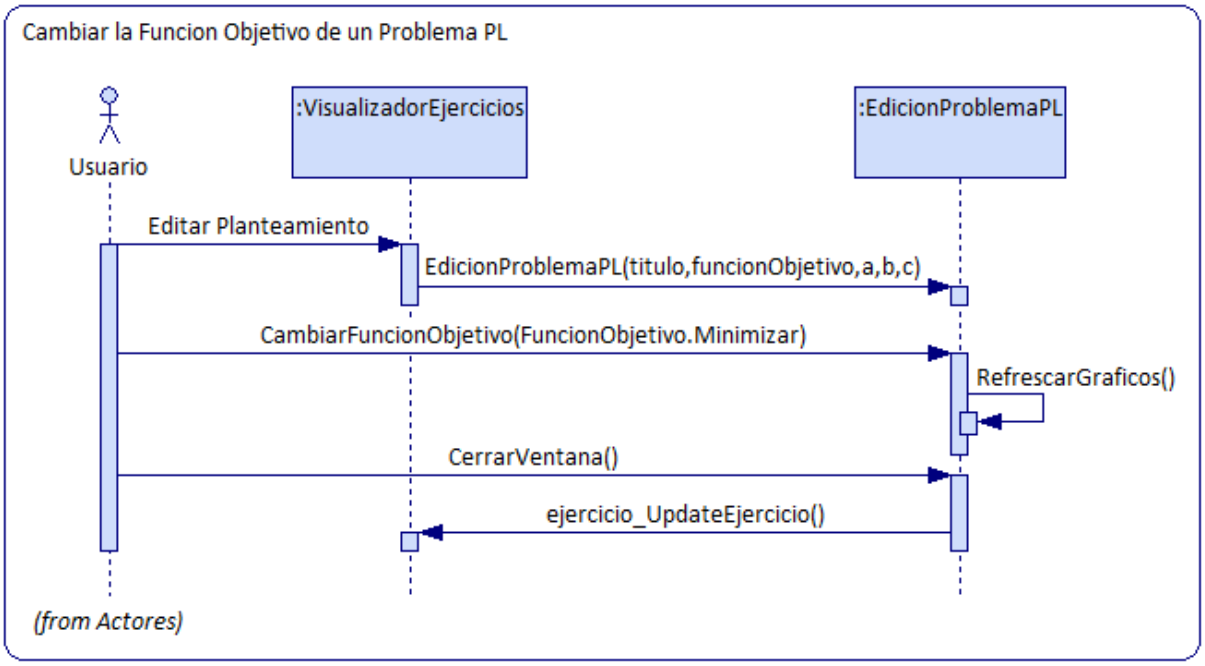
- El sistema validará que el algoritmo seleccionado pueda resolver el planteamiento actual del problema y le informará al Usuario.

6.2.2.16 Cambiar la función objetivo de un problema PL

6.2.2.16.1 Diagrama de clases



6.2.2.16.2 Diagrama de secuencia



6.2.2.16.3 Flujo de sucesos – diseño

El Usuario indica al *VisualizadorEjercicos* que desea editar el planteamiento de un problema de programación lineal. El *VisualizadorEjercicos* abre la ventana de *EdicionProblemaPL* y le envía los datos del problema actual.

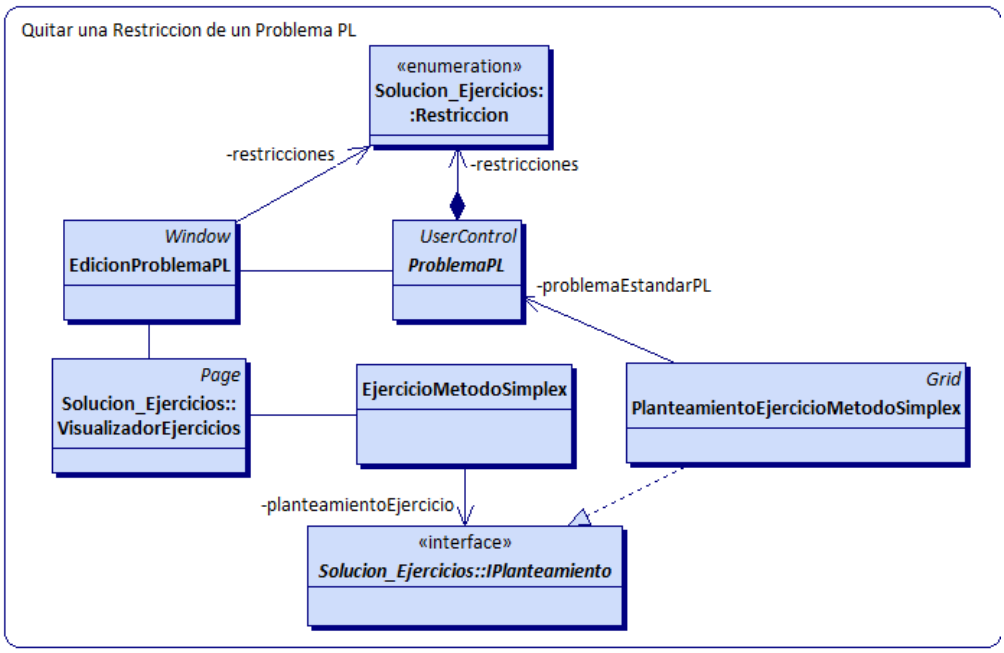
El Usuario indica a *EdicionProblemaPL* que desea cambiar la Función Objetivo del problema. *EdicionProblemaPL* actualiza los gráficos con la información.

El Usuario cierra la ventana *EdicionProblemaPL* y el *VisualizadorEjercicos* recibe el ejercicio modificado y lo actualiza en pantalla.

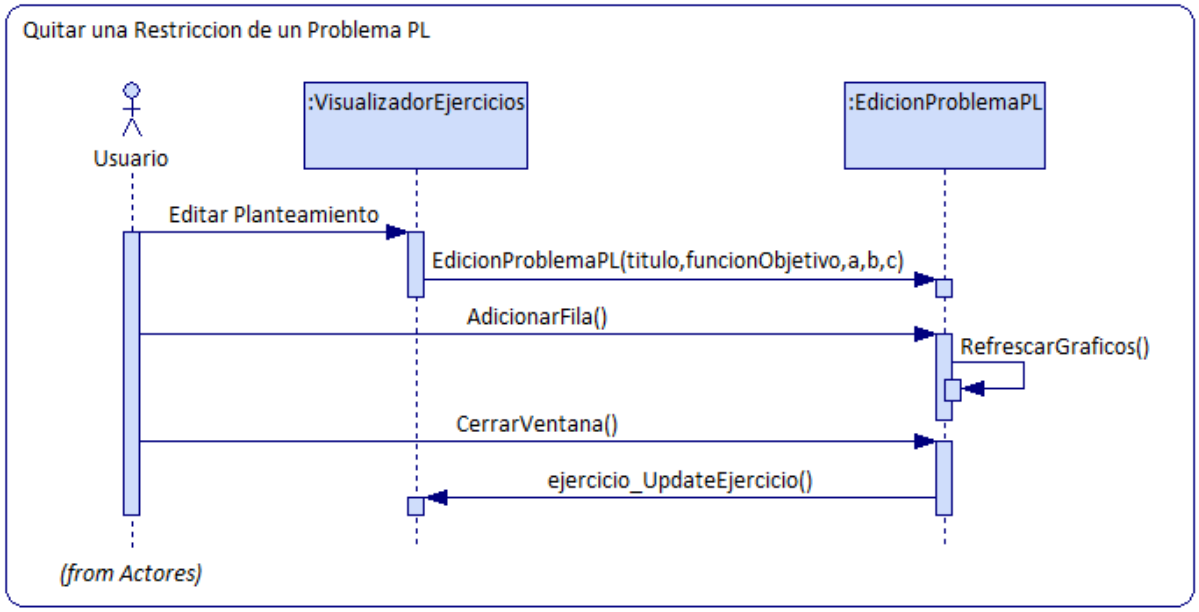
Nota: De forma similar al cambio de la Función Objetivo, se pueden cambiar los signos de relación de las restricciones del problema del Método Símplex.

6.2.2.17 Quitar una restricción de un problema PL

6.2.2.17.1 Diagrama de clases



6.2.2.17.2 Diagrama de secuencia



6.2.2.17.3 Flujo de sucesos – diseño

El Usuario indica al *VisualizadorEjercicos* que desea editar el planteamiento de un problema de programación lineal. El *VisualizadorEjercicos* abre la ventana de *EdicionProblemaPL* y le envía los datos del problema actual.

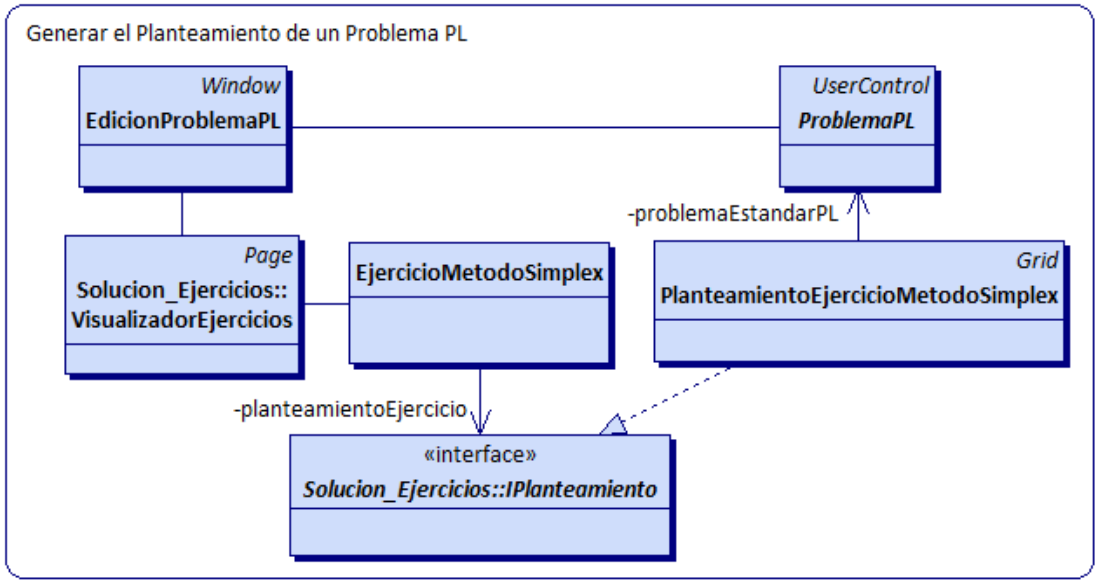
El Usuario indica a *EdicionProblemaPL* que desea quitar una restricción del problema. *EdicionProblemaPL* actualiza los gráficos. El Usuario cierra la ventana *EdicionProblemaPL* y el *VisualizadorEjercicos* recibe el ejercicio modificado y lo actualiza en pantalla.

Nota: Si el usuario desea adicionar una restricción al problema se procede de igual manera. Además, los procesos de adicionar y quitar restricciones son equivalentes a los de agregar y remover variables del problema. También, los valores de los coeficientes de las variables del problema pueden ser editados tal como se lo hace con las matrices.

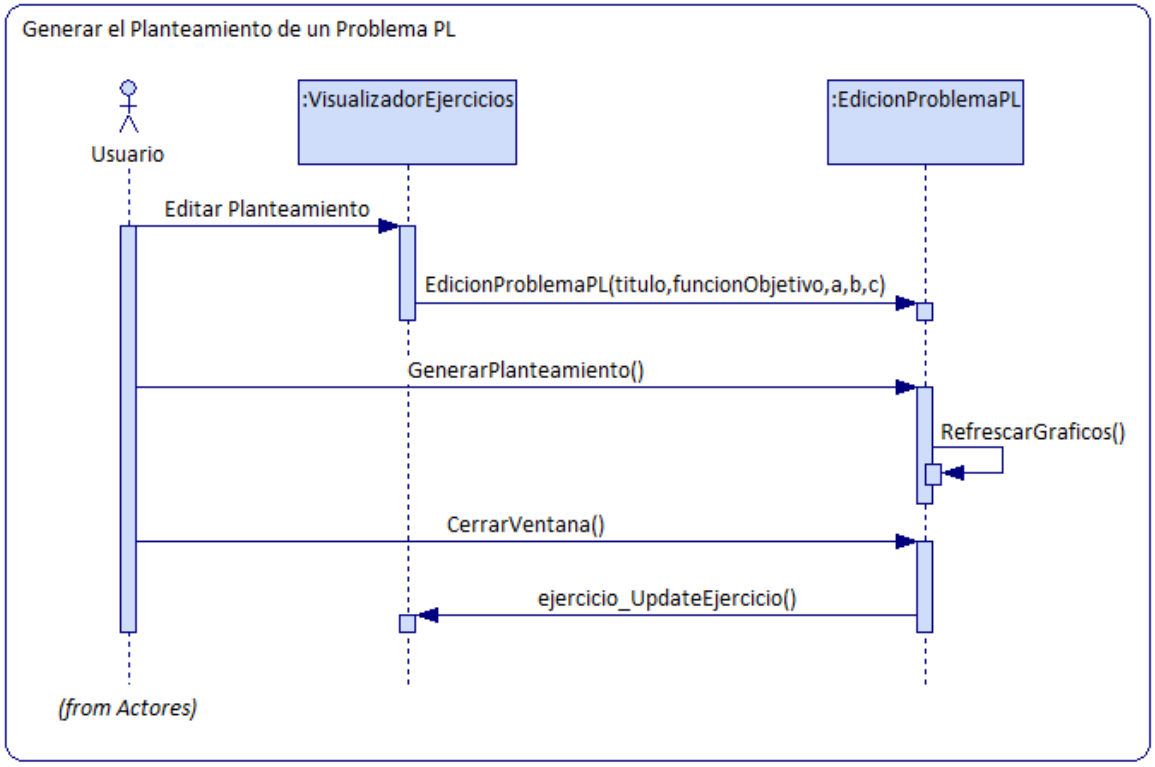
Véase la sección 7.2.2.12 acerca de agregar una ecuación a un problema EL.
Véase la sección 7.2.2.14 acerca de cambiar los valores de una matriz.

6.2.2.18 Generar el planteamiento de un problema PL

6.2.2.18.1 Diagrama de clases



6.2.2.18.2 Diagrama de secuencia



6.2.2.18.3 Flujo de sucesos – diseño

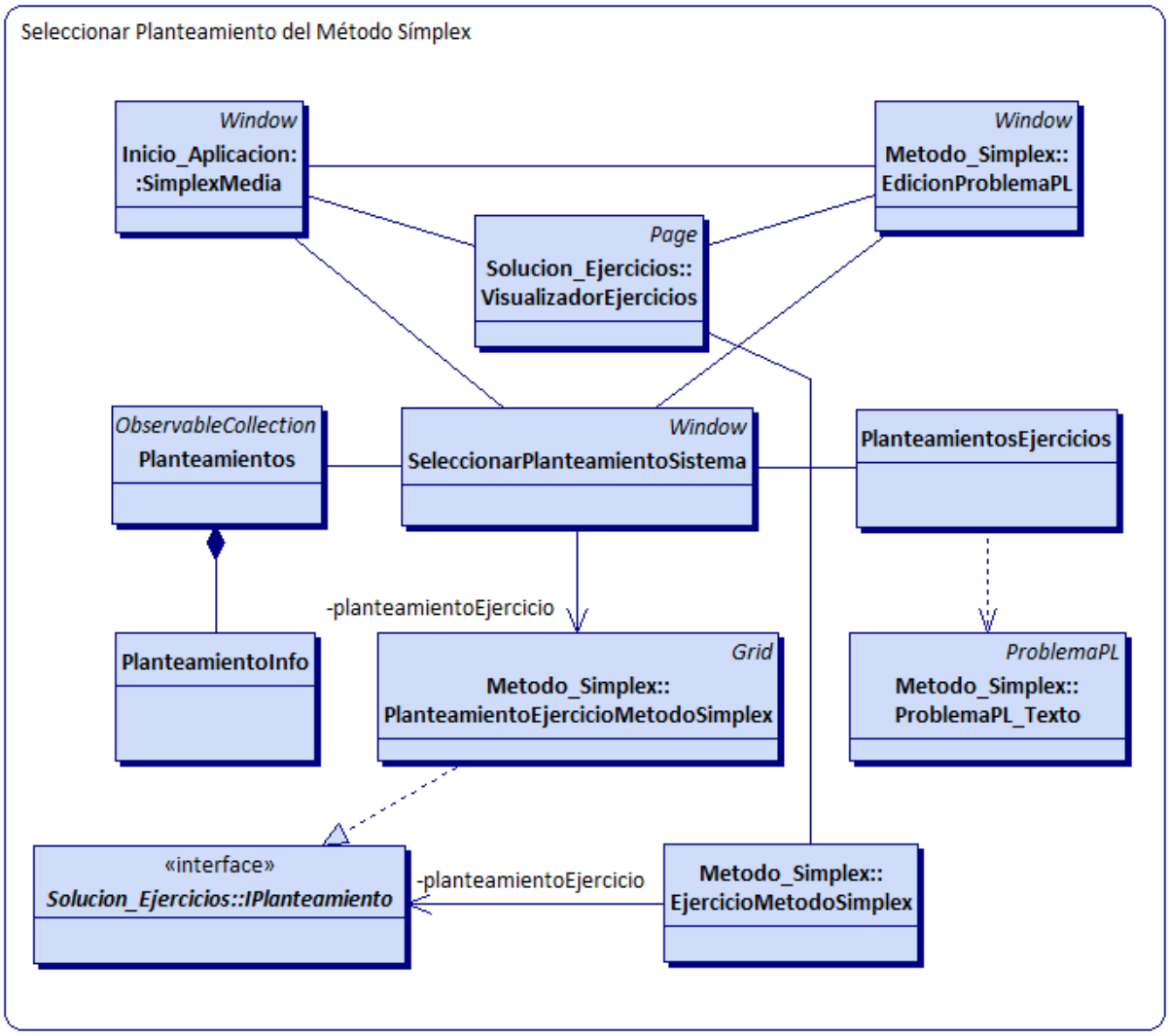
El Usuario indica al *VisualizadorEjercicos* que desea editar el planteamiento de un problema de programación lineal. El *VisualizadorEjercicos* abre la ventana de *EdicionProblemaPL* y le envía los datos del problema actual.

El Usuario indica a *EdicionProblemaPL* que desea que se genere un planteamiento de forma aleatoria. *EdicionProblemaPL* genera el nuevo planteamiento y se encarga de actualizar los gráficos. El Usuario cierra la ventana *EdicionProblemaPL* y el *VisualizadorEjercicos* recibe el ejercicio modificado y lo actualiza en pantalla.

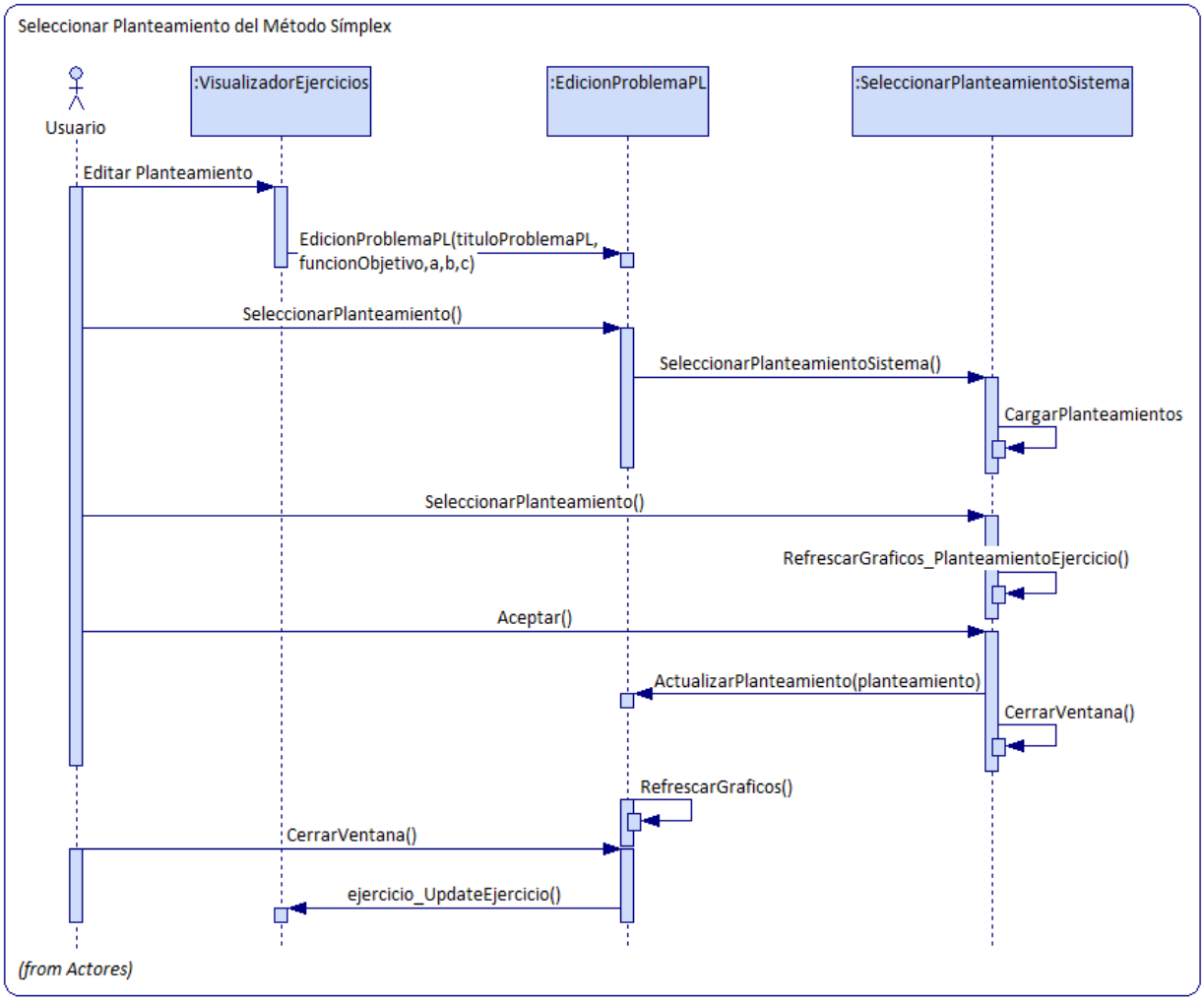
Nota: También es posible generar planteamientos aleatorios para ejercicios de Ecuaciones Lineales siguiendo este mismo lineamiento de ideas.

6.2.2.19 Seleccionar planteamiento del método símplex

6.2.2.19.1 Diagrama de clases



6.2.2.19.2 Diagrama de secuencia



6.2.2.19.3 Flujo de sucesos – diseño

El Usuario indica al *VisualizadorEjercicos* que desea editar el planteamiento de un problema de programación lineal. El *VisualizadorEjercicos* abre la ventana de *EdicionProblemaPL* y le envía los datos del problema actual.

El Usuario indica a *EdicionProblemaPL* que desea seleccionar un planteamiento del sistema. *EdicionProblemaPL* abre a *SeleccionarPlanteamientoSistema*, quien carga un listado de todos los planteamientos disponibles y los muestra al Usuario. El Usuario selecciona uno de ellos y se actualizan los datos en la pantalla de *SeleccionarPlanteamientoSistema*. El Usuario acepta la selección realizada y

SeleccionarPlanteamientoSistema para los datos a *EdicionProblemaPL* y se cierra para que se actualicen los datos dicha ventana. El Usuario cierra la ventana *EdicionProblemaPL* y el *VisualizadorEjercicos* recibe el ejercicio modificado y lo actualiza en pantalla.

6.2.2.19.4 Requisitos de la implementación

- También será posible seleccionar los planteamientos de los ejercicios explicados del módulo de Ejercicios Prácticos.

6.3 CONTRATOS DE OPERACIONES

Los contratos de operaciones definen el contenido de algunos de los métodos principales de las clases del diseño. A continuación se describirán dichos métodos u operaciones mediante diagramas de secuencia seguidos por una descripción textual de lo que realizan.

De esta manera el implementador tendrá una visión general de los que se realiza internamente al llamar a determinado método de alguna clase, cuyo nombre indique que tras él hay otros procesos complejos.

6.3.1 Operación: *Abrir módulo de información teórica*

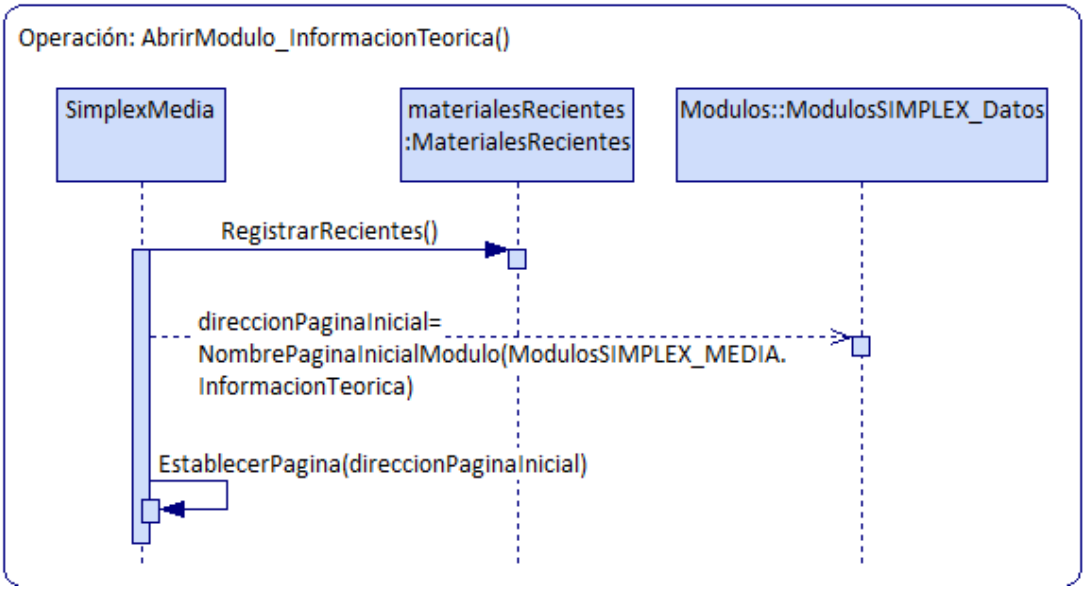
Nombre de la Operación: *AbrirModulo_InformacionTeorica()*
Parámetros: *Ninguno*
Retorno: *Ninguno*
Clases en las que aparece:

- SimplexMedia (Sección 7.1.1.1)

Realizaciones de casos de uso – diseño en las que aparece:

- Observar el Contenido de un Tema (Sección 7.2.2.1)

Diagrama de secuencia:



Descripción del diagrama: *Simplex Media ordena a MaterialesRecientes que registre los materiales recientes que aún no han sido registrados. Símplex Media trae desde ModulosSIMPLEX Datos la dirección de la página de bienvenida del módulo Información Teórica y la muestra en pantalla. De la misma forma se abrirán los módulos de Ejercicios Prácticos y Solución de Ejercicios.*

6.3.2 Operación: *Ubicar listado de materiales recientes*

Nombre de la Operación:

UbicarListadoMaterialesRecientes(materialesRecientes: MaterialesRecientes)

Parámetros:

- materialesRecientes de tipo MaterialesRecientes: contiene el listado de materiales (temas, ejercicios explicados o ejercicios del usuario) recientes que se desea ubicar en pantalla.

Retorno: *Ninguno*

Clases en las que aparece:

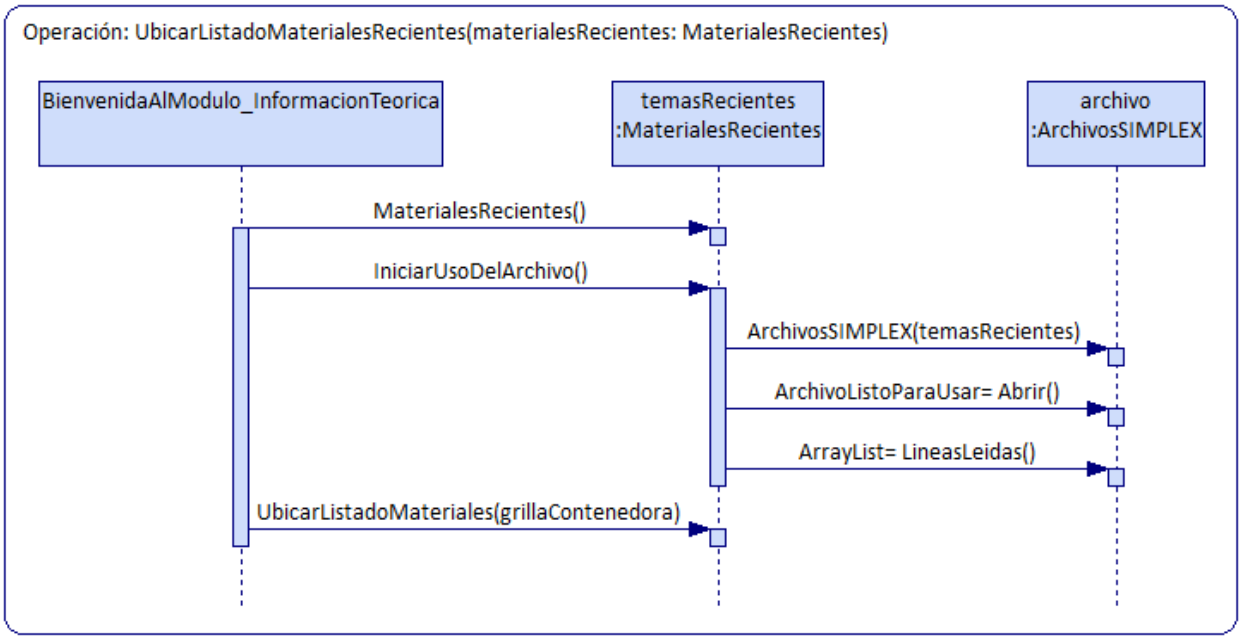
- IBienvenidaAlModulo (Sección 7.1.1.2)
- BienvenidaAlModulo_InformacionTeorica (Sección 7.1.1.3)
- BienvenidaAlModulo_EjerciciosPracticos (Sección 7.1.1.4)
- BienvenidaAlModulo_SolucionEjercicios (Sección 7.1.1.5)

Realizaciones de casos de uso – diseño en las que aparece:

- Observar el Contenido de un Tema (Sección 7.2.2.1)
- Observar el Contenido de un Ejercicio Explicado (Sección 7.2.2.2)
- Abrir Ejercicios Recientes del Usuario (Sección 7.2.2.6)

Descripción del diagrama: *La página de Bienvenida al Módulo crea un listado de temas recientes e inicia la apertura de los registros del archivo de materiales recientes. Materiales Recientes crea una instancia de ArchivoSIMPLEX y le pide que abra el archivo de temas recientes. Posteriormente, se saca el contenido del archivo y se los ubica en la grilla contenedora de los materiales recientes.*

Diagrama de secuencia:



6.3.3 Operación: *Nuevo ejercicio*

Nombre de la Operación:

NuevoEjercicio (tipoEjercicio: TiposEjercicios)

Parámetros:

- tipoEjercicio de tipo TiposEjercicios: especifica el tipo de ejercicio que se va a crear.

Retorno: *Ninguno*

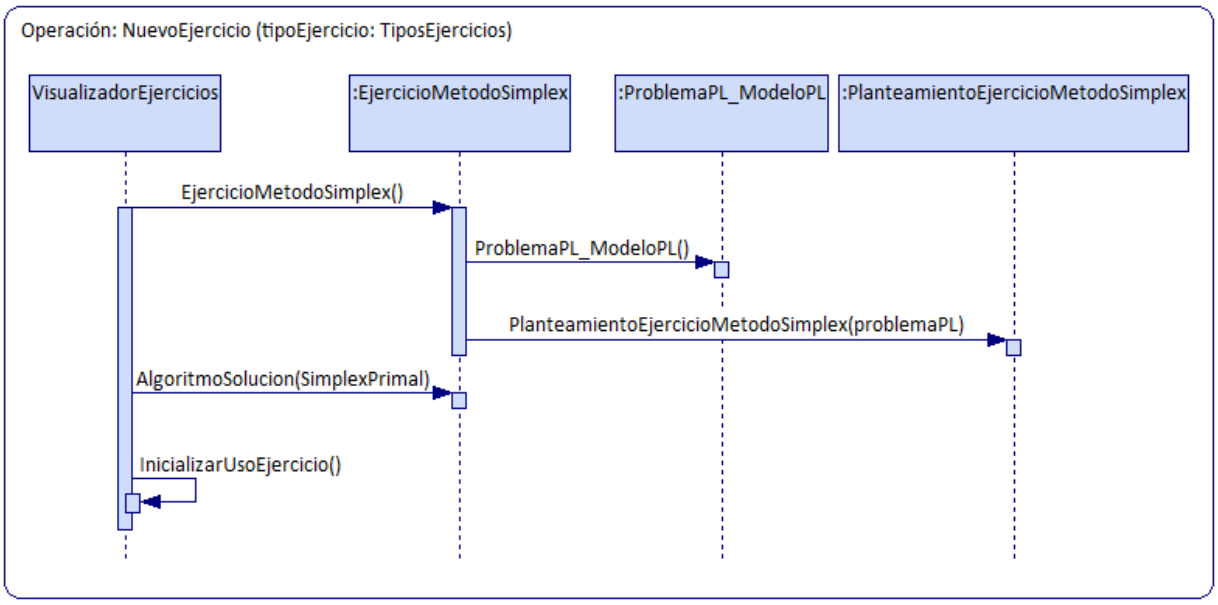
Clases en las que aparece:

- VisualizadorEjercicios (Sección 7.1.1.5)

Realizaciones de casos de uso – diseño en las que aparece:

- Nuevo Ejercicio (Sección 7.2.2.8)

Diagrama de secuencia:



Descripción del diagrama: *VisualizadorEjercicios analiza el tipo de ejercicio y según el tipo de ejercicio se escoge la clase adecuada para su creación. En este caso ya se hizo la selección y se ha determinado que el ejercicio es del método Símplex, por esto se crea una nueva instancia de la clase EjercicioMetodoSimplex. Al crear la instancia se crea un muevo problema de programación lineal que será enlazado a un nuevo planteamiento. Posteriormente, el VisualizadorEjercicios inicia el uso del ejercicio, con lo cual el usuario podrá manipular el ejercicio.*

6.3.4 Operación: *Abrir ejercicio*

Nombre de la Operación:

AbrirEjercicio([inout] Ejercicio: IEjercicio,
[inout] archivo: ArchivosSIMPLEX, ruta: string) : bool

Parámetros:

- Ejercicio de tipo IEjercicio: contiene el último ejercicio que se ha abierto y servirá como valor de retorno para devolver el nuevo ejercicio creado. Puede ser null.
- archivo de tipo ArchivosSIMPLEX: contiene el objeto de tipo archivo que se utilizará para abrir el ejercicio del usuario. Servirá como valor de retorno para devolver el archivo físico asociado al ejercicio abierto. Puede ser null.
- ruta de tipo string: especifica la dirección o ruta del archivo físico que se desea abrir.

Retorno:

Retorna un valor bool (verdadero o falso) que indica si el archivo fue abierto o no. False para indicar que no se ha abierto, true para indicar que si se abrió y que sí se pudo crear el ejercicio a partir del archivo abierto.

Clases en las que aparece:

- GestionEjercicios (Sección 7.1.1.5)

Realizaciones de casos de uso – diseño en las que aparece:

- Abrir Ejercicio (Sección 7.2.2.5)

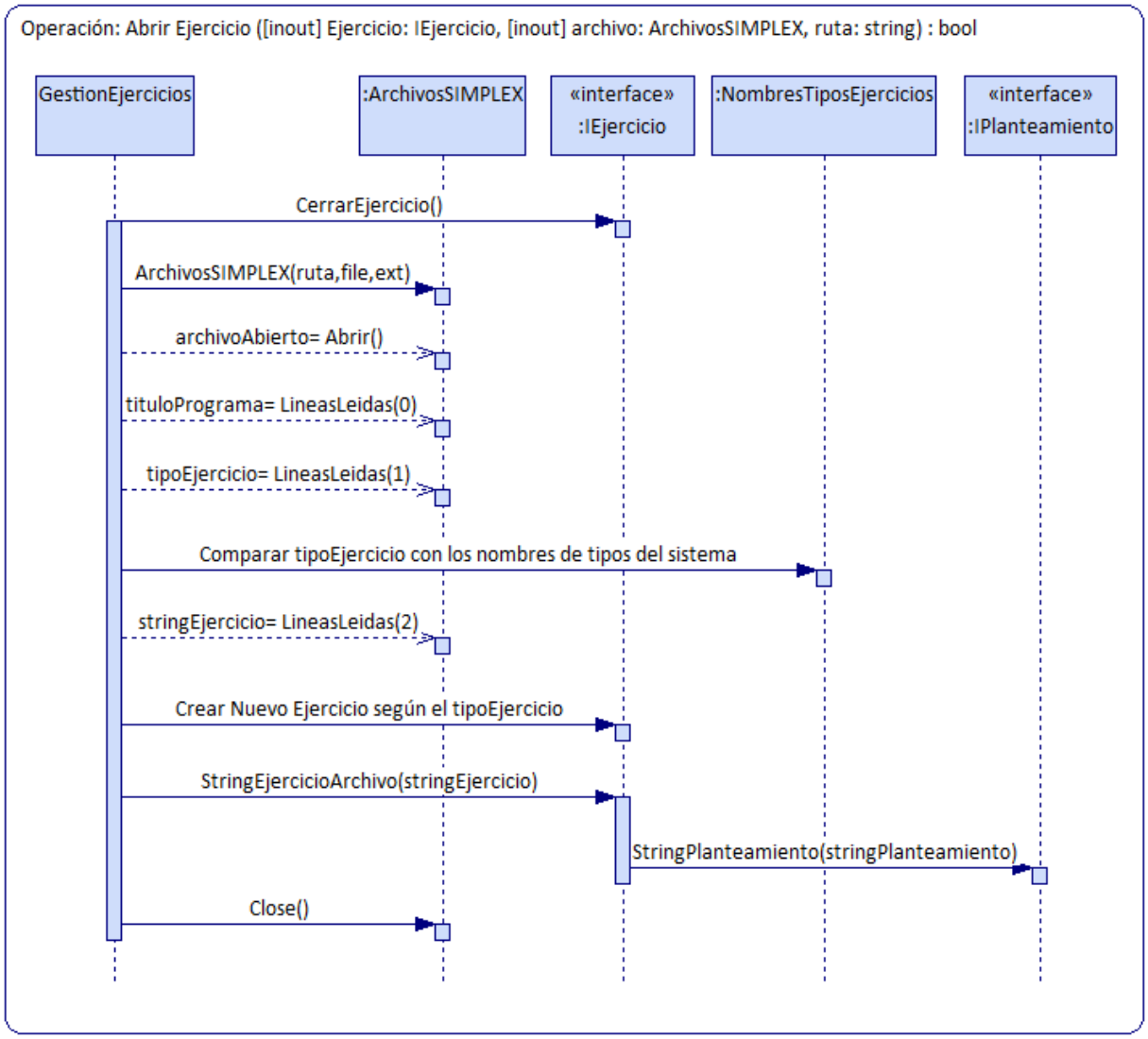
Descripción del diagrama: *GestionEjercicios analiza si el Ejercicio contiene información y en caso afirmativo, ya que existe la posibilidad de que haya un ejercicio que se encuentre abierto, lo cierra. GestionEjercicios crea una nueva instancia de ArchivoSIMPLEX enviándole la ruta del ejercicio a abrir. Se solicita la apertura física del archivo y se recibe un valor que indica si fue abierto o no. En caso que el archivo no haya sido abierto se suspende la operación y se retorna false (No se muestra en el diagrama).*

Se extrae la primera línea del archivo y se compara si es el nombre del sistema. Se extrae la segunda línea y se compara con los nombres de los tipos de ejercicios de NombresTiposEjercicios y se determina si es un tipo válido. En caso que las comparaciones anteriores sean negativas, se abandona la operación (No se muestra en el diagrama). Se extrae la tercera línea del archivo que contiene la definición del ejercicio y su planteamiento.

GestionEjercicios crea un nuevo ejercicio según el tipo de ejercicio extraído del archivo. Al ejercicio creado se le asocia la cadena que define el ejercicio y este, a su vez, descompone la cadena y entrega la porción del planteamiento al Planteamiento del Ejercicio.

GestionEjercicios le indica a ArchivosSimplex que debe cerrar el archivo físico. Se retorna true si la creación del ejercicio fue exitosa.

Diagrama de secuencia:



6.3.5 Operación: *Guardar ejercicio*

Nombre de la Operación:

GuardarEjercicio ([inout] Ejercicio: IEjercicio, ruta: string)

Parámetros:

- Ejercicio de tipo IEjercicio: contiene el último ejercicio que se ha abierto y servirá como valor de retorno para devolver el nuevo ejercicio creado. No puede ser null.
- ruta de tipo string: especifica la dirección o ruta del archivo físico que se desea abrir.

Retorno:

Retorna un valor bool (verdadero o falso) que indica si el archivo fue guardado o no. False para indicar que no se ha guardado, true para indicar que si se ha creado el archivo físico y que el ejercicio sí fue guardado.

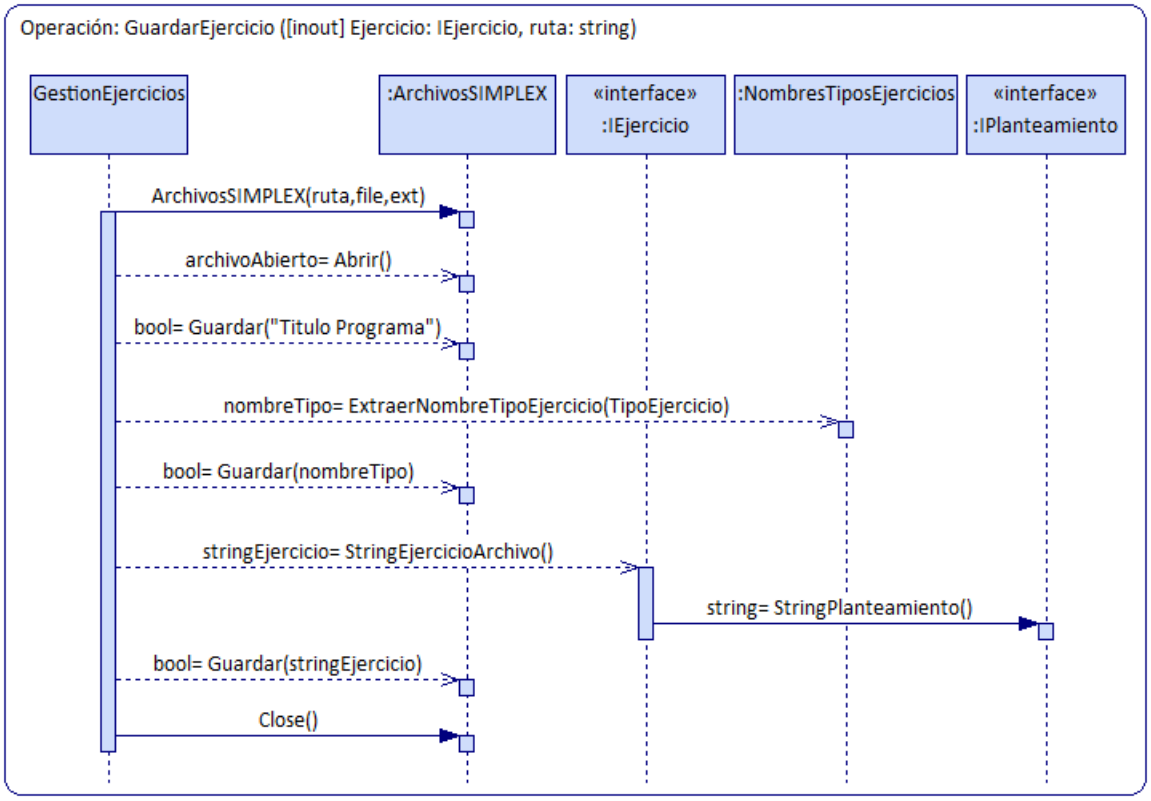
Clases en las que aparece:

- GestionEjercicios (Sección 7.1.1.5)

Realizaciones de casos de uso – diseño en las que aparece:

- Guardar Ejercicio (Sección 7.2.2.7)

Diagrama de secuencia:



Descripción del diagrama: *GestionEjercicios* crea una instancia de *ArchivoSIMPLEX con la ruta del ejercicio a guardar. Ese abre (o crea) el archivo físico. En caso que el archivo no pueda ser abierto se termina la ejecución de la operación y se retorna false (No se muestra en el diagrama).*

Si el archivo físico pudo ser creado, se guarda en él el título del programa (esto servirá como identificación para la apertura de archivos provistos por SÍMPLEX MEDIA). Por medio

de NombresTipoEjercicio se extrae el nombre del tipo de ejercicio del Ejercicio que se está guardando. Se guarda el tipo del ejercicio en el archivo (esto servirá para volver a crear uno del mismo tipo en el momento de la apertura del archivo).

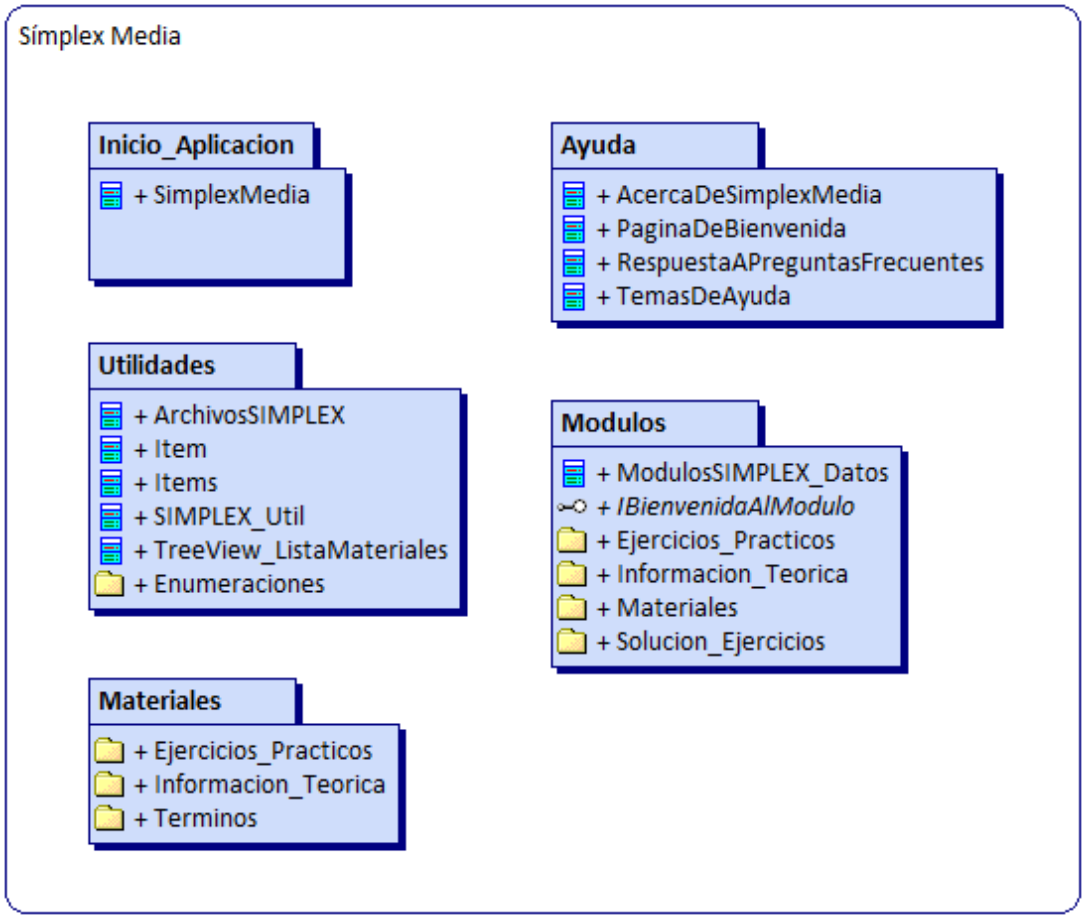
Se solicita al Ejercicio la cadena que define su planteamiento e información adicional. Se guarda la cadena con la definición del ejercicio y se cierra el archivo físico.

6.4 PAQUETES DEL DISEÑO

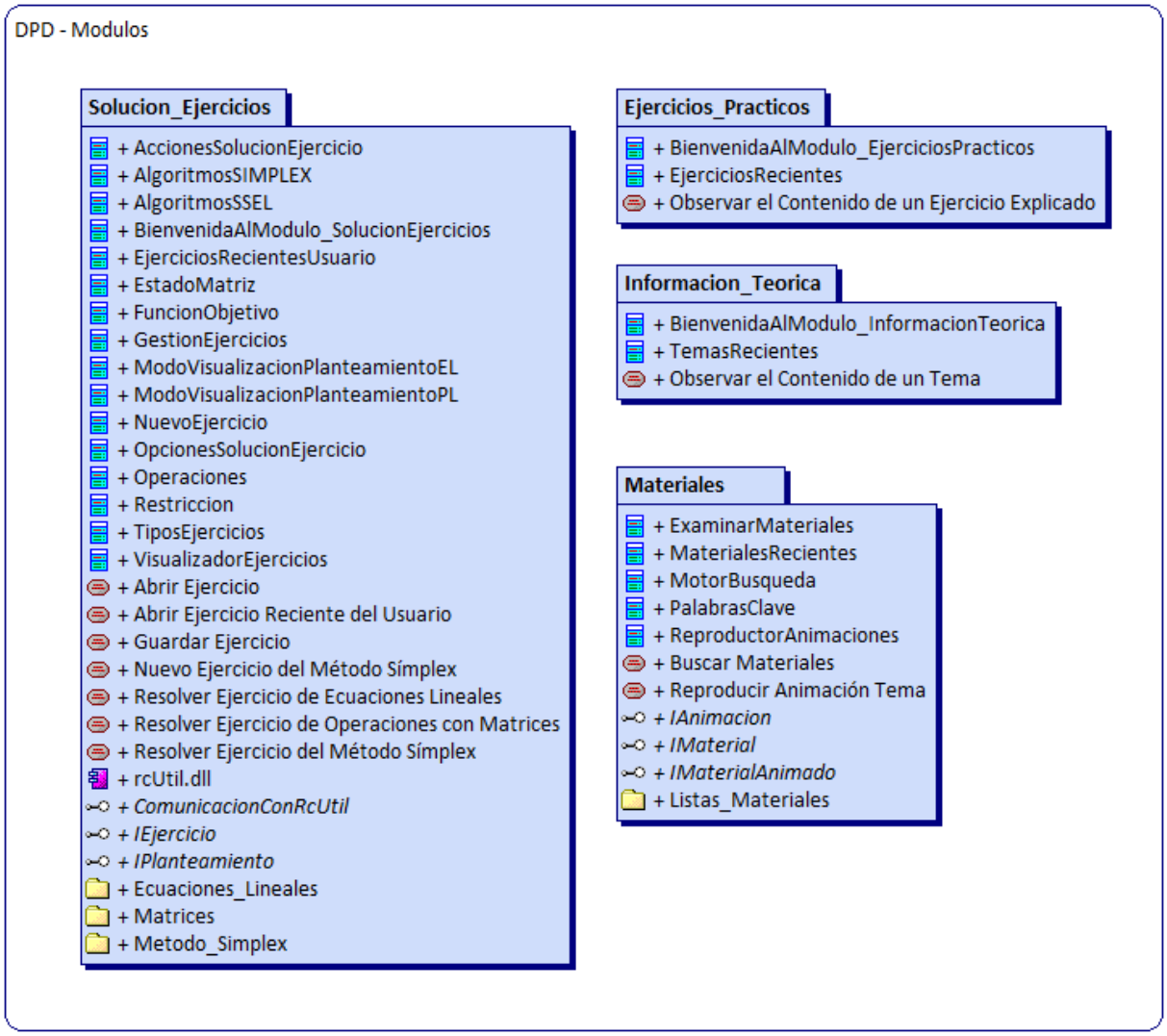
Por medio de los paquetes del diseño se organizan los artefactos que resultaron del modelo de diseño. La organización refleja la estructura que tendrá el código fuente de C# agrupado en *namespaces* (espacios de nombres) y, también, agrupa las colaboraciones del diseño.

6.4.1 Paquete: Sistema de diseño

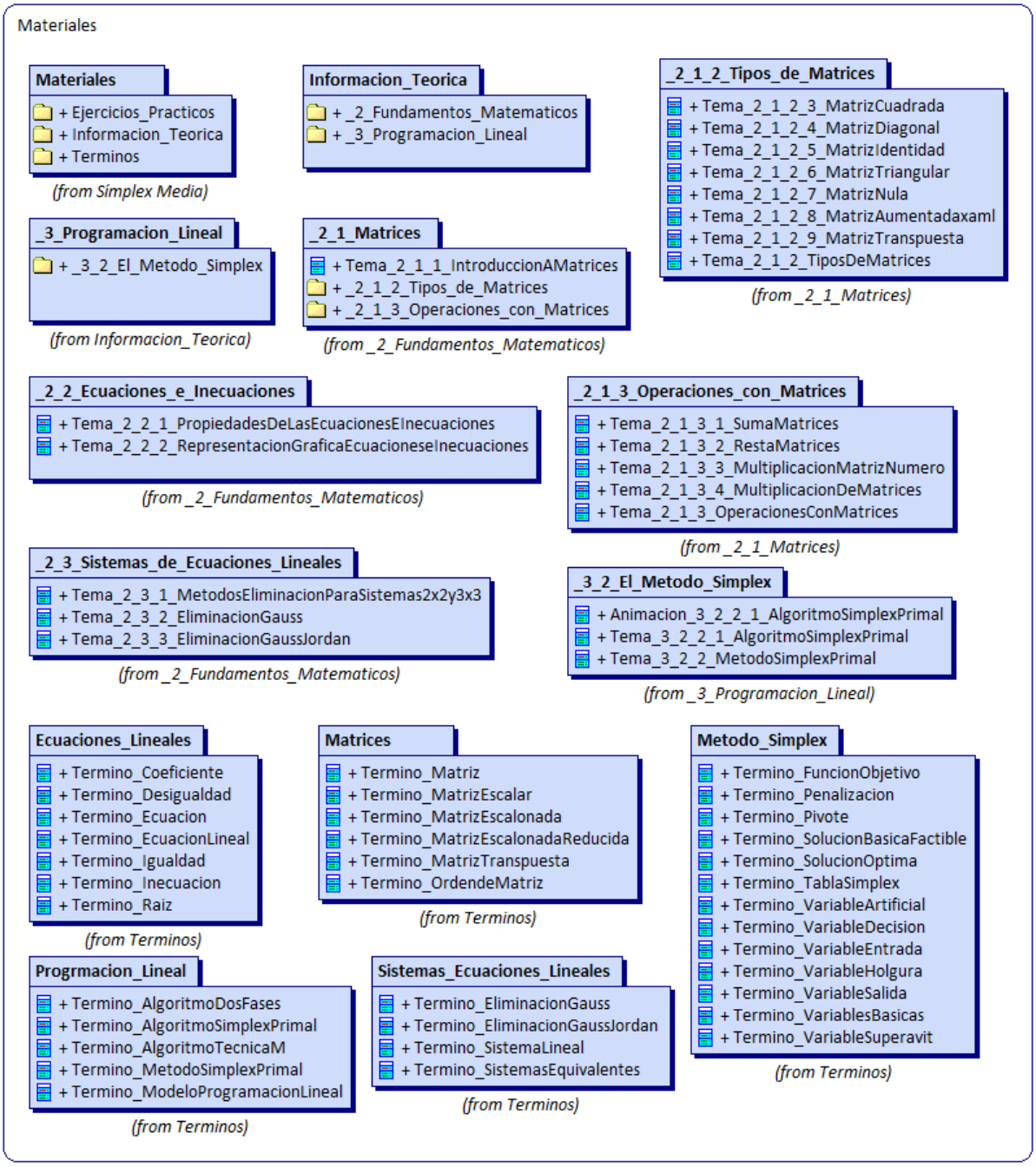
6.4.2 Paquete: SÍMPLEX MEDIA



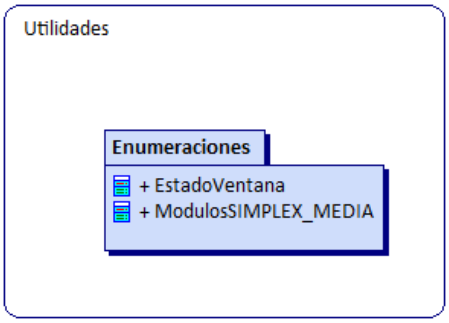
6.4.3 Paquete: Módulos



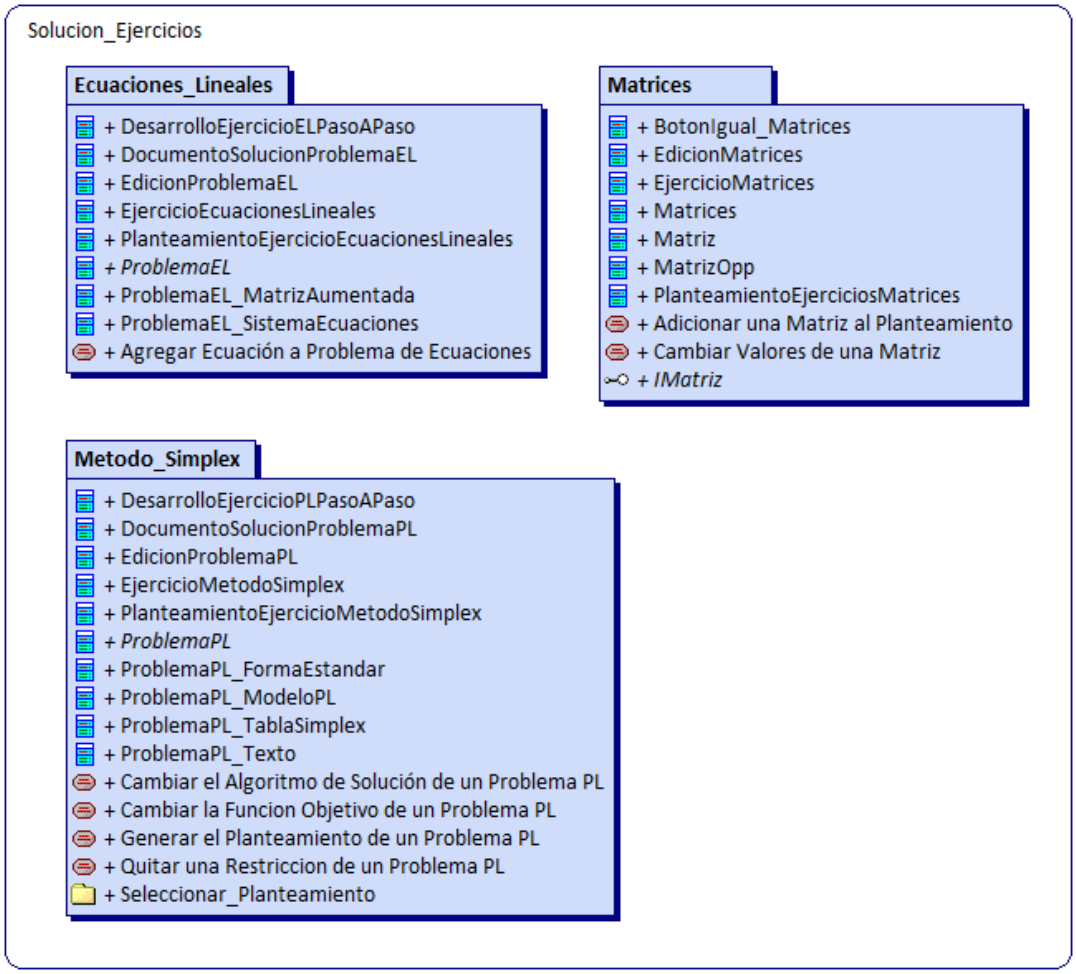
6.4.4 Paquete: Materiales



6.4.5 Paquete: Utilidades

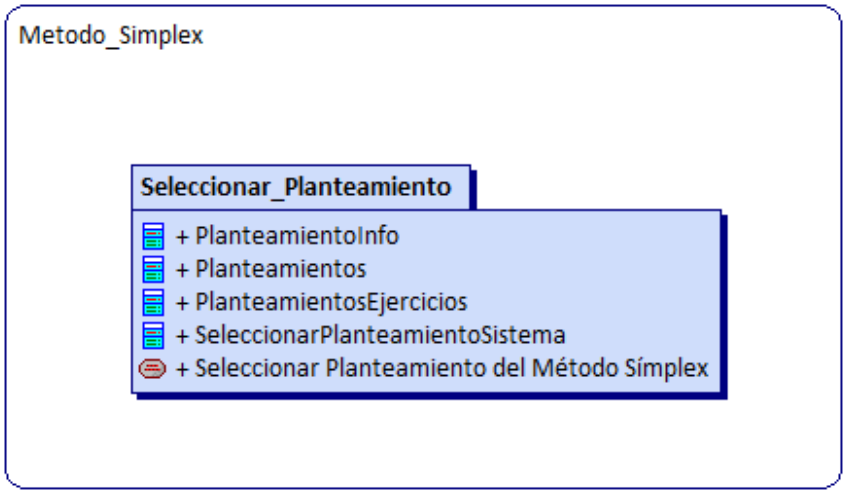


6.4.6 Paquete: Solución de ejercicios



6.4.7 Paquete: Modulos.Materiales

6.4.8 Paquete: Método Símplex



CONCLUSIONES

- La información que se explica en SÍMPLEX MEDIA, acerca del funcionamiento de los algoritmos del Método Símplex y temas relacionados como matices, sistemas de ecuaciones lineales, y programación lineal, ha sido enfocada desde perspectivas pedagógicas y didácticas, facilitando el proceso de estudio y enseñanza.

- SÍMPLEX MEDIA es una herramienta didáctica, realizada a partir del análisis de las debilidades y carencias que suelen poseer la mayoría de las personas que inician el estudio del Método Símplex.

- Entre las características de SÍMPLEX MEDIA, una de las que más sobresale es el manejo que se ha hecho de las herramientas de .Net, para brindar al usuario una interfaz amigable, que facilita la interrelación que hay entre el software y el estudiante.

- Las explicaciones han sido analizadas y estructuradas de tal manera que los algoritmos para la solución de problemas presenten pasos concretos que faciliten entenderlos y recordarlos. Además, han sido pensadas para abordar los temas de forma sencilla, pero al mismo tiempo profundizar con exactitud en las bases de los procedimientos, haciendo uso de un enfoque práctico. Es por esto que el módulo de *Información Teórica* se basa en ejemplos para realizar las explicaciones de la mayoría de los temas.

- El Software cuenta con un módulo dedicado a explicar problemas de la vida real, dando significado a los resultados obtenidos para facilitar la comprensión de los temas. Esto lo hace el módulo *Ejercicios Prácticos*.

- SÍMPLEX MEDIA incluye un módulo dedicado específicamente, a la *Solución de Ejercicios*, el cual brinda múltiples opciones que permiten al usuario resolver problemas que él mismo plantee, en temas relacionados con

matrices, ecuaciones lineales, o método Símplex dando una explicación de cada una de las iteraciones y los pasos que se dan tras cada iteración.

- SÍMPLEX MEDIA ha sido probado con una serie de problemas para determinar la eficacia del software en cuanto a la exactitud de los resultados que se obtienen tomando como referente reconocidos programas como Tora.

RECOMENDACIONES

Hacer uso de calculadora, lápiz y papel a la hora de estudiar, para reforzar los conceptos adquiridos mediante él y obtener buenos resultados con SÍMPLEX MEDIA.

Analizar la posibilidad de dar un enfoque diferente a la metodología que se usa para explicar materias como programación lineal, ya que actualmente se da prioridad a los cálculos matemáticos, cuando lo esencial debería ser desarrollar en los estudiantes habilidades en el planteamiento de problemas que puedan presentarse en el entorno y resolverlos mediante el algoritmo que le corresponda haciendo uso de herramientas como SÍMPLEX MEDIA.

Utilizar SÍMPLEX MEDIA en el estudio de la programación lineal, para agilizar el tiempo de desarrollo de los problemas, el mismo que puede ser utilizado en profundizar sobre Análisis de Sensibilidad, Dualidad y Paramétrico, por lo tanto se recomienda a los docentes y estudiantes pasar al análisis de los resultados proporcionados por el sistema.

Motivar a otros grupos de investigación para que puedan dar continuidad al proyecto incluyendo nuevos temas en la Información Teórica y otros algoritmos de solución a problemas de programación lineal. Temas como, Transportes, Transbordo, Asignación, Colas, Pilas y Programación Entera. Además, implementar los Análisis de Sensibilidad, Dualidad y Paramétrico.

Convertir a SÍMPLEX MEDIA en un Software Educativo.

BIBLIOGRAFÍA

BOOCH, GRADY; RUMBAUGH, JAMES; JACOBSON, IVAR. El Lenguaje Unificado de Modelado. Madrid, 1999, Addison Wesley, todo el libro.

BOOCH, GRADY; RUMBAUGH, JAMES; JACOBSON, IVAR. El Proceso Unificado de Desarrollo. Madrid, 2000, Addison Wesley, todo el libro.

FAULIN, JAVIER; ÁNGEL A. JUAN. Aplicaciones de la Programación Lineal. Disponible en Internet: http://www.unavarra.es/estadistica/LADE/Inv.Operativa/Apuntes%20IO/Temas%20InvOperativa-emath/Aplicaciones_PL.pdf

GALVIS PANQUEVA, ÁLVARO H; GÓMEZ CASTRO, RICARDO A; MARIÑO DREWS, OLGA. Ingeniería de Software Educativo con Modelaje Orientado a Objetos. Disponible en Internet: http://www.c5.cl/ieinvestiga/actas/ribie98/184.html

GONZÁLEZ REYESH, FABIO H. Tipos de software educativo. Disponible en Internet: http://www.mailxmail.com/curso/informatica/disenosoftware/capitulo9.htm

GROSSMAN STANLEY I. Aplicaciones de Álgebra Lineal. México, McGraw-Hill, 1992, Cuarta Edición, págs. 1 - 91.

LÓPEZ, H. AQI-Diet, Software para el Balanceo de Dietas en Especies Icticas. Pasto, 2006, todo el Libro.

MICROSOFT: http://msdn.microsoft.com/es-es/library/ms745659.aspx

MOSKOWITZ, WRIGHT G. Investigación de Operaciones. Prentice Hall Hispanoamericana S.A, 1991, págs. 51 - 355.

ORTEGA, EDUARDO. Visual Estudio 2008 y CF 3.5. Disponible en Internet: http://www.mobilenug.com/descargas/Descargas/Eventos/%5BTES%5DPresentacion_Visual_Studio_2008_Compact_Framework_3.5.pdf

SCHMULLER, JOSEPH. Aprendiendo UML en 24 horas. Prentice Hall Hispanoamericana S.A, todo el libro.

TAHA, H. Investigación de Operaciones. México, Alfa omega, 1995, 5ª edición, págs. 17 – 354.

ANEXOS