Universidad de La Habana
Facultad de Matemática y Computación

Herramienta para la migración de datos del Sistema de Inventario Participativo al ASSETS.

Trabajo de Diploma
en opción al título de Licenciado en Ciencias de la Computación

Autor: Meisbel Daudinot López

Tutores: Dr. Pascual Verdecia Vicet
Ing. Alexei Zubizarreta Pérez

Ciudad de La Habana, Junio del 2004
"Año del 45 aniversario del Triunfo de la Revolución."

Resumen

En este trabajo se estudia la migración de datos, para cumplir los nuevos requisitos de un sistema de información para el control del inventario de la UCI, el ASSETS, que ha de dar surgimiento al Sistema de Inventario Participativo para satisfacer las necesidades del entorno, pero que no va a dejar de existir en los momentos actuales por su utilidad en otras actividades económicas de la escuela, manteniendo los datos generados hasta el momento en la nueva aplicación.

En la actualidad, son los Sistemas Gestores de Base de Datos (SGBD) los que dan un soporte para evolución de los datos, porque éstos son las que almacenan la información que se generan con las aplicaciones

Se propone una herramienta con la cual se automatizará la migración de datos al ASSETS, eliminando de esta forma, las actividades manuales e intelectuales asociadas a esta tarea, así como la posibilidad de insertar información errónea en la base de datos, la cual integrada al Sistema de Inventario Participativo, contribuirá a elevar la eficiencia y la eficacia de la información del estado del patrimonio de la universidad.

Introducción

El hombre desde tiempos memorables se ha empecinado en llevar un control exhaustivo de todos los movimientos de recursos que se ejecutan en sus pequeñas, medianas o grandes empresas. Por consiguiente, se ha apoyado en diversas formas para lograr su fin. En un principio, lo realizó en procesos muy simples, sin embargo con el transcurrir del tiempo, el avance tecnológico y las exigencias empresariales los procesos y técnicas contables han evolucionado.

La economía del país está en un proceso de cambio, y se hace cada vez más imprescindible contar con la información en formato electrónico de toda la parte contable de cualquier empresa, para de esta manera facilitar el trabajo de la esfera, brindando con el mismo nuevas facilidades a los usuarios.

En la UCI, como en cualquier lugar en Cuba, se trabaja actualmente con un software llamado Sistema de Gestión Integral (ASSETS). El mismo cuenta con módulos que trabajan en conjunto, es decir, uno depende del otro; como todo sistema contable tiene sus restricciones de acceso.

El problema existente en el ASSETS, es que no brinda toda la información que los usuarios de la universidad desearían que tuviera, vista en reportes para su posterior análisis y hacer estimaciones de las inversiones en los medios adquiridos.

Contando con esto y con las posibilidades de conectividad con que cuenta la escuela, surge la idea de que además el inventario puede hacerse participativo, es decir que cualquier persona pueda realizar un inventario de los medios que tiene a cargo desde cualquier punto de la escuela. Por otra parte que cuando se realice el inventario se haga de manera eficiente y rápida. De ahí surge la necesidad de un nuevo software para la realización del inventario. El ASSETS continuaría siendo explotado, pues con él también se realizan otras operaciones contables que no son de interés en la

actividad del inventario, por lo que será necesario migrar datos del nuevo software a él.

Es un hecho destacado por numerosos investigadores y profesionales que los sistemas informáticos deben evolucionar para adecuarse a los siempre cambiantes requisitos del entorno. Las estadísticas indican que entre el 65% y el 75% de las fuerzas vivas relacionadas con el mundo de la informática y aproximadamente el 80% de los gastos totales del software se dedican al mantenimiento del software existente. [3]

Idealmente, a la hora de introducir una modificación en una aplicación informática se debe seguir un proceso en el cual sea identificada la razón del cambio, se analice cómo afecta el cambio a introducir en la aplicación, para posteriormente realizar los cambios allí donde sean necesarios, finalizando con una exhaustiva fase de pruebas que demuestre que la modificación no ha introducido errores. Todo el proceso anterior debe estar bajo un estricto control de cambios que recoja todos los documentos que se generan y almacene las diferentes versiones.

Lamentablemente, muchas empresas simplifican el proceso anterior reduciéndolo, en la mayoría de los casos, a un análisis de la modificación para posteriormente aplicarla directamente al código sin excesivas pruebas y menor documentación.

En la actualidad, numerosas herramientas CASE son capaces de generar todo o parte del código que implementa una aplicación a partir de la información que se introduce en los modelos del método que se esté utilizando. Así por ejemplo, herramientas como Rational Rose, Together o System Architect, son capaces de generar esqueletos de programas en lenguajes de programación como Java, C++, VBasic y scripts en SQL para generar las bases de datos necesarias a partir de la información estructural que se introduce mediante dichas herramientas en los diagramas de clases.[9]

Otras herramientas CASE como OO-Method/CASE u Oblog/CASE, son capaces de generar aplicaciones completas a partir de la información de modelado que se introduce en sus modelos gracias a su sólido soporte formal.

Todas las herramientas CASE anteriores no ofrecen un buen soporte al paso del tiempo. La introducción de un nuevo requisito en las aplicaciones en funcionamiento, siguiendo algún método de gestión de cambios, consiste en introducirlo en los modelos pertinentes para posteriormente, aprovechando las capacidades generadoras de las herramientas CASE, regenerar la aplicación y el esquema de la base de datos acorde con el modelo del sistema actualizado. Finalmente, existen dos esquemas conceptuales del sistema de información, uno inicial con la definición del sistema antes de introducir los nuevos requisitos y uno final en el que ya se han introducido cambios. Además, existen dos bases de datos, una inicial con toda la información que se ha generado mientras la aplicación ha estado en funcionamiento, y una final que satisface los nuevos requisitos del sistema, sin información. El problema es ¿cómo trasvasar la información de la base de datos inicial a la final?

En la actualidad, son los Sistemas Gestores de Base de Datos (SGBD) los que dan un soporte para evolución de los datos, porque éstos son las que almacenan la información que se generan con las aplicaciones.

Dichos SGBD permiten, bien de manera interactiva manipulando directamente la estructura de la base de datos a través de una interfaz gráfica de usuario, bien mediante la ejecución de scripts en los que previamente se debe codificar la modificación a realizar, modificar los esquemas de las bases de datos, siendo los administradores de las bases de datos los responsables últimos de que las modificaciones que se realicen sean correctas. Pero no ofrecen un claro soporte a la migración de los datos, siendo un problema frecuente y absurdamente desatendido por la Ingeniería del Software.

Este trabajo está dirigido a resolver el problema de desarrollar una herramienta informática que de soporte a la migración automática de datos del nuevo sistema de inventario al ASSETS.

Por lo anterior, el objetivo general del presente trabajo de tesis es: *concebir, analizar y diseñar una herramienta que permita migrar datos del Sistema de Inventario Participativo al ASSETS.*

Con la herramienta que se propone, se logrará automatizar la migración de datos al ASSETS, eliminando de esta forma, las actividades manuales e intelectuales asociadas a esta tarea, así la posibilidad de insertar información errónea en la base de datos, la cual, integrada al Sistema de Inventario Participativo, contribuirá a elevar la eficiencia y la eficacia de la información del estado del patrimonio de la universidad. La herramienta contará con dos partes fundamentales que consisten en configurar y ejecutar la migración.

Para cumplir con el objetivo propuesto es necesario realizar, en primer lugar, un estudio amplio para elegir y aplicar una metodología de análisis y diseño de sistemas informáticos.

Con el presente trabajo, se pretende obtener un producto cubano, que sirva para migrar información no solo hacia el ASSETS, sino que pueda ser utilizado para realizar la migración entre bases de datos independientes.

El presente documento está estructurado de la siguiente manera:

- Capítulo 1. Fundamentación Teórica: recoge el análisis de la información existente acerca del tema a tratar y las tendencias actuales que existen en el mundo. También incluye como aspectos de actualidad una descripción del lenguaje de programación a utilizar para la implementación así como del lenguaje de modelación usado.

- Capítulo 2. Característica de la herramienta: describe el objeto de estudio, el entorno de trabajo en que se desarrolla la herramienta, se especifican los detalles de la propuesta, los requerimientos funcionales y los casos de uso del sistema.
- Capítulo 3. Análisis y Diseño de la herramienta: documenta las etapas de análisis y diseño, mediante la expansión de los casos de uso, el modelo conceptual, los diagramas de interacción y el diseño de la base de datos.

Capítulo 1. Fundamentación teórica

Introducción

En este capítulo, se hará un estudio de algunos conceptos necesarios para una buena comprensión de este trabajo, que hemos escogido para llevar a cabo la herramienta que se pretende desarrollar.

1.1 El inventario en la UCI.

Antes de analizar como se realiza el inventario en la UCI veamos primeramente qué cosa es un inventario.

La actividad de inventario es la más importante visión de los activos que debe tener un centro o institución. La verificación de su patrimonio es indispensable para un adecuado manejo contable y económico-financiero.

Aunque el vocablo en nuestro idioma significa lo mismo que un balance general, tratamos el inventario desde el punto de vista del conteo que hacemos de las mercancías en existencia, siendo este el sentido en el que generalmente se usa en Cuba. Un inventario es además una relación de los activos circulantes, que posee la entidad en un momento dado y que pueden estar destinados para ser vendidos o para ser insumidos en el proceso productivo.

El inventario tiene como objetivo facilitar la información adecuada para el cumplimiento de los fines para los que se realiza, es decir:

- Identificación de los bienes para los que integran el inmovilizado material e inmaterial, sirviendo de soporte a la contabilidad.
- Conocer la situación geográfica y centro de gasto al que pertenece.
- Conocer el valor económico y las amortizaciones aplicadas.

- Aportar datos necesarios para el desenvolvimiento de la contabilidad analítica.
- Facilitar a los órganos directivos y de gestión un conocimiento exacto de todos los bienes de que dispone la institución.

La información que debe recoger el inventario hace referencia a los datos relacionados con las características físicas, técnicas, económicas y jurídicas, que a través de su permanente actualización, permitan satisfacer los fines anteriores. Desde el punto de vista económico del bien, se recogerá su valoración, ya que esta incide directamente en el cálculo del patrimonio de la universidad, teniendo en cuenta la obsolescencia y depreciación que pudieran afectar a su valor.

Muchos son los sistemas que existen el mundo para realizar el inventario, con el objetivo de controlar lo que se tiene en la institución que se refiera; lo que diferencia a un sistema de otro es la forma en que lo hacen.

En la UCI, como en cualquier lugar en Cuba, se trabaja actualmente con un software llamado Sistema de Control Integral (ASSETS), que no es más que un sistema integral multiusuario concebido para el control de la actividad empresarial. Permite realizar, controlar y contabilizar todas las transacciones comunes relacionadas con el proceso de compra-venta e incluye, asimismo, los procedimientos necesarios para registrar los movimientos de los activos fijos y de los útiles y herramientas. Este sistema estándar y parametrizado, permite el control del inventario continuo de múltiples almacenes, ofreciendo un control estricto de las existencias, reservas y disponibilidad de productos, genera automáticamente los comprobantes contables de las operaciones, así como actualiza y controla las cuentas por cobrar y pagar. Admite la obtención de los estados financieros, realiza conciliaciones bancarias y consulta los saldos de sus cuentas contables. Dispone además, de una amplia gama de análisis y consultas que le proporcionarán no sólo conocer exactamente la situación actual, sino, además, proyectar el rumbo que tomará su empresa.

El ASSETS es un sistema flexible, de fácil uso, con ayuda en línea, que puede ser instalado en una microcomputadora o sobre varias. Proporciona opciones de

seguridad que le permiten limitar el acceso a los diferentes procesos del sistema de acuerdo con el perfil de cada usuario.

Pese a esto el ASSETS no brinda toda la información que los usuarios desearían que tuviera, vista en reportes para su posterior análisis y hacer estimaciones de las inversiones en los medios adquiridos.

1.2 La seguridad de los datos

Los datos son uno de los principales activos de todas las empresas, y prácticamente en todos los proyectos de sistemas de información existe una importante fase de migración o integración de datos.

Ya sea una migración tradicional entre bases de datos, o un novedoso desarrollo de una aplicación e-commerce, siempre habrá un factor común: el movimiento de información. Si el usuario no cuenta con una información correcta en el momento en que la necesita, el proyecto estará abocado al fracaso, por muy compleja que sea su funcionalidad.

No entender claramente la validez o no de los datos de los sistemas de información puede conducir a la toma de decisiones equivocadas. Los datos malos o erróneos cuestan dinero, reducen la productividad y finalmente conducen a una baja satisfacción de los clientes. Las decisiones estratégicas basadas en información poco confiable conducen inevitablemente a decisiones equivocadas.

La calidad de datos es una preocupación cada vez más prominente en la agenda de los directores de informática y de los gestores de grado superior. De hecho, ¿para qué sirven las grandes inversiones en tecnologías de la información, si los datos que éstas albergan no tienen calidad?

Este problema es particularmente agudo cuando las migraciones de datos entre sistemas (por ejemplo cuando un nuevo sistema viene a sustituir uno antiguo y se observa que muchos de los datos no obedecen a las reglas definidas para la

migración) o en la canalización de los datos para Data Warehouses, donde las conclusiones retiradas sobre las agregaciones de datos errados acaban por estar también erradas, sin que sus gestores nunca, o demasiado tarde, se den cuenta.

1.3 Solución aproximada para realizar un proyecto de migración.

Muchas empresas se enfrentan a la necesidad de adaptar sus aplicaciones informáticas, críticas para su negocio, al funcionamiento que imponen las nuevas tecnologías y requerimientos del mercado (como Internet, Web-services...)

Hay dos formas de solventar esta situación [8]:

- Mediante las técnicas de integración de sistemas heterogéneos (EAI/HIS).
- Migrando las aplicaciones al nuevo entorno, lenguaje y SGDBR.

1.3.1 Técnicas de integración de sistemas heterogéneos

En las organizaciones, tanto públicas como privadas, se disponen de múltiples sistemas heterogéneos de carácter departamental que dan servicio a diferentes áreas de gestión y que conviven con sistemas corporativos de tipo horizontal. Estos sistemas tarde o temprano han de integrarse para compartir información entre sí, facilitando que determinadas fuentes de información en una misma organización sean únicas, y manteniendo la integridad y seguridad del sistema de información corporativo. [15]

El problema de integración de sistemas de información consiste en proporcionar una visión global integrada de datos distribuidos almacenados en repositorios heterogéneos. Para esto deben combinarse diversos sistemas componentes en aparentemente un solo sistema de información. Sin embargo la mayoría de los componentes no fueron diseñados tomando en cuenta la integración, por lo que hay que superar diversos tipos de heterogeneidad (hardware y sistemas operativos, DBMS's, modelos de datos, semántica de los datos, interfaces de usuario, etc.).

La integración de sistemas de información puede realizarse a diferentes niveles:

- Integración manual: los usuarios deben realizar todo el trabajo por lo que deben familiarizarse con diferentes interfaces de usuario, lenguajes de consulta y semántica de los datos.
- Interfaz de usuario común: se provee una interfaz de usuario común entre los diferentes SI (por ejemplo un navegador). Esto provee cierta uniformidad pero diferentes fuentes de datos se presentan típicamente en ventanas diferentes.
- Integración por aplicaciones: las aplicaciones acceden a diversas fuentes de datos y regresan resultados integrados al usuario. Cada aplicación debe usar diferentes interfaces para acceder a los datos lo que resulta en aplicaciones complejas, sin embargo los usuarios no necesitan preocuparse por la localización u organización física de los datos.
- Integración por middleware: el uso de este permite definir interfaces estandarizadas para acceder a los datos así como transacciones para actualizaciones consistentes entre múltiples fuentes de datos.
- Acceso uniforme a los datos: la integración en el nivel de acceso a los datos brinda una integración lógica de los datos. Se provee una visión global unificada a las aplicaciones, pese a que sólo se dispone de datos virtuales en este nivel. Diversos enfoques pertenecen a esta técnica de integración y están relacionados con la integración de datos que se describe en la siguiente sección.
- Portal: provee rápidamente al usuario con la información en la que está interesado por medio de una sola interfaz (el navegador). Ofrece transparencia, personalización y búsqueda inteligente de información estructurada y no estructurada.
- Mediated query system: provee un solo punto de acceso a múltiples fuentes de datos, pero sólo acceso de lectura, además de que combinan datos de resultados y no sus instancias.
- SGBD Federado: proveen una integración lógica de los datos pese a que éstos todavía residen en fuentes locales. Un SGBD federado debe implementar las

mismas funcionalidades que cualquier SGBD. Si bien pueden parecer la solución óptima al problema de integración, surgen serias dificultades cuando se requiere de funciones de escritura sobre los datos.

Los sistemas se clasifican según la estrategia de materialización de datos que adoptan. Desde este punto de vista los datos son materializados en extensión o en intensión, es decir, sistemas materializados, sistemas virtualmente integrados. Los sistemas materializados implican la migración de los datos de diversas fuentes en una base de datos y las aplicaciones que los utilizaban deben migrarse también. Generalmente es demasiado costosa.

Las soluciones del tipo EAI/HIS son rápidas de implementar y no precisan de la modificación de las aplicaciones heredadas; pero no constituyen una solución al problema planteado [8].

1.3.2 Migración

Un proyecto de migración es por su importancia en muchos casos "crítico", tanto por la relevancia de los entornos migrados (datos y aplicaciones), que deberán ofrecer finalmente la misma eficiencia y operatividad que daban en el entorno anterior (con independencia del cambio en Base de Datos y/o Lenguaje de Programación); así como por otras cuestiones que forman parte ineludible del global proyecto y que han de tenerse en cuenta, como por ejemplo: cambio de cultura tecnológica producido, entrenamiento necesario de nuestro personal técnico y usuarios en lo que resulte imprescindible, etc.; en definitiva, esfuerzo de adaptabilidad a las nuevas posibilidades y entendimiento "interno" de las potencialidades que ofrecerá la realización del mismo [15].

El proceso de la migración de datos puede ser bastante complejo y, como hay tantas bases de datos distintas, resulta difícil dar una receta que funcione en todos los casos. Además, aparte de la dificultad de transferir la información entre los dos sistemas gestores de base de datos, también nos influirá mucho en la complejidad del problema el tipo de los datos de las tablas que estamos utilizando. Por ejemplo, las fechas, los

campos numéricos con decimales o los booleanos pueden dar problemas al pasar de un sistema a otro porque pueden almacenarse de maneras distintas o, en el caso de los números, con una precisión distinta.

Toda la migración tiene que tener en cuenta muy especialmente, como ya se señaló, las maneras que tenga cada base de datos de guardar la información, es decir, del formato de sus tipos de datos. Tenemos que contar siempre con la posible necesidad de transformar algunos datos como pueden ser los campos booleanos, fechas, campos memo (texto con longitud indeterminada), etc., que pueden almacenarse de maneras distintas en cada uno de los sistemas gestores, origen y destino.

Una migración requiere de tres conceptos: una metodología, un conjunto de herramientas y unas técnicas de pruebas y personalización.

- La metodología nos garantizará en primer lugar que disponemos de un repositorio con toda la información necesaria para abordar la migración: programas, estructuras de datos (en ficheros o en bases de datos), librerías de funciones, etc., en segundo lugar, contempla la obtención del modelo de negocio desde y al que migrar partiendo de la información contenida en el repositorio. Además, define las reglas de generación del código a migrar, conforme a los estándares establecidos, las librerías de funciones usadas por la empresa, y cualquier otra consideración necesaria. Por último y además, contempla la realización de los planes de prueba de las aplicaciones migradas.
- Las herramientas de migración, capaces de obtener automáticamente un modelo del negocio a migrar en un formato descriptivo estándar, por ejemplo XML. De este modo se independiza el formato de la información de modelo de negocio, del lenguaje de programación; con lo cual el modelo obtenido sería válido en caso de requerir futuras migraciones a otras tecnologías.
- Las técnicas de pruebas y personalización que incorporan las reglas de generación introducidas por la metodología, para así obtener aplicaciones fiables, funcionales y operativas, que optimizan su funcionamiento en el entorno informático existente en la empresa.

Más allá de la complejidad tecnológica de un proyecto de migración, existen otras cuestiones sobre las que interesa prestar mucha atención y dada la importancia que al final tiene, el que ningún elemento quede fuera de nuestras previsiones y capacidades de ejecución.

1.4 Tecnologías y metodologías utilizadas para el desarrollo de la herramienta.

1.4.1 Plataforma .NET y lenguaje C#

La plataforma .NET es un nuevo entorno de programación especialmente diseñado para la creación de aplicaciones y de servicios web. Podría decirse que supone un cambio tan grande en el modo de programar como en su día lo fue la transición desde MS-DOS a Windows. Ahora el programador no trabaja directamente contra un sistema operativo concreto como Windows, sino que lo hace frente a una máquina virtual (el CLR o Common Language Runtime) que le ofrece los servicios que ante le proporcionaba el sistema operativo de forma más simplificada y adecuada a los tiempos actuales [5].

La mejor forma de resumir de las características de la plataforma .NET es enumerar los servicios que proporciona el CLR a todas las aplicaciones que desarrolladas para la misma. Entre éstas destacan las siguientes:

- ✓ Sencillo modelo de programación
- ✓ Tratamiento homogéneo de errores mediante excepciones
- ✓ Desarrollo interlenguaje
- ✓ Ejecución multiplataforma
- ✓ Gestión automática de memoria con recolección de basura
- ✓ Aislamiento de procesos
- ✓ Soporte multihilo
- ✓ Seguridad avanzada basada en el usuario y la procedencia del código
- ✓ Interoperabilidad con código antiguo

Una de las principales y más novedosas características de la plataforma .NET, que acaba de señalarse es su orientación hacia el desarrollo interlenguaje. Esto significa que en un proyecto .NET puede escribirse cada clase en cualquiera de los diferentes lenguajes que se han adaptado para funcionar en .NET, y la integración entre las clases escritas en los diversos lenguajes se hará perfecta y transparentemente, siendo incluso posible definir en cualquier lenguaje clases derivadas de clases escritas en cualquiera otro lenguaje.

Se han adaptado a .NET la inmensa mayoría de lenguajes de programación existentes, como C++, JScript, Visual Basic, Java, Eiffel, Cobol, Perl, Python, Fortran, Pascal, Smalltalk, etc. Sin embargo, Microsoft también ha creado un nuevo lenguaje llamado C# especialmente recomendado para la programación de las aplicaciones .NET y al que se suele bautizar como el lenguaje estrella de .NET.

1.4.2 Metodología RUP

El Rational Unified Process (RUP) es un proceso de ingeniería de software que mejora la productividad del equipo de trabajo y entrega las mejores prácticas del software a todos los miembros del mismo. Los contenidos específicos para e-business del RUP proporcionan una guía específica en áreas tales como la de Modelación de Negocios, Arquitecturas Web, Pruebas y Calidad. También proporcionan lineamientos para desarrollar en plataformas IBM Websphere y Microsoft Solution con el fin de acelerar proyectos de desarrollo Web. RUP esta fuertemente integrado con las diferentes herramientas Rational, permitiéndole a los equipos de desarrollo alcanzar todos los beneficios de las características de los productos Rational, el Unified Modeling Language (UML), y otras mejores prácticas de la industria [2].

La creación sólida de software de calidad requiere el conocimiento específico de las tareas que deben llevarse a cabo en cada entorno. Ahí radica la importancia de aplicar un proceso de desarrollo flexible y adaptado a cada objetivo de desarrollo. El proceso RUP combina un conjunto básico de mejores prácticas aprobadas por el

sector con una serie de complementos opcionales del proceso a fin de dar cabida y soporte a proyectos de cualquier envergadura o alcance.

Cualquier tipo de proyecto (incluidos los pequeños, los basados en Web, aquéllos fundamentales para un proyecto y los proyectos integrados) permiten obtener unos resultados más acordes con las previsiones gracias a la aplicación del proceso RUP™

1.4.3 Lenguaje de modelación UML

Unified Modeling Language (UML) es una especificación de notación orientada a objeto, resultado de la consolidación de muchas de las notaciones y conceptos más usados orientados a objetos. Divide cada proyecto en un número de diagramas que representan las diferentes vistas del proyecto. Estos diagramas juntos son los que representa la arquitectura del proyecto [11].

Tiene diagramas que dan una visión dinámica del sistema.

También intenta solucionar el problema de propiedad de código que se da con los desarrolladores, al implementar un lenguaje de modelado común para todos los desarrollos se crea una documentación también común, que cualquier desarrollador con conocimientos de UML será capaz de entender, independientemente del lenguaje utilizado para el desarrollo.

Su utilización es independiente del lenguaje de programación y de las características de los proyectos, ya que UML ha sido diseñado para modelar cualquier tipo de proyectos, tanto informáticos como de arquitectura, o de cualquier otro ramo. Permite la modificación de todos sus miembros mediante estereotipos y restricciones. Un estereotipo nos permite indicar especificaciones del lenguaje al que se refiere el diagrama de UML. Una restricción identifica un comportamiento forzado de una clase o relación, es decir mediante la restricción estamos forzando el comportamiento que debe tener el objeto al que se le aplica.

Conclusiones

Una incorrecta migración de información entre el Sistema de Inventario Participativo y el ASSETS implica inseguridad y descontrol del patrimonio de la escuela, implicando por tanto, dificultad en la toma de decisiones económica en la misma.

Por tales motivos, la creación de una herramienta para la migración, adaptado para los gestores de bases de datos ORACLE y SQL Server que son los gestores de base de datos para las aplicaciones Sistema de Inventario Participativo y ASSETS respectivamente, constituye una necesidad para las actuales tendencias en el uso de las nuevas tecnologías y facilitar el trabajo de la universidad en este esfera.

Capítulo 2 Característica de la herramienta

Introducción

El presente capítulo está dedicado al estudio preliminar de la herramienta para la migración de datos del Sistema de Inventario Participativo al ASSETS. En esta etapa, de acuerdo con la metodología de desarrollo seleccionada, se identifican y analizan las necesidades del usuario y se efectúa una caracterización del sistema propuesto.

2.1 Objeto de estudio

La continua evolución de las necesidades de los sistemas de información provoca cambios en las estructuras de las bases de datos utilizadas para gestionar la permanencia y renovación de la información. La evolución que sufren estas estructuras hace necesario trasvasar la información contenida en la base de datos inicial a la base de datos evolucionada. El hecho de que los sistemas de información pueden llegar a contener una gran cantidad de datos que deberemos trasladar, y que las tecnologías utilizadas para la gestión de los mismos pueden también evolucionar o cambiar, dificulta el proceso de migración, siendo éste un proceso clave en la evolución de los sistemas de información.

En el caso específico de este trabajo, la base de dato del Sistema de Inventario Participativo recoge información que no es utilizada por el ASSET, por lo que tendrá un mayor número de información en sus tablas, por otra parte, habrán tablas que cambiarán: a unas se le añadirá información, a otras se le eliminará y otro grupo se le modificará. No se tiene un estándar para establecer una correspondencia entre las bases de datos, por lo que habría que realizar la correspondencia manualmente, cosa que implicaría que se tuviera gran posibilidad de cometer errores. Para ello es necesario, primeramente, realizar un análisis detallado de las bases de datos para

detectar qué cambios son los que deben realizarse para que la migración sea óptima y fiable.

2.1.1 Problema

En la actualidad, la actividad económica de la UCI que es controlada y manipulada por el ASSETS, requiere de la información recogida en el Sistema de Inventario Participativo, por lo que sería óptimo realizar una migración de dicha información para evitar la que sea tecleada dos veces, y por tanto, que se comentan errores. Una vía para lograr mayor eficiencia en este proceso lo constituye la automatización. Sin embargo en la escuela no se cuenta con una herramienta que garantice dicha migración.

2.1.2 Restricciones

Dado que el sistema propuesto es una herramienta que se utilizará para pasar datos de una aplicación a otra es necesario crear una interfaz con ambas aplicaciones de manera que la información que maneja no se pierda con la nueva aplicación para realizar la actividad del inventario. Lo anterior, permitirá reutilizar gran parte de la información almacenada en el Sistema de Inventario Participativo al ser transferida a la base de datos del ASSETS. Esto reducirá los costos de implantación de un nuevo sistema, al considerar que los datos, resultado de un arduo trabajo, no se pierdan con la implantación del nuevo sistema para la realización de la actividad del inventario.

Para el desarrollo de la aplicación, se cuenta con el apoyo financiero de la institución. La adquisición de la tecnología necesaria para el desarrollo del sistema, los gastos en bibliografía actualizada y otros gastos, se verán limitados a la gestión financiera del centro, el cual no se dedica a la actividad comercial, sino a la formación de recursos humanos.

2.1.3 Propuesta

Teniendo en cuenta lo antes mencionado se propone crear una herramienta que de forma automática o semiautomática ejecute la migración de datos. Esta herramienta permitirá que el usuario escoja manualmente la correspondencia entre las tablas y los atributos de estas en las bases de datos, también va a proponer una correspondencia para facilitarle el trabajo al usuario. Constará de dos etapas fundamentales: configuración de la migración y ejecución de la migración, para ello se le deberá primeramente analizar mediante una comparación, cómo se va a hacer la migración, es decir establecer una correspondencia entre las tablas de las dos de bases de datos para analizar cuáles son lo datos de que realmente se van a migrar.

2.2 Requerimientos del sistema

A través de los requisitos funcionales, los cuales se describen a continuación, se puede expresar una especificación más detallada de las responsabilidades del sistema. Con ellos, se pretende determinar de manera clara y concisa lo que debe hacer el sistema siguiendo un enfoque funcional.

2.2.1 Requisitos funcionales

1. Conectarse a las bases de datos origen y destino.
2. Cargar información de las bases de datos.
   2.1. Guardar la información de las bases de datos en la base de datos de la herramienta.
3. Mostrar información de las bases de datos origen y destino en forma de árbol.
   3.1. Mostrar información de las bases de datos.
      3.1.1. En los nodos padres del árbol se mostraran los nombres de las tablas.
      3.1.2. En los nodos hijos se mostraran los nombres de los atributos con la especificación del tipo y si son llave.
4. Relacionar tablas.
   4.1. Mostrar los nombre de las tablas de las bases de datos.

4.2. Escoger tablas a relacionar.

4.3. Adicionar relaciones entre tablas.

4.3.1. Cada relación consta de un nombre de una tabla origen y otro de una tabla destino.

4.4. Eliminar relación entre tablas.

4.4.1. Eliminar también las relaciones entre los atributos de las tablas relacionadas.

4.5. Salvar información sobre las relaciones entre las tablas.

5. Relacionar atributos.

5.1. Mostrar relaciones entre las tablas.

5.2. Proponer relaciones entre atributos.

5.2.1. Mostrar una propuesta de correspondencia entre los atributos.

5.2.1.1. Identificar los atributos insertados, modificados, borrados e invariantes de cada relación entre tablas.

5.3. Eliminar relaciones entre atributos.

5.4. Adicionar relación entre atributos.

5.4.1. Verificar que hay compatibilidad entre los tipos de los atributos que se quieren relacionar.

5.4.2. Cada relación entre atributos consta del nombre del atributo de una tabla origen, uno de la tabla destino y el tipo de datos.

5.5. Salvar información sobre las relaciones entre tablas.

6. Visualizar relaciones establecidas entre las tablas y los atributos de las bases datos origen y destino.

6.1. Visualizar para cada tabla relacionada los atributos.

7. Migrar datos.

7.1. Identificar que datos son los que se van a migrar.

7.2. Ejecutar sentencias SQL.

2.3 Definición de casos de uso de la herramienta

Los casos de uso son documentos narrativos que describen la secuencia de los eventos de un actor (agente externo) que utiliza un sistema para completar un proceso.

2.3.1 Descripción de los actores del sistema

Los actores representan a cualquier elemento que interactúa con el sistema que puede ser un humano, un software o hardware. Lo actores identificados se describen a continuación. (Tabla 2.1)

Actores	Justificación
Administrador	Es la persona responsable de la actividad de migración del ASSETS al Sistema de Inventario Participativo. Tiene acceso total a la base de datos del Sistema de Inventario Participativo, así como a la base de datos del ASSETS. Será el encargado de relacionar las tablas de las bases de datos del ASSETS y del Sistema de Inventario Participativo, así como de validar las propuestas de relaciones.
SGBD	Sistema gestor de base de datos (origen o destino) que proporciona los datos de entrada del sistema: información sobre las tablas que van a ser objeto de correspondencia.
SGBD Destino	Es un SGBD que recibe los beneficios de la migración. En la figura 2.1 se muestra la relación entre el SGBD y este actor.
SIP	Software que realiza el inventario de forma participativa. Es quien ejecuta la migración.

Tabla 2.1: Descripción de los actores

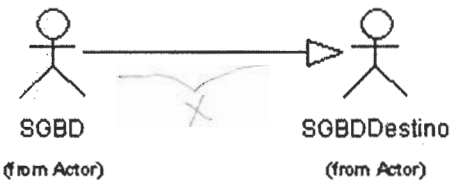

Figura 2.1: Relación entre los actores SGBD y SGBDDestino

2.3.2 Casos de usos

Los casos de uso han sido diseñados de manera que modelen la funcionalidad de sistema según lo perciben los actores. En la siguiente tabla se hace una descripción de los casos de usos. (Ver tabla 2.2)

No	Nombre	Actor(es)	Descripción	Referencia
CU-01	Conectar base de datos.	Administrador	Se establecen las conexiones entre las bases de datos, para acceder desde la herramienta.	1
CU-02	Cargar Información	Administrador, SGBD	Carga la información de las tablas (nombre) y los atributos (nombre, tipo y si es llave o no) de las bases de datos.	2,3
CU-03	Relacionar bases de datos	Administrador	Se establece una correspondencia entre las tablas y los atributos de las bases de datos.	4,5
CU-04	Proponer relaciones entre atributos	Administrador	Se propone una correspondencia entre los atributos de dos tablas previamente relacionadas para facilitarle el trabajo al usuario.	5
CU-05	Mostrar relaciones entre tablas	Administrador	Se visualiza como han quedado las relaciones entre las tablas y los atributos para verificar que la correspondencia establecida es correcta y que por tanto la migración también	4,5,6

CU-06	Modificar relaciones	Administrador	Si el usuario detecta que se ha equivocado y ha establecido una relación entre tablas o atributos de forma errónea, se modifican las correspondencias modificar esta relación. Esta modificación puede ser eliminar la relación.	6
CU-07	Migrar datos	SIP	Se ejecutar la migración de datos entre las dos bases de datos.	7

Tabla 2.2: Descripción de los casos de uso

2.3.3 Diagrama de casos de uso y paquetes.

Los paquetes de casos de uso son una forma de agrupar a los mismos teniendo en cuenta determinado criterio. Se representan a través de los diagramas que reflejan gráficamente la relación entre los actores y los casos de uso. Se definen tres paquetes: Configuración, Migración y Conexión. En el paquete Configuración se agrupan los casos de usos que se relacionan con las acciones que acometen los usuarios de la herramienta para configurar la migración, es decir establecer las correspondencias entre las tablas y los atributos de las bases de datos. En el paquete Migración están los casos de uso que representan las opciones para realizar la migración de datos. En el paquete Conexión se agrupan los casos de usos que se relacionan con las acciones para la conexión de las herramientas con las bases de datos. Para que se ejecute la Migración es necesario que antes se ejecute la Configuración, y para tener los datos para realizar la Configuración es necesario que antes se haya ejecutado la Conexión (.Ver figura 2.2).

En las figura 2.3, 2.4 y 2.5 se muestran los casos de usos de cada paquete.

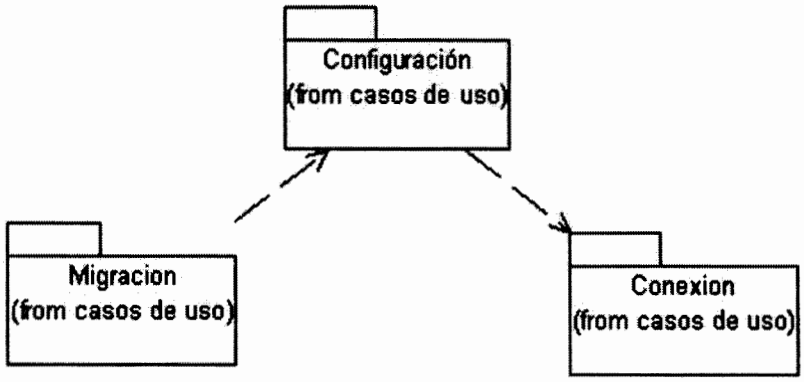

Figura 2.2 Relación de dependencia entre los paquetes de Configuración, Migración y Conexión

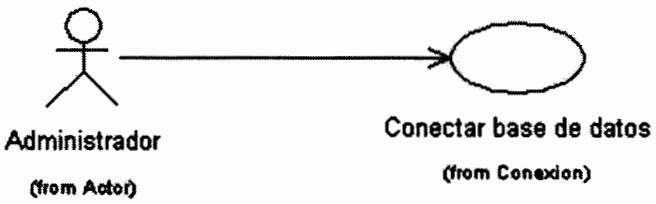

*Figura 2.3: Diagrama de casos de uso para el paquete **Conexión***

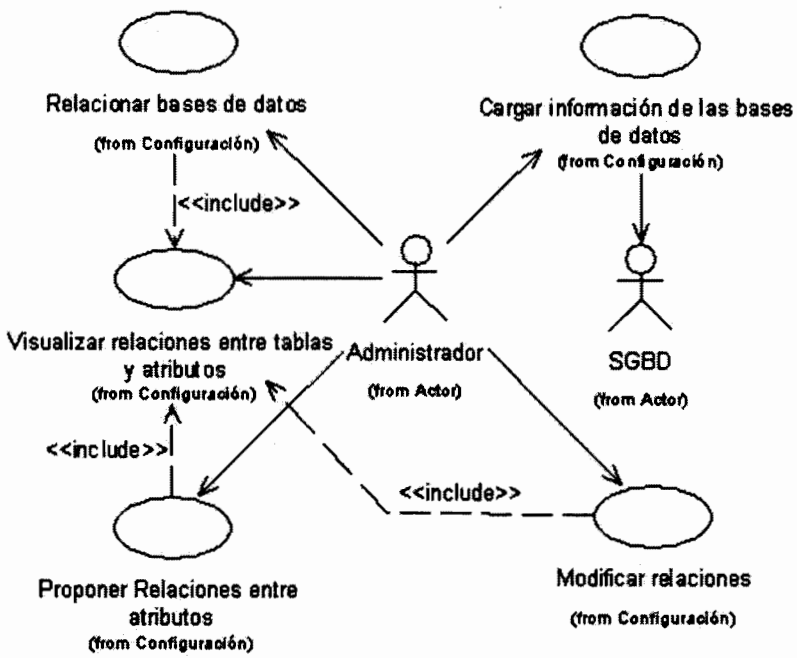

*Figura 2.3: Diagrama de casos de uso para el paquete **Configuración***

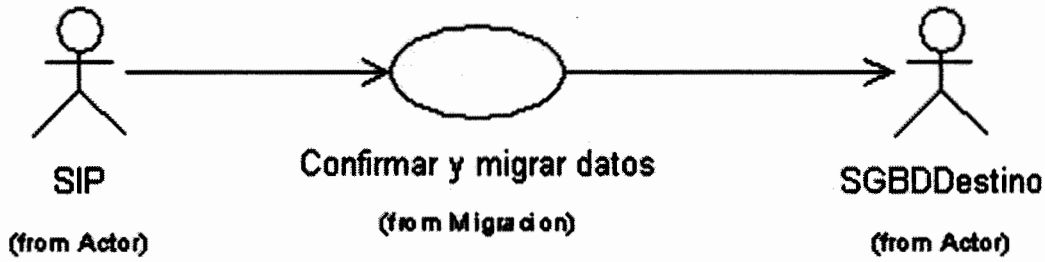

*Figura 2.4: Diagrama de casos de uso para el paquete **Migración***

2.3.4 Casos de uso y paquetes para cada ciclo

Teniendo en cuenta el desarrollo incremental e iterativo que propone UML, o sea que este se realice por ciclos, en cada uno de los cuales se agregue una funcionalidad adicional, se definieron dos ciclos de desarrollo para la elaboración de la herramienta. En la siguiente tabla se muestran los casos de usos de cada uno de estos ciclos. (Ver tabla 2.3)

Ciclo	Casos de Uso	Paquetes
(1)	▪ Conectarse a las bases de datos	Conexión
	• Cargar información de las bases de datos	Configuración
	• Relacionar bases de datos	
	• Visualizar relaciones	
	• Migrar datos	Migración
(2)	• Proponer relaciones	Configuración
	• Modificar relaciones	

Tabla 2.3: Casos de uso y paquetes para cada ciclo

Conclusiones

Al plantearse los problemas que dan surgimiento a la creación de un nuevo sistema para la realización del inventario en la UCI, se pone de manifiesto la necesidad de crear mecanismos que garanticen la interoperatividad e integración de los servicios del nuevo sistema y que posibilite un grado de personalización adecuado para acelerar los procesos de conocimiento, comunicación y toma de decisiones. La concesión de la Herramienta para la migración de datos del Sistema de Inventario Participativo al ASSETS es fundamental para ello.

Capítulo 3 Análisis y diseño de la herramienta

Introducción

El análisis forma parte del proceso de desarrollo de software, cuyo propósito primario es formular el modelo del dominio del problema.

El diseño es la etapa del proceso de desarrollo donde se decide cómo se llevará a cabo el sistema. A través de esta fase, se toman decisiones estratégicas y tácticas para cumplir los requerimientos funcionales y de calidad de un sistema.

En el presente capítulo se procede a representar la expansión de los casos de uso, el modelo conceptual, los diagramas de secuencia, así como se plasman los resultados de la etapa de diseño del sistema, utilizando UML.

Para ello se toma como referencia el caso de uso referente a la Relación de las bases de datos, teniendo en cuenta su importancia en la migración de datos. El proceso de análisis y diseño de los restantes casos de uso aparecen como anexos de este trabajo.

3.1 Expansión de los casos de uso

A través de la expansión de los casos de uso se describe paso a paso la secuencia de eventos que los actores utilizan para completar un proceso a través de la herramienta. El la siguiente tabla se expande le caso de uso Relacionar bases de datos. (Ver tabla 3.1.)

Caso de uso	
CU-05	Relacionar bases de datos
Propósito	Establecer una correspondencias entre
Actores Administrador	

Acción del actor	**Respuesta del sistema**
1. El caso de uso comienza cuando el administrador selecciona una de las tablas del SGBD origen y otra del SGBD destino y selecciona que desea relacionarlas.	
	2. El sistema inserta dichas tablas en la tabla de relaciones de tablas.
	3. El sistema muestra los atributos de las tablas relacionadas.
4. El administrador selecciona un atributo origen y uno destino e dos tablas relacionadas y selecciona que desea relacionarlas.	
	5. El sistema inserta dichos atributos en la tabla de relaciones de atributo.
6. El administrador selecciona que desea ver como fueron relacionadas las tablas.	
	7. El sistema muestra las balas relacionadas con la correspondencia entre los atributos

Tabla 3.1: Caso de usos expandido ***Relacionar base de datos***

En el Anexo I se muestra la expansión de cada de los demás casos de uso.

3.2 Modelo conceptual.

El modelo conceptual muestra los conceptos básicos del sistema y sus relaciones. Se realiza a través de un diagrama de clases de UML simplificado, en el cual se representan las clases preliminares, las asociaciones preliminares entre las clases, y los atributos de las clases. Una cualidad esencial que debe ofrecer un modelo conceptual es que representa cosas del mundo real, no componentes del software. En la figura 3.1 se muestra el modelo conceptual definido para la herramienta que se está desarrollando.

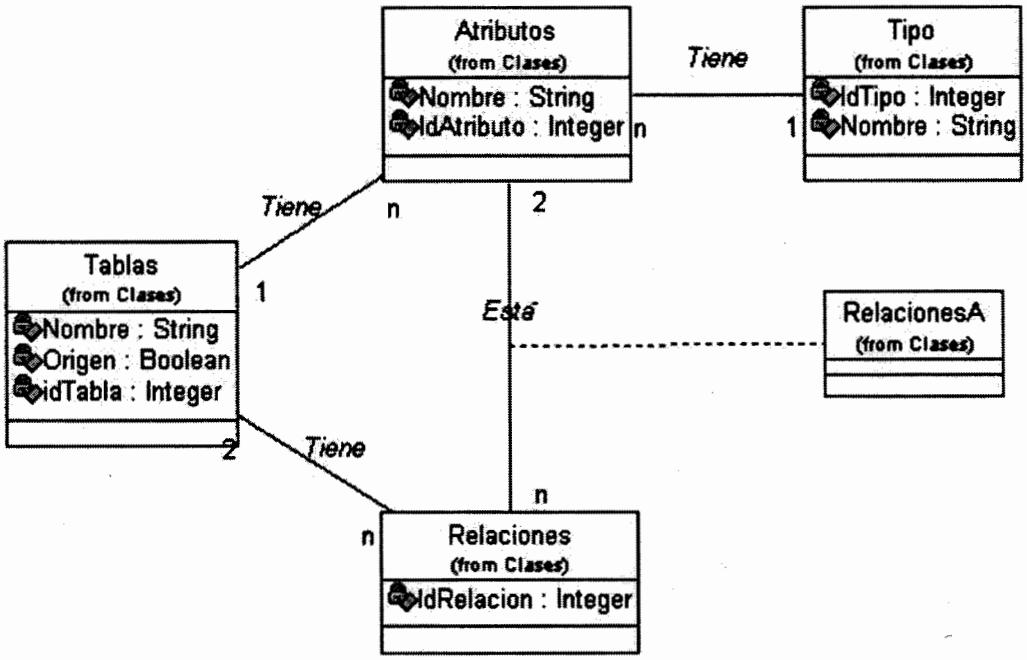

Figura 3.1: Modelo conceptual

3.3 Diagramas de interacción

Los diagramas de interacción no son más que una descripción del modo en el que cada operación detectada en los diagramas de secuencia lleva a cabo sus responsabilidades y modifica el estado del sistema. En UML los diagramas de

interacción pueden representarse a través de los diagramas de colaboración y/o de los diagramas de secuencia.

El tipo de diagrama seleccionado para construir los diagramas de interacción fue el de secuencia, debido a que muestra cómo los objetos se comunican unos con otros en una secuencia de tiempo, qué sucede en cada momento, y para ello contienen objetos con sus ciclos de vida y los mensajes que se envían entre ellos ordenados secuencialmente. En la figura 3.2 se muestra el diagrama de secuencia del caso de uso Relacionar bases de datos.

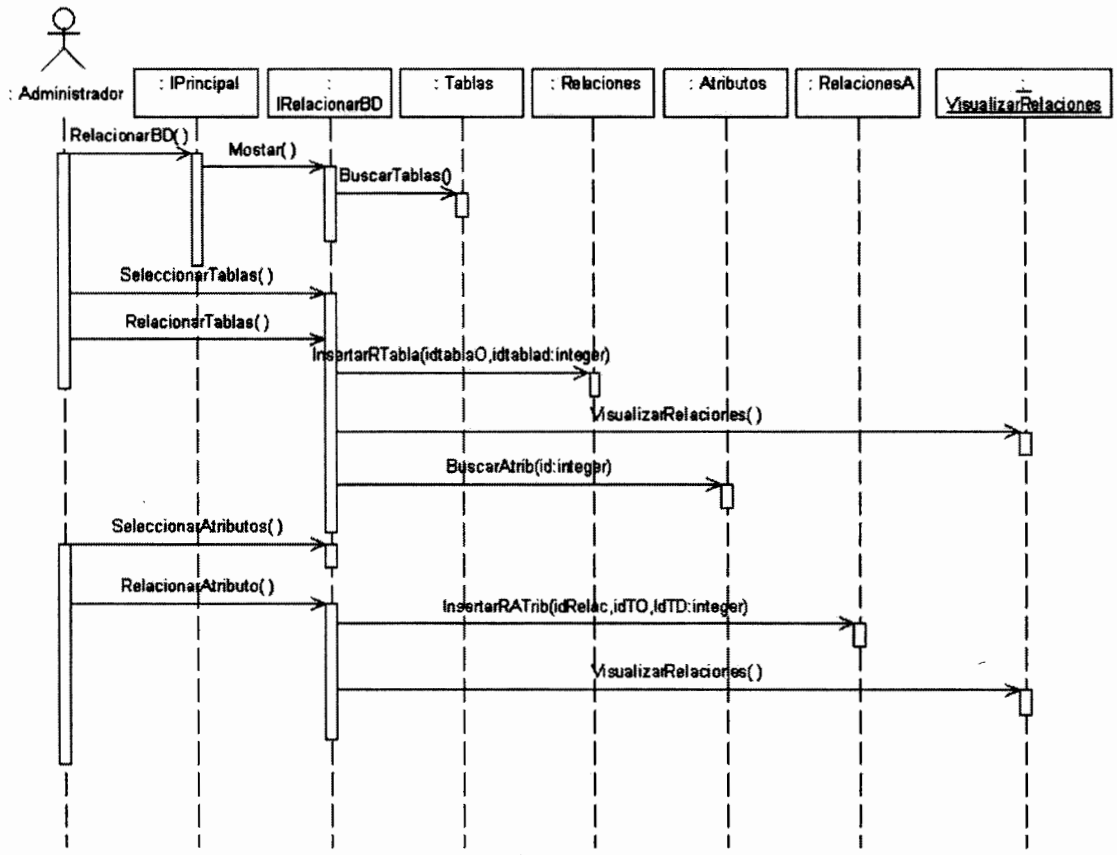

Figura 3.2: Diagrama de secuencia del caso de uso Relacionar bases de datos

En el Anexo II se representan los diagramas de secuencia de los otros casos de usos.

3.4 Diseño de la Base de Datos

Las tablas de la base de datos se originan a partir del modelo conceptual. La base de datos del sistema es sencilla, y se utiliza solo para almacenar algunos de los aspectos relacionados con la configuración de la herramienta. En la figura 3.3 se muestra el diagrama Entidad-Relación de la base de datos.

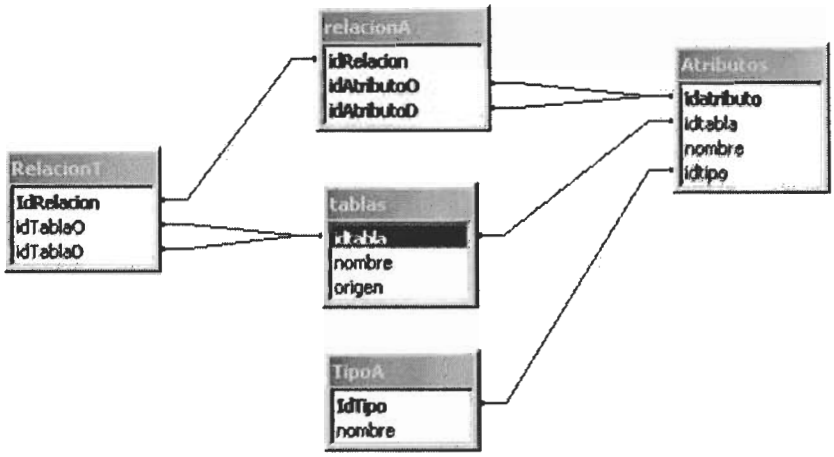

Figura 3.3: Diagrama Entidad-Relación de la base de datos

En el Anexo III se hace una descripción detallada de las tablas y cada uno de los atributos de la base de datos.

Conclusiones

Se finaliza la etapa de análisis y diseño de la herramienta, habiéndose expandido los casos de uso del núcleo central y se confeccionado sus respectivos diagramas de secuencia.

Con los resultados obtenidos, se creó el diagrama de diseño de la herramienta para la migración. Con todos estos elementos, se tiene una idea más precisa sobre los elementos constitutivos del sistema que se propone.

Conclusiones

Llegado a este punto se espera que el documento haya servido para la comprensión teórica del problema existente, arribándose a las siguientes conclusiones.

- Fue estudiada y modelada una solución práctica relacionada con la migración de datos.
- Se determinaron siete casos de uso, y se seleccionaron para desarrollar el núcleo central, cinco de ellos, teniendo en cuenta su importancia y nivel de complejidad.
- A través de este trabajo se logró elaborar la primera parte de una herramienta que permite:
    - Configurar la herramienta de modo que sea posible realizar una correcta migración.
    - Transferir la información de la base de datos origen a la destino, lo que hace que pueda garantizarse la existencia de los datos del control de inventario en el ASSETS.
    - Facilitar el trabajo del personal de economía para que no tengan que entrar los datos los mismos datos en dos aplicaciones diferentes.

Con el trabajo realizado se concibe, analiza y diseña una herramienta que permita migrar datos del Sistema de Inventario Participativo al ASSETS, dando así, cumplimiento al objetivo general del proyecto.

Después de terminado la etapa de análisis y diseño y la primera fase de desarrollo de la herramienta recomendamos:

- Dar continuidad a la elaboración de la herramienta.
- Continuar con la investigación para garantizar nuevas y buenas mejoras en futuras versiones del sistema.
- Profundizar en el conocimiento de las técnicas de la realización de la migración.
- Generalizar la concesión de la herramienta para la su uso entre dos gestores de bases de datos heterogéneas.

Bibliografía

1. *Ado. Net* www.elguille.info/NET/ADONET/ejemploAccess.htm (Abril/2004)
2. *Calidad* www.ne.com.co/html/esp/calidad.html
3. Carsí J.A., Ramos I., Silva J., Perez J., Anaya V., *Un generador automático de planes de migración de datos*, Revista I+D Computación, Julio 2002.
4. Ceria, S., *Casos de Uso:Un Método Práctico para Explorar Requerimientos.*
5. *Curso de C# y .Net* www.ciberaula.com/curso/puntonet/que_es/
6. *El inventario.* www.monografias.com (Abril/2004)
7. *Manual práctico para la utilización de ASSTES.*
8. *Migración a Internet de aplicaciones heredadas.* Computerworld www.idg.es/computerworld/articulo.asp?id=143705 (Abril/2004)
9. Pérez J.,Carsí J.A.,Ramos I.,Anaya V.,Silva J., *Migración de datos automática a partir de la información de los esquemas conceptuales*, JISBD'2001, VI Jornadas de Trabajo en Ingeniería del Software - Taller de Evolución del Software, Almagro - Ciudad Real, Noviembre 2001.
10. P. Letelier, P. Sanchez, I. Ramos, O. Pastor, *OASIS 3.0: Un Enfoque Formal para el Modelado Conceptual Orientado a Objeto*, Servicio de Publicaciones, Universidad Politécnica de Valencia, SPUPV -98.4011, ISBN 84-7721-663-0, 1998.
11. Rumbaugh, J. Jacobson, I. Booch, G. *El Lenguaje Unificado de Modelado. Manual de Referencia*. Pearson Education, S.A., Madrid, 2000.
12. Silva J.F., Carsí J.A., Ramos I., *Comparación automática de esquemas conceptuales orientados a objeto*, Revista Ingeniería Informática, Chile. Enero 2002.

13. Sitio de programación en castellano
http://www.programacion.com/tutorial/csharp.html (Mayo/2004)
14. Sitio web de la comunidad C#: http://www.csharp.net (Mayo/2004)
15. *Soluciones para Proyectos de Migración.*
www.nexteleng.es/catalogos/migracion.asp (Abril/2004)

Anexos

Anexo I. Casos de usos expandidos.

Caso de uso	
CU-01	Conectar base de datos
Propósito	Conectar las bases de datos para extraer la información de la descripción de ellas.
Actores Administrador, SGBD	

Acción del actor	**Respuesta del sistema**
1. El caso de uso comienza cuando el administrador selecciona que desea conectarse a la base de datos.	
	2. Se muestra la interfaz de conexión de las bases de datos.
3. El administrador entra la cadena de conexión de la base de datos origen.	
	4. El sistema se conecta a la base de datos. Si falla muestra un cartel de error diciendo que no se pudo conectar. Vuelve a 2. Al tercer intento fallido sale del programa.
5. Entra la cadena de conexión de la base de datos Destino.	
	6. El sistema se conecta a la base de datos. Si falla, muestra un cartel de error diciendo que no pudo conectar. Vuelve a 4. Al tercer intento fallido sale del programa.
	7. El sistema responde que se ha conectado a las bases de datos de forma satisfactoria.
8. El SGBD le brinda la sistema descripción de sus tablas y atributo	
	9. Sistema inserta en su base de datos la información referente a la descripción de las tablas y los objetivos.

Tabla I.1 Caso de usos expandido ***Conectar base de datos***

Caso de uso	
CU-02	Cargar Información
Propósito	Cargar las descripciones directamente de las bases de datos
Actores Administrador	
Acción del actor	**Respuesta del sistema**
1. El caso de uso se inicia cuando el Administrador selecciona que desea cargar la información de las bases de datos.	2. El sistema carga las descripciones de las tablas, es decir, los nombres de las tablas y los nombres y tipos de los atributos.

Tabla I.2 Caso de usos expandido ***Cargar información***

Caso de uso	
CU-04	Proponer relaciones entre atributos
Propósito	Hacer una propuesta de las relaciones entre atributo
Actores Administrador	
Acción del actor	**Respuesta del sistema**
1. El caso de uso se inicia cuando el administrador selecciona que desea ver una propuesta de relaciones entre atributos.	2. El sistema realiza una comparación entre los atributo de las tablas relacionadas y devuelve una propuesta de correspondencia entre ellos. 3. El sistema visualiza la propuesta de relación.

Tabla I.3 Caso de usos expandido ***Proponer relaciones entre atributos***

Caso de uso	
CU-05	Visualizar relaciones
Propósito	Mostrar las relaciones entre las tablas y los atributos de las bases de datos.
Actores Administrador	
Acción del actor	**Respuesta del sistema**
1. El caso de uso se inicia cuando el administrador selecciona visualizar las relaciones entre las bases de datos.	2. El sistema muestra la información de las relaciones entre las tablas y los atributos de las bases de datos. Por cada par de tablas relacionadas aparecerán los atributos comunes.

Tabla I.4 Caso de usos expandido ***Visualizar relaciones***

Caso de uso	
CU-06	Modificar relaciones
Propósito	Modificar las correspondencias entre las tablas o entre los atributos.
Actores Administrador	
Acción del actor	**Respuesta del sistema**
1. El caso de uso se inicia cuando el administrador selecciona modificar las correspondencias establecidas entre las bases de datos.	2. El sistema visualiza las relaciones establecidas entre las bases de datos.

3. El usuario selecciona qué relación entre atributos desea modificar.	
	4. El sistema elimina la relación entre los atributos seleccionados.
5. El usuario selecciona qué relación entre tablas desea modificar.	
	6. El sistema elimina la relación entre las tablas seleccionadas. Se elimina también la relación entre los atributos de dichas tablas.

Tabla I.5 Caso de usos expandido ***Modificar relaciones***

Caso de uso	
CU-07	Migrar datos
Propósito	Migrar los datos desde la base de datos origen hasta la destino
Actores SIP, SGBD	
Acción del actor	**Respuesta del sistema**
1. El caso de uso se inicia cuando el SIP desea migrar los datos.	
	2. El sistema copia la información que se va a migrar en el SGBD destino.

Tabla I.6 Caso de usos expandido ***Migrar datos***

Anexo II Diagramas de secuencia

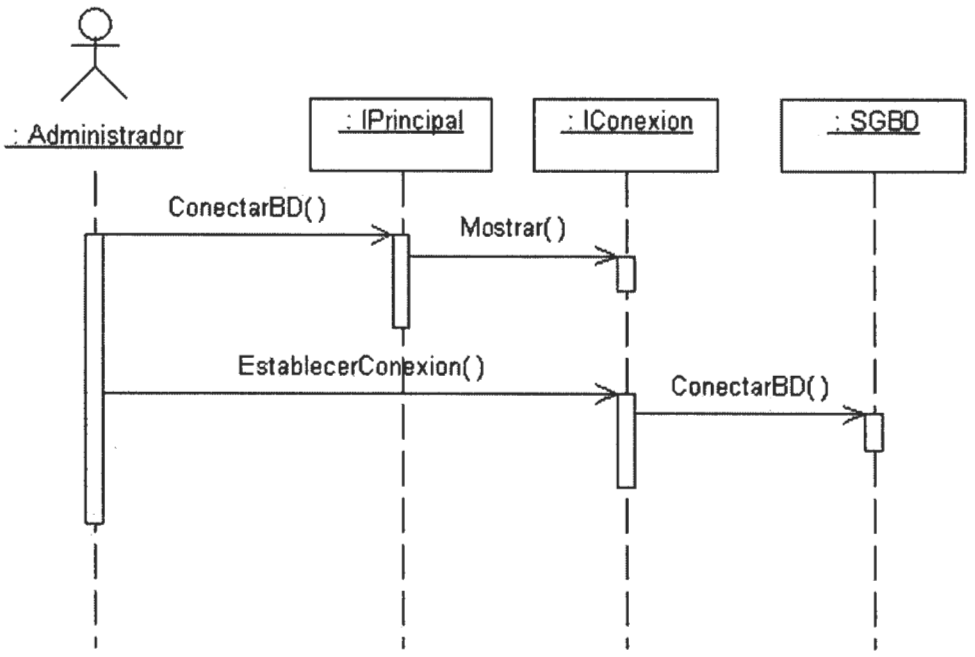

Figura II.1 Diagrama de secuencia del caso de uso ***Conectar base de datos***

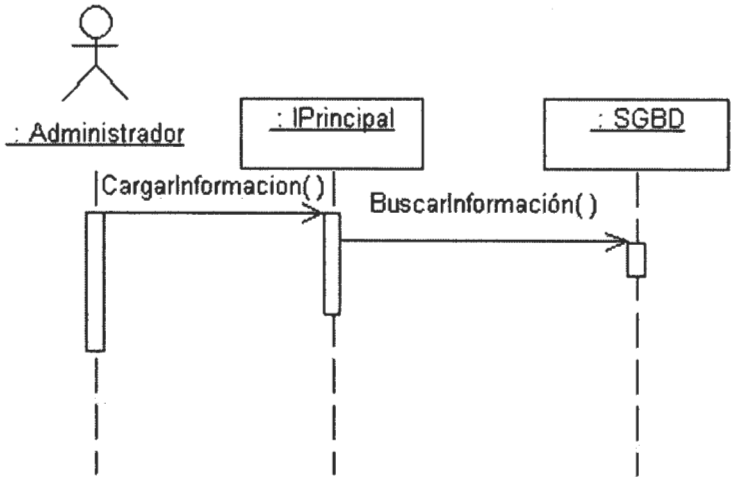

Figura II.2 Diagrama de secuencia del caso de uso ***Cargar información***

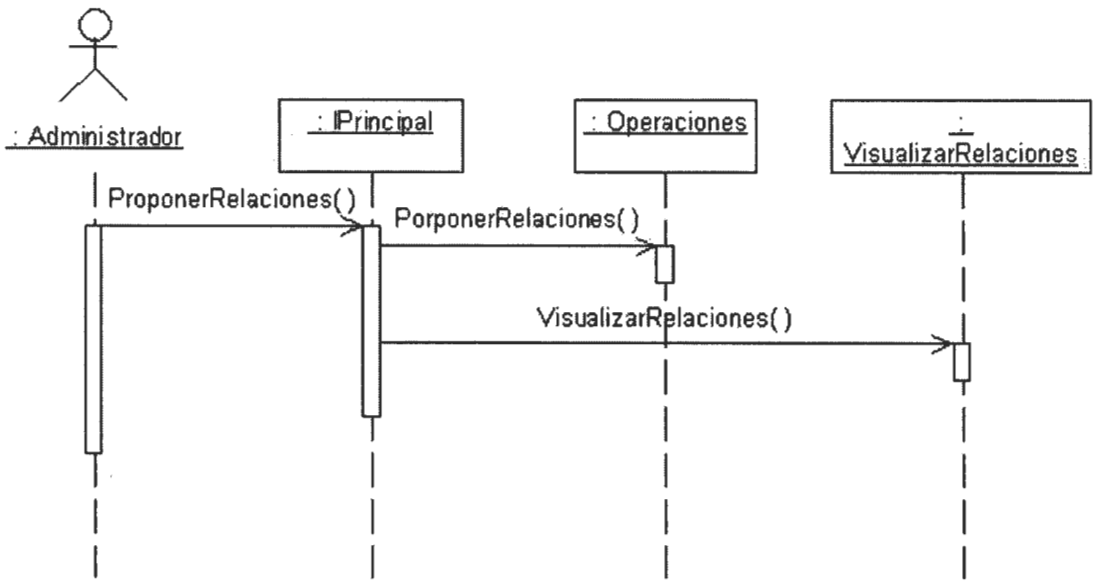

Figura II.3 Diagrama de secuencia del caso de uso ***Proponer relaciones entre atributos****.*

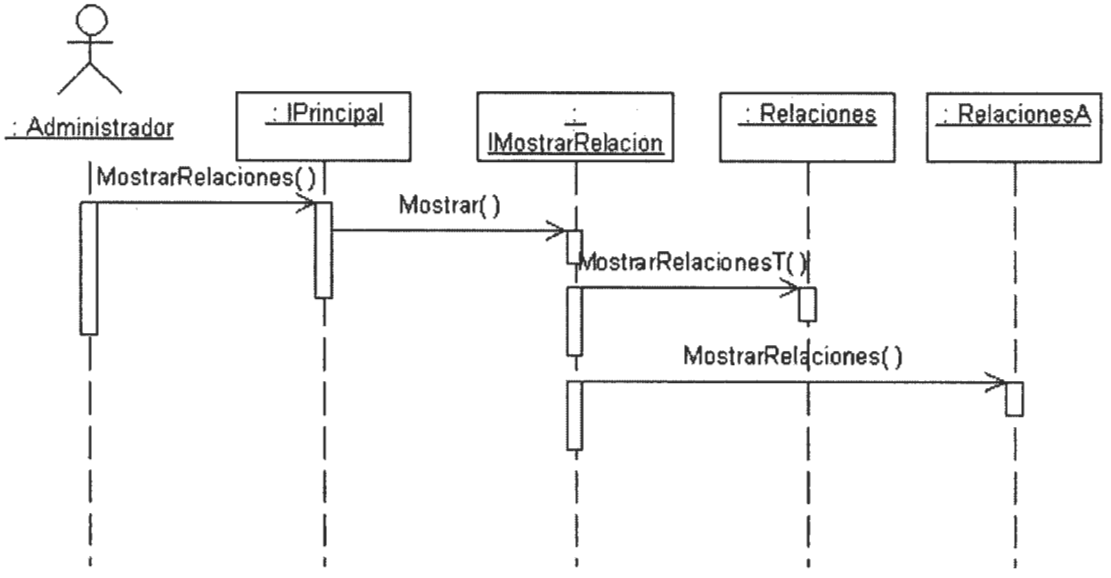

Figura II.4 Diagrama de secuencia del caso de uso ***Visualizar relaciones***

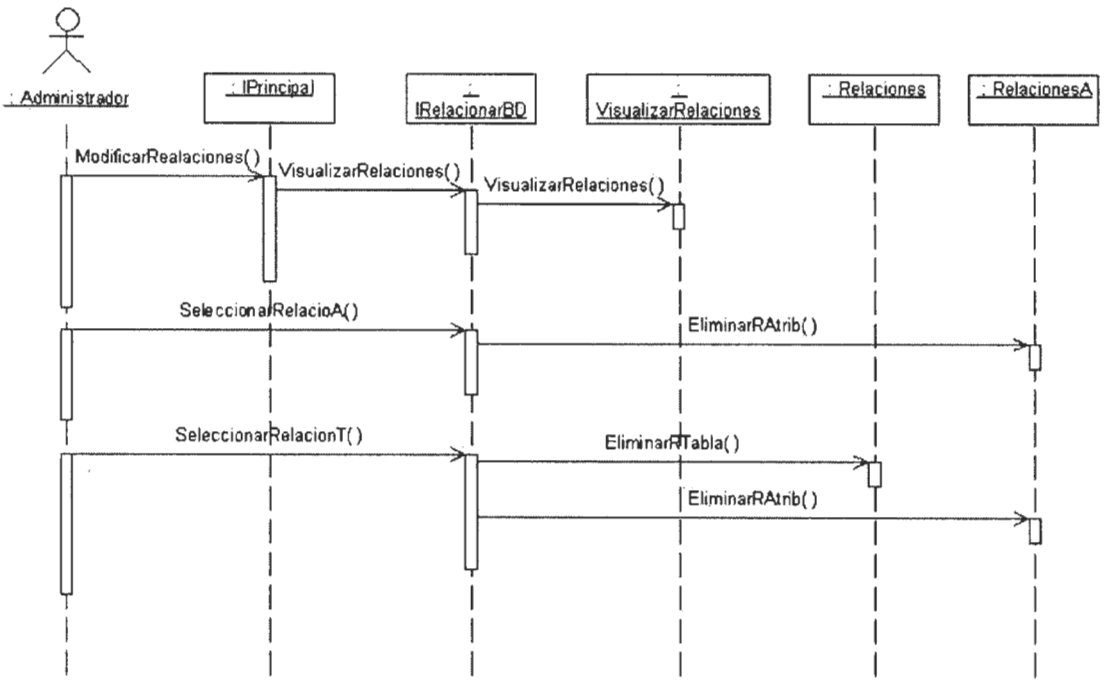

Figura II.5 Diagrama de secuencia del caso de uso ***Modificar relaciones***

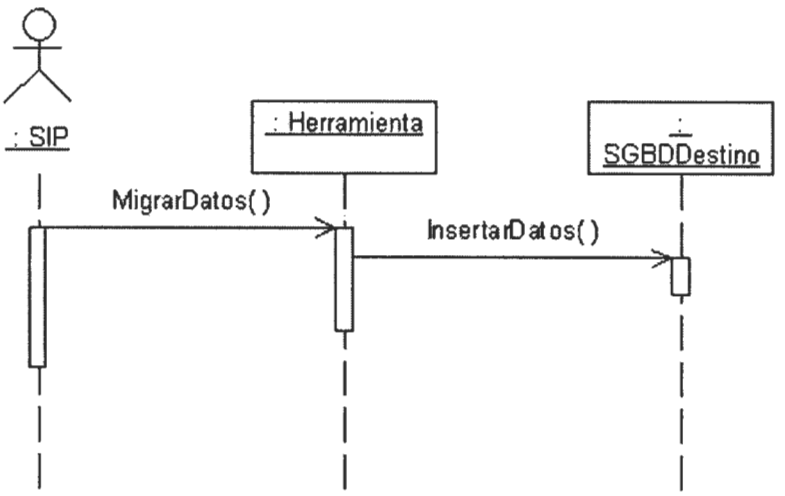

Figura II.6 Diagrama de secuencia del caso de uso ***Migrar datos***

Anexo III Descripción de las tablas de la bases de datos de la herramienta

Nombre: tablas		
Descripción: Almacena la descripción de todas las tablas (orígenes y destino).		
Atributo	**Tipo**	**Descripción**
idtabla	integer	Representa una tabla.
nombre	string	Nombre de la tabla.
origen	boolean	Si es VERDADERO en una tabla de la base de datos origen, en caso contrario es una tabla de la base de datos destino.

*Tabla III.1 Descripción de la tabla **tablas***

Nombre: Atributo		
Descripción: Almacena la descripción de los atributos de las tablas.		
Atributo	**Tipo**	**Descripción**
idatributo	integer	Representa un atributo.
idtabla	integer	Representa una tabla.
nombre	string	Nombre del atributo.
idtipo	integer	Representa el tipo de dato del atributo.

*Tabla III.2 Descripción de la tabla **Atributo***

Nombre: Tipo		
Descripción: Almacena los tipos de datos de los atributos.		
Atributo	**Tipo**	**Descripción**
idtipo	integer	Representa un tipo de dato.
nombre	string	Nombre del tipo de dato.

*Tabla III.3 Descripción de la tabla **Tipo***

Nombre: RelacionT		
Descripción: Almacena la correspondencia entre tablas		
Atributo	**Tipo**	**Descripción**
idrelacion	integer	Representa una relación entre dos tablas.
idTablaO	integer	Representa una tabla de la base de datos origen.

*Tabla III.4 Descripción de la tabla **RelacionT***

Nombre: relacionA		
Descripción: Almacena la correspondencia entre atributos de tablas relacionadas.		
Atributo	**Tipo**	**Descripción**
idrelacion	integer	Representa una relación entre dos tablas.
idAtributoO	integer	Representa un atributo de la tabla origen.
idAtributoD	integer	Representa un atributo de la tabla destino.

*Tabla III.5 Descripción de la tabla **relacionA***