Universidad de la República – Facultad de Ingeniería

Instituto de Computación

Estudio de Técnicas y Software para la Construcción de Sistemas de Data Warehousing

Taller V

Carrera de Ingeniero en Computación

Mayo 1998

Estudiantes: Alvaro Illarze (964306)
Verónika Peralta (964648)

Tutor: Dr. Raúl Ruggia

Resumen:

El presente documento presenta el proyecto de Taller 5 "Estudio de Técnicas y Software para la construcción de sistemas de Data Warehousing", desarrollado en la Facultad de Ingeniería de la República Oriental del Uruguay durante 1997.

En éste se detalla la construcción de un Sistema de Data Warehousing, que incluye el Data Warehouse, sus procesos de carga y actualización y varios Data Marts, tomando como ejemplo un centro de estudios.

Asimismo se plantean los principales problemas que se encuentran desarrollando un sistema de este tipo, evaluando distintas soluciones a los mismos. Se explica también la metodología usada para llevar a cabo el trabajo.

No es intención del proyecto, y por lo tanto no se encontrará en el presente, una evaluación de herramientas de software para Data Warehousing.

Por último se plantean las conclusiones de este trabajo y posibles áreas para mejorar o investigar.

Contenido

Apéndices

1. Introducción

1.1. Objetivos del Proyecto

El presente proyecto se enmarca dentro de un trabajo sobre los Sistemas de Data Warehousing en el Area de Concepción en Sistemas de Información (CSI) del Instituto de Computación de la Facultad de Ingeniería.

El objetivo principal es experimentar diferentes problemas del tipo de los que afectan a un Data Warehouse corporativo, y sobre esta base desarrollar estrategias y soluciones, lo más automáticas y generales posibles, para enfrentar estos problemas. En una perspectiva de largo plazo, se intenta sentar una base para el desarrollo de técnicas más generales.

En particular, se eligió implementar un sistema de apoyo a la gestión del Instituto de Computación de la Facultad de Ingeniería (In.Co.).

Los puntos más importantes del proyecto son el desarrollo del sistema arriba mencionado, y realizar una abstracción de la estrategia y metodología utilizadas, que serán el primer paso en el desarrollo de metodologías más generales. Este punto, uno de los principales objetivos del CSI, será ampliado en futuros proyectos y trabajos de investigación del área.

Cabe resaltar que en el proyecto no se intenta realizar una comparación entre herramientas de software o hardware para sistemas de Data Warehousing.

1.2. Cronograma de Trabajo

A continuación de detallan las etapas en las que consistió el proyecto y la duración de las mismas. En la Figura 1.1 se observa un diagrama de Gantt del cronograma.

- ♦ Estudio teórico de Data Warehousing: Abril – Mayo 1997 (2 meses).
  - – *Se siguió estudiando el tema hasta finalizado el desarrollo.*
- ♦ Análisis del sistema a desarrollar: Mayo 1997 (1 mes).
  - – *Estudio tecnología disponible. → Mayo 1997.*
  - – *Definición arquitectura. → Mayo 1997.*
  - – *Estudio general de requerimientos. → Mayo 1997.*
  - – *Estudio de bases fuentes. → Mayo 1997.*

- Análisis de requerimientos y datos: Junio – Octubre 1997 (5 meses)
  - *Análisis de requerimientos. → Junio 1997.*
  - *Análisis de datos de bedelía. → Julio – Agosto 1997.*
  - *Análisis de datos de presupuesto. → Octubre 1997.*
  - *Análisis de datos de asignaciones. → Octubre 1997.*
- Diseño 1er prototipo: Agosto – Setiembre 1997 (2 meses).
  - *Diseño multidimensional. (Access) → Agosto – Setiembre 1997.*
  - *Diseño de bases de datos. → Agosto – Setiembre 1997.*
- Diseño 2do prototipo: Noviembre 1997 – Febrero 1998 (4 meses).
  - *Diseño multidimensional. → Noviembre 1997 – Febrero 1998.*
  - *Diseño de base de datos. → Noviembre 1997 – Febrero 1998.*
  - *Programa de carga. → Diciembre 1997 –Febrero 1998.*
  - *Diseño de procesos. → Diciembre 1997 – Febrero 1998.*
- Verificación: Agosto 1997 – Febrero 1998 (7 meses).
- Metodologia: Febrero – Marzo 1998 (2 meses).
  - *Abstracción de una metodología. → Febrero – Marzo 1998.*
  - *Mecanismos especificación. → Marzo 1998.*
- Documentación: Marzo – Mayo 1998 (3 meses).

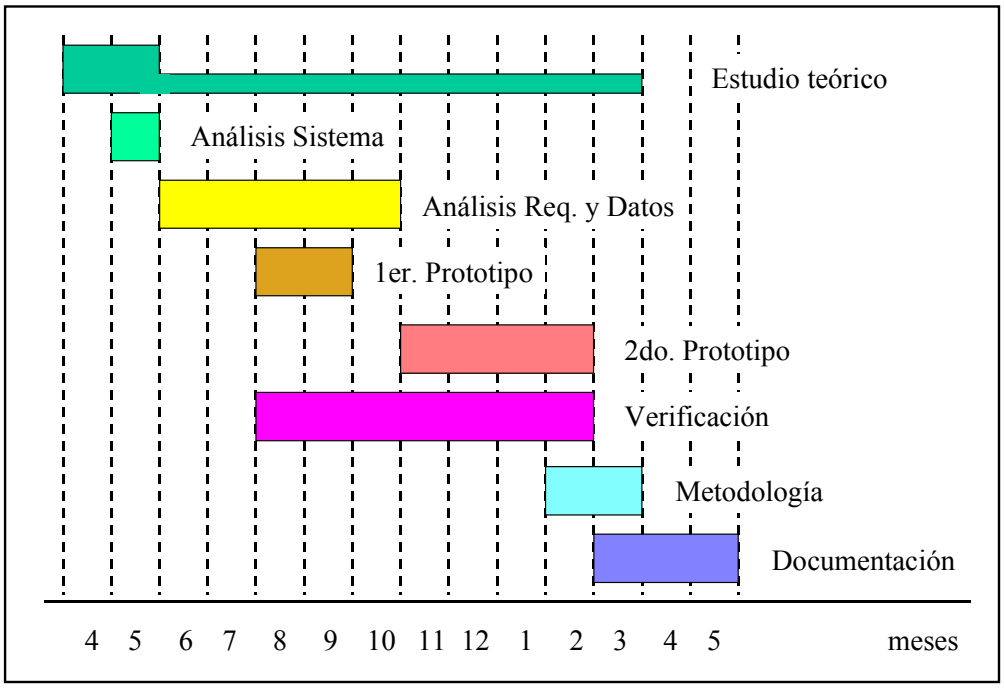


Figura 1.1 - Cronograma de trabajo

1.3. Descripción General del Sistema

El sistema construído responde a requerimientos de Análisis de Presupuesto, Asignación de docentes a cursos y exámenes y Seguimiento de estudiantes a lo largo de la carrera. Se estudian también otros requerimientos cuya implementación se deja para futuras versiones del sistema.

Se desarrolló una primera versión del sistema que abarca todos los niveles de la arquitectura de un Data Warehouse. Estos niveles pueden verse en la Figura 1.2 y serán estudiados en profundidad en el capítulo 4.

El sistema incluye desde el estudio de las bases fuentes hasta el diseño de los Data Marts e interfaces de usuarios, incluyendo la creación de pequeños programas de carga, en los casos en que no se disponía de datos computacionales.

Se integra la información a partir de las siguientes fuentes de datos:

- Base de Bedelía – Contiene información sobre los estudiantes (fecha de ingreso y egreso), materias dictadas, inscripciones a cursos, y resultados de cursos y exámenes.
- Base de Presupuesto – Contiene información sobre grado y horas presupuestadas para los docentes, además de información sobre cambios en el presupuesto (nombre genérico que representa ingresos, renuncias, aumento o reducción de horas).
- Base de Asignaciones – Contiene información sobre la asignación de docentes para impartir cursos, y tomar exámenes. La información disponible representa tanto las horas planificadas, como las reales.

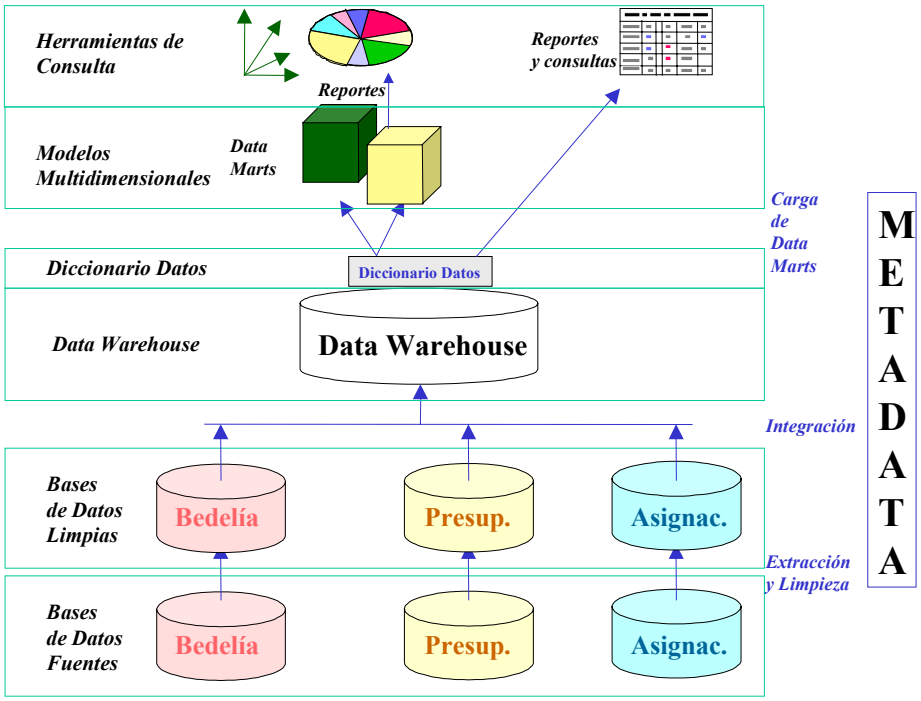


Figura 1.2 – Arquitectura del sistema construído.

1.4. Descripción del Informe

El capítulo 2 (Tecnología de Data Warehousing), presenta una introducción a los sistemas de Data Warehousing y los conceptos más importantes de su tecnología.

En los capítulos 3 (El Problema a Resolver), 4 (Descripción del Sistema) y 5 (Metodología Seguida) se describe la aplicación desarrollada. En el capítulo 3 se hace un análisis de la realidad, y se plantea el problema a resolver. Se describen a rasgos generales los requerimientos y los datos disponibles. En el capítulo 4 se especifica el sistema desarrollado y todos los aspectos relevantes de su arquitectura. Se hace énfasis especialmente en los problemas encontrados y los mecanismos de resolución de los mismos. En el capítulo 5 se describe la metodología usada a lo largo del proyecto.

El capítulo 6 (Conclusiones) presenta las conclusiones obtenidas a partir de este proyecto, y sugiere trabajo futuro.

El Apéndice A (Análisis de Requerimientos), detalla en amplitud los requerimientos planteados por los usuarios.

El Apéndice B (Descripción de las Tablas), tiene la descripción de cada uno de los campos, valores válidos y observaciones especiales para cada una de las tablas de las Bases de Datos Fuentes, Bases de Datos Limpias y del Data Warehouse.

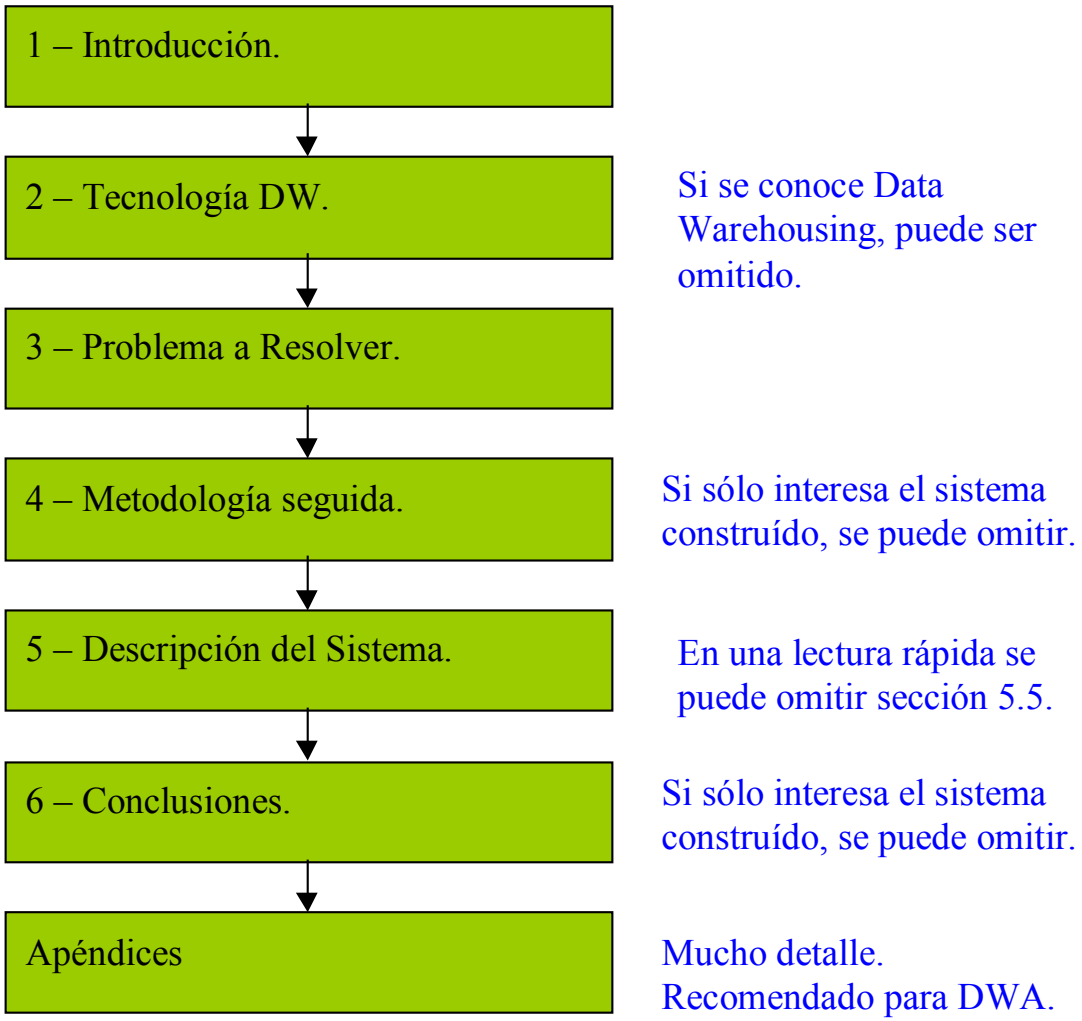
El Apéndice C (Guía de Convenciones), detalla un formato de documentación para las tablas del Data Warehouse y convenciones sobre el uso de nombres y tipos en los campos.

El Apéndice D (Consultas de Carga de los Data Marts), detalla las consultas a partir de las cuales se cargan los Data Marts, incluyendo el modelo conceptual dimensional, y el origen de los datos para generar dimensiones y medidas.

En la Bibliografía se pueden encontrar varios libros y artículos publicados para investigar más sobre la tecnología de Data Warehousing.

Por último, cabe reseñar que hay un informe anexo al presente, en el cual se presentan los manuales de usuario y de operación del sistema construído.

Flujo de los capítulos (según el interés principal):



2. Tecnología de Data Warehousing

2.1. Motivación

Los sistemas de Data Warehousing son el centro de la arquitectura de los Sistemas de Información de los 90's. Han surgido como respuesta a la problemática de *extraer información sintética a partir de datos atómicos almacenados en bases de datos de producción.* Uno de los objetivos principales de este tipo de sistemas es servir como base de información para la toma de decisiones.

Los beneficios obtenidos por la utilización de este tipo de sistemas se basan en el acceso interactivo e inmediato a información estratégica de un área de negocios. Este acercamiento de la información al usuario final permite una toma de decisiones rápida y basada en datos objetivos obtenidos a partir de las bases de datos de la empresa, eventualmente heterogéneas. Estos beneficios aumentan cuanto más importantes son las decisiones a tomar y cuanto más crítico es el factor tiempo.

Un *Sistema de Data Warehousing* incluye funcionalidades tales como:

- *Integración de bases de datos* heterogéneas. Puede comprender bases relacionales, documentales, geográficas, archivos, etc.
- *Ejecución de consultas complejas no predefinidas,* visualizando el resultado en forma de gráfica y en diferentes niveles de agrupamiento y totalización de datos.
- *Agrupamiento y desagrupamiento* de datos en forma interactiva.
- *Análisis de problemas en términos de dimensiones*. Por ejemplo, permite analizar datos históricos a través de una dimensión tiempo.
- *Control de calidad de datos,* para asegurar no sólo la consistencia de la base, sino también la relevancia de los datos en base a los cuales se toman las decisiones.

2.2. Características

W. Inmon en [WI-93]**Error! Bookmark not defined.** define a un Data Warehouse como una colección de datos:

- Orientada a sujetos.
- Integrada.
- Variante en el tiempo.
- No volátil.

Un Data Warehouse soporta procesamiento informático, brindando una sólida plataforma de datos históricos, integrados, de los cuales hacer análisis, para la toma de decisiones.

Es orientado a sujetos:

Un primer aspecto de un data warehouse es que esta orientado a los mayores sujetos de la empresa. El mundo operacional esta diseñado alrededor de aplicaciones y funciones, como por ejemplo pagos, ventas y entregas de mercadería para una institución comercial. Un data warehouse está organizado alrededor de los mayores sujetos, como cliente, vendedor, producto y actividades.

El mundo operacional concierne al diseño de la base de datos y al diseño de procesos. Un data warehouse está enfocado en la modelización de los datos y el diseño de la base de datos exclusivamente. El diseño de procesos (en su forma clásica) no es parte del data warehouse.

Los datos son integrados:

El aspecto más importante del ambiente de un data warehouse es que sus datos están integrados. La integración se realiza cuando los datos son movidos del ambiente operacional, antes de entrar en el warehouse.

Por ejemplo, un diseñador puede representar el sexo como "M" y "F", otro puede representarlo como "0" y "1", o "x" e "y", y otro usar las palabras completas "masculino" y "femenino". No importa la fuente de la cual el campo sexo llegue al data warehouse, debe ser guardado en forma consistente; es decir que los datos deben estar integrados.

Es variante en el tiempo:

Los datos en el data warehouse son precisos para un cierto momento, no necesariamente ahora; por eso se dice que los datos en el data warehouse son variantes en el tiempo.

La varianza en el tiempo de los datos se manifiesta de muchas maneras.

- El data warehouse contiene datos de un largo horizonte de tiempo. Las aplicaciones operacionales, por el contrario, contienen datos de intervalos de tiempo pequeños por cuestiones de performance (tamaño chico de las tablas).
- Toda estructura clave en un data warehouse contiene implícita o explícitamente un elemento de tiempo. Esto no necesariamente pasa en el ambiente operacional.
- En general, los datos de un data warehouse, una vez almacenados, no pueden ser modificados, no se permiten actualizaciones (updates). En el ambiente operacional, los datos, precisos al momento de acceso, pueden ser actualizados según sea necesario.

Es simple de manejar:

Actualizaciones, inserciones y borrados son efectuados regularmente a los datos operacionales, en una base de registro por registro (record by record). La manipulación de datos en un data warehouse es mucho más sencilla. Sólo ocurren dos operaciones, la carga inicial, y el acceso a los datos. No hay necesidad de actualizaciones (en su sentido general).

Hay consecuencias muy importantes de esta diferencia de procesos con un sistema operacional: A nivel de diseño, en un data warehouse no hay que controlar anomalías producidas por los updates, ya que no hay updates. Se pueden tomar libertades de diseño físico como optimizar el acceso a los datos, y denormalización física. Otra consecuencia es la simplicidad de la tecnología del data warehouse, en lo que respecta a backups, recuperación, locks, integridad, etc.

2.3. Arquitectura lógica

La arquitectura lógica de un sistema de Data Warehousing es del tipo mostrado en la Figura 2.1.

Un Sistema de Data Warehousing consta de cinco partes fundamentales:

- Bases de datos fuentes.
- Data Warehouse
- Data Marts.
- Herramientas de extracción de información.
- Meta Data.

Un sistema de Data Warehousing no es un sistema monolítico, sino una combinación flexible de múltiples módulos. Dichos módulos pueden ser productos o soluciones específicamente desarrolladas para la aplicación a resolver.

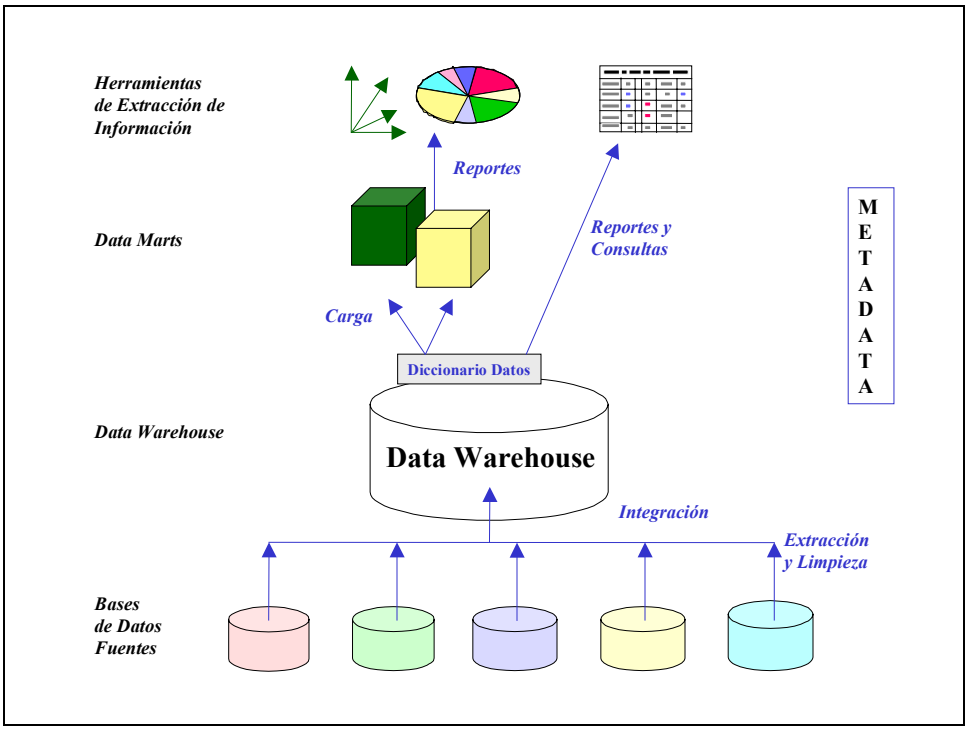


Figura 2.1 - Arquitectura lógica de un Sistema de Data Warehousing

2.3.1. Bases de Datos Fuentes

Consisten en bases de datos de producción e históricos. Son típicamente las bases operacionales de la organización. Sin embargo, se están integrando cada vez más bases de distribución pública sobre industria, demografía y clientes potenciales.

Estas bases de datos pueden estar implementadas en diferentes tipos de sistemas: BD Relacionales, BD geográficas, BD de textos, archivos, etc. Una característica común es que almacenan ítems de datos atómicos, los cuales son relevantes como datos de producción, pero pueden ser demasiado detallados como base para la toma de decisiones. Además, la noción de calidad de los datos en estas bases se basa en la consistencia de dichos registros, independientemente de la relevancia que éstos tengan dentro del problema.

2.3.2. Data Warehouse

Es una base de datos que incluye los datos relevantes para la toma de decisiones en un área de negocios o globalmente en la empresa. Los datos almacenados en el Data Warehouse son, fundamentalmente, agrupamientos y totalizaciones de los datos relevantes que se encuentran en las bases de producción y en los históricos.

Los bases de datos fuentes pasan primero por un proceso de extracción y limpieza individual, y luego son integradas y se cargan en el data warehouse.

Un componente importante del Data Warehouse es el Diccionario de Datos o Catálogo, el cual describe los datos almacenados con el objetivo de facilitar el acceso a los mismos a través de las herramientas de explotación. El Diccionario de Datos establece correspondencias entre los datos almacenados y los conceptos que estos representan de forma de facilitar la extracción de información por parte del usuario final.

El Control de Calidad de los datos en el Data Warehouse es un tema fundamental en el éxito de un proyecto de este tipo: *Si quien debe tomar decisiones duda de la calidad de la información, entonces no utilizará el sistema de Data Warehousing*.

La calidad de los datos en el Data Warehouse involucra, no sólo la consistencia clásica en bases de datos, sino también las nociones de *pertinencia* y *relevancia* de los datos almacenados. Por *pertinencia* se entiende la utilidad de un dato para ser usado en la toma de decisiones; por ejemplo, los resultados de ventas en zonas en las cuales no se ha terminado una campaña de publicidad pueden no ser pertinentes como base para decisiones. La *relevancia* de un dato es su significación como ítem de información; por ejemplo, un formulario de una encuesta en la cual un cliente opina sobre un producto que ha experimentado pocas veces podría no ser significativo en la evaluación del producto.

El Data Warehouse es tratado en toda la bibliografía. Son una buena introducción al tema [RK-96], [KB-97] **Error! Bookmark not defined.**y [SA-95b]**Error! Bookmark not defined.Error! Bookmark not defined.**.

2.3.3. Data Marts

Cada área de la empresa puede necesitar que su propia visión de los negocios sea organizada como un array o vista multidimensional de manera de optimizar sus requerimientos específicos. Generalmente no es deseable que la misma base multidimensional soporte los requerimientos de todas las áreas de la empresa.

Mientras que las vistas multidimensionales son diseñadas para optimizar el acceso de usuarios finales de cada área, la base de datos integrada del Warehouse es diseñada para optimizar el acceso de todas las áreas.

Se le llama Data Warehouse a la base integrada, y Data Marts a las vistas multidimensionales de cada área.

Los Data Marts son generalmente subconjuntos del Data Warehouse, pero pueden también integrar un número de fuentes heterogéneas.

El uso efectivo de los Data Marts en un ambiente de Data Warehousing es un factor importante para la efectividad del Data Warehouse, y puede también ser determinante en el éxito del proyecto de desarrollo.

Como discute P. Brooks en su artículo [PB-97]**Error! Bookmark not defined.**, los Data Marts están creciendo, llegando a tener tamaños semejantes a los Data Warehouse corporativos de menor escala. Aunque hoy en día es difícil diferenciar a los Data Marts y Data Warehouses por su tamaño, algunas distinciones entre ellos son todavía importantes:

- *Un Data Mart está enfocado a una sóla área o grupo de usuarios*, mientras que un Data Warehouse contiene información de diferentes sujetos y áreas de la corporación.
- Una organización puede tener un sólo Data Warehouse, pero varios Data Marts.
- *Como los Data Marts contienen menos información, son más fáciles de entender y navegar* que los Data Warehouses corporativos. Un Data Warehouse puede contener tanta información que es difícil de manejar por los usuarios.

2.3.4. Herramientas de explotación

Son herramientas con interfaces orientadas a usuarios, que les ayudan a conducir el análisis y extraer información para la toma de decisiones.

Hay diversos tipos de herramientas. Las clásicas son: interfaces para *consultas y reportes complejos* y productos de análisis multidimensional (*OLAPs*). Las nuevas tecnologías soportan la nueva generación de herramientas de análisis: *data mining* y *simulación de negocios*.

J. Weldom, en su artículo [JW-96]**Error! Bookmark not defined.**, brinda una discución detallada sobre este tipo de herramientas, y cita ejemplos de productos existentes.

2.3.4.1. Consultas y reportes complejos.

Permiten al usuario construir gráficas y reportes a partir de la información almacenada en el Data Warehouse y de información descripta a través del Diccionario de Datos.

Algunas funcionalidades típicas de estas herramientas son: agrupamiento y desagrupamiento dinámico de datos en reportes, cambios en el orden de los campos del reporte o visualización del resultado de las consultas en forma gráfica (barras, torta, puntos, etc.).

Estas herramientas generan las expresiones en el lenguaje de consulta que recupera los datos pedidos (típicamente SQL), se conectan al Data Warehouse, recuperan el resultado y lo formatean según la especificación dada.

2.3.4.2. OLAPs (On Line Analytical Processing)

Permiten representar los datos del problema en términos de *dimensiones*. Por ejemplo, si se trata de ventas de productos en diferentes zonas, una dimensión del problema son las zonas, otra los productos y otra el tiempo.

De esta manera, las consultas de análisis de datos de una dimensión en función de la otra se realizan en forma inmediata.

Una introducción sobre OLAP, puede encontrarse en [OLAP-97]**Error! Bookmark not defined.**. Los modelos multidimensionales y el diseño multidimensional son muy bien tratados en [KS-95]**Error! Bookmark not defined.**, [RK-96]**Error! Bookmark not defined.** e [IBM-98]**Error! Bookmark not defined.**.

2.3.4.3. Data mining

Permiten explorar el Data Warehouse en búsqueda de correlaciones desconocidas o inesperadas entre los datos. Por ejemplo, en un sistema médico, se pueden buscar relaciones entre enfermedades y decesos de pacientes. Algunas de éstas pueden ser candidatas a convertirse en nuevas causas de decesos en pacientes con ciertos perfiles, y otras podrían corresponder a datos erróneos – pero médicamente posibles – que se detectan por el bajo porcentaje de aparición en el total de datos de la base.

2.3.4.4. Simulación de negocios

Tienen como principal propósito chequear la efectividad de las reglas de empresa. Permiten crear modelos para testear el impacto de cambios en el ambiente de negocios.

Como resultado se pueden establecer nuevas reglas de empresa. Luego hay que realimentar los aplicativos operacionales.

2.3.5. Meta Datos

Son información descriptiva de los datos y procesos del Data Warehouse. Son una parte fundamental de la documentación del sistema, que ayuda al DWA con el mantenimiento del mismo.

Deben incluir información de dominio, reglas de validación, derivación y transformación de los datos extraídos. También describen las bases de datos del Data Warehouse, incluyendo reglas de distribución y control de la migración hacia los Data Marts.

El conocimiento de los meta datos es tan esencial como el conocimiento de los datos del Data Warehouse.

Los meta datos, deberían estar disponibles para los usuarios, para ser usados en sus análisis.

2.4. Funcionalidad

El objetivo de un ambiente de Data Warehousing es principalmente convertir los datos de aplicaciones del ambiente transaccional (OLTP), en ***datos integrados de gran calidad.*** *Luego se los debe almacenar en una* estructura que optimice el acceso por parte de usuarios finales en un ambiente decisional (OLAP). Durante este proceso, datos totalizados son agregados al Data Warehouse. Los datos son transferidos desde el ambiente operacional al Data Warehouse, en una base periódica, apropiada al tipo de análisis de negocios necesario.

Como plantean T. Moriarty y R. Greenwood en [TM-96]**Error! Bookmark not defined.**, se puede dividir las funcionalidades del Data Warehouse en cinco grandes grupos, cada uno de las cuales es responsable de un conjunto de procesos específicos, esenciales para el ambiente de soporte decisional:

- Acceso a Fuentes (Source).
- Carga (Load).
- Almacenamiento (Storage).
- Consultas (Query).
- Meta Datos (Meta Data).

Las funcionalidades de acceso a fuentes, carga y almacenamiento soportan la migración de los datos operacionales al data warehouse. La funcionalidad de consultas maneja los procesos que soportan el acceso y análisis de los datos para toma de decisiones. La funcionalidad de meta datos sirve como base para las otras cuatro, ya que provee los datos que controlan sus procesos e interacciones.

2.4.1. Acceso a Fuentes

La funcionalidad de acceso a fuentes incluye los procesos que se aplican en las bases de datos fuentes a los datos que serán transferidos. Los datos pueden provenir de fuentes muy diversas. Determinar la mejor fuente de datos, evitando redundancias, es una de las tareas más largas y difíciles.

Muchos de los procesos asociados con la función de acceso a fuentes, como **mapeo, integración, análisis y calidad de los datos**, ocurren durante la fase de análisis y diseño del Data Warehouse. En realidad entre un 70 y 90% del tiempo de desarrollo del Data Warehouse está destinado a estas actividades, según estudios de W. Inmon en [WI-93]**Error! Bookmark not defined.**.

Desafortunadamente, automatizar estas tareas no es nada fácil. Algunas herramientas pueden ayudar a detectar problemas en la calidad de los datos y generar programas de extracción, pero la mayor parte de la información requerida para el desarrollo está en la mente de los analistas que trabajan con las bases de datos fuentes.

Los factores que impactan directamente sobre el tiempo destinado a estas actividades son: el número de aplicativos fuentes que serán mapeados al Data Warehouse, la calidad de los meta datos mantenidos en esas aplicaciones, y las reglas de empresa que las gobiernan.

2.4.2. Carga

La funcionalidad de carga comprende los procesos asociados con la migración de los datos desde los aplicativos fuentes a las bases del Data Warehouse. Incluyen: **extracción, limpieza, transformación y carga de datos**.

La **extracción** involucra acceder a los datos de los aplicativos. Es el primer paso para la preparación de los datos. Hay varias alternativas de extracción que balancean la performance y las restricciones de tiempo y almacenamiento. Si las aplicaciones fuentes mantienen una base de datos en línea, se puede hacer una consulta que cree directamente los archivos de extracción. Hay que asegurarse que no se actualicen los datos mientras se hace la extracción para no generar inconsistencias. La performance puede caer si las transacciones en línea compiten con la extracción. Una solución alternativa es crear una vista materializada (ODS), desde la cual extraer los datos. El inconveniente aquí, es el espacio de disco adicional para guardar esa copia de la base. El tiempo es un factor crucial; muchos aplicativos de extracción tienen un ciclo batch, en el cual transacciones fuera de línea son aplicadas a la base de datos.

Luego de la extracción, los datos son accedidos para determinar si hay problemas de calidad. La **limpieza** de los datos puede ser manejada de muchas maneras. Si los errores son inherentes a los aplicativos fuentes, los datos pueden ser limpiados sistemáticamente como parte del proceso de transformación. Desafortunadamente, muchos errores ocurren porque los aplicativos fuentes sólo tienen una mínima validación de dominio, que permite la aparición de datos inválidos. La única manera de solucionarlos es corriendo rutinas pesadas de validación a nivel de fuentes. Los errores que surgen de tipeos incorrectos, son muy difíciles de detectar y corregir.

El paso final en la preparación de los datos para ser cargados en el Data Warehouse, es la **transformación**. Este proceso invoca reglas de conversión, de valores de aplicativos locales a valores globales, integrados.

Cuando este proceso es completado, se **cargan** los datos al Data Warehouse.

Algunos productos permiten automatizar parte de estos procesos, pero en general no son suficientemente potentes para resolver el universo de problemas involucrados.

Algunas herramientas de transformación de datos son discutidas por J. Williams en [JW-97b]**Error! Bookmark not defined.**, y R. Kimball en [RK-96b]**Error! Bookmark not defined.**. Un listado muy completo de herramientas se encuentra en el Data Warehouse Information Center – Herramientas de Extracción y Limpieza ([LG-97])**Error! Bookmark not defined.**. Todo el proceso de conversión es tratado en profundidad por K. Bohn en [KB-97]**Error! Bookmark not defined.**.

2.4.3. Almacenamiento

La funcionalidad de almacenamiento comprende la arquitectura necesaria para integrar las vistas varias al Data Warehouse. Aunque a menudo hablamos del Data Warehouse como si fuera un único almacén de datos, sus datos pueden estar distribuidos en múltiples bases manejadas por diferentes DBMSs. Dos tipos de manejadores se ajustan bien a esta tarea: relacionales (RDBMSs) y multidimensionales (MDDBMSs).

Un MDDBMS organiza los datos en un array de *n* dimensiones. Cada dimensión representa algún aspecto de los negocios a ser analizado. Las bases multidimensionales presentan los datos de manera que los usuarios puedan entenderlos y accederlos fácilmente.

Generalmente no es deseable que la misma base multidimensional soporte los requerimientos de todas las áreas de la empresa. Un RDBMS usualmente se ajusta más al manejo de la base integrada, y un MDDBMS al manejo de Data Marts.

La separación entre el Data Warehouse corporativo y sus Data Marts satélites introduce la necesidad de una estrategia que coordine la distribución de los datos hacia los Data Marts. Se debe considerar la incorporación de un servidor de replicación, que entregue los datos correctos al Data Mart correcto en el momento correcto.

A medida que el número de Data Marts va creciendo, crece también la necesidad de administración y coordinación central de actividades como manejar versiones, asegurar la consistencia e integridad de los datos, controlar la seguridad, y mantener la performance global. La coordinación y administración de toda la colección de Data Marts, debe tener un enfoque centralizado, en lugar de distribuir las actividades de administración entre los diferentes usuarios.

Como describe P. Brooks en [PB-97]**Error! Bookmark not defined.**, la administración de los Data Marts, es un área con crecientes requerimientos, como la coordinación, la extracción de los datos, la lectura, los procedimientos de replicación, los procedimientos de backup y recuperación, el manejo de metadatos, la seguridad, y la performance.

Los datos son almacenados en varios niveles. Los más actuales se guardan en un medio de fácil acceso en línea. Datos más viejos se pueden guardar en un medio seguro, pero más barato. Y los datos históricos pueden ser guardados en otros medios, o eliminados si ya no tienen más valor decisional.

Estos puntos son discutidos en profundidad en el artículo anterior ([PB97]).**Error! Bookmark not defined.**

2.4.4. Consulta

El ambiente de consultas permite a los usuarios conducir el análisis y producir reportes a través de sus herramientas de consulta y simulación. Se debe estudiar que tipo de herramienta se adecúa mejor a las necesidades de cada grupo de usuarios.

Muchas veces alcanza con un conjunto de reportes armados a medida, o pueden ser necesarias herramientas genéricas con consultas predefinidas y capacidad de consultas ad-hoc.

El principal compromiso de las herramientas es lograr una buena performance en las consultas de los usuarios, la mayor parte de las cuales se basan en totalizaciones y filtros.

El arquitecto del Data Warehouse debe determinar como totalizar los datos. Existen varios enfoques viables: la sumarización puede ser hecha durante la carga, y almacenada en el Data Warehouse; durante la replicación a los Data Marts; o a demanda, por las herramientas de consulta y simulación.

Los OLAPs pueden ser *servidores*, si almacenan los datos en un modelo dimensional; o *clientes* si se conectan a un servidor de base de datos (por ejemplo Relacional) y transforman los requerimientos dimensionales en términos relacionales. La utilización de unos u otros depende del tipo de problema a resolver, no existiendo un modelo que sirva para el 100% de los problemas.

2.4.5. Meta Datos

T. Moriarty y R. Greenwood, en su artículo [TM-96]**Error! Bookmark not defined.**, hacen especial hincapié en la definición de meta datos. Toda la información que ayude al DWA con el mantenimiento del sistema debe encontrarse en los meta datos. Todas las decisiones de diseño e implementación deben reflejarse también en los meta datos.

Se debe mantener al día la información de dominio, reglas de validación, derivación y transformación de los datos extraídos. También se deben describir las bases de datos del Data Warehouse, incluyendo reglas de distribución y control de la migración hacia los Data Marts.

Los aplicativos que monitorean los procesos del Data Warehouse (como extracción, carga, y uso) deben crear meta datos que serán usados para determinar que tan bien se comporta el sistema.

El conocimiento de los meta datos es tan esencial como el conocimiento de los datos del Data Warehouse.

Parte de los meta datos, deberían estar disponibles para los usuarios, para ser usados en sus análisis. Los administradores pueden manejar y proveer el acceso a través de los servicios del repositorio.

Las cinco funcionalidades del Data Warehouse proveen un marco de trabajo para controlar la arquitectura de los componentes.

Este marco, describe las transformaciones de los datos desde un ambiente OLTP, a un ambiente OLAP.

3. El Problema a Resolver

3.1. Introducción

Como se dijo anteriormente, el objetivo del proyecto es experimentar las diferentes estrategias y problemáticas de desarrollar e implementar un sistema de data warehousing. El planteo inicial era utilizar parte de la base de datos de bedelía y otras bases de datos que pudieran existir, y a partir de ellas construir un Data Warehouse.

Se visitaron usuarios potenciales del producto, en busca de requerimientos lo más variados posibles. La interacción con usuarios permitió enfrentarse a dificultades reales y concretas. Se pretendió desarrollar una aplicación real, con la suficiente complejidad, sin perder generalidad en la búsqueda de técnicas y soluciones.

El apoyo de los usuarios fue fundamental para:

- Definir los requerimientos y funcionalidades que pretenden del producto.
- Proporcionar datos con los que les sería útil trabajar; los cuales pueden no existir físicamente (en una base de datos).

En el capítulo 5 se establecen pautas para realizar este análisis.

3.2. Usuarios y Requerimientos

Se realizaron entrevistas a usuarios en cinco áreas: seguimiento de estudiantes, análisis de presupuesto, asignación de docentes, administración de electivas y administración de proyectos de taller V.

Las tres primeras áreas fueron analizadas en profundidad, y se implementó un prototipo que diera solución a los requerimientos. A continuación se brinda una idea global de los mismos, y se detallan en el apéndice A. Las otras dos áreas no fueron implementadas, pero igualmente se detalla la propuesta, los requerimientos planteados y los motivos por los que se excluyeron del sistema.

3.2.1. Seguimiento de estudiantes

Se entrevistó en varias ocasiones al doctor Juan Echagüe. Es docente del Instituto de Computación, e investigador del Pedeciba Informática en el área de Métodos Formales.

Como parte de sus tareas extra-enseñanza, se dedica a hacer un seguimiento de los estudiantes a lo largo de la carrera. Estudia el comportamiento de los estudiantes, para así poder explicar algunos fenómenos como las deserciones. Trabaja también en la identificación de los factores que influyen en ese comportamiento.

Desde hace algunos años, otros grupos de la facultad han estado trabajando en la elaboración de un modelo para describir tales fenómenos. Algunos de los estudios fueron realizados por el Ing. Eduardo Fernandez quien publicó un modelo estadístico de comportamiento de estudiantes ([EF-96]).

No se cuenta con los datos, ni las herramientas necesarias para tal tarea, por lo que muchos de los resultados son intuitivos o basados en la experiencia.

3.2.1.1. Preguntas que interesaría contestar

- ¿Dónde está la gente en la carrera?: Sería útil poder determinar la cantidad de estudiantes que hay en cada etapa de la carrera, cuántos de ellos están activos, y clasificarlos de acuerdo al tiempo que hace de su última actividad. Se desea también contar los ingresos y los egresos.
- ¿Cómo construir un perfil de carrera?: Se quisiera construir perfiles de estudiantes, para luego clasificarlos de acuerdo a su historia y poder predecir su comportamiento.
- ¿Cómo evaluar la velocidad de avance?: Se quiere medir la velocidad de avance de los estudiantes en la carrera de acuerdo a determinados patrones.
- ¿Cómo calcular el volumen de estudiantes activos?: Se necesita prever cuántos estudiantes se van a presentar a un curso o examen, para poder destinar recursos. La observación de datos históricos es esencial para efectuar una proyección.
- ¿Qué gente se presenta y cómo prever qué gente se va a presentar a un examen?: Es importante reconocer qué estudiantes se presentan en cada período.

3.2.1.2. Evaluación

El usuario tiene una idea utópica de lo que se querría obtener, pero no sabe como llegar a ella. Inclusive, algunas de las cosas que se quieren medir no tienen una definición clara; por ejemplo: cómo son los perfiles de estudiantes, o cuándo un estudiante está activo.

No está claro cual será el grado de ayuda de este proyecto a su trabajo, pero se evaluó que el mero hecho de acercar los datos y presentarlos adecuadamente, vale la inversión. Una vez desarrollado el primer prototipo aparecerían nuevas funcionalidades, más concretas.

3.2.2. Análisis de presupuesto

Se entrevistó por este tema al profesor Rodolfo Paiz. Es docente del Instituto de Computación desde 1988, y delegado docente en la Comisión de dicho Instituto.

Además, colabora con la Dirección del Instituto realizando la tarea de destinar los recursos presupuestarios del Instituto a la contratación de docentes.

Una de las funciones más importantes, es discriminar qué parte del presupuesto corresponde a cargos vacantes, y destinarlo a la contratación de nuevos docentes. Para ello es necesario disponer de datos detallados de la dedicación de cada docente, reducciones y ampliaciones horarias, y renuncias. Es fundamental contar con datos históricos.

3.2.2.1. Descripción del Presupuesto

El total del presupuesto destinado al instituto puede dividirse en cinco grupos:

- Ejecutado: Parte del presupuesto destinada al pago de salarios de los docentes que están trabajando con contrato. Es bastante estable. Disminuye con las renuncias y reducciones horarias, y aumenta con las ampliaciones horarias y las nuevas contrataciones.
- Vacante: Parte del presupuesto proveniente de cargos que quedaron vacantes, generalmente por renuncias. Puede ser utilizada para aumentar el ejecutado.
- Disponible: Parte proveniente de reducciones horarias. Puede ser utilizada para aumentar el ejecutado.
- Incremental: Aumento del presupuesto destinado al instituto. Ocurre con poca frecuencia, y en general se debe a situaciones excepcionales que el instituto no puede solventar, y la facultad aprueba un aumento para encargarse de esos gastos. El incremental generalmente es destinado para contrataciones especiales o ampliaciones horarias de docentes con postgrados en el exterior.
- Comprometido: Parte del presupuesto destinada al pago de nuevos cargos, que no han asumido aún. Se trata como el ejecutado; no puede hacerse uso de ese dinero, aunque nadie lo esté cobrando todavía.

3.2.2.2. Algunos de los requerimientos a resolver:

- Total ejecutado, discriminado por docente. El objetivo es evaluar como se está distribuyendo el presupuesto.
- Total de vacantes, disponible, e incrementos del presupuesto. El objetivo es aprovechar al máximo los recursos monetarios. Para ello es necesario saber el margen de gastos disponible.
- Total de comprometidos. Se quiere saber que docentes asumirán próximamente a fin de cumplir con actividades administrativas.
- Listado de docentes con grados y horas. Es útil tener la lista de docentes actuales para muchas actividades administrativas.
- Históricos.

3.2.2.3. Evaluación

No se cuenta con buenas herramientas para realizar la tarea, la que es realizada de manera semi-manual. En este caso se sabe exactamente lo que se quiere obtener, aunque podrían presentarse nuevas funcionalidades.

Actualmente se cuenta con planillas con parte de la información, pero muy mal organizada, lo cual dificulta mucho el análisis. Otros datos provienen de resoluciones del Consejo, en papel impreso.

El sistema se debe diseñar para que pueda ser generalizado fácilmente al presupuesto de toda la Facultad.

3.2.3. Asignación de docentes

Se entrevistó por este tema al profesor Gustavo Crispino. Es docente del Instituto de Computación desde 1986, en el área “Laboratorio de Tratamiento del Lenguaje Natural”.

Además de las tareas de enseñanza, colabora en la Comisión del Instituto. Parte de su trabajo en la Comisión es resolver el problema de la asignación de docentes a las distintas actividades (dictado de cursos, toma de exámenes, corrección, etc.) y hacer un seguimiento y evaluación de los mismos.

3.2.3.1. Algunos de los requerimientos a resolver:

- Planilla docente con grados, horas y cursos que dicta. El objetivo es distribuir con coherencia a los docentes en los cursos y exámenes.
- Medir el trabajo real: Las horas trabajadas no son exactamente las asignadas. El objetivo es determinar que tanto se apartó la asignación del trabajo real.
- Información mensual de cada responsable de materia sobre trabajo efectivo.

3.2.3.2. Evaluación

No se cuenta con datos precisos sobre los docentes, como grado, carga horaria, asignaciones anteriores, y otras actividades que desarrolla dentro del Instituto. Por tal motivo, hoy en día la asignación se hace en base a memoria, y experiencias personales, de una manera totalmente manual.

Los datos disponibles son de carácter informativo, preparados en un momento posterior a la asignación. No se cuenta con ninguna herramienta para realizar la tarea.

La información histórica no está muy refinada. Es interesante sobre todo a partir del primer semestre de 1997.

3.2.4. Administración de electivas

Se entrevistó por este tema a la profesora María Urquhart. Trabaja en el Instituto de Computación desde 1988, y es investigadora del Pedeciba Informática en el área de Investigación Operativa.

Además de las actividades de enseñanza, está encargada de todos los asuntos pedagógicos y administrativos relacionados con las materias electivas.

Una de las funciones más importantes, es resolver el problema de que alumnos destinar a cada electiva, siendo que muchas de ellas tienen un cupo máximo. Otra tarea muy importante es llevar registros históricos de electivas dictadas en años anteriores.

3.2.4.1. Algunas de los requerimientos a resolver:

- Cantidad de inscriptos en una electiva
- Listas de inscriptos ordenados por promedio de escolaridad, o por alguna materia o conjunto de materias.
- ¿Qué generaciones se inscribieron y qué carreras?
- ¿Cuánta gente puede presentarse a un examen de carrera vieja?
- Nota y cantidad de gente que cursó una electiva.
- Organización de horarios y salones para electivas.
- Históricos.

3.2.4.2. Evaluación

Actualmente cuenta con archivos de texto, Latex, y material impreso que describen el contenido, y características importantes de cada materia. Hay archivos de Word, que son los presentados en Bedelía. También se cuenta con una lista de materias, que son las ya validadas por el consejo.

La información que se quiere guardar consta de dos partes: una parte descriptiva (texto o imágenes) que incluye objetivos, programa y bibliografía; y una parte informativa (ciertos campos fijos) como docente, cupo y fecha. La misma está estructurada en una carpeta, ordenada por nombre de materia, la cual no es muy cómoda para las diferentes consultas que es necesario hacer a los datos.

Tampoco se cuenta con buenas herramientas para realizar la tarea, que se hace de forma totalmente manual.

No está muy claro que se pretende del producto. Se nota la necesidad de tener herramientas que automaticen al menos parte del trabajo, pero no hay seguridad de que ésta sea la herramienta adecuada. Tal vez sólo una base de datos, que permita almacenar imágenes (descripciones de electivas), sea más adecuada.

Se deja la elección de la herramienta adecuada para futuras extensiones del proyecto.

3.2.5. Administración de proyectos de taller V

Se entrevistó por este tema al profesor Héctor Cancela. Trabaja en el Instituto de Computación desde 1990, y es investigador del Pedeciba Informática en el área de Investigación Operativa.

Además de la enseñanza, se encarga de la administración, asignación y seguimiento de los talleres de quinto año.

Una de las funciones más importantes y difíciles, es controlar la evolución de los proyectos. Para eso es necesario tener datos actualizados de los puntos más relevantes de cada taller, así como información histórica para comparar.

3.2.5.1. Algunas de los requerimientos a resolver:

- Lista de talleres y alumnos asignados por año.
- Consultas comunes.
- Datos que asocien teléfonos y e-mail de tutores con alumnos.

3.2.5.2. Evaluación

Se cuenta sólo con archivos Ascii, sin formato. No se dispone de las herramientas necesarias para desempeñar la función fácilmente.

Se vio la necesidad de tener esa información estructurada de una manera más accesible, sin definirse claramente, cual sería su formato.

No está clara la utilidad que pueda tener dicho proceso de automatización. Se notó falta de motivación por el proceso. Se descartó la inclusión dentro del proyecto.

3.3. Sistema de Producción y Bases Fuentes

Hay 5 posibles fuentes de datos:

- Sobre estudiantes, materias, cursos y exámes, con información de Bedelía. (Base de Bedelía).
- Sobre asignación de docentes a cursos y examenes. (Base de Asignación).
- Sobre los docentes, grado, carga horaria y salario. (Base de Presupuesto).
- Sobre estudiantes que cursan taller V, y sus proyectos de grado. (Base de Taller V).
- Sobre las electivas y cursos de posgrados. (Base de Electivas).

Las mismas provienen de diferentes plataformas; algunas inclusive se diseñaron específicamente para este proyecto. A continuación se detallan las generalidades de cada una.

3.3.1. Base de Bedelía

Se dispone de 4 tablas en formato export de Oracle, que describen las inscripciones a carreras, inscripciones a cursos, actividades de inscripciones, cursos y exámenes, e información sobre las materias.

La más grande de ellas cuenta aproximadamente con unos 57.000 registros.

En general, es un buen diseño de tablas, normalizado. Se encontraron algunos problemas de representación y limpieza de datos, que se detallarán más adelante.

Se cuenta con datos históricos a partir de 1965. Se espera recibir actualizaciones semestralmente, aunque no está definido si será en forma incremental, o total.

3.3.2. Base de Asignación

Se dispone de archivos de texto, con cierta estructuración lógica, pero sin un formato homogéneo capaz de ser interpretado de una manera automática.

Un archivo es presentado cada semestre, con la información de la asignación a cursos de ese semestre, y otro archivo se presenta cada período de exámen con la información de la asignación para la toma de exámenes.

Se planteó la posibilidad de automatizar este proceso de presentación de la información, eligiéndose algún estructura (a convenir con los usuarios) para los archivos. Se deberá considerar también si mantenerlos en formato texto, o algún otro formato más conveniente, por ejemplo, planillas.

Se cuenta con datos históricos a partir de 1997. Se espera recibir información al comienzo de cada semestre, y grupo de períodos de exámenes (julio-agosto, diciembre y febrero-marzo).

3.3.3. Base de Presupuesto

Se dispone de archivos con formato d-Base (.dbf), con información de sueldos y carga horaria de docentes.

Los mismos están muy mal diseñados, no normalizados, con utilización de campos para diferentes objetos lógicos en función del contexto. Se tiene gran cantidad de campos, muy difíciles de comprender, y sin lugar de donde obtener la información pertinente. Además el formato de los mismos varía de una entrega a otra.

Muchos de esos campos no son de interés para el estudio, y el resto deben ser limpiados con mucho cuidado, con un proceso difícil de automatizar.

Hay más datos en papel impreso proveniente de resoluciones del Consejo.

Se tiene poca información, correspondiente al año 1994. Se espera recibir mensualmente la información correspondiente al mes anterior.

3.3.4. Base de Electivas

Se dispone de información muy variada: documentos de texto, documentos Word, documentos Latex, etc, ya sea en forma de archivos o material impreso. La documentación está muy bien organizada en una carpeta, y puede ser accesible. Se requerirá de una fase de carga de la misma.

Se cuenta con datos históricos precisos, a partir de 1995. Se espera recibir actualizaciones cada semestre.

3.3.5. Base de Taller V

No se dispone de ningún tipo de información. Se sabe de la existencia de texto impreso, describiendo las características de un taller en particular. Dicho texto no tiene ninguna estructura fija, si bien sigue una cierta cantidad de puntos, o items que ordenan la descripción. También existen documentos con la asignación de esos talleres a determinados estudiantes, y algún(os) tutor(es).

No se sabe con exactitud la cantidad de datos históricos, ni la calidad de los mismos. Se estima mucho trabajo de recopilación de información.

Se espera recibir información al comienzo de cada año.

3.3.6. Relaciones entre las Bases

Una primer propuesta, a muy alto nivel de cómo se relacionan las bases puede verse en la Figura 3.1. En el capítulo 4 (Descripción del Sistema) se estudiará el tema con más detalle.

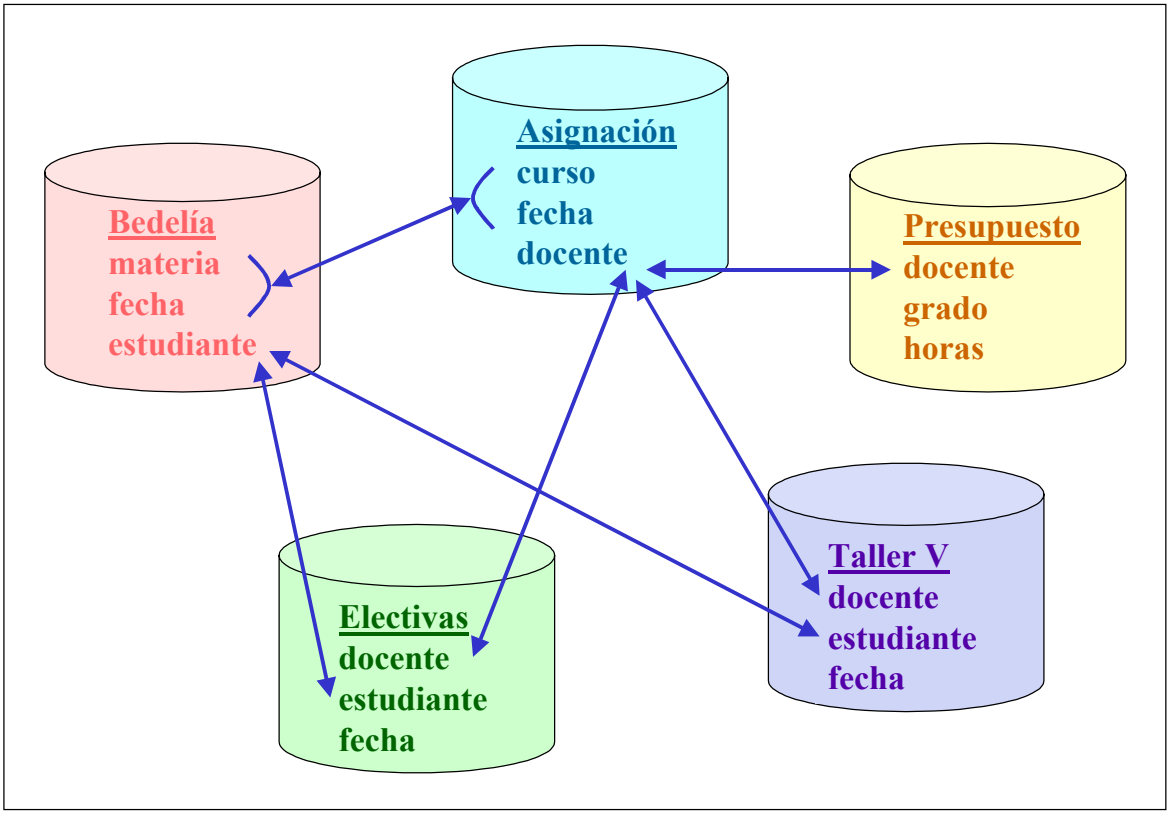


Figura 3.1 - Primera aproximación de Relaciones entre Bases

3.4. Tecnología disponible y alternativas técnicas

Manejadores de Base de Datos

El manejador de base de datos disponible es Oracle 7.3. Se cuenta también con manejadores de menor escala como Access y dBase. Hay posiblidad de instalar SQL*Net (comunicación con Oracle) y conseguir en internet los drivers ODBC para SQL*Net.

Herramientas de extracción y limpieza

Por motivos económicos no hay posiblidades de trabajar con una herramienta especializada. Los procesos que se programen, pueden hacerse en SQL, SQL aumentado (PL/SQL), SQL embebido u ODBC.

Herramientas Front-End

Se cuenta con las herramientas Cognos PowerPlay y Business Objects adquiridas para este y otros proyectos del área; y Startracker de distribución libre. Todas ellas son herramientas OLAP.

Se cuenta además con la herramienta Cognos Impromptu para la realización de reportes complejos.

Se tiene a disposición todo el software de desarrollo de Microsoft, como por ejemplo Visual Basic y Visual C++.

Sistemas Operativos y Redes

El sistema operativo dominante es Unix. Existen varias versiones del mismo, entre ellas, SunOS 4, SunOS 5, Solaris 1, Solaris 2 y AIX 4.

En equipos PC los sistemas operativos son Windows 3.11, Windows 95 y Windows NT.

Todos los equipos se encuentran conectados a una red Ethernet. Las estaciones Unix utilizan el protocolo NFS de Sun, y los PC tienen diferentes interfases de comunicación a la red, destacándose PcNfsPro y Samba.

Hardware

Se cuenta con diversos tipos de estaciones de trabajo para los puestos Unix, destacándose un equipo RS6000 de IBM, que funciona como servidor de Oracle. Hay alrededor de 20 equipos disponibles.

Los PC son también de configuraciones variadas, teniendo disponible para desarrollo un Pentium de 100 MHz.

4. Descripción del Sistema

4.1. Arquitectura

El sistema construído tiene la arquitectura mostrada en la Figura 4.1.

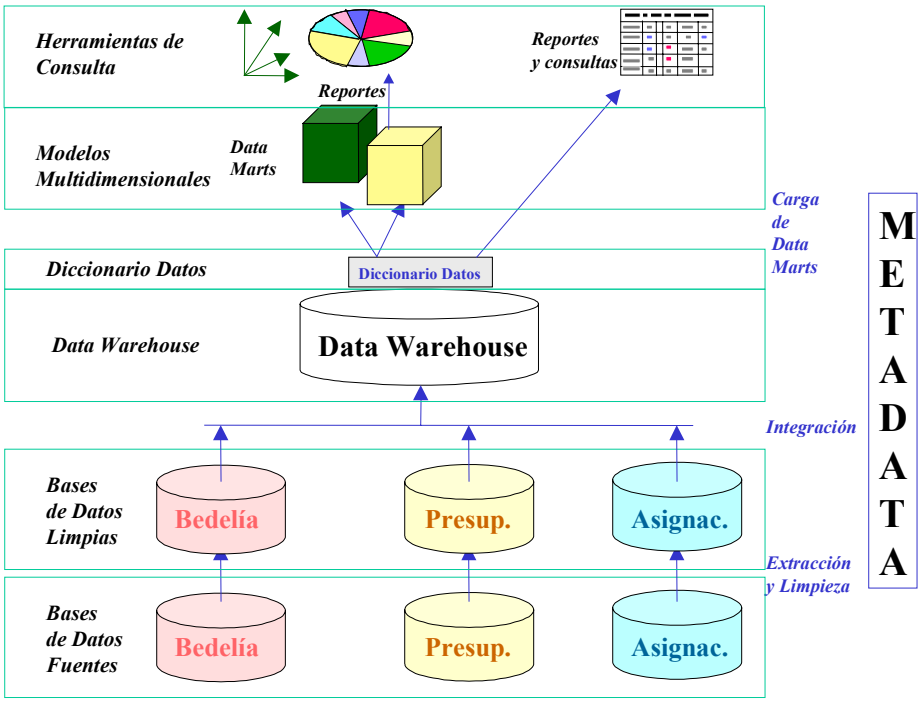


Figura 4.1 – Arquitectura del sistema construído

Como se observa en la figura, el sistema está dividido en varios niveles. A continuación se describen las generalidades de cada uno de ellos y en el resto del capítulo se explican con mayor detalle.

Las bases fuentes que fueron integradas al Data Warehouse son: base de Bedelía, base de Asignación y base de Presupuesto. Una idea general de las mismas ya fue dada en la Sección 3.3, y se ampliará en la seccion 4.2.

El nivel de base de datos limpios está creado en Oracle. Sólo se tendrán tablas para las cuales el proceso de limpieza es tan complejo que exige dividirlo en varias etapas. Se describen más adelante en la sección 4.3.

El Data Warehouse se encuentra en su totalidad en una Base de Datos Oracle. El diseño del mismo se describe más adelante en la sección 4.4.

Se utilizará como Diccionario de Datos el catálogo que provee la herramienta Impromptu de Cognos. El diseño del mismo se describe más adelante en la sección 4.4.

Los Data Marts están implementados como cubos multidimensionales, en formato propietario de Cognos PowerPlay. El detalle de los mismos es presentado en la Sección 4.7.

La metadata del sistema consiste en la información de la que disponen usuarios y administradores del sistema para poder manejar el mismo. La metadata creada en este proyecto se describe en la sección 4.6.

4.2. Bases de Datos Fuentes

Las Bases Fuentes que alimentan el sistema son 3; la base de Bedelía, la base de Asignaciones y la base de Presupuesto. Una idea general de las mismas ya fue dada en la Sección 3.3, y se ampliará a continuación[1].

4.2.1. Base de Bedelía

La Base de Bedelía está compuesta por información de:

- Estudiantes
- Materias
- Actividades (exámenes dados y cursos aprobados por los estudiantes)
- Inscripciones a cursos

4.2.1.1. Tabla Inscarr (información sobre Estudiantes)

Tiene datos históricos sobre todos los estudiantes que se inscribieron a las carreras 60, 61, 70 y 71 (respectivamente: Ingeniería de Sistemas en Computación, Analista Programador, Ingeniería en Computación e Ingeniería en Computación – reválida)

Sus atributos principales son: estudiante, carrera, fecha de ingreso a Facultad y fecha de egreso de cada una de las etapas de la carrera (Analista e Ingeniero).

El tamaño hasta Marzo 1997 era de aproximadamente 14.000 registros.

4.2.1.2. Tabla Matcarr (información sobre Materias)

Tiene datos históricos sobre todas las materias que se han dictado para las carreras 60, 61, 70 y 71. Sus atributos principales son: nombre, carrera a la que corresponde y semestre en que se dicta (relativo a la carrera).

El tamaño hasta Marzo 1997 era de aproximadamente 250 registros.

[1] Para quien esté interesado en una descripción más exhaustiva, en el Apéndice B se puede encontrar una descripción completa de las tablas, los valores posibles de los atributos codificados y otras particularidades de las bases.

4.2.1.3. Tabla Actividades

Tiene datos históricos sobre todas las actividades de los estudiantes de carreras 70 y 71.[2] Una actividad es un examen dado, un curso (aprobado o no), o una inscripción a curso (para materias que tienen como previa la inscripción a curso). Además, se tiene la nota correspondiente (hay valores especiales para las aprobaciones sin nota e inscripciones).

Las actividades de tipo Inscripción corresponden a materias que tienen como previa la inscripción a curso. Estos casos son muy pocos[3] y no tienen ningún requerimiento asociado, por lo que no serán incluídos en el Data Warehouse.

El tamaño hasta Marzo 1997 era de aproximadamente 57.000 registros.

4.2.1.4. Tabla Inscur (Inscripciones a cursos)

Tiene datos operacionales sobre las inscripciones a curso de estudiantes de carreras 60, 61, 70 y 71, en materias que tienen ganancia de curso. Sólo están las inscripciones que aún no han generado actividad por no contarse con el acta de curso. Una vez que se genera una actividad, el registro es borrado de esta tabla, por lo que se podría perder esta información entre dos actualizaciones del Data Warehouse. Esto sumado al hecho de que son muy pocas las materias que tienen ganancia de curso, hace que esta tabla no sea considerada para integrar el Data Warehouse.

El tamaño hasta Marzo 1997 era de 2.400 registros.

4.2.2. Base de Asignaciones

La Base de Asignaciones está compuesta por información de:

- Asignaciones a Cursos
- Asignaciones a Exámenes
- Trabajo Efectivo en Cursos
- Trabajo Efectivo en Exámenes

La diferencia entre Asignación y Trabajo Efectivo corresponde a que en las asignaciones se tiene en cuenta el tiempo planificado para una materia -antes de dictar el curso o tomar el examen-, mientras que en el trabajo efectivo se considera la comunicación de cantidad de horas dedicadas a la tarea -luego de dictado el curso o tomado el examen.

[2] Se tiene disponible información con el mismo formato para las carreras 60 y 61. Ésta no pudo ser cargada por razones de espacio.

[3] Hasta marzo de 1997, había aproximadamente 44.000 registros de exámenes, 12.300 registros de cursos y 800 registros de inscripciones(que corresponden a 4 materias).

Debido a la falta de estructuración de la información disponible, se diseñó una base específica para este tema, y se creó un programa de carga para la misma.

4.2.2.1. Tabla ASIGNA

Esta tabla contiene asignaciones y trabajos efectivos en cursos y exámenes para los profesores del In.Co. En esta tabla se detalla el año y período al que corresponde la asignación, y los docentes que dictaron la materia. A su vez se separan las horas trabajadas según el tipo de trabajo realizado. De esta manera se puede saber por ejemplo: la cantidad de horas dedicadas a teórico y a práctico, a preparación de exámen y a la muestra.

4.2.3. Base de Presupuesto

La Base de Presupuesto está compuesta por información de:

- Docentes del Instituto
- Presupuesto de esos docentes (grado, horas y cargo)
- Cambios de Presupuesto

En este sentido hay que destacar dos grandes áreas:

- Presupuesto. Es información que llega todos los meses del Departamento de Personal, con la información de docentes, grados y horas. A esta área corresponden las tablas Sueldos.dbf e Instiaux.dbf.
- Cambios. Esta información a diferencia de la anterior no es periódica, ni está estructurada. Proviene de Resoluciones del Consejo aprobando cambios en el Presupuesto (Renuncias, Ingresos, Ampliaciones o Reducciones horarias). Para esta área fue creada especialmente una base en Access.

4.2.3.1. Tabla Sueldos.dbf

Esta tabla contiene información sobre los sueldos de los docentes en función de su grado y su carga horaria. Se tienen los sueldos para los Grados 1 al Grado 5, desde 4 hasta 42 horas semanales.

De esta tabla no interesará el sueldo en sí, sino obtener una matriz (que es prácticamente invariante) que relacione los grado-horas con un valor base (sueldo de un Grado 1 – 10 hs).

4.2.3.2. Tabla Instiaux.dbf

Esta tabla contiene información sobre el grado y horas presupuestadas para cada docente del Instituto de Computación. La información está estructurada en torno al concepto de número de cargo. Para cada cargo se tiene el nombre, apellido, grado y horas del docente. Hay que destacar que un docente puede tener varios cargos, y a su vez dentro de un mes, un cargo puede pertenecer a distintos docentes (si bien no en el mismo instante). Esto hace imposible conocer todo el trabajo de un docente que tenga más de un cargo.

4.2.3.3. Tabla Cambios - Access

Esta tabla contiene información sobre cambios en el presupuesto del Instituto. Estos cambios se deben a Resoluciones del Consejo por las que se aprueban: renuncias, ingresos, ampliación o reducción de horas.

Debido a que no estaban bien especificados los requerimientos operativos sobre esta tabla, fue creada en una base Access, de forma de que el usuario pudiera fácilmente agregar campos que le resultaran de utilidad, aunque no tuvieran significación para el Data Warehouse. La información mínima que se consideró útil para el Data Warehouse consiste en el identificador de la Resolución del Consejo, la fecha de la misma y los docentes que cambian de grado o de horas, el grado y horas originales y finales del docente, además del motivo por el que se realizó el cambio (por ejemplo: renuncia o aumento de horas).

4.3. Bases de Datos Limpias

El nivel de bases de datos limpias, cómo se puede observar en la Figura 4.1 al inicio del capítulo, es un nivel intermedio entre las bases de datos fuentes y el Data Warehouse.

Este nivel intermedio facilita el proceso de carga del Data Warehouse permitiendo agregar nueva información o limpiar en varios pasos la información existente.

4.3.1.1. Docentes

Tiene los docentes del Instituto de Computación. En esta tabla se crea el identificador de docente que luego será usado en el Data Warehouse. Además se mantienen todos los nombres con los que se identifica a un docente, indicando cual es el nombre que se debe usar. De esta manera si el archivo de docentes viene con un nombre que ya había sido usado, el programa marcará que son equivalentes sin necesitar ayuda del usuario.

Utilidad: Un problema asociado a la tabla fuente Instiaux.dbf es que la información que contiene son los grados, horas y servicio de un cargo. Como los docentes pueden tener varios cargos, no hay forma de relacionar los distintos cargos de un mismo docente. Inclusive, los campos Nombre y Apellido no son de utilidad para realizar un matcheo de caracteres estricto, pues no siempre traen los mismos valores. En este caso el proceso más lógico consiste en realizar una carga semi-automatizada, que con interacción del usuario permita determinar fácilmente los cargos que corresponden a un docente, identificándolo correctamente.

4.3.1.2. Presupuesto

Tiene el grado, horas y servicio de cada docente. Es obtenida de la Sección Personal de la Facultad.

Utilidad: Esta tabla y la tabla anterior (Docentes) corresponden a la tabla Instiaux original que tiene un diseño completamente denormalizado. El diseño en este caso se normalizó para su carga en el Data Warehouse.

4.3.1.3. Tipos_materias

En esta tabla se cargan los tipos de materia (Curso, Taller o Electiva) para las materias nuevas.

Utilidad: En los estudios sobre los Data Marts de actividades resulta importante poder tener una jerarquía por tipo de materia, la que no se dispone en las tablas originales.[4] En esta tabla se cargan automáticamente el nombre y código de las materias nuevas que llegaron de Bedelía, y el usuario debe ingresar el tipo de la misma (**C**urso, **E**lectiva o **T**aller).

[4] Esta jerarquía permite mejorar el conocimiento de los resultados (por ejemplo: los talleres tienen un porcentaje de aprobación mucho mayor que las materias comunes).

4.3.1.4. Cambios

Tiene los cambios de presupuesto (Renuncias, reducción y ampliación de horas) para los docentes del In.Co. Además, se crea un identificador de cambio que será útil luego para poder determinar que parte de las horas ya han sido asignadas y cuáles quedan por asignar. En esta tabla, se calcula la cantidad de horas que aumenta el presupuesto a partir de los grados y horas origen y finales del docente, usando la tabla de equivalencias.

Utilidad: Las renuncias o reducciones de horas dejan disponible una porción del presupuesto. A los efectos de usar esta porción del presupuesto para un ingreso o aumento de horas, se debe tener una clara idea de que horas fueron reasignadas y cuáles aún queda por reasignar. Esta tabla nos permite tener porcentajes de reasignación de horas asociados a un cambio.

4.4. El Data Warehouse

El Data Warehouse construído se encuentra en su totalidad en una base de datos Oracle, en un ambiente centralizado.

Si bien en los últimos años ha habido un gran movimiento hacia los Data Warehouses distribuídos, o replicados, en el caso que nos ocupa este no fue un punto que mereciera estudio, debido a que la información sólo interesaba a un Instituto (el In.Co.), y no había dentro de éste grandes distancias o diferencias de uso que ameritaran otro entorno que no fuera centralizado. No obstante ello, consideramos que en el caso de que se ampliara el sistema a otros Institutos, podría ser un punto interesante para estudiar.

Diseño del Data Warehouse

El diseño de tablas del Data Warehouse y la estructura de Joins se puede ver en la Figura 4.2.

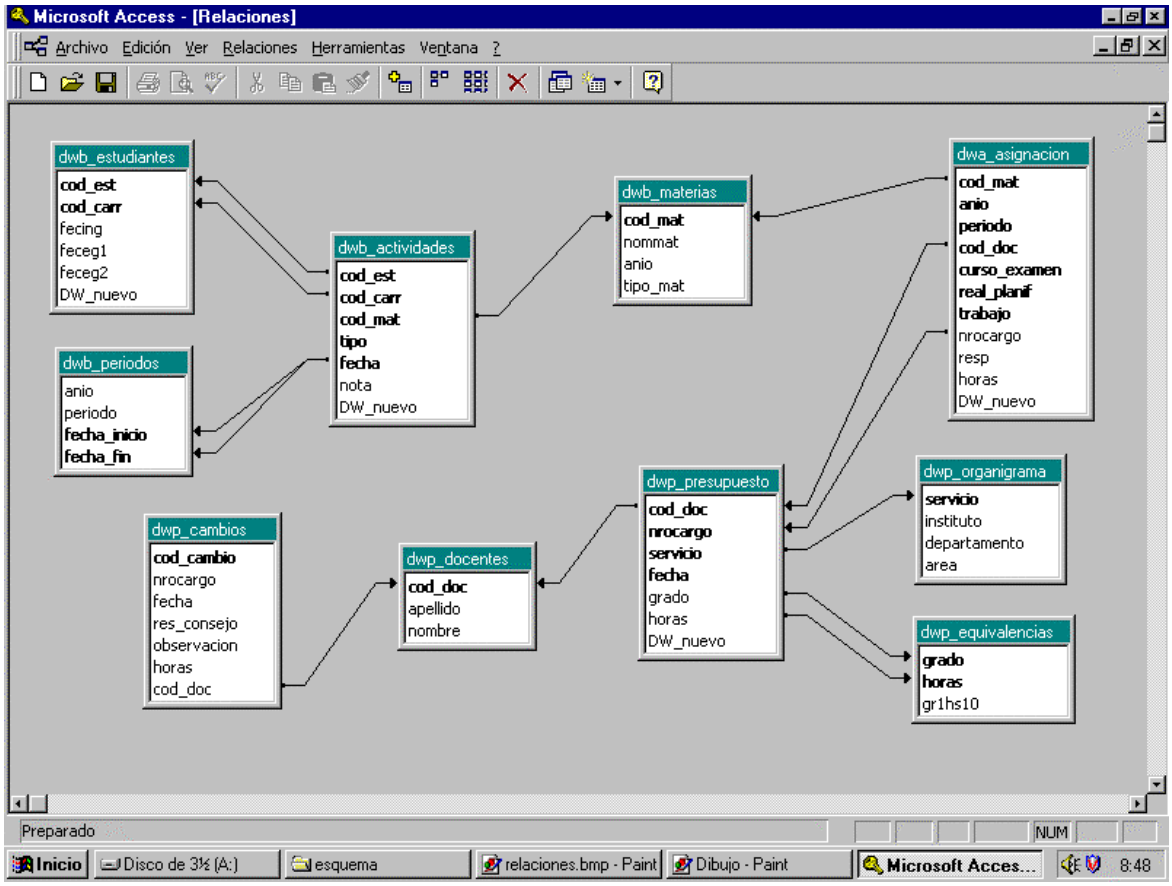


Figura 4.2 – Diseño del Data Warehouse

En esta figura se puede observar en los recuadros cada una de las tablas del Data Warehouse. El nombre de la tabla se encuentra en la parte superior del cuadro en texto inverso. En el cuerpo del cuadro se encuentran los atributos de la tabla, destacándose en negrita la clave de la misma. Las flechas que unen dos cuadros representan los joins definidos entre las tablas (unen los atributos que joinean). El sentido de las flechas es de la tabla foránea a la tabla origen del atributo.

Por ejemplo: La tabla DWP_DOCENTES que se encuentra en la parte inferior tiene como atributos cod_doc (clave), nombre y apellido. Tiene join definido por el atributo cod_doc con las tablas DWP_PRESUPUESTO y DWP_CAMBIOS.

El contenido de las tablas del Data Warehouse se describe a continuación.

4.4.1. DWB_ACTIVIDADES

Tiene cursos y exámenes dados por estudiantes de las carreras 70 y 71. Para estos se sabe la fecha y la nota obtenida.

Para llegar a esta tabla a partir de su base fuente (ACTIVIDADES), se realizan chequeos de los datos y se eliminan las inscripciones que no integran el Data Warehouse (pero sí están en la base fuente).

4.4.2. DWB_ESTUDIANTES

Tiene datos sobre los estudiantes de carreras 60, 61, 70 y 71. De cada estudiante se sabe a que carrera está inscripto (pueden ser varias), el año de ingreso en Facultad, y el año en que obtuvo los títulos (en caso de haberse recibido de Analista o Ingeniero).

Para llegar a esta tabla a partir de su base fuente (INSCARR), se realizan chequeos de los datos y se denormaliza la misma, de forma de tener para un estudiante-carrera las fechas de egreso a las distintas etapas en el mismo registro (hay 2 etapas: Analista e Ingeniero). Esto permite mejorar significativamente la performance de las consultas de carga del cubo.

4.4.3. DWB_MATERIAS

Para todas las materias de las carreras 60, 61, 70 y 61 se dispone del nombre, el año en que se dicta (relativo a la carrera) y el tipo de la misma (puede ser Electiva, Taller o Común).

A diferencia de la tabla de producción a partir de la cual se carga (MATCARR), no se consideran diferentes dos materias con el mismo código pertenecientes a distintas carreras.

4.4.4. DWB_PERIODOS

Tiene un registro por período de exámen, indicando fecha de inicio y fecha de fin del mismo. De esta forma se pueden responder a los requerimientos de comparar resultados en período de Diciembre contra resultados en Febrero, lo que de otra manera sería imposible debido a que en DWB_ACTIVIDADES sólo se tienen las fechas de los exámenes, no a que período corresponden.

Esta tabla es cargada manualmente. Los datos se obtienen de las Resoluciones del Consejo de Facultad que marcan fechas iniciales y finales de los períodos de exámen.

4.4.5. DWA_ASIGNACION

Tiene las asignaciones y trabajos efectivos de los profesores del In.Co., y las horas que se planifican (o las horas que tomó si son trabajos efectivos) para cada tipo de trabajo. Los tipos de trabajo son por ejemplo: preparación de curso, horas teórico, horas práctico.

4.4.6. DWP_ORGANIGRAMA

Tiene información sobre el Organigrama del Instituto de Computación. (Instituto, Departamento y Area). Tiene previsto la ampliación a Organigrama de Facultad.

4.4.7. DWP_EQUIVALENCIAS

Esta tabla tiene las equivalencias en grados 1 – 10 horas para todos los grados y horas posibles. Esto permite poder comparar los trabajos de profesores que tienen distintas cargas horarias o grados. Además permite resolver consultas que son útiles al evaluar un cambio en el presupuesto, por ejemplo: ¿si renuncia un grado 3-20 horas, cuántos grados 1 de 10 horas se pueden contratar?.

4.4.8. DWP_CAMBIOS

Tiene los cambios de presupuesto (Renuncias, reducción y ampliación de horas) para los docentes del InCo. La información que se dispone es de: Fecha y resolución del Consejo que aprobó el cambio, docentes que tuvieron cambios en presupuesto, y cantidad de horas de aumento en el presupuesto.

4.5. Carga y Control de Calidad

4.5.1. Introducción

La utilidad del Data Warehouse radica en que la información que presenta está en un formato consistente y pasó varios chequeos de errores. Por esta razón, el proceso de carga y control de calidad es el que toma más tiempo dentro de la construcción de un Data Warehouse. Según estimaciones realizadas por William Inmon en [WI-93], Ralph Kimball en [RK-96] y otros expertos en Data Warehousing, este proceso puede llegar a insumir entre un 70 y un 90% del tiempo de desarrollo.

En este proceso se deben definir varios puntos importantes, como ser:

- Extracción.
- Limpieza de datos.
- Transformación.
- Carga en el Data Warehouse.

4.5.2. Proceso General de Carga

Si bien cada base fuente tiene sus particularidades, se puede establecer un formato general para el proceso de carga. Este proceso consiste en los siguientes puntos:

1. Encontrar los registros que cambiaron.
2. Pasaje al Data Warehouse de los registros sin errores.
3. Pasaje a tablas de error de los registros que no cumplen control de calidad.
4. Publicar resultados (informar resultados y errores al DWA).
5. Arreglo de registros con error.
6. Vuelta al Punto 2.

4.5.2.1. Encontrar los registros que cambiaron

Se decidió que la carga del Data Warehouse se realice en forma incremental. Por ello se debe tener una forma de identificar los registros que cambiaron.

Esta tarea puede ser sencilla cuando – como en el caso de Asignaciones – se pueden modificar los programas para agregar un campo de registro modificado, o puede ser compleja – como en el caso de la base de Bedelía – cuando sólo se dispone de una copia de la base y no se pueden usar características especiales del SGBD.

A continuación se describen las alternativas planteadas para resolver este punto, enfocado en la base de Bedelía (que como se decía más arriba, es la única que plantea un problema interesante).

4.5.2.1.1. Opciones para actualización de Bedelía

Completa

1) Actualización completa.

Idea: Cargar todos los registros que vienen. Previamente se borra lo existente en las tablas del Data Warehouse.

Ventaja: Es la más sencilla. No hay problemas de que se pueda perder algún registro en el proceso (en la opción 3, esto puede ocurrir).

Desventaja: Demasiado larga (siempre se cargan todos los registros).

Incremental (suponiendo por el momento que se tuviera acceso a base original de Bedelía)

2) Por triggers

Idea: Programar un trigger que al cambiar (ingresar, modificar o borrar) un registro, realice los chequeos y lo ingrese al Data Warehouse con las conversiones necesarias.

Ventaja: Usa los mecanismos del SGBD. No se debe realizar un proceso de carga manual.

Desventaja: Al realizarse la actualización en tiempo real, se hace perder al DW la no volatilidad, los reportes van a cambiar muchas veces por día, no se va a saber hasta que datos están cargados en el Data Warehouse, y cuales no. Además, esta solución interfiere con el sistema de producción enlenteciéndolo.

3) *Por fecha de actividad o ingreso*

Idea: Cuando se corre el proceso de carga, cargar los registros de fecha X hasta fecha Y. (en la siguiente carga será de Y+1 hasta Z, y sucesivamente).

Ventaja: (Si se tienen los cuidados necesarios para que funcione) Se puede saber exactamente hasta que fecha se tiene ingresado en el Data Warehouse.

Desventaja: Es peligroso, pues la fecha en que la actividad ingresa a la base fuente (luego de que se reciben las Actas) no es la misma en que se dio, por lo que no tenemos una idea correcta de lo que se actualizó.

4) *Por timestamp*

Idea: Cada registro de la tabla de producción tiene un indicador que marca si ya está cargado en el Data Warehouse[5]. Cuando cargamos el DW, borramos todos los indicadores, los cuales se setean al cargar o modificar. En la actualización sólo tenemos en cuenta los registros que estén marcados como no ingresados al Data Warehouse.

Ventaja: Esta solución mantiene la consistencia de la base, puesto que no hay problemas de duplicar o saltearse un registro, y a la vez no sufre de las desventajas asociadas a la primera solución (cargar siempre todos los registros).

[5] Este indicador puede ser simplemente un bit que diga Si/No, o ser más complejo como un timestamp que diga la última vez que fue modificado, lo que luego se comparará con la fecha de actualización del Data Warehouse.

De estas opciones, la 4ª opción (por timestamp) creemos que es la mejor solución para este caso, porque no tiene los problemas de performance que afectan a la 1ª, no afecta al sistema de producción en gran medida (como la 2ª) y no hay que establecer mecanismos complicados para asegurar la consistencia como en la 3ª opción.

4.5.2.1.2. Forma de implementarla

Como no se tiene acceso a la base original de Bedelía, las opciones de carga incremental (entre ellas la 4ª) no son momentáneamente posibles.

Sin embargo, debido a que pensamos que la anterior es la mejor solución, se puede trabajar con esta opción, imitando el comportamiento que haría el timestamp. Para ello se utilizó el siguiente método:

- Se mantienen 2 versiones de Bedelía (la que teníamos **(1)**, y la que llegó **(2)**).
- Se alteran las tablas de **(2)**, agregandole un campo **modificado**.
- Recorriendo **(2)**, se consulta en **(1)** si está el registro. Si estaba tal cual, no se hace nada (**modificado** queda en NULL). Si no estaba, se pone **modificado** en ´S´
- Se elimina versión **(1)** y se renombra la versión **(2)** como versión **(1)**.
- De aquí en adelante se puede trabajar con la versión **(1)**, teniendo en cuenta que los registros que cambiaron son los que tienen **modificado='S'**.

Si bien este proceso es caro en tiempo de procesamiento, no afecta en gran medida al proceso de actualización, y el gran beneficio es que si se llega a disponer de acceso a la base original de Bedelía y se puede agregar un campo modificado que actualice el SGBD, sólo hay que eliminar este paso y el resto del proceso de carga no se debe tocar. (Como nota aparte, se destaca que para poder eliminar este paso más fácilmente, se corre siempre en scripts separados de la carga principal).

4.5.2.2. Pasaje al Data Warehouse

Una vez que se tienen los registros nuevos, hay que cargarlos en el Data Warehouse. En este proceso se cargan en el Data Warehouse los registros que pasan los chequeos de calidad, y se cargan en las tablas de errores, los registros que no los pasan.

4.5.2.3. Publicar resultados

Es inútil que los datos estén en el Data Warehouse, si nadie se entera que están, por lo que el último paso de la carga es publicar los resultados, informando al DWA (Data Warehouse Administrator) de que la carga se realizó con éxito y más importante aún, cuales son los registros que no pasaron los controles de calidad.

Este último punto (avisar que hay registros que no pasaron controles de calidad), pensamos que es de una gran utilidad, aunque no hayamos encontrado muchas referencias al mismo en la bibliografía. Con este informe, el DWA podrá solicitar más información a los responsables de las distintas bases para averiguar que es lo que falló. Es así que el DWA tendrá más rápidamente información de los usuarios sobre que era lo que debería haber tenido el registro o si hay que cambiar los chequeos.

A su vez para los usuarios es de utilidad saber que los registros con error no quedan perdidos en tablas de error, sino que mediante un informe van a ser investigados para poder integrar el Data Warehouse.

4.5.2.4. Pasaje al Data Warehouse de registros de las tablas de error

La corrección de errores plantea uno de los problemas más complejos en el diseño de un Data Warehouse.

Se tienen dos grandes opciones:

- No aceptar ninguna modificación hasta que todos los errores hayan sido corregidos.
- Ingresar la información que no contenga errores y guardar la información con errores en otra tabla para ser corregida luego. Esto es corregido en la tabla de error y luego se corre un programa de ingreso que tome los datos de esta tabla.

La primera opción es de mejor estilo, pero en el caso particular de este sistema, se dificulta su implementación para la base de Bedelía, debido a que no se tiene la posibilidad de solicitar cambios en la base original. Es por esto que se implementó la segunda opción.

4.5.2.4.1. Manejo de los errores

En el sistema de Data Warehousing se crean 2 tablas de error por cada tabla original.

La primera de estas tablas, a la que llamaremos **tabla de error actual**, se ocupa de mantener los errores que se produjeron en la última carga. De esta forma, si hubo un error en el pasaje de la tabla X al Data Warehouse, el registro que no pasó los chequeos de errores será ingresado a la tabla de error actual[6]. En esta tabla el DWA ingresará los valores correctos de los registros que hayan tenido errores.

[6] La tabla de error actual, tiene el mismo formato que la tabla original, y además tiene un campo código de error que facilitará al DWA el saber que chequeo no fue pasado.

La segunda de estas tablas, a la que llamaremos **tabla de errores históricos**, tiene el histórico de todos los errores que se han producido en la carga desde la primera actualización realizada. Esta tabla será usada para no tener en cuenta los registros de la tabla de producción que ya habían sido ingresados al Data Warehouse por la vía de corrección de errores. La utilidad de estas tablas puede no ser fácil de ver en primera instancia, pero un ejemplo puede aclararlo.

> Ejemplo: Supongamos que se realiza una actualización de la tabla de Actividades. En la actualización se encuentra que el estudiante 1 dio un exámen de Lógica el 1/12/10. Este error será detectado, y el registro ingresará a las tablas de error actual e histórica. El error se informa al DWA y éste averigua que la fecha correcta era 1/12/90, lo corrige y realiza la carga de este registro. Con esto se termina la primera actualización. El Data Warehouse tiene toda la información existente, y está en estado consistente.
>
> Unos meses después, llega una nueva actualización en la que Bedelía ya corrigió el error anterior. El hecho de corregirlo hace que parezca un registro nuevo (es nuevo porque difiere del anterior). Se corre una nueva actualización y se va a ingresar al Data Warehouse que el estudiante 1 tiene un exámen dado de Lógica el 1/12/90. En este momento, nuestro Data Warehouse acaba de perder la consistencia; el Data Warehouse va a decir que el estudiante 1 dio dos veces Lógica el 1/12/90, lo cual es falso. Para esto sirven las tablas de error históricas, antes de realizar una carga en el Data Warehouse, se eliminan los registros que ya habían ingresado al Data Warehouse por la vía de correcciones de errores, o sea los registros de la tabla histórica de errores.

Proceso de corrección

Los errores como se decía más arriba, son cargados en las tablas de error actual e histórica.

El DWA, una vez que obtienen los valores válidos, los cambiará en la tabla de error *actual*.[7] Desde este momento ya se puede correr el pasaje al Data Warehouse de los registros de la tabla de error. Este pasaje realizará los chequeos de errores correspondientes, e ingresará los registros que las pasen, borrando el registro de la tabla de error actual; y dejará los registros que no pasen los chequeos de errores (los que volverán a enviarse al DWA). La tabla de error histórica no interviene en este proceso.

[7] Las ventajas de ingresarlo en la tabla de error actual y no en el Data Warehouse (lo que sería más sencillo) se basan en que luego de ingresado en la tabla de error actual, se chequeará nuevamente si tiene errores antes de ingresar al Data Warehouse, mientras que si se ingresara directamente esquivaría todos los chequeos de errores.

Un punto que merece especial atención es que hacer cuando llegue la siguiente actualización. Si no se presta la debida atención, se puede generar información repetida en el Data Warehouse[8]. Para eso se utiliza la tabla de errores históricos como se detalla a continuación.

Dado que hubo un error en una base, puede ocurrir que en la siguiente actualización:

- No se haya corregido el problema.
  - En este caso, la versión del sistema de producción coincidirá con su versión previa, por lo que será descartada por el script que encuentra los valores que cambiaron.
- Se haya corregido el problema
  - En este caso, este registro será descartado por la vía de marcar como no modificados todos los registros de la tabla de producción que están en la tabla de errores históricos.

4.5.3. Actualización de las tablas del Data Warehouse

4.5.3.1. Base de Bedelía

Para la base de Bedelía se maneja que van a llegar actualizaciones semestralmente, por lo que la actualización será semestral.[9] Los datos que se reciben de Bedelía no siempre van a llegar a una fecha prefijada, por lo que el pasaje no se puede realizar en forma automática un día determinado. Los pasos a seguir en este caso son los siguientes:

Paso 0: Obtener las tablas tal como están en su lugar de origen

Este paso es necesario puesto que no se dispone de una conexión con las bases fuentes de Bedelía. Se reciben las tablas (probablemente en formato export de Oracle). Se crean las tablas ACTIVIDADES, INSCARR, MATCARR e INSCUR.

Este paso es completamente manual.

[8] Esto ocurre debido a que si se corrige el error, luego será cargado en el Data Warehouse, si la siguiente actualización está corregida, será marcada como que hubo cambios y por lo tanto se intentará ingresarla nuevamente al Data Warehouse.

[9] En caso de que llegaran más actualizaciones, es útil el aumentar la cantidad de actualizaciones del sistema hasta una vez por período de exámen (más que eso no tiene mucho sentido práctico).

Paso 1: Encontrar los registros que cambiaron

También es necesario por no disponer de una conexión con las bases fuentes de Bedelía. En caso de disponer de ella, se podrían implementar timestamps o triggers para marcar los registros modificados.

Con este paso se logra que los siguientes sólo necesiten considerar la información que cambió y no toda la ingresada.

Este paso es automatizado con el script marca_incremental_Bedelia (script PL/SQL).

Paso 2: Actualizar las tablas del Data Warehouse

Una vez que se sabe cuales son los registros que cambiaron, se hacen controles de calidad y se ingresan a las tablas del Data Warehouse.

Este paso es automatizado con el script SQL actualiza_Bedelia.

4.5.3.2. Base de Asignaciones

La Base de Asignaciones tendrá actualizaciones al principio y fin de cada período lectivo y de cada período de exámenes (correspondiendo a asignaciones y trabajo efectivo de cursos y exámenes, respectivamente).

Los pasos a seguir en este caso son los siguientes:

Paso 1: Ingreso de los datos fuente

Los datos fuente serán cargados por los encargados del Instituto en realizar las asignaciones. Éstos informarán al DWA el momento en que se finalizó el ingreso.

Paso 2: Actualizar las tablas del Data Warehouse

Este paso realiza pequeños controles de calidad (la mayor parte de ellos se realiza en la carga de la base fuente), y carga las tablas del Data Warehouse.

Este paso es automatizado con el script SQL actualiza_Asignaciones.

4.5.3.3. Base de Presupuesto

La Base de Presupuesto tendrá actualizaciones mensuales para el área Presupuesto y actualizaciones eventuales en el área Cambios.

4.5.3.3.1. Área Presupuesto

Los pasos a seguir en este caso son los siguientes:

Paso 0: Se reciben datos de Personal

Los datos fuente son recibidos del Dpto. de Personal. Se revisa el formato observando si cambió la estructura. En caso que así fuera, se usa el archivo de parámetros del programa de carga para cambiar el formato de ingreso.[10]

Paso 1: Limpieza de datos

El usuario de la base de Presupuesto debe correr el programa de limpieza asistido. Hay que resaltar la importancia de que sea el propio usuario el que corra este programa, debido a que es quien está mejor informado para cambiar o aceptar las sugerencias del programa.

Paso 2: Actualizar las tablas del Data Warehouse

Este paso realiza pequeños controles de calidad (la mayor parte de ellos se realizan en el paso anterior), y carga las tablas del Data Warehouse.

Este paso es automatizado con el script SQL actualiza_Presupuesto.

4.5.3.3.2. Área Cambios

Los pasos a seguir en este caso son los siguientes:

Paso 0: Se ingresan las Resoluciones del Consejo

El usuario recibe Resoluciones del Consejo y las ingresa en la base Access. Cuando se desee se corre una macro de Access que graba estas Resoluciones en un archivo de texto, que se le entrega al DWA.

[10] Este paso está asociado al hecho de que siempre viene la misma información, aunque no siempre en el mismo formato.

Paso 1: Pasaje a Oracle

Se hace un pasaje a tablas de Oracle de los datos que había en el archivo. Esta tabla intermedia permite realizar más fácilmente los controles de error – como por ejemplo, los chequeos referenciales.

Este paso es completamente manual.

Paso 2: Actualizar las tablas del Data Warehouse

En este paso se deben realiza varios controles de calidad, y cargar los datos correctos al Data Warehouse.

Este paso es completamente manual.

4.6. Metadata

En este proyecto se creó y entregó la siguiente documentación como Metadata:

- Manuales de usuario (Ver informe adjunto al presente)
  - Manual de usuario del programa de Asignaciones
  - Manual de usuario del programa de Presupuesto
  - Manual de usuario de la base de Cambios
- Guías de procesos de carga (Ver informe adjunto al presente)
  - Guía del Proceso de carga de Bedelía
  - Guía del Proceso de carga de Asignaciones
  - Guía del Proceso de carga de Presupuesto
- Documentación de sistemas fuente (Ver Sección 4.2 y Apéndice B)
  - Documentación del sistema de Bedelía
  - Documentación del sistema de Asignaciones
  - Documentación del sistema de Presupuesto
- Sistema de Data Warehousing - Guía de Convenciones y Formato de la Documentación (Ver Apéndice C)
- Documentación del sistema de Data Warehousing (Ver Sección 4.4 y Apéndice B)
- Documentación de los Data Marts (Ver Sección 4.7)

4.7. Data Marts

Se realizaron 4 Data Marts:

- Actividades: Mide la cantidad de actividades (cursos y exámenes) de los estudiantes, así como los promedios de aprobación de materias.
- Estados: Mide el estado de los estudiantes en la carrera, señalando cantidad de estudiantes en cada etapa, ingresos y egresos.
- Asignación a cursos y exámenes: Mide la cantidad de horas asignadas contra las trabajadas de cada docente en cada curso y exámen que dictó.
- Presupuesto: Mide los sueldos de cada docente del instituto.

Estos Data Marts surgen como respuesta a los requerimientos mencionados en el capítulo 3.

Si bien algunos de esos requerimientos eran demasiado ambiciosos, se intentó resolverlos lo más completamente posible, dando un buen punto de partida para el estudio del problema, cuando no era posible dar una respuesta directa a partir de los datos. Como ejemplo de esto, se puede citar el requerimiento de "construir perfiles de estudiantes", en el que si bien no se disponía de la información necesaria para hacerlo, se le dio al usuario datos suficientes de los estados de los estudiantes en la carrera, como para que pueda construir dichos perfiles.

A continuación se estudiará en profundidad cada Data Mart, detallando el esquema multidimensional, dimensiones, medidas reportes y proceso de carga. Las consultas de carga de los mismos se detallan en el Apéndice D.

En los esquemas se utilizará la notación propuesta por Maio, Golfarelli y Rizzi en [DM-98]. Se puede ver una descripción de la misma en la sección 4.7.1.1 (a continuación del esquema de Actividades).

Para el diseño de los Data Marts se utilizó la herramienta *PowerPlay Transformer* de *Cognos*. En la Figura 4.3 se muestra el ambiente de desarrollo utilizado. Allí se definen las dimensiones, las medidas, las consultas de carga y los cubos multidimensionales.

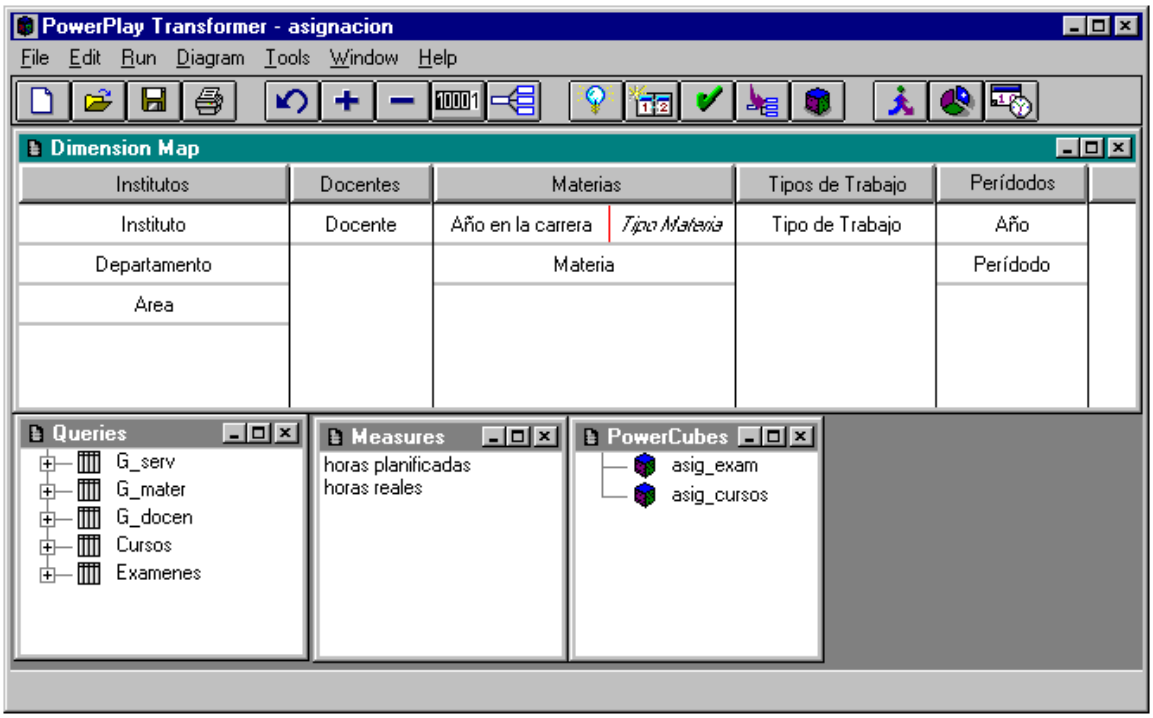


Figura 4.3 – Ambiente de desarrollo de los Data Mart

4.7.1. Actividades

Se resolvieron los siguientes requerimientos:

(1) ¿Qué gente se presenta y cómo prever que gente se va a presentar a un examen?: El objetivo es pronosticar cuántos estudiantes se van a presentar a un examen, y de que carrera son. Para ello se analizan datos históricos, buscando identificar que estudiantes se presentan en cada período.

(2) ¿Cómo construir un perfil de carrera?: El objetivo final era construir perfiles de estudiantes, para luego clasificarlos de acuerdo a su historia y poder predecir su comportamiento, como primer paso para explicar algunos fenómenos, como las deserciones. El usuario no cuenta con datos suficientes como para definir cada perfil. Para poder definirlos y luego construirlos, se analizará la historia de cada estudiante.

4.7.1.1. Esquema Multidimensional

El esquema multidimensional se puede ver en Figura 4.4.

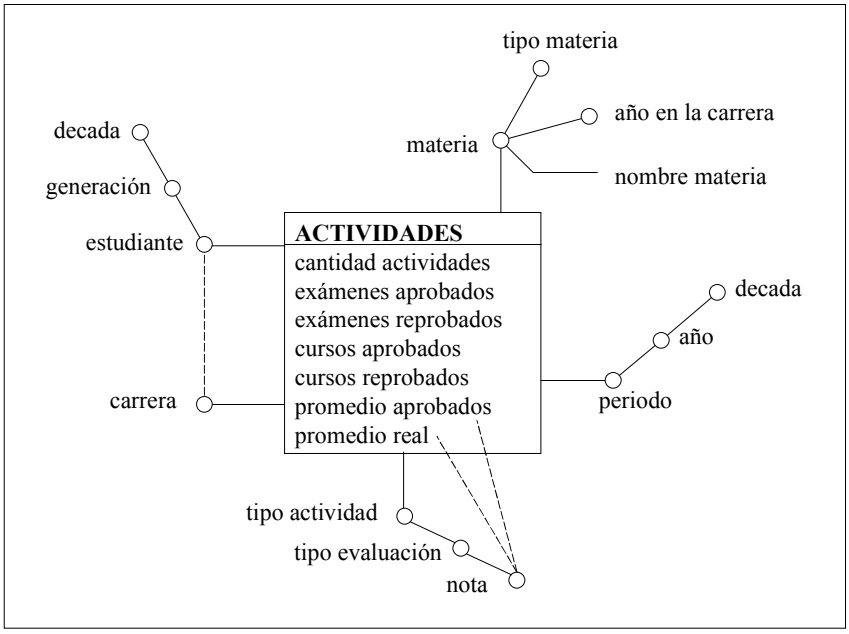


Figura 4.4 - Esquema Multidimensional de Actividades

El cuadro central está dividido en dos partes: el nombre del esquema y las medidas. En este caso el esquema se llama *actividades*, y tiene 7 medidas.

Del cuadro central cuelgan las dimensiones. Cada nivel en la jerarquía de una dimensión está representado por un círculo, y se unen entre sí por líneas rectas. Por ejemplo la dimensión *estudiantes* tiene 3 niveles: *estudiante*, *generación* y *década*. El nivel de granularidad más bajo es el que se encuentra más próximo al cuadro central.

Las jerarquías alternativas quedan representadas como bifurcaciones a partir de un mismo círculo. En el caso de la dimensión *materias*, se tienen jerarquías alternativas: *por tipo de materia,* y *por año en la carrera.*

Una dimensión puede tener otro tipo de atributos que no sean niveles en la jerarquía. Esos atributos son meramente descriptivos, y se llaman atributos no dimensionales. Se los representa con una línea quebrada que sale del nivel de la dimensión al que están asociados. La dimensión *materias* tiene un atributo no dimensional: *nombre materia*, asociado a *materia.*

Dos niveles de distintas dimensiones, o una dimensión y una medida pueden estar relacionados entre sí. Dicha correlación se nota uniéndolos con una línea punteada. En este caso, *estudiante* y *carrera* están relacionados, también lo están *nota* y los *promedios*.

4.7.1.2. Dimensiones

Se diseñaron 5 dimensiones. Las mismas se detallan desde el nivel de granularidad más bajo al más alto.

- Estudiantes:
  - Estudiante: Identificador de un estudiante, compuesto por su numero de estudiante, y número de la carrera a la que está inscripto. Un estudiante inscripto a más de una carrera figurará varias veces, una por cada carrera (con identificadores diferentes).
  - Generación: Año de inscripción del estudiante a la carrera.
  - Decada: Agrupamiento de la generación cada 10 años.
- Carreras:
  - Carrera: Identificador de la carrera. Hay una relación de dependencia entre el identificador de estudiante y la carrera.
- Materias:
  - *Jerarquía principal:*
    - Materia: Identificador de materia. Es un código no relacionado con la carrera.
    - Año en la carrera: Año relativo a la carrera, en que se dicta la materia (1º a 5º).
  - *Jerarquía alternativa:*
    - Materia: Identificador de materia. Es un código no relacionado con la carrera.
    - Tipo materia: Clasificación de las materias en: "común", "electiva" y "taller".
  - *Atributos no dimensionales:*
    - Nombre materia: Nombre descriptivo de la materia.
- Períodos:
  - Período: Período en que se registró la actividad. Es un número (1 al 5), correspondiente a los períodos de febrero, marzo, julio, agosto, y diciembre, respectivamente. Esto es independiente del mes en que haya ocurrido efectivamente la actividad.
  - Año: Año en que se registró la actividad.
  - Década: Agrupamiento de los años en grupos de a 10.

- Tipos de actividad:
  - Nota: Valor del 0 al 12. Es relevante sólo si el tipo de aprobación es con nota.
  - Tipo de aprobación: Subclasificación de los cursos y exámenes. Toma dos valores "Con nota" y "Sin nota".
  - Tipo actividad: Clasificación de las actividades en "exámen" o "curso".

Un diagrama de la dimensión Materias y sus dos jerarquías, puede verse en la Figura 4.5. Las categorías del nivel *materia* no se muestran en el diagrama, porque son muchas.

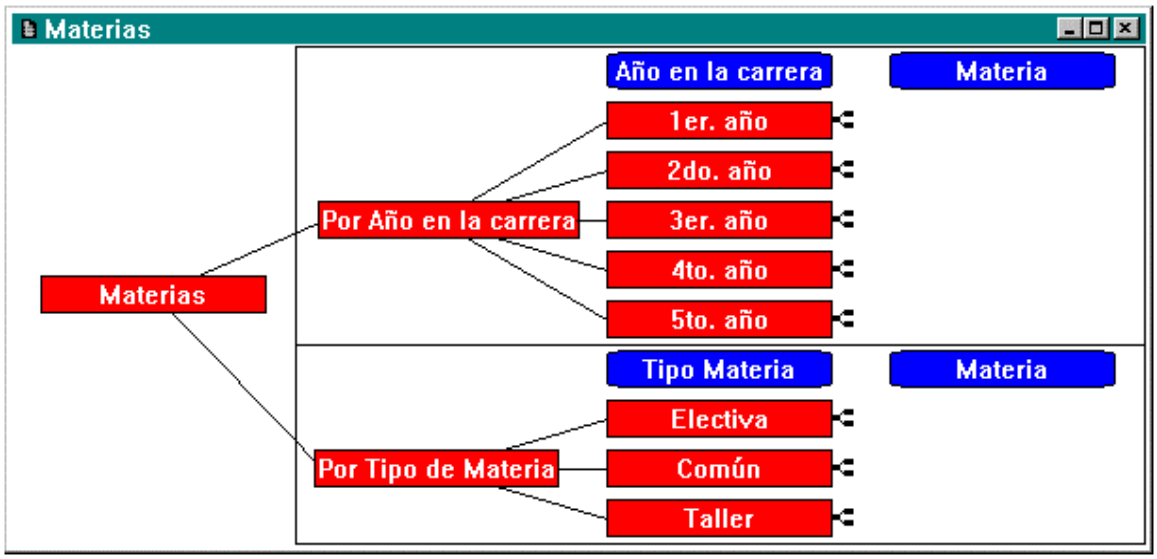


Figura 4.5 - La dimensión Materias

4.7.1.3. Medidas

- Cantidad de actividades: Cuenta de las actividades generadas, tanto cursos como exámenes.
  - *Granularidad más baja:* "1".
  - *Función de agregación:* suma.
- Exámenes aprobados: Cuenta de los exámenes aprobados.
  - *Granularidad más baja: "1" si la actividad es un exámen, y fue aprobado, o "0" en caso contrario.*
  - *Función de agregación:* suma.
  - *Restricciones: No se aplica cuando el tipo de actividad es un curso.*

- Exámenes reprobados: Cuenta de los exámenes reprobados.
  - *Granularidad más baja: "1" si la actividad es un exámen, y no fue aprobado, o "0" en caso contrario.*
  - *Función de agregación:* suma.
  - *Restricciones: No se aplica cuando el tipo de actividad es un curso.*
- Cursos aprobados: Cuenta de los cursos aprobados.
  - *Granularidad más baja: "1" si la actividad es un curso, y fue aprobado, o "0" en caso contrario.*
  - *Función de agregación:* suma.
  - *Restricciones: No se aplica cuando el tipo de actividad es un exámen.*
- Cursos reprobados: Cuenta de los cursos reprobados.
  - *Granularidad más baja: "1" si la actividad es un curso, y no fue aprobado, o "0" en caso contrario.*
  - *Función de agregación:* suma.
  - *Restricciones: No se aplica cuando el tipo de actividad es un exámen.*
- Promedio aprobados: Promedio de aprobación de exámenes, calculado sobre los exámenes aprobados. (suma de las notas sobre la cantidad de exámenes aprobados).
  - *Granularidad más baja:* nota del exámen.
  - *Función de agregación:* promedio.
  - *Restricciones: No se aplica cuando el tipo de actividad es un curso.*
- Promedio real: Promedio de aprobación de exámenes, calculado sobre todos los exámenes a los que se presentó. (suma de las notas sobre la cantidad de presentaciones).
  - *Granularidad más baja:* nota del exámen.
  - *Función de agregación:* promedio.
  - *Restricciones: No se aplica cuando el tipo de actividad es un curso.*

4.7.1.4. Vistas y Reportes

Se resolvieron los requerimientos solicitados:

(1) Cantidad de estudiantes que se presentaron al examen de una materia, por período; para pronosticar cuántos se presentarán en el futuro.

Se coloca en el eje horizontal los períodos, y en el display la cantidad de exámenes dados (medidas), discriminado en aprobados y reprobados. En las capas se colocan las materias.

En la Figura 4.6 se observa la *cantidad de exámenes aprobados y reprobados* en el período *decada 1990's* de la materia *lógica*, en formato de barras apiladas.

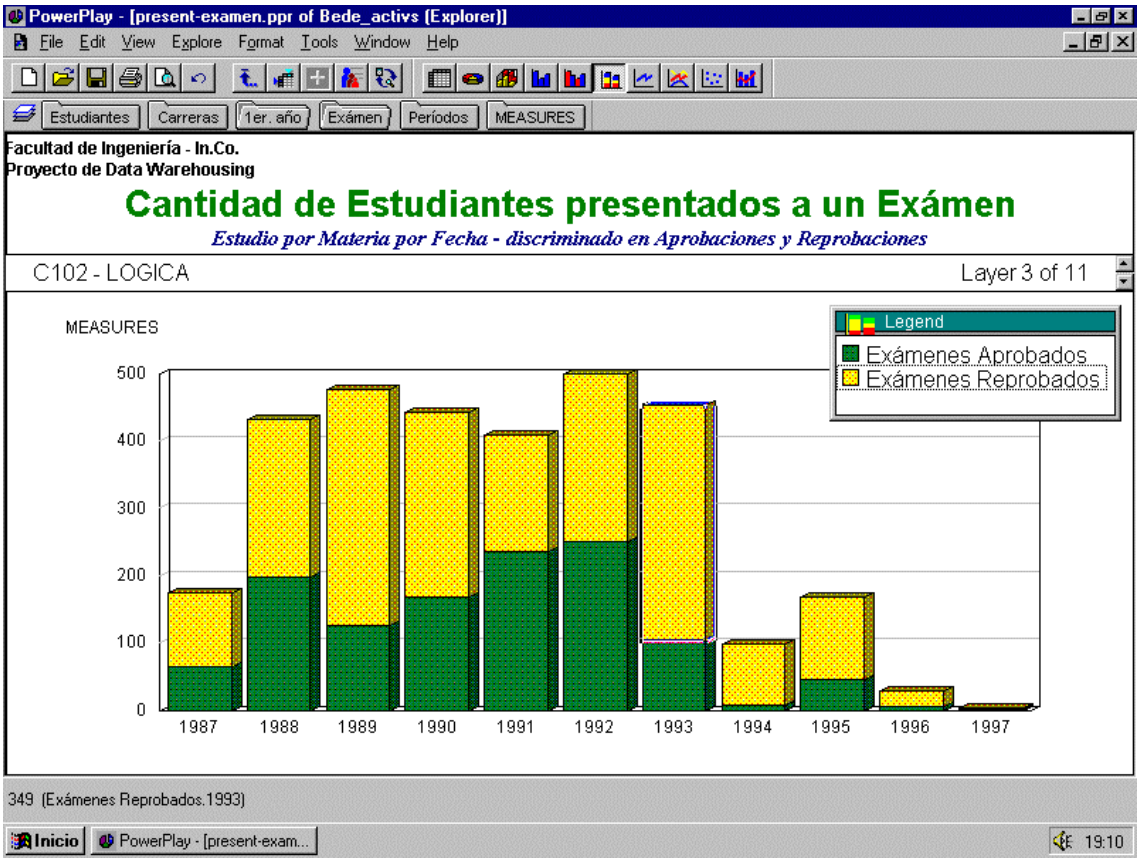


Figura 4.6 - Resolución del requerimiento (1) de Actividades

(2) Históricos de estudiantes para poder definir perfiles.

Se coloca en el eje horizontal los períodos, y en las líneas los estudiantes. En las capas se colocan las materias. Se mide la cantidad de actividades.

En la Figura 4.7 se observa la *cantidad de actividades* en el período *decada 1990's* de las materias *de 1er año,* para los estudiantes que ingresaron en la decada *1990's*, en formato multi-líneas.

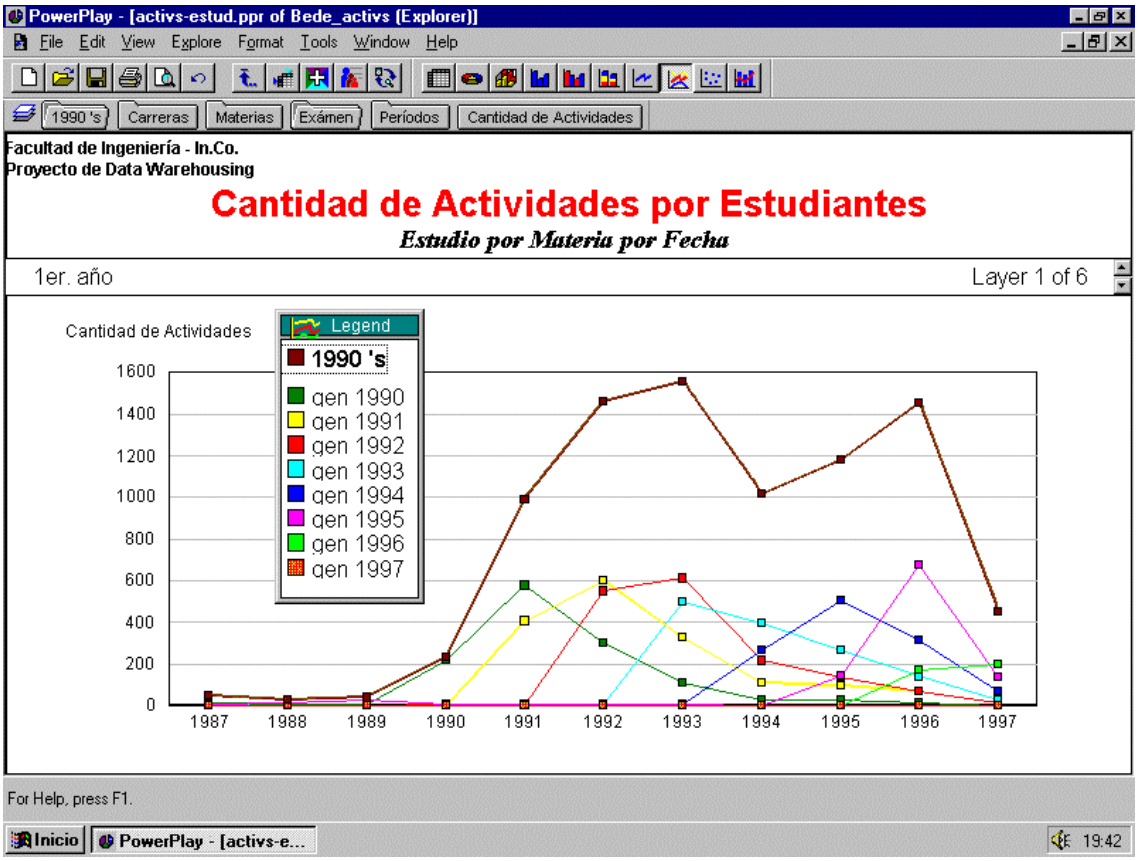


Figura 4.7 - Resolución del requerimiento (2) de Actividades

Se construyeron otros reportes que brindan conclusiones interesantes, aunque no estaban dentro de los principales requerimientos enunciados. Se citan como ejemplo:

- Porcentaje de Aprobación de Exámenes. Estudio por Materia por Fecha, comparando porcentajes por materia y sobre el total.
- Distribución de Aprobaciones según Generación. Estudio por Materia por Fecha, comparando porcentajes por materia y sobre el total.
- Evolución de la Cantidad de Actividades. Estudio por Estudiantes por Fechas.
- Cantidad de Exámenes Aprobados según Generaciones. Estudio por Períodos por Materia.
- Distribución de las Reprobaciones de Exámenes. Estudio por Estudiantes por Fechas.
- Correlación Promedio Real vs. Promedio Aprobaciones. Estudio por Materias y Estudiantes.

4.7.1.5. Carga

Se espera una frecuencia de carga semestral, pero no se sabe con anterioridad a que altura del semestre llegarán los datos. Es posible, también, que en el futuro los datos comiencen a llegar con más frecuencia, por ejemplo mensualmente.

El mecanismo de carga incremental es el que mejor se adapta a esta situación.

Se diseñó un sistema de carga manual del Data Mart; es decir, el DWA, o el operador responsable de la carga, debe invocar a un proceso que efectuará la misma.

Se provee también un proceso de carga total, por seguridad.

4.7.2. Estados

Se resolvieron los siguientes requerimientos:

(1) ¿Dónde está la gente en la carrera?: El objetivo es determinar la cantidad de estudiantes que hay en cada etapa de la carrera, cuántos de ellos están activos y cuantos no. Se definió que un estudiante está activo en un año, si éste registró alguna actividad durante ese año. Se definió que un estudiante hizo abandono de la carrera, si no registró actividades en los últimos 5 años. Se llamará no activos a los que no registraron actividades pero no han abandonado.

(2) ¿Cómo evaluar velocidad de avance?: Se quiere medir la velocidad de avance de los estudiantes en la carrera de acuerdo a determinados patrones.

(3) ¿Cómo calcular volumen de estudiantes activos?: El objetivo es prever cuántos estudiantes se van a presentar a un curso o examen, para poder destinar recursos. La observación de datos históricos es esencial para efectuar una proyección.

4.7.2.1. Esquema Multidimensional

El esquema multidimensional se puede ver en Figura 4.8.

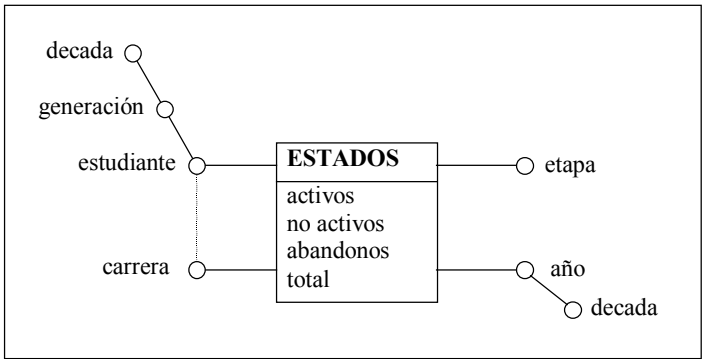


Figura 4.8 - Esquema Multidimensional de Estados

4.7.2.2. Dimensiones

Se diseñaron 4 dimensiones. Las mismas se detallan desde el nivel de granularidad más bajo al más alto.

- Estudiantes:
  - Estudiante: Identificador de un estudiante (Ver 4.7.1.2).
  - Generación: Año de inscripción del estudiante a la carrera.
  - Decada: Agrupamiento de la generación cada 10 años.
- Carreras:
  - Carrera: Identificador de la carrera. Hay una relación de dependencia entre el identificador de estudiante y la carrera.
- Años:
  - Año: Año en que se registró la actividad.
  - Década: Agrupamiento de los años en grupos de a 10.
- Etapas:
  - Etapa: Toma los siguientes valores: "ingreso", "egreso", "etapa1" o "etapa2", si en ese año, el estudiante ingresó, egresó, no ha obtenido el título de analista, o ya lo obtuvo, respectivamente. Son excluyentes. No se tienen en cuenta las actividades generadas en años anteriores al ingreso, y luego revalidadas.

4.7.2.3. Medidas

- Activos: Cuenta de los estudiantes activos.
  - *Granularidad más baja:* "1".
  - *Función de agregación:* suma.
- Abandonos: Cuenta de los estudiantes que abandonaron la carrera.
  - *Granularidad más baja:* "1".
  - *Función de agregación:* suma.
- Total: Total de estudiantes (activos, no activos y abandonos).
  - *Granularidad más baja*: "1".
  - *Función de agregación:* suma.
- No Activos: Cuenta de los estudiantes no activos.
  - *Función de cálculo: Total – Activos – Abandonos.*

4.7.2.4. Vistas y Reportes

Se resolvieron los requerimientos solicitados:

(1) Distribución de la cantidad de estudiantes en cada etapa de la carrera, discriminando activos y no activos.

Se coloca en el display las etapas y en las slices las medidas (activos, no activos, abandonos y total). En las capas se colocan las fechas.

En la Figura 4.9 se observa la cantidad de estudiantes *activos* en cada etapa, en el período *1995*, en formato de torta.

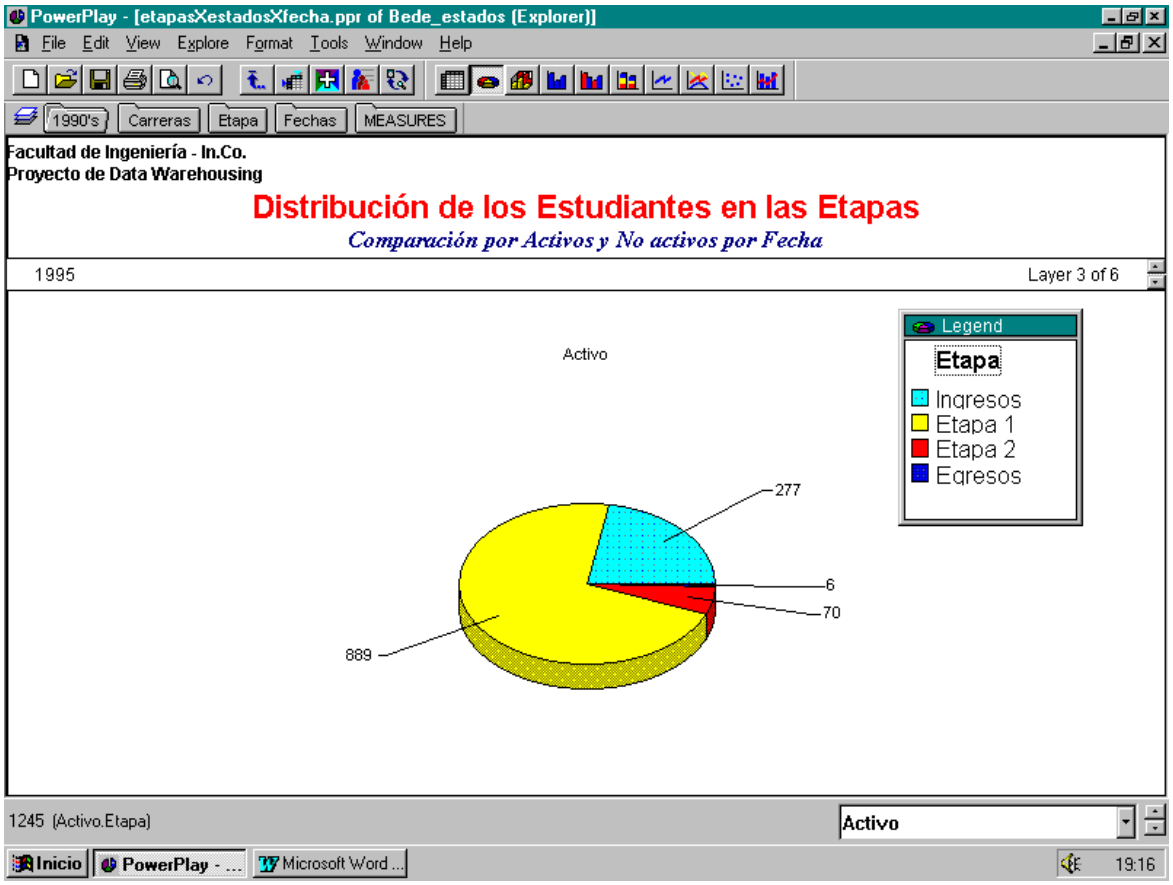


Figura 4.9 - Resolución del requerimiento (1) de Estados

(2) Velocidad de avance: Cantidad de estudiantes en cada etapa Se quiere medir la velocidad de avance de los estudiantes en la carrera de acuerdo a determinados patrones.

Se coloca en el eje horizontal los años y en el display las etapas. En las capas se colocan los estudiantes. Se mide la cantidad de estudiantes activos.

En la Figura 4.10 se observa la cantidad de *activos* en cada etapa, de los años *1993, 1994, 1995 y 1996,* de la generación *1990,* en dos formatos: barras apiladas y cuadro de valores.

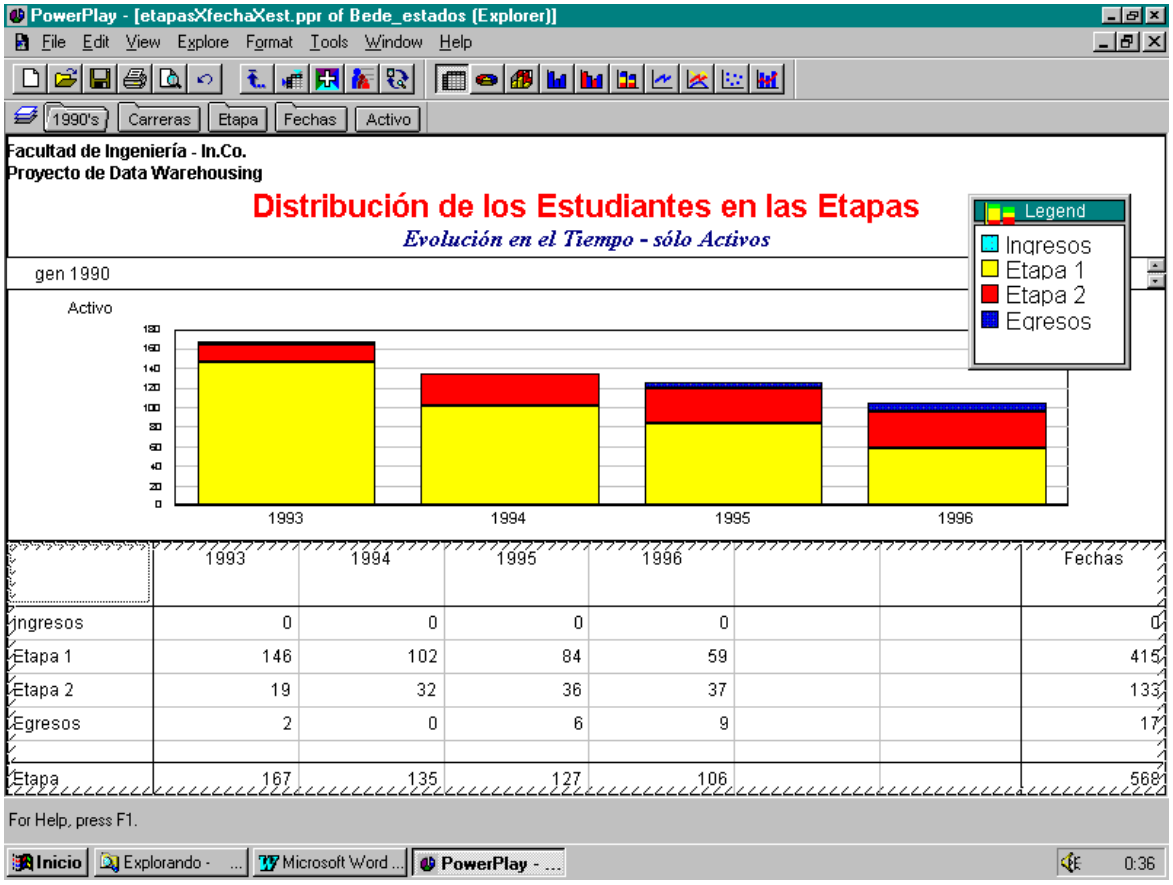


Figura 4.10 - Resolución del requerimiento (2) de Estados

(3) Evolución en el tiempo de la cantidad de estudiantes activos por etapa.

Se coloca en el eje horizontal los períodos y en las líneas las etapas. Se mide la cantidad de estudiantes activos.

En la Figura 4.11 se observa la cantidad de *activos* en cada etapa, en el período *1995*, en dos formatos: barras y cuadro de valores.

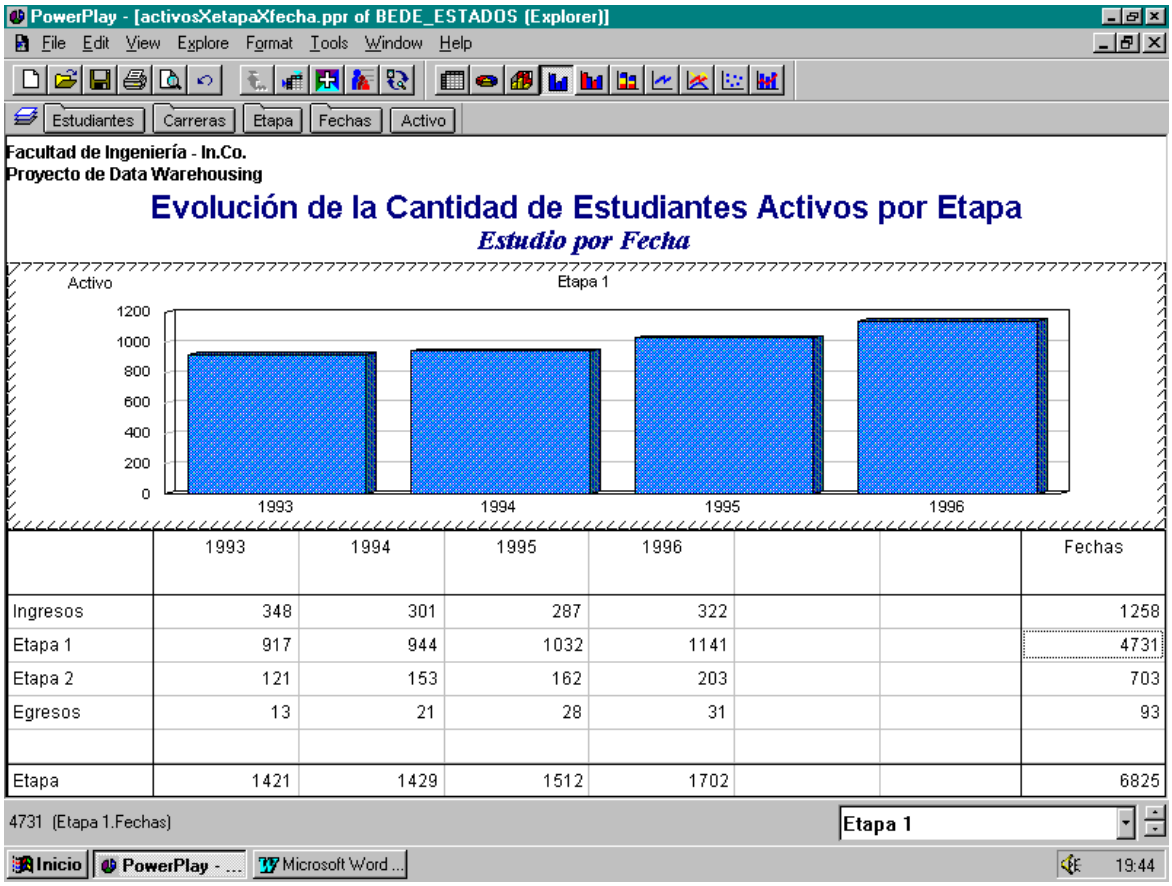


Figura 4.11 - Resolución del requerimiento (3) de Estados

4.7.2.5. Carga

Para poder evaluar si un estudiante en un año está activo o no, se necesita contar con las actividades de todo el año (se definió como activo a un estudiante que generó actividades durante ese año). Es por eso que sólo se cargarán años concluídos, y por tanto la frecuencia de carga debe ser anual.

Como ya se discutió en la sección 4.7.1.5, se espera la llegada de los datos con una frecuencia semestral, pero no se sabe exactamente a que altura del semestre llegarán. La carga se efectuará con una de las llegadas de datos, y se cargarán los datos correspondientes a los años terminados.

Es necesario que un operador ingrese como parámetro cuál es el último año concluído, y luego invoque al proceso de carga.

Como los resultados de algunos cursos y exámenes pueden retrasarse, es común que al siguiente semestre lleguen datos de años finalizados. Puede ocurrir que estudiantes catalogados como inactivos, estuvieran activos.

Esto hace que una carga incremental sea peligrosa desde el punto de vista de calidad de los datos. Como además la frecuencia de carga es muy baja (anual), no importa si el proceso es lento.

Se diseñó un sistema de carga total (no incremental) del Data Mart, dependiente de un parámetro: el último año finalizado. El DWA, o el operador responsable de la carga, debe invocar a un proceso que efectuará la misma.

4.7.3. Asignación

Se resolvieron los siguientes requerimientos:

(1) Planilla docente con grados, horas, cursos que dicta, y exámenes que toma. El objetivo es distribuir con coherencia a los docentes en los cursos y exámenes. Se utilizarán los datos actuales sobre grado y carga horaria, así como los históricos sobre los cursos y exámenes en los que trabajó.

(2) Medición del trabajo real. El objetivo es determinar que tanto se apartó la asignación del trabajo real. Para ello se compararán las horas trabajadas con las asignadas.

(3) Información de trabajo efectivo. El objetivo es asignar los recursos docentes de la mejor manera posible. Para ello se estudiarán los datos históricos de manera de optimizar la asignación y aprovechar mejor la carga horaria de cada docente.

Dado que las consultas se hacen para asignación a exámenes, o para asignación a cursos, se crearon 2 cubos multidimensionales, para exámenes y cursos respectivamente.

Como el tipo de análisis es el mismo, ambos cubos tendrán las mismas dimensiones y medidas, excepto la dimensión períodos, que para los exámenes consta de 5 períodos (febrero, marzo, julio, agosto y diciembre), y para los cursos consta de 2 semestres.

4.7.3.1. Esquema Multidimensional

El esquema multidimensional se puede ver en Figura 4.12.

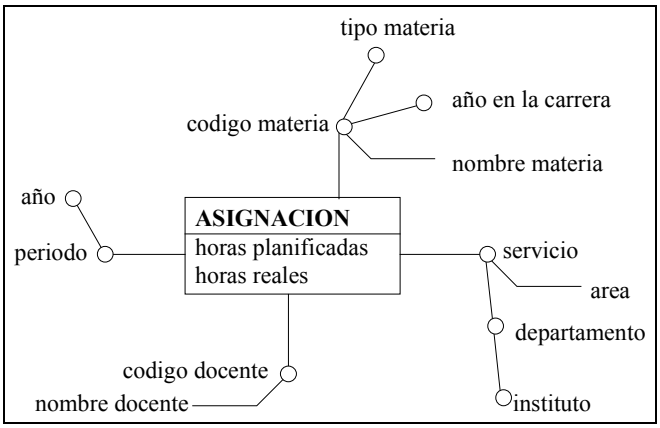


Figura 4.12 - Esquema Multidimensional de Asignación

4.7.3.2. Dimensiones

Se diseñaron 4 dimensiones. Las mismas se detallan desde el nivel de granularidad más bajo al más alto.

- Docentes:
  - *Jerarquía:*
    - Codigo docente: Código identificador único de un docente, independientemente del cargo que ocupe.
  - *Atributos no dimensionales:*
    - Nombre docente: Nombre del docente.
- Organigrama:
    - Servicio: un Código del área, dentro de un departamento.
    - Departamento: Código del departamento, dentro de un instituto.
    - Instituto: Código del instituto.
  - *Atributos no dimensionales:*
    - Area: Nombre descriptivo del área.
- Materias:
  - *Jerarquía principal:*
    - Materia: Identificador de materia. Es un código no relacionado con la carrera.
    - Año en la carrera: Año relativo a la carrera, en que se dicta la materia (1º a 5º).
  - *Jerarquía alternativa:*
    - Materia: Identificador de materia. Es un código no relacionado con la carrera.
    - Tipo materia: Clasificación de las materias en: “común”, “electiva” y “taller”.
  - *Atributos no dimensionales:*
    - Nombre materia: Nombre descriptivo de la materia.
- Períodos:
    - Período: Para el caso de exámenes es el ordinal del período. Es un número (1 al 5), correspondientes a los períodos de febrero, marzo, julio, agosto, y diciembre, respectivamente. Esto es independiente del mes en que se haya tomado efectivamente el exámen. Para el caso de curso es el ordinal del semestre. Es un número (1 o 2), correspondiente al primer o segundo semestre respectivamente.
    - Año: Año de la asignación.

Un diagrama de la dimensión Períodos para los Data Marts de *Exámenes* y *Cursos* puede verse en las Figura 4.13 y Figura 4.14 respectivamente.

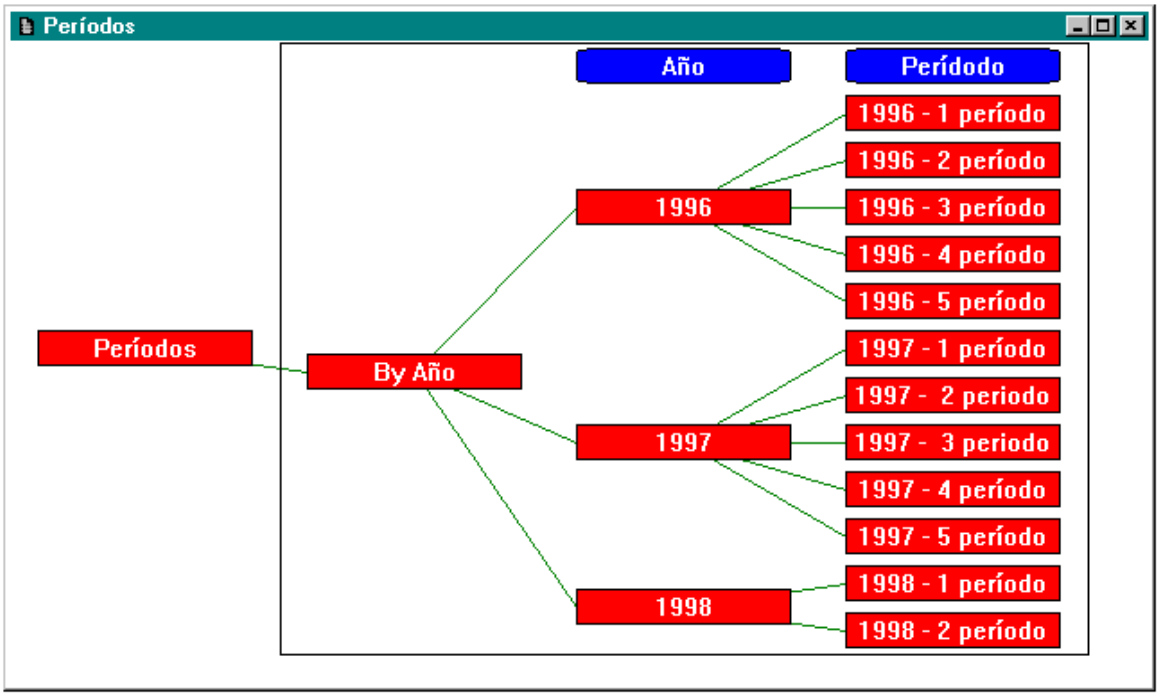


Figura 4.13 - La dimensión Períodos en el Data Mart de Exámenes

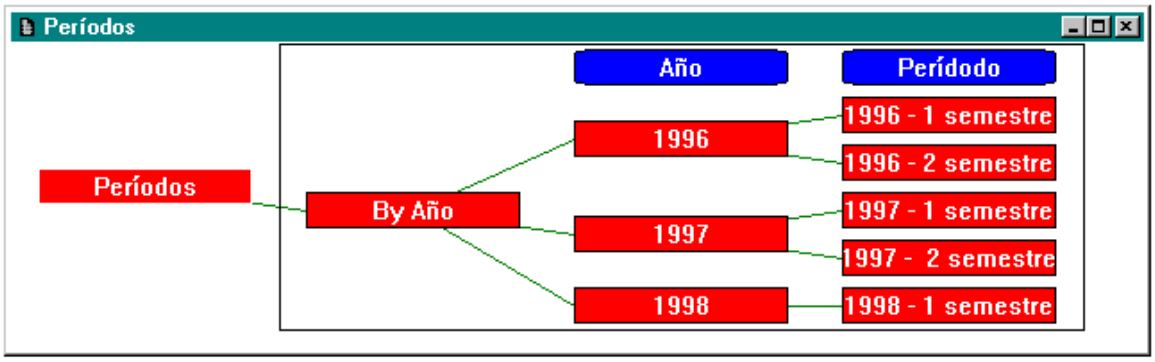


Figura 4.14 - La dimensión Períodos en el Data Mart de Cursos

4.7.3.3. Medidas

- Horas planificadas: Horas estimadas de trabajo durante la asignación.
  - *Granularidad más baja:* horas.
  - *Función de agregación*: suma.
- Horas reales: Horas trabajadas, informadas por el responsable de materia.
  - *Granularidad más baja:* horas.
  - *Función de agregación:* suma.

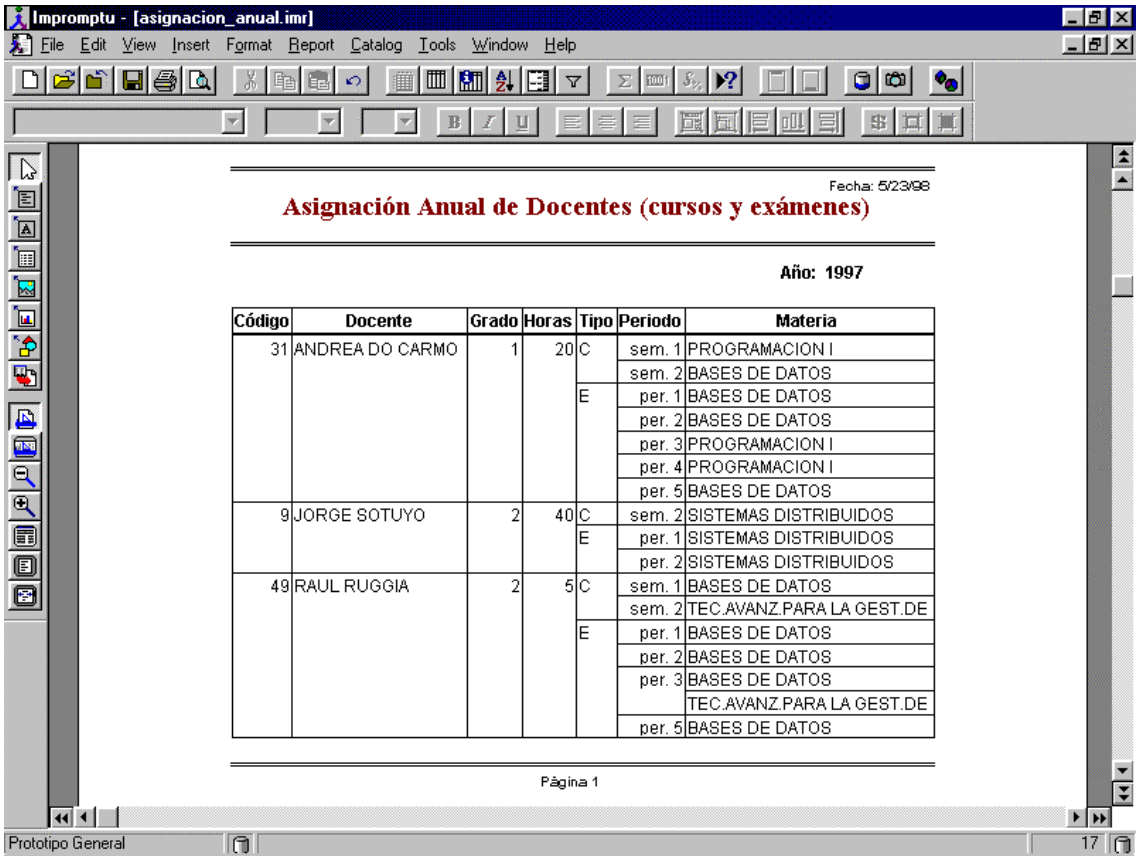
4.7.3.4. Vistas y Reportes

Se resolvieron los requerimientos solicitados:

(1) Planilla docente con grados, horas, cursos que dicta, y exámenes que toma.

Se construye un listado con *código, docente, grado, horas, tipo, período y materia,* filtrando un año que se solicita al usuario.

En la Figura 4.15 se observa una página del listado para el año *1997.*



Código	Docente	Grado	Horas	Tipo	Período	Materia
31	ANDREA DO CARMO	1	20	C	sem. 1	PROGRAMACION I
					sem. 2	BASES DE DATOS
				E	per. 1	BASES DE DATOS
					per. 2	BASES DE DATOS
					per. 3	PROGRAMACION I
					per. 4	PROGRAMACION I
					per. 5	BASES DE DATOS
9	JORGE SOTUYO	2	40	C	sem. 2	SISTEMAS DISTRIBUIDOS
				E	per. 1	SISTEMAS DISTRIBUIDOS
					per. 2	SISTEMAS DISTRIBUIDOS
49	RAUL RUGGIA	2	5	C	sem. 1	BASES DE DATOS
					sem. 2	TEC.AVANZ.PARA LA GEST.DE
				E	per. 1	BASES DE DATOS
					per. 2	BASES DE DATOS
					per. 3	BASES DE DATOS
						TEC.AVANZ.PARA LA GEST.DE
					per. 5	BASES DE DATOS

Figura 4.15 - Resolución del requerimiento (1) de Asignación

(2) Comparación de las horas trabajadas contra las planificadas.

Se coloca en uno de los ejes las medidas (horas planificadas y reales) y en el otro los períodos.

En la Figura 4.16 se observa la cantidad de *horas planificadas y reales* en los dos primeros períodos del año *1997*, filtrando por el departamento de *Programación,* en formato tridimensional. La figura corresponde al cubo de *Exámenes*.

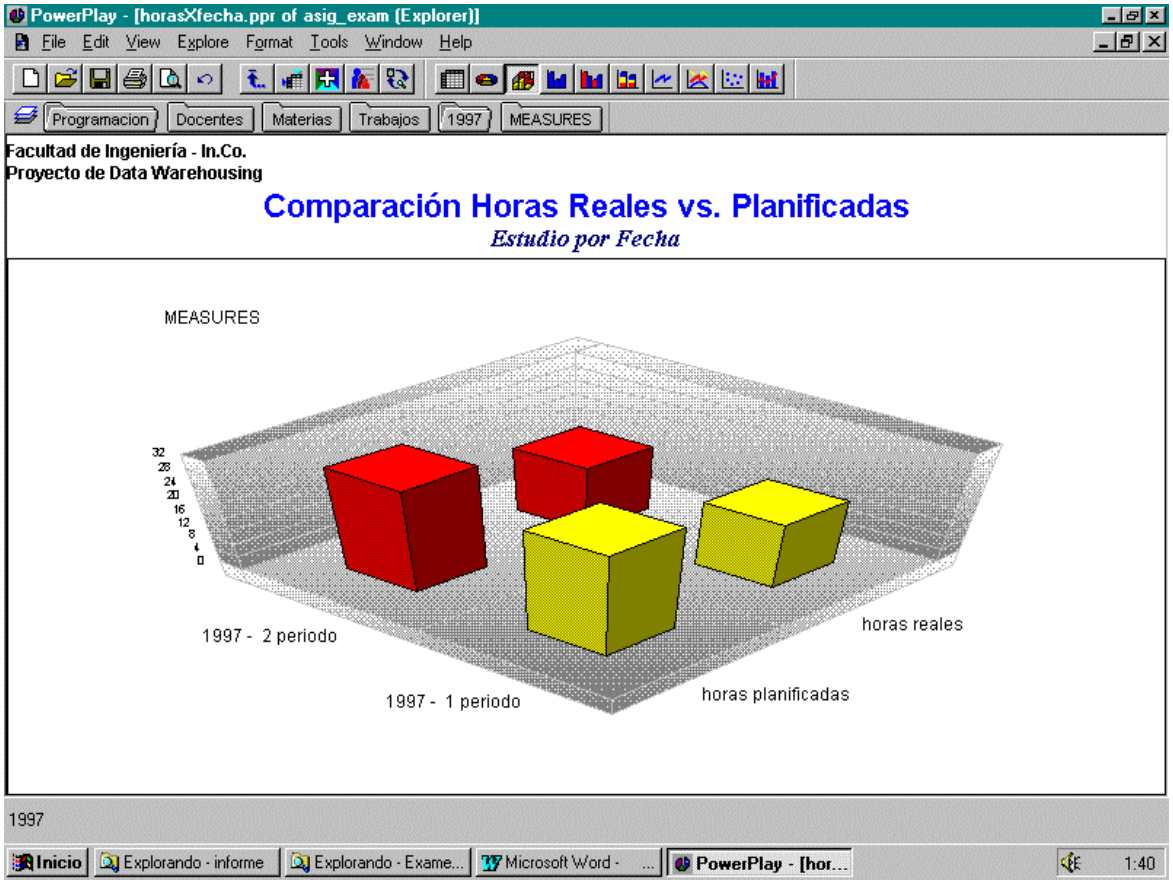


Figura 4.16 - Resolución del requerimiento (2) de Asignación

(3) Históricos del trabajo efectivo por docente.

Se coloca en el eje horizontal los períodos, y en el display las materias. Se miden horas reales y horas planificadas.

En la Figura 4.17 se observa la cantidad de *horas reales* contra *las horas planificadas* en los dos primeros períodos del año *1997*, filtrando por el departamento de *Programación,* en formato correlación. La figura corresponde al cubo de *Exámenes*.

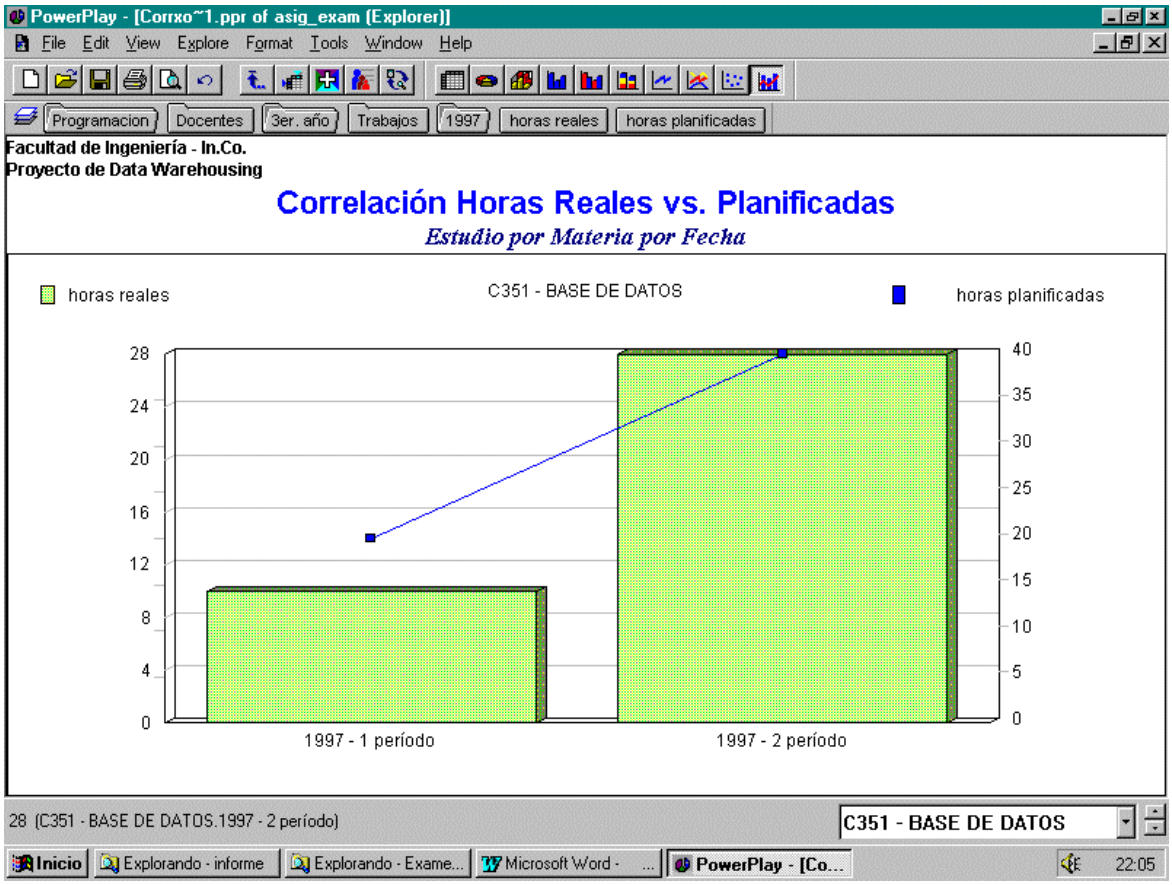


Figura 4.17 - Resolución del requerimiento (3) de Asignación

4.7.3.5. Carga

Los datos de asignación a exámenes y cursos serán cargados antes de los períodos de exámenes, y semestres de cursos, respectivamente. Los datos de trabajo efectivo serán cargados al final de los mismos.

Como las frecuencias de carga son diferentes, se proveen mecanismos para cargar cada una de las consultas transaccionales independientemente. El proceso será invocado por el DWA, o el operador responsable de la carga.

La carga será incremental, por lo que se controlará que no se cargue dos veces los mismos datos. Se provee también un mecanismo para cargar todo desde cero, por seguridad.

4.7.4. Presupuesto

Se resolvieron los siguientes requerimientos:

(1) Total ejecutado, discriminado por docente. El objetivo es evaluar como se está distribuyendo el presupuesto.

(2) Listado de docentes con grados y horas.

(3) Históricos.

4.7.4.1. Esquema Multidimensional

El esquema multidimensional se puede ver en la Figura 4.18.

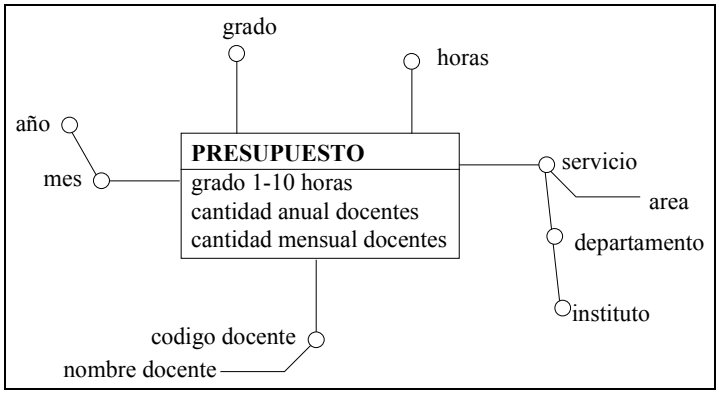


Figura 4.18 - Esquema Multidimensional de Presupuesto

4.7.4.2. Dimensiones

Se diseñaron 5 dimensiones. Las mismas se detallan desde el nivel de granularidad más bajo al más alto.

- Docentes:
  - *Jerarquía:*
    - Codigo docente: Código identificador único de un docente, independientemente del cargo que ocupe.
  - *Atributos no dimensionales:*
    - Nombre docente: Nombre del docente.
- Organigrama:
    - Servicio: un Código del área, dentro de un departamento.
    - Departamento: Código del departamento, dentro de un instituto.
    - Instituto: Código del instituto.
  - *Atributos no dimensionales:*
    - Area: Nombre descriptivo del área.

- Fechas:
  - Mes: Mes presupuestado.
  - Año: Año presupuestado.
- Grados:
  - Grado: Grado académico del docente. Es un número (1 al 5) correspondiente a Ayudante, Asistente, Profesor Adjunto, Profesor Agregado y Profesor Titular (respectivamente).

4.7.4.3. Medidas

- Grado 1- 10 horas: Medida del sueldo de un docente convertida a una unidad de referencia: salario de un grado 1 de 10 horas. Esto evita que las variaciones monetarias interfieran en los análisis que involucran comparar diferentes fechas.
  - *Granularidad más baja:* sueldo convertido.
  - *Función de agregación:* suma.
- Cantidad anual de docentes: Cantidad de docentes en un año dado.
  - *Granularidad más baja:* 1.
  - *Función de agregación:* suma.
  - *Restricciones de aditividad: No es aditiva con respecto a la dimensión Fechas.*
- Cantidad mensual de docentes: Cantidad de docentes en un mes dado.
  - *Granularidad más baja:* 1.
  - *Función de agregación:* suma.
  - *Restricciones de aditividad: No es aditiva con respecto a la dimensión Fechas.*

4.7.4.4. Vistas y Reportes

Se resolvieron los requerimientos solicitados:

(1) Total ejecutado, discriminado por docente.

Se construye un listado con *departamento*, *área*, *docente*, *cargo* y *ejecutado (gr1hs10),* filtrando un instituto y un mes que se solicita al usuario. Se agrupa por *departamento*, totalizando el ejecutado.

En la Figura 4.19 se observa la página del departamento de *arquitectura* del listado para el *Instituto de Computación* en *enero de 1998.*

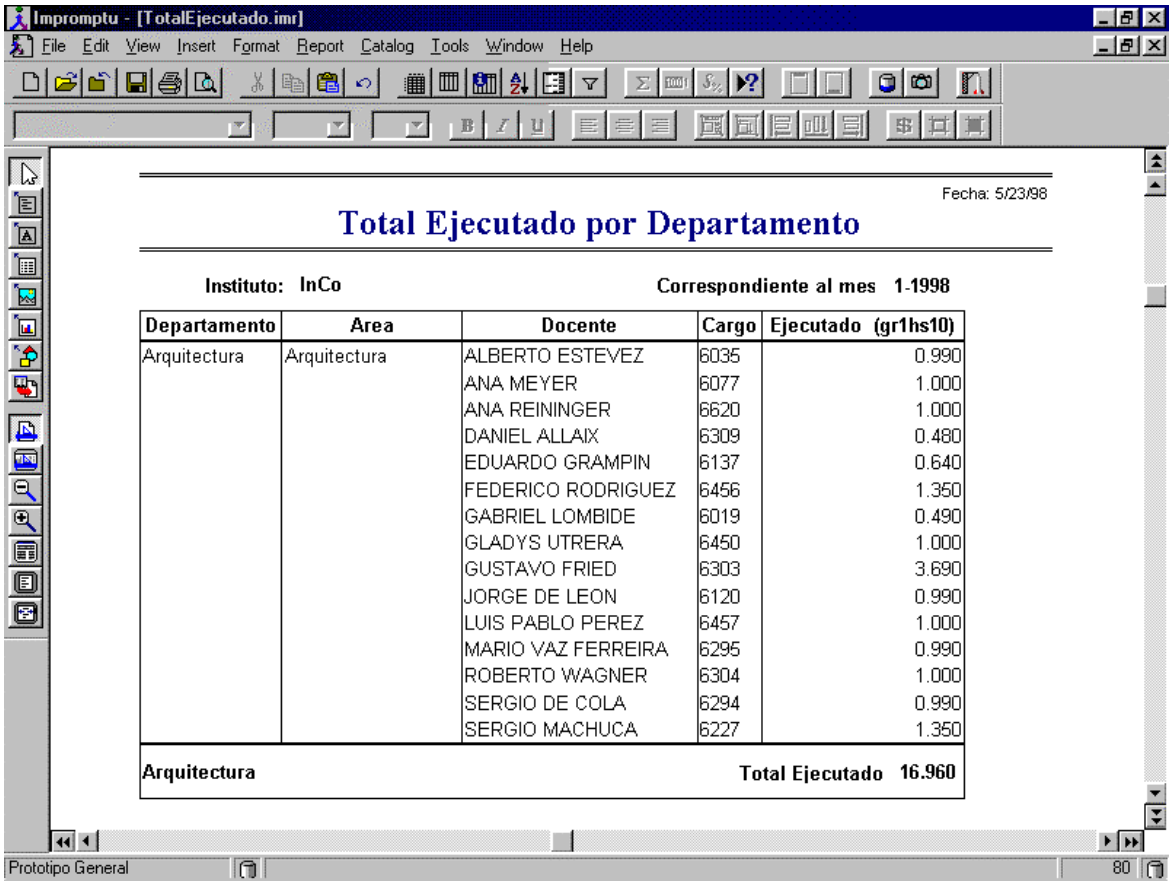


Figura 4.19 - Resolución del requerimiento (1) de Presupuesto

(2) Listado de docentes con grados y horas.

Se construye un listado con *departamento, área, docente, cargo, grado, horas, y gr1hs10,* filtrando un instituto y un mes que se solicita al usuario.

En la Figura 4.20 se observa parte del listado correspondiente al *Instituto de Computación* para el mes de *enero de 1998.*

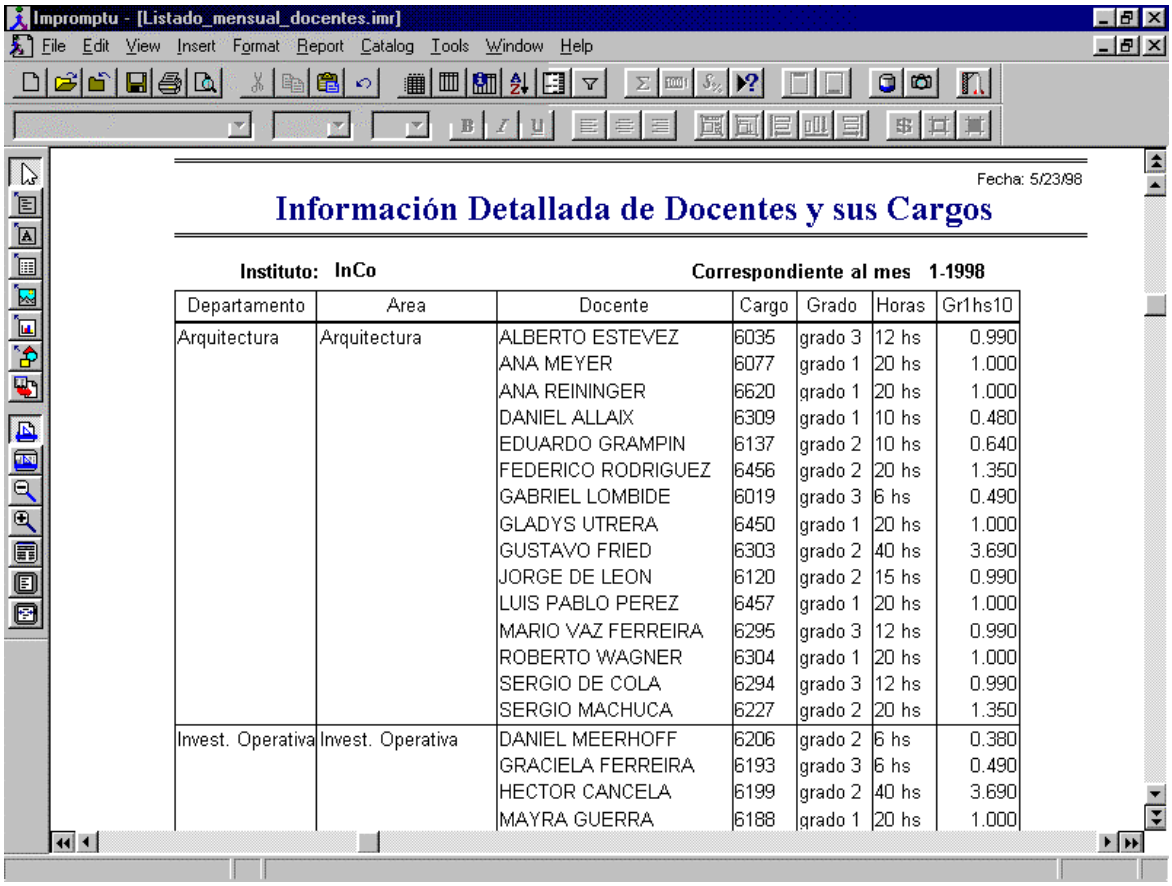


Figura 4.20 - Resolución del requerimiento (2) de Presupuesto.

(3) Históricos.

Se coloca en las filas las fechas, en las columnas el organigrama, y en las capas los grados. Se mide el sueldo (en grados 1 - 10 horas).

En la Figura 4.21 se observa el *sueldo (en grados 1 – 10 horas)* en los años *1997 y 1998*, para los departamentos del *In.Co.* y todos los grados.

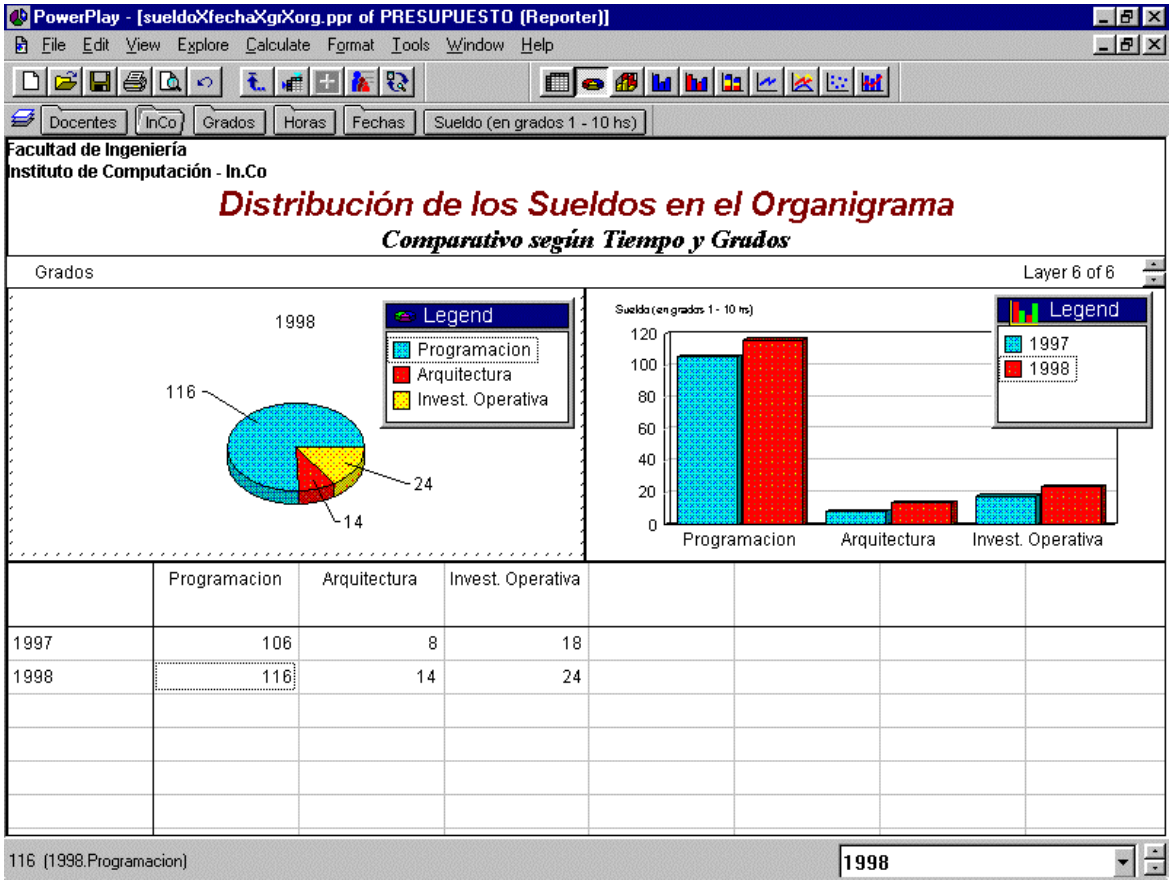


Figura 4.21 - Resolución del requerimiento (3) de Presupuesto.

4.7.4.5. Carga

Se espera una frecuencia de carga mensual. Sin embargo, no se sabe con anterioridad, a que altura del semestre llegarán los datos. Por tanto, se diseño un sistema de carga manual del Data Mart; es decir, el DWA, o el operador responsable de la carga, debe invocar a un proceso que efectuará la misma.

La carga será incremental, por lo que se controlará que no se cargue dos veces los datos correspondientes a un mes. Se provee también un mecanismo para cargar todo desde cero, por seguridad.

5. Metodología Seguida

En general la bibliografía consultada no trata el tema metodológico en profundidad, y cuando lo trata, es bajo ciertos supuestos que no se aplican a este proyecto (por ejemplo: inversiones millonarias, aplicación en áreas específicas – en general finanzas –, y con herramientas automatizadas para realizar tareas complejas como ser herramientas de extracción.

En este capítulo se discute la metodología empleada en este proyecto, relacionando las estrategias con las estrategias más conocidas – como ser modelo en cascada y espiral – comparando las distintas alternativas en caso de ser aplicable.

5.1. Visión global

La metodología descrita a continuación detalla los pasos que se considera necesario seguir para desarrollar un sistema de Data Warehousing. En la evaluación de cada actividad, se debe considerar que la duración estimada de la misma está basada en el marco del presente proyecto, por lo que sólo debe tomarse como una recomendación, restando un estudio más profundo para cada caso en particular.

En la Figura 5.1 se pueden ver las actividades de la metodología y su orden.

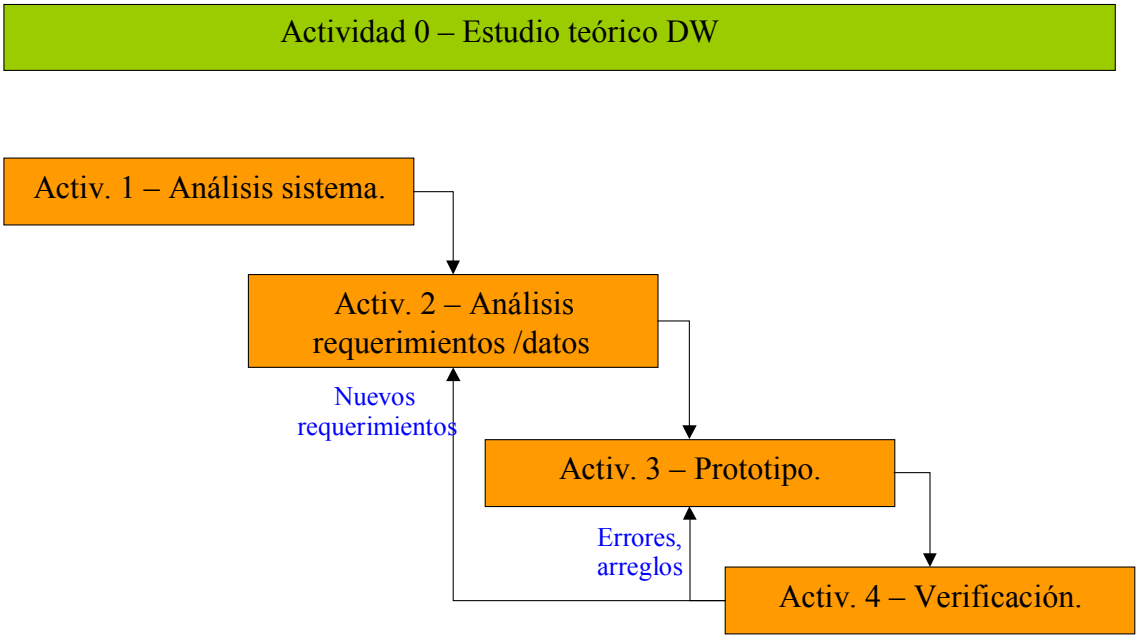


Figura 5.1 **– Actividades de la Metodología.**

Actividad 0: **Estudio conceptual de los Sistemas de Data Warehousing**

Objetivo: Mantenerse informado del estado del arte en la materia.

Principales pasos: En un primer acercamiento al tema, se recomienda comenzar con los libros de William Inmon y Ralph Kimball ([WI-93] y [RK96] respectivamente). En Internet hay páginas muy interesantes, con la ventaja agregada de la facilidad de actualización de este medio. Una de esas páginas muy completa es la de Larry Greenfield ([LG-97]). Una introducción a las bases de datos multidimensionales se puede encontrar en [KS-96].

Chequeo de finalización de la actividad: Al finalizar la actividad se debería tener un cabal conocimiento de las principales áreas de un sistema de Data Warehousing.

Actividad 1: **Análisis del sistema a desarrollar**

Objetivo: Tener una visión general del sistema que se piensa desarrollar.

Principales pasos: Se debe:

- Mantener entrevistas con los usuarios: los usuarios deben dar una visión general del tipo de trabajo que realizan.
- Estudiar las reglas de la empresa: comprender claramente como interactúan las distintas áreas y la información que comparten.
- Mantener entrevistas con DBAs: los DBA deben informar en grandes rasgos de que datos se disponen, para tener una visión general de los datos que poblarán el Data Warehouse.
- Relevamiento de la tecnología disponible: se debe relevar todas las herramientas disponibles que puedan ser usadas para el Data Warehouse (manejadores de bases de datos, herramientas de extracción y limpieza, herramientas front-end, software, hardware).

Salida de la actividad: Se debe generar un documento de análisis de sistema[11].

Chequeo de finalización de la actividad: Se debe evaluar con la gerencia el esquema del sistema y del proyecto de trabajo (se puede por ejemplo definir un esquema de trabajo en varias etapas y se debe tener una idea de que puntos priorizar).

[11] Esto es un campo de la Ingeniería de Software, para el que existe abundante documentación.

Actividad 2: **Análisis de requerimientos y datos**

Objetivo: Definir por completo lo que se espera del sistema y de donde se obtendrá.

Principales pasos: Se deben realizar entrevistas a los usuarios alternadas con entrevistas al/los DBA. A los usuarios se les debe preguntar ¿qué desean medir?. Las preguntas al DBA estarán enfocadas a una descripción detallada de cada una de las bases fuentes, formas de extracción de los datos y chequeos de consistencia necesarios.

Salida de la actividad: Se debe generar un documento con todos los requerimientos planteados (aún los que a priori no se resolverán) y un documento donde se detallan las bases fuente (relación entre ellas y descripción detallada). La especificación de los requerimientos se puede realizar en lenguaje natural. La especificación de las bases fuentes debe utilizar una notación más formal, como ser un MER y una descripción de las tablas.

Chequeo de finalización de la actividad: Al finalizar esta actividad se debe tener una clara definición de los requerimientos y los datos para el sistema que se desarrollará. El esquema de trabajo (definido en la actividad 1) puede dividir el desarrollo en varias etapas, por lo que puede ser necesario volver a este punto, pero para el estudio de otros requerimientos u otras bases.

Actividad 3: **Diseño y construcción del prototipo**

Objetivo: Diseño de Data Marts específicos para cada área y de un Data Warehouse corporativo.

Principales pasos: Se deben diseñar y construír: Data Marts para las principales áreas planteadas por los usuarios en la actividad anterior; un Data Warehouse integrado; y los procesos de carga de Data Warehouse y Data Marts.

Salida de la actividad: Se debe documentar el diseño del Data Warehouse, los Data Marts y los procesos de carga. La especificación del Data Warehouse es similar a la de las bases fuentes en la actividad anterior (preferiblemente con más detalle); la especificación de los Data Marts se debería hacer con una notación multidimensional (por ejemplo [DM-98]). La especificación de los procesos puede ser realizada mediante seudocódigo u otra descripción de alto nivel.

Chequeo de finalización de la actividad: Al finalizar esta actividad se debe tener un prototipo funcionando que pueda ser presentado a los usuarios para verificación y para obtener más información para una nueva etapa en el desarrollo.

Actividad 4: **Verificación**

Objetivo: Verificar el sistema y evaluar posibles mejoras.

Principales pasos: Se debe presentar a los usuarios el prototipo construído y alentar su uso desde que el primer prototipo está construído. Con la contribución de los usuarios se formará el nuevo conjunto de requerimientos/arreglos que formará parte de la siguiente versión.

Salida de la actividad: Problemas a solucionar, o requerimientos a resolver. Según la salida de esta actividad se puede volver a la actividad 2 (si se desea agregar nuevos requerimientos), o a la actividad 3 (si hay que corregir el diseño realizado, o mejorar el prototipo – por ejemplo en interface).

Chequeo de finalización de la actividad: Se debe dar un tiempo prudencial para que los usuarios puedan experimentar el prototipo. Luego de este tiempo, se debe tener algún comentario de los usuarios involucrados (caso contrario indicará que no fue usado y hay que tomar acciones correctivas inmediatas para no ver fracasar el proyecto).

5.2. Duración de las actividades técnicas

El principal problema de citar la duracion de cada actividad durante la descripción de la metodología seguida es que los tiempos puedan ser tomados fuera de contexto. Para esto es útil hacer una breve descripción de los principales problemas que formaron el contexto del proyecto.

- **Estudio conceptual de los Sistemas de Data Warehousing.**

  Tarea principal: Recopilar información y estudiar el tema.

  Duración: 2 meses.

  Contexto: Se comenzó el estudio desde 0, sin una base previa. Además del tiempo necesario para el estudio, se necesitó tiempo para recopilar la información.

- **Análisis del sistema.**

  Tareas principales: Entrevistas a usuarios, DBAs y relevamiento de tecnología disponible.

  Duración: 1 mes.

  Contexto: Se tuvo que encontrar a los usuarios claves (a los que les sería útil la información y podrían aportar datos). Coordinar reuniones tuvo su dificultad. Se notó la falta de responsables técnicos que supieran el formato de los datos, y este trabajo se tuvo que realizar sin ayuda.

- **Análisis de Requerimientos y Datos.**

  Tareas principales: Entrevistas a usuarios, estudio del formato de la información.

  Duración: 5 meses.

Nota: El tiempo no se corresponde al efectivamente realizado, debido a que por la indisponibilidad de Sql*Net no se pudo pasar a la siguiente actividad, y durante aproximadamente 2 meses sólo se realizaron pequeños chequeos, de mas o menos 1 hora para revisar algún detalle.

Contexto: El servidor donde estaba instalado Oracle no tenía espacio en disco, y hubo dificultades para realizar procesamiento simples desde el primer momento. La inexistencia de DBAs hizo que se tuviera que realizar el trabajo de éstos e intentar deducir información a partir de los datos. No se dispuso de SQL*Net (acceso a redes de Oracle) hasta el mes de noviembre. Como consecuencia de esto, se decidió realizar un prototipo con algunos datos en Access para adelantar trabajo. Este prototipo se construyó en 2 meses y tuvo las dificultades adicionales del pasaje de información desde Oracle (la mayor parte de los datos se cargó manual).

- **Diseño y construcción del prototipo**

  Tareas principales: Diseño de Data Marts, Diseño Data Warehouse, construcción de prototipos, construcción programas de ingreso de datos, programación de procesos de carga.

  Duración: 4 meses.

  Contexto: No se pudo empezar hasta el mes de noviembre porque no se disponía de Sql*Net, aunque la actividad anterior estaba terminada mucho antes. Por problemas de la herramienta de catálogo, no se pudo migrar el primer prototipo construído (en Access), y se tuvo que realizar nuevamente. Al probar el prototipo con Oracle, se notó que parte del diseño era inviable por la cantidad de datos, por lo que se tuvo que rediseñar. La falta de espacio en disco impidió mejorar la performance mediante la utilización de índices (que hubieran evitado en ciertos casos un arreglo a las tablas del Data Warehouse).

- **Verificación**

  Tareas principales: Poner el prototipo en producción, pedirle a los usuarios comentarios o mejoras que hay que realizar.

  Duración: 1 mes.

  Contexto: El prototipo estuvo terminado en el período de licencia de los usuarios, y luego de finalizado el plazo inicial del taller, por lo que no se pudo realizar una verificación grande de los Data Marts con los usuarios.

5.3. Estudio conceptual Sistemas de Data Warehousing.

Hay que considerar que los sistemas de Data Warehousing son sistemas de aparición relativamente reciente. Son una gran fuente de investigación, tanto académica como comercial, y como tales, están en continua evolución y desarrollo tecnológico.

Mantenerse informado de los avances en este terreno puede ser de gran utilidad si se está pensando en el desarrollo de un sistema de Data Warehousing.

Es por ello que este punto estuvo presente durante toda la duración del proyecto, no sólo en la fase de aprendizaje, recabándose información de las diversas fuentes citadas en la bibliografía.

5.4. Proceso de desarrollo

El proceso de desarrollo propuesto tiene los mismos pasos que un sistema convencional, pero el enfoque en cada uno de esos pasos es un poco diferente, y las dificultades encontradas también lo son.

Una característica importante de los sistemas de Data Warehousing es su continuo crecimiento – no sólo en tamaño, sino también en áreas de explotación –. En general comienzan como pequeños Data Marts aislados, que integran pocos datos y tienen un conjunto pequeño de funcionalidades; para luego ir escalando hacia sistemas realmente grandes y complejos. Esto hace que el software sea muy difícil de diseñar, debiéndose prestar especial importancia a las cualidades de generalidad y evolutividad.

Como muestra de lo anterior, se puede destacar que el estudio realizado en este proyecto para las carreras 70 y 71 puede ser ampliado inmediatamente a las carreras 60 y 61, y sin dificultad a las otras carreras de facultad (crecimiento en tamaño). Además, en el caso de la base de Presupuesto, se agregaron a último momento los requerimientos del Area Cambios, los que no habían sido planteados (crecimiento en áreas de explotación).

El dimensionamiento del proyecto es muy difícil, tanto para estimar costos, y recursos, como para estimar el tiempo de desarrollo.

5.4.1. Análisis del sistema a desarrollar

Dado que el concepto de Data Warehousing es muy nuevo, en especial en nuestro país, hay que tener mucho cuidado al definir un sistema de este tipo. Generalmente se da una definición muy pequeña de lo que se pretende del sistema, y las exigencias empiezan a crecer a medida que evoluciona el desarrollo.

El análisis del sistema debe ser realizado con más cuidado que en un proyecto típico de base de datos o ingeniería de software. Requerirá de un esfuerzo especial del analista para prever y controlar los cambios en las exigencias y enfrentar las dificultades que esto traerá.

5.4.1.1. Estudio general de los requerimientos.

Como todo análisis de un sistema, se debe comenzar por obtener una visión general de los requerimientos.

Aquí es donde se conoce a grandes rasgos, la organización de la empresa, las áreas o sectores que la forman, y el tipo de análisis de los datos que se realizan en cada una de ellas.

Es importante definir a cuáles de esas áreas está destinado el sistema, y si hay posibilidades de una ampliación futura. No hay que olvidarse que la mayor parte de los sistemas de Data Warehousing surgen con pequeños prototipos de Data Marts para algunos sectores de la empresa, que luego se integran en un sistema corporativo.

Se definen así los distintos perfiles de usuarios que tendrán acceso al sistema. Junto con esto, se logra una primera idea de los requerimientos, que se ampliará al entrevistar a los usuarios de cada área.

En este punto, tambien es importante obtener una lista de personas comprometidas en el proyecto, que apoyen tanto en la definicion de requerimientos como en la recopilación de información.

5.4.1.2. Estudio de las reglas de empresa.

Desde el inicio se deben comprender bien las reglas de empresa. Muchas veces las reglas no están definidas explícitamente y es bueno dedicar el suficiente tiempo para que queden claras.

Se debe evitar que durante el desarrollo se produzcan cambios en la visión de empresa lo que retrasará sin duda la evolución del proyecto. Por ejemplo, en este proyecto, se asumió inicialmente que un número de cargo determinaba a un docente, y se implementaron mecanismos de carga que se basaban en ello; cuando se descubrió que luego de una renuncia el número de cargo se utiliza nuevamente para otro docente, hubo que redefinir los programas de carga.

Las reglas de empresa controlan como interactúan las distintas áreas, y la información que comparten, lo que será fundamental para el diseño del sistema.

5.4.1.3. Estudio del sistema de producción y las bases fuentes.

Se deben comprender claramente las características generales de los sistemas de producción. Son relevantes el ambiente en que corren los distintos aplicativos, el software y hardware que utilizan, y como se encadenan o interactúan los diferentes procesos que los gobiernan y controlan. Este relevamiento será útil a la hora de comparar distintos productos (manejadores de bases de datos, herramientas de extracción) para el Data Warehouse, prestando especial atención a las facilidades de comunicación con los sistemas construídos. Se recomienda no obviar este punto, ya que puede facilitar el resto del proceso en gran medida (por ejemplo en el proyecto, el sistema de Bedelía en Oracle Xenix 5 no es compatible con el Data Warehouse en Oracle AIX 7, por lo que se deben realizar varias conversiones previas a la carga)

Pero, sin duda, el aspecto más importante de dichos sistemas es la información que manejan. Si bien en la siguiente actividad (Análisis de Requerimientos y Datos) se estudiarán con profundidad las bases fuentes, su definición, la granularidad de los datos, y la redundancia entre ellas, en esta primera actividad es útil tener un conocimiento general de cuáles son los datos existentes. Este criterio, fue usado en el proyecto. Puede verse claramente la diferencia de enfoques entre la sección 3.3, donde se precisaba saber de qué información se disponía, y el capítulo 4 (y Apéndice B), donde se estudian las bases para ser cargadas en el Data Warehouse.

5.4.1.4. Estudio de la tecnología disponible y las alternativas técnicas.

Se debe comenzar por tener una idea de las tecnologías disponibles en la empresa, tanto de software, como de hardware, y plantear la adquisición de otras tecnologías más adecuadas para el sistema.

Un sistema de Data Warehousing evoluciona muy rapidamente, y se construye sobre un sistema de producción, generalmente cambiante, dos motivos por los cuales deben evitarse todo tipo de soluciones a medida.

Manejadores de Bases de Datos

En lo que respecta al software, sin duda el elemento más importante es el manejador de bases de datos. La primera decisión a tomar es si se utilizará un manejador tradicional, o se implementará el acceso a los datos.

No siempre es necesaria la utilización de un manejador. Para sistemas sencillos, en los que los datos no provienen de sistemas muy heterogéneos, y sin fuerte integración, pueden construirse los Data Mart directamente, sin materializarse el Data Warehouse. Como entrada de datos pueden considerarse la utilización de archivos planos, o comunicación directa con el sistema de producción, por ejemplo vía ODBC.

En sistemas más complejos, es casi indispensable la utilización de un manejador que provea las funcionalidades de almacenamiento y acceso eficiente a los datos, así como controles mínimos de acceso.

Generalmente el manejador utilizado en el sistema de producción puede ser utilizado para el sistema de Data Warehousing, pero no siempre es lo indicado. A veces nos encontramos con distintos manejadores dentro de la empresa. Como un ejemplo, los sistemas de producción pueden funcionar sobre pequeñas bases estilo Microsoft Access, y manejar pequeños volúmenes de datos, pero no ser indicadas para manejar grandes volúmenes de datos históricos o procesar controles complicados.

Se debe tratar de elegir un motor de base de datos que brinde el máximo de las facilidades que se necesitan, como mecanismos de triggers, procedimientos almacenados, control de claves foráneas, etc., para no tener que programar todos esos controles.

Sin duda, ésta es una decisión fundamental para el sistema. Aquí no sólo interviene el factor económico contra complejidad y performance, sino que el volúmen de datos tiene un peso decisivo.

No obstante, generalmente, la elección del manejador, se ve condicionada a los disponibles en la empresa, retrasando la elección para futuras versiones del sistema. En las fases de diseño deberán preverse cambios en la arquitectura.

Por ejemplo, entre las alternativas disponibles en Facultad, se disponía de Oracle 7.3 bajo Unix, y una base de datos de escritorio Access. La mayor parte de los datos de las bases fuentes están en Unix (en una base de datos Oracle), mientras que las herramientas para el usuario final corren en un PC (herramientas de Cognos y Business Objects). La disyuntiva era elegir tener los datos en la base de datos Oracle (o sea más cerca de las bases de producción), o tener los datos en Access (más cerca del usuario final). Se eligió finalmente Oracle, además de por su mayor potencia y mejor escalabilidad, por el hecho de disponer de mayor cantidad de herramientas, lo que facilitaría los procesos de carga.

Herramientas de extracción y limpieza

Gran parte del esfuerzo de desarrollo, como ya se ha citado, está destinado a los procesos de limpieza e integración; el uso de herramientas especializadas puede resultar fundamental en disminuír el tiempo y el esfuerzo de desarrollo.

Lamentablemente, estas herramientas no están al alcance de la mayoría de los proyectos, y se opta por programar.

Se puede resumir las ventajas y desventajas de cada una de estas opciones como sigue:

- Herramienta especializada:

Hay herramientas especializadas que se dedican a solucionar el problema de la lectura de varios sistemas (ej.: sistemas relacionales, jerárquicos, de archivos, etc) y la carga en otro sistema (en general relacional). Como ejemplo de estas herramientas podemos citar: ETI Extract y Carleton Passport. Algunas herramientas, además se ocupan de la integración de bases de datos de ambientes heterogéneos. Un ejemplo de este tipo de herramientas es Leonardo´s Logic Genio.

La mayor desventaja es que estas herramientas son bastante costosas pues están pensadas para ser usadas en la creación de un Data Warehouse de gran tamaño y de gran confiabilidad.

- Scripts de limpieza y carga:

También se pueden definir scripts de limpieza y carga que se ejecuten contra las bases fuente, y que realicen los procesos de extracción, y de conversión de datos necesarios para cargar el Data Warehouse.

La ventaja de esta opción es que es una opción económica, que puede ser usada como primer escalón, mientras el proyecto de Data Warehouse comienza, para luego utilizar herramientas como las mencionadas más arriba.

En general, los métodos más usados se basan en el uso de scripts en SQL, SQL aumentado (PL/SQL), SQL embebido u ODBC.

En este proyecto, la razón principal por la cual se usaron scripts para la carga y limpieza fue económica. Además no se estaba ante un sistema con grandes complicaciones que justificara la compra de una herramienta de extracción y limpieza.

Herramientas Front-End

Es una parte fundamental del sistema, ya que es la parte que los usuarios apreciarán. A diferencia de las herramientas de extracción y limpieza, las herramientas disponibles en el mercado no son muy caras, y ofrecen una amplia gama de funcionalidades, por lo que en general es mejor inversión el uso de estas herramientas genéricas que el crear una especializada.

No obstante, para resolver algunos requerimientos más complejos puede ser necesaria un poco de programación, ya sea en la construcción de pequeñas macros que automaticen consultas complejas, o para diseñar una interfase de comunicación con otros aplicativos, por ejemplo pequeños programas en Visual Basic. Aquí también se observa la importancia de la correcta elección de las herramientas. Se debe estudiar las posibilidades de programación y comunicación que ofrecen éstas con el medio exterior.

Otro software a considerar

Hay otros elementos del software que también deben ser estudiados, como son: el sistema operativo, el software de comunicación en la red, los aplicativos de usuarios existentes, y el software de integración de componentes.

Es importante saber con que programas está familiarizado el usuario, ya que esto podría ayudar en la elección de la interface a utilizar.

Hardware

Se debe tener en cuenta el equipo en el que funcionará el sistema: servidores, estaciones de trabajo, topología de la red, etc. Esto está muy estrechamente relacionado con la performance que se desea del sistema, pero no es el único factor a considerar.

La migración de los datos desde el sistema de producción al sistema de Data Warehousing debe ser el centro de atención, ya que es uno de los procesos más complejos.

Una decisión importante es que parte del sistema se encontrará en cada plataforma, lo cual es fundamental para diseñar los mecanismos de comunicación y movimiento de datos.

Las herramientas, tanto de extracción como Front End, son para ambientes específicos, y no se puede lograr cualquier combinación de ellas. Asimismo, los manejadores no funcionan en cualquier plataforma, y tienen requerimientos de hardware bastante exigentes.

Por ejemplo en Facultad se cuenta con una importante cantidad de estaciones de trabajo con distintas versiones del sistema operativo Unix, comunicándose sobre una red Ethernet; existen también varios PCs con Windows 95, Windows 3.11 y Windows NT, integrados a la red.

5.4.1.5. Definición del sistema.

Habiendo estudiado los puntos anteriores se debe definir cómo será el sistema.

Se documentan los grandes conjuntos de requerimientos, y las áreas o usuarios a los que corresponden. Se registran también las reglas de empresa, y la interacción entre las distintas áreas.

Se documenta también lo relativo a las bases fuentes disponibles, y frecuencia de actualización del sistema con respecto a cada una de ellas. Se puede utilizar un diagrama para representar de manera general las relaciones entre las bases.

Por último se define la arquitectura a grandes rasgos. Se especificará la base del Data Warehouse: tipo de manejador (relacional o multidimensional), los criterios de carga y validación, y otras restricciones generales. Se especificará también los distintos Data Marts: el tipo de almacenamiento (Rolap o Molap), los procesos de carga y la frecuencia de los mismos, las interfaces deseadas, y otras funcionalidades adicionales como integración con otros componentes, publicación en el Web, etc.

Toda esta documentación no difiere de la realizada tradicionalmente en los proyectos de Ingeniería de Software, y entendemos que debe ser realizada en un lenguaje informal o semi-formal, de forma de que pueda ser entendida por los usuarios en momentos en que surja alguna diferencia sobre lo que se había propuesto. En caso de haber un contrato de por medio, este documento debería ser Anexo del mismo, indicando un consentimiento entre las partes sobre los alcances del sistema.

5.4.2. Análisis de requerimientos y datos

El análisis de requerimientos debe realizarse con mucho cuidado, y dedicarse el suficiente tiempo para ello. Se debe enfocar en las necesidades de información de los usuarios, sin apartarse demasiado de lo especificado en el análisis del sistema, en cuánto a los requerimientos y los datos disponibles.

Según las recomendaciones de Ralph Kimball en "The Data Warehouse Toolkit" [RK-96], se deben realizar alternadamente, consultas sobre requerimientos y datos disponibles[12]. Esto permite cotejar los requerimientos solicitados con los datos que se disponen, y evitará prometer información que es imposible obtener.

5.4.2.1. Entrevistas a usuarios

En las entrevistas con usuarios se debe tener especial cuidado al definir los requerimientos ya que los usuarios en general no conocen lo que es un Data Warehouse y por tanto lo que les puede ofrecer.

Se tienen los 2 extremos:

- Se pide poco más que lo que brinda un sistema de producción, pero con una buena interface.
- Se pide un sistema milagroso que resuelva todos los problemas que no cubre el sistema de producción existente.

El primer caso es muy común, los usuarios tienen como requerimientos fundamentales poder hacer todo lo que hacen actualmente, con buenos gráficos y una excelente performance; y tal vez quieren también, alguna funcionalidad que actualmente no pueden resolver o es muy costosa. Pero no pueden ver o imaginarse el nuevo conjunto de operaciones y análisis estratégico que tendrán disponible por el mero hecho de tener acceso a datos resumidos e históricos.

Es normal que en las sucesivas versiones del sistema surjan más y más requerimientos y se integren nuevos datos e indicadores, cuya importancia no había sido analizada. En este ambiente, el diseño para el cambio no es sólo importante, sino que es esencial, y el analista lo debe tener en cuenta durante toda la duración del proyecto. No se deben desechar a priori parte de los datos o caer en casos particulares o soluciones demasiado a medida ya que el sistema tenderá a crecer, y muy rápidamente. Se debe evitar que cosas sencillas como agregar una nueva tabla o un nuevo indicador obliguen a una reestructuración del sistema, ya que esto ocurrirá con mucha frecuencia.

El segundo caso (usuarios que piden sistemas milagrosos) es mas fácil de atacar, ya que desde el inicio se sabe todo lo que el usuario desea. El problema aquí es no dejar que se creen falsas expectativas y que el sistema lleve a una desilusión. Debe quedar muy en claro que cosas se podrán lograr , que cosas quedarán para versiones futuras y que cosas no serán resueltas.

Por supuesto, no hay ningún usuario que se encuentre en los extremos; los usuarios son siempre una combinación de los dos casos mencionados. El conocer los casos extremos nos ayuda a conceptualizar los principales rasgos de cada uno de ellos y a poder entrevistarlos mejor.

[12] Aunque en el entorno de este proyecto, eran las mismas personas.

Es fácil darse cuenta cuando están llevando alguno de los comportamientos descritos, lo que es difícil es poder abstraer lo suficiente lo que ellos cuentan, y llegar a lo que realmente esperan.

¿Cómo lograr que los usuarios nos expliquen algo que ellos conocen tanto que dan por sentado que todos lo saben? ¿Cómo lograr que una entrevista se convierta en una fuente importante de información, y no sólo en un texto a desentrañar? Estas son las preguntas que todos nos hacemos en el momento de enfrentar una entrevista con los usuarios.

Para ello encontramos que la pregunta clave es: "**¿Qué quiere medir?**". De esta manera se establecerán las relaciones necesarias entre el trabajo que realiza y los datos que consulta.

Se debe evitar el explicarles a los usuarios qué es un sistema de Data Warehousing, su composición y objetivos pues esto puede coartar sus respuestas. Este no fue el criterio que se usó en el proyecto y más que facilitar el desarrollo lo complicó.

Se debe tratar de enfocar la entrevista en las necesidades del usuario de la forma más precisa posible. Luego, el analista deberá separar los requerimientos que se podrán realizar, los que no corresponden al sistema, y los que no se podrán realizar (sea por falta de tiempo o por falta de información); pero esto, debe ser tarea del analista, **no** del usuario.

5.4.2.2. Análisis de los datos fuentes

La base sobre la que se apoya un Data Warehouse son sus datos fuente. En función de los datos fuente que se disponga se podrán resolver o no los requerimientos.

Para ello se precisa no sólo tener una lista de las fuentes de datos, sino también comprender su significado. Para esto último, es fundamental el apoyo de los usuarios involucrados y de los diseñadores o administradores de la Base de Datos (estos últimos no existentes en el proyecto).

En general, se nota mucha más aceptación en cuanto al desarrollo del proyecto que apoyo en la comprensión de los datos[13]. En algunos casos apenas se tenía una idea de los datos existentes, o pequeñas muestras de información, muchas veces confidencial.

[13] Esto pudo deberse tambien al hecho de que en este caso no existía una estructura que se ocupara de las bases fuentes (sea un DBA, sea un equipo de desarrollo), y por ello se solicitó la colaboración de los usuarios – los que con buena voluntad – apoyaron en todo lo que conocían, aún sin ser su responsabilidad.

El problema de la falta de estructuración de la información es muy importante. En este proyecto, un porcentaje muy alto de los datos no contenían un formato fijo, y a veces los datos provenían de formularios o papel impreso. Intentar obtener respuestas de los datos mismos fue difícil, y muchas veces se llegó a conclusiones erróneas que obligaron a rever el diseño.

La comprensión de los datos significa saber que significa cada campo, y que valores puede contener. Esto último es mucho más difícil de establecer y en general no se dispone de suficiente información como para saberlo. Por ejemplo, es claramente inválido un valor NULL para calidad de un estudiante (activo, egresado, provisorio y suspendido son válidos). Sin embargo, no es inmediatamente claro los valores válidos en la fecha de ingreso de un estudiante, aunque todos estarán de acuerdo que no puede ser correcto una fecha de ingreso en 1910 siendo que la carrera se creó en la década de 1960.

Por último, pero no menos importante, no hay que olvidar que luego habrá que extraer los datos de estas bases fuentes, por lo que no sólo se debe estudiar la estructura, sino que se deben analizar diferentes opciones para obtenerlos.

Esta tarea no siempre es trivial, y puede llegar a ser muy compleja, según el sistema base.

Lo mejor que puede pasar es que el sistema base esté en una base de datos Relacional. Siendo así, no será complicado obtener los datos (se puede usar sintaxis SQL), y no habrá que hacer conversión entre diferentes paradigmas.

Sin embargo, en empresas grandes, es difícil terminar la inversión en sistemas gigantes – tipo mainframe – con datos en archivos. Esto representa un problema de extracción y conversión, que puede llevar mucho tiempo solucionar. Puede ser necesario tener que correr una consulta predefinida y grabarla como la única forma de obtener datos desde el sistema.

5.4.2.3. Especificación

La salida de la actividad de *Análisis de Requerimientos y Datos* se debe documentar de forma de poder ser utilizada en la actividad de *Diseño*.

Se debe documentar lo siguiente:

Requerimientos

Como notación, en este caso se utilizó lenguaje natural, pero en caso de manejarse algún otro tipo de modelo, es igualmente válido. Tener en cuenta que siempre que se debe definir todos los vocablos especializados usados (como por ejemplo se definió en el capítulo 3: actividad, estudiante activo, abandonos, etc.).

Bases fuentes

Como notación, se puede utilizar alguna simplificación del MER ([PC-76]) u otro tipo de diagramas, para mostrar la relación entre las distintas bases de los sistemas. Además se debe obtener una descripción detallada de las tablas con sus atributos y tipos (para ello se puede usar una notación estilo headers de Cobol ("copy books") para describir cada una de las tablas).

También se deben especificar, en caso de que ser necesario, las opciones disponibles para obtener datos del sistema (esto no debería ser necesario en sistemas basados en Bases de datos).

5.4.3. Diseño y construcción del prototipo

5.4.3.1. Diseño del prototipo

El siguiente paso del desarrollo consiste en comenzar el diseño del sistema. Como se explicará más adelante, consideramos que el proceso de diseño debe seguir un modelo incremental (con prototipación). Básicamente se debe atacar al problema por dos flancos principales:

- Pasaje de las bases fuentes a un entorno limpio e integrado.
- Diseño de Data Marts que atacaran los principales problemas que se planteaban los usuarios.

Cada flanco debe servir de apoyo al otro, desarrollándose en paralelo, y compartiendo resultados y problemas, especialmente en las primeras etapas del desarrollo.

Lo anterior lleva a que durante las primeras etapas, en el diseño de Data Marts, no se cuente con un Data Warehouse armado y estable. Por razones de seguridad y performance global, es mejor no trabajar sobre las bases de producción, inclusive, puede ser que algunas de ellas no existan (como en el caso de Asignaciones, en que los programas y la base fuente no estuvieron disponibles hasta avanzado el desarrollo). En este caso, consideramos que se deben crear pequeñas bases, con datos de muestra o ingresados a mano, y trabajar sobre éstas, hasta obtener un nivel de estabilidad aceptable.

En este proyecto, se encontró además otra versión del problema anterior, que era disponer de los datos, pero no de la forma de comunicación (en nuestro caso se precisaba Sql*Net, el cual no estuvo disponible hasta 5 meses más tarde). Esto llevó a la creación de bases en otros sistemas (Access) para poder crear los Data Marts, aunque luego tuviera que repetirse el trabajo para adaptarlo al sistema Oracle.

Como se puede ver, el problema de disponer de todos los componentes necesarios para realizar el sistema **no** debe ser subestimado, pues inclusive en las empresas es difícil justificar una inversión – que no suele ser pequeña – para sistemas muy nuevos, como lo son los sistemas de Data Warehousing.

5.4.3.2. Diseño de bases de datos

Un componente fundamental del prototipo es el diseño de sus bases de datos, tanto la base del Data Warehouse en sí mismo, como las de los Data Marts. Cómo se expresaba en el capítulo 2 (Introducción a la Tecnología de Data Warehousing), en general se crean modelos relacionales para el Data Warehouse y multidimensionales para los Data Marts.

El diseño de las bases de datos debe ser realizado de forma cuidadosa. A diferencia de las bases fuentes, es esencial el mantener la consistencia, tanto en nombres como en unidades de medidas. Esta consistencia sólo es posible alcanzarla documentando que es lo que se puede y que es lo que no se puede hacer. En este proyecto se creó una Guia de Convenciones (la que se detalla en el Apéndice C) que es un ejemplo de lo que hay que definir para mantener la consistencia.

5.4.3.3. Diseño de Procesos

Para el sistema a construír, también se debe diseñar los procesos de carga de las Bases de Datos, así como las consultas necesarias para crear los Data Marts.

Estos pasos serán más sencillos cuanto más herramientas automatizadas se dispongan, especialmente en el diseño de los procesos de extracción.

5.4.3.4. Especificación

5.4.3.4.1. Objetos y relaciones

Al igual que en las Bases Fuentes, consideramos que los mejores modelos para especificar los resultados del Diseño son el MER y una descripción de las tablas (es deseable que la descripción de las tablas sea más descriptiva que para las bases fuente).

5.4.3.4.2. Jerarquías e indicadores

Para especificar los modelos multidimensionales se eligió una variante de la notación creada por D. Maio, M. Golfarelli y S. Rizzi, que se puede ver en la Bibliografía ([DM-98]). Ésta nos permite expresar dimensiones, medidas, jerarquías y otras restricciones (por ejemplo funciones de rollup), en una notación uniforme e independiente de la implementación física.[14]

[14] Esto es importante pues otros modelos, en particular star-schema tienen una gran componente de representación física, que puede "marear" en un modelo conceptual.

5.4.3.4.3. Procesos

Para especificar los procesos, se eligió usar seudocódigo de los scripts SQL.[15] Como se decía previamente, el uso de herramientas automatizadas facilita los procesos de extracción, e inclusive la mayor parte de ellas generan documentación sobre los procesos programados, lo que puede ser usado como especificación.

5.4.4. Verificación

Por último, corresponde la verificación de los sistemas construídos. En lo posible se debe realizar la misma con la presencia de los usuarios. Esta actividad se debe ejecutar simultáneamente con la anterior, en el entendido de que son los usuarios quienes más fácilmente detectarán mejoras al prototipo construído.

Los sistemas de Data Warehousing son sistemas en constante desarrollo, por lo que no se les puede poner un punto final. Según W. Inmon, los proyectos tradicionales empiezan con requerimientos y terminan con datos, por el contrario, los proyectos de Data Warehousing empiezan con datos y terminan con requerimientos. Una vez que los usuarios ven lo que se puede obtener, quieren mucho más.

La propia evaluación del Data Warehouse hace que de esta fase se obtengan nuevos elementos o requerimientos para seguir alimentando las fases previas, comenzando otra nueva fase en el proceso de diseño. Para ello se debe solicitar la colaboración constante de los usuarios, quienes deben poder usar el prototipo por un tiempo prudencial para evaluar nuevas funcionalidades, mejoras deseadas, o sencillamente para detectar errores en el sistema.

5.5. Estrategia

5.5.1. ¿Cascada o Espiral con Prototipos?

Los proyectos de Data Warehousing no son iguales a la mayoría de los proyectos tradicionales. Esto ya es sabido. Mientras que en los proyectos tradicionales se dispone de un conjunto fijo de requerimientos, en los proyectos de Data Warehousing esto no es nunca cierto. Una vez que los usuarios comienzan a ver resultados se les ocurren nuevas consultas que no estaban previstas, nuevas áreas de aplicación.

Para enfrentar esto, creemos que lo mejor es utilizar una metodología que ataque el problema en forma incremental (como por ejemplo el desarrollo en espiral con prototipación), y no una que tenga como base el terminar una etapa para seguir en la siguiente (como la metodología en cascada).

[15] Es mejor el uso de seudocódigo a código SQL puro debido a que el SQL es un lenguaje de bastante bajo nivel, que puede dificultar la comprensión del proceso.

Se entendió que un desarrollo con prototipación es lo más conveniente para acercar las herramientas de explotación al usuario lo más tempranamente posible, y de esta forma obtener un "feedback" rápido de ellos para poder mejorar el sistema.

Debemos remarcar, en este sentido, que ésta es la estrategia que se siguió en este proyecto, y por lo tanto los pasos descritos previamente no se realizaron uno luego del otro en estricta secuencia, sino que fueron realizados en forma alternada.

Esto permitió tener sistemas funcionando pronto – con características mínimas, si, pero con suficiente funcionalidad como para poder ver si se llegaba a cumplir los requerimientos de los usuarios o no.

5.5.2. Espiral con Prototipos

En la estrategia en espiral, se va avanzando en el desarrollo cuando se cumplen ciertas etapas, en general relacionadas con un aumento de funcionalidad. En los sistemas de Data Warehousing, encontramos que las etapas más representativas son las siguientes:

- Bases — Se estudian los problemas separados por base fuente. De esta forma no se precisa tener en mente todas las bases, lo que puede complicar el diseño.
- Problemas técnicos — Si bien las bases no son iguales, los problemas que las afectan y los diseños conceptuales pueden ser similares. Cuando se observa esto, conviene estudiar el problema en conjunto para encontrar una solución genérica y poder reusarla.
- Producto — El prototipo va variando desde varios puntos de vista, como su tamaño, su nivel de refinamiento, la importancia prestada a algunos detalles (por ejemplo de interface), la herramienta utilizada en el desarrollo, etc.
- Funcionalidad — Por supuesto uno de los pasos más importantes del desarrollo es incrementar la funcionalidad básica con la que se comienza para agregarle nuevos requerimientos.

6. Conclusiones

6.1. Consideraciones generales

Durante un año nos dedicamos a estudiar la problemática de Data Warehousing, no desde el punto de vista teórico, sino que intentando descubrir, o redescubrir, los problemas que se presentan desarrollando un Data Warehouse. *Este era el objetivo principal del proyecto, y creemos que fue cumplido*.

Durante la duración del proyecto, se enfrentaron distintas problemáticas, algunas de ellas vinculadas al tema tecnológico, y otras vinculadas principalmente a problemas organizacionales. Los problemas tecnológicos eran parte innegable del desarrollo del proyecto, sin embargo los problemas organizacionales, si bien nos afectaron, no eran de exclusivo resorte nuestro. Dentro de estos últimos podemos citar, por ejemplo, la falta de espacio en disco en el servidor de Oracle (problema que hizo imposible una carga más extensa de datos), la falta de interfase de red de Oracle (por lo cual los primeros cinco meses sólo se pudo trabajar con pocos datos en Access, y luego migrar a Oracle), o la falta de herramientas de consulta, entre los más importantes.

Varios de los problemas anteriores se pueden circunscribir dentro de un gran área que es conseguir financiación para el proyecto, y creemos[16] que este no es un problema aislado, sino que por el contrario, es la regla en lo que a nuestro mercado se refiere. La tecnología de Data Warehousing es demasiado nueva como para ser un punto de inversión en las empresas, y es difícil para las secciones informáticas conseguir que se apruebe una inversión en esta tecnología. Creemos que este problema se va a ir superando en el transcurso de los próximos años, donde mejores tecnologías y herramientas de apoyo se ofrecerán en un mercado más conocedor de las ventajas de disponer de un Data Warehouse.

[16] En base a la experiencia de los integrantes del grupo en otros proyectos privados de Data Warehousing en el país.

6.2. Lo que el proyecto aportó

El proyecto se enmarca dentro de un conjunto de estudios más amplios que realiza el Área de Concepción de Sistemas de Información de la Facultad de Ingeniería. Para estos estudios, realizados por estudiantes de maestría, es indispensable el poder disponer de un caso de estudio real en el que pudieran complementar y perfeccionar los estudios teóricos realizados. Este proyecto brinda ese marco práctico en el que apoyar y verificar sus estudios.

Se vio que no hay una metodología estandarizada para los sistemas de Data Warehousing. Algunos autores proponen alguna, pero enfocada a algún área específica, lo que las hace de difícil aplicación en un problema general. Un aporte del proyecto en este sentido consistió en proponer una metodología, la que se puede encontrar en el Capítulo 5 (Metodología Seguida). La misma es fruto de la experiencia del desarrollo, y de los errores cometidos durante el mismo.

De la misma manera, la metadata del sistema es un tema en el que existen pocos estudios, y eso se nota especialmente en el área de documentación. En este proyecto se realizaron sugerencias sobre la documentación que se debería generar en cada actividad, pero creemos que es un tema que por sí solo merece ser una fuente importante de investigación.

Además de los resultados antes mencionados se puede destacar que el presente informe será publicado en Internet, y parte del mismo (basicamente Tecnología de Data Warehousing, donde se introduce el tema) se encuentra a disposición desde hace varios meses. Esto provocó que el proyecto pudiera ser conocido en todo el mundo, e inclusive que se recibieran mails de personas interesadas en conocer más sobre éste. Un punto recurrente en los comentarios era la necesidad de disponer de más material en castellano sobre Data Warehousing[17].

Desde el punto de vista personal, el proyecto nos involucró en un campo de las Bases de Datos de creciente importancia, que nos era por completo desconocido. Además, el proyecto resultó un elemento fundamental para ingresar a un emprendimiento privado donde se trabajó en varios proyectos de Data Warehousing de empresas del medio; experiencia que influyó en gran medida en el resultado del proyecto.

[17] Por esto mismo se decidió la publicación en el Web del presente informe.

6.3. Trabajo futuro

En todo trabajo inicial sobre un tema siempre quedan elementos por considerar, ideas que se consideran valiosas pero no se tiene el suficiente tiempo para experimentar, más requerimientos que atender, y nuevas puntas de investigación que no estaban previstas en los objetivos. Este proyecto no fue la excepción.

A continuación se detallan los puntos que consideramos deben ser estudiados con mayor detalle, o agregados al estudio realizado.

- Extender el sistema con más requerimientos e información de más bases fuentes:

  Se consideró que la información mínima para integrar era la correspondiente a las actividades de los estudiantes, asignaciones de docentes y presupuesto. Este criterio motivó que fuera excluída información – de la que se conoce su existencia – y que podría ser útil.

  Para un siguiente taller se sugiere integrar las siguientes áreas: electivas, talleres 5, proyectos y publicaciones. Se puede pensar en extender el sistema para el resto de la Facultad.

  Algunos requerimientos de estas áreas son descriptos en el capítulo 3 (El problema a resolver), pero se sugiere estudiarlos más en profundidad. Con el uso del sistema aparecerán nuevos requerimientos para las áreas implementadas.

- Estudiar el problema del plan nuevo:

  Este proyecto se ubicó justo en un punto de inflexión correspondiente al cambio de planes de estudio en la Facultad. Toda la información recibida se refiere a los planes anteriores, por lo que no se tuvo conocimiento de los cambios que requiere el plan nuevo. Por lo que se sabe, éste involucra varios cambios que afectan no sólo a los estudiantes nuevos sino a los ya existentes (como por ejemplo el hecho de que ya no se utiliza más el Número de Estudiante sino que se usa la Cédula de Identidad).

  Esto indica que se debe estudiar cuidadosamente los cambios para que el Data Warehouse siga manteniendo sus propiedades de consistencia e integración.

- Definir mejor ciertos conceptos:

  Se definieron algunos conceptos, como por ejemplo: que significa que un estudiante esté activo, o que un estudiante abandonó la carrera. Estos conceptos no estaban definidos, por lo que la definición que se dio, se basó principalmente en sentido común. Asimismo, un punto importante en los requerimientos de seguimiento de estudiantes es poder definir perfiles de estudiantes más significativos que las 2 etapas en que los divide Bedelía.

  El hecho de poder disponer de un primer prototipo hace que en un nuevo proyecto sea más sencillo poder encontrar definiciones satisfactorias para estos conceptos.

- Investigar nuevas tecnologías para interfaces:

  En este proyecto se trabajó con herramientas OLAP, y herramientas de consulta sobre la base relacional. Existen otros tipos de herramientas, como por ejemplo de Data Mining y Simulación que sería bueno experimentar.

  Se pueden obtener resultados interesantes por el lado de interfaces en el Web, es decir llevando los resultados del Data Warehouse a Internet o Intranet.

  Otro punto interesante es la conexión del Data Warehouse con otro tipo de sistemas, como lenguajes de programación orientados a interfaces (por ejemplo Visual Basic), interfaces geográficas o geoespaciales, sistemas de imágenes, plataformas orientadas a objetos (por ejemplo Corba), entre otros.

- Investigar nuevas tecnologías para extracción e integración de datos:

  Todos los mecanismos de extracción, limpieza e integración utilizados, fueron programados. Los mecanismos consisten en scripts sql, o programas en C.

  Sería interesante trabajar con herramientas de extracción e integración, que en este proyecto no estuvieron al alcance por un tema económico.

  Asimismo se podría probar otros mecanismos programados, o semiprogramados, para trabajar con otras fuentes de datos más heterogéneas, no relacionales, de imágenes, geográficas, etc.

- Profundizar en la formalización y documentación:

  Se propuso una metodología para enfrentar el desarrollo de un sistema de Data Warehousing. Esta metodología se basó en la experiencia práctica de este proyecto. Es necesario enriquecer la misma con conclusiones extraídas de desarrollos teóricos y de otras experiencias prácticas.

  Es necesario, sobre todo profundizar en vías más formales de especificación para las diferentes actividades del desarrollo, y el intercambio de información entre ellas.