UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES ACATLÁN

ANÁLISIS Y DISEÑO ORIENTADO A OBJETOS PARA EL SISTEMA DE CONTROL ESCOLAR DE LA ESCUELA SUPERIOR DE INGENIERÍA AUTOMOTRIZ

TESIS

QUE PARA OBTENER EL TITULO DE

LICENCIADO EN MATEMÁTICAS APLICADAS Y COMPUTACIÓN

PRESENTA

BRAULIO ANTONIO FONSECA OROZCO

ASESOR: LIC. GABRIEL DÍAZ MIRÓN MAC DONOUGH

SEPTIEMBRE 2007

Agradezco todo el apoyo que siempre me han brindado mis padres, mi esposa, mi asesor y todos los profesores que me han compartido sus conocimientos.

Índice

Índice de figuras

INTRODUCCIÓN

El Análisis y Diseño Orientado a Objetos (A/DOO) esta centrado en la identificación de conceptos u objetos para comprender el problema, posteriormente la estructura del software se determina en base a estos objetos como elementos básicos de construcción del sistema .

El Proceso Unificado representa una metodología de trabajo, que basada en el A/DOO, identifica las actividades y sus requisitos relacionados en un área determinada. Dicha metodología, generalmente es aplicada a casos concretos y de mediana complejidad, misma que permite de una forma eficiente y relativamente rápida establecer la idea general del proyecto, así como identificar los elementos conflictivos que deberán ser resueltos en el desarrollo posterior del sistema.

El Proceso Unificado consta de cuatro fases para determinar actividades y requisitos. Tres de ellas se refieren al inicio, elaboración y construcción. En la fase de inicio se establecen los requerimientos del sistema, además se describe cómo opera actualmente el proceso de control escolar.

En la fase de elaboración se establece la base arquitectónica del sistema y se definen los requisitos del sistema. En esta fase quedan establecidos los límites del sistema, el flujo de información y la referencia básica de las operaciones.

En la fase de construcción se presentan las interfaces y reportes que se desean manejar como los entregables finales dando un panorama claro de cual sería la interacción entre el usuario y el sistema.

Cabe indicar que la aplicación de esta metodología, por lo que corresponde a las tres fases anteriores, únicamente busca identificar la problemática y establecer las actividades que deberán ser resueltas en la programación del sistema.

Como una forma de robustecer la metodología del Proceso Unificado, se hace uso de la herramienta CASE System Architect y la notación del Lenguaje Unificado de Modelado (Unified Modeling Language, UML).

De lo anterior, en este documento se señala la aplicación de la metodología de Proceso Unificado en la Escuela Superior de Ingeniería Automotriz, particularmente en la identificación de sus actividades y requerimientos de control escolar basados en las primeras tres etapas de dicho proceso.

Objetivo general

El objetivo general del presente documento es utilizar el Análisis y Diseño Orientado a Objetos como fundamento para señalar la aplicación del Proceso Unificado en el control escolar de la Escuela Superior de Ingeniería Automotriz.

Objetivos específicos

De forma complementaria al objetivo general, los objetivos específicos son:

- Señalar el marco teórico que corresponde al enfoque del Proceso Unificado.
- Aplicar el Proceso Unificado en el requerimiento específico de la Escuela Superior de Ingeniería Automotriz, en relación al control escolar.

Problemática

La Escuela Superior de Ingeniería Automotriz (ESIA) fue fundada en 1996. Desde sus inicios ha crecido de manera constante. A lo largo del tiempo ha utilizado registros manuales y programas comerciales para administrar la información sobre alumnos y profesores en relación a grupos, materias y calificaciones.

En la actualidad, la ESIA tiene en promedio 650 estudiantes distribuidos en 8 semestres. Conforme se ha incrementado la matrícula, los directivos estiman que también se han venido multiplicando las limitaciones de los registros actuales, ocasionando errores en la toma de decisiones.

La problemática que se ha diagnosticado incluye aspectos de seguridad de la información, redundancia de registros, inconsistencia de información, disponibilidad para acceder a la información en pantalla o en reportes adecuados, y en los tiempos de respuesta para realizar concentrados e informes finales.

De lo anterior, cabe preguntarse ¿En qué consiste la metodología del Proceso Unificado?, ¿Cuáles son las actividades y requerimientos del control escolar del ESIA? y ¿de que manera se aplica el proceso unificado en la solución de la problemática de la ESIA?

Justificación

Actualmente, la Escuela Superior de Ingeniería Automotriz carece de un sistema computacional que realice de forma eficiente la tarea de control escolar.

Hasta la fecha la institución ha utilizado procesos manuales, susceptibles de falla, contratiempo y pérdida de información relevante para su toma de decisiones académicas y administrativas.

En función del requerimiento expresado por la Escuela, relativo al control escolar, se busca contar con una solución que administre la información de alumnos, profesores, grupos, materias, calificaciones e historial académico, misma información que ha venido aumentando de forma consistente a lo largo del tiempo.

De lo anterior, el tema de Proceso Unificado se justifica en la medida de ofrecer una solución adecuada a problemáticas concretas, relativamente complejas y de una escala mediana.

Igualmente, la aplicación del Proceso Unificado se justifica en el requerimiento de una solución de control escolar para la Escuela Superior de Ingeniería Automotriz, en la medida en que por sus actividades y por sus requerimientos específicos es necesario identificar por medio del análisis y el diseño la idea general del proyecto antes de proceder a la programación del sistema.

CAPÍTULO I

PROCESO UNIFICADO

1.1 El análisis

El análisis y diseño son la estructura fundamental de los sistemas de información. Ello comprende la etapa de investigación del problema, el establecimiento de los requisitos del sistema y la propuesta de solución a los planteamientos que se han investigado.

Durante el análisis se busca información relacionada con el problema en los diferentes departamentos o áreas de trabajo, ya sean controles de asistencia, archivos históricos, folletos o material no almacenado. Se debe comprender cómo trabaja el sistema actual y, de manera más específica, cuál es el flujo de información en todo el sistema.

Por otra parte, los analistas necesitan saber los motivos que tiene el negocio para cambiar su modo de operación. Por ejemplo, ¿tiene la empresa problemas con la búsqueda de datos, con la entrega de resultados finales o con el dinero? ¿Se rezaga el registro de documentos? ¿Se necesita un sistema más eficiente como requisito previo para poder aumentar el número de operaciones?

Sólo después de haber reunido todos los hechos, el analista se encuentra en posición de determinar cómo y dónde un sistema está orientado hacia el futuro, ya que todavía no existe ningún sistema como tal. El analista valora, de manera cuidadosa, las necesidades futuras del negocio y los cambios que deben considerarse para satisfacer esas necesidades. En este caso, como en muchos otros, los analistas recomiendan opciones para mejorar la situación, siendo lo usual tener varias estrategias posibles.

Al trabajar con los gerentes y empleados de la organización, los analistas de sistemas pueden recomendar que opciones adoptar de acuerdo con la forma en que se adecua la solución al negocio y su ambiente en particular así como al soporte que, por parte de los empleados, tenga la solución propuesta.

Algunas veces el tiempo necesario para desarrollar una opción, comparado con el de otras, es el aspecto más crítico. Los costos y beneficios también son factores determinantes. Al final, es la administración la que paga, hace uso de los resultados y es la que decide qué opción aceptar.

Una vez tomada la decisión, se diseña el plan para implementar la recomendación. Los diseños para el negocio proporcionan las diferentes maneras para capturar datos relacionados con el control de usuarios, además de especificar la forma en que estos datos serán almacenados.

El personal de control también necesitará información de retroalimentación para conocer el estado actual. Cada diseño describe las diferentes salidas generadas por el sistema, tales como reportes, análisis estadísticos, listas de movimientos y evaluaciones. Los analistas de sistemas deciden qué salidas utilizar y cómo generarlas.

No obstante toda la tecnología con que se cuenta hoy en día, son las personas las piezas más importantes para que una organización trabaje. En los procesos descritos anteriormente se menciona la participación de diferentes personas. Los gerentes y empleados son generalmente quienes tienen las mejores referencias acerca de lo que si funciona y qué es lo que no, que causa problemas, dónde son necesarios los cambios y dónde no y, especialmente en qué partes será aceptado el cambio y en cuáles no.

Respecto a la determinación de los requerimientos, este es el estudio de un sistema para conocer cómo trabaja y dónde es necesario efectuar mejoras. Los estudios de sistemas dan como resultado una evaluación de la forma en que trabajan los métodos actualmente empleados y si es necesario o posible realizar ajustes.

Un requerimiento es una característica que debe incluirse en un nuevo sistema. Esta puede ser la inclusión de determinada forma para capturar o procesar datos, producir información, controlar una actividad de la empresa o brindar soporte a la gerencia. Es así como la determinación de requerimientos vincula el estudio de un sistema existente con recopilación de detalles relacionado con él.

El primer paso del analista es comprender la situación. Ciertos tipos de requerimientos son tan fundamentales que son comunes en casi todas las situaciones. Dar respuestas a un grupo específico de preguntas será de gran ayuda para comprender los requerimientos básicos.

También existe otra clase de requerimientos relacionados con el sistema orientado a transacciones, toma de decisiones o su extensión por varios departamentos. Por ejemplo, la necesidad de informar a los directivos de un incremento repentino en el número de usuarios que está por llegar subraya la importancia de eslabonar los departamentos de control, finanzas y dirección.

Los analistas utilizan métodos específicos, denominados técnicas para encontrar hechos, con el objeto de reunir datos relacionados con los requerimientos. Entre éstos se incluyen la entrevista, el cuestionario, la revisión de

los registros (en el sitio donde se encuentran éstos) y la observación. En general los analistas emplean más de una de estas técnicas para estar seguros de llevar a cabo una investigación amplia y exacta.

Los requerimientos de diseño se formulan a partir de los resultados del análisis. Los requerimientos de un nuevo sistema son aquellas características o detalles que deben incorporarse para producir las mejoras o cambios que el analista determinó como necesarios. En otras palabras los requerimientos son las actividades o mejoras que debe proporcionar el nuevo sistema y se obtienen al comparar el rendimiento actual con los objetivos de desempeño aceptables de un sistema.

Los requerimientos comunes de los sistemas incluyen mejoras en la operatividad, tales como el aumento del volumen de trabajo o un tiempo menor para la recuperación de información.

Existen también beneficios económicos obtenidos, al disminuir ya sea los costos de procesamiento o el número de errores. También son requerimientos comunes de sistemas la integración de datos o de varias áreas de las organizaciones.

Actualmente está aumentando el número de organizaciones que desarrollan sistemas de información con la finalidad de obtener ventajas competitivas en el mercado.

Particularmente, el Análisis y Diseño Orientado a Objetos (A/DOO) se aplica una metodología para describir los objetos o conceptos que intervienen en el problema, y posteriormente especificar los objetos del software para establecer cómo interactúan entre si para satisfacer los requerimientos.

El Análisis Orientado a Objetos es el proceso de clasificación e interpretación de hechos, diagnóstico de problemas y empleo de la información para recomendar mejoras al sistema. Antes de que se pueda diseñar un sistema ya sea para capturar datos, actualizar archivos o emitir reportes, primero se debe averiguar acerca de cómo opera el negocio, con qué documentación cuenta (requisiciones, listas, notas, etcétera.) para guardar la información manualmente y qué documentos, si es que los hay, se producen y cómo se emplean.

El análisis hecho con el estudio de los datos sugiere, en general, varios caminos que conducen hacia el cambio o mejora deseada. Por consiguiente, el analista debe tratar de identificar aquéllos que son más factibles desde el punto de vista técnico, económico y operacional. Las opciones identificadas de esta manera son las estrategias para satisfacer los requerimientos. Ellas forman la base para el diseño de sistemas.

1.2 El diseño

Durante el análisis se determina qué debe hacer el sistema, mientras que en el diseño se debe establecer cómo alcanzar el objetivo.

El diseño de sistemas tiene dos etapas: el diseño lógico y la construcción física del sistema. Cuando el analista formula el diseño lógico, escribe las especificaciones detalladas del nuevo sistema. Es decir, aquellas que describen sus características: salidas, entradas, archivos o bases de datos y los procedimientos, todo en una forma que satisfaga los requerimientos del proyecto. El conjunto formado por todas estas características recibe el nombre de especificaciones de diseño del sistema.

El diseño lógico de un sistema de información es similar al proyecto de ingeniería de un automóvil: muestra las características más sobresalientes (como el motor, la transmisión y el espacio para los pasajeros) y la relación que guardan entre sí (dónde se conectan los componentes unos con otros o cuál es la separación que existe entre las puertas).

Los reportes y salidas generadas por el analista son similares a los componentes de diseño de un ingeniero. Los procedimientos y datos se enlazan entre sí para producir un sistema que trabaja.

El diseño lógico también especifica los formatos de entrada y la distribución de la salida en pantalla para dar mantenimiento a los datos del registro escolar, como son el ingreso al sistema de los datos generales y calificaciones de alumnos.

Las especificaciones de procedimientos describen los métodos utilizados para ingresar datos en el sistema, copiar archivos y detectar problemas, si éstos se presentaran.

La construcción física, que es la siguiente actividad después del diseño lógico. Esta produce el software, los archivos y un sistema que funciona. Las especificaciones de diseño indican a los programadores lo que el sistema debe hacer. A su vez, los programadores escriben programas que aceptan la entrada proporcionada por los usuarios, procesan los datos, producen los reportes y guardan los datos en los archivos.

El diseño físico para el sistema está formado por instrucciones de programa, escritas en algún lenguaje de programación. Durante la construcción física, los programadores escriben las instrucciones necesarias del programa para calcular porcentajes o evaluaciones definitivas.

Un objetivo fundamental en el diseño de un sistema de información es asegurar que éste brinde apoyo a la actividad de la empresa para el que fue desarrollado, es decir, la tecnología de cómputo y comunicaciones especificadas en el diseño tienen un papel secundario en relación con los resultados que se pretende que el sistema proporcione, lo más importante es el factor humano.

El diseño se ajusta en función de la forma de trabajo de la empresa, esto quiere decir que se toman en cuenta las políticas, y procesos del negocio para adaptarlos a la funcionalidad del sistema.

Durante el diseño se incluye las especificaciones del software. Estas especificaciones establecen las funciones de entrada, salida y procesamiento así como los algoritmos necesarios para efectuarlas. Los módulos de software junto con las rutinas, se enfocan sobre lo que cada función realiza; asimismo, se especifican los procedimientos necesarios para llevar a cabo dichas funciones.

La selección de lenguajes de programación, paquetes de software y utilerías se efectúa durante el proceso de diseño lógico y las recomendaciones se incluyen como parte de las especificaciones del software.

En el diseño se describen las características del sistema, los componentes o elementos del sistema y la forma en que estos aparecerán ante los usuarios. Los componentes de un sistema de información descritos durante el análisis de requerimientos, son el punto principal del diseño de sistemas conocidos como elementos del diseño. También se deben diseñar las salidas, los archivos, interacciones con la base de datos, entradas, controles, procedimientos y especificaciones para programas.

A continuación se da la descripción de estas características.

Elementos del diseño

- Flujos de datos.
- Movimientos de datos hacia, alrededor y desde el sistema.
- Almacenes de datos.
- Conjuntos temporales o permanentes de datos.
- Procesos.
- Actividades para aceptar, manejar y suministrar datos e información, ya sean manuales o basadas en computadora, por ejemplo, llenar un formato, imprimirlo y entregarlo a otra área para que continúe un determinado proceso.
- Métodos y rutinas para utilizar el sistema de información y lograr con ello los resultados esperados.
- Controles.
- Estándares y lineamientos para determinar si las actividades están ocurriendo en la forma anticipada o aceptada, es decir si se encuentran bajo control.
- Procesos del negocio en general.

Igualmente, se especifican las acciones que tienen que emprenderse cuando ocurren problemas o se presentan circunstancias inesperadas. Puede incluirse un reporte sobre las excepciones o procedimientos para la corrección de los problemas.

Por otra parte, en el diseño de salida, el término salida se refiere a los resultados e información generados por el sistema. Para muchos usuarios finales, la salida es la única razón para el desarrollo del sistema y la base sobre la que ellos evaluarán la utilidad de la aplicación.

En la realidad, los usuarios interactúan con el sistema de diferentes maneras, algunos ingresan datos en él, otros validan la información, otros la modifican y otros utilizan la salida generada por el sistema.

Al momento de diseñar la salida se toma en cuenta lo siguiente:

- Determinar que información presentar.
- Decidir si la información será presentada en forma visual, verbal o impresa y seleccionar el medio de salida.
- Disponer la presentación de la información en un formato entendible para quien la vaya a utilizar.
- Decidir como distribuir la salida entre los posibles destinatarios.

En el diseño de interacciones con la base de datos, dada la importancia que tienen las bases de datos en la mayoría de los sistemas, su diseño es establecido y vigilado por un administrador de base de datos, que es una persona (o grupo de personas) que tienen la responsabilidad de desarrollar y mantener la base de datos.

En estos casos, el analista de sistemas no efectúa el diseño de la base de datos sino que consulta al administrador de la base para determinar las interacciones más apropiadas con la base de datos. El analista proporciona al administrador la descripción de 1) los datos que son necesarios de la base de datos, y 2) las acciones que tendrán efecto sobre la propia base.

A su vez el administrador tiene las siguientes responsabilidades:

- Describir los métodos para interactuar con la base de datos.
- Asegura que la aplicación no pueda dañar la base de datos o que la afecte de manera adversa a las necesidades de otros sistemas de información.

En cuanto al diseño de la entrada, para esta característica el analista considera los siguientes detalles:

- Que datos ingresan al sistema.
- Que medios utilizar.
- La forma de presentar o codificar los datos.
- El dialogo que servirá de guía a los usuarios para dar entrada a los datos.

- Validación necesaria de datos y transacciones para detectar errores.
- Métodos para llevar a cabo la validación de las entradas y los pasos a seguir cuando se presentan errores.
- Diseño de controles.

También se anticipan los errores que se cometerán al ingresar los datos en el sistema o al solicitar la ejecución de ciertas funciones. Algunos errores no tienen consecuencias para la consistencia del sistema como, pero otros pueden ser tan serios que ocasionarán el borrado de datos o el uso inapropiado del sistema.

Aunque exista sólo la más mínima probabilidad de cometer un error serio, un buen diseño de sistema de información ofrecerá los medios para detectar y manejar el error.

En el diseño de los procedimientos, los procedimientos especifican qué tareas deben efectuarse al utilizar el sistema y quiénes son los responsables de llevarlas a cabo. Los procedimientos más importantes son:

- Procedimientos para entrada de datos.
- Procedimientos durante la ejecución.
- Procedimientos para el manejo de errores.
- Procedimientos de seguridad y respaldo.

En el diseño de especificaciones para programas, las especificaciones para programas son por sí mismas un diseño. Ellas describen cómo transformar las especificaciones de diseño del sistema – salidas, entradas, procesamientos y otras- en software de computadora.

Respecto al Desarrollo Orientado a Objetos, actualmente se están poniendo en marcha grandes proyectos utilizando el desarrollo orientado a objetos, además de cientos de casos exitosos que ya se encuentran en producción.

Esta es una motivación extra para proseguir con el desarrollo orientado a objetos y parece no haber objeciones para que sea utilizado en la mayoría de los proyectos de software. Cabe mencionar que la reducción de tiempo no es el principal beneficio del desarrollo orientado a objetos en comparación con el desarrollo convencional.

El tiempo utilizado hasta que se alcanza el primer código completo es probablemente el mismo o quizá mayor. Sin embargo, las siguientes iteraciones del desarrollo orientado a objetos son más rápidas y fáciles que empleando el desarrollo convencional.

Además suelen ser requeridas menos iteraciones debido a que se descubren y se corrigen más problemas durante el desarrollo, a parte del alto grado de reutilización que tiene este tipo de software.

1.3 El Lenguaje Unificado de Modelado

El Lenguaje Unificado de Modelado (Unified Modeling Language, UML) es un lenguaje estándar para escribir planos de software. En su totalidad UML esta enfocado principalmente para ser utilizado en sistemas grandes o complejos, ya que permite visualizar, especificar, construir y documentar los artefactos de un sistema.

En la década de los ochenta cuando surgían los lenguajes de programación orientados a objetos y la necesidad de aplicaciones cada vez más complejas, empezaron desarrollarse diferentes metodologías para el análisis y diseño.

Los métodos que más destacaron fueron el de Booch, el método OOSE (Object Oriented Software Engineering, Ingeniería del Software Orientada a Objetos) de Jacobson y el método OMT (Object Modeling Technique, Técnica de Modelado de Objetos) de Rumbaugh. Cada uno estaba completo, pero tenían sus puntos fuertes y sus puntos débiles.

Poco a poco fueron evolucionando estas tres metodologías adoptando aspectos entre ellas, hasta que decidieron fusionarse y crear un lenguaje unificado de modelado, de esta manera podían colaborar para cubrir problemas que no se podían manejar anteriormente además, se le dio cierta estabilidad al mercado orientado a objetos ofreciendo un lenguaje de modelado que facilitaba la estandarización de los proyectos.

El desarrollo del nuevo lenguaje estaba enfocado como una solución robusta. En primer lugar se tuvo que acotar el problema tal como si se tratara de un lenguaje de programación, durante esta etapa se hicieron planteamientos para ver si debía incluir la especificación de requisitos o si llegaría a tener la potencia como para permitir la programación visual.

En segundo lugar se debía tener un equilibrio en el alcance de problemas a resolver, ya que un lenguaje simple ofrecería una escasa gama de alternativas y por otro lado si se hacia muy complejo, entonces se sobrecargaría a las personas que lo utilizaran.

Finalmente se debían tener en cuenta los antecedentes de métodos, de manera en que no se excedieran en cambios para no confundir a los usuarios, pero que tampoco frenaran la evolución del lenguaje para que llegara a más usuarios y se volviera más sencillo.

El esfuerzo de UML que comenzó en 1994 dio su primer resultado para octubre de 1995 con la versión de borrador 0.8. Para 1997 y debido también al apoyo de diversas organizaciones interesadas en el proyecto como fueron

Rational, IBM, Hewlett-Packard, Microsoft, Texas Instruments, se liberó la versión 1.0, ofrecida para su estandarización al OMG (Object Management Group).

Actualmente el OMG es el organismo encargado del control de mantenimiento de UML y hasta la fecha la versión más reciente publicada es UML 1.4 a la cual se apega este documento.

Con lo anterior, un UML es un lenguaje estándar para escribir los planos del software. Se aplica para modelar sistemas de información en empresas hasta aplicaciones distribuidas para la Web.

UML es independiente de cualquier proceso que se utilice. Sin embargo, para sacarle mejor provecho se recomienda un proceso que sea:

- Dirigido por los casos de uso.
- Centrado en la arquitectura.
- Iterativo e incremental.

Dirigido por los casos de uso quiere decir que los casos de uso se utilizan como un artefacto básico para establecer el comportamiento deseado del sistema, para verificar y validar la arquitectura del sistema, para las pruebas y para la comunicación entre las personas involucradas en el proyecto.

Centrado en la arquitectura significa que la arquitectura del sistema se utiliza como un artefacto básico para conceptuar, construir, gestionar y hacer evolucionar el sistema en desarrollo.

Un proceso iterativo es aquél que involucra la gestión de un flujo de ejecutables del sistema. Un proceso incremental es aquél que involucra la continua integración de la arquitectura del sistema para producir esos ejecutables.

El Proceso Unificado consiste de una serie de etapas para la construcción de sistemas orientados a objetos. El principal enfoque es el desarrollo iterativo en el cual se establecen una serie de mini-proyectos cortos y de duración fija llamados iteraciones; el resultado de cada uno es un sistema que puede ser probado, integrado y ejecutado. Cada iteración incluye sus propias actividades de análisis de requisitos, diseño, implementación y pruebas.

Durante el ciclo de vida iterativo la base principal es la ampliación y refinamiento sucesivos del sistema mediante múltiples iteraciones, con retroalimentación cíclica y adaptación como elementos principales que dirigen para converger hacia un sistema adecuado. El sistema crece incrementalmente a lo largo del tiempo, iteración tras iteración, y debido a esto también se le conoce como desarrollo iterativo e incremental.

Entre las ventajas del desarrollo iterativo destacan:

- Mitigación tan pronto como sea posible de riegos altos (técnicos, requisitos, objetivos, usabilidad y demás).
- Progreso visible en las primeras etapas.
- Una temprana retroalimentación, compromiso de los usuarios y adaptación que nos lleva a un sistema refinado que se ajusta más a las necesidades reales del personal involucrado.
- Gestión de la complejidad; el equipo no se ve abrumado por la “parálisis del análisis” o pasos muy largos y complejos.
- El conocimiento adquirido en una iteración se puede utilizar metódicamente para mejorar el propio proceso de desarrollo, iteración a iteración.

1.4 Herramientas CASE System Architect

En el ámbito de sistemas de información la frase de Ingeniería de Software Asistida por Computadora (CASE, Computer Aided Software Engineer) es muy utilizada, ya que implica el uso de herramientas para llevar a cabo alguna o varias tareas durante el ciclo de desarrollo de sistemas.

Algunas tareas de las más socorridas para utilizar estas herramientas es la ingeniería de código hacia delante o en reversa, por ejemplo, a partir de un diagrama de clases generar el esqueleto de código de las clases para empezar a programar u obtener el diagrama de entidad-relación a partir de una base de datos.

Las herramientas CASE incluyen principalmente los siguientes 5 componentes: repositorio de información, herramientas de diagramación, generador de interfaces, generador de código y herramientas de administración.

Repositorio de información: Es un depósito centralizado de información con el objetivo de compartir, controlar y proteger todos los datos que son introducidos al CASE para preservar la exactitud y consistencia de los detalles del sistema, de esta manera las diferentes herramientas utilizan las definiciones capturadas previamente, y a su vez complementan el repositorio con más información que puede ser utilizada en etapas posteriores del desarrollo.

Herramientas de diagramación: Las herramientas de diagramación van de la mano con las metodologías soportadas por el CASE, por lo tanto muchas herramientas se especializan en ciertas teorías, por ejemplo las estructuradas o las orientadas a objetos, sin embargo, la tendencia es que las herramientas CASE abarquen toda una gama de metodologías de diferentes autores, esto permite que los analistas adapten estas facilidades ofrecidas de acuerdo a las necesidades de los diferentes y muy variados proyectos de ingeniería de sistemas.

Generador de interfaces: Esta funcionalidad básicamente nos ayuda a crear pantallas prototipo para las interfaces con los usuarios. Generalmente ofrecen la posibilidad de crear menús de demostración para el sistema, pantallas de presentación y el formato de los informes, y la ventaja es que estos se generan a partir del diccionario de datos o de una entidad definida en algún diagrama.

Generadores de código: El generador de código que ofrece un CASE permite automatizar la preparación del software, lo que permite convertir las definiciones del sistema en código ejecutable.

Aun no se ha perfeccionado completamente la generación de código. En el mejor de los casos se puede conseguir hasta un 75% del código fuente de una aplicación. Los programadores deben escribir el resto.

En esta sección cabe mencionar la ingeniería inversa. Esta facilidad es el complemento del generador de código ya que a partir del software se pueden generar los diagramas que muestran las descripciones y relaciones de los objetos.

En cuanto a las herramientas de administración, este componente ayuda a los analistas a llevar un control del proyecto. Algunos sistemas CASE permiten calendarizar las actividades de análisis y diseño así como la asignación de recursos a las diferentes actividades del proyecto. También ofrecen la posibilidad de generar informes utilizando los detalles contenidos en el repositorio y presentándolos en diversos formatos y niveles, ya sea con el objetivo de repartir información a los programadores o publicar un sitio con información general o muy detallada acerca del proyecto que este disponible para todos.

Algunas herramientas CASE también permiten que se personalicen elementos como símbolos, pantallas de captura e incluso permiten definir metodologías de desarrollo propias, incluyendo las reglas de validación y los estándares para datos y nombres de procedimientos acorde al funcionamiento de las empresas.

En la valoración de los CASE, ciertamente las herramientas CASE ofrecen un gran apoyo para facilitar y hacer más eficiente el desarrollo de sistemas, pero también es cierto que actualmente muchas herramientas CASE proveen aún insuficiente valor agregado, por lo que una herramienta bien diseñada, en conjunto con usuarios, tiene un gran nicho para ser explotado. Existe una gran oportunidad aquí, la sugerencia es que se trabaje en problemas reales, junto con los usuarios ya sean programadores o analistas que proporcionen opiniones independientes sobre la utilidad de lo desarrollado.

A continuación se presentan algunos beneficios y debilidades que se deben considerar al evaluar una herramienta CASE.

En cuanto a los beneficios se tiene:

- Facilidad para el mantenimiento de aplicaciones: Una vez implantado el sistema requiere revisiones y mejoras, esto es facilitado por el CASE ya que en lugar de hacer ajustes al código fuente, contribuye a mejorar el sistema por medio de cambios en las especificaciones.
- Agilidad para desarrollar prototipos de sistemas: El verdadero beneficio del desarrollo de prototipos es la creación rápida de pantallas que permitan saber si es lo que realmente quiere el usuario y la facilidad para cambiarlas sin la necesidad de tener que programar ni una línea de código.
- Generación de código: La generación de código ahorra tiempo de programación además ofrece una base estándar para el desarrollo de código.
- Mejor habilidad para satisfacer los requerimientos del usuario: Los diferentes entregables del CASE como son; diagramas, prototipos de pantallas, reportes, diccionarios de datos entre otros, facilitan la interacción con el usuario ofreciendo diferentes perspectivas para el mismo sistema, aumentado la probabilidad de éxito del proyecto.
- Soporte iterativo para el proceso de desarrollo: Generalmente los sistemas se desarrollan de manera iterativa. Las herramientas CASE apoyan este proceso al permitir la creación de diagramas y otros artefactos de manera automática a partir de la información contenida en el repositorio. Esto da como resultado la revisión de detalles del sistema con mayor frecuencia y en forma consistente.

En cuanto a las desventajas se tiene:

- Soporte estándar de metodologías: Muchas herramientas soportan los diagramas que emplea una metodología pero no imponen sus reglas y procesos. Por otro lado las herramientas que proporcionan un soporte limitado a una sola metodología pueden forzar e l uso riguroso de reglas, procedimientos y estándares de esta; además brindan ayuda sensible al contexto y bases de conocimiento que ofrecen asistencia experta, sin embargo, entre más metodologías soporte una herramienta, existe la posibilidad cada vez mayor que la seguridad y ayuda que esta ofrece sea menor.
- Diagramas no utilizados: Uno de los argumentos de los vendedores de herramientas es que las presentaciones graficas y la documentación mejoran la comunicación entre los desarrolladores y la productividad del desarrollo. Sin embargo algunos profesionales de los sistemas, emplean las herramientas para automatizar la producción de informes y documentación del sistema una vez que se ha terminado del sistema en lugar de utilizarlas en el desarrollo de software.

- Dependencia del factor humano: Las herramientas CASE soportan las funciones de modelado, verificación, manejo de datos y utilerías necesarias para el desarrollo, sin embargo, las herramientas deben estar en manos de expertos y adaptarse a la arquitectura de la información y las metodologías de desarrollo utilizadas por la organización. Una de las tareas críticas es donde las personas interactúan entre sí: determinación y verificación de requerimientos con el usuario. Y a medida que mejoren las funciones de modelado y búsqueda de errores, la responsabilidad del éxito en un sistema de información caerá cada vez más sobre aquellos que especifican los requerimientos de información.

Por otra parte, el System Architect soporta ampliamente UML, además trabaja también con la mayoría de las metodologías estructuradas y permite crear o adaptar una metodología propia de acuerdo a las necesidades de la empresa. Cuenta con un repositorio de información, de esta manera todas las descripciones que se capturen en el repositorio se podrán utilizar con alguna otra función de la herramienta. La captura de información es a través de pantallas de propiedades que también pueden ser adaptadas para mayor especificación del diccionario de datos.

El modo de trabajo de System Architect se basa en una arquitectura que denomina el “Framework de Zachman” que permite a los usuarios crear una matriz que muestra cómo diferentes modelos soportan diferentes elementos dentro de una infraestructura, en realidad lo que hace esta arquitectura es organizar una vista de modelos almacenados en el repositorio de System Architect.

Entre sus funciones principales se tienen:

- Diagramación: Una de las principales utilidades de la herramienta es la ayuda que proporciona a la hora de dibujar ya que con gran facilidad se pueden armar diagramas y describir las propiedades de los elementos tan detallado como sea necesario. Posteriormente, la información almacenada por los diagramas se puede explotar de diferentes maneras; puede ser publicada como reportes, examinada, consultada por interfaces o analizada por matrices de cruces de información.
- Metodologías: La versatilidad de la herramienta es uno de sus puntos fuertes. Actualmente hay una gran diversidad de teorías adoptadas por las empresas y es fácil encontrarse tanto con una metodología estructurada como orientada a objetos o bien quienes optan por metodologías híbridas en las que se explotan los beneficios de diversas corrientes de Análisis y Diseño.

El System Architect soporta las siguientes metodologías ampliamente:

- Modelado de Datos
  - Entidad Relación
  - Liga SystemArchitect/Power Builder

- Modelado del Negocio
  - Empresarial
  - Funcional (IDEF0)
  - Flujo de Procesos (IDEF3)

- Análisis y Diseño Estructurado
  - Gane/Sarson
  - Ward/Mellor
  - Yourdon/DeMarco

- Modelado de Objetos
  - UML
  - OMT
  - Booch94

- Ingeniería hacia delante e inversa: El concepto de ingeniería hacia delante se aplica al proceso de crear una base da datos física a partir de un modelo de datos lógico, de manera inversa, al proceso de generar un diagrama entidad-relación es decir obtener el modelo de datos lógico a partir de una base datos física se le conoce como ingeniería inversa o de reverso.
  System Architect saca provecho de esta tecnología. La ingeniería hacia delante inicia con un diagrama entidad-relación y continúa creando un diagrama físico, el cual se apega a la notación y tipos de datos del manejador, al cual se va a exportar el modelo que incluye a las firmas de software de bases de datos más populares.
  La ingeniería inversa puede realizarse vía ODBC, archivo DDL o utilizando la liga directa con SQL Server.

- Generación de código (aplicaciones y código): System Architect permite generar esqueleto de código en C++, Delphi, Java, Visual Basic y HTML pero la funcionalidad de las operaciones debe ser completada por los programadores. También permite crear aplicaciones del tipo Maestro-Detalle en Visual Basic, Power Builder y Magic.

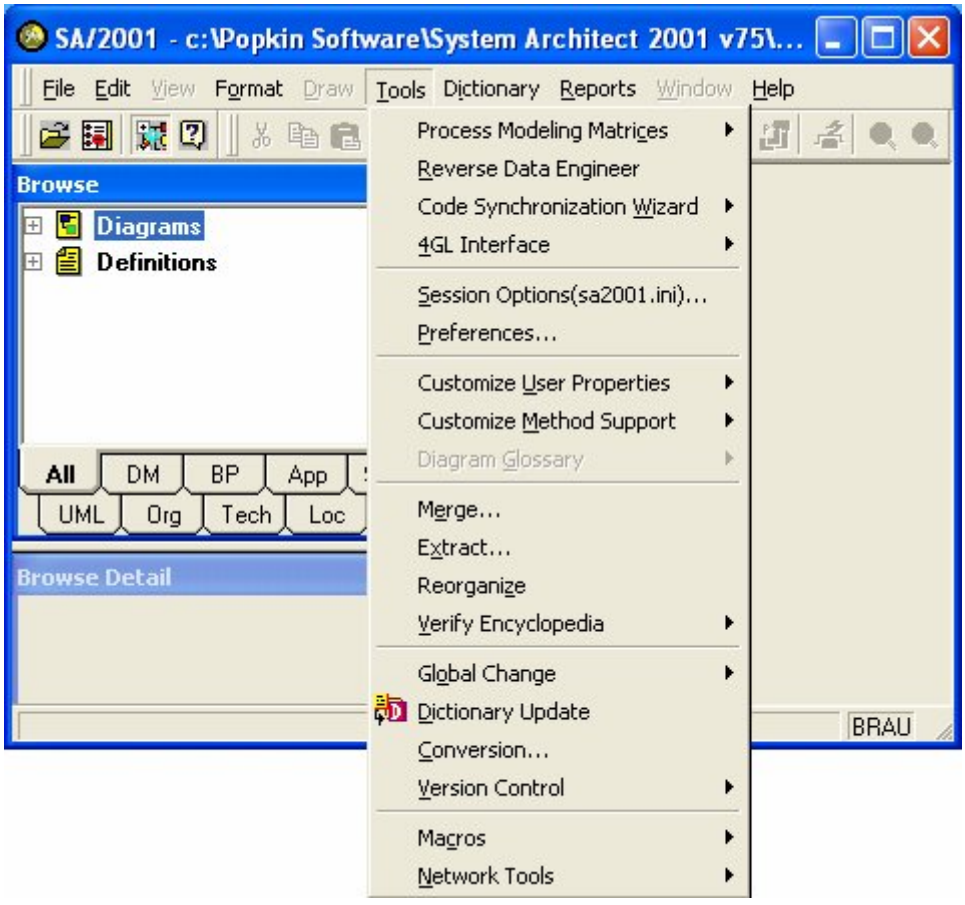
- Pantallas prototipo: La generación de pantallas prototipo se logra a partir de los objetos definidos en un diagrama de UML o de Entidad-Relación o bien en el diccionario de datos con solo arrastrar el objeto

dentro de la ventana, posteriormente se puede guardar la pantalla como un formulario de Visual Basic, Delphi o Power Builder

- Generación de reportes (Word, HTML y Texto): System Architect contiene un catalogo predefinido de reportes ya sean por diagramas o definiciones. Se puede especificar los elementos que deben aparecer, hasta obtener el nivel de detalle deseado. El reporte se genera con un índice y con las imágenes de los diagramas.

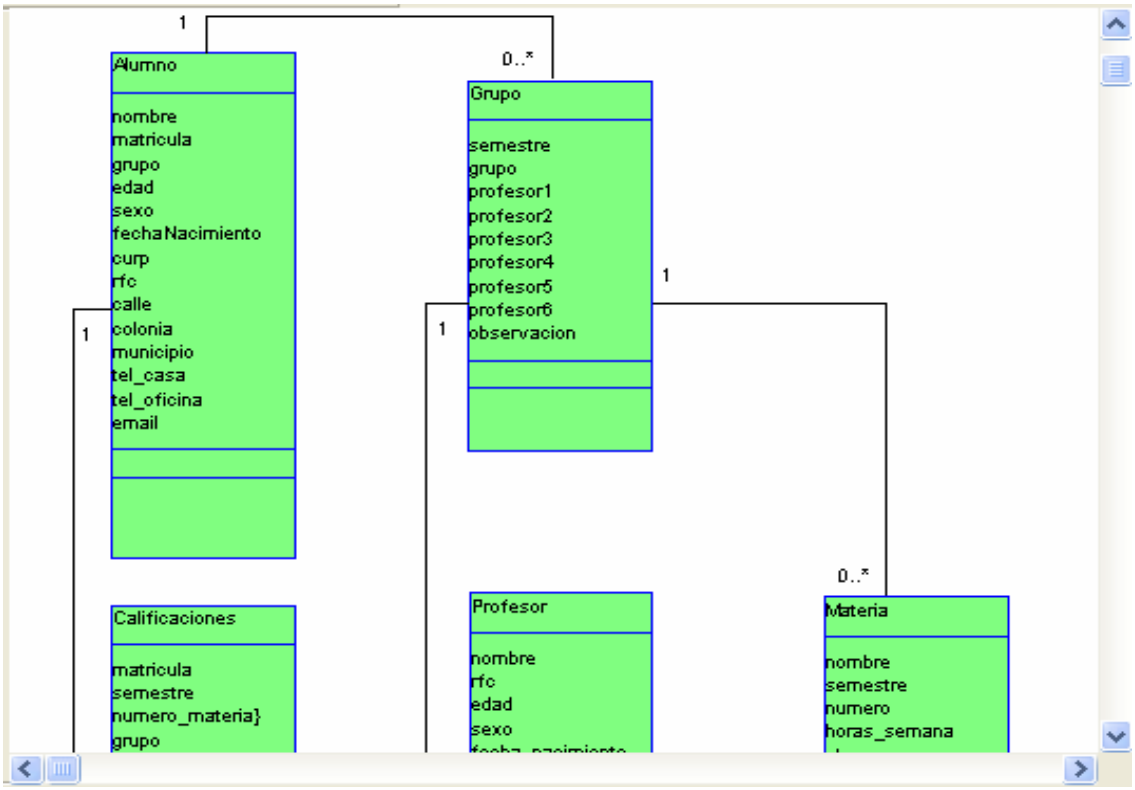
- Personalización de elementos: La manera de personalizar las propiedades de los elementos del diccionario de datos o de los diagramas es mediante un archivo de texto USRPROPS, el cual es leído por System Architect cada vez que inicia para aplicar los cambios dentro de la herramienta. Por este medio se pueden crear plantillas para diagramas o definiciones tomando como base las ya existentes para poder formar nuevas notaciones o pantallas de captura de información.

Figura 1.1
Escritorio de trabajo de System Architect



La facilidad para dibujar los diagramas y la capacidad para generar código a partir de estos mismos aporta una gran ayuda al proyecto.

Figura 1.2
Diagrama de clases elaborado con System Architect



1.5 Proceso Unificado

El Proceso Unificado describe actividades de trabajo, como escribir casos de uso, en disciplinas. Una disciplina se puede describir como un conjunto de actividades (y artefactos) relacionados en un área determinada, como las actividades en el análisis de requisitos.

En el Proceso Unificado, un artefacto es el término general para cualquier producto del trabajo: código, gráficos Web, esquema de base de datos, documentos de texto, diagramas, modelos, etcétera.

Hay varias disciplinas en el Proceso Unificado; los artefactos más utilizados son de las siguientes tres:

- Modelado del Negocio. Cuando se desarrolla una única aplicación, esto incluye el modelado de los objetos del dominio. Cuando se está haciendo análisis del negocio a gran escala o reingeniería del proceso del negocio, esto incluye el modelado dinámico de los procesos del negocio de toda la empresa.
- Requisitos. Análisis de los requisitos para una aplicación, como escritura de casos de uso e identificación de requisitos no funcionales.
- Diseño. Todos los aspectos de diseño, incluyendo la arquitectura global, objetos, bases de datos, red y cosas parecidas.

Algunas prácticas y principios del Proceso Unificado deben seguirse siempre, como el desarrollo iterativo y dirigido por el riesgo, y la verificación continua de la calidad.

Es importante saber que en el Proceso Unificado todas las actividades y artefactos (modelos, diagramas, documentos, etcétera) son opcionales. El conjunto de artefactos posibles del Proceso Unificado debería entenderse como una serie de opciones a seguir, donde hay disponibles varias, pero uno elige únicamente los que se consideran útiles en determinado momento.

Figura 1.3
Artefactos de Proceso Unificado y su evolución

Disciplina	Artefactos Iteración =>	Inicio 1	Elab. 1…n	Const. 1…n	Trans. 1…n
Modelado del Negocio	Modelo del dominio		c		
Requisitos	Modelo de casos de uso Visión Glosario	c c c	r r r		
Diseño	Modelo de diseño Arquitectura Modelo de datos		c c c	r r r	
Implementación	Modelo de implementación		c	r	
Gestión del proyecto	Plan de desarrollo de software	c	r	r	r
Pruebas	Modelo de pruebas		c	r	
Entorno	Marco de desarrollo	c	r		

Elab. = Elaboración.
Const. = Construcción.
Trans. = Transición.

c = crear.
r = refinar.

Como en el caso de ESIA, donde se aplica el Proceso Unificado, debe haber un equipo encargado de seleccionar el pequeño subconjunto de artefactos que servirán para tratar los problemas y necesidades particulares del proyecto. En general, éste debe centrarse en un pequeño conjunto de artefactos que demuestren tener un gran valor práctico.

En el marco de desarrollo, los artefactos seleccionados del Proceso Unificado para un proyecto se especifican en un breve documento llamando Marco de Desarrollo (un artefacto de la disciplina Entorno Figura 1.3).

Cuando se habla del desarrollo secuencial o también conocido como desarrollo en cascada se refiere a un proceso que abarca cuatro etapas: Análisis, Diseño, Implementación y Pruebas, en este proceso se puede pasar a una etapa siguiente solo hasta haber concluido la etapa actual.

En una situación ideal, este proceso funciona de manera óptima; sin embargo, es común que durante el desarrollo de un sistema se presenten situaciones como cambios de objetivos, redefinir los requisitos que no son claros, errores cometidos durante el desarrollo o algún contratiempo que no fue contemplado al inicio del proyecto.

Si se observa el ciclo de vida se puede decir que una etapa concluida implica pasar a la siguiente, sin embargo el mayor riesgo se presenta al final cuando surgen problemas que no se pudieron detectar antes, por ejemplo, un sistema que haya sido diseñado para que los usuarios (ejecutivos de ventas) ingresen los datos de una cotización para un cierto cliente y esta cotización sea almacenada en la base de datos central. Al poner en marcha el sistema resulta que el ejecutivo tiene que realizar una cotización en las instalaciones del cliente y no puede conectarse a la base de datos central, por lo tanto el sistema tiene que ser rediseñado para que sea capaz de guardar la cotización y cuando el ejecutivo tenga la oportunidad de conectarse a la red, entonces se transfiera la información de la cotización a la base de datos central.

Para hacer mas claro este ejemplo supóngase que se tienen las cuatro etapas del proceso de desarrollo secuencial Análisis, Diseño, Implementación y Pruebas, si se asigna un valor arbitrario fijo a cada etapa que represente el costo, se puede obtener un estimado del costo total del proyecto. Ahora si para un retraso en cada etapa se asigna un costo de penalización fijo, se puede hacer el cálculo para determinar el costo en caso de tener que trabajar nuevamente en una etapa, tomando en cuenta que para hacer modificaciones en una etapa se tiene que regresar a la etapa anterior para que todo esto quede fundamentado y documentado.

Figura 1.4
Costo normal de cada etapa durante el proyecto

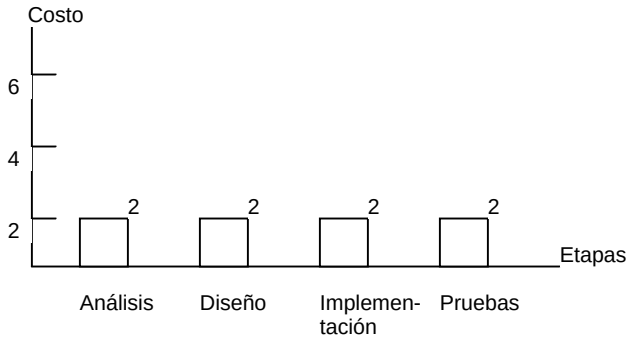
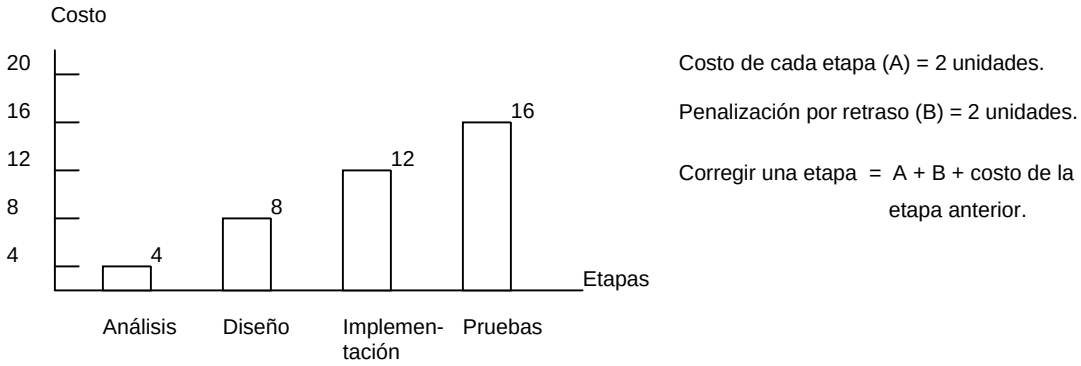


Figura 1.5
Costo relativo aproximado por detectar y corregir fallas durante cada etapa



En la realidad, la presencia de cualquier tipo de contratiempos durante el desarrollo de los sistemas es prácticamente una cuestión inherente. Como consecuencia, el ciclo de vida de los sistemas se asemeja más al proceso de la figura del desarrollo iterativo, figura 1.7.

En el desarrollo iterativo se hace un recorrido de todo el ciclo en cada fase, de esta manera se puede poner mayor atención en los problemas más críticos.

A continuación se muestran los gráficos sobre la evolución del sistema en el tiempo:

Figura 1.6
Ciclo de vida secuencial o en cascada

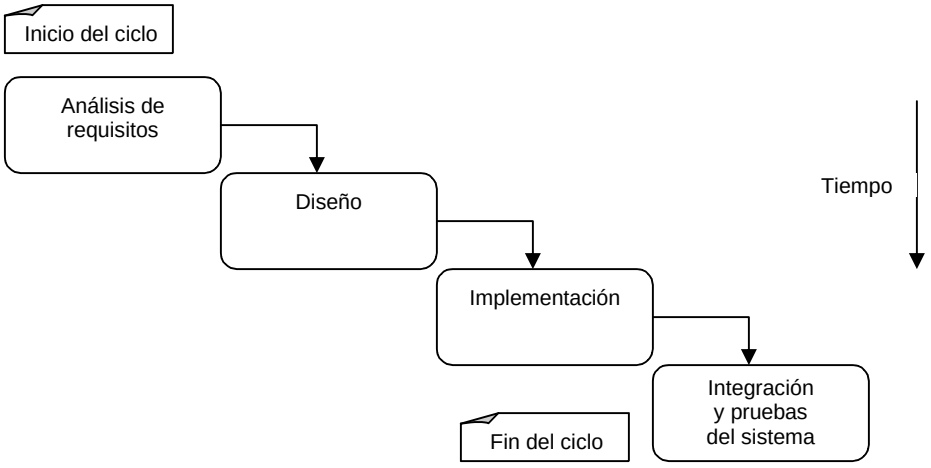
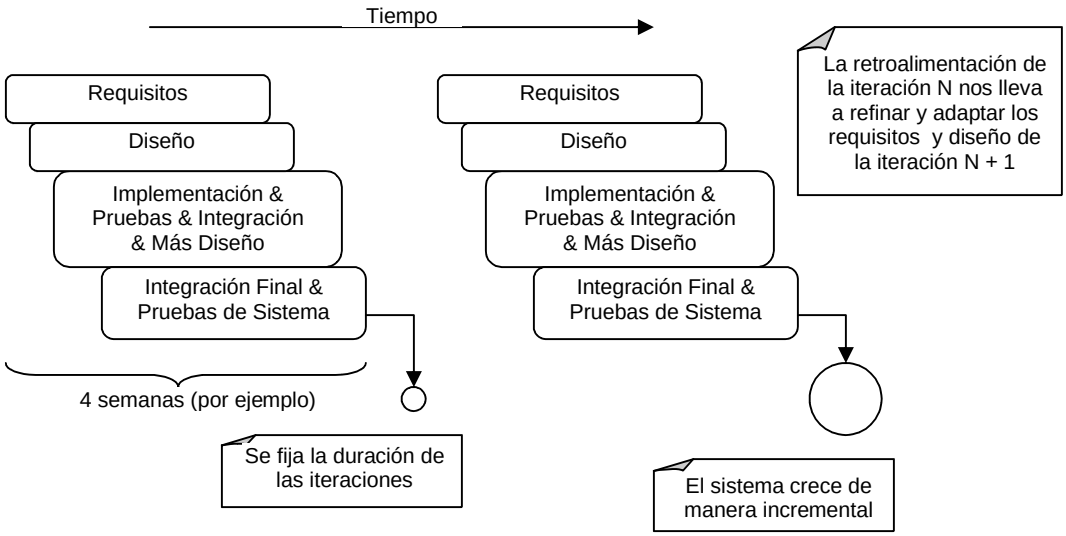


Figura 1.7
Desarrollo iterativo e incremental



De lo anterior, el Proceso Unificado organiza el trabajo y las iteraciones en cuatro fases fundamentales:

1. Inicio: visión aproximada, análisis del negocio, alcance, estimaciones imprecisas.
2. Elaboración: visión refinada, implementación iterativa del núcleo central de la arquitectura, resolución de los riesgos altos, identificación de más requisitos y alcance, estimaciones más realistas.
3. Construcción: implementación iterativa del resto de requisitos de menor riesgo y elementos más fáciles, preparación para el despliegue.
4. Transición: pruebas beta, despliegue.

CAPÍTULO II

APLICACIÓN DEL PROCESO UNIFICADO: FASE DE INICIO

2.1 Problemática de la Escuela Superior de Ingeniería Automotriz

El Centro Educativo Grupo Cedva nació en Julio de 1973, en el municipio de Tlalnepantla, Estado de México. Inicialmente, el grupo contaba con una sola escuela dedicada a impartir cursos de Mecánica Automotriz a nivel técnico.

A través de los años, ya para 1996, tenía más de 30 escuelas técnicas dentro del territorio nacional. En este año, se funda la Escuela Superior de Ingeniería Automotriz (ESIA), que fue la primera escuela que impartió una carrera con nivel de licenciatura dentro del grupo con el nombre de Ingeniería Mecánica Automotriz.

La carrera de Ingeniería Mecánica Automotriz se cursa en 8 semestres Hasta el ciclo escolar 2006 la ESIA tenía una matrícula de 650 estudiantes.

El proceso de control escolar que se lleva a cabo en la ESIA implica las siguientes actividades:

Actividad	Descripción
Inscripción de alumnos	El alumno se presenta en la ventanilla de control escolar con su credencial y copia de la ficha de pago con su matrícula e indica a la secretaria el grupo en el que se va a inscribir. La secretaria guarda la copia en un fólder correspondiente al grupo.
Creación de grupos académicos	La secretaria valida la ficha de pago y que el alumno tenga aprobado el semestre anterior. Si es correcto agrega al alumno en el archivo de Excel del grupo que le corresponde.
Registro de calificaciones	El profesor se presenta en la ventanilla de control escolar e indica a la secretaria el grupo y materia y le entrega en una hoja la lista de calificaciones de cada alumno. La secretaria captura las calificaciones en el archivo de Excel correspondiente
Imprimir boletas e historiales académicos	La secretaria abre el archivo de Excel del grupo que imprimirá, por medio de una formula indica el numero de lista del alumno para desplegarlo en el formato requerido e imprimir la hoja.
Entrega de boletas	El alumno se presenta en la ventanilla de control escolar

e historiales académicos	y solicita su boleta o historial de calificaciones. Si es el final del semestre, la secretaria ya tiene impresas las boletas e historiales, entonces busca en la carpeta que corresponde al grupo del alumno y se la entrega. Si es durante el semestre, la secretaria apunta la matrícula y numero de lista del alumno y al siguiente día el alumno pasa a recoger el documento.

Estas actividades se realizan durante el transcurso del semestre por 2 secretarias que registran la información de los alumnos en hojas de Excel.

De acuerdo a la manera en que se realizan estas actividades, a continuación se muestra la problemática de este modo de operar:

- Seguridad. Se cuenta con una mínima seguridad de la información ya que los archivos de Excel están expuestos a ser editados y modificados por cualquier persona que tenga acceso a ellos.
- Redundancia. Un registro de un alumno se repite en cada hoja de Excel donde se haga referencia para presentar sus datos, lo cual requiere mayor espacio de almacenamiento.
- Inconsistencia. la información puede variar cuando se hace una copia de los registros de una hoja a otra, cuando se edita cada vez que se modifican calificaciones o datos personales, o como efecto de la redundancia, debido a que no hay un repositorio central de información que garantice que el origen de los datos esté actualizado.
- Disponibilidad. Actualmente solo se puede visualizar información en los formatos de boletas, historiales y concentrados de calificaciones por grupos. No es posible hacer un análisis de datos utilizando cruces de información, o estadísticas, por ejemplo: porcentajes de alumnos reprobados por materias, por semestres o por algún parámetro específico.
- Tiempos de respuesta. Para cumplir con las actividades es necesario disponer de dos personas dedicadas de tiempo completo a la administración de datos y archivos de Excel y en las etapas de inicio y fin de semestre es insuficiente.

De lo anterior, la ESIA ha decidido renovar su aplicación de control escolar de manera que pueda aprovechar las ventajas tecnológicas que se ofrecen. Al diseñar un sistema de control escolar, las especificaciones del mismo incluyen definiciones de reportes y pantallas de presentación que describen las listas de asistencias, actas de calificaciones, boletas de calificaciones e historiales académicos.

Basado en el proceso unificado, se procede a completar tres fases: la de inicio, la elaboración y la construcción. De cada una de ellas se procederá a:

- Inicio: visión aproximada y el análisis de requerimientos o requisitos.

- Elaboración: visión refinada, implementación iterativa del núcleo central de la arquitectura, resolución de los riesgos altos, identificación de más requisitos y alcance, estimaciones más realistas.
- Construcción: implementación iterativa del resto de requisitos de menor riesgo y elementos más fáciles, preparación para el despliegue.

2.2 Visión aproximada

La fase de inicio tiene como objetivo detectar los riesgos más significativos del proyecto y definir el marco de desarrollo del proyecto. En ella es conveniente establecer la visión aproximada, el análisis de requisitos, el alcance y las estimaciones imprecisas.

De la visión aproximada, en el caso de ESIA, el sistema de control escolar debe cumplir con las expectativas tanto del personal administrativo, así como de los alumnos.

El sistema deberá tener la suficiente flexibilidad para adaptarse a nuevas reglas del negocio y contar con la capacidad para interactuar e integrase con sistemas de terceras partes, como pueden ser bases de datos o dispositivos periféricos.

El manejo de información, utilizando hojas de cálculo en Excel, o inclusive documentos archivados en carpetas, da lugar a problemas en la entrega de resultados oportunos y libres de errores. Además, el volumen y flujo de datos que se manejan actualmente requieren cada vez de una mayor inversión de tiempo y esfuerzo.

Por otra parte, el aislamiento de la información representa una pérdida de conocimiento acerca del funcionamiento de la institución imprescindible para la identificación de puntos débiles o bien las ventajas competitivas.

Objetivos de alto nivel y problemas claves del personal involucrado

Objetivo de alto nivel	Prioridad	Problemas e inquietudes	Soluciones actuales
Elaboración de boletas de alumnos, concentrados por grupo e historias académicas de	Alta	Se deben reunir las actas de calificaciones y tener cuidado de seleccionar las líneas correspondientes a cada	Dos personas se encargan de recopilar la información.

alumnos.		alumno en las diferentes materias.	
Capturar las calificaciones y faltas de los alumnos en sus respectivos grupos, materias y periodos parciales.	Media	El proceso llevado a cabo es lento y se presta a errores de captura.	Se captura la información en hojas de Excel y se revisan al final.
Concentrar los datos de alumnos y profesores (altas bajas y cambios).	Media	Para preparar y un documento que requiera los datos del profesor o alumno se debe abrir el archivo del grupo al que esta asignado y copiarlo	Se debe capturar nuevamente el nombre completo a partir de la lista de grupo.

Especificación de beneficios

Característica soportada	Beneficio del personal involucrado
Se ofrecerá la opción de imprimir el registro de calificaciones en diferentes formatos (boletas, concentrados, historias académicas).	Este modulo agiliza la entrega de boletas, actas, historias académicas y concentrados.
Base de datos con los registros de los alumnos, calificaciones, grupos y profesores.	Los datos de los alumnos solo se capturan una vez, posteriormente se buscara el registro y se recuperaran los datos cuando se requiera.
Facilidad para corrección de datos.	En cualquier momento se puede localizar el registro de un alumno o profesor y modificar los datos correspondientes.
Rapidez en la captura de la información.	Disponibilidad de tiempo para realizar otras actividades.
Interfaz de trabajo amigable con el usuario.	Facilidad para entender el sistema.
Adaptabilidad a nuevos procesos o medios de control.	En algún momento se pueden adaptar periféricos de entrada instantánea como los scanner de láser.
Documentación completa de las operaciones y características del sistema.	Se cuenta con la información necesaria de los procesos para un mantenimiento eficiente.

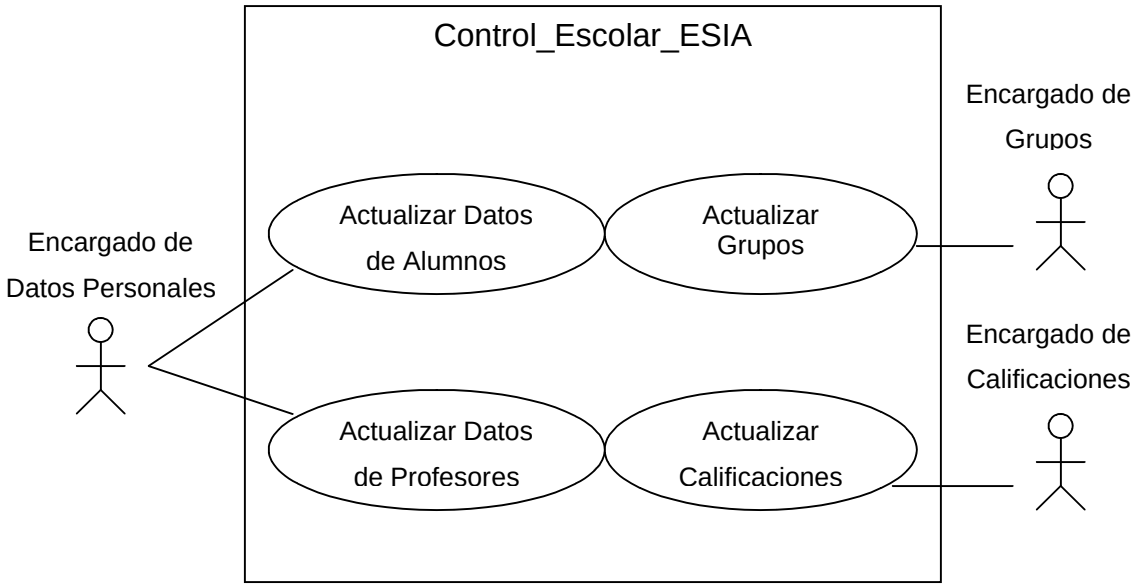
2.3 Análisis de requerimientos

Para describir los requerimientos del sistema se utilizan los casos de uso. Los casos de uso describen un “escenario” de cómo se utiliza el sistema y se expresan como una serie de pasos e interacciones entre un “actor” y el sistema, a continuación se definen los elementos:

- Escenario.- es una secuencia específica de acciones, representa un momento en particular del sistema. Para un caso de uso se tiene un escenario principal y uno o varios escenarios alternativos.
- Actor.- es algo con comportamiento, generalmente representa una persona pero también puede ser otro sistema o una organización. Un actor inicia o dispara los procesos del sistema.

La descripción de los casos de uso se hace generalmente mediante texto, pero también es importante tener una visión general del planteamiento del problema. El diagrama de casos de uso muestra de manera gráfica cómo se conceptualiza el problema en general.

Figura 2.1
Diagrama contextual de casos de uso del ESIA



La descripción básica de casos de uso del ESIA comprende las altas y bajas y cambio de datos del alumno así como del profesor, asignación del alumno a un grupo, quitar alumnos, asignar profesores, quitar profesores, generar listas de asistencia, capturar calificaciones, generar concentrados e historias académica.

Enseguida se muestra la acción del actor y la respuesta del sistema para cada caso de uso (CUn):

CU1 Alta de alumno

Acción del Actor	Respuesta del Sistema
1. Introduce los datos del alumno	
2. Guarda la información	3. Almacena el registro en la base de datos
	4. Presenta una nueva pantalla

CU2 Baja de alumno

Acción del Actor	Respuesta del Sistema
1. Introduce la matrícula del alumno	2. Presenta los datos del alumno
3. Borra el registro	4. Solicita confirmación
5. Confirma la solicitud	6. Elimina el registro de la base datos

CU3 Cambio de datos

Acción del Actor	Respuesta del Sistema
1. Introduce la matrícula del alumno	2. Presenta los datos del alumno
3. Modifica los datos deseados	
4. Guarda la información	5. Actualiza el registro en la base de datos

Los casos de uso 1 al 3 muestran el segmento de Altas, Bajas y Cambios de un alumno. Los casos de uso 4 al 6 corresponden a los escenarios de Altas, Bajas y Cambios de un profesor, en estos casos el comportamiento del actor y del sistema son idénticos a los casos del alumno, por consiguiente serán omitidos en la presentación de esta investigación, la única diferencia que se debe mencionar es que el parámetro de búsqueda de un profesor es su rfc en lugar de la matrícula.

CU7 Asignar alumno a un grupo

Acción del Actor	Respuesta del Sistema
1. Introduce el año, periodo, semestre y grupo	2. Busca en la base de datos los alumnos que en el campo grupo coincida con el introducido por el usuario
	3. Muestra la lista de alumnos que ya están asignados
4. Introduce la matrícula del nuevo alumno	
5. Busca el registro del alumno	6. Encuentra y presenta los datos del alumno
	7. Habilita la opción de asignar al grupo
	8. Por default marca todas las materias en las que el alumno será inscrito

9. Asigna el alumno al grupo	10. Actualiza el registro del alumno en la base de datos con el grupo al que se esta asignando
	11. Actualiza el registro del alumno en la base de datos con el valor de “true” para las materias que le corresponden

Escenario alterno del CU7

Selecciona solo las materias en las que será inscrito el alumno	Solo actualizará el registro del alumno con valor de “true” para las materias que se hayan seleccionado

CU8 Quitar alumnos

Acción del Actor	Respuesta del Sistema
1. Introduce el año, periodo, semestre y grupo	2. Busca en la base de datos los alumnos que en el campo grupo coincida con el introducido por el usuario
	3. Muestra la lista de alumnos que ya están asignados
4. Selecciona el registro del alumno que desea quitar del grupo	
5. Borra el registro	6. Solicita confirmación
7. Confirma la solicitud	8. Actualiza el registro del alumno en la base de datos borrando el grupo al que estaba asignando
	9. Actualiza el registro del alumno en la base de datos con el valor de “false” para las materias que le corresponden

Los casos de uso 7 y 8 corresponden a quitar y asignar alumnos cuyo comportamiento es representativo de los casos de uso 9 y 10 correspondientes a quitar y asignar profesores por lo cual no es necesario mostrarlos en esta investigación.

CU11 Generar lista de asistencia

Acción del Actor	Respuesta del Sistema
1. Introduce el año, periodo, semestre y grupo	2. Presenta las materias correspondientes al semestre
3. Selecciona la materia	4. Busca en la base de datos los alumnos que en el campo grupo coincida con el introducido por el usuario y que tengan el valor de “true” en el campo de la materia seleccionada
	5. Busca en la base de datos el profesor correspondiente a la materia

	seleccionada
	6. Muestra una vista previa de la lista
7. Imprime la lista	

CU12 Capturar calificaciones

Acción del Actor	Respuesta del Sistema
1. Introduce el año, periodo, semestre y grupo	2. Presenta las materias correspondientes al semestre
3. Selecciona la materia	4. Busca en la base de datos los alumnos que el grupo coincida con el introducido por el usuario y que tengan el valor de “true” para la materia seleccionada. También busca el registro de calificaciones del semestre y materia indicados para cada alumno
	5. Muestra la lista de alumnos con las calificaciones que ya tenga registradas
6. Selecciona el alumno y captura sus calificaciones	
7. Guarda la información	9. Actualiza el registro de calificaciones con la matrícula del alumno, semestre, materia y las correspondientes calificaciones y faltas

CU13 Generar boletas de calificaciones

Acción del Actor	Respuesta del Sistema
1. Introduce la matrícula del alumno	2. Busca en la base de datos el alumno y busca los registros de calificaciones que correspondan a su matrícula y a su grupo
	3. Muestra una vista previa de la boleta
4. Imprime la boleta	

CU14 Generar actas de calificaciones

Acción del Actor	Respuesta del Sistema
1. Introduce el año, periodo, semestre y grupo	2. Presenta las materias correspondientes al semestre
3. Selecciona la materia	4. Busca en la base de datos los alumnos donde el grupo coincida con el introducido por el usuario y que tengan el valor de “true” en la materia seleccionada. También busca el registro de calificaciones del semestre y materia indicados para cada alumno
	5. Muestra una vista previa del acta
6. Imprime el acta	

CU15 Generar concentrados

Acción del Actor	Respuesta del Sistema
1. Introduce el año, periodo, semestre y grupo	2. Presenta las materias correspondientes al grupo y los respectivos profesores
	3. Busca en la base de datos los alumnos que en el campo grupo coincida con el introducido por el usuario. También busca el registro de calificaciones definitivas de cada alumno para el grupo indicado
	4. Calcula el promedio de calificaciones y porcentaje de alumnos reprobados.
	5. Muestra una vista previa del concentrado
6. Imprime el concentrado	

CU16 Generar historia académica

Acción del Actor	Respuesta del Sistema
1. Introduce la matrícula del alumno	2. Busca en la base de datos el alumno y busca los registros de calificaciones que correspondan a su matrícula
	3. Muestra una vista previa de la historia académica
4. Imprime la historia académica	

Dentro del análisis de requerimientos existe lo que se llama la especificación complementaria. Respecto a la interfaz de usuario, el objetivo es lograr un sistema actualizado de manera que los usuarios comprendan la aplicación como un caso típico en el que se manejan ventanas, botones, menús desplegables o formularios del tipo de Windows y con el cual se puede interactuar por medio del teclado y el ratón.

La posibilidad de utilizar un escáner láser de código de barras está abierta y será adaptable al sistema en una versión posterior.

En relación a las reglas de dominio para la Escuela en cuestión se tiene:

ID	Regla	Grado de variación	Fuente
R1	La carrera tiene una duración de 8 semestres	Bajo	Actual programa de estudios de la carrera

R2	Durante cada semestre se cursan 6 materias	Bajo	Actual programa de estudios de la carrera
R3	En un semestre se cuentan 3 calificaciones parciales, una calificación final, un porcentaje de asistencias y la calificación definitiva	Bajo	Actual programa de estudios de la carrera
R4	Los alumnos con al menos el 60 % de asistencia tienen derecho a presentar exámenes extraordinarios, si es que no aprueban el curso ordinario.	Bajo	Actual programa de estudios de la carrera
R5	Si el porcentaje de asistencia de un alumno es menor que el 60 % el profesor asentara en el acta de calificaciones "S/D" y el alumno deberá recursar la materia.	Bajo	Actual programa de estudios de la carrera
R6	Los alumnos con al menos 80 % de asistencia tienen derecho a calificación final, y si no aprueban el curso, a presentar exámenes extraordinarios.	Bajo	Actual programa de estudios de la carrera
R7	Los alumnos con derecho a calificación final con 8.5 o más de promedio en los tres exámenes de periodo exentan el examen final.	Bajo	Actual programa de estudios de la carrera
R8	La calificación definitiva en un acta de calificaciones o en una historia académica o en cualquier reporte se expresa con un valor entero mayor o igual que 5 y menor o igual que 10.	Bajo	Actual programa de estudios de la carrera
R9	Las 3 calificaciones de periodo o la calificación final pueden expresarse como un valor real positivo mayor o igual que 0 y menor o igual que 10 (se recomienda utilizar una mantisa con 2 o 3 cifras como máximo).	Bajo	Actual programa de estudios de la carrera
R10	Si el alumno no se presenta en algún examen de periodo, el profesor asentara en el acta "N.P."	Bajo	Actual programa de estudios de la carrera
R11	El valor del promedio para obtener la calificación definitiva se puede "redondear" si es un valor aprobatorio.	Bajo	Actual programa de estudios de la carrera

Del glosario utilizado para la aplicación de la fase de inicio en la Escuela Superior de Ingeniería Automotriz (ESIA) se tiene:

Término	Descripción	Tipos de dato
Año, periodo, semestre y grupo	Datos para la clasificación de los grupos; Año – año en curso, Periodo – corresponde al primer o segundo semestre del año en curso Semestre – semestre de la carrera que cursa el alumno Grupo – letra distintiva para reconocer los diferentes grupos de un mismo semestre.	Alfanumérico (10)
Materia	Asignatura impartida por el profesor.	Alfanumérico (20)
Calificación	Valor obtenido por el alumno en cada materia.	Numérico (2)
Matrícula	Clave de identificación personal única para cada alumno	Alfanumérico (10)
RFC	Registro Federal de Contribuyentes	Alfanumérico (10)
Acta	Reporte elaborado para cada materia del semestre que muestra los alumnos inscritos con su respectiva calificación y faltas	Reporte listado
Boleta	Reporte elaborado para cada alumno que muestra las materias en las que estaba inscrito con su respectiva calificación y faltas	Reporte listado
Concentrado	Reporte elaborado para cada grupo que muestra las materias correspondientes al semestre y los alumnos inscritos con su respectivas calificaciones definitivas	Reporte listado

CAPÍTULO III

FASE DE ELABORACIÓN

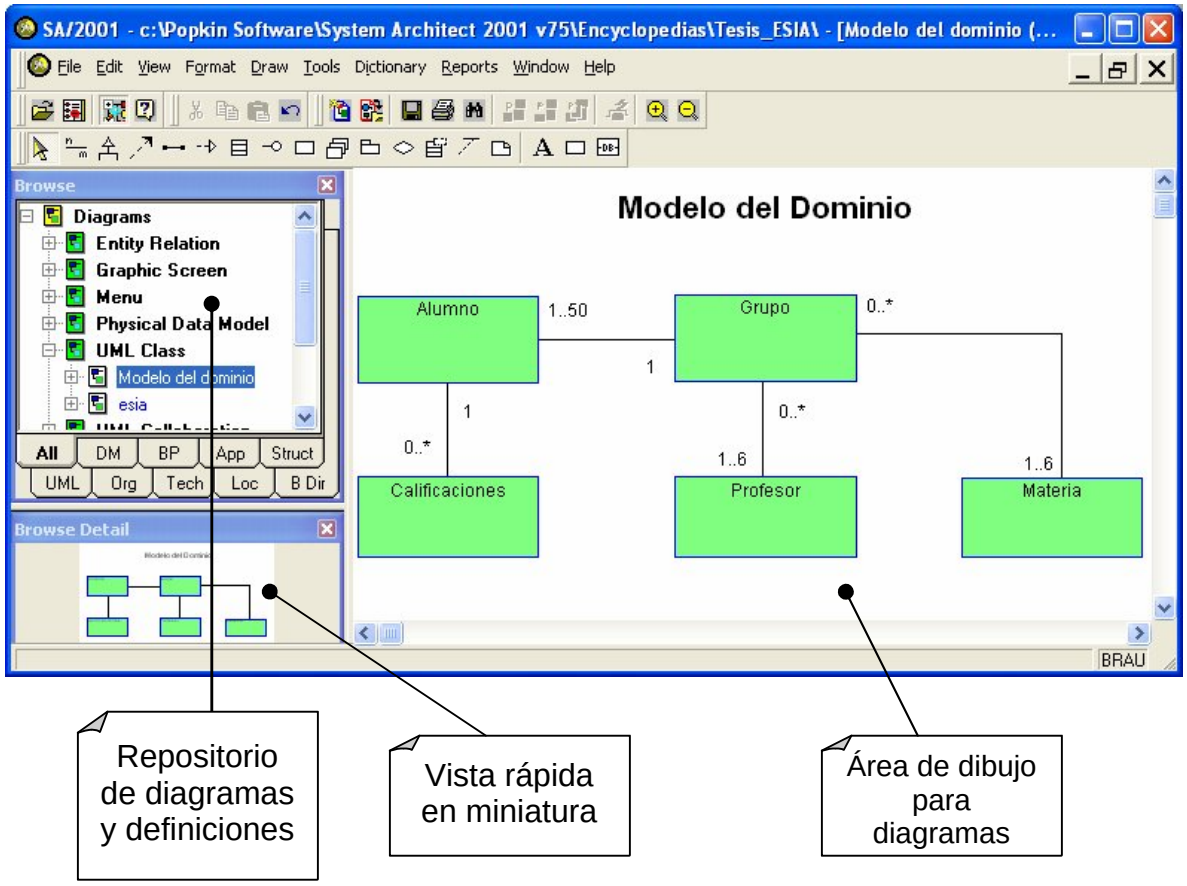
3.1 Modelado del negocio

En la fase de elaboración se tiene como objetivo determinar una visión más refinada, implementar iterativamente el núcleo central de la arquitectura, considerar la resolución de los riesgos altos e identificar otros requisitos importantes para obtener una estimación más realista. Para lograr lo anterior es necesario partir del modelado del negocio, de la determinación de los diagramas de secuencia y del refinamiento de requerimientos.

Dentro de esta fase del Proceso Unificado es conveniente adoptar un modelado de negocio como herramienta para entender y comunicar la estructura y la dinámica de la organización en la que se va a desplegar un sistema. Para ello, se vale de dos artefactos clave; el Modelo del Dominio y los Diagramas de Secuencia del Sistema.

El modelo del dominio sólo representa en principio; los objetos del mundo real no necesariamente deben ser las clases u objetos de software que se utilizarán. Sin embargo, generalmente se utilizan como los prototipos de las estructuras de datos. Este modelo es el primer acercamiento de la arquitectura del sistema, de manera que a partir de este modelo se puedan agregar los detalles de comportamiento para estos objetos. En la siguiente figura se muestra este modelo de dominio aplicado a la Escuela Superior de Ingeniería Automotriz:

Figura 3.1
Modelo del dominio elaborado con System Architect

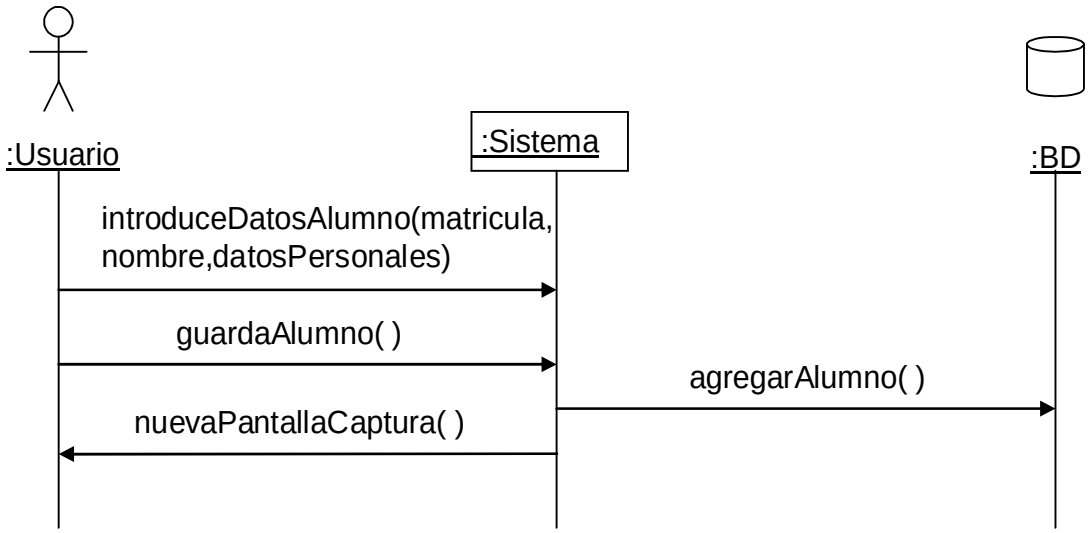


3.2 Diagramas de secuencia del sistema

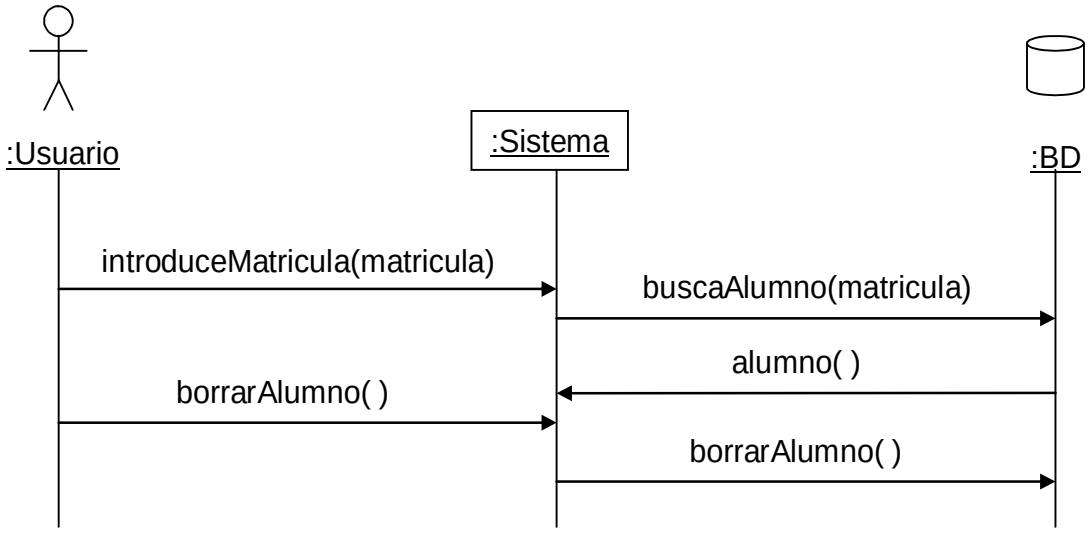
Un diagrama de secuencia muestra los eventos del sistema para un escenario de un caso de uso específico, los actores externos que interactúan directamente con el sistema, el sistema (como una caja negra) y los eventos del sistema que genera el actor. El tiempo avanza de arriba hacia abajo, y el orden de los eventos debe seguir la ordenación en el caso de uso.

En la aplicación de los diagramas de secuencia para el ESIA se considera lo que corresponde a los alumnos, al profesor, al grupo, calificaciones, concentrados e historia académica, en seguida se muestran los diagramas de secuencia respectivos:

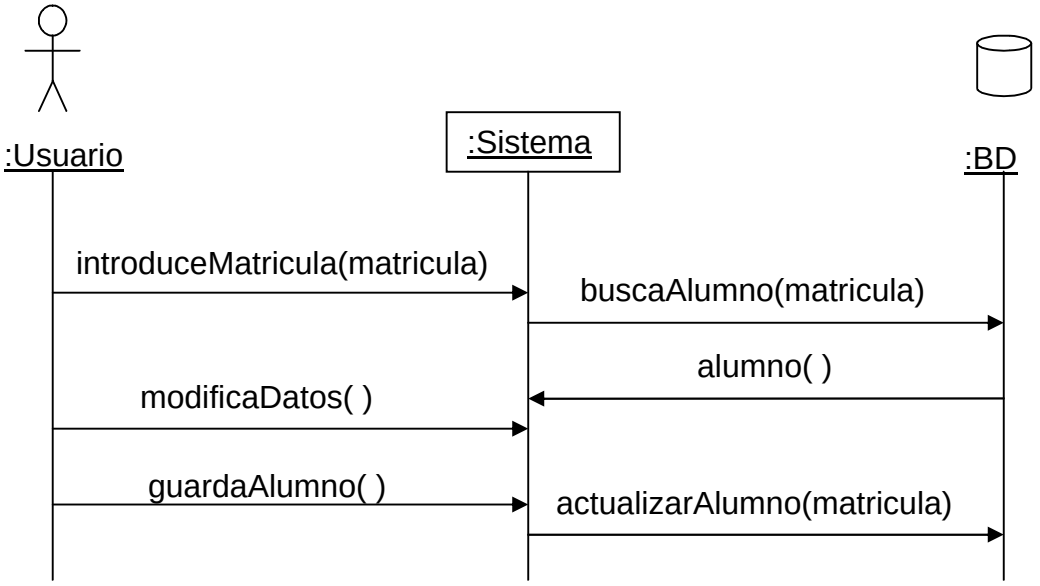
CU1 Alta de alumno



CU2 Baja de alumno

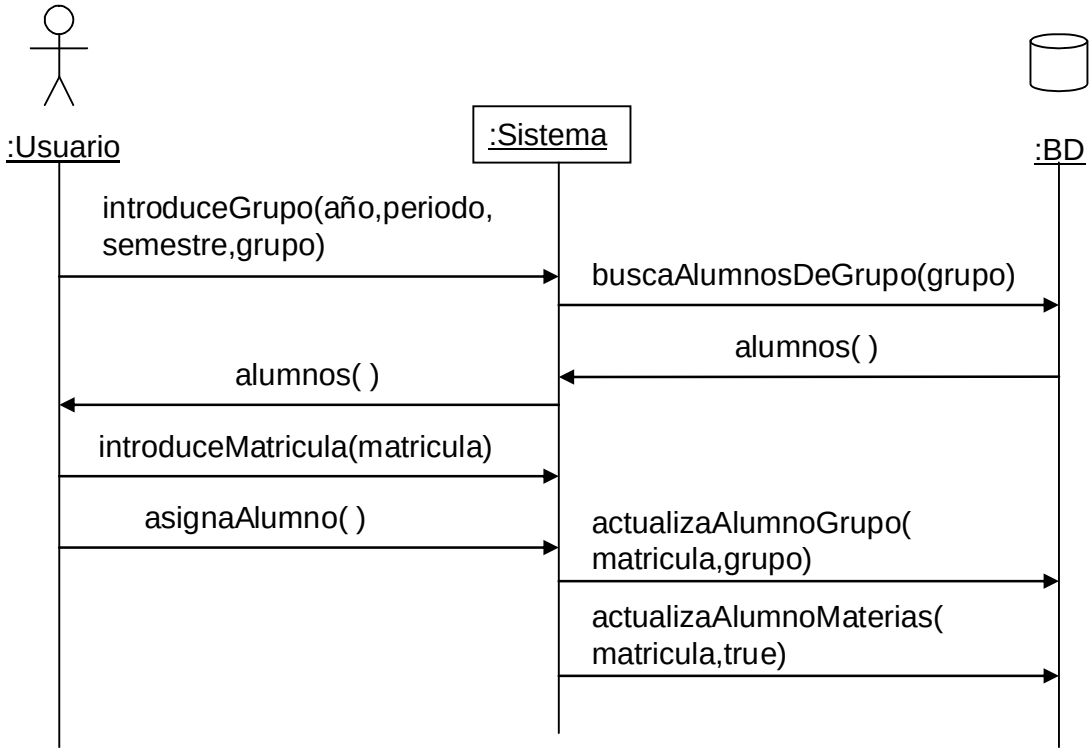


CU 3 Cambio datos de alumno

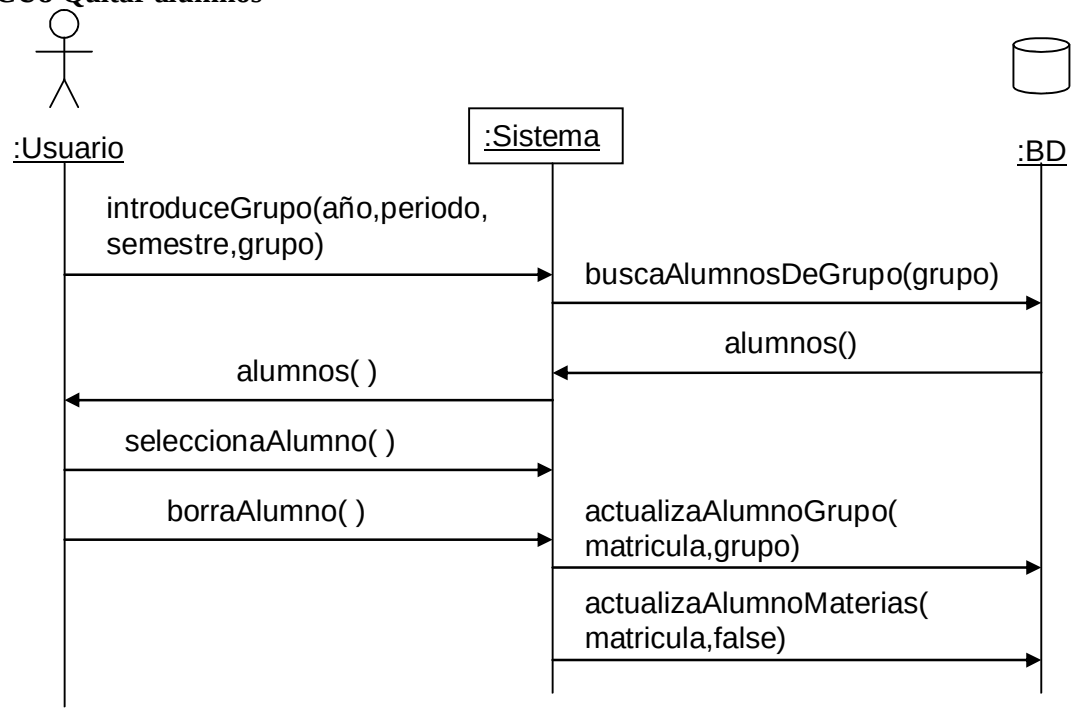


Los diagramas de secuencia del sistema que corresponden a los casos de uso 4, 5 y 6 son idénticos a los anteriores tres diagramas por lo cual no es necesario mostrar dichas secuencias, solo cabe mencionar que para los profesores se requiere utilizar el parámetro rfc en lugar de la matrícula.

CU7 Asignar alumno a un grupo



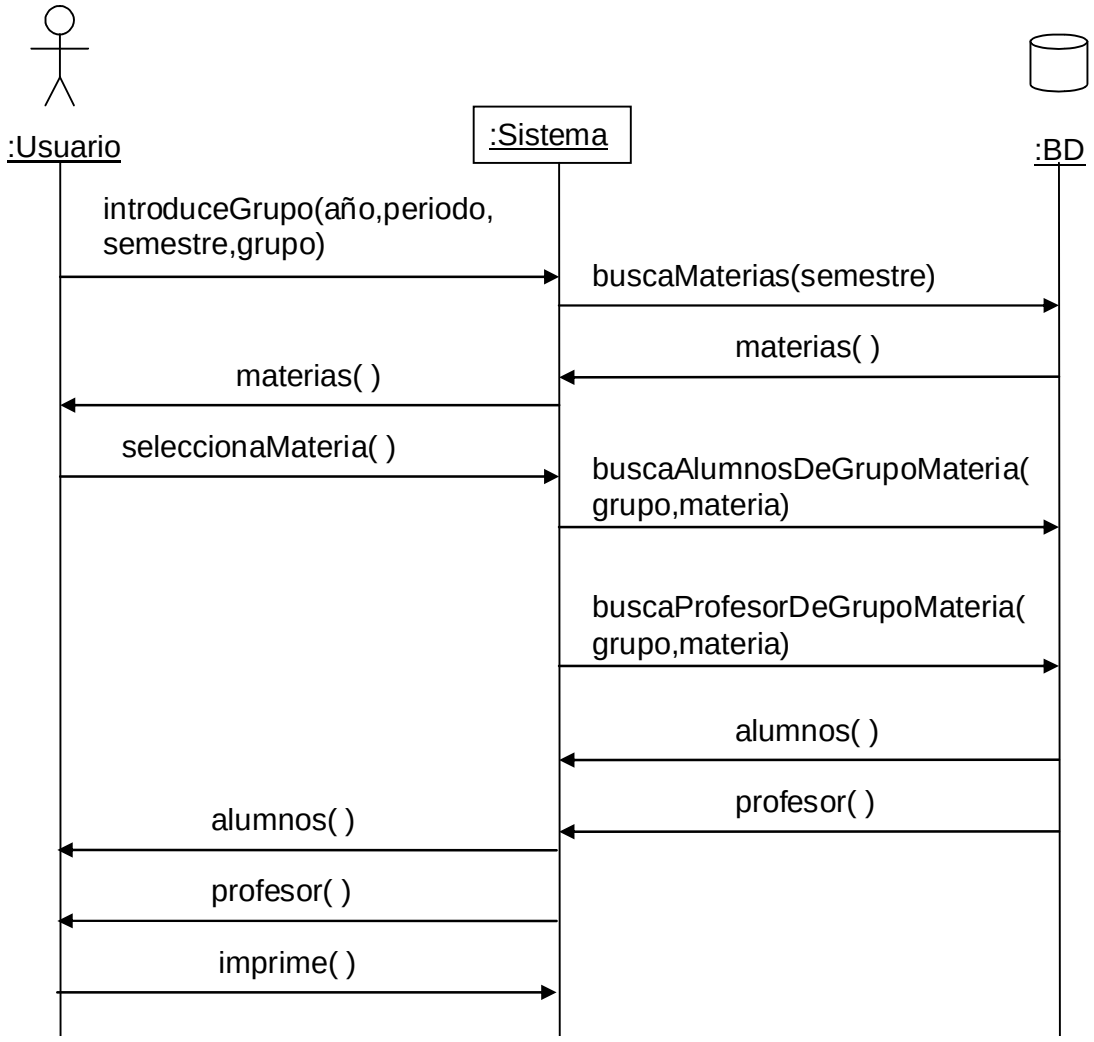
CU8 Quitar alumnos



Los casos de uso 9 y 10 muestran la secuencia de mensajes para asignar y quitar profesores de un grupo, estas secuencias son idénticas a los diagramas 7 y 8 pero utilizando el rfc del profesor en lugar de la matrícula. En estos diagramas la importancia es la de resaltar los siguientes mensajes:

1. introduceGrupo().- cuando el usuario envía este mensaje, el sistema prepara una consulta a la base de datos para regresar una colección o arreglo de registros de alumnos que sea presentado en pantalla.
2. asignaAlumno().- el usuario envía este mensaje con los datos de un alumno seleccionado en una lista, en la base de datos se agregan o se prenden los valores que indican el grupo y materias correspondientes para el alumno.
3. borraAlumno().- con este mensaje la finalidad es que el objeto alumno cambie de estado y NO que sea borrado de la base de datos.
4. actualizaAlumnoGrupo().- este mensaje significa una instrucción a la base de datos para cambiar el valor del grupo al que pertenece el alumno.
5. actualizaAlumnoMaterias().- con este mensaje lo que se desea es cambiar el valor de las materias que indican cuales son las que un alumno esta cursando actualmente.

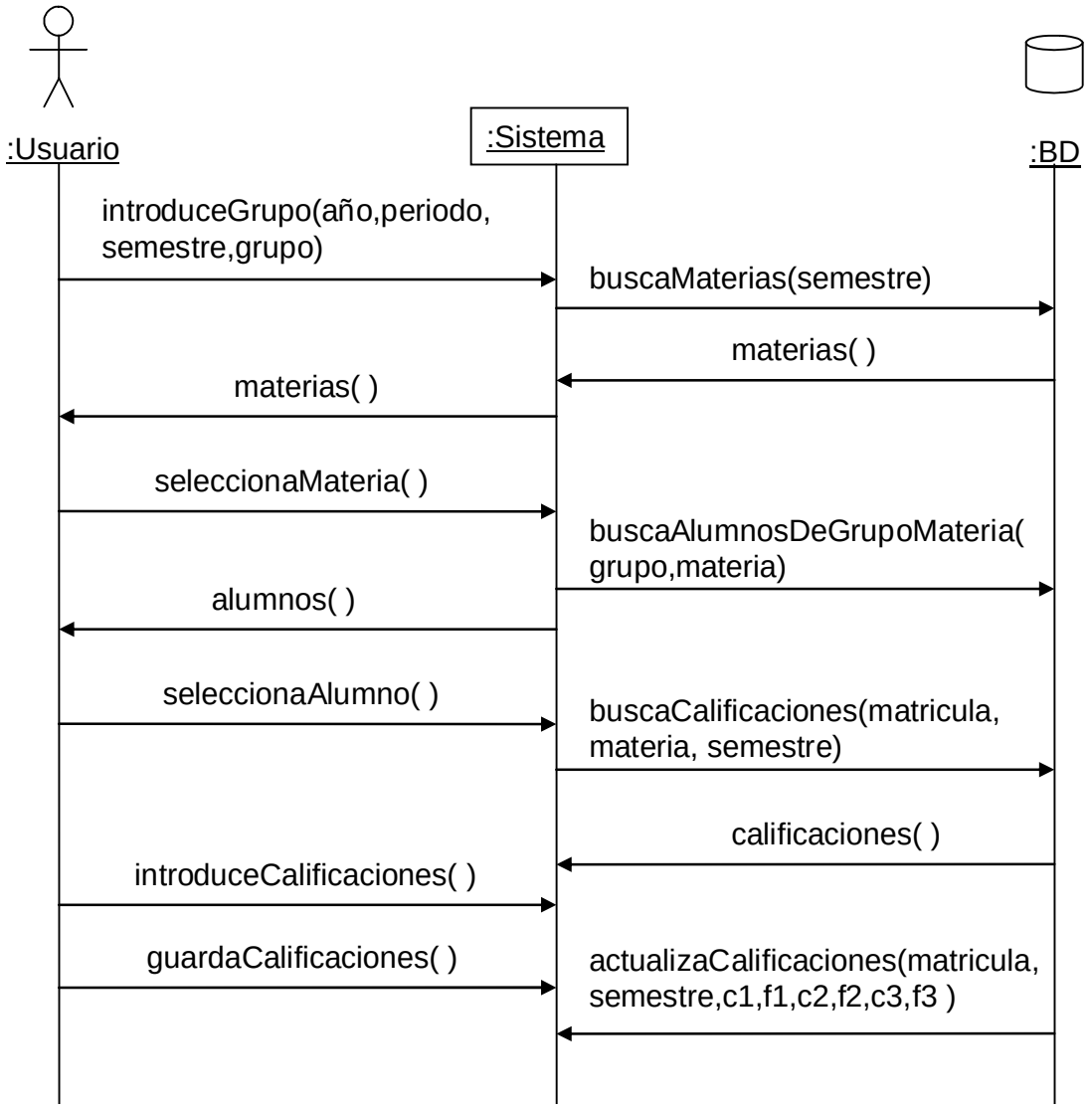
CU11 Generar lista de asistencia



Mensajes importantes:

En este diagrama se observa que los mensajes materias(), alumnos() y profesor(), solo se pasan de un objeto a otro porque finalmente este caso de uso es para reportar un contenido presentándolo en pantalla al usuario y no requiere realizar operaciones o el cambio de estado de algún objeto en particular.

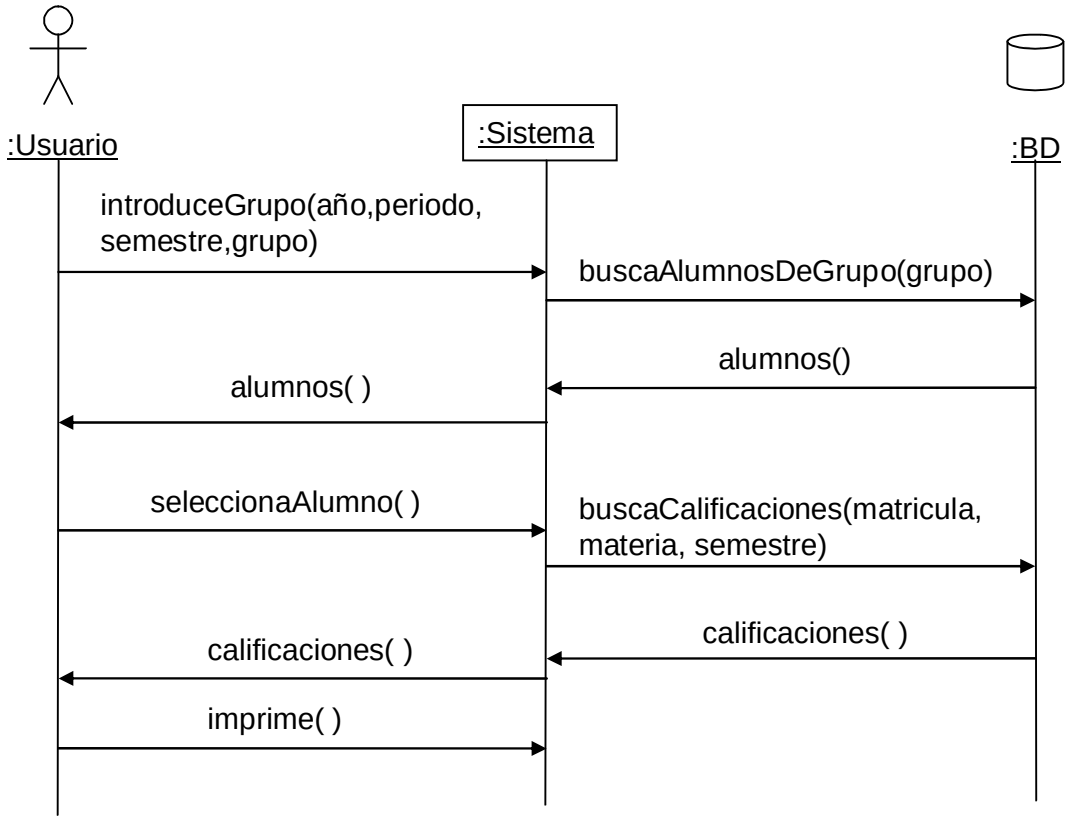
CU12 Capturar calificaciones



Mensajes importantes:

actualizaCalificaciones().- en este mensaje se consideran todos los parámetros para capturar todos los periodos correspondientes a una materia en un semestre determinado. Se considera de esta manera porque existe la posibilidad de capturar los valores en un componente que se le conoce como una rejilla o grid, es decir, una hoja con renglones y columnas que permite al usuario navegar entre las celdas y modificar los valores en ellas y al final se guarda la información de una sola vez.

CU13 Generar boletas de calificaciones

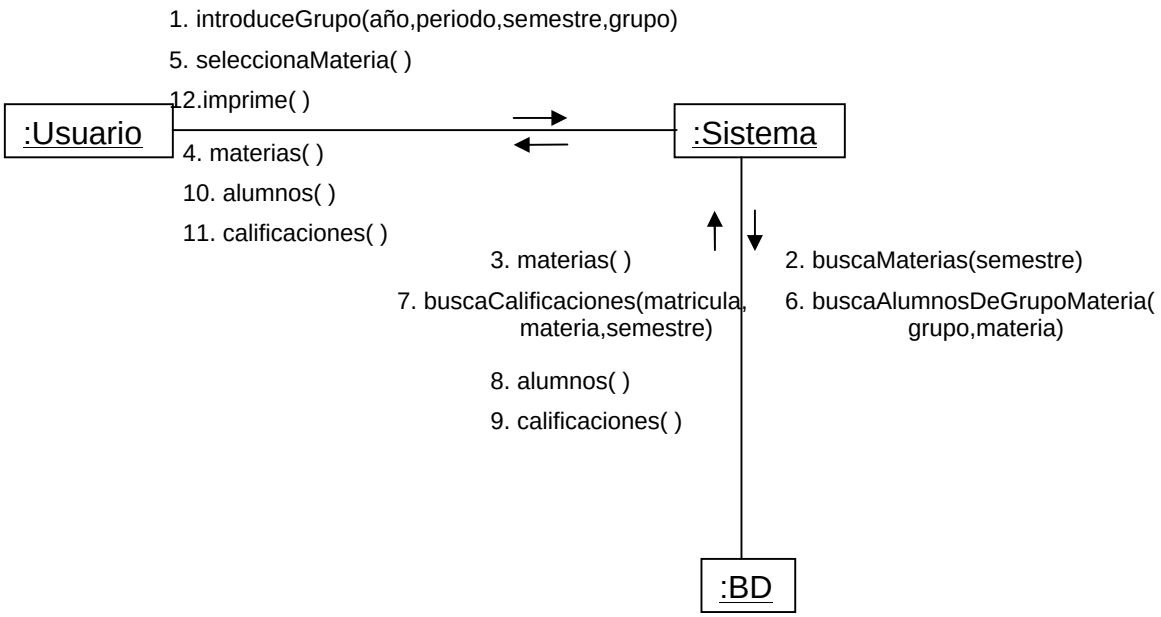


En los diagramas de los casos de uso 13,14,15 y 16 se describe la secuencia de mensajes para los eventos del sistema que requieren presentar información en pantalla. Las operaciones que más se utilizan para estos eventos son las de búsquedas de información y las operaciones para enviar y recibir colecciones o arreglos de objetos, por ejemplo calificaciones.

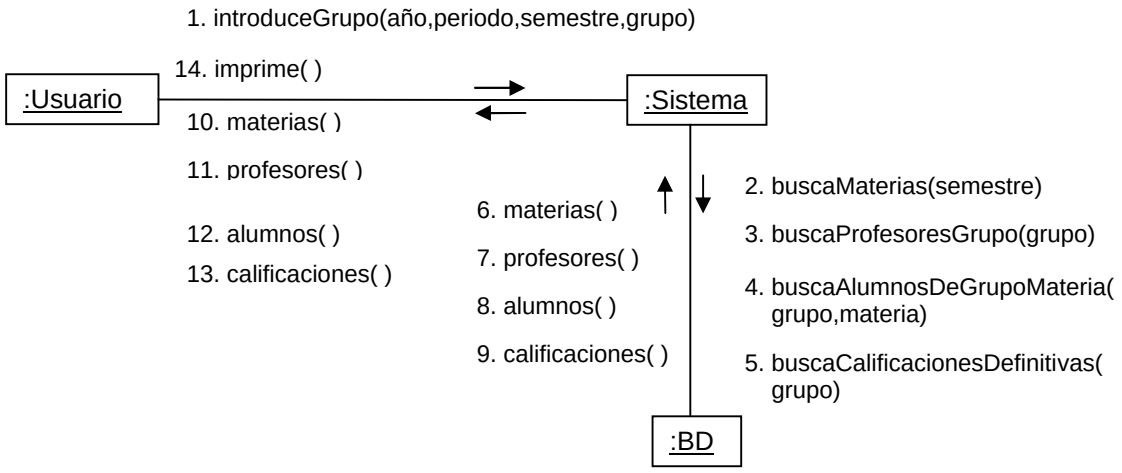
Otra forma de representar la secuencia de envió de mensajes es utilizando los diagramas de colaboración. Estos diagramas muestran el orden de los mensajes enumerando cada uno según corresponda al caso de uso. Los diagramas de colaboración son otra forma de representar la interacción entre objetos. Los diagramas de secuencia y los diagramas de colaboración no son excluyentes, sino que se complementan para dar mayor claridad al entendimiento de casos de uso en específico.

Los diagramas de los casos de uso 14,15 y 16 se muestran como diagramas de colaboración.

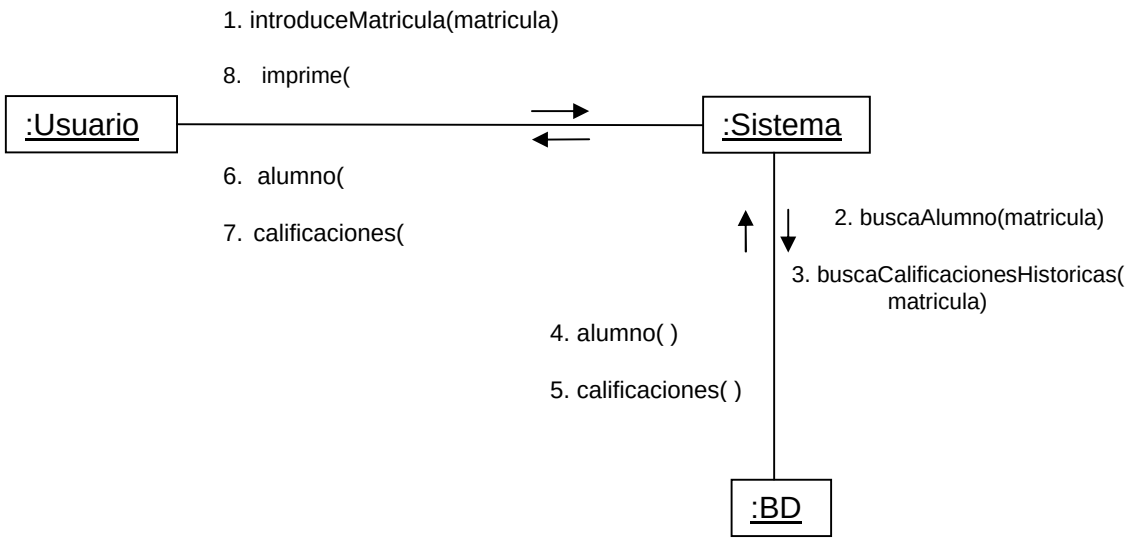
CU 14 Generar acta de calificaciones



CU 15 Generar concentrados



CU 16 Generar historia académica



3.3 Refinamiento de requerimientos

Habiendo aplicado el modelado del negocio y dentro del refinamiento de requerimientos se utilizan los casos de uso en detalle (contratos de operaciones). Los contratos de operaciones sirven para describir de manera más detallada el comportamiento del sistema para los casos de uso que resultan generales o complejos. Se requiere mayor entendimiento para describir los cambios de estado de los objetos del modelo del dominio.

En general, un contrato está definido en tres secciones que son:

- Operación: Nombre de la operación y parámetros
- Referencia: Casos de uso en los que puede tener lugar esta operación
- Detalle: Declaraciones relevantes sobre las acciones del sistema o de los objetos del modelo de dominio durante la ejecución de la operación. Y declaraciones de los objetos del modelo del dominio que entregan como resultado.

En la aplicación de los contratos de operaciones para el ESIA se hace referencia a las operaciones de búsqueda, creación y actualización de alumnos, materia, profesor y calificaciones. En seguida se muestran los contratos de operaciones respectivos:

1. **Busca Alumnos De Grupo (grupo)**

Referencia	CU7, CU8, CU13
Detalle	-Busca en la base de datos los alumnos que en el campo grupo coincida con el introducido por el usuario. -Crea las instancias de alumnos y las actualiza con los datos de la base. -Crea un arreglo del tipo Alumno [ ] y agrega las instancias.

2. **buscaMaterias (semestre)**

Referencia	CU9, CU11, CU12, CU14, CU15
Detalle	-Busca en la base de datos las materias del semestre indicado por el usuario. -Crea las instancias de materias y las actualiza con los datos de la base. -Crea un arreglo del tipo Materia [ ] y agrega las instancias.

3. **buscaProfesor (rfc)**

Referencia	CU9
Detalle	-Busca en la base de datos el profesor con el RFC indicado por el usuario. -Crea una instancia de profesor y la actualiza con los datos de la base.

4. **buscaProfesoresGrupo (grupo)**

Referencia	CU11, CU12, CU14, CU15
Detalle	-Busca en la base de datos los profesores del grupo indicado por el usuario. -Crea las instancias de profesor y las actualiza con los datos de la base. -Crea un arreglo del tipo Profesor [ ] y agrega las instancias.

5. **buscaAlumnosDeGrupoMateria (grupo, materia)**

Referencia	CU11, CU12, CU14, CU15
Detalle	-Busca en la base de datos los alumnos que en el campo grupo coincida con el introducido por el usuario y que contengan el valor de true en el campo de la materia indicada. -Crea las instancias de alumnos y las actualiza con los datos de la base. -Crea un arreglo del tipo Alumno [ ] y agrega las instancias.

6. **buscaProfesorDeGrupoMateria (grupo, materia)**

Referencia	CU11
Detalle	-Busca en la base de datos el grupo indicado por el usuario. -Obtiene el RFC del profesor para la materia indicada por el usuario. -Busca en la base de datos el profesor con el RFC obtenido. -Crea una instancia de profesor y la actualiza con los datos de la base.

7. **buscaCalificaciones (matrícula, materia, semestre)**

Referencia	CU12, CU13, CU14
Detalle	-Busca en la base de datos las calificaciones que en los campos matrícula, materia y semestre coincidan con los indicados. -Crea las instancias de calificaciones y las actualiza con los datos de la base. -Crea un arreglo del tipo Calificación[ ] y agrega las instancias.

8. **buscaCalificacionesDefinitivas (grupo)**

Referencia	CU15
Detalle	-Busca en la base de datos las calificaciones que en el campo grupo coincida con el indicado. -Crea las instancias de calificaciones y las actualiza con los datos de la base. -Crea un arreglo del tipo Calificación[ ] y agrega las instancias.

9. busca alumno (matrícula)

Referencia	CU16
Detalle	-Busca en la base de datos el alumno con la matrícula indicada por el usuario. -Crea una instancia de alumno y la actualiza con los datos de la base.

10. buscaCalificacionesHistoricas (matrícula)

Referencia	CU16
Detalle	-Busca en la base de datos las calificaciones que en el campo matrícula coincida con el indicado. -Crea las instancias de calificaciones y las actualiza con los datos de la base. -Crea un arreglo del tipo Calificación [ ] y agrega las instancias.

11. actualizaAlumnoGrupo (matrícula, grupo)

Referencia	CU7, CU8
Detalle	-Actualiza en la base de datos el alumno que coincida con la matrícula indicada por el usuario poniendo en el campo grupo el valor que se esta indicando.

12. actualizaAlumnoMaterias (matrícula, true)

Referencia	CU7, CU8
Detalle	-Actualiza en la base de datos el alumno que coincida con la matrícula indicada por el usuario poniendo en los campos mat_1 hasta mat_6 el valor de “true” que indica que si esta inscrito en dicha materia.

13. actualiza calificaciones (matrícula, semestre, c1, f1, c2, f2, c3, f3)

Referencia	CU9, CU10
Detalle	-Actualiza en la base de datos el registro de calificaciones que coincida con la matrícula y semestre indicados por el usuario poniendo en los campos calificación y faltas los respectivos valores que se indican.

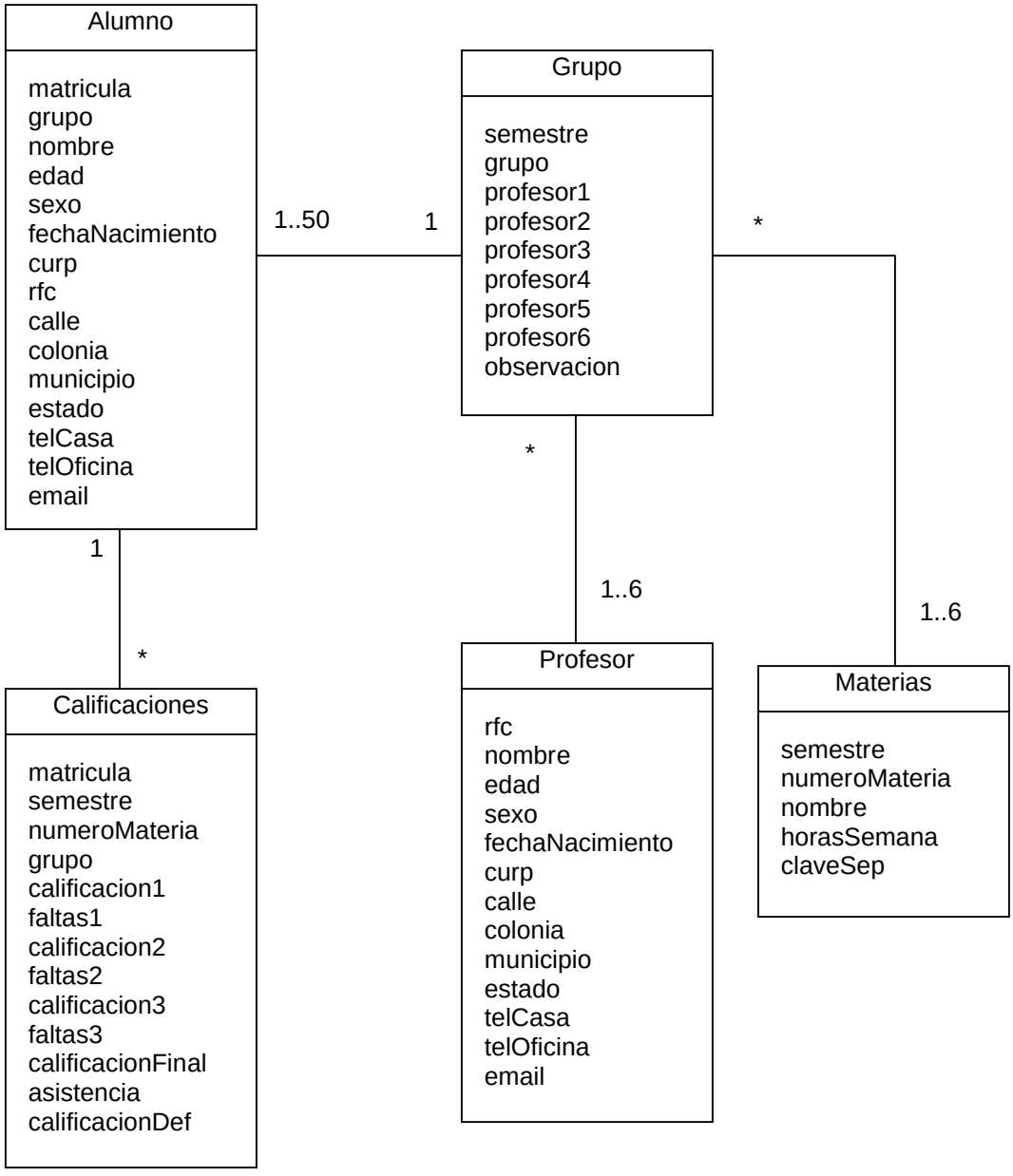
14. actualizaGrupoProfesor (grupo, rfc)

Referencia	CU9, CU10
Detalle	-Actualiza en la base de datos el grupo que coincida con el indicado por el usuario poniendo en el campo profesor el RFC que se esta indicando.

Una vez definido los contratos de operaciones, se procede a la elaboración de diagramas de clases. Ellos se establecen de forma definitiva las clases del sistema agregando los atributos a los objetos.

Para definir o elaborar el diagrama de clases, nos apoyamos en el modelo del dominio. Generalmente las clases del dominio prevalecen como las clases del sistema. En el caso del ESIA se muestran las clases que corresponden al alumno, grupo, calificaciones, profesor y materias, estableciendo las relaciones entre ellos.

Figura 3.2
Diagrama de clases



De lo anterior, la descripción del diagrama de clases se tiene lo siguiente para la aplicación en la Escuela Superior de Ingeniería Automotriz:

- Clase Grupo: Un grupo se identifica por el semestre que va del 1 al 8, por el grupo que es una letra, por ejemplo A, B o C y por la observación que indica que puede ser Ordinario o Extraordinario.

- Clase Alumno: Un alumno tiene un identificador único que es su matrícula, y se sabe en que grupo va porque cuando se inscribe en un grupo se guarda ese valor en el campo grupo.
- Relación Grupo-Alumno: Un solo grupo tiene desde 1 hasta 50 alumnos inscritos, cada alumno tiene solamente un grupo en el cual se inscribe.
- Clase Profesor: Un profesor tiene un identificador único que es su rfc.
- Relación Grupo-Profesor: Un solo grupo tiene desde 1 hasta 6 profesores asignados, cada rfc de los profesores asignados al grupo se guardan en los campos profesor1 hasta profesor6.
- Clase Materias: Cada materia se identifica por el semestre y por un número de materia que va del 1 al 6 ya que es el número de materias que se imparte por semestre.
- Relación Grupo-Materias: [1] A cada grupo le corresponden de 1 a 6 materias y se sabe cuales son por el semestre de cada materia.
- Clase Calificaciones: Las calificaciones tienen 4 llaves para su identificación que son: matrícula, semestre, grupo, número Materia.
- Relación Alumno-Calificaciones: Cada alumno puede tener varias calificaciones incluso para una misma materia y semestre.

[1] Nota: En un grupo se sabe que materia imparte cada profesor porque se puede hacer una deducción de que la materia con el numero 1 es impartida por el profesor1

CAPÍTULO IV

FASE DE CONSTRUCCIÓN

4.1 Ingeniería de código

En la fase de construcción se busca la implementación iterativa de los requisitos, empezando por los de menor riesgo y por los elementos más fáciles, como también se busca la preparación para el despliegue, es decir la puesta en producción del sistema para que pueda ser utilizado por los usuarios. Para lograr lo anterior es necesario realizar la ingeniería de código.

La ingeniería de código consiste en el desarrollo de software a partir de los modelos de diseño que se hayan realizado, se puede utilizar la información de los modelos para generar el código fuente en un lenguaje de programación especifico como Java, también para generar los scripts de SQL para una base de datos y de las clases para crear los prototipos de las interfaces de usuario.

La idea de realizar una implementación iterativa se apoya en el uso de las herramientas CASE, ya que así podemos revisar de manera rápida la presentación visual del sistema, que precisamente se genera con estas herramientas. En esta primera iteración no se aplican los detalles de funcionalidad y no es sino hasta que es aceptada y se esta conforme con dicha presentación que se procede a implementar la programación y configuración completa.

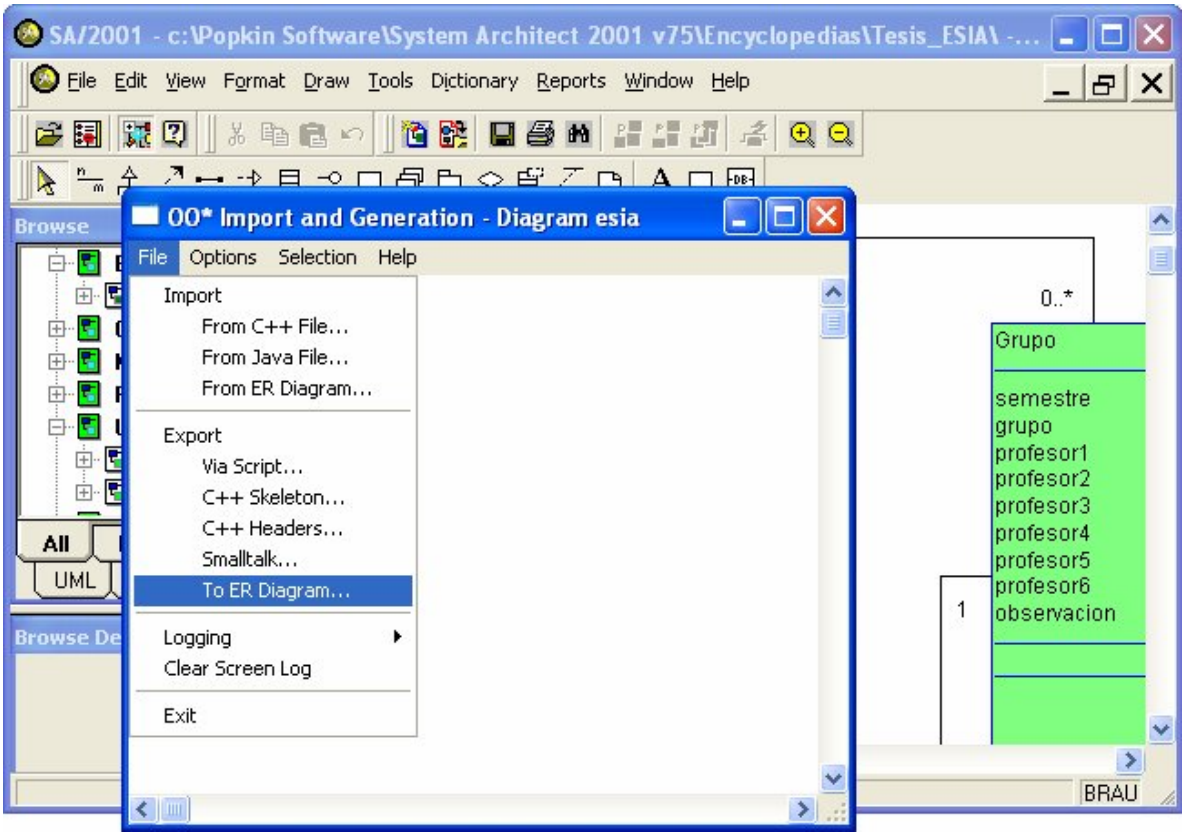
Entre los elementos que ofrecen más valor para el sistema y que a partir de los modelos requieran poca inversión de tiempo y esfuerzo, tenemos la base de datos y aplicaciones del tipo maestro detalle.

Base de datos:

Para crear la estructura de la base de datos en un Sistema Manejador como lo es Oracle, es necesario partir de un diagrama de Entidad-Relación. Una de las ventajas de utilizar una herramienta CASE para llevar a cabo esta tarea es que se pueden controlar los cambios al modelo de datos guardando la versión correspondiente, y para su implementación se recurre a las utilerías de generación del modelo físico.

Para realizar esta tarea en System Architect el proceso inicia con el diagrama de clases para después generar el diagrama de entidad-relación ya que en esta transformación, se hace una mapeo o correspondencia de una clase a una entidad.

Figura 4.1
Creación del diagrama de Entidad-Relación a partir del diagrama de clases



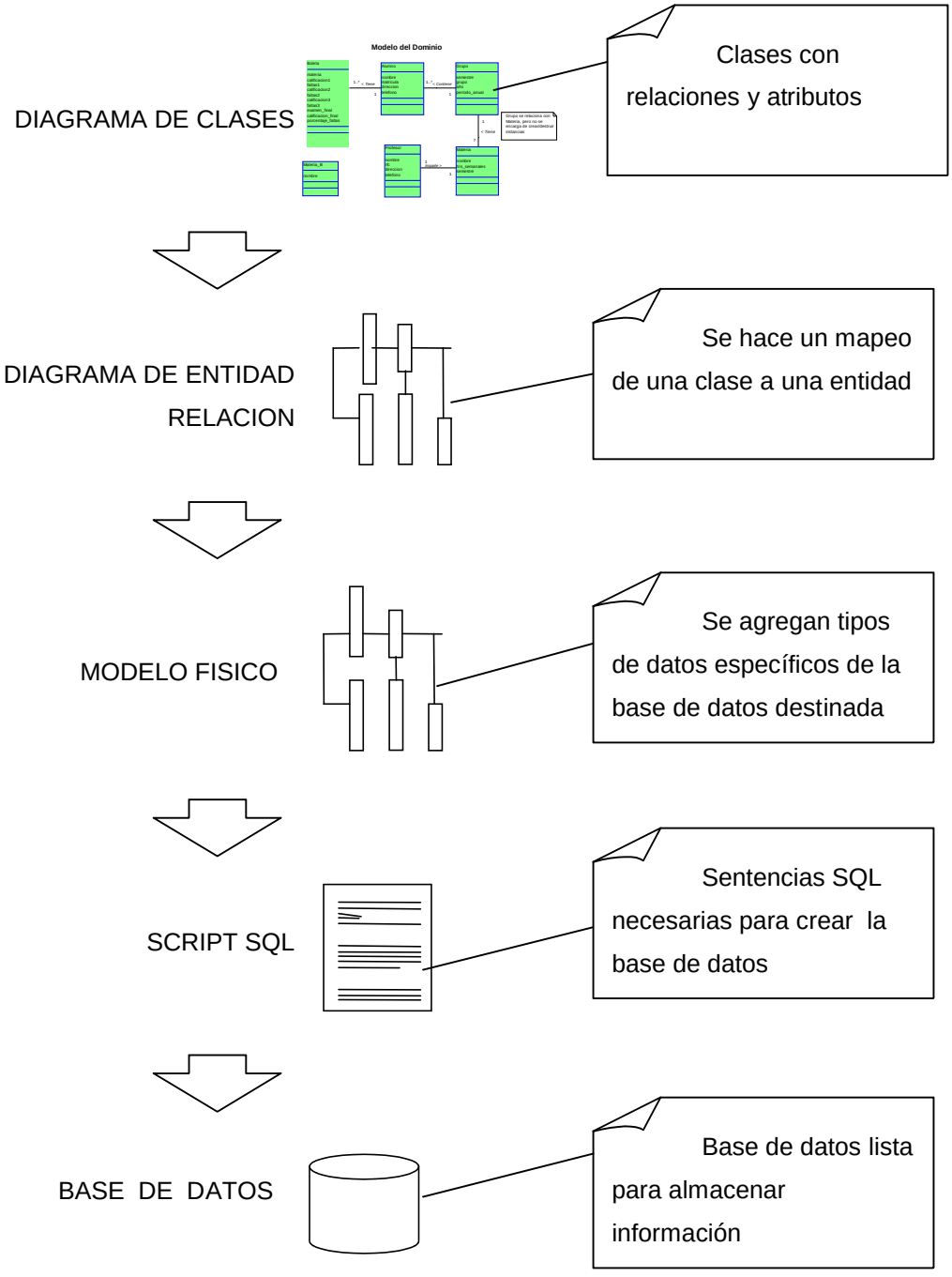
Después de tener el diagrama de Entidad-Relación (E-R) se genera el modelo físico de datos. El modelo físico es la representación del diagrama E-R pero cumpliendo con los tipos de datos del sistema donde radicará la base de datos.

Una vez que se tiene el modelo físico se emplea la utilidad llamada Generador de Esquema lo que entrega un archivo con el script de SQL, este script se ejecuta en la base de datos, mismo que fue aplicado al caso de la ESIA.

A continuación en la figura 4.2, se muestra la secuencia que se realiza para generar la estructura de la base comenzado desde los modelos de diseño.

Figura 4.2

Creación de la Base de Datos a partir de un diagrama de clases



Aplicación del tipo maestro detalle:

Las aplicaciones del tipo maestro-detalle facilitan el mantenimiento a las tablas que poseen relaciones con otras tablas.

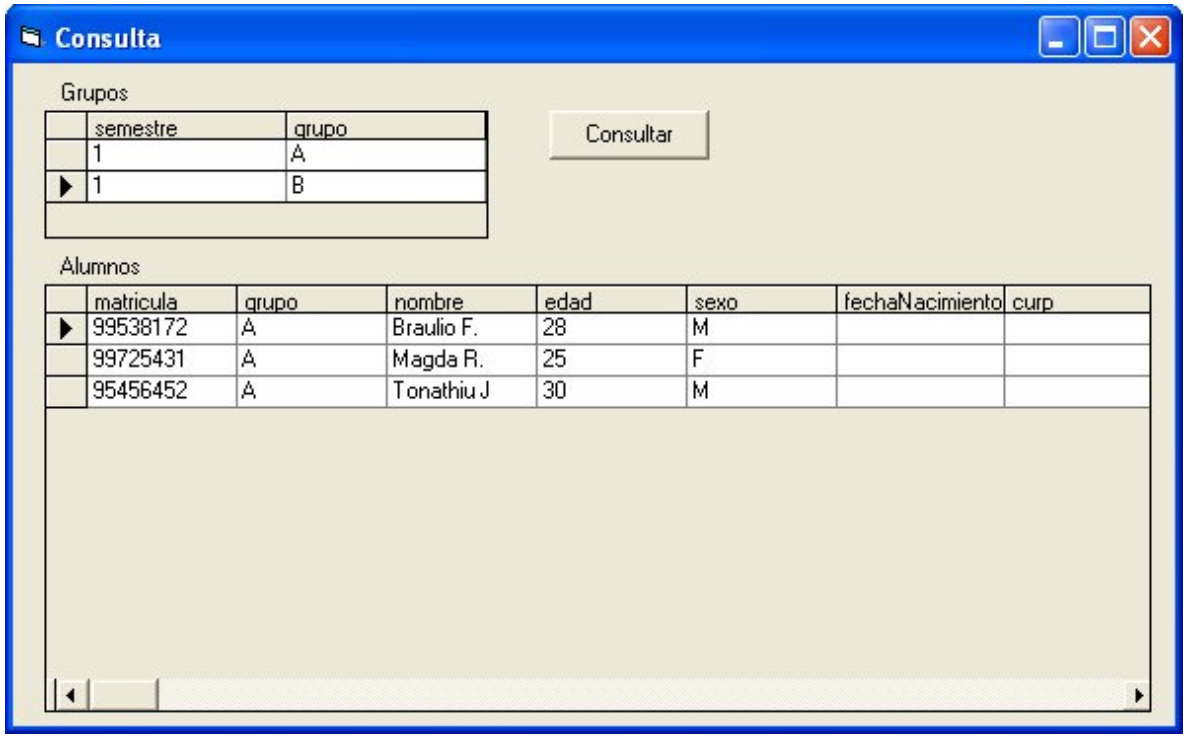
En el caso del modelo de datos del sistema de control escolar para la ESIA, cada grupo tiene una relación directa con alumnos, entonces la aplicación maestro detalle genera una ventana dividida en dos listas. En la lista superior todos los grupos disponibles y con cada grupo que se selecciona se pueden consultar en la lista inferior los alumnos que se encuentran inscritos en dicho grupo.

Para elaborar lo anterior, el System Architect tiene una función que le llama “Generación hacia 4GL” que significa generación de código hacia lenguajes de cuarta generación (4th Generation Language).

Esta función trabaja a partir del modelo de datos con la selección del diagrama de Entidad-Relación generando la aplicación en Visual Basic.

El System Architect nos entrega la pantalla que se muestra en la figura 4.3. El tiempo utilizado para generar la base de datos y después generar esta pantalla es de 15 minutos, esto quiere decir que a partir de los diagramas elaborados, se emplean estos 15 minutos para generar una muestra de la aplicación de consulta de alumnos por grupo.

Figura 4.3
Creación de una pantalla maestro-detalle



4.2 Interfaces de usuario

Para diseñar las pantallas requeridas en el sistema nos basamos en la lista de casos de uso que se definieron en la fase de inicio, los parámetros que se indican en los diagramas de secuencia y las operaciones del refinamiento de requisitos para después proceder a estructurar la información de acuerdo al cumplimiento de dichas operaciones.

Veamos la lista de casos de uso:

CU1 Alta de alumno
CU2 Baja de alumno
CU3 Cambio de datos
CU4 Alta de profesor
CU5 Baja de profesor
CU6 Cambio de datos
CU7 Asignar alumno a un grupo
CU8 Quitar alumnos
CU9 Asignar profesores
CU10 Quitar profesores
CU11 Generar lista de asistencia
CU12 Capturar calificaciones
CU13 Generar boletas de calificaciones
CU14 Generar actas de calificaciones
CU15 Generar concentrados
CU16 Generar historia académica

Esta lista la estructuramos según las funciones.

CU1 Alta de alumno	Altas, Bajas y Cambios de Alumnos
CU2 Baja de alumno	
CU3 Cambio de datos	
CU4 Alta de profesor	Altas, Bajas y Cambios de Profesores
CU5 Baja de profesor	
CU6 Cambio de datos	
CU7 Asignar alumno a un grupo	Asignar y Quitar Alumnos
CU8 Quitar alumnos	
CU9 Asignar profesores	Asignar y Quitar Profesores
CU10 Quitar profesores	
CU12 Capturar calificaciones	Ingresar Calificaciones
CU11 Generar lista de asistencia	Reportes y Listas
CU13 Generar boletas de calificaciones	
CU14 Generar actas de calificaciones	
CU15 Generar concentrados	
CU16 Generar historia académica	

Ahora tomamos las funciones de altas, bajas y cambios de alumnos y diseñamos las siguientes pantallas:

Figura 4.4 Alta Alumno

Figura 4.5 Baja de alumno

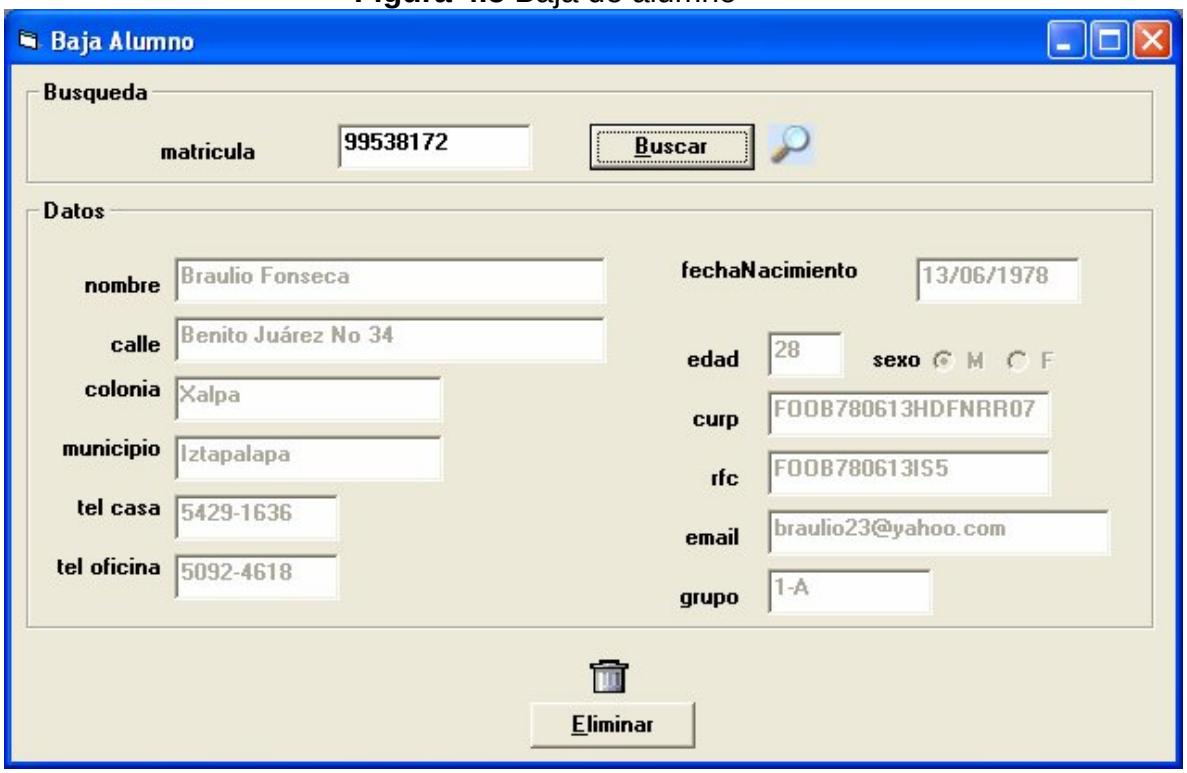
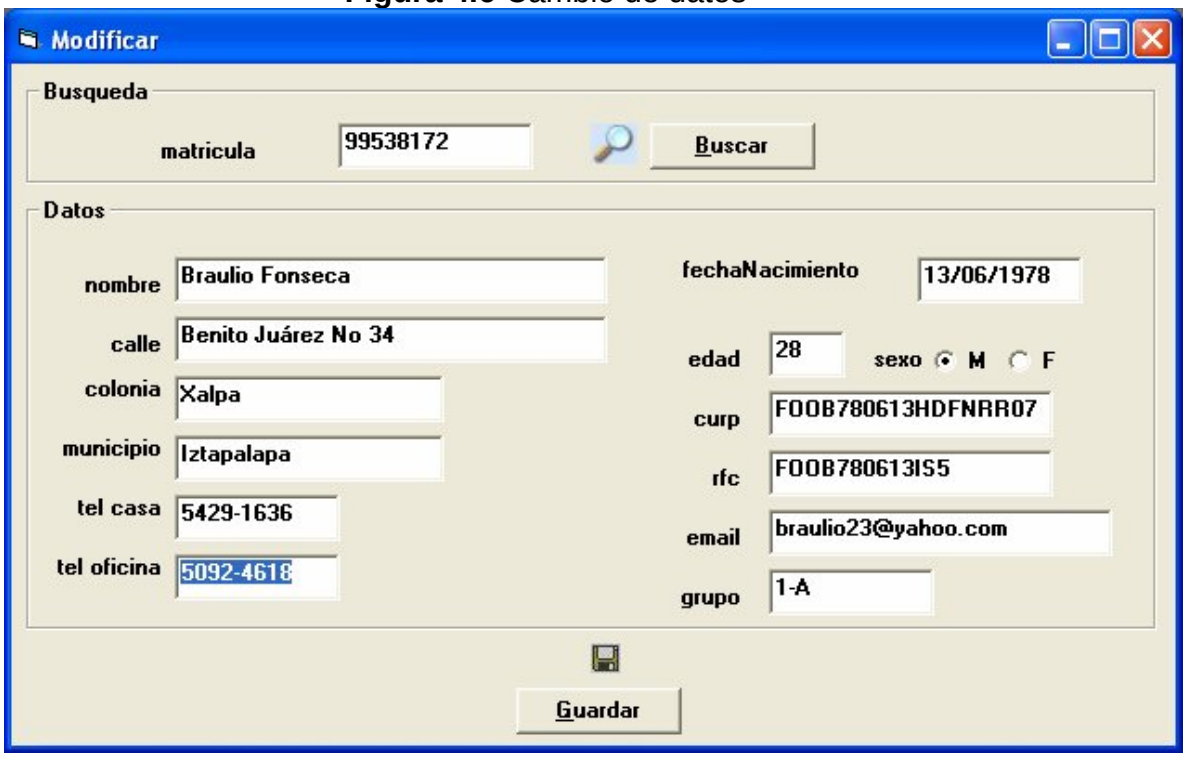


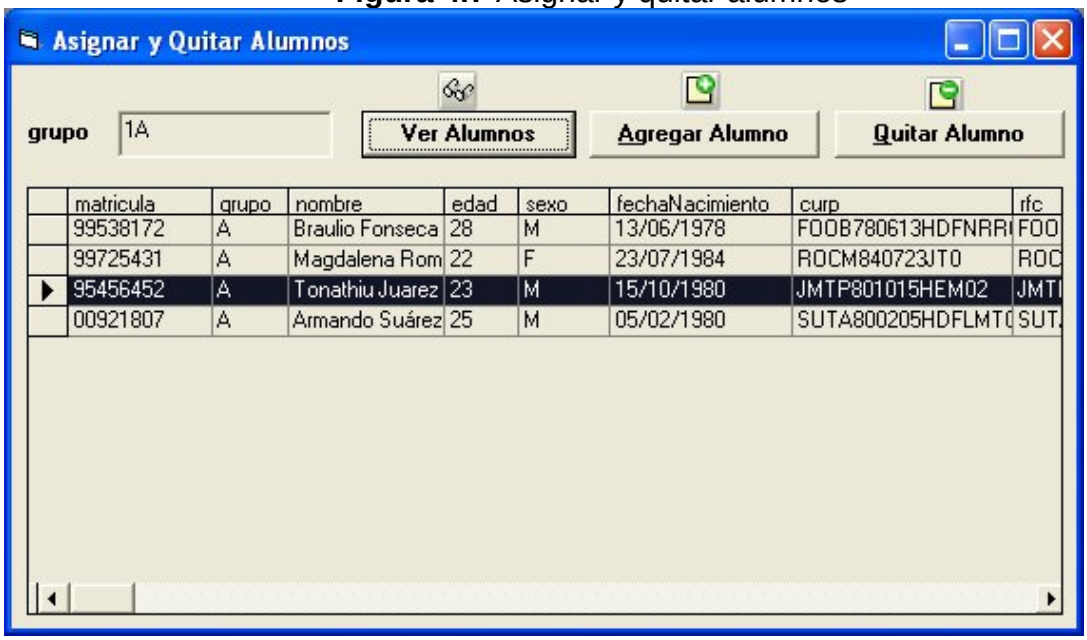
Figura 4.6 Cambio de datos



El siguiente grupo de funciones corresponde a las altas, bajas y cambios de profesores. El diseño y funcionalidad para estas pantallas es idéntica a las pantallas de altas, bajas y cambios de alumnos, por lo cual omitimos su presentación en este documento y hacemos una aclaración acerca del campo matricula, ya que los datos del profesor no requieren de este campo.

El siguiente grupo de funciones corresponde a asignar y quitar un alumno de un grupo, entonces el diseño de la pantalla para cumplir con estas funciones es el siguiente:

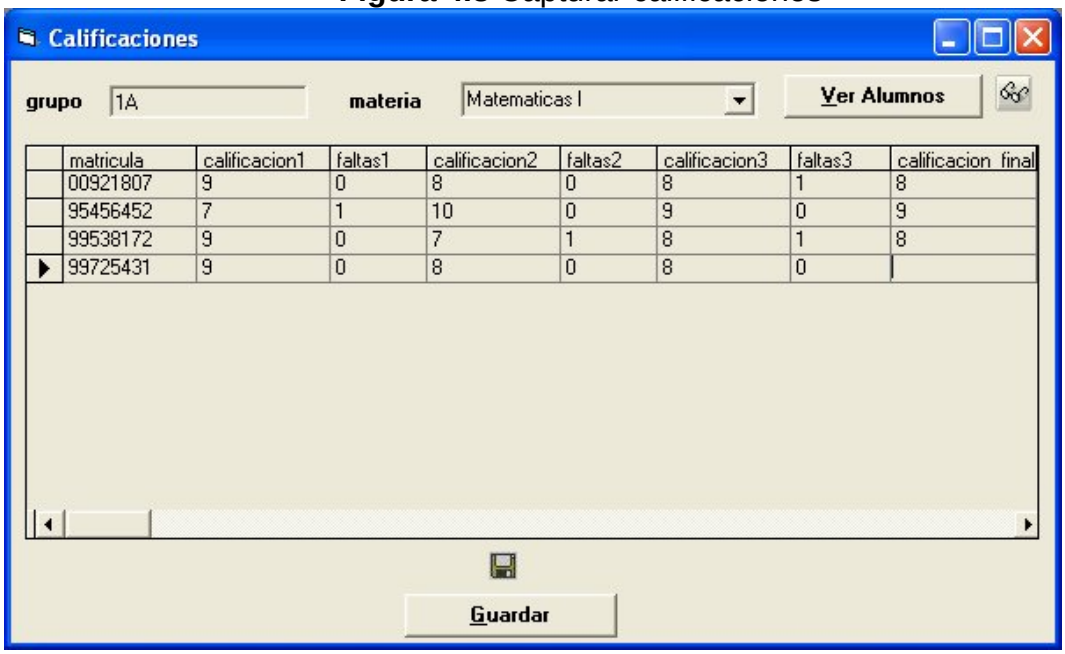
Figura 4.7 Asignar y quitar alumnos



El siguiente grupo de funciones corresponde a asignar y quitar profesores. El diseño y funcionalidad para esta pantalla es idéntica a la pantalla de asignar y quitar alumnos, por lo cual omitimos su presentación en este documento.

La siguiente función tiene el objetivo de capturar las calificaciones, en esta pantalla el diseño queda de la siguiente manera:

Figura 4.8 Capturar calificaciones



4.3 Pantallas de reportes

Los reportes propuestos para la Escuela Superior de Ingeniería Automotriz están basados en formatos sugeridos por la oficina de control escolar de la institución. En este sentido, se han elaborados los siguientes:

- Lista de asistencia
- Boleta de calificaciones
- Concentrado de calificaciones
- Acta de calificaciones

Figura 4.9
1. Lista de asistencia

CICLO 2005-2006

ESCUELA SUPERIOR DE INGENIERIA AUTOMOTRIZ

GRUPO _____ CLASES PROGRAMADAS _____
PROFESOR _____ CLASES IMPARTIDAS _____
MATERIA _____ No HORAS IMPARTIDA _____

MATRICULA	N/P	NOMBRE DEL ALUMNO	SEPTIEMBRE 2005						
			LUNES 1	MARTES 2	MIERCOLES	JUEVES 4	VIERNES 5	FIRMA DEL ALUMNO	OBSERVACIONES

Figura 4.10
2.- Boleta de calificaciones

NOMBRE DEL ALUMNO:_____

GRUPO: _____ SEMESTRE: _____

ASIGNATURA	CAL.1	FAL1	CAL.2	FAL2	CAL.3	FAL 3	CAL.FINAL	P.A	CAL.DEF.
1.-									
2.-									
3.-									
4.-									
5.-									
6.-									

Figura 4.11
3.- Concentrado de calificaciones

GRUPO: _____ **SEMESTRE:** _____

ASIGNATURA **PROFESOR**

1: _____ _____
2: _____ _____
3: _____ _____
4: _____ _____
5: _____ _____
6: _____ _____

No	NOMBRE DEL ALUMNO	CALFS. DEFINITIVAS					
		1	2	3	4	5	6

PROMEDIO DE CALIFICACIONES POR GRUPO: _____

PORCENTAJE DE ALUMNOS APROBADOS: _____

Figura 4.12

4.- Acta de calificaciones

ESCUELA SUPERIOR DE INGENIERIA AUTOMOTRIZ

RVO No 982013 DE FECHA 10 DE FEBRERO 1998

FOLIO _____

ACTA DE CALIFICACIONES DE EXAMENES ORDINARIOS O EXTRAORDINARIOS

CARRERA: _____ CICLO ESCOLAR: _____

PROFESOR: _____ SEMESTRE: _____

ASIGNATURA: _____ GRUPO: _____

CLAVE: 314 **TURNO : (M) (V)**

No	NOMBRE DEL ALUMNO	I PARCIAL		II PARCIAL		III PARCIAL		CAL. FINAL		CAL. DEF.
		CAL	FAL	CAL	FAL	CAL	FAL	CAL	% DE ASIS	CAL

CUADRO ESTADISTICO

	I	II	III	TOTAL	OBSERVACIONES
No DE ALUMNOS APROBADOS					
No DE ALUMNOS REPROBADOS					
PORCENTAJE DE APROBACION					

FECHA DE ENTREGA : _____

ENTREGA

_____

PROFESOR TITULAR

Vo Bo.

_____

ING. MIGUEL ANGEL LOPEZ VEGA

DIRECTOR

CONCLUSIONES

En relación al objetivo general del presente documento, relativo a señalar la aplicación del Proceso Unificado en el control escolar de la Escuela Superior de Ingeniería Automotriz tomando como fundamento el Análisis y Diseño Orientado a Objetos se concluye que:

- El Análisis y Diseño Orientado a Objetos requiere una metodología que integre el proceso de desarrollo y un lenguaje de modelado con herramientas y técnicas adecuadas.

- Los objetos del mundo real tienen características y comportamiento, y de esa misma manera, los objetos del mundo del software tienen variables y métodos.

- El Proceso Unificado es intuitivo e incremental; se comienza con las cosas más simples y fáciles de observar, para después ir avanzando hasta obtener el grado de detalle requerido.

- Dicho proceso es fácil de comprender y reproducir en el análisis y diseño de nuevos sistemas.

- La aplicación de la notación de UML facilita la explicación de los conceptos y se adapta al Proceso Unificado. Además, permite la representación de conceptos reales plasmados mediante diagramas y dibujos.

- La representación gráfica facilita el desarrollo de programas con diferentes lenguajes de programación, en lugar de utilizar diagramas de flujo o de bloques que son utilizados en la programación estructurada.

- La aplicación del Proceso Unificado en la Escuela Superior de Ingeniería Automotriz muestra la gran adaptabilidad del proceso a sistemas medianos y no tan complejos.

- De la aplicación de dicho proceso se logra hacer una variación al análisis y diseño adecuada para definir el problema, las actividades y las especificaciones particulares de la institución educativa.

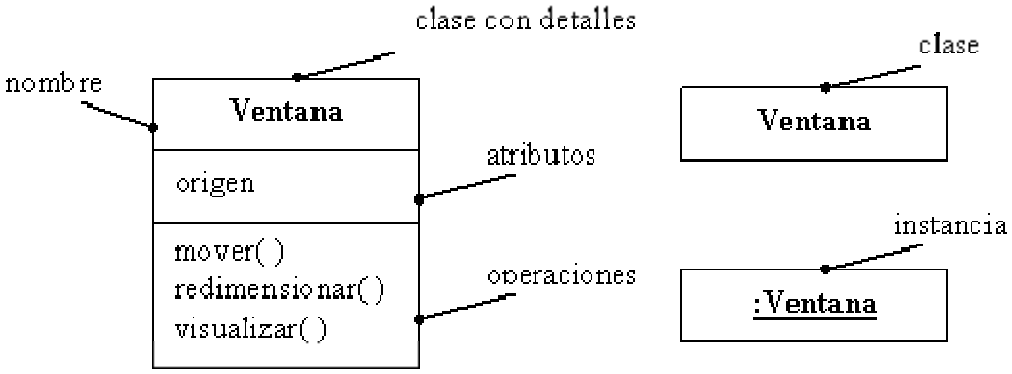
- El resultado inmediato es que ahora se cuenta con una documentación clara y bien definida a partir de la cual se puede llevar a cabo la programación y la implementación del sistema de control escolar.

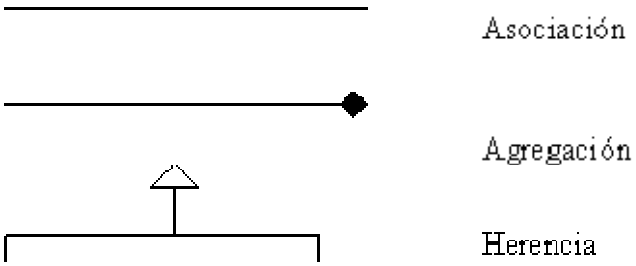
Finalmente, puede indicarse que este proyecto está listo para implantar la solución a los problemas de control escolar de la Escuela Superior de Ingeniería Automotriz, ya que se ha establecido un flujo de tareas con el objeto de ser automatizadas.

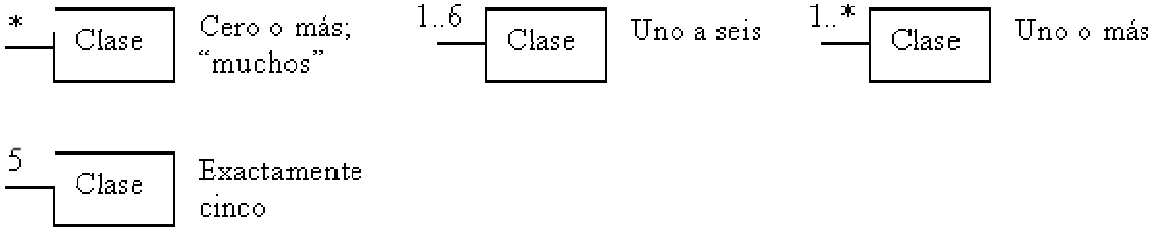
Por otro lado, de la definición del problema, el establecimiento de actividades y requerimientos, en el caso del ESIA y de otras empresas a través de la metodología del Proceso Unificado, se abre la posibilidad para incrementar la utilidad y la eficiencia del sistema, como pueden ser la incorporación de lectores de códigos de barras en credenciales o realizar consultas de información desde Internet a la base de datos.

APÉNDICE

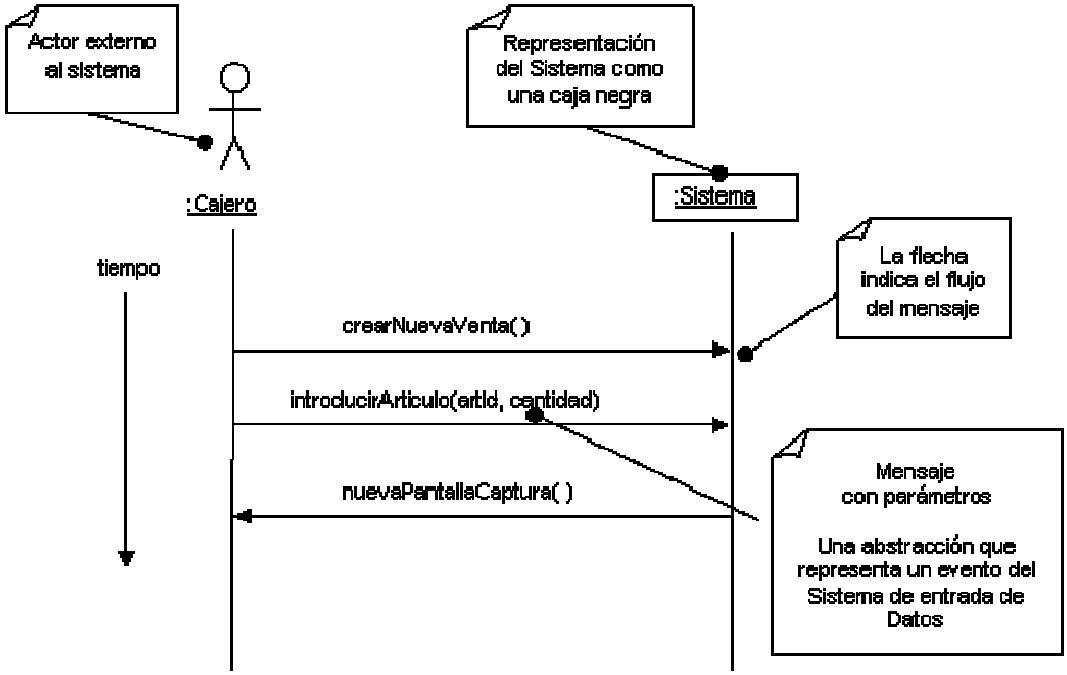
Clases: Se pueden representar con detalle dividido en tres secciones o de forma simple como una sola caja.



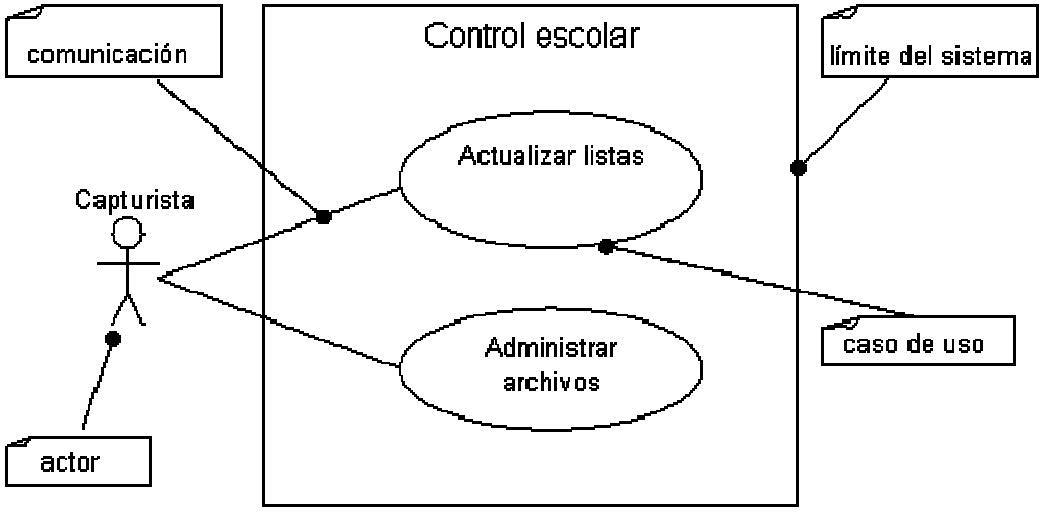




Diagramas de secuencia



Casos de uso



GLOSARIO

Actor conjunto coherente de roles que juegan los usuarios de los casos de uso cuando interactúan con éstos.

Arquitectura conjunto de decisiones significativas acerca de a organización de un sistema software, la selección de los elementos estructurales (y sus interfaces) de los que se compone el sistema, junto con su comportamiento tal y como se especifica en las colaboraciones entre esos elementos, la composición de esos elementos estructurarles y de comportamiento en subsistemas cada vez mayores y el estilo arquitectónico que oriente esta organización (esos elementos y sus interfaces, sus colaboraciones y su composición). La arquitectura del software no sólo tiene que ver con la estructura y el comportamiento, sino también con las restricciones y los compromisos entre uso, funcionalidad, rendimiento, flexibilidad, reutilización comprensibilidad, economía y tecnología, y con intereses estéticos.

Artefacto pieza de información que es utilizada o producida por un proceso de desarrollo de software.

Asociación relación estructural que describe un conjunto de enlaces, donde un enlace es una conexión entre objetos: relación semántica entre dos o más clasificadores que implica la conexión entre sus instancias.

Atributo una característica o propiedad de una clase a la que se le asigna un nombre.

Cardinalidad numero de elementos de un conjunto.

Caso de uso descripción de un conjunto de secuencias de acciones, incluyendo variantes, que ejecuta un sistema para producir un resultado observable, de valor para un actor.

Clase descripción de un conjunto de objetos que comparte los mismos atributos, operaciones, relaciones y comportamiento.

Construcción tercera fase del ciclo de vida del desarrollo de software, en la cual el software se plantea desde una base arquitectónica ejecutable básica.

Diagrama representación gráfica de un conjunto de elementos, representado la mayoría de las veces como un grafo conexo de nodos (elementos) y arcos (relaciones).

Diagrama de casos de uso diagrama que muestra un conjunto de casos de uso, actores y sus relaciones.

Diagrama de clases diagrama que muestra un conjunto de clases, interfaces, colaboraciones y sus relaciones; los diagramas de clases cubren la vista estática del diseño de un sistema; diagrama que muestra una colección de elementos declarativos (estáticos)

Diagrama de interacción diagrama que muestra una interacción que consta de un conjunto de objetos y sus relaciones, incluyendo los mensajes que pueden enviarse entre ellos; los diagramas de interacción cubren la vista dinámica de un sistema.

Diagrama de secuencia diagrama de interacción que resalta la ordenación cronológica de los mensajes.

Dominio una delimitación formal de una área de conocimiento o actividad que se caracteriza por un conjunto de conceptos y una terminología específica.

Elaboración segunda fase del ciclo de vida del desarrollo del software, en la cual se definen la visión del sistema y su arquitectura.

Fase intervalo de tiempo entre dos hitos impórtate del proceso de desarrolló durante el cual se satisface un conjunto bien definido de objetivo, se completan artefactos se toman las decisiones sobre si pasar a la siguiente fase.

Framework patrón arquitectónico que proporciona una plantilla de elementos como diagramas y definiciones catalogadas por el dominio en que son aplicables.

Implementación realización concreta del contrato declarado por una interfaz; definición de cómo se construye o se computa algo.

Incremental en el contexto del ciclo de vida del desarrollo del software, proceso que implica la continua integración de la arquitectura del sistema para producir versiones, donde cada nueva versión incluye mejoras incrementales sobre las otras.

Ingeniería directa proceso de transformar un mole o en código a través de una correspondencia con un lenguaje de implementación específico.

Ingeniería inversa proceso de transformación de código en un modelo a través de una correspondencia con un lenguaje de implementación específico.

Inicio primera fase del ciclo de vida del desarrollo, en el cual se lleva la idea inicial del desarrollo hasta el punto de estar suficientemente bien fundada para justificar la entrada en la fase de elaboración.

Interacción comportamiento que comprende un conjunto de mensajes que se intercambian entre un conjunto de objetos, dentro de un contexto particular, para lograr un objetivo.

Interfaz colección de operaciones que se utiliza para especificar un servicio de una clase o un componente.

Iteración conjunto bien definido de actividades, con un plan base y unos criterios de evaluación que produce una versión, ya se interna o externa.

Iterativo en el contexto del ciclo de vida del desarrollo, proceso que implica la gestión de un flujo de versiones ejecutables.

Mensaje especificación de una comunicación entre objetos que transmite información con la expectativa de que se desencadenará activada; la recepción de la instancia de un mensaje se considera normalmente una instancia de un evento.

Método implementación específica o algoritmo de una operación para una clase. Informalmente, el procedimiento software.

Modelo simplificación de la realidad, creada para comprender mejor el sistema que se está creando. Una descripción de las características estáticas y/o dinámicas de un área de estudio, descritas mediante una serie de vistas (normalmente diagramas o texto).

Multiplicidad el número de objetos que se permite que participen en una asociación. Especificación del rango de cardinalidad.

Objeto manifestación concreta de una abstracción; entidad con unos límites bien definidos e identidad que encapsula estado y comportamiento; instancia de una clase.

Operación implementación de un servicio que puede ser requerido a cualquier objeto de la clase para que muestre un comportamiento.

Parámetro especificación de una variable que puede cambiarse, pasarse o ser devuelta.

Proceso flujo de control que puede ejecutarse concurrentemente con otros procesos.

Refinamiento relaciones que representa una especificación más completa de algo que ya ha sido especificado a cierto nivel de detalle.

Relación conexión semántica entre elementos.

Requisito característica, propiedad o comportamiento deseado de un sistema.

Rol comportamiento de un a entidad que participa en un contexto particular.

Sistema conjunto de elementos organizados para lograr un propósito específico y que se describe por un conjunto de modelos con su arquitectura y funcionamiento.

Transición cuarta fase del ciclo de vida del desarrollo del software, en la que el software se pone en las manos de la comunidad de usuarios.

UML Lenguaje Unificado de Modelado, un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema con gran cantidad de software.

BIBLIOGRAFIA

Booch, G., Rumbaugh, J. y Jacobson, I. (2000). El proceso unificado de desarrollo de software. *Addison Wesley Iberoamericana*.

Booch, G., Rumbaugh, J. y Jacobson, I. (1999). El lenguaje unificado de modelado. *Addison Wesley Iberoamericana*.

Larman, C. (2003). UML Y Patrones. Una introducción al análisis y diseño orientado a objetos y al proceso unificado. *Prentice Hall*.

Rumbaugh, Blaha, Premerlani y Lorensen. (1991). Modelado y diseño orientado a objetos. Metodología OMT. *Prentice Hall*.

Schach, S. (2005). Análisis y diseño orientado a objetos con UML y el Proceso Unificado. *Mc Graw Hill*.

Senn, J. A. (1998). Análisis y diseño de sistemas de información. *Mc Graw Hill*.

REFERENCIAS DIGITALES

Objects by Design. (2005, Octubre). UML Products By Company. Extraído el 15 de Abril del 2006 desde http://www.objectsbydesign.com/tools/umltools_byCompany.html

OMG, (2005, Enero). Unified Modeling Language Specification. Extraído el 7 de Febrero de 2007 desde http://www.omg.org/docs/formal/05-04-01.pdf

OOSE, (s.f.). UML Tools. Extraído el 15 de Abril de 2007 desde http://www.oose.de/umltools.htm

Telelogic. (s.f.). System Architect Data Model. Extraído el 10 de Marzo de 2006 desde http://download.telelogic.com/download/article/Data_Sheet_SA_Data_Modeling_Sept05.pdf