Proyecto de grado

Extracción y análisis de información en bases de datos documentales

Integrantes

Brian Hill

Diego Ricca

Francisco Álvarez

Juan Martín Valsangiacomo

Tutora:

Libertad Tansini

Montevideo, Uruguay
2014

Resumen

Este trabajo responde a la necesidad de extraer información de bases de datos documentales con el fin de poder analizar dicha información. Las bases de datos documentales difieren de las conocidas bases de datos relaciones ya que disponen de un modelo de registro flexible, esto es porque permite a cada registro de datos estar compuesto por campos de longitud variable, campos con múltiples valores, cantidades de campos variables, etc. Esta flexibilidad agrega dificultad a la extracción y la estructuración de los datos requerida para su análisis.

Por otro lado, un servicio requerido por las organizaciones es la gestión documental (Bedoya 2014) que se define como el conjunto de normas técnicas y practicas usadas para administrar el flujo de documentos de todo tipo en una organización, permitir la recuperación de información desde ellos, determinar el tiempo que los documentos deben guardarse, eliminar los que ya no sirven y asegurar la conservación indefinida de los documentos más valiosos. Es una actividad casi tan antigua como la escritura, que nació debido a la necesidad de "documentar" o fijar actos administrativos y transacciones legales y comerciales por escrito para dar fe de los hechos.

Luego realizar una búsqueda acerca de herramientas disponibles en el mercado específicas para el análisis de información en bases de datos documentales, no encontramos ninguna que responda a la necesidad que da lugar a este proyecto.

El proyecto se enmarca en la empresa ISA Ltda. (ISA Ltda. 2014) la cuál es propietaria de un sistema desarrollado sobre bases de datos documentales. Dicho sistema es utilizado por importantes organismos públicos y empresas, hace ya varios años, generando información útil que no está siendo aprovechada. La empresa desea analizar estos datos en una herramienta de análisis de información y un requerimiento es no comprometerse con herramientas propietarias, por lo cual todo el proyecto se basa en herramientas Open Source (Código Abierto 2014). De igual manera la solución aquí propuesta no se liga a ninguna herramienta de análisis de información en particular.

Este proyecto tiene 2 objetivos principales, el primero consiste en realizar una investigación sobre herramientas Open Source existentes en cuanto a análisis de información, en particular se focalizó sobre herramientas Business Intelligence (BI) (Díaz 2012). Esta etapa tiene como resultado un documento con dicho estudio, comparando las mismas entre si y dejando como conclusión la elección de Pentaho (Pentaho 2014), debido a la facilidad de instalación y configuración, y opciones de trabajo que ofrece (ver sección 4).

Para la segunda etapa se construyó una aplicación capaz de extraer datos de sistemas construidos sobre bases de datos documentales, para ser analizada mediante herramientas de análisis de información utilizando la herramienta seleccionada en la etapa anterior. La aplicación fue desarrollada en dos grandes módulos.

Un módulo para la extracción, transformación y carga de los datos (ETL) (ETL 2014) el cuál cumple la función de recorrer los datos contenidos en las bases de datos, transformarlos y darle una estructura determinada para cargarlos en un Data Warehouse (DW) (Peralta 2001) el cuál sirve como entrada para la herramienta de análisis de datos como Pentaho. En el diseño de este módulo dedicamos especial atención e invertimos muchas horas de trabajo, ya que esta herramienta es el pilar fundamental de nuestro proyecto. Si podemos diseñar una herramienta que extraiga información de bases de datos no relacionales, transformarla e ingresarla a una base

de datos relacional según un modelo recibido como parámetro, el resto del problema (hacer un modelo multidimensional para analizar información que reside en un DW) es un problema ya conocido y más fácil de resolver.

El otro módulo es una aplicación web la cual hace posible la especificación de qué datos se extraerán y como se relacionaran entre sí para generar información útil. Ésta aplicación web presenta ventajas en cuánto a flexibilidad, reúso y mantenimiento de la especificación de la información a ser extraída.

Como caso de estudio recurrimos al Banco de Seguros del Estado (BSE 2014), respondiendo a sus necesidades reales en base al uso del sistema, contando con su colaboración en interés en el proyecto. Realizamos pruebas con los datos brindados por el BSE, los cuales fueron procesados por la aplicación desarrollada obteniendo resultados muy positivos y una devolución alentadora de parte del BSE.

Índice

1 Introducción

En la actualidad las organizaciones utilizan sistemas de información informáticos para gestionar sus procesos. Los componen el hardware necesario para utilizarlo, el software que simula los procesos y las personas que interactúan con él. Al ser utilizados generan grandes cantidades de información que son almacenados en bases de datos que brindan servicios que ayudan a mantenerla, como lo son independencia de los datos con su tratamiento, restricciones de seguridad al acceso/modificación de los datos, etc.

La empresa ISA ha desarrollado durante diez años una plataforma para la gestión de documentación electrónica (iGDoc) (ISA Ltda. 2014). Dicha plataforma cuenta con varios módulos: formularios, expedientes, documentos, contratos, etc.

Este plataforma, que de ahora en más será llamada iGDoc, es un sistema implementado sobre bases de datos documentales, más específicamente sobre IBM Lotus Domino (IBM 2014).

Las bases de datos documentales, cómo se verá en detalle en la sección 2, son de gran utilidad para la implementación de sistemas poco estructurados y brindan la posibilidad de realizar búsquedas por texto, sin la necesidad de agregar una sola línea de código.

En Uruguay no son muy conocidas, pero paradójicamente, son bastante utilizadas en el ámbito público y privado para la gestión de documentación electrónica, como por ejemplo: BSE, Banco Santander, Intendencia de Montevideo, BanRed, Secretaría del Mercosur, entre muchos otros.

Los usuarios de iGDoc generan diariamente formularios electrónicos, expedientes electrónicos, documentos electrónicos, entre otros. Algunos de los documentos generados, por ejemplo: renovación de la licencia de conducir, solicitar licencias de trabajo, digitalizar documentos en papel para poder buscarlos por meta-tags, etc.

Cómo se dijo anteriormente, iGDoc está instalado en organizaciones gubernamentales y no gubernamentales en Uruguay y en el exterior. Actualmente en su totalidad el sistema es utilizado por 15.000 usuarios, se generan 200.000 nuevos documentos y se producen 900.000 movimientos por mes, a la fecha se estima que se han digitalizado 20:000.000 de documentos.

El análisis de la información en sistemas de datos documentales (expedientes electrónicos, formularios electrónicos, documentos electrónicos, etc.) resulta vital para las organizaciones. La eliminación del papel cumple con un objetivo primario, pero el gran objetivo siempre es mejorar la gestión. El cliente (ISA Ltda.) es consciente de esto y por lo tanto necesita brindar este servicio con las dos opciones, con licenciamiento y sin licenciamiento, de manera de no atarse a ningún proveedor para poder brindar el servicio requerido.

En estos momentos la empresa ISA Ltda. se encuentra desarrollando un sistema para el análisis de información con una herramienta con licenciamiento pago. Para complementar el desarrollo y poder brindar mejor servicio quiere proveer una solución con licenciamiento libre.

Por lo tanto, los objetivos de este proyecto son: la investigación de herramientas para el análisis de datos con licenciamiento libre y Open Source, y el desarrollo de una aplicación para la extracción, la transformación y el análisis de información de un sistema implementado sobre bases de datos documentales.

1.1 **Problema planteado**

La empresa ISA Ltda. ha desarrollado un sistema de gestión documental llamado iGDoc, el cual está implementado sobre un servidor de bases de datos documentales, Lotus Domino. Dicho producto comprende una plataforma que brinda servicios para la Gestión Electrónica de Documentos a diversas formas documentales, como son formularios, expedientes, escaneos digitalizados de documentos, entre otros. Estas formas documentales tienen en común requerir de un repositorio de documentos, numeración centralizada, niveles de seguridad para garantizar su acceso y confidencialidad, posibilidad de ser firmados digitalmente, localizarlos mediante un buscador de texto completo y de contenidos y además ser accesibles desde un navegador Web. En particular se puede ver en mayor detalle en el Anexo 1 iGDoc.

Hace varios años el producto se expandió por todo el país y Latinoamérica, sumando una gran cantidad de clientes que lo utilizan. El sistema iGDoc permite almacenar y administrar una gran cantidad de datos los cuales se van incrementando durante el tiempo de funcionamiento de la aplicación. Existen datos que brindan importante información, como el tiempo desde que el documento llega a un sector de la empresa, hasta que el encargado empieza a trabajar con él. Este dato no se utiliza en la actualidad y sería de gran utilidad poder visualizar posibles cuellos de botella en el flujo del documento y así mejorar la organización y los resultados en las empresas/organizaciones que la utilizan, logrando así mayor eficiencia.

iGDoc pretende de ser lo más genérico posible pero los distintos clientes modifican la estructura de los datos dependiendo de los trámites que requieren sus organizaciones. Un ejemplo es el de los formularios, que es uno de los módulos de los que está compuesto. Un formulario por ejemplo puede contener 2 campos y transitar por un solo sector de la empresa, o puede contener 200 campos, y transitar por 20 sectores de la empresa. Esto deriva en que los datos y su estructura son distintos en cada cliente que utiliza iGDoc, haciendo difícil el manejo de esos datos por procesos automáticos, como puede ser la generación de reportes.

Del relevamiento surge que iGDoc brinda los servicios de reporte con un módulo de reportes, el cual permite obtener listados a demanda, pero tiene deficiencias notorias, ya que cada vez que un cliente solicita un nuevo tipo de reporte debe pedirle a ISA Ltda. que lo implemente.

En cuanto al análisis de datos de tipo BI, explicado en la sección 2.5, en el último tiempo se le ha incorporado a iGDoc un servicio OLAP (Thomsen 2014), pero solamente fue desarrollado para el módulo de expedientes. Está exclusivamente ligado a éste módulo, siendo difícilmente adaptable para la utilización del servicio en otros módulos de iGDoc, como formularios, comunicaciones, etc. Además éste servicio OLAP, utiliza un producto propietario llamado O3, perteneciente a la empresa IdeaSoft (Ideasoft 2014), por lo que es necesario que sus clientes adquieran la licencia de dicha plataforma.

1.2 **Objetivos**

El objetivo primario es desarrollar una herramienta para extraer datos de iGDoc, permitiendo ingresar los mismos a una herramienta de análisis de datos.

Para esto es deseable maximizar la flexibilidad de qué datos se desean extraer y cómo relacionarlos entre ellos, con el objetivo de brindar análisis de la información útil a través de la herramienta de análisis de datos. Ésta flexibilidad implica que las

decisiones puedan tomarse a nivel de usuario, con la menor necesidad posible de ingresar nuevas porciones de código.

Se requiere que la solución no se ligue a una herramienta de análisis de datos específica, por lo tanto el formato de los datos preparados para ingresarse en la herramienta debe posibilitar la utilización de cualquier otra herramienta. Por ejemplo una con licencia paga. Para esto se deberán investigar los productos disponibles para poder tener una visión de que ofrece cada uno y poder elegir uno de ellos para este sistema, según el que mejor se adapte a los requerimientos importantes. Estos fueron definidos durante el relevamiento, como por ejemplo el manejo de los indicadores, performance a la hora de presentar los datos, capacidad de generar informes, facilidad de uso y aprendizaje, etc.

Otro objetivo es utilizar únicamente herramientas Open Source y con algún tipo de licenciamiento libre de manera de eliminar la necesidad de que el usuario final deba adquirir una licencia paga para utilizar la herramienta resultado de éste proyecto. Cabe aclarar que iGDoc es una aplicación propietaria de la empresa ISA Ltda.

También se desea desarrollar un prototipo, que muestre el flujo completo de los datos, que va desde la extracción de iGDoc hasta la presentación de los resultados en la herramienta de análisis de datos.

La solución propuesta no debe ligarse a ningún módulo de iGDoc en particular, aunque si se probará solo para el módulo de formularios, que es uno de los más complejos. Eso se debe a que los formularios no tienen una estructura fija y su flujo de trabajo puede variar dependiendo del tipo de formulario y de los datos ingresados en el mismo. Para más información de este módulo, ver el Anexo 1.

Se tiene como objetivo adicional el desarrollar una herramienta web que permite a los clientes configurar la herramienta de extracción desarrollada, para especificar cómo se debe realizar la carga de datos (fuentes de datos, campos de donde se obtienen los valores, como se deben realizar los cálculos, etc.). Además se deberá poder configurar la herramienta de análisis de datos elegida. Esta herramienta web aporta un alto nivel de abstracción de los datos y permite inferirlos, modificarlos, reusarlos, etc.

Finalmente se pretende evaluar dicho sistema con uno de los clientes de ISA, BSE utilizando datos reales para obtener una evaluación real del producto, en cuánto a satisfacción. También se pretende realizar pruebas de estrés, con grandes volúmenes de datos dado que muchos de los clientes poseen en sus sistemas miles de documentos como se menciona en la sección 1.

1.3 **Contenido del documento**

Este documento está compuesto por 10 capítulos. En el capítulo 1 se realiza una introducción general del proyecto, describiendo el contexto en el que se desarrolla, cual es el problema que se plantea y sus objetivos. Los conceptos importantes utilizados, los describimos en detalle en el capítulo 2. En el capítulo 3 se explica la etapa de Análisis, donde se especifican los requerimientos y el alcance. La investigación de las herramientas Open Source de análisis de datos se encuentra en el capítulo 4 con los resultados obtenidos y conclusiones. La etapa de diseño se describe en el capítulo 5 donde se especifica la arquitectura utilizada y el resto de las decisiones de diseño (diagramas de clases, etc.). En el capítulo 6 se explican las decisiones tomadas en la etapa de implementación y las herramientas utilizadas. Se presentan los resultados del proyecto en el capítulo 7. En el capítulo 8 se presenta el trabajo a futuro, esto se refiere a funcionalidades y mejoras que se pueden agregar en el futuro para mejorar aún más la solución. En el capítulo 9 están las conclusiones del

proyecto. El capítulo 10 es el glosario donde se explican las siglas y en el capítulo 11 están las referencias a los conceptos mencionados a lo largo de este documento. El capítulo 10 contiene los índices de las figuras e imágenes utilizadas en el documento. Finalmente comienza la sección de anexos con información adicional del proyecto, en la cual se explica más en detalle el sistema iGDoc, el relevamiento de las herramientas de análisis encontradas y la ejecución del proyecto.

2 Marco conceptual

2.1 Bases de datos documentales

2.1.1 Introducción a base de datos documentales

En esta sección se explica el concepto de base de datos documental (Yunta 2001), basándose en algunos conceptos que pertenecen a la implementación del sistema Lotus Domino, que es sobre el cual se trabaja en este proyecto.

Una base de datos documental está constituida por un conjunto de programas que almacenan, recuperan y gestionan datos de documentos o datos de algún modo estructurados. Este tipo de bases de datos constituyen una de las principales sub-categorías dentro de las denominadas bases de datos no SQL (G. Quintana 2008). A diferencia de las bases de datos relacionales, estas bases de datos están diseñadas alrededor de una noción abstracta de “documento”.

Bajo éste concepto de base de datos se pierde el concepto de tupla y tablas. Los atributos están relacionados entre sí, solo para el documento en cuestión. Un documento puede tener una cantidad potencialmente no acotada de atributos y éstos pueden agregarse o quitarse al documento en tiempo de ejecución. No existe una estructura fija para los documentos por lo tanto podrían existir infinitos tipos de documentos en cuánto a los atributos que posee, tamaño, etc.

Cada documento almacena los siguientes datos entre otros: fecha y hora de creación, fecha y hora de última modificación, una lista de atributos, usuario que lo creó y usuario que efectuó el último cambio.

Cada atributo de cada documento, es descripto por siguientes datos entre otros: nombre, tipo, valor y tamaño en bytes.

Una base de datos documental en Domino, contiene tanto los documentos con los datos, como los elementos de diseño que permiten manipularlos, crearlos, ordenarlos, etc. Los elementos de diseño pueden ser entre otros, vistas, formularios, agentes, bibliotecas de script, consumidores y proveedores de Web Services (Newcomer 2002). Todos los nombrados anteriormente serán explicados a continuación uno a uno.

Una vista permite agrupar documentos según un criterio especificado en su definición. Las vistas, ordenan los documentos en un orden definido, según la primera columna seteada en la misma, esto permite hacer búsquedas de documentos en orden 1 sobre la vista.

Los formularios posibilitan especificar código HTML (HTML 2014) el cual permitirá crear aplicaciones web. Por ejemplo, todo lo que se ve al abrir iGDoc, está definido en un formulario. A estos elementos se les especifica un parámetro para indicar si van a crear o editar un documento existente. Brindan facilidades como relacionar los campos de ingreso de texto del formulario con un atributo de un documento.

Los agentes encapsulan código tanto Java (ORACLE 2014) como LotusScript (IBM 2005) para manipular, referenciar, crear o eliminar documentos mediante código. Un ejemplo podría ser un agente que recorra todos los documentos creados el día de la fecha (ya que estaría programado para correr diariamente) y envíe por correo electrónico el listado de dichos documentos. Se utilizan frecuentemente para hacer procesos que actúen sobre varios documentos o ejecutar alguna tarea particular. Pueden ser agendados para que corran en determinado momento o pueden ser

llamados por otros agentes, mediante una invocación HTTP (HTTP 2014), o por medio de Ajax (AJAX 2014), desde HTML.

Bibliotecas de script, pueden ser escritas tanto en Java, Lotus Script, o JavaScript (JavaScript 1995). Nos permiten encapsular código de clases, pudiendo instanciarlas desde otras bibliotecas, o agentes.

Para trabajar con Web Services contamos con **consumidores y proveedores**, los cuáles facilitan el uso de los mismo implementando las clases proxy necesarias para usarlos a partir de un WSDL que define el Web Service (consumidor), o generando el mismo en base a una firma de operación dada (proveedor).

2.1.2 Ventajas de bases de datos documentales

La principal ventaja es que permite una estructura muy flexible para los datos. Los documentos pueden ser alterados en su composición más básica de un momento a otro. Esto se separa de la noción de tuplas en bases de datos relacionales, donde mayormente se comparte una estructura para las tuplas de una misma tabla, en las bases documentales, los documentos no comparten estructura alguna y podría no haber dos documentos con la misma estructura.

A diferencia de las bases de datos relacionales, la falta de esta estructura permite un mejor desempeño y mayor facilidad de desarrollo en sistemas que crecen mucho y cuando los datos no son del todo fijos, ya que por ejemplo se les pueden agregar atributos en cualquier momento. Por ejemplo en sistemas de gestión documental, formularios electrónicos, gestión de expedientes, etc.

Si los sistemas son muy grandes, se pueden distribuir las bases de datos en diferentes servidores. Lo que Domino hace por defecto al ejecutar búsquedas o al ejecutar un agente, lo hace distribuyendo el procesamiento en los distintos servidores maximizando la performance general de la aplicación ya que utiliza por separado el poder de procesamiento de cada servidor.

2.1.3 Desventajas de bases de datos documentales

La alta flexibilidad que permite éste tipo de base de datos hace a se torne engorroso estructurar los datos, como es requerido en nuestro proyecto, para poder ingresarlo a la herramienta de análisis de información. Es complejo estructurar datos que no están naturalmente estructurados.

Se detecta la gran desventaja en la implementación de la concurrencia cuando se quiere guardar datos sobre un documento. Por ejemplo: si dos usuarios están editando al mismo tiempo un documento, al querer grabar los datos en la base de datos documental, se crea (por defecto) lo que se denomina conflicto de grabación, creándose X documentos réplicas (a partir del mismo documento), cada uno con la información grabada por cada usuario que editó concurrentemente el documento más la información que ya tenía el mismo. El primero que guarda mantiene el documento "original" y los siguientes que guarden crean los conflictos de grabación. Esto nos trae el problema de que no todos los datos contenidos en las bases de datos son relevantes, por lo tanto hay que tenerlo en cuenta para hacer una limpieza antes de ingresarlos a la herramienta de análisis de información.

2.1.4 Lotus Notes (Domino)

IBM Notes (anteriormente Lotus Notes) es un sistema software cliente/servidor de colaboración y correo electrónico, desarrollado por Lotus Software, empresa que a posteriori la absorbiera IBM.

La parte del servidor recibe el nombre Lotus Domino, mientras que el cliente se llama Lotus Notes.

El servidor dispone de versiones para distintas plataformas, incluyendo Windows NT, Windows 2000, Windows 2003, Linux de distintas distribuciones, HP-UX y Solaris, i5OS (antes OS/400) y z/OS.

El cliente Lotus Notes dispone de versiones nativas para Windows y Mac OS (9 y X; siendo Universal desde la versión 8.5), y para Linux (a partir de la versión 7.0.1 (alrededor del 2006), aunque anteriormente era posible ejecutarlo a través de WINE)

Lotus Domino Notes es un sistema de comunicación el cual permite enviar correo electrónico y manejo de Calendarios y Agendas. También es una plataforma de colaboración que permite compartir bases de datos con información, como por ejemplo bases documentales, de procedimientos, manuales o foros de discusión. Y finalmente es una plataforma de Coordinación - utilizando aplicaciones Notes con flujo de trabajo. Ejemplo de ello sería cualquier proceso de una empresa que requiere que un documento fluya entre varias personas o departamentos para su autorización, como por ejemplo una solicitud de vacaciones, solicitud de anticipo de viáticos y cuentas de gastos, etc. Todo esto es susceptible de manejarse de forma electrónica mediante Lotus Notes.

2.2 Data warehouse

Uno de los activos más importantes de cualquier organización es su información. Este activo casi siempre se utiliza con dos fines: el mantenimiento de registros operacionales y la toma de decisiones analítica. En pocas palabras, los sistemas operacionales son donde se guardan los datos, y el sistema data warehouse (DW) (Mark Humphries 1999) es donde se obtiene los datos de salida.

En este proyecto trabajamos sobre el contexto BI, donde los usuarios del sistema operacional son los que mueven las ruedas de la organización. Toman órdenes, registran nuevos clientes, monitorean el estado de las actividades operacionales, registran quejas, entre tantas otras tareas. Los sistemas operacionales están optimizados para procesar rápidamente estas transacciones. Ellos previsiblemente realizan las mismas tareas operativas una y otra vez, ejecutando los procesos de negocio de la organización. Teniendo en cuenta este enfoque de ejecución, los sistemas operacionales no suelen mantener historial, sino que se actualizan los datos para reflejar el estado más actual.

Por otra parte, los usuarios del sistema DW, observan los datos de la organización para evaluar performance. Necesitan grandes cantidades de información detallada para poder realizar dichas evaluaciones. Estos sistemas están optimizados para consultas de alta performance debido a que las preguntas evaluativas que realizan los usuarios normalmente requieren que se realicen búsquedas entre miles o millones de transacciones comprimiéndolas en un conjunto respuestas. Todo esto lleva a la necesidad de que se preserve un historial para poder evaluar de forma precisa la performance de la organización a través del tiempo.

Existen múltiples definiciones del término data warehouse, de los cuales el más popular y adecuado para este proyecto, es la de **Bill Inmon** (Data Warehouse Definition 2014) que lo define de la siguiente manera: “data warehouse es un conjunto de datos integrados, históricos, variantes en el tiempo y unidos alrededor de un tema específico, que es usado por la gerencia para la toma de decisiones”, también lo define en términos de las características del repositorio de datos:

- **Orientado a temas** quiere decir que los datos en la base de datos están organizados de manera que todos los elementos de datos relativos al mismo evento u objeto del mundo real queden unidos entre sí.
- **Variante en el tiempo** implica que los cambios producidos en los datos a lo largo del tiempo quedan registrados para que los informes que se puedan generar reflejen esas variaciones.
- **No volátil** implica que la información no se modifica ni se elimina, una vez almacenado un dato, éste se convierte en información de sólo lectura, y se mantiene para futuras consultas.
- **Integrado** expresa que la base de datos contiene los datos de todos los sistemas operacionales de la organización, y dichos datos deben ser consistentes.

2.2.1 Entorno de un sistema data warehouse

La estructura básica (véase Figura 2.1) incluye:

1. **Datos operacionales** se refiere a uno o más fuentes de datos desde donde se extraen los datos.
2. **ETL (Extracción, transformación y carga)**
    - **Extracción de datos** es la selección sistemática de datos operacionales usados para poblar el componente de almacenamiento físico.
    - **Transformación de datos** son los procesos para sumariar y realizar otros cambios en los datos operacionales para reunir los objetivos de orientación a temas e integración principalmente.
    - **Carga de datos** es la inserción sistemática de datos en el componente de almacenamiento físico DW.
3. **Data warehouse** es el almacenamiento físico de datos de la arquitectura data warehousing.
4. **Herramientas de acceso al componente de almacenamiento físico** son aquellas herramientas que proveen acceso a los datos.

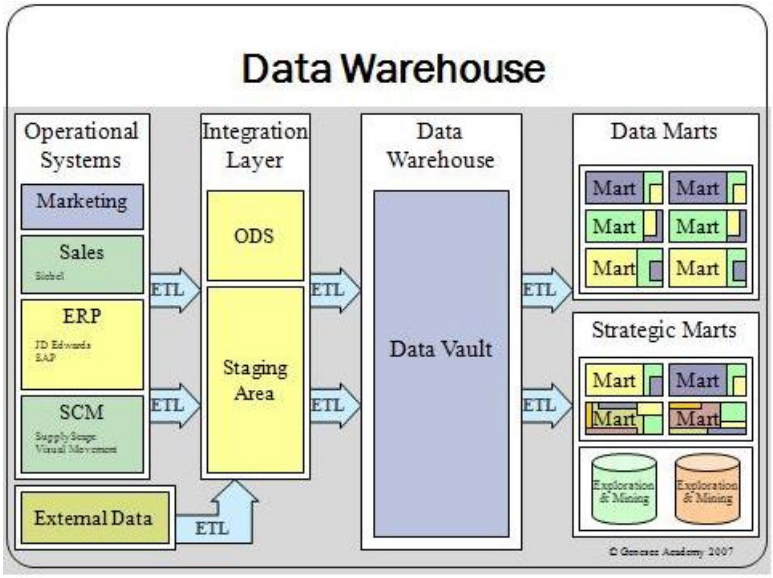


Figura 2.1 - Estructura básica de un DW (Descripción data warehouse n.d.)

2.2.2 Modelado dimensional

Modelado dimensional (Modelado Dimensional 2014) es ampliamente aceptado como la técnica preferida para presentar datos analíticos debido a que aborda simultáneamente dos requerimientos:

- Entregar datos que sean comprensibles por los usuarios del negocio.
- Disponer de un rendimiento rápido en las consultas.

Modelado dimensional es una conocida técnica para la simplificación de bases de datos. Simpleza es necesaria debido a que asegura que los usuarios puedan entender los datos, así como permitir que el software pueda navegar y entregar los resultados de forma rápida y eficiente.

Como forma de ejemplo, imagine un ejecutivo que describe su negocio como, “Vendemos productos en varios mercados y medimos nuestra performance a través del tiempo.” Diseñadores dimensionales captan el énfasis en los términos producto, mercado, y tiempo. La mayoría de la gente encuentra intuitiva imaginarse al negocio como un cubo de datos, donde las aristas están etiquetadas con producto, mercado, y tiempo. Imagine rebanando un cubo a través de dichas dimensiones. Los puntos dentro del cubo es donde las medidas, también conocidas como indicadores, tal como volumen de ventas o ganancias, para dicha combinación de producto, mercado, y tiempo están almacenadas.

Las dimensiones se pueden organizar en jerarquías de agregación para representar diferentes niveles de análisis, donde cada nivel se le denomina nivel de agregación. Un ejemplo de esto se puede observar en la Figura 2.2 donde por ejemplo se requiere analizar el total de la medida ventas para cada producto por ciudad, estado y finalmente por país, entonces en la dimensión Producto se utiliza una jerarquía de agregación Ciudad, Estado y País.

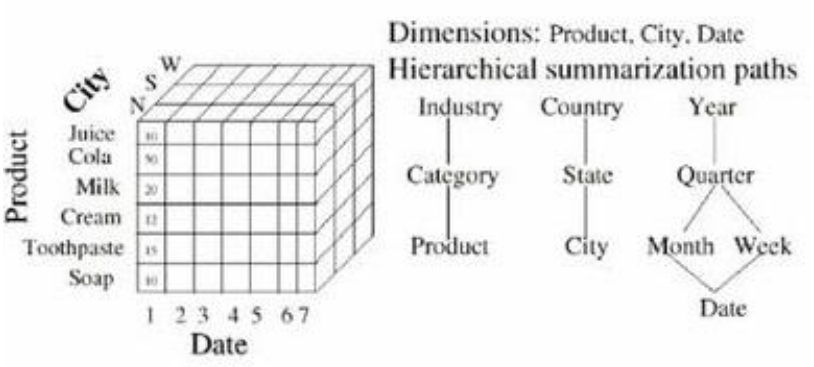


Figura 2.2 - Cubo y sus jerarquías de agregación (Definiciones de Conceptos - Base de datos Estratégicas n.d.).

Existen operaciones para navegar por la información contenida en un modelo dimensional. Las más comunes que se pueden realizar sobre los cubos son:

- **Roll up** se incrementa en el nivel de agregación de los datos (véase Figura 2.3).
- **Drill down** se incrementa en el nivel de detalle, caso opuesto a roll up (véase Figura 2.3).
- **Slicing** es la selección de un grupo de celdas de la totalidad de una matriz multidimensional especificando el valor para una o más dimensiones.
- **Dicing** implica seleccionar un subconjunto de celdas especificando un rango de valores de un atributo.

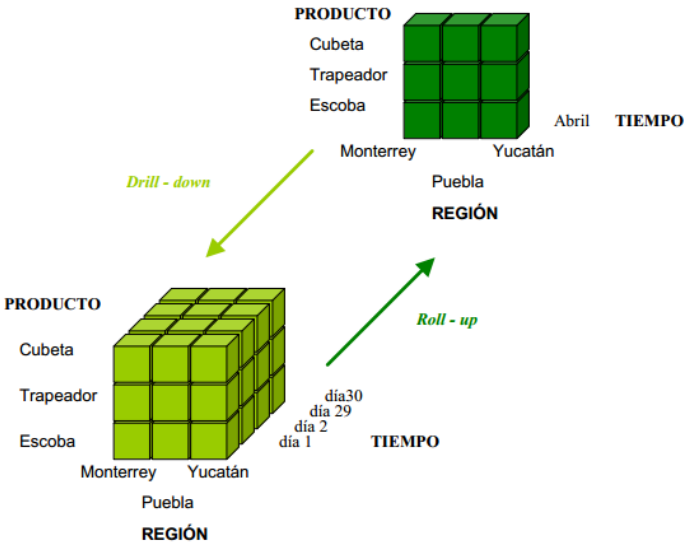


Figura 2.3 - Operadores drill-down y roll-up.

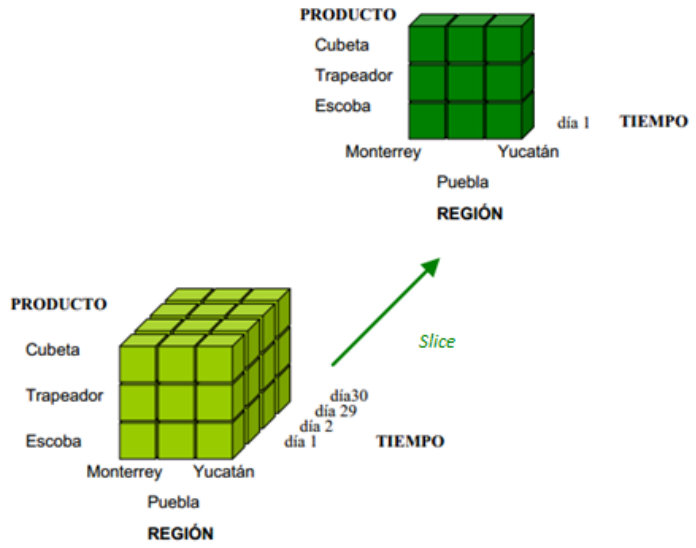


Figura 2.4 - Operador Slice

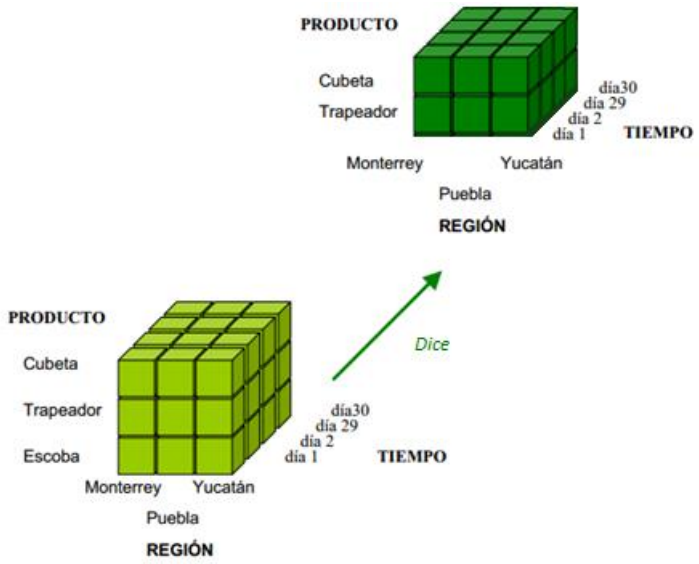


Figura 2.5 - Operador Dice

Roll-up consiste en la agregación de los datos que se puede acumular y calcular en una o más dimensiones. Por ejemplo, acumular todas las ventas de todas las oficinas para sumar el total de ventas de un departamento o división. Por el contrario, el drill-down es una técnica que permite a los usuarios navegar a través de los detalles. Por ejemplo, los usuarios pueden ver las ventas por productos individuales que componen las ventas de una región. Slicing and dicing es una utilidad por la cual los usuarios pueden sacar (slicing) un conjunto específico de datos del cubo OLAP y ver (dicing) las rebanadas de diferentes puntos de vista.

2.3 **Business Intelligence**

Debido al rápido avance y desarrollo de la tecnología en las últimas cuatro décadas, se hace cada vez más común que las organizaciones utilicen sistemas de información informáticos para gestionar sus operaciones. Estos este tipo de sistemas al ser utilizados a lo largo del tiempo, generan información acerca de las operaciones del negocio. El problema es que en muchos casos esa información no es aprovechada en

su totalidad. Los datos generados, por si solos, no constituyen información relevante para la toma de decisiones. Se requiere, entonces, aplicar algún conjunto de técnicas y herramientas de análisis que conviertan esos datos en información estratégica para el negocio. Adicionalmente, la información debe presentarse de manera que sea fácilmente accesible para las personas encargadas de la toma de decisiones.

El concepto BI se refiere a un conjunto de tecnologías utilizadas para transformar datos en información comprensible y útil para la realización de análisis de negocio. El objetivo principal de BI es el de ofrecer conocimientos para respaldar las decisiones organizacionales y empresariales. Con el uso de aplicaciones BI, la gran cantidad de datos originados en muchos formatos diferentes pueden ser unificados de manera rápida y con mayor precisión. Con esta información más sencilla de entender y analizar, los encargados de la toma de decisiones pueden efectuar cambios y desarrollar planes para lograr mejoras en sus organizaciones. Debido al incremento en los datos a nuestra disposición a menudo se hace difícil entender que significan. El avance de la tecnología no parece tener fin, pero la tecnología por sí sola no es la solución al problema. Para avanzar con sus estrategias, las organizaciones necesitan personas que sean capaces de tomar decisiones tácticas para solucionar los problemas de negocio.

Para tomar decisiones rápidas y mejores, los gerentes y ejecutivos necesitan información en forma de datos útiles que estén disponibles y sean fáciles de analizar. La brecha que existe entre la información del negocio, las necesidades de las personas y los datos recogidos por las empresas se incrementa rápidamente a lo largo del tiempo y a medida que crece el volumen de la información recogida. Para sobrepasar esta brecha, las organizaciones deben reconocer cómo los grandes volúmenes de datos pueden ser utilizados para analizar el negocio y de qué manera pueden ser convertidos en información útil. A través de una serie de procesos que comienzan con la extracción de datos y terminan con sistemas BI se puede sobrepasar esta brecha.

Para realizar informes y analizar datos, los sistemas BI pueden utilizar soluciones de OLAP. Estos sistemas proporcionan facilidades para cruzar información, visualizar datos en múltiples dimensiones y navegar a través de jerarquías. Las capacidades OLAP estándar permiten al usuario final acceder de manera rápida a las respuestas de sus preguntas. Se detallarán este tipo de sistemas en la siguiente sección del documento.

A menudo los sistemas BI utilizan como fuente de información a analizar datos almacenados en un DW. Para esto las soluciones BI deben proporcionar integraciones con este tipo de fuente de datos.

El término BI es utilizado para agrupar una serie de tecnologías, plataformas de software, aplicaciones específicas y procesos. El concepto de BI presenta tres perspectivas importantes

- El uso de un enfoque razonable para la gestión
- Los datos pueden ser convertidos en información útil
- La toma de decisiones de manera eficiente

La mayoría de las herramientas BI poseen una capa de presentación que permite a los usuarios finales crear reportes. Cuando se evalúan las herramientas de reportes se buscan las siguientes capacidades

- Conexión a la fuente de datos (Data Source)
- Automatización y distribución
- Seguridad
- Personalización
- Exportación de Datos

2.4 On-line Analytical Processing (OLAP)

OLAP es una tecnología utilizada para crear sistemas de ayuda para la toma de decisiones. Es una categoría dentro de la tecnología de software que permite a los analistas, gerentes y ejecutivos introducirse rápidamente, de manera consistente e interactiva a los datos extraídos y transformados que reflejan la realidad de la organización.

OLAP permite a los usuarios analizar rápidamente la información que ha sido resumida en jerarquías y vistas multidimensionales. Simplificando las consultas previstas en vistas multidimensionales antes del tiempo de ejecución, las herramientas OLAP nos benefician con un incremento en la performance sobre el acceso convencional a las bases de datos. La mayoría de los cálculos que son requeridos para simplificar los datos son realizados antes de que se realice la consulta.

Como un componente más en los sistemas para tomas de decisiones (como por ejemplo sistemas BI), OLAP interactúa con otros componentes como pueden ser DW para asistir al analista.

Los DW usualmente almacenan todos los datos obtenidos de múltiples fuentes de datos como una base de datos convencional. Los datos son limpiados y transformados en un formato consistente y común antes de ser almacenados en el DW. A diferencia de las bases de datos convencionales los datos almacenados no varían constantemente, sino que estos son refrescados periódicamente.

OLAP por sí solo no analiza la información pero si, ayuda a los usuarios a observar la información interpretando los resultados de las consultas. Utilizando OLAP, los analistas pueden obtener todo tipo de patrones y tendencias en lugar de obtener simplemente conocimiento presentado en formularios fijos. En una sesión OLAP típica, el analista plantea ciertas consultas sobre los datos seleccionados. Los sistemas OLAP pueden retornar generalmente el resultado en solo segundos, incluso si la consulta involucra una cantidad importante de registros.

2.4.1 Beneficios del OLAP

La popularidad que han adquirido las herramientas OLAP se debe a la capacidad que poseen de presentar información relevante y coherente. Estos sistemas ayudan a revelar inconsistencias y tendencias en los datos que no pueden ser vistas en los sistemas a simple vista. Los usuarios pueden visualizar y analizar datos que han sido resumidos y consolidados mediante la estructura OLAP. Otros beneficios que poseen las herramientas OLAP son los siguientes

- Acceso rápido, realización de cálculos y capacidad de resumen sobre los datos de la organización
- Soportan múltiples y usuarios y múltiples consultas

- Poseen la habilidad de manejar múltiples jerarquías y múltiples niveles de datos
- Poseen la habilidad de pre-resumir y consolidar los datos para realizar consultas más rápidas y desplegar reportes
- Poseen la habilidad de expandir el número de dimensiones y los niveles de los datos a medida que la organización crece

2.4.2 Variaciones del OLAP

MOLAP (multidimensional on-line analytical processing) es un tipo del OLAP que almacena los datos en estructuras de bases de datos multidimensionales. Los cubos MOLAP son más adecuados para el reordenamiento y partición de los datos y son utilizados principalmente cuando la velocidad de las consultas y la performance son críticas. Una propiedad única del MOLAP es que los cálculos son predeterminados y almacenados con el cubo. Si bien MOLAP encaja perfectamente cuando se requieren consultas y cálculos complejos, la desventaja que se encuentra es que la cantidad de datos que puede manejar se ve limitada.

ROLAP (relacional on-line analytical processing) es un tipo del OLAP en el que los datos multidimensionales son almacenados en una base de datos relacional como por ejemplo MySQL, SAS, ORACLE, etc. ROLAP es más escalable que los otros tipos de OLAP y maneja una gran cantidad de datos correctamente. Aunque en algunos casos la performance puede ser un poco más lenta, ROLAP sólo se ve limitado por el tamaño de la base de datos que fue seleccionada para almacenar los datos. Los datos ROLAP pueden ser almacenados en un archivo plano o en un esquema de estrella. Con ROLAP cada instancia de una consulta sobre los datos se puede ver como parte de una consulta SQL (o múltiples consultas) y es comparable como la sentencia where de una consulta SQL.

HOLAP (hybrid on-line analytical processing) es un tipo del OLAP en el que multidimensional OLAP y relacional OLAP son combinados. En HOLAP la fuente de los datos es almacenada usualmente utilizando una estrategia ROLAP y las agregaciones son almacenadas utilizando una estrategia MOLAP. HOLAP combina las mejores características de ambos ROLAP y MOLAP. La combinación generalmente resulta en un tamaño de almacenamiento menor a las anteriores. En HOLAP las agregaciones pueden ser pre calculadas y ligadas a un modelo hibrido de almacenamiento.

2.4.3 Herramientas OLAP para el análisis de datos

Las herramientas OLAP son las aplicaciones que se utilizan para trabajar sobre cubos multidimensionales con el objetivo de producir, obtener y analizar información producida por las aplicaciones de las organizaciones de manera más completa y organizada. Gracias a estas herramientas los usuarios corporativos tienen la oportunidad de sacar el máximo partido a las bases de datos de información.

La principal particularidad de las herramientas OLAP es que permiten el análisis de distintas fuentes de datos, como por ejemplo DW, de manera intuitiva y avanzada por parte de cualquier tipo de usuario de organizaciones conformadas por múltiples sectores, sin importar el nivel de exigencia y conocimientos del que disponga cada uno.

Su principal finalidad es beneficiar a la hora de llevar a cabo análisis completos del estado de la empresa y de obtener las estadísticas más completas y detalladas de

todo lo relacionado con los resultados corporativos. En base a estos resultados proporcionados por OLAP, las empresas tienen la posibilidad de poner en marcha procesos de optimización y de tomar decisiones en base a la situación de cada momento.

Gracias a los cubos multidimensionales de OLAP, presentados de manera gráfica y visual, con información detallada, esta tecnología proporciona todo lo que necesitan las empresas para un control efectivo de los recursos de la misma.

Las herramientas de OLAP presentan al usuario una visión multidimensional de los datos (esquema multidimensional) para cada actividad que es objeto de análisis. El usuario formula consultas a la herramienta OLAP seleccionando atributos de este esquema multidimensional sin conocer la estructura interna (esquema físico) de la fuente de datos. La herramienta OLAP genera la correspondiente consulta y la envía al gestor de consultas del sistema (p.ej. mediante una sentencia SELECT).

Un componente importante en las herramientas OLAP son las tablas de pivote. Entre otras funciones, las tablas de pivote pueden de forma automática clasificar, contar, totalizar o dar la media de los datos almacenados. Estas tablas son las que presentan la información a analizar y se generan luego de cruzar las distintas dimensiones.

2.5 **Open Services Gateway Initiative (OSGI)**

Actualmente la tendencia del desarrollo de aplicaciones va hacia la implementación de aplicaciones grandes y de alta complejidad. Con esto el mantenimiento tiende a ser complejo y costoso.

Por lo tanto, el camino que se está intentando recorrer cada vez más asiduamente es el camino del desarrollo de aplicaciones modularizadas.

Si bien Java nativamente no permite la modularización (se limita a la gestión de paquetes, clases y métodos) se han desarrollado frameworks para facilitar la modularización y para aproximarnos a la idea de que se pueden crear aplicaciones modulares; por ejemplo Spring (Pivotal Software 2014).

Veamos el ejemplo de Spring: si bien este framework para el desarrollo de aplicaciones web, permite separar el código en varios componentes, a la hora de desplegar nuestra aplicación en el servidor de aplicaciones, los diferentes componentes se reducen a un único módulo monolítico. Por este motivo, podemos decir que se pierde la modularización de la aplicación ya que por ejemplo no se puede actualizar un módulo particular manteniendo la disponibilidad del resto de los servicios de nuestra aplicación.

En teoría una aplicación modular, no debería dejar de funcionar cuando uno de sus módulos está siendo sustituido, ni deberían dejar de funcionar los módulos que no dependan de un módulo que no funciona correctamente, ni debería tener que recompilar todos los módulos si un módulo es modificado.

Para satisfacer estas necesidades y poder implementar aplicaciones modulares se creó el framework OSGI (Alliance 2014).

OSGI se puede definir como un framework de Java que establece la forma de crear módulos e implementar la interacción entre los mismos en tiempo de ejecución.

Este framework proporciona un entorno orientado a servicios y basado en componentes. Ofrece estándares para que el desarrollador pueda manejar los ciclos de vida del programa.

OSGI posee una arquitectura en capas (relativamente simple) como lo muestra la Figura 2.4.

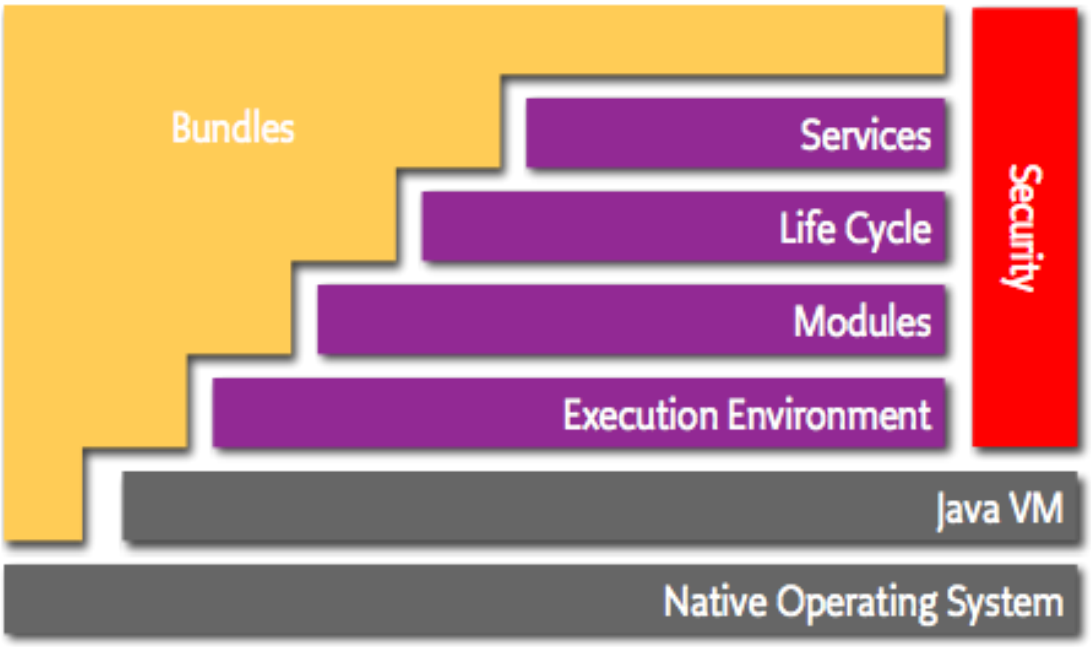


Figura 2.6 - Diagrama de la arquitectura de OSGI

En OSGI cada módulo tiene su propio fichero donde se pueden diferenciar de manera intuitiva las características del mismo. Este fichero es llamado bundle. Cada bundle contiene un fichero de metadatos organizado por pares de nombre clave-valor donde se describen las relaciones del bundle con el mundo exterior a él.

Desde su versión 8.5.2 Domino incorpora el framework OSGI en la tarea HTTP del servidor. Un claro ejemplo de aplicaciones modularizadas en Domino 8.5.2+ mediante OSGI son las XPages (OpenNTF, XPages.info 2014).

XPages es una plataforma de desarrollo de aplicaciones web y móviles. Permite extraer información de aplicaciones IBM Lotus Notes, así como datos de fuentes de datos relacionales y otras. Las mismas tienen interfaz web la cual se presenta en navegadores web en todas las plataformas. El desarrollo de XPages permite desarrollar distintas páginas web, independientes una de la otra, dentro de una misma aplicación.

En la versión 8.5.3 ya extiende OSGI para soportar otros servicios. Uno de ellos es Domino OSGI Tasklet Services (Paul Fiore 2001).

2.6 Domino OSGI Tasklet Service (DOTS)

Los DOTS en Domino son la generación siguiente a los denominados Agentes (ver en marco conceptual). DOTS, como lo indica la sigla, son tareas que corren en el framework OSGI del servidor Domino. Estas tareas son plugins implementados en Java que pueden ser desarrolladas utilizando el entorno de desarrollo Integrado (IDE) Eclipse (The Eclipse Foundation 2014). Una vez que son desarrolladas e instaladas,

estas tareas pueden correr periódicamente, ser programadas para un día y hora, ser ejecutadas a partir de un evento dado o ser ejecutadas manualmente.

Los DOTS utilizan una API de dominio (notes.jar) la cual implementa primitivas para acceder a las bases de datos, documentos, vistas, etc. A su vez, estos plugins pueden utilizar cualquier biblioteca de Java convencional implementadas por terceros.

El ciclo de vida de un DOTS en Domino es el mismo que cualquier bundle en OSGI. Se ilustra dicho ciclo en la Figura 2.5.

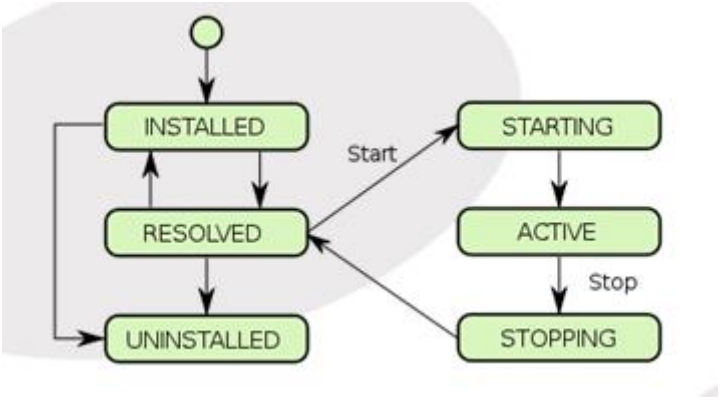


Figura 2.7 - Ciclo de vida de un DOTS

Installed es el primer estado de un bundle. Significa que este fue instalado correctamente. Resolved es el estado que indica que todas las clases Java están disponibles y el bundle está pronto para ser ejecutado. Una vez que se quiere ejecutar el bundle este pasa al estado Starting donde se ejecuta una función que prepara la ejecución. Active es el estado que indica que el bundle está siendo ejecutado. Stopping indica que se detuvo la ejecución del bundle y se ejecuta un método que finaliza la ejecución del mismo. Una vez que se desinstala el bundle, este pasa al estado Uninstalled.

Como se indicó anteriormente los DOTS son denominados la siguiente generación de los Agentes Domino, la idea de su creación es la de reemplazar eventualmente a dichos agentes. En la tabla 2.1 se realiza una comparación entre Agentes Domino y DOTS.

Agentes	**DOTS**
Ejecutan sobre la tarea AMGR o HTTP	Corren sobre el framework OSGI
Son implementados a nivel de diseño de base de datos	Son implementados a nivel de servidor
Se implementan utilizando el IDE Lotus Domino Designer (IBM 2014).	Se implementan en cualquier IDE Java. Por ejemplo Eclipse.
El servidor debe realizar cuatro pasos previos a su ejecución. (ver Figura 2.6)	No tienen pasos previos para su ejecución. (ver Figura 2.7)
Más lentos (ver Figura 2.8)	Más rápidos (ver Figura 2.9)

Tabla 2.1 - Comparación entre Agentes y DOTS

Figura 2.8 – *Pasos que ejecuta el servidor para correr un agente (5 pasos)*

Figura 2.9 – *Pasos que ejecuta el servidor para correr un DOTS (1 paso)*

Mirando las Figuras 2.6 y 2.7 se puede ver que la ejecución de los Agentes requiere un proceso de inicialización cada vez que ejecutan mientras que las tareas DOTS siempre están listas para ejecutar. Esto hace que el proceso previo de ejecución de los DOTS sea significativamente más rápido que los Agentes.

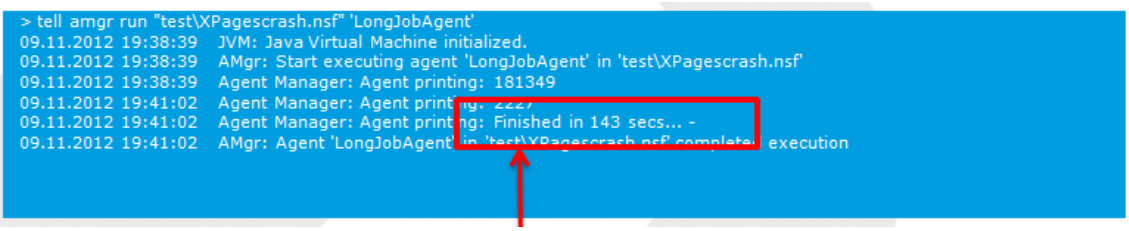


Figura 2.10 – *Tiempo que demora la ejecución de un agente con código X*

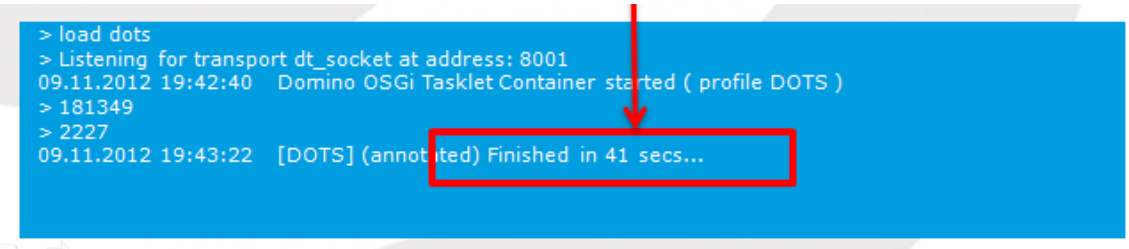


Figura 2.11 – *Tiempo que demora la ejecución de un DOTS con código X*

En las Figuras 2.8 y 2.9 se muestra la ejecución de dos tareas exactamente iguales donde 2.8 muestra la ejecución utilizando un Agente y 2.9 muestra la ejecución utilizando DOTS. Se puede ver que el tiempo de ejecución de un DOTS es significativamente menor al tiempo de ejecución de un Agente realizando el mismo proceso.

Por lo visto anteriormente se puede concluir que los DOTS son más eficientes que los Agentes Domino.

3 Análisis

En esta sección se encuentra descripto el proceso de análisis de requerimientos del proyecto, donde se describen los requerimientos relevados en las reuniones con el gerente de servicios de ISA Ltda., e intercambio de correos con funcionarios de BSE. Además se incluye aquellos requerimientos del proyecto que por nuestra cuenta decidimos agregar para ofrecer un mejor servicio, nos referimos a los requerimientos funcionales 3.1.5, 3.1.6 y 3.1.7.

3.1 Requerimientos funcionales

3.1.1 ETL sobre una base documental

El sistema debe poder transformar los datos cargados de la base de datos documental iGDoc Formularios en un componente que permita visualizar indicadores de dichos datos desde una herramienta de análisis de datos. A continuación se describen algunos de los indicadores.

3.1.2 Indicadores de tiempos

Los diferentes indicadores de tiempos que se deben poder visualizar son los descriptos a continuación.

Tiempo de espera del formulario.

Es el tiempo desde que el formulario llegó a la bandeja de entrada de una unidad, y el tiempo en el cuál fue tomado o asignado a un usuario de esa unidad.

Interesa poder visualizar los tiempos en los cuales el formulario está disponible para actuar y por falta, o no, de personal no se le da continuidad.

Tiempo de actuación del formulario

Es el tiempo entre que el formulario se recibe desde la bandeja de entrada de una unidad (quedando en la bandeja de trabajo) y el tiempo en que llega a la bandeja de salida de esa unidad (también es la bandeja de entrada de la próxima unidad) o se finaliza.

Interesa poder visualizar el tiempo operativo de procesamiento de cada formulario. A diferencia del indicador anterior que representa el tiempo transcurrido (antes de finalizado) que el formulario no ha sido procesado en alguna sección.

Tiempo total de resolución

Es el tiempo entre que el formulario es creado y es finalizado. También se puede definir como la suma de todos los tiempos de espera y los tiempos de actuación por las cuales pasó el formulario. Aplica a formularios finalizados.

Para todos estos interesa visualizar máximos, mínimos y promedios en la gestión del formulario. También se debe permitir visualizar estos indicadores en un rango de fechas dadas.

3.1.3 Indicadores de cantidades

Se debe poder visualizar indicadores de cantidades numéricas sobre propiedades de los formularios, permitiendo cruzar esta información con: estructura organizacional

(secciones, unidades), estado, tipo de formulario y camino recorrido (conjunto de pares sección unidad).

Estos indicadores de cantidades son los descriptos a continuación.

Cantidad de formularios activos

Se listarán los formularios en el estado activo actual correspondientes a la fecha en que se ejecutó la extracción de datos. Esto nos permitirá contar los formularios activos, por sección, unidad, estado, camino (parcial), tipo.

Cantidad de formularios creados

Para cada formulario se cuenta con la fecha de creación. Esta fecha va a indicar el inicio de un trámite y se quiere poder visualizar el total de los formularios creados en un período de tiempo.

Cantidad de formularios finalizados

Para cada formulario se cuenta con la fecha de finalización y/o archivado. Estas fechas marcan el fin del trámite, y se quiere poder visualizar el total de los formularios finalizados/archivados en un período de tiempo.

3.1.4 Cruzamiento por flujo del trámite

Cada formulario electrónico tiene un flujo asociado el cuál determina las secciones donde podrá pasar el formulario. Por lo tanto es importante definir posibles categorías de caminos dentro del flujo a analizar, ya que no es lo mismo un formulario que sigue un camino de dos secciones a diferencia de otra instancia de un formulario que puede seguir un camino de diez secciones.

Mediante este indicador se quiere estudiar la frecuencia de los caminos que se recorren, y poder cruzarlos con: tiempos, estructura organizacional (secciones, unidades) y tipo de formularios.

3.1.5 Interfaz web para la creación del esquema multidimensional

El sistema debe poder generar el esquema multidimensional para configurar el análisis de datos desde una herramienta.

Para esto se debe desarrollar una aplicación web, la cual será responsable de generar el archivo de configuración de la estructura multidimensional. Esta configuración será interpretada por la herramienta de análisis de información elegida y con ella creará el esquema multidimensional deseado.

3.1.6 Flexibilidad para la extracción de información

La herramienta de extracción, transformación y carga de datos no se debe atar a este sistema en particular (iGDoc), sino que debe ser flexible para cualquier sistema documental implementado sobre Lotus Domino.

Por lo tanto la herramienta ETL a desarrollar tiene que ser totalmente configurable en cuanto a la extracción, transformación y carga de datos.

3.1.7 Interfaz web para la configuración de la ETL

El sistema debe poder generar la configuración inicial para la extracción, transformación y carga desde una herramienta.

En esta herramienta generará la configuración para la ETL, que constará de un archivo XML (XML 2014) con la información de todas las tablas que serán necesarias para poder generar los indicadores definidos en los requerimientos.

3.2 Requerimientos no funcionales

3.2.1 La herramienta de BI debe ser OLAP

La herramienta de análisis de datos que se utiliza para visualizar los indicadores debe ser una herramienta de BI y en particular OLAP descartando las herramientas de data mining.

3.2.2 Investigación de herramientas de análisis de datos Open Source

Se deben investigar herramientas de análisis de datos Open Source y elegir la más adecuada según criterios que tenemos que especificar. El resultado de esta investigación será validada por el cliente.

3.2.3 Antecedentes de la herramienta utilizada por el cliente

Se desea que la herramienta a utilizar pueda brindar lo que actualmente brinda la herramienta que utiliza el cliente, la cual es O3.

3.2.4 La herramienta de ETL debe funcionar de forma genérica con los módulos de iGDoc

Se desea que la herramienta de ETL debe poder utilizar como fuentes de datos aquellas bases de datos documentales Domino, cuyos datos fueron generados por cualquier módulo de iGDoc.

4 Investigación de herramientas de análisis de datos

La empresa que presenta el problema base de este proyecto en la actualidad posee un sistema para el manejo y la gestión de trámites en una gran cantidad de organizaciones públicas y privadas. El sistema centraliza grandes volúmenes de información útil almacenada en sus servidores la cual no se está aprovechando en su totalidad.

Es de gran interés poder aprovechar dicha información para comprender y analizar de manera útil los datos existentes de la mejor manera posible.

El análisis de los datos facilitará la toma de decisiones y la comprensión del funcionamiento actual de la organización, permitiendo la anticipación de acontecimientos y ofrecer conocimientos para respaldar las decisiones organizacionales.

Un ejemplo puede ser un formulario que tiene tres recorridos posibles. Los tres recorridos realizan las mismas tareas sobre el formulario y se identifica que uno de los recorridos lleva más tiempo que los otros dos. Esto nos indica que existe un problema en dicho recorrido. Se puede seguir investigando este recorrido y a partir de los resultados obtenidos se pueden tomar medidas. Por ejemplo, se encuentra que en una de las oficinas del recorrido con problemas solo trabajan ocho personas cuando en los otros recorridos trabajan veinte. Una posible solución sería asignar más personas a las oficinas del recorrido con problemas.

La empresa interesada en este proyecto manifiesta que a los usuarios de su sistema les resultaría de gran utilidad una herramienta que permita este tipo de análisis ya que hoy en día se les dificulta la toma de decisiones de este tipo. Esto se debe a que no tienen la posibilidad de indagar y analizar los datos almacenados a lo largo del tiempo.

Existen actualmente herramientas y aplicaciones para analizar grandes cantidades de información. Entre ellas las herramientas OLAP por las cuales las organizaciones interesadas en este proyecto tienen particular interés.

Las herramientas OLAP permiten a los usuarios analizar datos multidimensionales de forma interactiva desde múltiples perspectivas. OLAP consiste en cuatro operaciones analíticas básicas: roll-up, drill-down, slicing y dicing las cuales fueron explicadas en la sección 2.2.2.

Cabe destacar que es un requerimiento que las herramientas a investigar tengan algún tipo de licenciamiento libre. Esto para que las organizaciones no tengan que pagar licencias extra las cuales en ocasiones son muy costosas.

El principal motivo de esta sección es el de realizar una comparación entre las herramientas de análisis OLAP Open Source y con licenciamiento libre que se encuentran actualmente disponibles.

4.1 Requerimientos para las herramientas de análisis de datos

La herramienta seleccionada debía cumplir los siguientes requerimientos.

Sistema operativo

La herramienta seleccionada debe ser multiplataforma. Es esencial que la misma pueda ser instalada en servidores Linux y Windows.

Licenciamiento

La herramienta debe tener licenciamiento libre u Open Source, lo cual implica que la organización que utilice el sistema no tendrá que pagar licenciamiento extra.

Interfaz

La herramienta seleccionada debe tener interfaz web.

Reportes

La herramienta seleccionada debe permitir realizar reportes OLAP y las tres operaciones analíticas básicas mencionadas en la introducción de esta sección 4.

4.2 Criterios de evaluación de las herramientas de análisis de datos

Se tuvieron en cuenta los siguientes factores para la evaluación

Instalación y configuración

- Dificultad de instalación
- Dificultad de configuración

Análisis OLAP

- Análisis en formato de tabla
- Análisis en formato de gráfica
- Manejo de dimensiones y medidas
- Ocultamiento de filas vacías
- Rotación de ejes
- Visualización Expresiones Multidimensionales (MDX 2014)
- Edición MDX
- Exportación de información

Tiempo de respuesta

- Tiempo de respuesta durante el cruzamiento de información

Interfaz gráfica

- Interfaz en idioma español
- Elementos modernos tales como drag and drop, Ajax, diálogos, etc
- Apariencia estética y facilidad de uso

Manejo de usuarios y roles

- Manejo de usuarios y roles

Manejo de usuarios y roles

- Manejo de usuarios y roles

4.3 Herramientas encontradas

Como primera instancia se buscaron e investigaron diferentes herramientas para análisis de datos. Los resultados de la búsqueda se detallan en esta sección.

4.3.1 Pentaho

Las soluciones que Pentaho pretende ofrecer se componen fundamentalmente de una infraestructura de herramientas de análisis e informes, integrado con un motor de workflow de procesos de negocio.

En su versión comunitaria posee licenciamiento libre y Open Source. Se puede modificar el código de la misma como contribuidor pero no se permite la redistribución de la herramienta.

La plataforma será capaz de ejecutar las reglas de negocio necesarias, expresadas en forma de procesos, actividades y de presentar y entregar la información adecuada en el momento adecuado.

Incluye herramientas integradas para generar informes, minería de datos, ETL, análisis OLAP, generación de reportes, etc.

Los productos que ofrece esta plataforma que nos podría interesar son:

Nombre	***Descripción***
Pentaho Data Integration (Kettle)	*Herramienta utilizada para realizar ETL.*
Pentaho Analysis Services (Mondrian Documentation 2014)	*Herramienta OLAP.*

Tabla 4.1 – Productos de Pentaho

Algunos puntos fuertes que encontramos fueron:

- Esta desarrollada enteramente en Java y mientras este publicado en un servidor J2EE, elegir un sistema operativo no es un problema.
- No se tiene porque utilizar todos los componentes de Pentaho sino que se puede integrar con otras herramientas.
- Gracias a su arquitectura modular y por ser Open Source la comunidad puede implementar diferentes plugins que agreguen o mejoren funcionalidades a esta plataforma.

4.3.2 SpagoBI

SpagoBI (Engineering Ingegneria Informatica S.p.A. 2014) es una plataforma integrada para BI, desarrollada enteramente de acuerdo con la filosofía del software libre y de código abierto (FOSS 2014). El tipo de licenciamiento de SpagoBI nos permite estudiar, usar, modificar y distribuir copias modificadas de la misma. Además no posee ningún tipo de licenciamiento pago.

Es una plataforma que cubre y satisface todos los requisitos de BI, tanto en términos de análisis y de gestión de datos, administración y seguridad.

Ofrece soluciones para la presentación de informes, análisis multidimensional (OLAP), minería de datos (Data Mining) (Galit Shmueli 2011), tableros de mando (Dashboard) y consultas ad-hoc.

Cuenta con herramientas ETL y apoya al administrador en el mantenimiento de los documentos analíticos, la gestión para el control de versiones y la aprobación de flujos de trabajo workflow.

Como producto ofrece un módulo SpagoBI Server que abarca las áreas de Reporting, OLAP, Data Mining y ETL, incluyendo varias otras más.

Los productos que ofrece esta plataforma que nos podría interesar son:

Nombre	***Descripción***
Reporting	*SpagoBI permite realizar informes estructurados, con vistas de información estructurada (por ejemplo, listas, tablas, tablas de contingencia, informes) y exportarlos con diversos formatos (HTML, PDF, XLS, XML, TXT, CSV, RTF).* *Los motores posibles son: JasperReport, BIRT, Accessible report, BO*
Multidimensional Analysis (OLAP)	*SpagoBI permite el análisis multidimensional OLAP a través de motores flexibles y fáciles de usar. Los usuarios pueden supervisar los datos sobre los distintos niveles de detalle y desde diferentes perspectivas, a través de procesos de drill-down, drill-across, slice-and-dice, drill-through.* *Los motores posibles son: Jpivot/Mondrian, JPalo/Mondrian, JPXMLA*
Charts	*SpagoBI ofrece un motor gráfico específico, basado en JFreeChart, que permite desarrollar widgets simples de gráficas listas para usar (tales como histogramas, gráficos circulares, gráficos de barras, etc) y los interactivos (sliders temporales, añadir / eliminar series)* *Los motores posibles son:*

	JFreeChart, HChart, ExtChart
Data Mining	*SpagoBI permite el análisis avanzado de datos, gracias a los procesos de minería de datos con el objetivo de descubrir patrones ocultos de información entre una gran cantidad de datos.* *Los motores posibles son: Weka, R.*
ETL	*SpagoBI permite cargar los datos en el almacén de datos y su gestión.* *Los motores posibles son: Talend*

Tabla 4.2 – Productos ofrecidos por SpagoBI

Algunos puntos fuertes que encontramos fueron:

- SpagoBI es enteramente de software libre (FOSS), solamente posee una versión la cual es gratuita.
- Cuenta con una arquitectura escalable que continúa en la lógica del software libre.
- Brinda servicios en cloud, mobile, y otros dispositivos cada vez más utilizadas en el ámbito empresarial.
- Es multiplataforma.

4.3.3 Jaspersoft

Los productos de Jaspersoft (TIBCO 2014) brindan los servicios de BI para que las empresas puedan tomar decisiones más rápidas debido a que se ponen a su disposición información crítica para el negocio en tiempo real dentro de sus aplicaciones y procesos de negocio a través de una plataforma integrable de análisis y de generación de reportes.

Los productos que ofrece esta plataforma que nos podría interesar son:

Nombre	***Descripción***
JasperReports Server	*Servidor de reportes autónomo e integrable.*
JasperReports Library	*Motor de reportes Open Source. Programado enteramente en Java crea documentos utilizando fuentes de distintos tipo, permitiendo ver, imprimir o exportar en diversos formatos.*
Jaspersoft ETL	*Sistema que brinda servicios de ETL.*

Jaspersoft Studio	*Diseñador de reportes basado en eclipse para JasperReports y JasperReports Server.*
iReport	*Diseñador de reportes basado en netbeans para JasperReports y JasperReports Server.*

Tabla 4.3 – Productos ofrecidos por Jaspersoft

Algunos puntos fuertes que encontramos fueron:

- Tiene versión Open Source.
- Está desarrollado para ser integrable con otras aplicaciones y con diferentes dispositivos.
- Brinda los servicios de Reporting y ETL.

4.3.4 Jedox Palo

La Suite Palo (Jedox 2014) combina todas las aplicaciones centrales (OLAP Server, Palo Web, Palo ETL Server y Palo for Excel) en una única plataforma de BI comprensible y customizable. La plataforma se basa completamente en productos Open Source representando una solución de BI de alta gama la cual no está disponible en su totalidad gratuitamente.

Los productos que ofrece esta plataforma que nos podría interesar son:

Nombre	***Descripción***
Palo OLAP Server	*Es un servidor OLAP multidimensional en memoria. Todos los datos que contiene servidor Palo OLAP son organizados en cubos, dimensiones, elementos y atributos de elementos. Este modelo es muy perdurable, ya que refleja con precisión la forma en que los usuarios piensan en su negocio. Palo ofrece un modelo que es ideal para soluciones de BI departamentales y de autoservicio, ya que es muy fácil de usar.*
Palo Web	*Palo Web combina todos los componentes de Palo en una plataforma web uniforme. Palo Web incluye todas las herramientas y módulos que son necesarios para crear aplicaciones completas de Planeamiento, Análisis e Informes. Todos los componentes son 100% basado en la web y no necesita ser instalado en su PC.*

Palo ETL	*Módulo de extracción, transformación y carga de datos masivos de fuentes heterogéneas*

Tabla 4.4 – Productos que ofrece Jedox Palo

Algunos puntos fuertes que encontramos fueron:

- Posee un servidor OLAP que implementa MOLAP esto podría ser una alternativa a los que implementan ROLAP como Pentaho
- Posee una versión gratuita

4.3.5 BMP-Conseil Vanilla

Vainilla (BPM-Conseil 2014) es una plataforma de BI Open Source construida utilizando las últimas tecnologías (Java, GWT) que permiten a los usuarios crear y desplegar informes, vistas OLAP, etc.

Vanilla es capaz de ejecutar múltiples proyectos / de múltiples bases de datos en una sola instancia.

Los productos que ofrece esta plataforma que nos podría interesar son:

Nombre	***Descripción***
WebPortal	*Aplicación Tomcat que contiene el portal Web, el runtime engine para objetos BI, los módulos web para los usuarios finales y también módulos de web designer.*
FreeMetadata	*Es una herramienta para la creación de capas de negocios abstractas.*
Vanilla Plugins for Birt	*Plugin que permite a Birt (otra herramienta de análisis) utilizar recursos como pueden ser metadatos para la creación de reportes Birt*
FreeDashboard	*Es un componente para la creación de Dashboards. Este componente, puede crear Dashboards (pestañas múltiples) e insertar elementos interactivos. Por ejemplo, gráficos flash y mapas, cuadros de lista, campos de texto o pop-up.*
FreeWebReport	*Es el componente para la creación de reportes. Está basado en GWT*

FreeAnalysis Schema Designer	*Es el componente para la creación de los cubos OLAP*
BiGateway	*Es el componente ETL para la suit Vanilla.*
BiWorkflow	*Es el componente para la creación de procesos de negocio.*
BiProcessManager	*Es el componente de scheduling de procesos de la suit Vanilla.*

Tabla 4.5 – Productos que ofrece Vanilla

Algunos puntos fuertes que encontramos fueron:

- Posee una versión gratuita
- Esta implementada en Java y GWT

Se adjunta al informe el Anexo 2 el cual presenta en formato de tabla las características ofrecidas por los proveedores de las distintas herramientas. Se tuvo encuentra esta tabla a la hora de evaluar y comparar las herramientas.

4.4 Prueba realizada para la comparación

Para esta comparación se realizó la siguiente prueba. Se instalaron y se dejaron funcionales cada una de las aplicaciones encontradas para un usuario el cual debía ser capaz de realizar un análisis OLAP sobre un juego de datos ubicados en una base de datos relacional MySQL. Esta base de datos contiene un DW llamado Foodmart (Mondrian Documentation 2014) el cual viene dentro del paquete de Pentaho. Además de los datos también se poseía el diseño de las dimensiones y medidas del cubo. De la instalación, la configuración y el análisis de los datos se evaluaron las herramientas siguiendo los criterios previamente definidos.

4.5 Comparación

En primera instancia se presenta un cuadro comparando la dificultad de instalación y configuración de las herramientas encontradas.

	Instalación y configuración	
Herramienta	**Dificultad de instalación**	**Dificultad de configuración**
Pentaho Community	Dificultad de instalación media. Se requieren varios archivos y no se tiene un instalador ejecutable. Se deben tener ciertos conocimientos en plataformas JAVA.	Dificultad de configuración media. Parte de la configuración se realiza directamente en formularios de la aplicación instalada y otra parte debe realizarse en archivos XML que componen la instalación de Pentaho.

Palo	Dificultad de instalación baja. Solo se requiere ejecutar un archivo instalador y seguir los pasos indicados. Luego de finalizada la instalación se constató que la versión gratuita no contaba con el OLAP manager por lo que no se continuaron las pruebas. Además de esto, al ingresar a la aplicación se despliega un cartel indicando que existe una versión más nueva cuyo link nos lleva a la descarga de la versión paga de la suite.	
Spago BI	Dificultad de instalación baja. Solo se requiere ejecutar un archivo instalador y seguir los pasos indicados.	Dificultad de configuración media. Parte de la configuración se realiza directamente en formularios de la aplicación instalada y otra parte debe realizarse en archivos XML que componen la instalación de Spago Bi. La documentación no fue del todo completa.
BMP-Conseil Vanilla	Dificultad de instalación media. No se tiene un archivo instalador. Se requieren conocimientos en Bases de Datos SQL y plataformas JAVA	Dificultad de configuración baja. Se posee una aplicación dentro de la suite que se encarga de la configuración en pocos pasos.
Jaspersoft Community	Dificultad de instalación baja. Solo se requiere ejecutar un archivo instalador y seguir los pasos indicados.	Dificultad de configuración baja. La aplicación se configura dentro de formularios de la aplicación instalada siguiendo pasos definidos correctamente en el manual de la misma.

Tabla 4.6 – Cuadro comparativo de instalación y configuración de las herramientas encontradas

Para continuar con la comparación de las herramientas se evaluaron las distintas tablas de pivote ofrecidas por las mismas. Una tabla de pivote es una herramienta de análisis de datos que se encuentran en los programas de visualización de datos, tales como hojas de cálculo o de programas de BI. Entre otras funciones, las herramientas de tabla de pivote pueden de forma automática clasificar, contar, totalizar o promediar datos almacenados en una base de datos o una hoja de cálculo. Presentan además los datos en forma de tabla o en forma de gráficas.

En la tabla 4.7 se presentan las distintas tablas de pivote que ofrecen cada una de las herramientas estudiadas.

Es importante estudiar los distintos pivotes de las herramientas ya que esta es el área de trabajo que utilizan los analistas para armar los reportes.

	JPivot	JPalo	Jaspersoft	Saiku	OpenI	Vanilla
SpagoBi	Si	Si	No	No	No	No
Jaspersoft	No	No	Si	No	No	No
Pentaho	Si	No	No	Si	Si	No
Vanilla	No	No	No	No	No	Si

Tabla 4.7 – Cuadro comparativo de que pivotes posee cada herramienta

Viendo la tabla 4.7 se puede ver que Pentaho es la herramienta que ofrece mayor variedad de pivotes. Inclusive presenta un sistema de plugins permitiendo utilizar otros pivotes de terceros. Esto permite al usuario de la aplicación una mayor variedad de pivotes pudiendo elegir el que mejor se adapte a sus requerimientos.

A partir de ahora vamos a estudiar los distintos pivotes. Esto nos va a permitir identificar cuál de ellos cumple de mejor manera con los requerimientos. Luego de estudiados, utilizaremos los resultados para concluir cual es la herramienta que mejor se adapta a nuestros requerimientos.

En las tablas 4.8 y 4.9 se muestra la comparación realizada entre los distintos pivotes siguiendo los criterios de evaluación definidos en la sección 4.2.

	Análisis OLAP							
Tabla de Pivote	**Análisis en formato de Tabla**	**Análisis en formato de gráfica**	**Manejo de dimensiones y medidas**	**Ocultamiento de filas vacías**	**Rotación de ejes**	**Visualización MDX**	**Edición de MDX**	**Exportación de información**
JPivot	Si	Si	El manejo de dimensiones y medidas no es del todo bueno. Se debe configurar todo y una vez finalizado se aplican los cambios a la tabla.	Si	Si	Si	Si	PDF, Excel
JPalo	Si	No	Es bastante simple. Se presentan todas las medidas y dimensiones como filtros. Luego se pueden arrastrar a las columnas y las filas.	Si	Si	No	No	PDF
Jasper Soft	Si	Si	El manejo de dimensiones y medidas no es del todo bueno. Se debe configurar todo y una vez finalizado se aplican los cambios a la tabla.	Si	Si	Si	Si	PDF, Excel.
Saiku	Si	Si	Es bastante simple. Se presentan todas las medidas y dimensiones como filtros. Luego se pueden arrastrar a las columnas y las filas.	Si	Si	Si	No	CSV, XLS
OpenI	Si	Si	Es bastante simple. Se presentan todas las medidas y dimensiones como filtros. Luego se pueden arrastrar a las columnas y las filas.	Si	Si	Si	No	CSV, XLS
Vanilla	Si	Si	El manejo de dimensiones y medidas se realiza arrastrando a la tabla las dimensiones y medidas desplegadas en una barra que se ubica a la izquierda de la pantalla.	Si	Si	Si	No	PDF, HTML y Excel.

Tabla 4.8 – Primera parte del cuadro comparativo de los pivotes

	Tiempo de Respuesta	Interfaz gráfica		
Tabla de Pivote	**Tiempo de Respuesta**	**Interfaz en idioma español**	**Elementos modernos tales como drag and drop, Ajax, diálogos, etc**	**Apariencia**
JPivot	No tuvimos problemas de respuesta significantes. De todos modos a medida que aumenta el tamaño de la tabla se puede notar una demora un poco mayor en el renderizado de la misma.	Si	Básica, se puede ver a simple vista que no utiliza elementos de interfaz modernos tales como drag and drop, diálogos, etc. Es muy estática. No se adapta a la resolución de la pantalla. Todos los elementos se visualizan en la misma página, esto quiere decir que por ejemplo no utiliza pestañas. Aparentemente recarga la página con cada acción que realizamos.	Poco elaborada, básica, poco estética.
JPalo	No presentó grandes problemas de velocidad	No	Utiliza una interfaz moderna. Utiliza elementos modernos como pueden ser drag and drop, diálogos, etc. Es simple. Claramente se adapta a la resolución de la pantalla y a los posibles resize de la ventana. No refresca toda la pantalla a la hora de realizar cambios.	Simple, pocos colores, muy estética
Jasper Soft	No presentó grandes problemas de velocidad	Si	Aparenta ser una mejora de JPivot pero presenta las mismas características.	Simple, pocos colores, poco elaborada.
Saiku	No tuvimos problemas de respuesta significantes. De todos modos a medida que aumenta el tamaño de la tabla se puede notar una demora un poco mayor en el renderizado de la misma.	Si	Es una interfaz algo moderna donde se utilizan elementos tales como drag and drop, diálogos, etc., es bastante sencilla de aprender y de utilizar.	Simple, pocos colores, muy estética
OpenI	No tuvimos problemas de respuesta significantes.	No	Es una interfaz algo moderna donde se utilizan elementos tales como drag and drop, diálogos, etc., es bastante sencilla de aprender y de utilizar.	Simple, pocos colores, muy estética

Vanilla	Con menos de 10000 datos en una medida no se presentaron problemas de respuesta. Pero a medida que aumenta la tabla el renderizado de la misma se hace más lento.	No	Maneja elementos modernos como pestañas, drag and drop y Ajax para la carga de datos. Se adapta a los cambios de resolución y tamaño de la ventana pero en algunos casos se presentan errores ya que recorta algunos elementos.	La interfaz es un poco grotesca. Los iconos no siguen un criterio. Las pestañas, listas y tablas no tienen un diseño elegante.

Tabla 4.9 – Segunda parte del cuadro comparativo de los pivotes

De las tablas 4.8 y 4.9 se desprenden los siguientes resultados separados por criterios de evaluación.

Con respecto a los criterios de análisis OLAP vemos que los pivotes que presentan mejores prestaciones son Saiku, OpenI y Vanilla. Esta decisión la tomamos dado que son los que presentan mejor manejo de dimensiones y medidas. Además cumplen con todas las prestaciones OLAP salvo la edición de MDX, que si bien puede ser útil, no creemos que sea algo fundamental sobre todo para usuarios no expertos en el manejo de este tipo de lenguajes.

Con respecto a los criterios de tiempo de respuesta No tuvimos problemas de respuesta significantes con la mayoría de los pivotes salvo con Vanilla que presentó problemas al renderizar más de 10000 datos.

Con respecto a los criterios de interfaz gráfica vemos que el que posee mejores prestaciones es Saiku. Este ofrece una interfaz en idioma español al igual que JPivot y Jaspersoft. Que tenga interfaz en idioma español es muy importante dado que muchos de los usuarios finales pueden no saber leer en idioma inglés. Pero los puntos en contra que poseen JPivot y Jaspersoft se deben a que su interfaz es muy básica y no poseen elementos modernos tales como drag and drop, Ajax y diálogos, que facilitan y mejoran sensiblemente la usabilidad.

De esta comparación vemos que Saiku es el pivote que obtuvo mejores resultados. Cabe destacar que pudimos ver funcionando a Saiku únicamente en Pentaho.

4.6 Conclusión y resultados obtenidos

De la anterior comparación de herramientas se decidió optar por Pentaho. Del relevamiento y búsqueda de herramientas pudimos concluir que Pentaho es la herramienta libre más conocida en el ámbito del BI. Todas las herramientas mencionadas en la sección 4.3 utilizan el mismo motor OLAP llamado Mondrian. Este es un servidor OLAP comunitario provisto por Pentaho.

La instalación de las herramientas es simple en todos los casos, sin embargo a la hora de configurar las vistas de análisis OLAP algunas fueron más difíciles que otras. El método de configuración de SpagoBI fue el más costoso, seguido por Vanilla, luego Jaspersoft y finalmente el menos complejo fue Pentaho.

Pentaho es la herramienta que posee más opciones a la hora de elegir los distintos pivotes. Entre ellos se encuentra JPivot que es el más conocido pero también es el menos moderno, muy similar al pivote ofrecido por Jaspersoft. También encontramos a Saiku y OpenI que son un poco más modernos y en comparación con el resto de los

pivotes no poseen ninguna desventaja. De la comparación de pivotes realizada pudimos observar que el más completo es Saiku.

La interfaz más completa fue la de Saiku pivote utilizado en Pentaho. Además los únicos que poseían interfaz en idioma español eran Pentaho y Jaspersoft pero el pivote de Jaspersoft no poseía elementos modernos en la interfaz que facilitaran la usabilidad del mismo.

5 Diseño

En esta sección se encuentra descripto el proceso de diseño de la solución del proyecto. Se definieron dos etapas: prototipado y diseño formal.

5.1 **Prototipado**

Durante el análisis y definición del alcance se detectaron posibles riesgos en cuanto a la factibilidad de la implementación. Entre estos riesgos estaba el desarrollo de la herramienta de extracción, transformación y carga (ETL) que debía interactuar con tecnología recientemente implementada por IBM y puesta en producción en su última versión del servidor. Este riesgo junto con la incertidumbre de la performance de esta tecnología en cuanto a tiempos de ejecución, condujo a la decisión de realizar un prototipo de la herramienta ETL con el fin de definir con exactitud la factibilidad de la implementación.

En el prototipo se probó la extracción, transformación y carga de la información que residía en las bases de datos documentales a un DW mediante una tarea implementada como un plugin que corre en el framework OSGI del servidor Domino del cliente, conocido como DOTS.

Al ser OSGI un framework incorporado en la última versión del servidor Domino, teníamos incertidumbre de cómo iba a funcionar, básicamente se probó la performance que iba a tener el plugin para la extracción de información. Veíamos que no tenía sentido implementar un proyecto de BI sobre bases de datos documentales, si la herramienta de ETL que proponíamos demoraba considerablemente, o tenía errores que no podíamos mitigar. Esto último era una posibilidad ya que es algo muy nuevo y podría no estar del todo listo para su utilización.

El prototipo consiste en un DOTS implementado en Eclipse que corre sobre OSGI del Domino Server y accede a una base de datos documental de aproximadamente 35000 documentos. Por cada documento de la base de datos inserta una cantidad de información fija (definida en el prototipo mismo) en un DW que reside en un MySQL server.

Como se observa en la Figura 5.1, el prototipo consta de un servidor Domino donde residen los componentes de ETL y las bases de datos documentales (fuentes de datos del ETL), y un servidor MySQL donde reside el data warehouse en la base de datos PROTOTIPO.

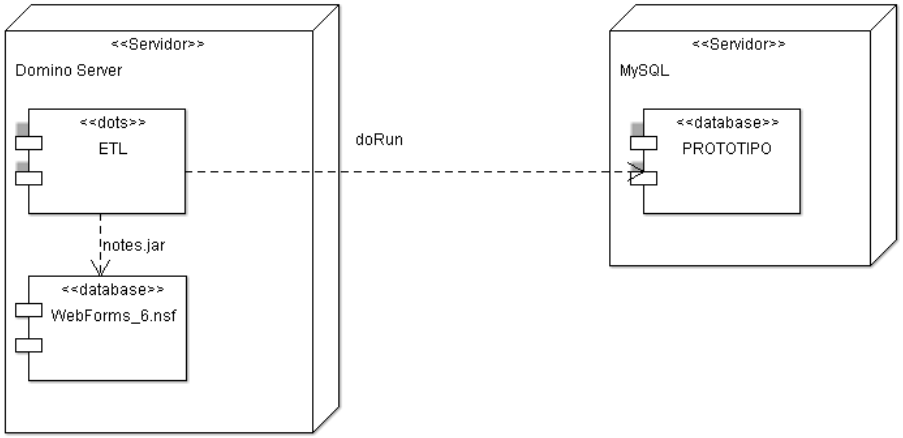


Figura 5.1 – Diagrama de componentes del prototipo

La primer etapa de diseño culmina al implementar el prototipo, analizar los datos de performance y documentar el esfuerzo en cuanto a la implementación.

Para analizar dicho esfuerzo, se registraron las horas de esfuerzo en la Tabla 5.1 y se lo comparó con los resultados obtenidos.

Tarea	**Horas**
Estudio de la tecnología	48
Implementación del prototipo	62
Testeo y análisis de datos	18
Total	128

Tabla 5.1 – Horas de esfuerzo por tarea

Se necesitaron 128 horas para implementar una herramienta simple de extracción de la información.

Con estos datos concluimos que es factible y con esfuerzo limitado implementar un DOTS para la extracción, transformación y carga de datos desde una base documental a una base relacional.

Para el estudio de la performance se procesaron 35000 documentos de los cuales se les extrajo datos y se persistieron en una base de datos relacional. El tiempo de ejecución resultante fue de 4 minutos

Concluimos que los tiempos son buenos al conocer que el cliente ingresa en promedio cincuenta mil documentos anualmente. Por lo tanto concluimos también, que es altamente factible la implementación de la herramienta ETL con un plugin implementado en OSGI sobre el servidor Domino del cliente.

5.2 **Diseño formal**

Luego de tener la certeza en cuanto a la factibilidad de la implementación de la ETL, esto definido y concluido en la etapa anterior, restaba diseñar de forma robusta y configurable los dos componentes fundamentales: la herramienta ETL (la que se prototipó) y la herramienta web para la construcción de la configuración diseño de cubos. Esta última herramienta permitirá al usuario final generar su DW como su negocio lo requiera.

En la Figura 5.2 se presenta un diagrama de componentes de lo que deberá implementar la solución para poder satisfacer los requerimientos planteados en la etapa de análisis.

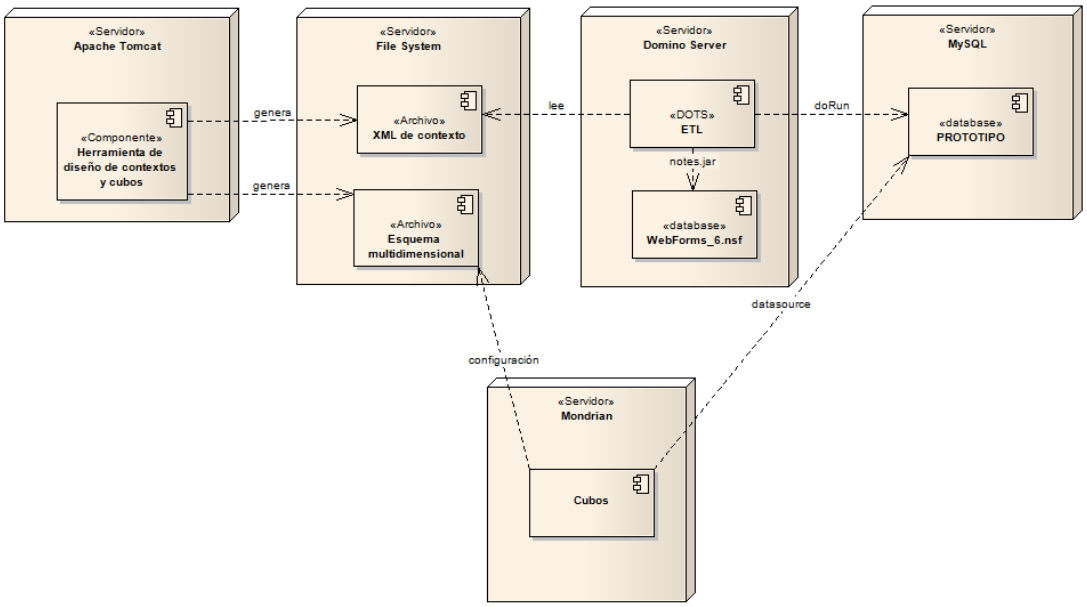


Figura 5.2 - Diagrama de componentes del sistema completo

En el diagrama de la Figura 5.2 se pueden ver los componentes que conforman la solución.

En esta sección se va a describir el diseño de cada componente a implementar en la versión final de la aplicación. Estos componentes son los siguientes: ETL, XML de Contexto, la Herramienta de Diseño de Contextos y Cubos, y Esquema Multidimensional.

5.2.1 XML de Contexto

Este componente es un archivo XML para la configuración de la extracción, transformación y carga de los datos, recibido como parámetro de entrada por el ETL. De ahora en más nos referiremos a este como archivo de contexto.

Para la configuración de la **extracción** de la información se deberá definir:

- Nombres de los servidores Domino (de donde provienen las fuentes)
- Cantidad máxima de documentos a procesar
- Rutas de las bases de datos documentales (relativa a su correspondiente servidor Domino)
- Vistas a recorrer (por cada base de datos documental)

- Los nombres de formas de extracción, son clases que se pueden utilizar para extraer datos de los documentos.

Para la configuración de la transformación se deberá definir la forma en que cada dato se va a extraer del documento. A estas formas de ahora en más las vamos a llamar “extractores”. Cada extractor va a tener:

- Un identificador
- Una forma de extracción (clase que implementa la extracción)
- Una lista de parámetros.

Para configurar la carga de los datos a una base de datos relacional (DW) se deberá especificar datos de la conexión a la base de datos relacional (nombre de la base, usuario y contraseña), si se borran todas las tablas al ejecutar el componente ETL o si se conservan los datos previos a la nueva carga, y la estructura del DW. Para esto último se tendrán que definir las dimensiones y los hechos.

Una dimensión va a tener los siguientes campos:

- Un nombre identificador
- Una lista de campos de la dimensión que resultaran en los nombres de las columnas en la tabla de dimensión (ej. Tipo de formulario, Número).
- Una lista de tipos para los campos
- Una lista de extractores (id de los extractores) que indicarán la forma de extracción de los datos para los campos
- Nombre de la tabla que se crea en la base de datos relacional, que es donde se guardan los valores cargados y extraídos de las fuentes

Un hecho va a tener los siguientes campos:

- Un nombre identificador
- Una lista de nombres de las Dimensiones que participan del hecho, que forman la clave primaria de la resultante tabla de hecho
- Una lista de campos (nombre de las columnas medidas)
- Una lista de tipos de campos (tipos de medidas)
- Una lista de extractores (id de los extractores) que indicarán la forma de extracción para los campos
- Nombre de la tabla que se irá a crear en la base de datos relacional
- Una lista de valores booleanos que indican si el campo (medida) es obligatorio o no para el hecho

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<contexto nombre="test">
    <borrarTablas>true</borrarTablas>
    <cantidadDocumentos>35000</cantidadDocumentos>
    <conexiondb url="jdbc:mysql://localhost">
        <baseDeDatos>PROTOTIPO</baseDeDatos>
        <password>1234</password>
        <usuario>root</usuario>
    </conexiondb>

    <rutasBases>
        <rutaBase>BSE/WebForms_1.nsf</rutaBase>
        <rutaBase>BSE/WebForms_2.nsf</rutaBase>
        <rutaBase>BSE/WebForms_3.nsf</rutaBase>
        ...
    </rutasBases>
    <servidores>
        <servidor>LNTEST/IGDOC</servidor>
        <servidor>LNTEST/IGDOC</servidor>
        <servidor>LNTEST/IGDOC</servidor>
        ...
    </servidores>
    <vistas>
        <vista>Todos por numero</vista>
        <vista>Todos por numero</vista>
        <vista>Todos por numero</vista>
        ...
    </vistas>

    <formasExtraccion>
        <formaExtraccion>GIVstr</formaExtraccion>
        <formaExtraccion>GIVint</formaExtraccion>
        <formaExtraccion>EVAfrm</formaExtraccion>
        ...
    </formasExtraccion>

    <dimensiones>
        ...
        <dimension nombre="FORMULARIO">
            <campos>
                <campo>tipo</campo>
                <campo>numero</campo>
            </campos>
            <extractores>
                <extractor>2</extractor>
                <extractor>3</extractor>
            </extractores>
            <tabla>DIM_FORMULARIO</tabla>
            <tipos>
                <tipo>varchar(100)</tipo>
                <tipo>varchar(20)</tipo>
            </tipos>
        </dimension>
        ...
    </dimensiones>

    <hechos>
        <hecho nombre="VOLUMEN_DE_DOCUMENTOS">
            <tabla>FAC_VOLUMEN_DE_DOCUMENTOS</tabla>
```

```
            <campos>
                <campo>activo</campo>
                <campo>finalizado</campo>
                <campo>creado</campo>
            </campos>
            <tipos>
                <tipo>Integer</tipo>
                <tipo>Integer</tipo>
                <tipo>Integer</tipo>
            </tipos>
            <obligatorios>
                <obligatorio>NO</obligatorio>
                <obligatorio>NO</obligatorio>
                <obligatorio>NO</obligatorio>
            </obligatorios>

            <extractores>
                <extractor>30</extractor>
                <extractor>35</extractor>
                <extractor>36</extractor>
            </extractores>

            <dimensiones>
                <dimension>FORMULARIO</dimension>
                ...
            </dimensiones>
        </hecho>
        ...
    </hechos>
    <extractores>
        ...
        <extractor id="3">
            <multivaluado>false</multivaluado>
            <parametros>
                <parametro>numero</parametro>
            </parametros>
            <formaExtraccion>GIVstr</formaExtraccion>
        </extractor>
        ...
        <extractor id="30">
            <multivaluado>false</multivaluado>
            <parametros>
                <parametro>@if(@IsAvailable(sEstado);@if(sEstado=
"Finalizado" | sEstado="Archivado" ;0;1);"0")</parametro>
            </parametros>
            <formaExtraccion>EVAfrm</formaExtraccion>
        </extractor>
        ...
    </extractores>
</contexto>
```

Figura 5.3 - Ejemplo de un archivo de contexto mostrando los diferentes elementos que se especifican para configurar el componente ETL

5.2.2 ETL

La herramienta ETL (Extraction, Transformation and Load) sirve para, cómo lo dice su sigla: extraer, transformar y cargar la información. Este componente es clave en nuestro proyecto por la naturaleza del mismo.

Tenemos la información en un sistema implementado sobre bases de datos documentales, sin ninguna estructura definida, sin ninguna restricción que defina qué información hay en cada documento, ni de qué tipo es, y distribuida en N bases de datos.

Por ende el objetivo de esta herramienta es estandarizar y unificar la información. Definir un esquema estructurado y único en donde resida toda la información dependiendo el negocio de cada cliente.

Por esto último, la herramienta ETL recibe como parámetro de entrada un archivo de contexto. En base a este componente, va a extraer, transformar y cargar la información a una única base de datos relacional (DW). De esta forma tenemos estandarizada y unificada la información. Estamos un paso más cerca para poder analizarla mediante una herramienta de BI.

Para diseñar el componente ETL nos tomamos mucho tiempo e invertimos muchas horas de trabajo, porque esta herramienta es el pilar de nuestro proyecto. Si podemos diseñar una herramienta que extraiga información de bases de datos no relacionales, transformarla e ingresarla a una base de datos relacional según un modelo recibido como parámetro, el resto del problema (hacer un modelo multidimensional para analizar información que reside en un DW) es un problema ya conocido y más fácil de resolver.

Por esto mismo, pensamos y diseñamos de la forma más robusta y configurable posible este componente. El objetivo es que con esta herramienta se pueda extraer, transformar y cargar a una base de datos relacional, datos de cualquier sistema implementado sobre bases de datos documentales con la herramienta IBM Lotus Domino.

5.2.2.1 Cómo funciona el componente ETL

Para entender el diseño de este componente, se tiene que entender cómo funciona. Con el objetivo de dar una línea base de conocimiento para poder interpretar los conceptos de diseño, se resume el funcionamiento del componente en esta sección.

Este componente recorre todos los documentos, que hay en todas las vistas de todas las bases de datos configuradas en el componente XML de Contexto. Se puede observar una estructura lógica de dichos elementos a procesar en la Figura 5.4. Por cada documento procesado se llenan las tablas de dimensiones y hechos que se quieren generar en el DW. Las dimensiones y hechos, como explicamos en la sección 5.2.1 se reciben del archivo de contexto.

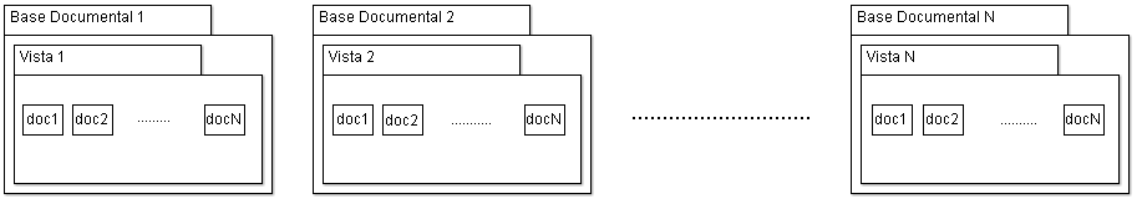


Figura 5.4 - Estructura lógica de un conjunto de bases de datos documentales

Para llenar las tablas de dimensiones que indique el archivo de contexto, se recorren los campos asociados a cada dimensión, y para cada campo se obtiene el extractor asociado. Una vez obtenido el extractor para el campo, se invoca la función de extracción del extractor con el documento que estoy procesando como parámetro.

Como resultado de esa invocación se obtiene el valor que hay que ingresar en la tabla dimensión. Para cada valor a insertar, se obtiene el tipo de campo, y de acuerdo al tipo, se ingresa (si corresponde, o sea si ya no está ingresado) en la tabla de la dimensión de la base de datos relacional. La clave primaria es un valor autogenerado, numérico y sucesivo. Un ejemplo de estas tablas se puede observar en la Figura 5.5 donde la clave primaria está en rojo.

PK_Dimension	Campo_Dimension1	Campo_Dimension2	.....	Campo_DimensionN
1	valor11	valor12		valor1N
2	valor21	valor22		valor2N
3	valor31	valor32		valor3N
4	valor41	valor42		valor4N
.				.
.				.
.				.
.				.
.				.
.				.
M	valorM1	valorM2	.....	valorMN

Figura 5.5 - Ejemplo genérico de una tabla de dimensión

Una vez que se ingresan los registros en las dimensiones, se procesan los hechos.

Un hecho está compuesto de un conjunto de medidas y un conjunto de dimensiones. Para cada dimensión de un hecho, se busca el identificador de la dimensión que corresponda a ese hecho para ese documento (ingresado en el paso previo) y se ingresa en la tabla de hechos dicha referencia.

Luego de ingresar todas los identificadores de la dimensión en la tabla de hechos, se deberán calcular las medidas para ese hecho. De igual forma como se ingresan las dimensiones (en la tabla dimensiones), se obtiene el extractor asociado a la medida del hecho, se invoca a la función de extracción del extractor para el documento y se ingresa dicho valor en el campo medida de la tabla de hechos. Notar que la lógica para la obtención de la medida del hecho reside en el extractor. La clave primaria está formada por las columnas que identifican a las dimensiones que participan en el hecho. Un ejemplo de estas tablas se puede observar en la Figura 5.6.

FK_Dimension1	FK_Dimensio2	FK_Dimensio3	.....	FK_DimensionN	Medida1	.....	MedidaN
12	43	4		4	medida11		medida1N
32	323	86		86	medida21		medida2N
65	515	6165		6165	medida31		medida3N
76	674	743		743	medida41		medida4N
.				..			..
.				..			..
.				..			..
.				..			..
.				..			..
.				..			..
23	2	213521	.....	213521	medidaM1	.....	medidaMN

Figura 5.6 - Ejemplo genérico de una tabla de hecho

La decisión de tener la lógica de la extracción de la información en el extractor no es casualidad. La idea es poder tener tantos extractores como quiera el usuario del sistema. El sistema tendrá extractores nativos, pero también se permitirá ingresar extractores (que el usuario defina la forma de obtener el valor) y agregarlos a la herramienta. De forma de solo configurar en el archivo de contexto que extractor quiero usar para cada dimensión o medida y el sistema invoca que lógica procesar, independientemente si es implementada por nosotros o por el usuario final.

Posteriormente, luego que se recorran todos los documentos, vamos a tener un DW como el de la Figura

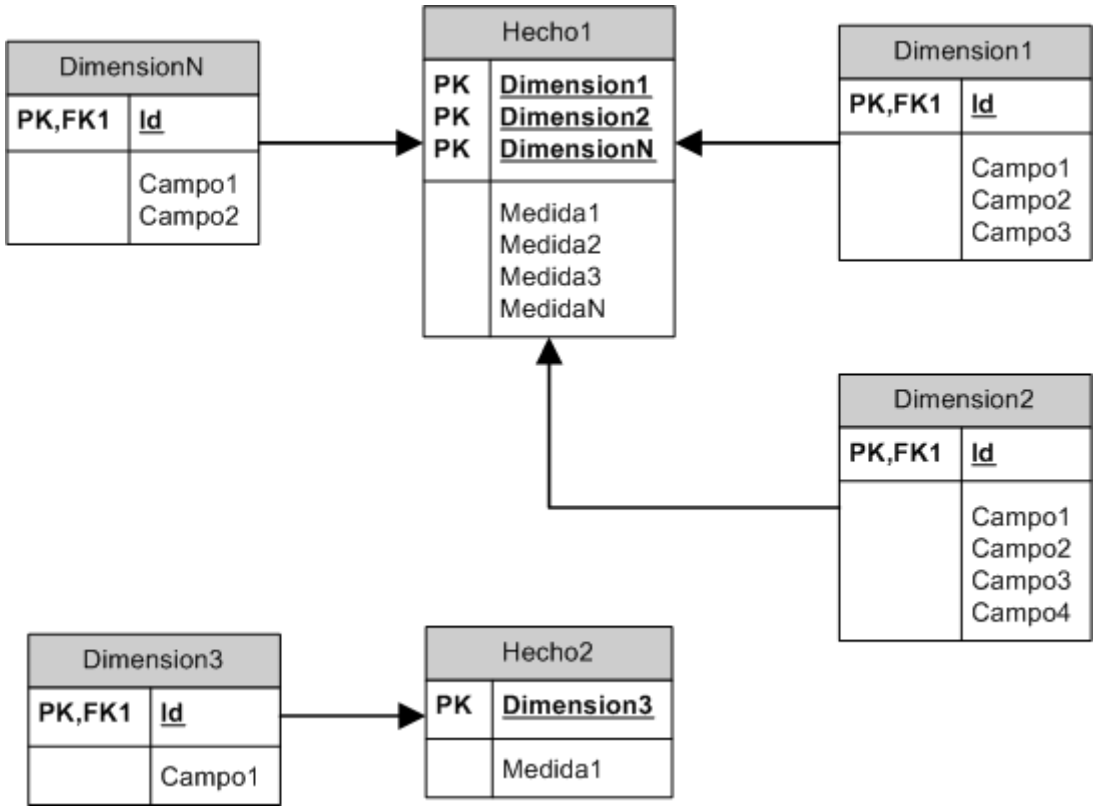


Figura 5.7 - Ejemplo genérico de la estructura de un DW generado por la ETL

En este esquema se tiene 4 dimensiones configuradas (Dimensión1, Dimensión2, Dimensión3, DimensiónN) y dos hechos (Hecho1, Hecho2). Cada dimensión tendrá sus campos, y cada hecho sus medidas. Por cada documento de cada vista, de cada base de datos, se llenarán las tablas de dimensiones y hechos según corresponda.

Una vez explicado el funcionamiento deseado de la herramienta, estamos en condiciones de crear el diagrama de clases que va a ser el modelo de implementación del componente ETL. Este diagrama se puede observar en la Figura 5.8.

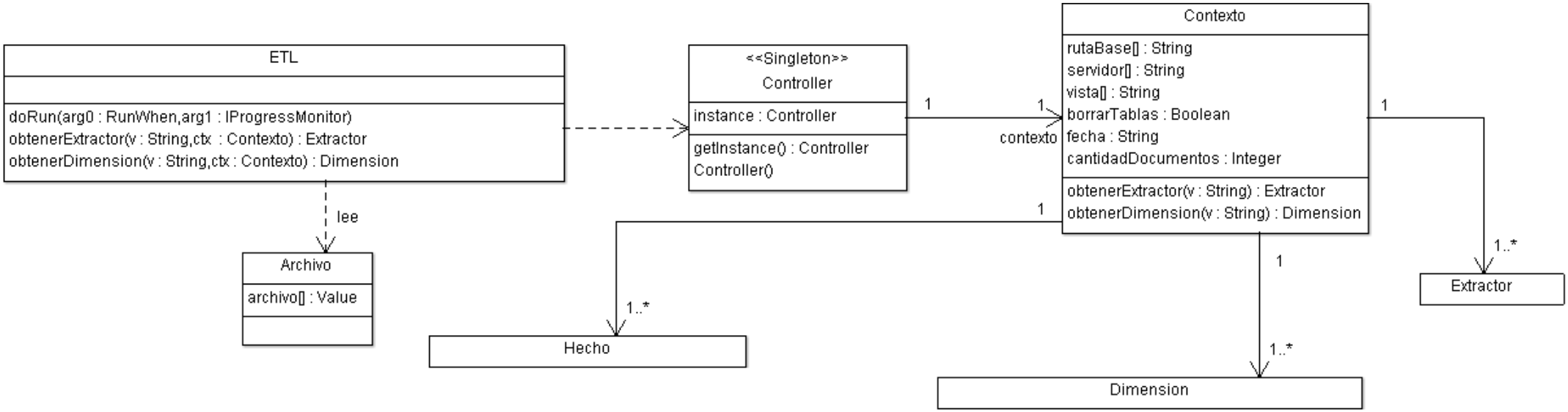


Figura 5.8a - Diagrama de clases del componente ETL – Parte 1

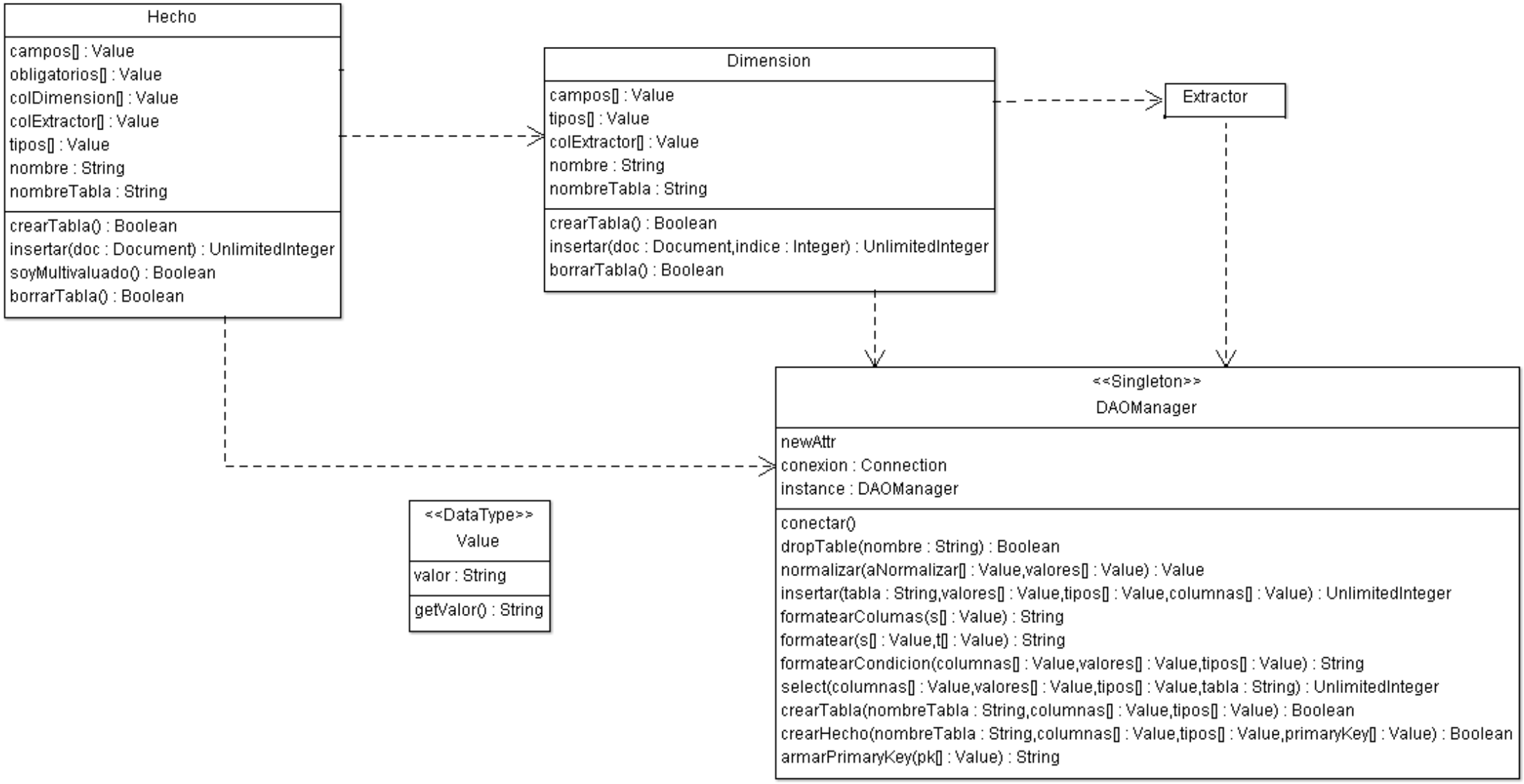


Figura 5.9b - Diagrama de clases del componente ETL – Parte 2

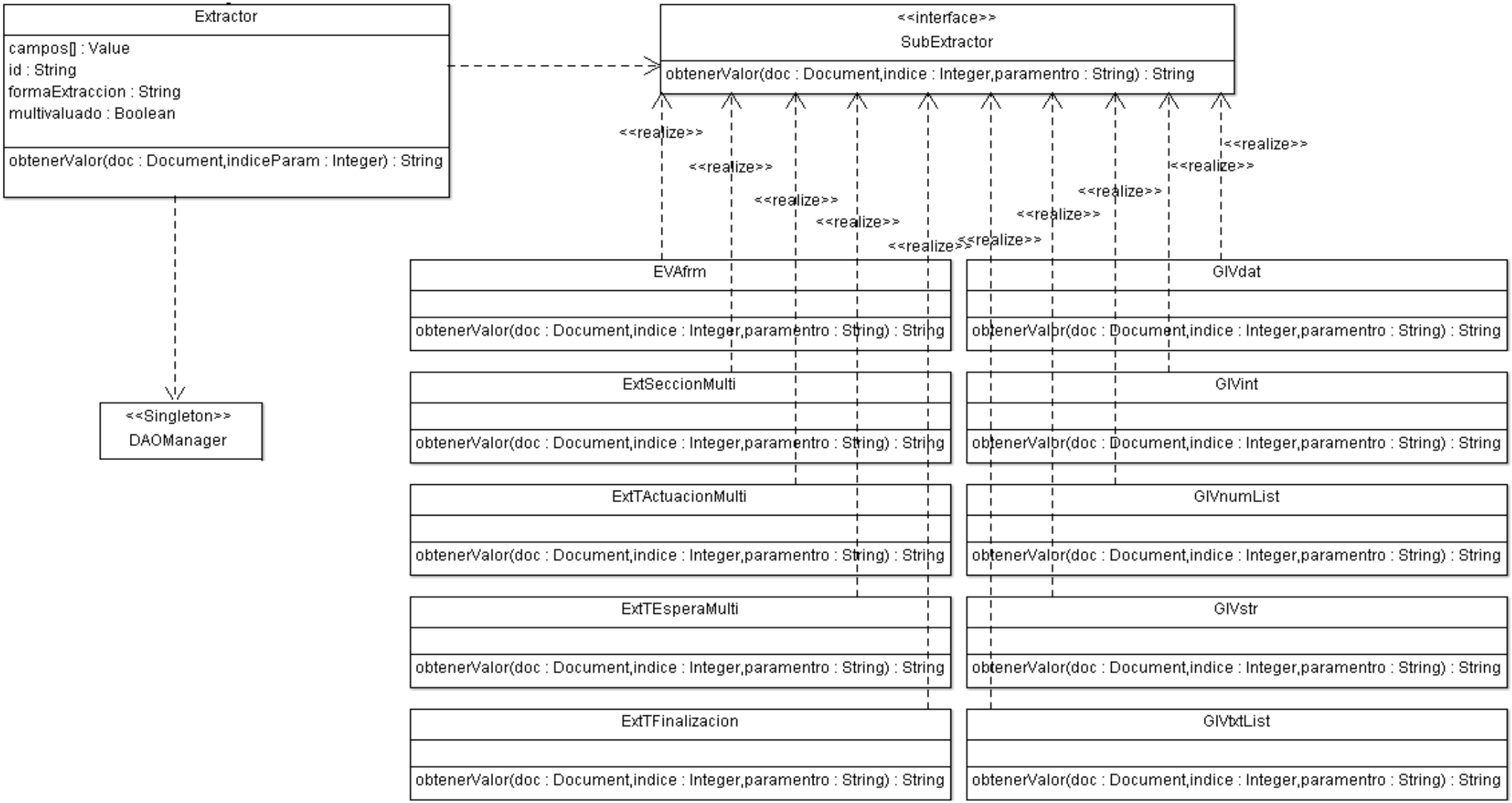


Figura 5.10c - Diagrama de clases del componente ETL – Parte 3

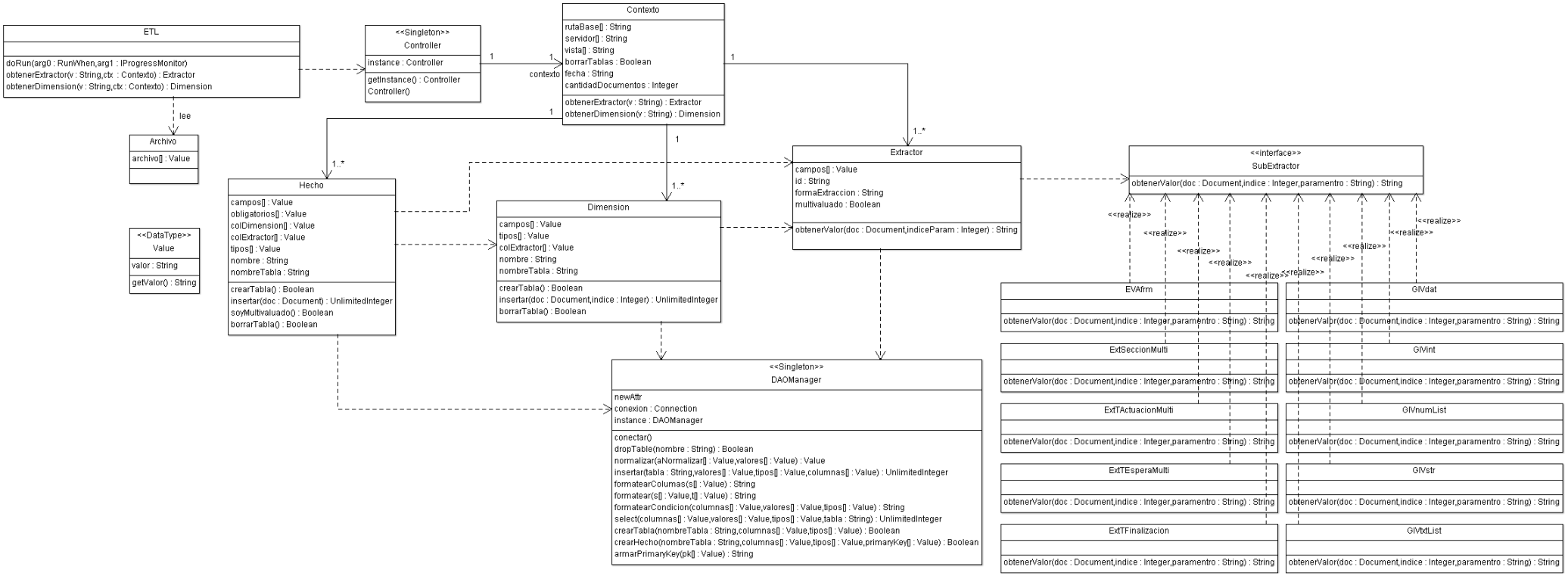


Figura 5.11 - Diagrama de clases del componente ETL

5.2.2.2 Descripción de las clases del componente ETL

En esta sección se describe en detalle las clases del diagrama de clases del componente ETL mostrado en la Figura 5.8.

ETL

Descripción:

La clase ETL extiende la clase AbstractServerTaskExt, para implementar dos de los métodos fundamentales, para que nuestra aplicación ETL se comporte como un plugin DOTS.

Estos métodos son el doRun y el dispose. El doRun se ejecuta cuando la tarea se invoque desde OSGI. Se pude invocar manualmente, o se puede planificar su ejecución.

Atributos:

- No tiene

Operaciones:

- doRun ()
    - Operación que se invoca al momento que la tarea se ejecuta desde OSGI.
    - Esta clase va a ser la encargada de:
        - Leer y decodifica el archivo XML de contexto de entrada.
        - Comunicarse con el objeto DAOManager para conectarse y generar (en caso de que tenga que ser así) la estructura de la base de datos.
        - Asociar la clase Singleton llamada Controlador con el contexto que se deberá procesar según lo especificado en el archivo de contexto.
        - Realizar la iteración de las bases de datos, vistas y documentos según lo especificado en el archivo de contexto.
        - Instanciar para cada iteración a la clase Controlador, para que éste coordine el procesamiento de cada documento.
- dispose()
    - Operación que se invoca al momento de terminar la ejecución.
    - Realiza la desconexión con la base de datos y la eliminación de los objetos utilizados.

Controller

Descripción:

El controller es una clase que implementa el patrón Singleton. Esta clase se utiliza como nexo entre el objeto que recorre la o las bases de datos documentales y la estructura en memoria de los objetos que procesan cada documento.

Atributos:

- instance: instancia del Singleton
- ctx: objeto de tipo Contexto.

Operaciones:

- getContexto():Contexto
- setContexto(ctx)
- getInstance():Controller

Contexto

Descripción:

Es el encargado de instanciar en memoria todos los objetos que forman parte del negocio de ese contexto. Esto es, los hechos, las dimensiones y los extractores.

Esta clase es la encargada de definir si la estructura de la base de datos se tiene que crear nuevamente o se tiene que utilizar la que ya está creada.

Por último este objeto es el que conoce las direcciones de las bases de datos documentales a abrir dentro del servidor, de las vistas a recorrer en cada base de datos, la cantidad de documentos a procesar y en que servidor están las bases de datos.

Atributos:

- dimensiones: colección de dimensiones
- extractores: colección de extractores
- hechos: colección de hechos
- rutaBases: colección de ruta a las bases de datos del servidor
- servidores: colección de nombres de servidores Lotus Domino
- vistas: colección de nombres de vistas
- borrarTablas: bandera para indicar si la estructura del DW se tiene que borrar y crear nuevamente previo al procesamiento o no.
- fecha: fecha del procesamiento.
- cantidadDeDocumentos: cantidad de documentos a procesar por cada base de datos.

Todos estos atributos son obtenidos del archivo de contexto.

Operaciones:

- obtenerExtractor(strNombre):Extractor
    - Dado un id de extractor retorna el objeto extractor correspondiente.
- obtenerDimension(strNombre):Dimension
    - Dado un id de dimensión retorna el objeto dimensión correspondiente.

Dimensión

Descripción:

Esta clase representa una dimensión en nuestro sistema. Cada dimensión conoce sus campos, los tipos de datos de estos y la forma de extraer cada atributo de los documentos para obtener el valor que se guarda en dichos campos.

Por otra parte tiene que saber comunicarse con el DAOManager para poder insertar en su tabla (que representa la dimensión en el DW) los valores correspondientes a su lista de campos.

Atributos:

- campos: lista de nombre de los campos de la dimensión
- tipos: lista de tipos de cada campo de la dimensión
- extractores: lista de identificadores de extractores con los cuales se van a obtener los valores de los campos.
- nombre: nombre de la dimensión en el sistema.
- nombreTabla: nombre de la tabla en el DW.

Operaciones:

- crearTabla(): crea la tabla de la dimensión en el DW.
- insertar(doc, índice): inserta los campos de la dimensión en la tabla de la misma.
- borrarTabla(): borra la tabla dimensión del DW.

Hecho

Descripción:

Esta clase representa un hecho en nuestro sistema. Cada hecho conoce sus campos (medidas), los tipos de datos de estos, y la forma de extraer cada atributo de los documentos para obtener los valores de dichos campos. Por otra parte, cada hecho conoce las dimensiones que lo completan. Por esto mismo cada hecho tiene una colección de identificadores de dimensiones.

De igual forma a las Dimensiones, cada Hecho tiene que saber comunicarse con el DAOManager para poder insertar en su tabla (que representa el hecho en el DW) los valores correspondientes a su lista de campos y sus dimensiones.

Atributos:

- campos: lista de nombre de las medidas de un hecho
- tipos: lista de tipos de cada medida del hecho
- extractores: lista de identificadores de extractores con los cuales se van a obtener los valores de las medidas.
- nombre: nombre del hecho en el sistema.
- nombreTabla: nombre de la tabla en el DW
- obligatorios: lista de valores que determina si una medida es obligatoria para el hecho o no. En caso de no poder obtener una medida obligatoria para el hecho, el hecho no se ingresa en el sistema. De forma contraria, si no se puede obtener una medida no obligatoria para el hecho, el sistema ingresa null y sigue ingresando medidas para el hecho.

Operaciones:

- crearTabla(): crea la tabla del hecho en el DW.
- insertar(doc, índice): inserta las medidas y las referencias a las dimensiones en la tabla del hecho.
- soyMultivaluado(): función que evalúa si en las medidas del hecho existe al menos una medida que tiene una forma de extracción multivaluada (se verá en la próxima sección)
- borrarTabla(): borra la tabla dimensión del DW.

Extractor

Descripción:

Esta clase representa un extractor en nuestro sistema. Cada uno tiene la responsabilidad de leer del archivo de contexto la forma de extracción que deben utilizar, los parámetros que recibe dicho extractor y si es multivaluado o no. Luego de leer esta información deberá poder invocar la operación “obtenerValor” del objeto que le corresponda para la forma de extracción que tiene configurado. En el sistema existen clases para extraer valores ya implementadas, pero si el negocio del cliente requiere otra forma de extracción, el sistema se adapta a esa necesidad y permite que se agreguen clases para extraer la información de forma definida por el cliente.

Más adelante, en la sección 5.2.2.3, se detalla el funcionamiento de los extractores dada la complejidad y la importancia en el sistema de los mismos.

Atributos:

- campos: lista de valores de los parámetros que recibe el extractor.
- id: identificador del extractor.
- formaExtraccion: nombre de la clase que implementa el “obtenerValor” para el extractor de identificador Id.

- multivaluado: índica si el extractor es multivaluado o no, esto es si el dato que tiene que extraer del documento reside en un campo multivaluado del documento (un campo que es un arreglo de valores de cualquier tipo)

Operaciones:

- obtenerValor(documento, indice): función que instancia el objeto de la clase "formaExtracción" e invoca la función obtenerValor(doc, parámetros, indice). Esta función retorna siempre un valor de tipo texto.

SubExtractor

Descripción:

Interfaz que deberán implementar todos las formas de extracción. Esta interfaz solo define la operación obtenerValor.

Operaciones:

- obtenerValor(documento, parámetro, indice)

DAOManager

Descripción:

Esta clase es la encargada de gestionar e implementar los accesos a la base de datos del DW. Esta clase implementa el patrón Singleton para asegurarnos que la conexión sea única para una instancia del sistema en ejecución.

Atributos:

- conexion: de tipo java.sql.Connection es el parámetro que representa la conexión a la db.
- instance: parámetro que implementa el patrón Singleton.

Operaciones:

- conectar(): operación para crear la conexión con la base de datos.
- borrarTabla(table): elimina la tabla "table" de la base de datos.
- insertar(table, columnas, valores, tipos): inserta una tupla en la tabla "table" en las columnas "columnas", los valores "valores" del tipo "tipos, y retorna el estado de la operación luego de efectuarse.
- seleccionar(table, columnas, valores, tipos): retorna la cantidad de tuplas que existen en la tabla "table", con los valores de las columnas "columnas" igual a los valores "valores" de tipo "tipos".
- crearHecho(table, columnas, tipos): crea una tabla en la base de datos con el nombre "table", las columnas "columnas" de tipos "tipos" y con clave primaria "id" que es un auto-numerado, y retorna el estado de la operación luego de efectuarse.
- crearHecho(table, columnas, tipos, primaryKey): crea una tabla en la base de datos con el nombre "table", las columnas "columnas" de tipos "tipos" y con

clave primaria “primaryKey”, y retorna el estado de la operación luego de efectuarse.

Archivo

Descripción:

Esta clase es la encargada de leer del archivo “rutas” donde es que reside el/los archivos de configuración “contexto”.

Atributos:

- archivos: colección de rutas de archivos de contexto.

Value

Descripción:

Este datatype representa los valores manejados en el sistema. Solo tiene un atributo de tipo String donde guarda el valor que representa. Se utiliza principalmente en la extracción de los valores del archivo de contexto.

Atributos:

- value: parámetro de tipo String que representa un valor en el sistema.

5.2.2.3 Extractores

La decisión tomada para definir que la lógica para extraer los valores de los documentos, para luego ingresarlos en el DW, resida en los extractores no es una decisión arbitraria. Se realizó de esta manera para poder darle flexibilidad a la herramienta y no atarnos a las formas de extracción, ni al negocio puntual del cliente de nuestro proyecto.

Por ejemplo, si tengo que extraer un dato de un documento que es un dato de un campo del tipo cadena de caracteres. Es una extracción simple. El extractor deberá acceder al campo del documento y retornar el valor del campo como una cadena de caracteres.
Pero si necesitamos extraer la diferencia de dos fechas, tomando en cuenta los días feriados del año. Esto es una extracción más compleja, y depende exclusivamente del negocio puntal de un cliente, en este caso el de Isa Ltda. Para esto el usuario simplemente deberá agregar un componente a la herramienta ETL. Llevando esto al plano de la implementación, deberíamos agregar una clase que implemente la interfaz SubExtractor y hacer referencia a esta clase en la configuración del extractor en el archivo de contexto.

El método de obtención de la información del extractor siempre va a retornar un valor del tipo de dato cadena de caracteres ya que no es necesario manejarlo con su tipo verdadero debido a que no sufrirá más transformaciones que las que tuvo durante la extracción. Luego para ingresarlo al DW la herramienta ETL se fijará en el tipo de campo definido en el archivo de contexto para esta dimensión o hecho y hará la conversión correspondiente.

De esta forma tenemos la flexibilidad buscada a priori para la extracción y la estructuración de los datos. Le damos flexibilidad al usuario para poder extraer lo que quiere y no nos atamos a ningún negocio en particular. Por ejemplo para este cliente hemos implementado veintitrés extractores distintos.

Como lo vemos en el diagrama de clases, el concepto extractor es traducido a una clase que tiene asociada una interfaz SubExtractor. Dicha interfaz es la que se tiene que implementar para agregar un extractor al sistema.

En resumen, agregando un componente con una clase implementando dicha interfaz se agrega un extractor al sistema, tan simple como eso y tan universal como programar en java.

Un extractor además puede ser multivaluado, esto lo explicaremos a continuación.

En las bases de datos no relacionales, existe el concepto de “campo multivaluado”. Un campo multivaluado en un documento de Lotus Domino, es un campo que tiene más de un valor básicamente. Es simple la explicación pero agrega complejidad a nuestro sistema tener un hecho o dimensión cuyo valor se tenga que extraer de un campo multivaluado.

Por ejemplo, si tenemos una dimensión oficina y se quiere conocer el tiempo que un documento permaneció en cada oficina, y además las oficinas por las cuales estuvo están en un campo multivaluado, no basta con especificar cuál es el campo del cual extraer sino que se necesita especificar cuál es la oficina que se requiere extraer de dicho campo para luego asignarle el tiempo que estuvo en dicha sección.

Lo que creamos en este caso es el concepto de extractor multivaluado. Para este caso la forma de extracción recibe un valor en el campo “índice” y retorna el valor que se le pide en dicha posición.

Cada hecho evalúa si existe una o varias medidas que tienen asociado un extractor multivaluado. Si es así el procesamiento de ese hecho va insertar tantos registros en la tabla de hecho como valores haya en el campo multivaluado con más valores.

Es importante siempre remarcar, que la extracción la puede definir el cliente, y por más que el extractor multivaluado reciba el índice del valor que se deberá obtener del campo multivaluado, la lógica de extracción puede retornar un valor cualquiera.

Por ejemplo, tenemos un campo multivaluado cuyos valores son “perro”, “gato”, “pájaro”, y la forma de extracción recibe como parámetro el índice 2. Lo normal sería retornar “pájaro”, pero la implementación de la forma de extracción es la que define qué valor se retorna, si “perro”, “gato” o “pájaro”. Si la forma de extracción define que al índice que se recibe como parámetro hay que sumarle 1 y hacer el módulo 2, en este caso se retornará “gato”.

5.2.3 Data warehouse

El desarrollo de un DW se basa en el diseño de un modelo conceptual que incluye tanto los requisitos de información de los usuarios así como las fuentes de datos operacionales. A partir de éste se obtiene un modelo lógico.

Actualmente muchas de las propuestas disponibles para la comunidad no definen mecanismos para estructurar de manera sistemática el desarrollo de un DW, convirtiéndolo en una tarea compleja y artesanal.

En esta sección se listan los indicadores que se van a implementar y el desarrollo de las dos etapas mencionadas.

5.2.3.1 Indicadores

Los indicadores implementados son:

- *Tiempos*
    - *Tiempo de espera del formulario.*
    - *Tiempo de actuación del formulario*
    - *Tiempo total de resolución*
- *Cantidades*
    - *Cantidad de formularios activos*
    - *Cantidad de formularios creados*
    - *Cantidad de formularios finalizados*
- *Caminos*

5.2.3.2 Diseño conceptual

En esta etapa se tiene por objetivo la construcción de una descripción abstracta y completa del problema. Recibe como entrada el análisis de requerimientos de los usuarios y de reglas de negocio, finaliza con la construcción de un esquema conceptual expresado en términos de un modelo conceptual y la construcción del cuadro de aditividad.

Los diagramas están basados en el modelo conceptual multidimensional CMDM (Carpani 2000).

5.2.3.2.1 Dimensiones

Los diagramas utilizados representan su jerarquía de agregación en forma árbol donde la granularidad de los datos aumenta al aumentar el nivel de los nodos.

Secciones

Esta dimensión representa las secciones que componen a un formulario, debido a eso se desprende la jerarquía entre la sección y el tipo del formulario.

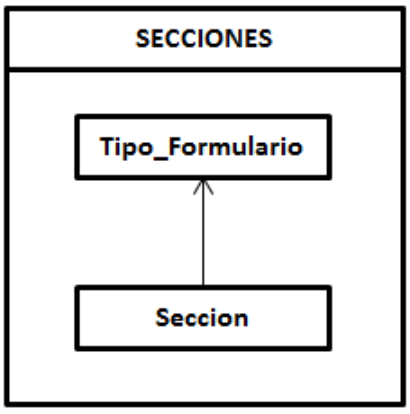


Figura 5.12 - Diagrama de la dimensión sección

Formularios

Esta dimensión representa los diferentes formularios que existen. Los formularios se identifican por un número cuyo formato es único para cada tipo, se desprende entonces la jerarquía entre el número del formulario y el tipo de formulario.

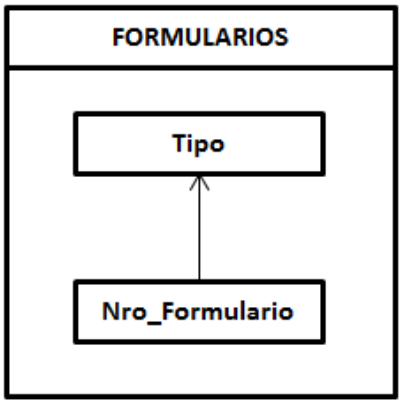


Figura 5.13 - Diagrama de la dimensión formularios

Motivos de Finalización

Esta dimensión representa los diferentes motivos de finalización que existen. Estos motivos se especifican para cada tipo de formulario y por eso se desprende la jerarquía entre motivo y tipo de formulario.

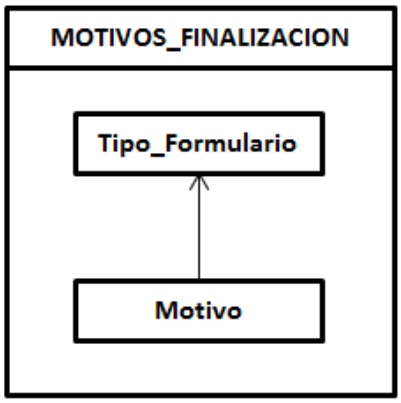


Figura 5.14 - Diagrama de la dimensión motivos de finalización

Unidades

Esta dimensión representa las diferentes unidades. Se identifican por un código, pero se muestra un nombre a los usuarios.

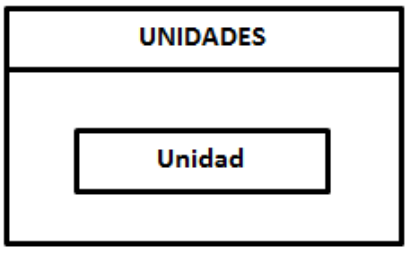


Figura 5.15 - Diagrama de la dimensión unidades

Estados
Esta dimensión representa los diferentes estados que puede tener un formulario.

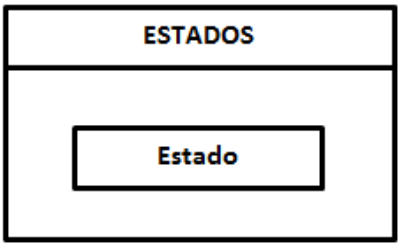


Figura 5.16 - Diagrama de la dimensión estados

Fechas
Esta dimensión representa las diferentes fechas que puede tener un formulario. Por ejemplo se utiliza para representar la fecha de creación, la fecha de envío, la fecha de finalización, etc. de un formulario.

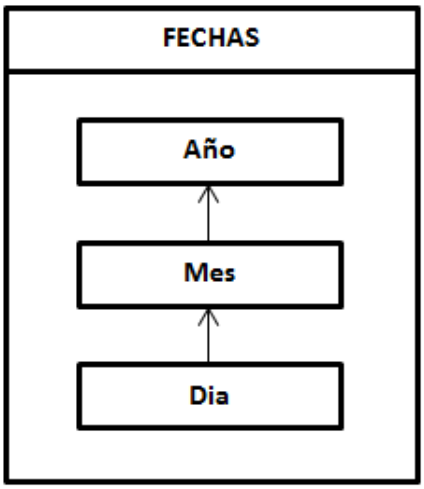


Figura 5.17 - Diagrama de la dimensión fechas

Caminos
Esta dimensión representa los diferentes caminos que puede tomar un formulario en el flujo de su gestión.

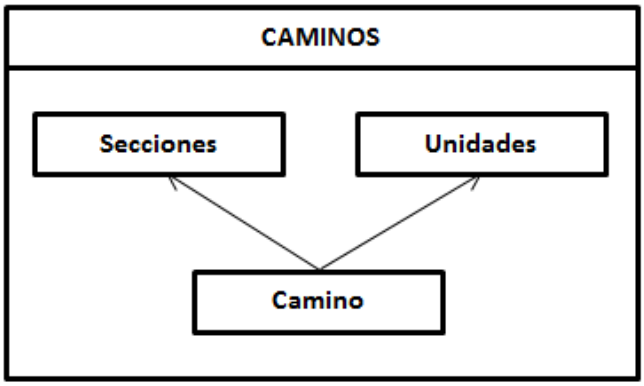


Figura 5.18 - Diagrama de la dimensión caminos

Versión

Los datos pueden ser cargados en diferentes momentos, se creó esta dimensión para permitir diferenciarlos según cuando fueron cargados.

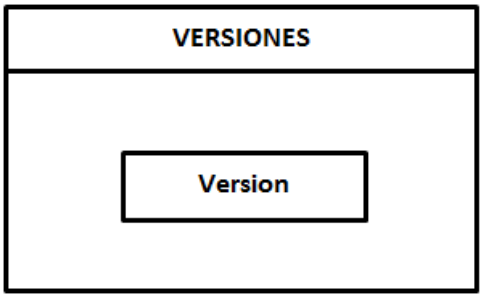


Figura 5.19 - Diagrama de la dimensión versión

5.2.3.2.2 Relaciones Dimensionales

En esta sección se describen los cruzamientos entre las dimensiones para obtener los indicadores, estos participan en el diagrama que son los apuntados por flechas. Se toman estos elementos para formar conjuntos relación donde un elemento está en el conjunto si y solo si hay cruzamiento, obligando que las dimensiones participantes realmente sean cruzables, esto es que se obtenga algún indicador requerido. Cada relación representa un concepto del problema y forman los cubos del modelo multidimensional.

Tiempos de Respuesta de Trabajo

Para esta relación interesaba observar los indicadores tiempos de actuación y tiempos de espera en función de los formularios tramitados dado en la dimensión formularios, de las secciones y unidades por las que circularon los formularios dado por las dimensiones con el mismo nombre. Se agrega la dimensión versiones para identificar cuando fue que se cargaron y calcularon los datos.

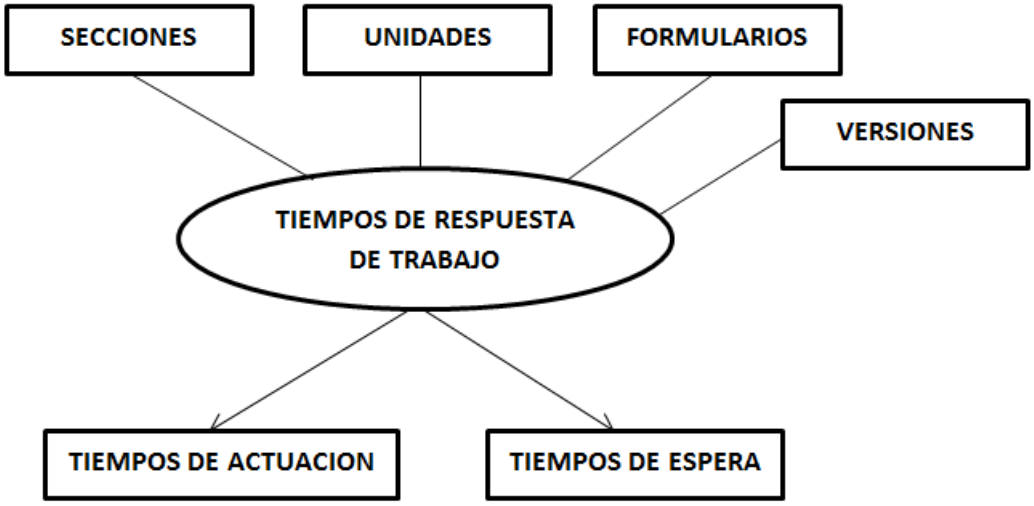


Figura 5.20 - Diagrama de la relación dimensional tiempos de respuesta de trabajo

Tiempos de Resolución

Para esta relación interesaba observar el indicador tiempos totales de resolución en función de los formularios que se tramitaron, utilizando la dimensión del mismo nombre, y los pares sección-unidad donde estuvo el formulario, utilizando la dimensión caminos. Se agrega la dimensión versiones para identificar cuando fue que se cargaron y calcularon los datos.

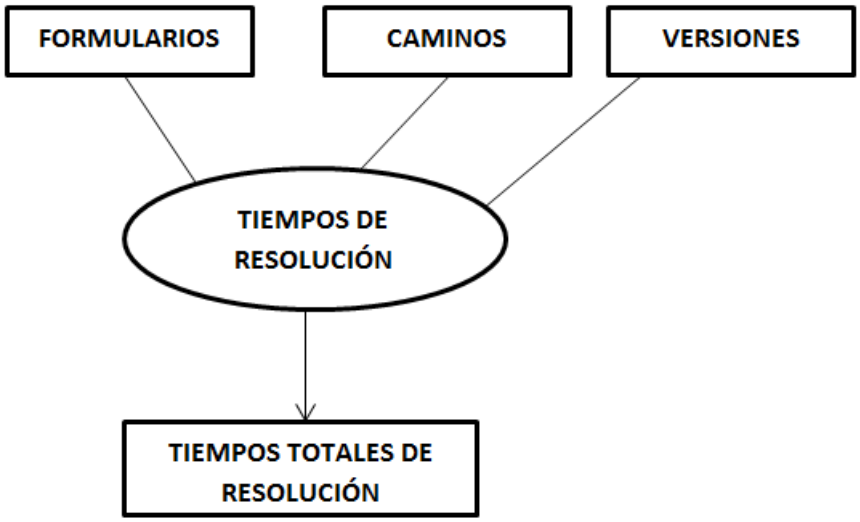


Figura 5.21 - Diagrama de la relación dimensional tiempos de resolución

Volúmenes de Documentos

Para esta relación interesaba observar los indicadores cantidades de formularios activos, finalizados y creados en función de los formularios que se tramitaron dado por la dimensión formularios, los motivos por el cual se finalizó dado en la dimensión motivos_finalizacion y la fecha de finalización utilizando la dimensión fechas, los pares sección-unidad donde circuló el formulario dado por la dimensión caminos, la fecha de creación del mismo utilizando la dimensión fechas, su estado en el momento de la carga de datos dado por la dimensión estados, la fecha en la que se archivó utilizando

la dimensión fechas, las unidades por las que circuló. Se agrega la dimensión versiones para identificar cuando fue que se cargaron y calcularon los datos.

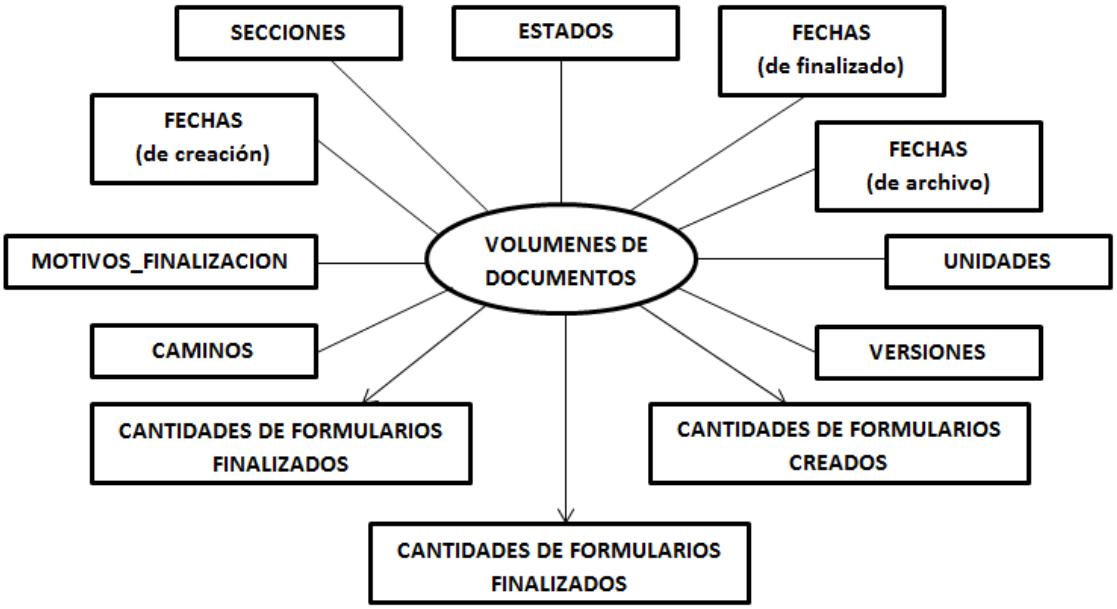


Figura 5.22 - Diagrama de la relación dimensional volúmenes de documentos

5.2.3.2.3 Estudio de aditividad

Para completar el diseño conceptual, realizamos un estudio de aditividad o *Roll-Ups*, el cual nos permitirá visualizar para cada medida y jerarquía dentro de cada dimensión, si la primera es aditiva o no en la segunda, y en caso de que sí, qué operación corresponde para esta relación, pudiendo ser suma o promedio.

Tiempos de actuación	1
Tiempos de espera	2
Tiempos totales de resolución	3
Cantidades de formularios activos	4
Cantidades de formularios finalizados	5
Cantidades de formularios creados	6

Tabla 5.2 - Numeración de medidas

Dimensión \ Medida	**Operación**	**1**	**2**	**3**	**4**	**5**	**6**
Secciones	Seccion->Tipo_Formulario	+	+				
	Tipo_Formulario ->All	+	+				

Formularios	Nro_Formulario -> Tipo	+	+	Pr.	+	+	+
	Tipo ->All	+	+	NA	+	+	+
Motivos_Finalizacion	Motivo ->Tipo_Formulario				+	+	+
	Tipo_Formulario ->All				+	+	+
Unidades	Unidad ->All	+	+		+	+	+
Estados	Estado ->All				+	+	+
Fechas	Dia -> Mes				+	+	+
	Mes -> Año				+	+	+
	Año ->All				+	+	+
Caminos	Camino -> Secciones			Pr.	+	+	+
	Secciones ->All			NA	+	+	+
	Camino -> Unidades			Pr.	+	+	+
	Unidades ->All			NA	+	+	+
Versiones	Versiones ->All	+	+	Pr.	+	+	+

Tabla 5.3 - Estudio de aditividad del diseño conceptual (Roll-Ups)
+ (Suma), Pr. (Promedio), NA (No Aditivo), <Vacío> (No aplica)

En este cuadro podemos observar problemas de aditividad con aquellas medidas cuya operación es un promedio, ya que no es lo mismo realizar el promedio de un conjunto de valores que el promedio de valores resultado del promedio de subconjuntos del mismo. En la sección 5.2.3.3 se explica cómo se resolvió dicho problema.

5.2.3.3 Diseño lógico

En esta etapa se toma como entrada un esquema conceptual y se basa en las fuentes de datos para generar un esquema lógico relacional o multidimensional, donde los objetivos son satisfacer no solo los requerimientos funcionales de información, sino también requerimientos de performance en la realización de consultas complejas de análisis de datos. En nuestro caso tomamos como entrada el diseño conceptual que especificamos en la anterior sección.

Dado que utilizamos el modelo relacional, las dimensiones se guardan en tablas, los indicadores que resultan de la intersección de las dimensiones se guardan en tablas denominadas “tablas de hechos”, esto implica que vamos a tener uno para cada relación dimensional.

Para representar las relaciones dimensionales existen dos esquemas principales, estos son el esquema estrella y el esquema de copo de nieve. El primero implica que las tablas de las dimensiones estén des normalizadas, esto quiere decir que estén todos los valores de una dimensión dentro de una sola tabla, el segundo implica lo contrario, que estén des-normalizadas, esto quiere decir que tenemos una tabla para cada criterio de análisis diferente.

La herramienta elegida Pentaho es una implementación de Mondrian (Mondrian Documentation 2014), que soporta ambos esquemas. Elegimos el esquema estrella debido a que produce respuestas más rápidas a las consultas que el esquema copo de nieve. Esto es beneficioso a la hora de manejar consultas sobre grandes cantidades de datos (Norberto Mazón López 2010). Además la complejidad de cada dimensión no es tan grande como para necesitar des normalizarlas. Esto es porque no poseen más de 3 niveles de agregación, que se traducen en un esquema estrella en 3 columnas como máximo para las tablas de dimensiones.

Previa lectura de los manuales de Mondrian (Hyde 2011), determinamos que no sería necesario resolver problemas de agregación o aditividad mencionadas en el cuadro de aditividad, ya que la herramienta permite resolver aquellos que podríamos atacar a nivel de diseño lógico desde la definición del esquema. Esto es porque Mondrian siempre realiza las operaciones con los valores iniciales, no opera con resultados intermedios. Esto nos permitió diseñar tablas de hechos con los valores asociados a los niveles básicos de cada dimensión relacionada sin agregar estructuras aparte de las necesarias para el esquema estrella.

5.2.3.3.1 Esquemas

En esta sección presentamos los esquemas estrella de cada relación dimensional encontrada en el diseño conceptual. Se utilizó el data modeling del programa MySQL Workbench (ORACLE 2014) para representarlos en modelos EER (S. Sumanthi 2010), es un diagrama donde se especifican las tablas para las dimensiones y las tablas de hechos que guardan los indicadores cuya clave primaria está compuesta de las claves primarias de las dimensiones con las que se cruza.

Tiempos de Respuesta de Trabajo

Este diagrama representa la relación dimensional tiempos de respuesta de trabajo. En esta relación tenemos 4 tablas de dimensiones.

La dimensión secciones viene dada en la tabla DIM_SECCION_MULTI, compuesto por las columnas sección que es el número que la identifica y tipo_formulario que es el tipo del formulario al cual pertenece.

La dimensión formularios viene dada en la tabla DIM_FORMULARIO, compuesta por las columnas número que lo identifica y en la columna tipo se especifica su tipo. La tabla DIM_UNIDADES representa la dimensión unidades donde la columna nombre específica el nombre de la unidad.

Finalmente la tabla DIM_VERSION corresponde con la dimensión versiones y su columna versión identifica cuando fue que se hizo la carga de los datos.

La tabla de hechos en este caso es FAC_TIEMPOSTRABAJO donde tenemos el tiempo de actuación y tiempo de espera disponible en diferentes unidades de tiempo, la primera compuesto por las columnas minutos_actuacion, horas_actuacion, días_actuacion y semanas_actuacion, y la segunda por horas_espera, días_espera y semanas_espera.

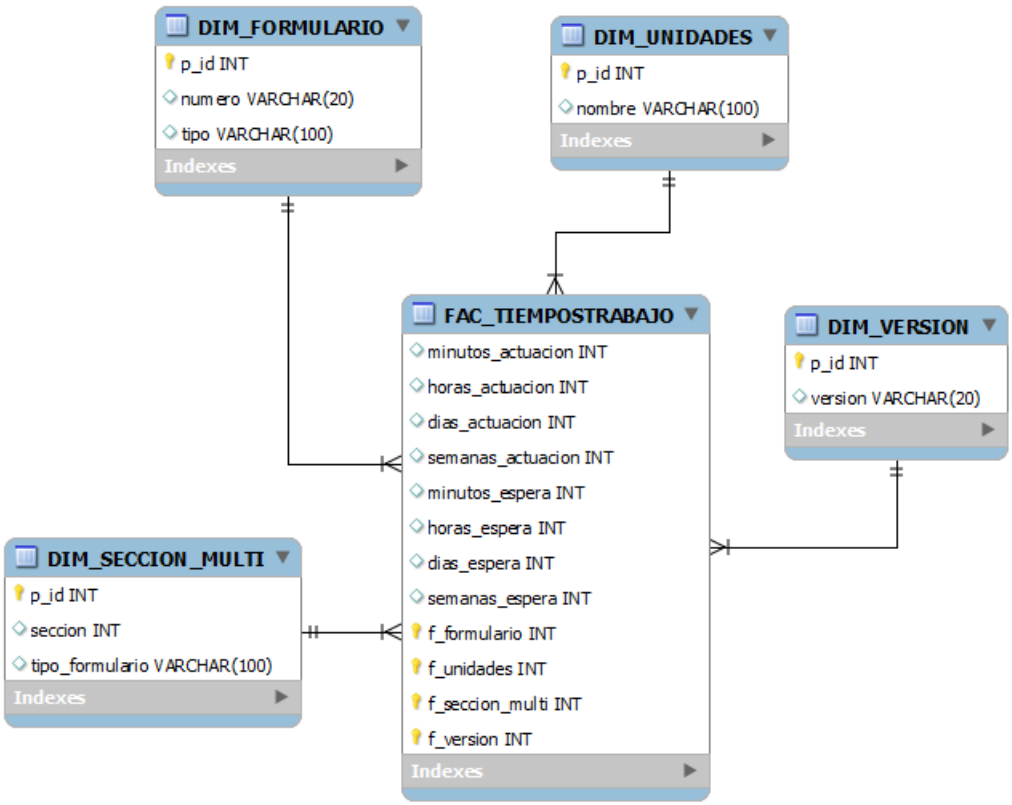


Figura 5.23 - Esquema estrella de la relación dimensional tiempos de respuesta de trabajo

Tiempos de Resolución

Este diagrama representa la relación dimensional tiempos de resolución. En esta relación tenemos 3 tablas de dimensiones.

La dimensión formularios viene dada en la tabla DIM_FORMULARIO, compuesta por las columnas número que lo identifica y en la columna tipo se especifica su tipo.

La tabla DIM_CAMINOS representa la dimensión caminos donde la columna camino son pares unidad y sección por las que estuvo el formulario, las otras dos columnas unidades y secciones son el conjunto de secciones y unidades por separado de los pares antes mencionados.

Finalmente la tabla DIM_VERSION corresponde con la dimensión versiones y su columna versión identifica cuando fue que se hizo la carga de los datos.

La tabla de hechos en este caso es FAC_TIEMPOTOTALRESOLUCION donde tenemos el tiempo total de resolución disponible en diferentes unidades de tiempo, compuesto por las columnas horas_total_resolucion, días_total_resolución y semanas_total_resolucion.

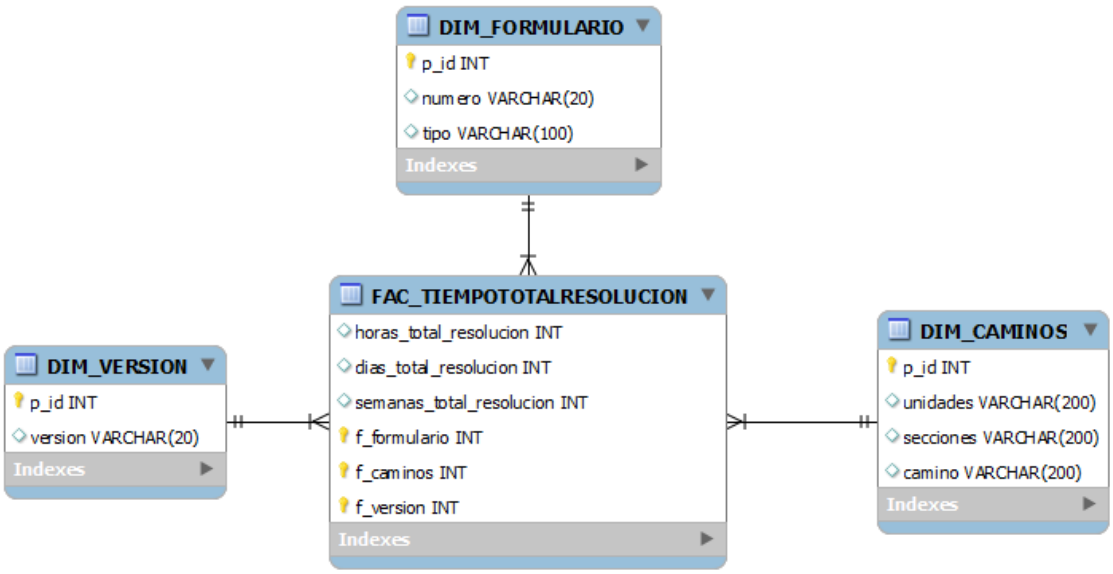


Figura 5.24 - Esquema estrella de la relación dimensional tiempos de resolución

Volúmenes de Documentos

Este diagrama representa la relación dimensional tiempos de resolución. En esta relación tenemos 9 tablas de dimensiones.

La dimensión formularios viene dada en la tabla DIM_FORMULARIO, compuesta por las columnas número que lo identifica y en la columna tipo se especifica su tipo.

La dimensión estados viene dada en la tabla DIM_ESTADOS, compuesta por la columna estado que lo identifica por su nombre.

Participan además 3 dimensiones fechas que se corresponden con las tablas DIM_FECHAS_FINALIZACION para la dimensión fechas de finalizado, DIM_FECHAS para fechas de creación, y DIM_FECHAS_ARCHIVADO para fechas de archivo, todas están compuestas por las columnas dia, mes, año y nombreMes que es el nombre del mes utilizado para mostrar dicho valor en vez del número del mes.

La tabla DIM_CAMINOS representa la dimensión caminos donde la columna camino son pares unidad y sección por las que estuvo el formulario, las otras dos columnas unidades y secciones son el conjunto de secciones y unidades por separado de los pares antes mencionados.

La tabla DIM_UNIDADES representa la dimensión unidades y está compuesta por la columna nombre que es el nombre de cada unidad.

Para la dimensión motivos_finalizacion se implementó la tabla DIM_MOTIVO_FINALIZACION compuesto por las columnas motivo que representa el motivo por el cual fue finalizado y tipo_formulario que es el tipo del formulario al cual está asociado dicho motivo.

Finalmente la tabla DIM_VERSION corresponde con la dimensión versiones y su columna versión identifica cuando fue que se hizo la carga de los datos.

La tabla de hechos en este caso es FAC_VOLUMEN_DE_DOCUMENTOS donde tenemos el volumen de documentos activos, finalizados y creados, en las columnas activo, finalizado y creado respectivamente.

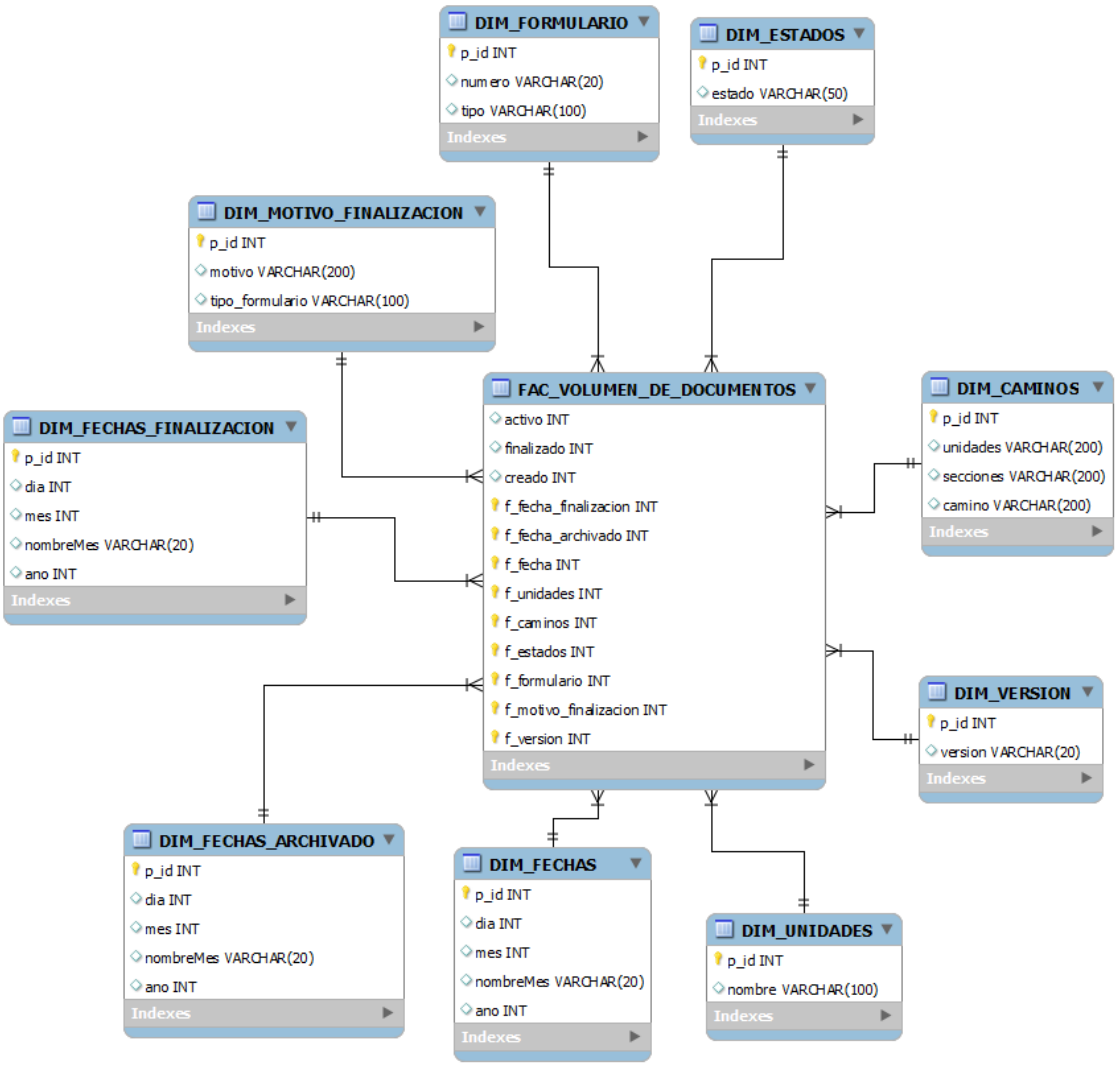


Figura 5.25 - Esquema estrella de la relación dimensional volúmenes de documentos

5.2.4 Herramienta de diseño de Contextos y Cubos

Como podemos observar en el diagrama de la Figura 5.2, el sistema a desarrollar cuenta con dos aplicaciones las cuales son configuradas a través de archivos en formato XML.

La aplicación ETL como lo mencionamos anteriormente, se alimenta de un XML al cual denominamos contexto. Este archivo es donde se define la estructura de base de datos o modelo físico del DW a crear. Define también los métodos de extracción y las transformaciones que se realizarán a los datos almacenados en las bases de datos documentales antes de persistirlos.

Como resultado de la investigación de las distintas herramientas Open Source para el análisis de los datos en el DW fue elegida Pentaho. Esta herramienta requiere la configuración de un archivo en formato XML el cual es denominado Esquema Multidimensional. Éste define un modelo multidimensional, un modelo lógico que define cubos, medidas, dimensiones y jerarquías. Además contiene el mapeo del modelo físico a dicho modelo lógico, esto es a que tablas corresponde cada dimensión y hecho, cuales son las columnas de donde se obtienen los valores que se van a mostrar tanto en las dimensiones como en las medidas, etc. Estos datos son

presentados a partir de la estructura definida por el modelo lógico. Luego de creado el Esquema Multidimensional, es necesario modificar otros archivos XML utilizados por Pentaho para realizar la publicación de dicho modelo lógico en el portal OLAP donde se realiza el análisis de los datos extraídos.

La necesidad de configurar los archivos XML anteriormente mencionados sumado a la dificultad que esto conlleva, motivó a la implementación de una interfaz web para hacer más práctico e intuitivo el proceso de confección de estos archivos XML de manera más simple y transparente para el usuario encargado de realizar esta tarea.

Considerando que las estructuras de ambos XML eran similares en varias partes, se decidió nuclear la configuración de las dos aplicaciones (ETL y Pentaho) en una única interfaz.

Dado que tanto Pentaho como iGDoc son aplicaciones web, se decidió implementar una aplicación con interfaz web para poder realizar la configuración. Esto además permite realizar la configuración de forma remota desde cualquier terminal con un navegador web.

5.2.4.1 Estructura de la solución

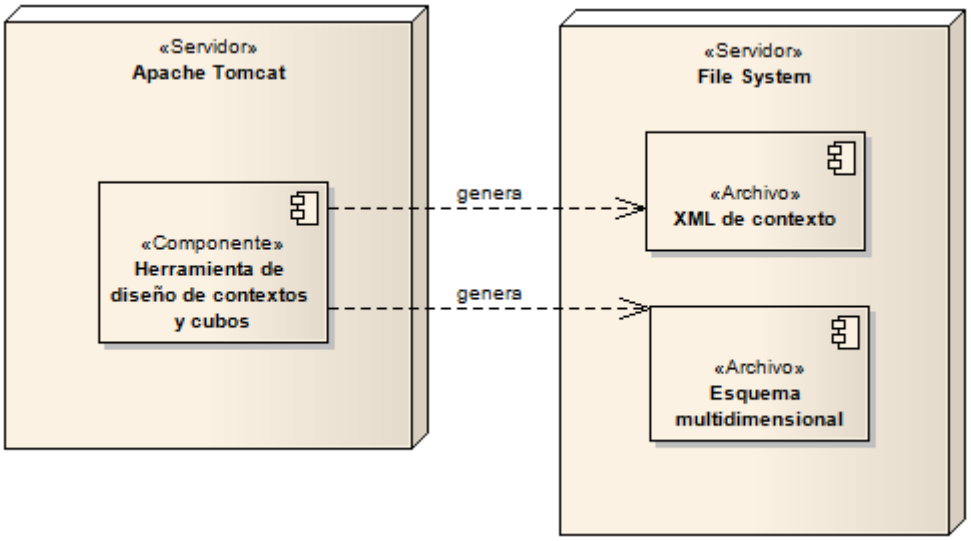


Figura 5.26 - Diagrama de componentes de la interfaz web

Podemos observar en el diagrama de componentes (véase Figura 5.17) la estructura general del sitio web.

Para escoger un servidor de aplicaciones web, decidimos optar por una de las dos tecnologías ya utilizadas. Por el lado de la ETL teníamos la posibilidad de implementar una aplicación web utilizando Lotus Domino, que como ya hemos mencionado, este servidor de aplicaciones posee un servidor HTTP, protocolo utilizado por los navegadores web. Por otro lado teníamos la posibilidad de implementar una aplicación web utilizando Apache Tomcat (The Apache Software Foundation 2014) que es el servidor de aplicaciones utilizado por Pentaho.

En este punto optamos por utilizar Apache Tomcat debido a la simplicidad de la aplicación. Apache Tomcat es un servidor de aplicaciones gratuito por el contrario Lotus Domino posee una licencia costosa. Si bien para utilizar la ETL los usuarios deben tener una licencia Lotus Domino, de todos modos optamos por seguir utilizando herramientas y tecnologías libres de licenciamiento. Además si se quiere tener esta

aplicación en otro servidor distinto al servidor donde se encuentra la ETL, sería necesario instalar otro servidor Lotus Domino lo cual implicaría comprar otra licencia. Apache Tomcat es un servidor liviano, no ocupa mucho espacio en disco duro y es un servidor multiplataforma.

Esta herramienta abarca la generación de dos tipos de archivos XML, el utilizado para configurar el componente ETL en términos de saber cómo extraer los datos (véase 5.2.1 XML de Contexto), donde y como guardarlos, y el archivo de especificación que utiliza Pentaho para visualizar la información en cubos siguiendo la estructura que se especifica en el manual (Mondrian Documentation 2014). Además para este último, se agrega la funcionalidad de publicarlo en Pentaho, de esta forma el usuario para visualizar los cubos basta solo con entrar al sitio web del servidor Pentaho local.

5.2.4.2 Diagrama de clases de la lógica de contextos

El diagrama de clases utilizado, que se muestra en la Figura 5.24, es una fusión de los requerimientos de ambos archivos XML de configuración para ETL y Pentaho. Esto es debido a que observamos información similar en ambas, entonces decidimos en vez de repetir la estructura fusionarla y agregar lo que falta. Por ejemplo, las dimensiones las especificamos en ambos archivos, cada uno necesita conocer cuál es el nombre de la tabla en el DW de dicha dimensión, pero por ejemplo solo para Pentaho se necesita conocer cómo están compuestas las jerarquías que se van a visualizar y solo para el ETL se necesita conocer cuál es el extractor que se va a utilizar para extraer cada valor de cada columna de la tabla de la dimensión.

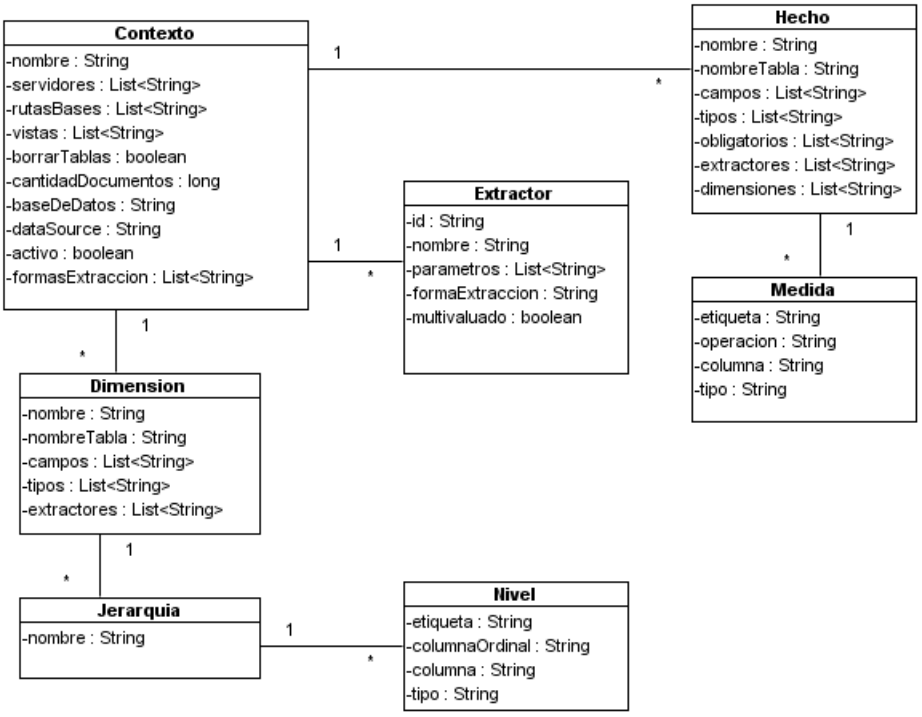


Figura 5.27 - Diagrama de clases de la lógica de contextos.

La clase Contexto es la que engloba a toda la información necesaria para generar el XML que configura al componente ETL para extraer, transformar y cargar los datos en el DW y luego la información que necesitamos para generar otro para Pentaho para poder visualizar dichos datos en forma de cubos. Tiene una colección de objetos de la clase Dimension conteniendo la información de cada dimensión que se utilizan en el cubo que se está especificando. Tiene una colección de objetos de la clase Extractor

donde se especifica cada extractor disponible en dicho contexto que representa. Finalmente tenemos la colección de objetos de la clase Hecho conteniendo la información de cada cubo.

La clase Dimension especifica la extracción de los datos de una dimensión y donde se tiene que persistir en el DW. La colección de objetos de la clase Jerarquia es debido a que en Pentaho se tiene que configurar los niveles de agregación en jerarquías.

La clase Jerarquia especifica la jerarquía de cada dimensión, esto es por ejemplo para las fechas puede ser una de 3 niveles donde el día está en el nivel más bajo, luego estos se agrupan por mes y en el último nivel los agrupamos por año. Posee entonces un conjunto de objetos de la clase Nivel que especifica cada uno de estos niveles en mayor detalle.

La clase Nivel corresponde a un nivel de la jerarquía de una dimensión especificando de que columna se obtienen los datos de dicho nivel y como se ordena.

La clase Extractor especifica cómo se va a realizar la extracción en términos de cuál es la forma de extracción y parámetros que configuran dicho proceso.

La clase Hecho especifica la extracción de los datos de un hecho y donde se tiene que persistir en el DW. Posee una colección de objetos de la clase Medida debido a que se necesita información adicional para visualizarlos en Pentaho.

La clase Medida especifica de qué columna de la tabla de hechos Pentaho obtiene los valores de la medida, información adicional de visualización (etiqueta, tipo), y cuál es la operación que se efectúa cuando se realiza un roll up sobre alguna dimensión del cubo del cual pertenece dicha medida.

6 Implementación

En esta sección se encuentra descripto el proceso de implementación de la solución del proyecto. Se explican las herramientas utilizadas y decisiones tomadas a lo largo de este proceso, para cada componente descripto en la sección 5.2.

6.1 **Prototipo**

La idea de implementar un prototipo se basó en la premisa de minimizar los riesgos y confirmar la factibilidad de la implementación del alcance.

Este prototipo tuvo como único objetivo implementar el ciclo: extracción, transformación y carga de información. Este ciclo fue el primer acercamiento que tuvimos a la extracción de información de una base de datos no relacional y la carga a un DW. Pudimos ver que era factible y también nos dimos cuenta que la herramienta final tenía que ser extremadamente flexible y configurable para que realmente sea una herramienta útil, ya que la falta de estructura de las bases de datos no relacionales aumentaba drásticamente la casuística a tener en cuenta durante este ciclo.

Luego de implementado el prototipo se evaluó el costo de la implementación de este ciclo, la fiabilidad de la tecnología y los tiempos de ejecución. De esta forma pudimos concluir que tan factible sería la implementación del alcance definido en la etapa de análisis y qué problemas podrían surgir para la implementación final.

6.1.1 Configuración del ambiente de desarrollo

Otro de los objetivos implícitos en la construcción del prototipo fue la configuración del ambiente de desarrollo del DOTS.

Para el desarrollo de DOTS utilizamos el IDE Eclipse en su versión Kepler y recurrimos a la asistencia del equipo de desarrollo de IBM quien nos facilitó la documentación para la configuración de DOTS en el IDE en cuestión. En la sección de referencias se adjunta la documentación consultada (OpenNTF, Introduction to DOTS s.f.)

Brevemente las tareas de configuración que se realizaron fueron:

- Instalación de los archivos del núcleo de DOTS en Eclipse (ndots.exe)
- Se añadió la referencia a la biblioteca notes.jar en Eclipse
- Se especificó la ruta en donde está la configuración del framework DOTS en Eclipse (Target Plataform)

Luego de estos pasos estuvimos en condiciones de poder crear un proyecto de OSGI y generar un DOTS.

La configuración de Eclipse para el desarrollo fue relativamente sencilla debido a la calidad de la información brindada por los desarrolladores expertos en estas tecnologías.

Lo último que faltaba para culminar la configuración del ambiente era la definición de un repositorio donde poder versionar el código. Para esto utilizamos Assembla (Assembla 2014) y la herramienta de versionado Subversion (SVN) (The Apache Software Foundation 2014). Esta herramienta está integrada en Eclipse por lo que la configuración fue trivial.

Luego de instalar y configurar Eclipse, virtualizamos este ambiente con VMWare (VMware 2014) e hicimos un snapshot del entorno de desarrollo. De esta forma todos los integrantes del equipo desarrollamos en el mismo ambiente de trabajo.

6.1.2 Configuración de la base de datos relacional

Para la base de datos utilizamos el manejador de bases de datos (DBMS) MySQL. Este DBMS cumple con el requerimiento de ser Open Source y es el más utilizado en el mundo en esta categoría. Al ser un DBMS Open Source y almacenar el DW del sistema, es necesario definir un producto con un soporte confiable y experiencia larga en el mercado.

Si bien la idea del prototipo y la herramienta futura es poder ser independiente del DBMS a utilizar, en esta etapa de prototipado la base de datos definida fue MySQL.

Para el prototipo, la configuración de la base de datos se basó en la instalación del DBMS y la creación de una base de datos llama da “PROTOTIPO”.

Para el acceso desde el DOTS utilizamos el usuario administrador (creado por defecto con la instalación) y el driver Java Database Connectivity (JDBC) (ORACLE 2014) de java.

6.1.3 Prototipo ETL

La implementación del prototipo se basó en un DOTS que realizó una iteración sobre N documentos en la vista “Todos” de la base de datos documental “WebForms.nsf”.

Para cada documento de esta base de datos se crearon tres dimensiones en la base de datos relacional “PROTOTIPO”: dimensión Número, dimensión Formulario y dimensión Fecha. También se creó un hecho (Tiempo activo) en la misma base de datos que tenía las tres dimensiones antes descriptas y una medida que representaba el tiempo que el formulario había estado activo (fecha actual - fecha de creación).

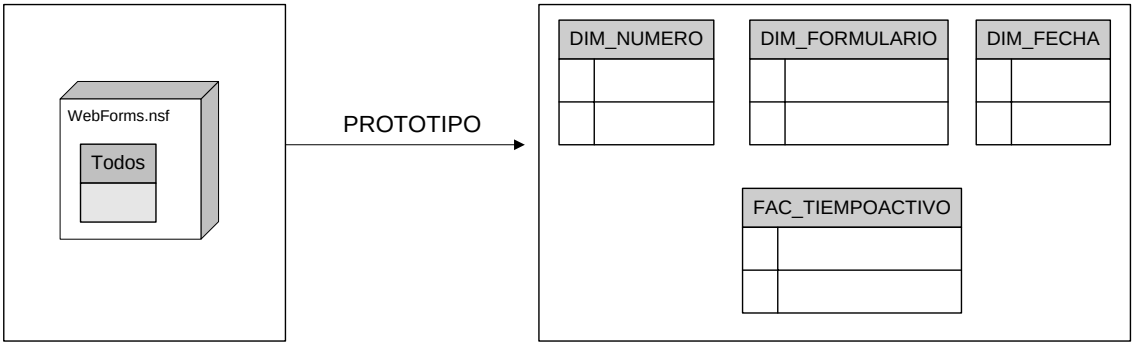


Figura 6.1 – Origen y destino de información

En la Figura 6.1 se muestra gráficamente como partiendo de una base de datos documental WebForms.nsf con una vista “Todos” en donde la información reside en forma documental y sin estructura definida se pasó a un DW de cuatro tablas, con información: estructurada, relacionada y con formato definido. Luego que el DW quedara definido, el problema de analizar la información que ahí reside se resume a configurar la aplicación OLAP creando un archivo schema de Pentaho, al cual le llamamos esquema multidimensional en este proyecto. Éste define un modelo multidimensional, un modelo lógico que define cubos, medidas, dimensiones y jerarquías. Además contiene el mapeo del modelo físico a dicho modelo lógico, esto es a que tablas corresponde cada dimensión y hecho, cuales son las columnas de donde

se obtienen los valores que se van a mostrar tanto en las dimensiones como en las medidas.

6.1.4 Conclusiones y resultados de la etapa de desarrollo del prototipado

Cómo conclusiones principales se puede decir que:

1- El alcance es factible de implementar en un gran porcentaje. Esta afirmación se deduce de la factibilidad de poder almacenar información que reside en una base de datos documental en un DW, ya que el problema de analizar información con herramientas de BI de información residente en DW es un problema de resolución conocida.

2- El problema de implementar un DOTS utilizando JDBC y notes.jar es de complejidad relativa, sin llegar a ser tedioso ni complejo.

3- La performance del prototipo es muy buena, registrando tiempos alentadores.

Yendo estrictamente a los resultados obtenidos, nos interesan ver dos indicadores: esfuerzo realizado y performance obtenida.

Para analizar dicho esfuerzo se registraron las horas de esfuerzo en la Tabla 6.6.1.

Tarea	**Horas**
Estudio de la tecnología	48
Implementación del prototipo	62
Testeo y análisis de datos	18
Total	128

Tabla 6.1 – Horas de esfuerzo por tarea

Se necesitaron 128 horas para implementar una herramienta simple de extracción de la información.

Con estos datos concluimos que es factible y con esfuerzo limitado implementar un DOTS para la extracción, transformación y carga de datos desde una base documental a una base relacional.

Para el estudio de la performance se procesaron 35000 documentos de los cuales se les extrajo datos y se persistieron en una base de datos relacional. El tiempo de ejecución resultante fue de 4 minutos

Concluimos que los tiempos son buenos al conocer que el cliente ingresa en promedio cincuenta mil documentos anualmente. Por lo tanto concluimos también, que es altamente factible la implementación de la herramienta ETL con un plugin implementado en OSGI sobre el servidor Domino del cliente.

6.2 ETL

Como se ha mencionado en secciones anteriores, la idea de la herramienta ETL es poder tener una forma de extracción, transformación y carga de información altamente

flexible y configurable. De esta forma, esta herramienta podrá adaptarse a cualquier negocio y sistema implementado por el cliente.

Para implementar esta herramienta, se diseñó (ver sección 5.2) un componente de entrada (archivo de contexto) que es el que va a tener los parámetros con los cuales la herramienta ETL va a: acceder a la fuente de datos, procesar, transformar y cargar la información.

Esta herramienta consta de cuatro pasos claves: lectura de la configuración, acceso a las bases de datos fuentes (bases de datos no relacionales), transformación de la información dependiendo de la configuración asociada, y por último la carga de la información transformada a un DW.

En las secciones siguientes se detallara cómo se configuraron e implementaron estos cuatro pasos, de forma de explicar en un nivel técnico, no tan exhaustivo, cómo hemos construido la herramienta ETL y como hemos resuelto problemas asociados a la tecnología utilizada.

6.2.1 Configuración del ambiente de desarrollo

Para el desarrollo de la ETL utilizamos el mismo ambiente de desarrollo virtualizado con el que desarrollamos el prototipo.

6.2.2 Configuración de la base de datos relacional

Debido al buen funcionamiento obtenido en el prototipo desde el punto de vista de la base de datos relacional. Utilizamos el mismo DBMS (MySQL), por ende, la configuración fue la misma que creamos para el prototipo.

6.2.3 Interfaz con IBM Lotus Domino

Para poder comunicarnos con las bases de datos documentales desde el DOTS (Java), utilizamos la biblioteca notes.jar.

Esta biblioteca provee las primitivas fundamentales para el acceso a los datos de cada documento. A modo de ejemplo, una de estas funciones más utilizadas para el acceso a los datos en la base documental: documento.getItemValueString(<nombreCampo>), vista.getNextDocument(<documento>), documento.hasItem(<nombreItem>).

La primera función retorna el valor en String del documento “documento” del campo “nombreCampo”. En la segunda función, se está solicitando el próximo documento de la vista “vista” del documento “documento”, esto nos sirve para iterar en la vista configurada en el parámetro de entrada. Por último, la función hasItem retorna verdadero si el documento “documento” tiene el campo “nombreItem” y falso en caso contrario.

El código que utiliza la biblioteca en cuestión está centralizado en dos objetos. Los extractores y el objeto ETL.

6.2.4 Interfaz con componente XML de Contexto

Esta interfaz es muy importante para nuestro sistema. Es la que, mediante un archivo XML de entrada, crea los objetos en memoria de acuerdo al diseño realizado (ver sección 5.2).

Para implementar esto, utilizamos la biblioteca JAXB (java.net 2014). Esta biblioteca vincula el esquema de un archivo XML a código Java. Esto, lo hace mediante anotaciones en la firma de los atributos y las operaciones. (Binding s.f.)

De esta forma, si somos inteligentes y consistentes con el diseño a la hora de la conformación del archivo XML, utilizando esta biblioteca, podemos pasar de un archivo XML de estructura compleja a objetos Java en memoria de manera muy fácil y rápida.

En nuestro caso tenemos los mapeos: Extractor, Dimension, Hecho y Contexto.

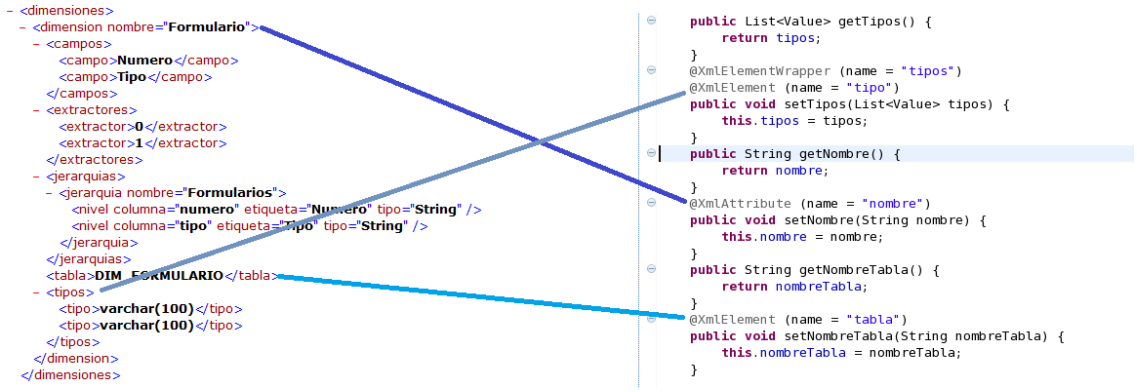


Figura 6.2 – Mapeo entre una dimensión en el archivo XML y un objeto dimensión en código Java.

Para ver el funcionamiento de esta biblioteca de manera resumida ver la Figura 6.3.

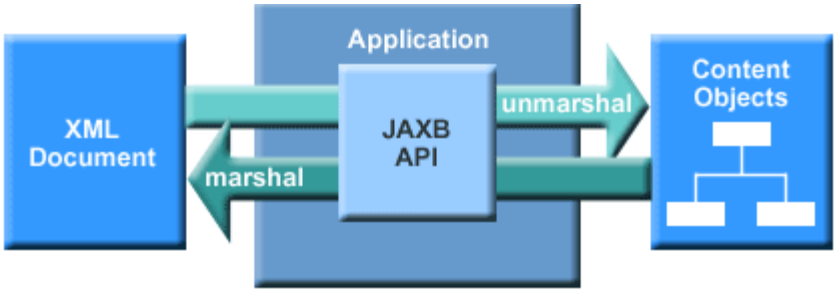


Figura 6.3 – Funcionamiento JAXB

Si partimos de un documento XML al aplicarle el método unmarshal de la biblioteca JAXB pasamos los elementos de la estructura del mismo a una estructura de clases Java en memoria. También se puede aplicar el proceso inverso aplicando a objetos Java en memoria la función marshal, creando en el disco duro un archivo XML que representa la estructura de dichos objetos.

6.2.5 Interfaz con base de datos relacional

La interfaz con la base de datos relacional difiere del prototipo, ya que, en este caso el esquema de la misma varía según la entrada del archivo de contexto.

Por esto mismo, encapsulamos la lógica del acceso a la base de datos relacional en una clase llamada DAOManager. Esta clase es la encargada de: crear una tabla, insertar un registro, eliminar registros, seleccionar registros y borrar tablas.

La decisión de encapsular el código en una clase es debido a la criticidad de la misma y la decisión de diseño de separar totalmente la capa de negocio y la capa de datos.

La capa de datos si bien forma parte del sistema ETL, tiene que ser totalmente independiente de la capa de negocio. Si en un trabajo futuro queremos comunicarnos con otro tipo de bases de datos, solo debemos re implementar esta clase y el sistema funcionará en su totalidad sin necesidad de hacer un test general nuevamente.

Dada la flexibilidad propuesta en el diseño, esta clase tiene la responsabilidad de crear tablas dinámicamente, ajustándose a los tipos de datos recibidos en la configuración para cada columna. También se deberán generar las claves primarias y las claves forañas dinámicamente.

Otro de los problemas que se resolvieron en esta clase (interfaz) fueron: el ingreso, selección y borrado de registros dentro de tablas, que a priori no tenían definida una estructura.

Por último, cabe destacar lo compacta y fácil de entender que resultó la implementación de la interfaz, dejando abierta la puerta para migrar el sistema de almacenamiento de datos. Por ejemplo si se decide que en lugar de utilizar MySQL se desea utilizar otro DBMS como por ejemplo Oracle (ORACLE 2014), bastaría con re implementar la interfaz DAOManager para que se conecte a este otro tipo de implementación de DBMS.

6.2.6 Testeo de la implementación de la ETL

El testeo de la herramienta se dividió en dos partes.

La primer parte constató la prueba unitaria de todos los componentes principales del desarrollo de la herramienta. Cuando hacemos referencia a “componentes”, queremos hacer referencia a los objetos que tienen mayor lógica y complejidad, como lo son: Hecho, DAOManager, Dimensión, Extractor.

Se hicieron pruebas unitarias para cada componente que constaron de ejecutar la ETL en una base de 10000 documentos con información verídica, con un archivo de contexto específico para cada componente. Por ejemplo para probar el componente Dimensión se ejecutó la ETL con el que se muestra en la Figura 6.4.

Luego se verificó la información extraída y almacenada en el DW. Para esto se seleccionó un subgrupo de los documentos procesados de forma aleatoria y se verificó que la información extraída y procesada a partir de estos documentos fuera correcta. Por ejemplo si en el DW tenemos que para el documento de formulario número 12-1-2014 el tiempo activo (desde que se creó el formulario hasta que fue archivado) fue 20 días, buscamos el documento en la base documental correspondiente y verificamos que efectivamente el tiempo activo de dicho formulario fue de 20 días.

Las mismas pruebas se realizaron con el componente Hecho y con el componente Extractor.

Para las pruebas unitarias del componente DAOManager se realizaron programas simples (test unitarios Java), que invocaban esta clase, para probar sus métodos. Luego se comparó el estado final (luego de la invocación del método) de la base de datos con el estado final que debería tener la base si el método se hubiese invocado correctamente.

La segunda parte del testeo se realizó implementando los hechos y dimensiones que se analizaron en el comienzo del proyecto. Hechos y Dimensiones que se documentaron en un primer momento cuando se estaba relevando los requerimientos (ver sección 3.1). Esto nos sirvió como una prueba global del funcionamiento integrado

de los componentes desarrollados y nos sirvió para medir la facilidad real de implementar el DW del cliente con la herramienta ETL.

Estas pruebas revelaron menos errores de lo que pronosticábamos en un principio.

Concluimos que las pruebas unitarias fueron de gran ayuda y efectividad, y que el gran desacople de los componentes desarrollados facilitó el trabajo de testeo.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
- <contexto activo="true" nombre="Contexto_peter">
    <baseDeDatos>PROTOTIPO</baseDeDatos>
    <borrarTablas>true</borrarTablas>
    <cantidadDocumentos>1000</cantidadDocumentos>
    <dataSource>SampleData</dataSource>
  - <dimensiones>
    - <dimension nombre="Formulario">
      - <campos>
          <campo>Numero</campo>
          <campo>Tipo</campo>
        </campos>
      - <extractores>
          <extractor>0</extractor>
          <extractor>1</extractor>
        </extractores>
      - <jerarquias>
        - <jerarquia nombre="Formularios">
            <nivel columna="numero" etiqueta="Numero" tipo="String" />
            <nivel columna="tipo" etiqueta="Tipo" tipo="String" />
          </jerarquia>
        </jerarquias>
        <tabla>DIM_FORMULARIO</tabla>
      - <tipos>
          <tipo>varchar(100)</tipo>
          <tipo>varchar(100)</tipo>
        </tipos>
      </dimension>
    </dimensiones>
  + <extractores>
  + <formasExtraccion>
  + <hechos>
  - <rutasBases>
      <rutaBase>iqdoc/WebFormsArchivo_1.nsf</rutaBase>
```

Figura 6.4 – Ejemplo de la configuración de la dimensión que se utilizó para testear el componente

6.2.7 Caso de estudio real - BSE

Durante la etapa de desarrollo le mostramos el avance del producto a la gerente de proyectos, ingeniera Ana Erosa, que actualmente está liderando el proyecto de accidentes de trabajo en el BSE. Ella nos sugirió la posibilidad de probar la herramienta desarrollada con un caso real. Puntualmente el caso de Accidentes de trabajo y enfermedad profesional (ADTEP) (Erosa 2012).

Para esto le pedimos que especifique cinco indicadores de negocio que querían estudiar. Luego de un par de semanas recibimos la especificación detallada del trabajo a realizar que se describe a continuación:

El BSE necesita los siguientes indicadores de negocio. Éstos se especifican a continuación

- *Tiempos de espera del formulario DVI y DVM en cada oficina*
- *Tiempo de trámite desde presentación de denuncias obreras y patronales hasta que se procesa el siniestro en el sistema rector. Indicando si para el siniestro ha habido una investigación.*
- *Cantidad de días de rentas temporarias abonadas por el banco desde el inicio del proceso con el sistema ADTEP hasta la fecha.*
- *Cantidad de rechazos/aceptaciones por:*
    - *Denuncias Vinculadas, Reaperturas y reconsideraciones.*
    - *Montevideo e interior*
    - *por tipo de rechazo (abandono, rehúsa asistencia, rechazo médico, administrativo, sin perjuicio de seguir investigando)*
    - *por aceptación (con o sin pagos, con o sin seguro)*
- *Cantidad de Amparos por parámetros (por producto, etc.)*

El lector no tiene por qué entender la especificación ya que contiene lenguaje del negocio del Sistema, solo se agrega al documento a modo cuantificar lo que se realizó.

Figura 6.5 - Indicadores pedidos por BSE

La implementación de estos indicadores nos llevó aproximadamente 18 horas. Esto fue un excelente resultado en cuanto a trabajo realizado y nos dio a nosotros y al cliente una imagen clara de lo útil que puede ser el sistema en casos reales.

Luego de tener implementado estos indicadores se los presentamos en una demo al cliente y, al momento de la redacción de este informe, estamos coordinando una reunión con el BSE para mostrarle los resultados de sus indicadores.

El testeo del sistema con un caso real nos dio una gran satisfacción personal y los resultados reflejan la finalización de un producto en su primera versión. Si bien quedan por delante muchas funcionalidades a agregar, podemos decir que las funcionalidades básicas para implementar indicadores de negocio en un sistema de análisis de información a partir de bases de datos documentales están implementadas.

6.3 Herramienta de diseño de contextos y cubos

Luego de diseñar la herramienta ETL, se tomó la decisión de implementar otra aplicación con interfaz web en la cual los usuarios de la solución puedan definir y configurar todas las herramientas utilizadas. Se decidió crear una única aplicación que nuclee tanto la configuración de la ETL como la configuración de la herramienta OLAP Pentaho.

La decisión de implementar una aplicación con interfaz web se debe a las capacidades ya conocidas que tienen este tipo de aplicaciones. Son aplicaciones prácticas ya que son accedidas desde cualquier navegador web, tienen independencia del sistema operativo y son muy fáciles de actualizar y mantener.

En la etapa de diseño escogimos Apache Tomcat como servidor de aplicaciones web. La siguiente decisión que debimos tomar, era elegir las tecnologías a utilizar para la implementación. En este caso optamos por usar Java Server Faces (JSF) (ORACLE 2014). JSF es una tecnología y framework para aplicaciones Java basadas en web que simplifica el desarrollo de interfaces de usuario en aplicaciones Java. La decisión de utilizar JSF se debe principalmente a la experiencia que se tenía implementado este tipo de aplicaciones. JSF permite con gran facilidad implementar aplicaciones con interfaz web. Además, para implementar los componentes de la interfaz se utilizó la biblioteca PrimeFaces (PrimeTek 2014). Esta es una biblioteca de componentes para JSF de código abierto, que cuenta con un conjunto de componentes enriquecidos que facilitan la creación de las aplicaciones web.

6.3.1 Configuración del ambiente para la implementación de la herramienta de diseño de cubos

Una vez seleccionada las tecnologías a utilizar, se configuró un ambiente de desarrollo. Para programar se utilizó el IDE Eclipse en su versión Juno. Eclipse es uno de los IDE's más utilizados a la hora de hablar de Java.

Como herramienta de versionado de código y repositorio se utilizó SVN. SVN es una herramienta de control de versiones Open Source basada en un repositorio cuyo funcionamiento se asemeja al de un sistema de ficheros.

Como servidor y repositorio SVN se utilizó Assembla. Esta es una herramienta on-line gratuita que permite crear repositorios para proyectos. Maneja versiones y además mantiene la auditoría de todas las acciones que ejecutan los usuarios con sus respectivos comentarios.

Como servidor de aplicaciones se descargó el servidor Apache Tomcat en su versión 7.0.42. Se configuró además el Eclipse para que realice el despliegue de la aplicación automáticamente en este servidor.

Se creó un proyecto del tipo Dynamic Web Project. Este tipo de proyecto es utilizado para crear aplicaciones web dinámicas, en nuestro caso una aplicación JSF. Para poder utilizar la tecnología JSF es necesario configurar el proyecto para que utilice las bibliotecas requeridas para implementar este tipo de aplicaciones.

Se configuró como core JSF Apache MyFaces JSF Core-2.0 (The Apache Software Foundation 2014). Este es un proyecto de la fundación Apache que provee la implementación de la especificación de JSF 2.0.

Finalmente para configurar PrimeFaces se descargó de la página web de PrimeFaces el archivo Jar primefaces-4.0.jar que contiene la biblioteca PrimeFaces versión 4.0. Se agregó dicho archivo al BuildPath (configuración donde se especifican las dependencias del proyecto) del proyecto creado.

6.3.2 Pantallas de configuración implementadas para diseñar los cubos

En la siguiente sección se explicarán las páginas web implementadas para realizar la configuración de la aplicación.

6.3.2.1 Inicio

En el inicio de la aplicación se despliega una lista con todos los contextos configurados. Además existen dos botones que permiten editar un contexto seleccionado o crear un nuevo contexto.

6.3.2.2 Contexto

La página de contexto contiene la información de configuración de un contexto. En esta pantalla se configuran los siguientes datos.

- Nombre

  Nombre por el cual se va a identificar el contexto a configurar.

- Borrar Tablas

  Nos indica si al correr el proceso de ETL se borran los datos en recorridas anteriores o si se versionan los datos procesados manteniendo los datos extraídos anteriormente.

- Cantidad de documentos

  Indica la cantidad máxima de documentos a procesar.

- Data Source

  Indica el Data Source a utilizar configurado en la aplicación Pentaho.

- Base de Datos

  Indica el nombre de la base de datos en la cual se van a almacenar los datos transformados

- Activo

  Indica si el contexto se encuentra activo o no. En el caso que esté activo al ejecutar la ETL se procesaran los datos configurados en el contexto.

- Lista de Fuentes

  La lista de Vistas de Lotus Domino que contienen los documentos de las bases de datos no relacionales a procesar.

Además desde esta pantalla se podrán agregar al contexto formas de extracción, dimensiones y medidas.

6.3.2.3 Forma de extracción

La página de forma de extracción nos permite configurar las distintas formas de extracción utilizadas en el contexto seleccionado. En esta pantalla se configura el siguiente dato.

- Nombre

  Nombre por el cual se identificará la forma de extracción dentro del contexto.

6.3.2.4 Dimensión

La página de dimensión nos permite configurar las distintas dimensiones configuradas en el contexto seleccionado. En esta página se configuran los siguientes datos.

- Nombre

  Nombre por el cual se identificará la dimensión dentro del contexto.

- Nombre Tabla

  Nombre de la tabla que se creará en la base de datos relacional utilizada para crear el DW

Además desde esta página se podrán agregar a esta dimensión campos y jerarquías.

6.3.2.5 Campo

La página de campo nos permite configurar los distintos campos para una dimensión seleccionada. En esta página se configuran los siguientes datos.

- Nombre

  Nombre por el cual se identificará el campo dentro de la dimensión. Además este será el nombre de la columna en la tabla de la dimensión configurada anteriormente.

- Tipo

  Tipo del campo en la tabla de la dimensión seleccionada

- Forma de Extracción

  La forma de extracción indica el método de extracción previamente definido que se utilizará a la hora de realizar la extracción y la transformación del dato que se insertará en la tabla de la dimensión seleccionada.

- Multivaluado

  Indica si para un único documento se extraen más de una dimensión por iteración. Ej. Un documento que posee el nombre de tres personas, al ser multivaluado va a generar tres tuplas a la tabla de la dimensión Persona.

- Parámetros

  Lista de parámetros que recibe el extractor seleccionado.

6.3.2.6 Jerarquía

La página de jerarquía nos permite configurar las jerarquías de la dimensión seleccionada que se mostrarán en la aplicación Pentaho. En esta página se configuran los siguientes datos.

- Nombre

  Nombre por el cual se identificará la jerarquía dentro de la dimensión seleccionada.

- Lista de niveles

  Es una tabla en la cual se configuran los distintos niveles dentro de una jerarquía. La tabla posee los campos etiqueta, indica la etiqueta a mostrar dentro de la jerarquía, columnaOrdinal, nos indica el campo o columna de la tabla de la dimensión por el cual se ordenará el nivel de la jerarquía, columna, nos indica el campo o columna de la tabla de la dimensión que será mostrado en el nivel de la jerarquía y tipo, nos indica el tipo de dato del nivel de la jerarquía.

6.3.2.7 Hecho

La página de Hecho, nos permite configurar los distintos hechos del contexto seleccionado. En esta página se configuran los siguientes datos.

- Nombre

  Nombre por el cual se identificará el hecho dentro del contexto.

- Nombre Tabla

  Nombre de la tabla que se creará en la base de datos relacional utilizada para crear el DW.

- Dimensiones

  Lista de dimensiones que se podrán utilizar para cruzar información.

Además desde esta página se podrán agregar a este hecho medidas.

6.3.2.8 Medida

La página de Medida, nos permite configurar las distintas medidas para el hecho seleccionado. En esta página se configuran los siguientes datos.

- Nombre

  Nombre por el cual se identificará una medida dentro de un Hecho.

- Tipo

  Tipo de dato de la medida.

- Etiqueta

  Etiqueta que se mostrará en Pentaho para indicar la medida dentro de un cubo.

- Operación

  Es la operación de agregación que se utilizará en el roll-up. Ej. Promedio o suma.

- Obligatorio

  Indica si es obligatorio que el cálculo de la medida de un valor para insertar una tupla en la tabla de hechos.

- Forma de Extracción

  La forma de extracción indica el método de extracción previamente definido que se utilizará a la hora de realizar el cálculo de la medida que se insertará en la tabla de la medida seleccionada.

- Multivaluado

  Indica si para un único documento se calcula más de un dato por iteración (ej. tiempo de espera de un formulario). Dado que un formulario tiene más de una sección, por cada una de ellas se calcula un tiempo de espera diferente.

- Parámetros

  Lista de parámetros que recibe el extractor seleccionado.

6.3.3 Generación de archivos de contexto

Este archivo de configuración guarda toda la información necesaria para que el componente ETL extraiga, transforme y cargue los datos de varias bases de datos documentales en el DW, y a su vez incluye información adicional necesaria para generar el archivo esquema multidimensional utilizada por Pentaho para visualizar y analizar dichos datos.

Para poder manejar la creación y edición de múltiples contextos vimos necesario manejar un conjunto de ellos, donde el usuario escoge en la página de inicio cual es el que quiere editar o crear uno nuevo. Agregado a este problema se encuentra la necesidad de conocer varias ubicaciones en el sistema de archivos: donde se guardan los archivos de contexto, donde se guardan los esquemas multidimensionales para que los pueda consumir Pentaho, y sobre este último la ubicación del archivo donde se publican los cubos. Toda esta información la guardamos en otro XML para poder modificarla dinámicamente sin necesidad de recompilar el código.

Para el manejo de estos archivos XML utilizamos la biblioteca JAXB el cual simplifico mucho el trabajo de parseo y creación de los mismos. El funcionamiento de dicha biblioteca ya fue explicado en la sección 6.2.4

6.3.4 Testing

Testeo unitario del controlador SchemaController para asegurarnos que las operaciones del manejo de los XML's de configuración funcionen correctamente antes de invocarlos desde la interfaz. También se realizó el testeo unitario para la creación de archivos "schemas" de Pentaho (esquema multidimensional) publicándolo en el mismo y asegurándose que la sintaxis sea correcta.

El testeo de la interfaz web se realizó en dos etapas. Para la primer etapa se contaba con un archivo de contexto que fue el que se utilizó para realizar las pruebas de la ETL con los indicadores que se mencionan en los requerimientos. La prueba consistió en crear un XML con las mismas características desde la aplicación web y luego se realizó una comparación de los mismos, constatando que no hubiera diferencias.

La segunda etapa consistió en crear nuevos indicadores al XML y se probaron en la ETL verificando que el funcionamiento de la extracción sea el esperado. Finalmente se probó la integración con Pentaho publicando en dicho servidor el schema creado, verificando que las jerarquías especificadas tuvieran la estructura correcta y que las operaciones de agregación presenten resultados acordes a los tipos de operaciones configuradas (promedio, suma, etc.).

7 Resultados obtenidos a lo largo del proyecto

En esta sección se va a hablar de los resultados obtenidos a lo largo del proyecto. Los resultados se separarán en dos categorías.

Primero se hablará de los resultados durante el trabajo, esto es de qué manera cumplimos con los requerimientos planteados. Luego los resultados de la solución donde se explican cuáles fueron las aplicaciones desarrolladas y que servicios brindan. Finalmente se describe la retroalimentación que se obtuvo desde una organización importante como es BSE, luego de presentarles nuestra solución.

7.1 Resultados generales del trabajo realizado

Como primer resultado se destaca que se cumplió con todos los requerimientos, y se colmaron ampliamente las expectativas de los integrantes del proyecto, la empresa ISA Ltda., y el BSE.

La primera etapa tuvo como resultado un documento con un análisis detallado de distintas herramientas de BI disponibles en el mercado, acotando únicamente a cuáles fueran Open Source. La devolución por parte de ISA Ltda. fue muy buena, y el documento nos sirvió como punto de partida para adentrarnos en las temáticas que competen al proyecto.

Un resultado tal vez no tan tangible, pero no por eso menos importante, fue superar el desafío de no centrarnos únicamente en lo conocido tratando de resolver un problema puntual. En lugar de esto, se decidió apostar por encontrar una manera de hacer el trabajo que permita ampliar el resultado. Con esto nos referimos a buscar una solución al problema de manera tal que pueda aplicarse a otros problemas de similares características. Por ejemplo, podríamos habernos ligado fuertemente al módulo de iGDoc formularios, pero en lugar de eso, se pensó una solución que pueda utilizarse sobre cualquier aplicación desarrollada con Lotus Domino. Esto a nivel de grupo fue todo un desafío, ya que no siempre es sencillo salirse de los lineamientos. Sin embargo luego de haber analizado el problema base y gracias a esta mirada más amplia, pudimos desarrollar una solución con el valor agregado mencionado anteriormente.

El tiempo del proyecto se alargó un poco más de lo estimado en un principio, y duró aproximadamente 1 año y medio. Este retraso se debe principalmente a un error de estimación en el tiempo de trabajo de realizar la documentación del proyecto. Otro factor influyente fue el desafío del grupo de extender el alcance del proyecto para lograr una solución con el valor agregado mencionado anteriormente. Todo esto se puede ver con más detalle en el ANEXO 3. Creemos de todos modos que fue una buena decisión ya que gracias a esto, el resultado final deja una puerta abierta al trabajo a futuro y la utilización de esta solución a otros problemas.

7.2 Resultados de la solución implementada

El resultado del proceso de implementación se puede dividir en tres herramientas que conforman la solución diseñada y desarrollada por el equipo del proyecto.

La primer herramienta resultado de este proyecto es una herramienta de ETL desarrollada utilizando la tecnología DOTS de Lotus Domino. Esta herramienta tiene la capacidad de extraer diversa información almacenada en bases de datos documentales Lotus Domino. Posteriormente procesa y transforma estos datos con formato y estructura variable para luego almacenarlos en una base de datos racional

MySQL ya con una estructura fija y formato definido. Como resultado del proceso de la herramienta obtenemos un DW donde se encuentra la información procesada que alimentará la herramienta Pentaho donde se puede realizar un análisis de los datos extraídos.

La segunda herramienta es donde se realizan los análisis de los datos procesados por la ETL. Esta herramienta es Pentaho. Es la herramienta de análisis OLAP que seleccionamos luego de un proceso de investigación donde comparamos las herramientas OLAP libres de licenciamiento disponibles en el mercado. Pentaho es la herramienta que mejor se adaptaba a nuestra solución. Con esta herramienta podemos conectarnos al DW resultado del proceso de ETL y desde allí realizar reportes sobre esos datos. Podemos analizar distintos cubos multidimensionales en donde podemos cruzar distintas dimensiones y obtener resultados o medidas para poder analizar determinado hecho.

La tercera herramienta que completa la solución es una aplicación con interfaz web para configurar las dos herramientas anteriormente mencionadas. Denominamos esta aplicación como herramienta de diseño de contextos y cubos. Posee una interfaz web sencilla e intuitiva que da la noción de una única configuración en lugar de dos configuraciones diferentes. En ella podemos definir donde extraer los datos. Esto es que bases de datos Lotus Domino vamos a procesar. Luego de esto definimos las diferentes dimensiones de los cubos. De las dimensiones especificamos cómo vamos a extraer los datos, que transformación les vamos a aplicar, como vamos a almacenar los mismos en el DW y como Pentaho debe presentar los datos almacenados. Luego de definida las dimensiones podemos definir las medidas. De las medidas definimos que dimensiones utiliza, como se calcula, como es almacenada en el DW y como Pentaho debe presentar los datos almacenados. Finalmente esta aplicación se encarga de publicar los contextos, proceso en el cual se configura la ETL y la herramienta Pentaho.

La solución resultado de este proyecto presenta una gran escalabilidad y permite intercambiar sus componentes, por ejemplo utilizar otro tipo de base de datos, o alguna herramienta BI que no sea Open Source. Esto se debe a la modularización aplicada en el diseño.

7.3 **Resultados del caso de estudio con el BSE**

Por parte del BSE, se relevó qué información les serviría tener disponible en una herramienta de análisis BI, y dichos requerimientos se cubrieron con la herramienta desarrollada sin necesidad de cambios, obteniendo los resultados esperados por el BSE. Esto nos da la pauta, que en un sistema real, en funcionamiento y de gran porte, la herramienta cumplió con lo esperado, y deja en claro que es posible utilizarla en su versión actual.

Se realizó una presentación en el BSE. A la misma asistieron por parte del BSE, Ana Erosa, gerente de proyectos, Álvaro Fierro, analista, Beatriz Fraquia, supervisora del producto accidentes de trabajo del BSE y Mabel Iraola, gerente de reclamaciones. Por parte del equipo del proyecto de grado asistieron Juan Martín Valsangiacomo y Francisco Álvarez. En la presentación primero se explicó en qué consistía el proyecto y el marco de trabajo. Luego se realizó una demo para los usuarios gerenciales en la cual se mostró las capacidades de la herramienta con datos reales de su negocio. Finalmente para los usuarios técnicos se mostró el funcionamiento del resto de la herramienta, ya sea configuración y mantenimiento, y los requerimientos necesarios para implantar la solución.

La presentación de la herramienta en el BSE, fue muy alentadora y la respuesta de los presentes fue de asombro por la potencia de lo que se estaba mostrando. Los presentes fueron usuarios de iGDoc con roles gerenciales y dos de las personas que más intervinieron por parte del BSE en el desarrollo de la solución del módulo de formularios. En esta etapa se relevó información sobre indicadores a desarrollar en trabajo futuro.

8 Trabajo a futuro

En esta sección se describen aquellas funcionalidades que aunque no se implementaron, realizarlas haría de este producto una herramienta práctica y aún más útil para las organizaciones. Se describirán estas mejoras para cada componente implementado.

8.1 **ETL**

Para comenzar, se debería encriptar la contraseña de la base de datos del DW. En el archivo de contexto se debe ingresar el usuario y la contraseña para acceder a la base de datos para poder realizar operaciones sobre la misma. Se podría agregar algún tipo de encriptación para que la contraseña no figure explícitamente en lenguaje natural donde cualquiera que lea el archivo de contexto puede verlo.

Se podrían realizar algunas mejoras a la aplicación para mejorar la performance en tiempo de procesamiento. Se podría optimizar el código desarrollado, mejorando los algoritmos y las operaciones realizadas.

Se debería implementar un log en el cual se registren todos los errores sobre documentos que no se hayan podido procesar correctamente. También se deberían registrar las actualizaciones que se hayan realizado a las bases de datos y el tiempo de procesamiento. Un log es necesario para que quede registrado todo lo ocurrido en el proceso de extracción, transformación y carga lo cual nos permitirá analizar posibles problemas.

Se podría mejorar la estructura del XML del contexto para que sea un poco más mantenible. Por ejemplo reestructurarlo para que los extractores sean definidos dentro de las dimensiones y no a nivel de contexto.

También se podría mejorar el método de extracción. Hoy en día se extraen primero las dimensiones y luego las medidas. Esto hace que si una medida no retorna un valor no nulo, ya sea “” o 0, de todos modos previamente se extraen y se persisten las dimensiones. Se podría mejorar que si la extracción de la medida retorna un valor nulo, no se extraigan y almacenen los valores de las dimensiones para esa medida. Tampoco se insertará un hecho que tenga todos los valores de las medidas nulos o 0.

Se podrían realizar mejoras en la clase DAOManager para hacerla más genérica y por lo tanto permitir una variedad más amplia de manejadores de bases de datos. Por ejemplo bases de datos relacionales que no usan el lenguaje SQL en sus consultas (NoSQL 2014).

8.2 **Herramienta de diseño de contextos y cubos**

Como trabajo a futuro podemos plantear algunas mejoras con respecto a manejo de errores a nivel de interfaz. Si bien a nivel de controladores se manejan errores, falta implementar a nivel de interfaz gráfica las páginas que muestran los errores al usuario.

Todavía no se realizó el manejo de sesiones de usuarios. Se podría agregar usuarios y roles por ejemplo, consultando algún servidor de directorio como puede ser LDAP. De este modo los usuarios deberían iniciar sesión antes de poder ingresar a la aplicación de configuración.

Se puede mejorar la interfaz gráfica. Si bien se utilizó Prime Faces que ya posee estilos predefinidos, se podría crear estilos propios que mejore visualmente la

aplicación. Además, se podría mejorar aún más a nivel de usabilidad, por ejemplo se podría guardar automáticamente los cambios hechos cuando se edita algún elemento en vez de tener que explícitamente realizar click en el botón de guardar.

Se podría mejorar todavía más la capacidad de la aplicación a la hora de configurar la aplicación Pentaho, ya que se implementaron los métodos de configuración para un subconjunto de las capacidades que este posee. Por ejemplo permitir crear dimensiones a nivel de los hechos en vez de a nivel del cubo para restringir su uso solo a dicho cubo.

9 Conclusiones

El objetivo de esta sección es resumir y concluir las experiencias del proyecto a lo largo del año y medio de trabajo.

9.1 Proceso de desarrollo del proyecto

Para comenzar podemos decir que el proyecto implementó la totalidad del alcance definido en la etapa de análisis. Si bien se extendió más de lo planificado en el tiempo, atribuimos eso a problemas externos al trabajo meramente técnico.

El alcance se definió en el equipo de trabajo: Libertad Tansini, equipo de trabajo del proyecto, y cliente. Al poder implementar todo el alcance fue muy gratificante para el equipo poder brindarle al cliente, y la facultad, el trabajo terminado. Si bien se podría realizar mejoras, descriptas en la sección 8, el núcleo del sistema está implementado y funcionando.

La otra parte del trabajo, la que no fue puramente técnica, como lo es la planificación, la gestión y el seguimiento de las tareas, fue para nosotros un problema complejo. Muchos de nosotros no teníamos experiencia en la gestión de proyectos, y la experiencia obtenida en el ámbito laboral, siempre nos ponía en el lugar del técnico y no del gerente de proyecto. Para mitigar el impacto de este problema apelamos al compromiso y nos planificamos el trabajo semanal dividiendo tareas, haciendo una reunión semanal para coordinación.

El trabajo de: planificación, división del equipo, tareas realizadas, tiempos estimados, etc. se detalla en el ANEXO 3: Ejecución del proyecto.

A modo de conclusión podemos decir que si bien hubo retrasos en la planificación inicial, se puede afirmar que la planificación fue buena ya que el gran retraso se dividió en pocas tareas. En el anexo mencionado se explica la causa del retraso general y se compara el tiempo estimado contra el tiempo en ejecución de cada tarea implementada.

9.2 Plano técnico

En el plano técnico, destacamos la estabilidad y el buen funcionamiento de las herramientas elegidas para la implementación del proyecto. Por ejemplo DOTS y JSF.

Si bien DOTS es una herramienta nueva en el mercado, lo que generaba a priori varias dudas al respecto de su funcionamiento, no tuvimos ningún problema en instancias de implementación. Esto se debe al temprano prototipado de la solución, donde pudimos comprobar que dicha herramienta se comportaba como esperábamos.

La herramienta ETL implementada tiene la capacidad de ser extensible a nivel de la implementación de los extractores, permitiendo crear distintos tipos de extractores dependiendo de la estructura de las fuentes de datos y del negocio objetivo.

Gracias a que fue implementado utilizando la tecnología DOTS, el procesamiento de la extracción se realiza en otra tarea distinta al servidor HTTP sin consumir capacidad de procesamiento de este último. Por ejemplo, esto permite reiniciar la tarea donde se ejecutan los DOTS sin la necesidad de reiniciar el servidor HTTP.

Otro punto a destacar es que puede utilizar como fuente de datos cualquier base de datos domino. Esto es gracias a la flexibilidad que ofrece, debido a que el núcleo de la herramienta no fue implementado de tal manera que no se ata a alguna aplicación

específica. En nuestro caso de estudio, puede ejecutarse tanto con el módulo de Formularios de iGDoc como cualquier otro, por ejemplo Expedientes.

El tiempo de ejecución de un ciclo de la herramienta ETL fue dentro de lo esperado. Como es una herramienta que no pretende mostrar datos en tiempo real sino de momentos específicos en el tiempo. Por ejemplo, se puede programar para realizar el proceso en la noche, teniendo al día siguiente los datos listos para ser analizados.

Pentaho fue la herramienta de análisis OLAP que seleccionamos luego del proceso de investigación realizado. Fue seleccionado de entre varias herramientas de licenciamiento libre disponibles en el mercado. Es la herramienta que mejor se adaptaba a nuestra solución. Gracias a ella, se pudo conectar al DW (resultado del proceso de la ETL), para poder analizar dichos datos. Inclusive nos permitió poder mostrar en una demo a usuarios gerenciales de BSE, el potencial que tiene el análisis de los datos de dicha organización y lo rápido sencillo que es realizarlo desde esta herramienta.

La aplicación web no era un requerimiento planteado por el cliente. Decidimos implementarla por un tema de practicidad a la hora de la creación de los archivos de configuración XML de las dos aplicaciones involucradas, y de esta forma enriquecer la solución.

Si no se utiliza la Herramienta de Diseño de Cubos y Contextos implementado en nuestra solución, se debe realizar lo siguiente.

Para configurar las dos aplicaciones era necesario realizar dos procesos separados cada uno con su propia complejidad. La configuración del ETL implicaba un XML complejo, donde en principio su creación y edición se debía realizar mediante un programa de edición de texto. Esto puede introducir errores de sintaxis y de coherencia a la hora de armar la estructura. Para luego copiar dicho archivo al sitio del sistema de archivos del servidor donde se encuentra instalado el servidor domino.

Para visualizar los datos en Pentaho se puede realizar la creación del modelo multidimensional desde un programa de edición de texto o desde la aplicación Schema Workbench (Pentaho 2014). Esta se instala en el servidor donde se encuentra Pentaho. Pero esta aplicación tiene las desventajas de que esta en inglés y utiliza términos más técnicos que complican el aprendizaje de los usuarios no técnicos. Esto incluye la sintaxis de Mondrian que debe utilizar para la creación de estos archivos. Además, se tiene que tener en cuenta los datos que se habían ingresado en el contexto para que Pentaho pueda realizar la consulta a las estructuras correspondientes en el DW, y una vez creada se tiene que publicar en Pentaho utilizando la misma aplicación.

En contraste, utilizando la aplicación web el usuario percibe una creación lógica de un solo archivo de configuración. Se elimina la necesidad de tener que aprender y asociar dos estructuras con sintaxis diferentes (Mondrian y XML de Contexto), y además se puede realizar la publicación directamente desde allí sin necesidad de conocimiento técnico, basta con realizar click al botón correspondiente. Además tenemos la ventaja que el usuario que configura la solución, no tiene por qué tener acceso directo a los servidores, ni permisos a nivel del sistema operativo para poder manipular el sistema de archivos del mismo. Puede acceder remotamente desde cualquier dispositivo que disponga navegador web con acceso al dominio web del servidor donde este instalada la herramienta.

9.3 **Repercusiones de la solución implementada**

Luego de la presentación realizada en el BSE se recibió una lista de indicadores solicitados por parte de los usuarios gerenciales del sistema de formularios instalado en el BSE, demostrando el interés existente en la herramienta desarrollada en este proyecto. Es importante aclarar que los indicadores solicitados en esta etapa, no requieren modificaciones a la solución desarrollada.

Durante dicha presentación se pudo observar un entusiasmo muy grande de los usuarios gerenciales por el potencial que posee la herramienta la cual resuelve un problema existente hoy en día en su organización. Este problema es el de tener acceso a los datos de manera rápida, organizada y estructurada

9.4 **En resumen**

El desarrollo del proyecto llevó más del tiempo esperado, donde la parte de documentación fue lo que más complico debido a la falta de experiencia del equipo en este tema.

El modo de trabajo utilizado consideramos que fue de gran ayuda para lograr las metas planteadas. Al dividirnos de a pares pudimos paralelizar trabajo, permitiendo avanzar de manera continuada sin la necesidad de estar los cuatro presentes a la hora de realizar el trabajo. Al ser cuatro tuvimos la posibilidad de definir un alcance bastante amplio y nos permitió desarrollar un prototipo funcional muy completo. Aunque dicha cantidad incrementa la complejidad de la comunicación entre los integrantes, pudimos mitigar problemas que podrían surgir gracias a que cada grupo realizaba tareas lo más independientes posibles. Por ejemplo, un grupo trabajando en el desarrollo de la ETL y otro con la Herramienta de diseño de Contextos y Cubos.

Quedamos muy conformes con la solución obtenida ya que al final del proyecto pudimos visualizar todo lo que habíamos ideado en el principio. A eso se agrega los comentarios positivos obtenidos por una organización consolidada como es el BSE luego de mostrarles la solución implementada.

Finalmente concluimos que pudimos culminar la totalidad del proyecto con una solución que no solo cumplió nuestras expectativas, sino que tiene el potencial para ser utilizada en organizaciones de gran importancia como lo es BSE.

10 Glosario

Base de datos documentales

Una base de datos documental está constituida por un conjunto de programas que almacenan, recuperan y gestionan datos de documentos o datos de algún modo estructurados. A diferencia de las bases de datos relacionales, estas bases de datos están diseñadas alrededor de una noción abstracta de “documento”.

Open Source

Es la expresión con la que se conoce al software distribuido y desarrollado libremente. Se focaliza más en los beneficios prácticos (acceso al código fuente) que en cuestiones éticas o de libertad que tanto se destacan en el software libre. En este proyecto se enmarca en sistemas sin licenciamiento pago.

BI

Business Intelligence. Se denomina inteligencia empresarial, inteligencia de negocios o BI (del inglés Business Intelligence), al conjunto de estrategias y aspectos relevantes enfocados a la administración y creación de conocimiento sobre el medio, a través del análisis de los datos existentes en una organización o empresa.

Pentaho

Es una herramienta de BI, que permite el análisis de información. Es la herramienta seleccionada en este proyecto en base a una comparación con otras restantes existentes en el mercado.

ETL

Extract, Transform and Load («extraer, transformar y cargar», frecuentemente abreviado ETL) es el proceso que permite a las organizaciones mover datos desde múltiples fuentes, reformatearlos y limpiarlos, y cargarlos en otra base de datos para analizar, o en otro sistema operacional para apoyar un proceso de negocio.

DW

En el contexto de la informática, un almacén de datos (del inglés data warehouse) es una colección de datos orientada a un determinado ámbito (empresa, organización, etc.), integrado, no volátil y variable en el tiempo, que ayuda a la toma de decisiones en la entidad en la que se utiliza.

ISA Ltda.

Es la empresa Uruguaya que propuso este proyecto de grado.

iGDoc

Es un sistema implementado sobre bases de datos documentales, más específicamente sobre IBM Lotus Domino por la empresa ISA Ltda.

Lotus Domino

IBM Notes (anteriormente Lotus Notes) es un sistema software cliente/servidor de colaboración y correo electrónico, desarrollado por Lotus Software, filial de IBM.

OLAP

Es el acrónimo en inglés de procesamiento analítico en línea (On-Line Analytical Processing). Es una solución utilizada en el campo de BI cuyo objetivo es agilizar la consulta de grandes cantidades de datos. Para ello utiliza estructuras multidimensionales (o Cubos OLAP) que contienen datos resumidos de grandes Bases de datos o Sistemas Transaccionales (OLTP).

O3

O3 Browser es el componente de análisis de IdeaSoft. Ofrece gran capacidad de análisis de la información gracias a un potente ambiente gráfico que requiere una mínima capacitación para su explotación.

Proporciona acceso a los modelos de análisis personalizados, y le permite navegar a través de los mismos de forma intuitiva, sin requerir conocimientos detallados de las estructuras de datos subyacentes.

SQL

El lenguaje de consulta estructurado o SQL (por sus siglas en inglés Structured Query Language) es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones en ellas. Una de sus características es el manejo del álgebra y el cálculo relacional que permiten efectuar consultar con el fin de recuperar de forma sencilla información de interés de bases de datos, así como hacer cambios en ellas.

Web Services

Un servicio web (en inglés, Web Service o Web services) es una tecnología que utiliza un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Distintas aplicaciones de software desarrolladas en lenguajes de programación diferentes, y ejecutadas sobre cualquier plataforma, pueden utilizar los servicios web para intercambiar datos en redes de ordenadores como Internet.

HTML

HTML, siglas de HyperText Markup Language («lenguaje de marcas de hipertexto»), hace referencia al lenguaje de marcado para la elaboración de páginas web. Es un estándar que sirve de referencia para la elaboración de páginas web en sus diferentes versiones, define una estructura básica y un código (denominado código HTML) para la definición de contenido de una página web, como texto, imágenes, entre otros.

Java

Java es un lenguaje de programación de propósito general, concurrente, orientado a objetos y basado en clases que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible.

Lotus Script

LotusScript es un lenguaje de programación interpretado en tiempo de ejecución utilizado por diversos productos de IBM como Lotus SmartSuite, Lotus 1-2-3, Lotus Approach, Freelance Graphics, Word Pro y Lotus Notes/Domino.

Ajax

JAX, acrónimo de Asynchronous JavaScript And XML (JavaScript asíncrono y XML), es una técnica de desarrollo web para crear aplicaciones interactivas o RIA (Rich Internet Applications). Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre las páginas sin necesidad de recargarlas, mejorando la interactividad, velocidad y usabilidad en las aplicaciones.

JavaScript

Es un lenguaje de programación interpretado. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.

WSDL

WSDL son las siglas de Web Services Description Language, un formato XML que se utiliza para describir servicios Web (Web Services).

OSGI

OSGI son las siglas de Open Services Gateway Initiative, más precisamente el OSGi14. Su objetivo es definir las especificaciones abiertas de software que permita diseñar plataformas compatibles que puedan proporcionar múltiples servicios. Fue pensado principalmente para su aplicación en redes domésticas y por ende en la llamada Domótica o informatización del hogar.

Spring

Spring es un framework para el desarrollo de aplicaciones y contenedor de inversión de control, de código abierto para la plataforma Java.

DOTS

Domino OSGI Tasklet Service (DOTS) en Domino son la generación siguiente a los denominados Agentes. DOTS, como lo indica la sigla, son tareas que corren en el framework OSGI del servidor Domino.

API

Interfaz de programación de aplicaciones (IPA) o API (del inglés Application Programming Interface) es el conjunto de funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

AMGR

Es una tarea de Domino, que es encargada de ejecutar agentes.

BSE

Es el Banco de Seguros del Estado del Uruguay. Entidad pública que se utilizó en este proyecto como cliente real donde se utilizaron sus datos para pruebas y presentaciones.

MDX

Las expresiones multidimensionales (MDX es el acrónimo de MultiDimensional eXpressions) es un lenguaje de consulta para bases de datos multidimensionales sobre cubos OLAP, se utiliza en Business Intelligence para generar reportes para la toma de decisiones basados en datos históricos, con la posibilidad de cambiar la estructura o rotación del cubo.

J2EE

Java Platform, Enterprise Edition o Java EE (anteriormente conocido como Java 2 Platform, Enterprise Edition o J2EE hasta la versión 1.4; traducido informalmente como Java Empresarial), es una plataforma de programación—parte de la Plataforma Java—para desarrollar y ejecutar software de aplicaciones en el lenguaje de programación Java. Permite utilizar arquitecturas de N capas distribuidas y se apoya ampliamente en componentes de software modulares ejecutándose sobre un servidor de aplicaciones.

MySQL

MySQL es un sistema de gestión de bases de datos relacional, multihilo y multiusuario con más de seis millones de instalaciones. Desarrollada como software libre.

XML

XML, siglas en inglés de eXtensible Markup Language ('lenguaje de marcas extensible'), es un lenguaje de marcas desarrollado por elWorld Wide Web Consortium (W3C) utilizado para almacenar datos en forma legible.

OpenNTF

OpenNTF, es una comunidad que provee frameworks para desarrollar aplicaciones Open Source sobre Lotus Domino.

Assembla

Assembla es una empresa que provee herramientas de colaboración y de seguimiento de errores (Bug Tracking System) y tareas basadas en la nube para organizar y administrar proyectos de código abierto y comerciales para el desarrollo de software.

SVN

Subversion (SVN) es una herramienta de control de versiones Open Source basada en un repositorio cuyo funcionamiento se asemeja enormemente al de un sistema de ficheros. Es software libre bajo una licencia de tipo Apache/BSD.

VMWare

VMware es un sistema de virtualización por software. Un sistema virtual por software es un programa que simula un sistema físico (un computador, un hardware) con unas características de hardware determinadas. Cuando se ejecuta el programa (simulador), proporciona un ambiente de ejecución similar a todos los efectos a un computador físico (excepto en el puro acceso físico al hardware simulado), con CPU (puede ser más de una), BIOS, tarjeta gráfica, memoria RAM, tarjeta de red, sistema de sonido, conexión USB, disco duro (pueden ser más de uno), etc.

DBMS

Una base de datos o banco de datos es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso. En este sentido; una biblioteca puede considerarse una base de datos compuesta en su mayoría por documentos y textos impresos en papel e indexados para su consulta.

JDBC

Java Database Connectivity, más conocida por sus siglas JDBC,1 2 es una API que permite la ejecución de operaciones sobre bases de datos desde el lenguaje de programación Java, independientemente del sistema operativo donde se ejecute o de la base de datos a la cual se accede, utilizando el dialecto SQL del modelo de base de datos que se utilice.

JAXB

Java Architecture for XML Binding (JAXB) permite a los desarrolladores Java asignar clases de Java a representaciones XML. JAXB proporciona dos características principales: la capacidad de serializar las referencias de objetos Java a XML y la inversa, es decir, deserializar XML en objetos Java.

ADTEP

Es el nombre proyecto del BSE enmarcado en el contexto de accidentes de trabajo y enfermedad profesional. Es el proyecto que enmarca la parte práctica de este proyecto de grado.

DVI, DVM

Denuncia Vinculada Interior y Denuncia Vinculada Montevideo. Son dos de los formularios más usados en el proyecto ADTEP.

11 Bibliografía

Para las referencias bibliográficas se utilizó el estilo Chicago Fifteenth Edition, cuya especificación se accede en el libro “The Chicago Manual of Style, 15th Edition”.

"AJAX." *Wikipedia.* 2014. http://es.wikipedia.org/wiki/AJAX.

Alliance, OSGi. *The OSGi Architecture.* 2014. http://www.osgi.org/Technology/WhatIsOSGi.

Assembla. *Assembla.com.* 2014. https://www.assembla.com/home (accessed 2014).

Bedoya, Manuel. *Slideshare.* 2014. http://www.slideshare.net/d1305/qu-es-la-gestin-documental-12635702 (accessed 2014).

Binding, Java Architecture for XML. n.d. http://www.oracle.com/technetwork/articles/javase/index-140168.html.

BPM-Conseil. 2014. http://www.bpm-conseil.com/ (accessed 2014).

BSE. 2014. http://www.bse.com.uy/bse/.

Carpani, Fernando. "Master Thesis de Fernando Carpani." *CMDM: A conceptual multidimensional model for Data Warehouse.* Universidad de la República, 2000.

"Código Abierto." *Wikipedia.* 2014. http://es.wikipedia.org/wiki/C%C3%B3digo_abierto (accessed 2014).

"Data Warehouse Definition." *1Keydata.* 2014. http://www.1keydata.com/datawarehousing/data-warehouse-definition.html.

"Definiciones de Conceptos - Base de datos Estratégicas." n.d. http://definiciondeconceptos.blogspot.com/ (accessed 2014).

"Descripción data warehouse." n.d. http://es.wikipedia.org/wiki/Almac%C3%A9n_de_datos#mediaviewer/File:Data_warehouse_overview.JPG (accessed 2014).

Díaz, Josep Curto. *Introducción al Business Intelligence.* El Ciervo 96 S.A., 2012.

Engineering Ingegneria Informatica S.p.A. *SpagoBI.* 2014. http://www.spagoworld.org/xwiki/bin/view/SpagoBI/ (accessed 2014).

Erosa, Ana M. "Integración en estado: Caso de accidentes de trabajo y enfermedades profesionales." *agesic.gub.uy.* 2012. http://www.agesic.gub.uy/innovaportal/file/2423/1/integracion_en_estado_caso_de_accidentes_de_trabajo_y_enfermedades_profesionales.pdf (accessed 2014).

ETL. "ETL." *Wikipedia.* 2014. http://en.wikipedia.org/wiki/Extract,_transform,_load (accessed 2014).

"FOSS." *Wikipedia.* 2014. http://es.wikipedia.org/wiki/Software_libre_y_de_c%C3%B3digo_abierto (accessed 2014).

G. Quintana, M. Marqués, J.I. Aliaga, M.J. Aramburú. *Aprende SQL.* Universitat Jaume I., 2008.

Galit Shmueli, Nitin R. Patel, Peter C. Bruce. *Data Mining for Business Intelligence.* John Wiley & Sons, 2011.

"HTML." *Wikipedia.* 2014. http://es.wikipedia.org/wiki/HTML (accessed 2014).

"HTTP." *RFC-Base.* 2014. http://www.rfc-base.org/txt/rfc-2616.txt (accessed 2014).

Hyde, Julian. *Pentaho Mondrian Documentation.* 2011. http://mondrian.pentaho.com/documentation/schema.php (accessed 2014).

IBM. *IBM Domino.* 2014. http://www-03.ibm.com/software/products/es/ibmdomino (accessed 2014).

—. *IBM Domino Designer.* 2014. http://www-03.ibm.com/software/products/es/ibmdominodesigner/ (accessed 2014).

—. *Lotus Script Language Guide.* IBM, 2005.

Ideasoft. *Ideasoft - Technology and Business Performance.* Vers. http://www.ideasoft.biz/. 2014. http://www.ideasoft.biz/ (accessed 2014).

ISA Ltda. 2014. http://isaltda.com.uy/.

java.net. *Project JAXB.* 2014. https://jaxb.java.net/ (accessed 2014).

JavaScript. 1995. http://javascript.about.com/od/reference/p/javascript.htm (accessed 2014).

Jedox. *Jedox Products.* 2014. http://www.jedox.com/en/products/jedox-base.html (accessed 2014).

Mark Humphries, Michael W. Hawkins, Michelle C. Dy. *Data Warehousing.* Prentice Hall Inc., 1999.

"MDX." *Wikipedia.* 2014. http://en.wikipedia.org/wiki/MultiDimensional_eXpressions (accessed 2014).

"Modelado Dimensional." *Base de Datos Dimensionales.* 2014. http://bddimensionales.wikispaces.com/Modelado+Dimensional (accessed 2014).

Mondrian Documentation. 2014. http://mondrian.pentaho.com/documentation/ (accessed 2014).

Newcomer, Eric. *Understanding Web Services.* David Chappell, 2002.

"Modelado Multidimensional." In *Diseño y explotación de almacenes de datos*, by Jesús Pardillo Vela, Juan Carlos Trujillo Mondejar Norberto Mazón López, 43, 44. Editorial Club Universitario, 2010.

"NoSQL." *Wikipedia.* 2014. http://es.wikipedia.org/wiki/NoSQL (accessed 2014).

OpenNTF. *Introduction to DOTS.* n.d. http://es.slideshare.net/flinden68/let-me-introduce-you-dots.

—. *XPages.info.* 2014. http://xpages.info/XPagesHome.nsf/Home.xsp (accessed 2014).

ORACLE. *Java.* 2014. http://www.java.com/es/ (accessed 2014).

—. "JDBC." *Java SE Documentation.* 2014. http://docs.oracle.com/javase/7/docs/technotes/guides/jdbc/ (accessed 2014).

—. "JSF." *Oracle.com.* 2014. http://www.oracle.com/technetwork/java/javaee/javaserverfaces-139869.html (accessed 2014).

—. "MySQL Workbench." *MySQL.* 2014. http://www.mysql.com/products/workbench/ (accessed 2014).

—. *Oracle Database 11g.* 2014. http://www.oracle.com/webapps/dialogue/ns/dlgwelcome.jsp?p_ext=Y&p_dlg_id=14009505&src=7878529&Act=130&sckw=WWMK13048383MPP011 (accessed 2014).

Paul Fiore, David Taieb, Niklas Heidloff. *OSGI Tasklet Service for IBM Lotus Domino.* 2001. http://www.openntf.org/main.nsf/project.xsp?r=project/OSGI%20Tasklet%20Service%20for%20IBM%20Lotus%20Domino (accessed 2014).

Pentaho. 2014. http://community.pentaho.com/ (accessed 2014).

Peralta, Verónika. "Tesis de Maestría." *Diseño Lógico de Data Warehouses a partir de Esquemas Conceptuales Multidimensionales.* Instituto de Computación – Facultad de Ingeniería, 2001.

Pivotal Software. "Spring Framework." *Spring.* 2014. http://projects.spring.io/spring-framework/ (accessed 2014).

PrimeTek. *PrimeFaces.* 2014. http://www.primefaces.org/ (accessed 2014).

S. Sumanthi, S. Esakkirajan. *Fundamentals of Relational Database Management Systems.* Springer, 2010.

The Apache Software Foundation. *Apache MyFaces Project.* 2014. http://myfaces.apache.org/ (accessed 2014).

—. *Apache Tomcat.* 2014. http://tomcat.apache.org/ (accessed 2014).

—. *subversion.apache.org.* 2014. https://subversion.apache.org/ (accessed 2014).

The Eclipse Foundation. *Eclipse.* 2014. https://www.eclipse.org/org/ (accessed 2014).

Thomsen, Erik. *OLAP Solutions.* John Wiley & Sons Inc., 2014.

TIBCO. *Jaspersoft.* 2014. http://community.jaspersoft.com/ (accessed 2014).

VMware. *vmware.com.* 2014. http://www.vmware.com/ (accessed 2014).

"XML." *Wikipedia.* 2014. http://es.wikipedia.org/wiki/Extensible_Markup_Language (accessed 2014).

Yunta, Luis Rodríguez. "BASES DE DATOS DOCUMENTALES: ESTRUCTURA Y PRINCIPIOS DE USO." *Universidad de Navaria.* 2001. www.unav.es/dpp/documentacion/proteger/lryunta.pdf (accessed 2014).

12 Indice de Figuras

ANEXO 1: iGDoc

iGDoc es el sistema del cliente en el cual nos basamos para hacer el proyecto. Este sistema está hecho en Domino (servidor) implementando sobre bases de datos Documentales. iGDoc es una plataforma documental completa para proveer a las organizaciones de una solución para gestionar a través de una interfaz Web diversas formas documentales tales como: expedientes, formularios, comunicaciones, contratos, resoluciones, entre otros.

Cuando hablamos de plataforma documental, nos referimos a un sistema de información construido sobre bases de datos documentales. En este caso, bases de datos Domino, producto de IBM.

iGDoc es una solución completa porque no requiere de ningún otro componente adicional. Esto significa que incluye un servidor Web, un servidor de aplicaciones Web/Java/Web Services, un motor de bases de datos documental, un motor de búsqueda textual, un directorio LDAP, un emisor de certificados electrónicos e inclusive la capacidad de clúster en forma nativa sin requerir esta funcionalidad en el sistema operativo. Esto último es lo que contiene el servidor en donde corre esta plataforma, el Domino Server de IBM.

Desde el año 2003 iGDoc está presente en varias organizaciones de diversos tamaños abarcando Gobiernos Municipales, Entes Autónomos, Bancos, Ministerios y Direcciones de Gobierno.

Es multiplataforma, ya que se puede instalar en Microsoft Windows, Linux, Mac Os, AIX, Solaris, etc.

Además iGDoc adopta las reglamentaciones que permiten reemplazar el uso de documentos en papel por documentos electrónicos. En Uruguay cumple con las leyes de Procedimiento Administrativo y Firma Electrónica tales como el Decreto 500, Decreto 65/998, Ley N° 18.600 Documento Electrónico y Firma Electrónica, entre otros.

El sistema se basa en conceptos que solo están disponibles en bases documentales. Cómo se explicó en la sección 2.1, las bases documentales gestionan documentos auto-contenidos, seguridad a nivel de datos del documento, la posibilidad de indexar por palabras localizadas en cualquier campo o archivo adjunto de cada uno de los documentos.

En la actualidad al sistema iGDoc lo utilizan 15.000 usuarios, se generan 200.000 documentos (en todas sus formas) al mes y se mueven 900.000 con la misma frecuencia. Actualmente se están administrando 20:000.000 de documentos.

Sobre esta plataforma comenzamos nuestro proyecto de grado, tomando el módulo formularios como base para el desarrollo del problema. A continuación se describe cómo funciona el módulo iGDoc-Formularios, según el relevamiento hecho en la etapa de análisis.

iGDoc-Formularios

El módulo formularios de iGDoc gestiona y agiliza trámites definidos. Para esto se deben crear y configurar lo que se llaman tipos de formularios. Estos tipos definen: el flujo del trámite, la numeración, condiciones de visualización, motivos de finalización y archivado, si se va a firmar digitalmente en el inicio del trámite o no, etc.

Lo más importante de los tipos de formularios es que definen que elemento de diseño de la base de datos lo implementa. Es decir, donde reside el código HTML, JavaScript, Java, LotusScript que hacen al funcionamiento del formulario.

En el flujo del formulario se definen las secciones que va a tener el mismo, y a cada sección se le asignará una unidad, pudiendo elegir distintos tipos de unidad, por ejemplo: “cualquier unidad”, “unidad superior”, “unidad inicial”, o una unidad de la estructura de la organización.

Posteriormente a la configuración del tipo de formulario se puede comenzar a generar formularios de ese tipo.

Cuando se inicia un formulario se podrán ejecutar acciones sobre el mismo (siempre y cuando se tengan los roles necesarios) como pueden ser: editar, guardar, finalizar, archivar, dar pase, reservar, recibir, etc.

Una vez creado el formulario, comienza a transitar el flujo definido en el tipo de formulario. En primera instancia el formulario va a quedar en tránsito a la segunda sección del flujo, quedando en la bandeja de salida de la unidad correspondiente a la sección uno y en la bandeja de entrada de la unidad asignada en la sección dos. Cualquier usuario de la unidad que esté asignada a la sección dos deberá recibir el formulario, luego de realizar lo que corresponda en esta sección del trámite se deberá firmar digitalmente el formulario para a posteriori darle “pase” de tal forma que el formulario quede en tránsito a la sección siguiente. Esta acción colocará el formulario en la bandeja de salida de su unidad y en la bandeja de entrada de la siguiente unidad que corresponda con la siguiente sección (sección 3). Estos pasos se repetirán para todas las secciones que tenga el flujo del tipo de formulario.

De esta forma va quedando registrado qué se realiza en cada sección quedando certificado mediante la firma digital cada accionar.

Una vez que el formulario llega a la sección final, el formulario se finaliza y se da por terminado el trámite, registrando usuario, fecha y hora de la acción.

Pantalla principal

Al acceder al módulo de Formularios se visualizará una pantalla similar a la que se muestra en la Figura A1.1.

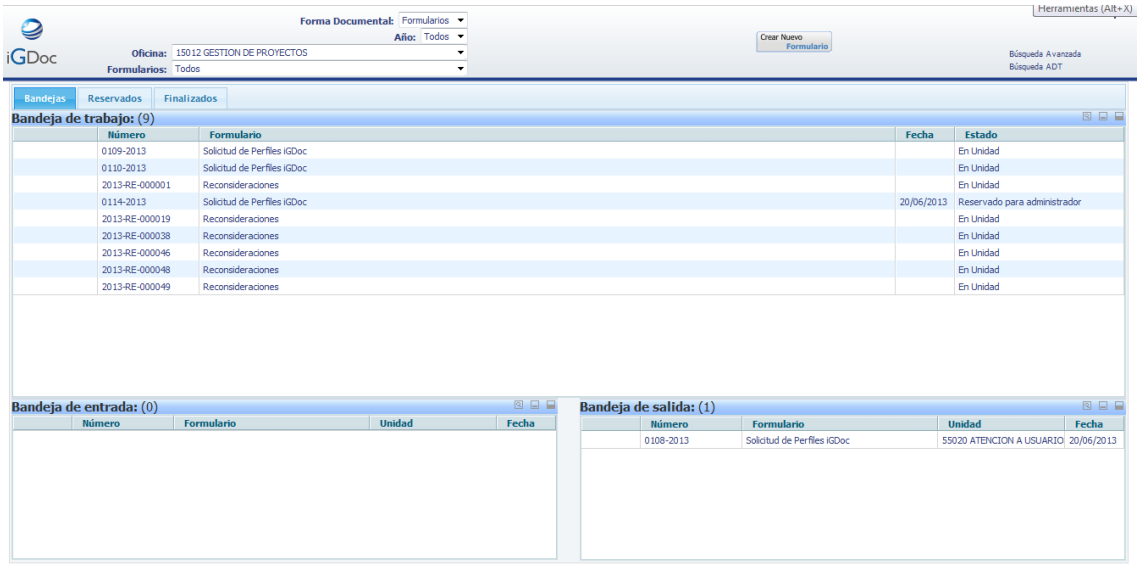


Figura A1.1 – Página de inicio del módulo Formularios de iGDoc

En el Encabezado se encuentra:

- La identificación de la forma documental (formularios, expedientes, documentos, etc.), mostrado en la Figura A1.2.

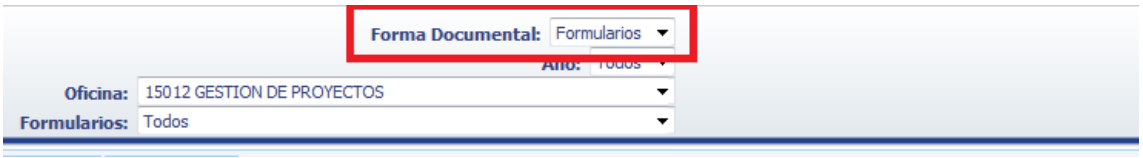


Figura A1.1 – Captura de pantalla mostrando donde se especifica la forma documental

- La identificación de la Oficina a la que pertenece el usuario, en caso que forme parte de más de una oficina podrá seleccionarla, mostrado en la Figura A1.3.

Figura A1.3 – Captura de pantalla mostrando donde se especifica la oficina

- El año de los documentos que se verán en las bandejas, mostrado en la Figura A1.4.

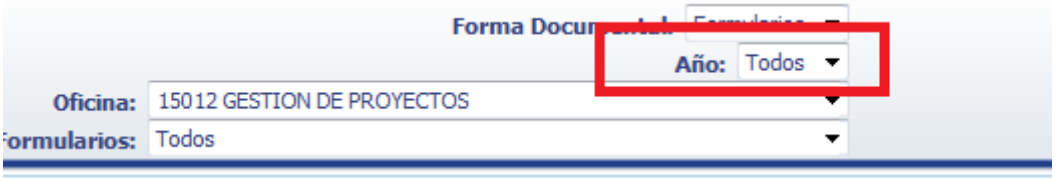


Figura A1.2 – Captura de pantalla mostrando donde se especifica el año de los documentos

Las Bandejas visibles en la pantalla principal son:

- **Bandeja de Trabajo:** contiene los formularios que se encuentran en la oficina a la que pertenece el usuario. Como se muestra en la Figura A1.5, se visualiza el Número de formulario, el tipo, la fecha y el Estado del documento

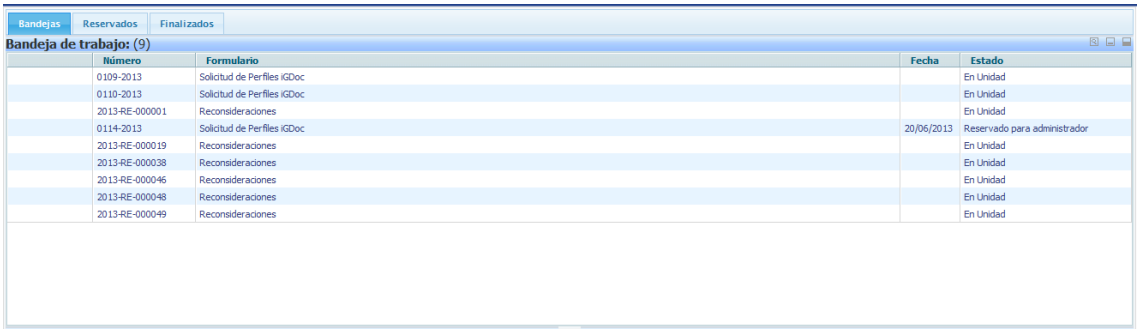


Figura A1.5 – Captura de pantalla mostrando la Bandeja de Trabajo

- **Bandeja de Entrada:** contiene los formularios que han sido enviados por otras unidades y no han sido recibidos aún, por lo que su estado es EN TRÁNSITO. Como se muestra en la Figura A1.6, se visualizan los campos: Número de formulario (Número), tipo (Formulario), Unidad de Origen (Unidad) y Fecha.

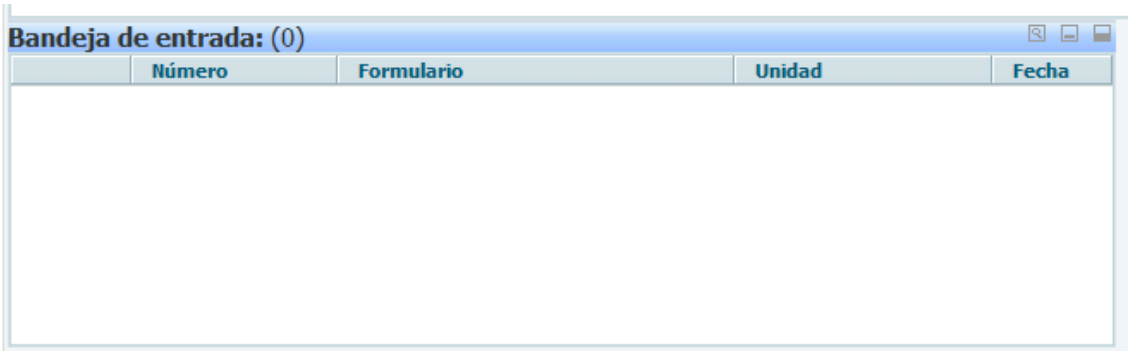


Figura A1.6 – Captura de pantalla mostrando la Bandeja de Entrada

- **Bandeja de Salida:** contiene los formularios que han sido enviados por la unidad en la que se encuentra el usuario y no han sido recibidos aún, por lo que su estado es EN TRÁNSITO. Visualiza los campos: Número de formulario, tipo, Unidad Destino y fecha.

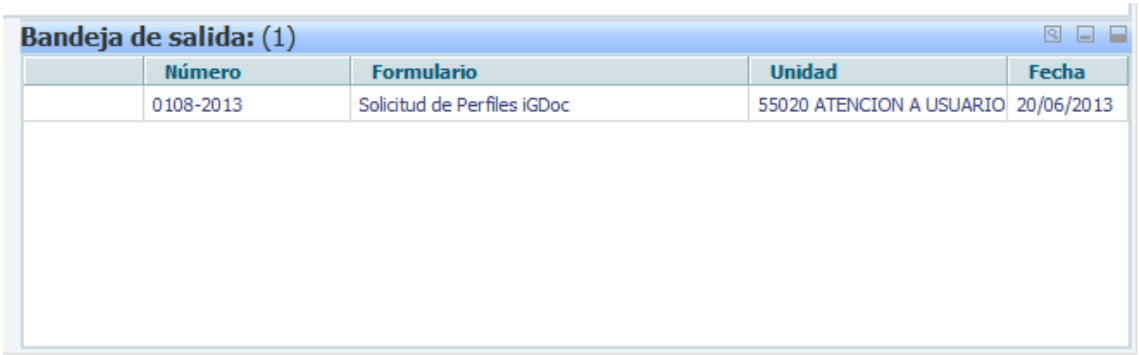


Figura A1.7 – Captura de pantalla mostrando la Bandeja de Salida

Estados del formulario

A medida que se van realizando las distintas operaciones sobre los formularios, éstos van cambiando su estado según una máquina de estados como se muestra en la Figura A1.8.

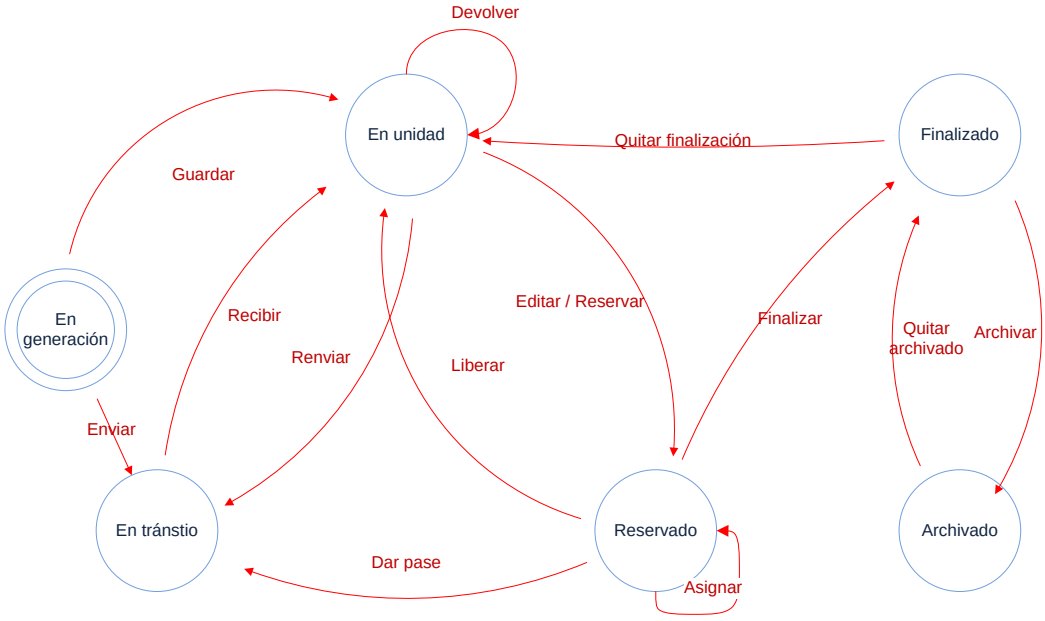


Figura A1.8 – Máquina de estados del flujo de los documentos de iGDoc

Operaciones sobre formularios

Recibir formulario

Dar por recibido un formulario es lo que lo habilita para que se comience a trabajar con él. Para poder hacerlo, el usuario de la oficina accede a él desde su Bandeja de Entrada. Con el formulario abierto, se debe seleccionar Recibir en el menú Acciones.

El formulario pasará de la Bandeja de Entrada a la Bandeja de Trabajo y su estado pasará a ser EN UNIDAD.

Devolver / Reenviar

Una vez recibido el formulario el sistema permite realizar las acciones Devolver y Reenviar para aquellos casos que el formulario se haya recibido en una unidad que no correspondiera.

La operación Devolver permite al usuario devolver el formulario a la unidad de origen. La operación se habilita solo luego de recibido el formulario.

El formulario quedará en estado EN UNIDAD en la bandeja de la unidad a la que fue devuelto.

Esta operación se podrá realizar a partir del paso 3 del formulario en adelante, no en el caso del paso siguiente al inicial, en el que la operación no estará habilitada.

La operación Reenviar permite al usuario reenviar el formulario a otra unidad sin necesidad de actuar o dar pase. La operación se habilita solo luego de recibido el formulario.

El formulario quedará en estado EN TRANSITO en la bandeja de salida de la unidad de origen y en la de entrada de la unidad a la que fue reenviado.

Al reenviar el sistema mantiene la sección de datos correspondiente para que la unidad a la que se reenvía el formulario pueda actuar.

Reservar / Asignar

El sistema permite que el usuario que desee pueda reservar el formulario para trabajarlo en otro momento, esto puede hacerlo accediendo al formulario en el menú Acciones/Reservar.

El formulario quedará en estado RESERVADO para el usuario y lo podrá visualizar también en su Bandeja de Reservados.

El sistema también permite la asignación a otro usuario de la misma oficina a través de la acción Asignar.

Esta acción se habilita luego de editado por primera vez.

Editar / Guardar la sección correspondiente

Una vez recibido el formulario, se habilita la sección correspondiente a la etapa del formulario que debe ser completada por la unidad. Para poder comenzar a completar los campos del formulario se deberá ir al

menú Archivo/Editar.

Luego de completados los campos se debe guardar el formulario, en menú Acciones/Guardar. Solo permitirá guardar cuando se hayan completado todos los campos obligatorios.

Luego de guardada la sección del formulario, el sistema habilita las operaciones Liberar Formulario, Firmar – Firmar y Dar Pase, y Dar Pase.

Liberar Formulario

Esta operación deja un formulario previamente reservado, disponible para que otros editores de la misma unidad puedan trabajar con él. Puede realizar esta operación únicamente el usuario que tiene reservado el formulario.

Firmar – Firmar y Dar Pase

Para permitir que el formulario sea enviado a otra unidad, cuando se configure con firma obligatoria, la sección debe ser firmada por al menos una persona. Será necesario que tengan instalado en su máquina el certificado digital que garantiza la firma.

Cuando para el formulario se configure la firma obligatoria, solo habilitará la acción luego de firmada la sección de datos.

El sistema también permite realizar en forma abreviada la acción de firmar y dar pase.

Dar Pase

Esta operación, luego de completados todos los campos obligatorios del formulario, permite realizar el pase a la siguiente unidad.

El formulario quedará en estado EN TRANSITO entre la unidad de origen y la destinataria.

Visualizar recorrido

Esta operación muestra el recorrido que tiene definido el formulario, para poder visualizar su recorrido vamos a Acciones/Visualizar recorrido.

Una vez seleccionada esta opción el sistema desplegará en pantalla un diagrama definido en la configuración del formulario, detallando el recorrido del mismo.

Finalizar y Archivar Formularios

La operación Finalizar deja el formulario en estado FINALIZADO accesible en la bandeja de finalizados. La operación Archivar deja el formulario en estado ARCHIVADO accesible mediante búsquedas. Esta acción solo se habilitará luego de finalizado el formulario, en el menú Acciones/Archivar.

Imprimir

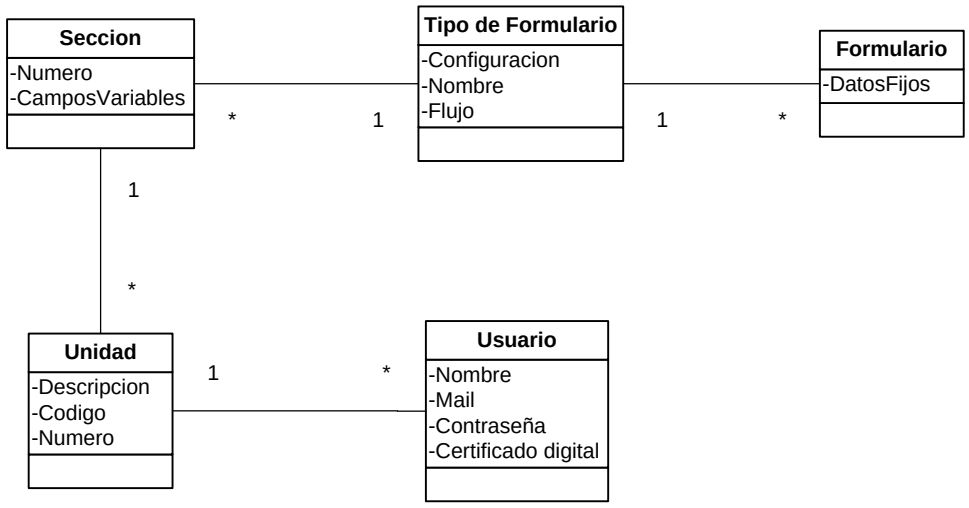
Esta acción del sistema permite visualizar el formulario en una nueva ventana desde donde se podrá imprimir el mismo.

Información del Recorrido

Esta acción del sistema permite conocer el recorrido del formulario, la información que se muestra detalla: el número de sección, la fecha de recepción y la fecha de salida.

Para poder ver el recorrido vamos al menú Acciones / Información del recorrido.

Conceptos técnicos y modelado de realidad



Un formulario es un documento de una base de datos documental (WebForms.nsf) que representa un trámite ingresado por un usuario del sistema iGDoc.

Los documentos que representan formularios tienen un subconjunto de campos fijos, y un subconjunto de campos variables, estos dependen de la realidad que quiere modelar cada cliente. Estos datos son los que les interesan persistir en la base de

datos a lo largo del trámite electrónico. Sin embargo los campos fijos, son los que hacen funcionar al módulo dentro de la plataforma y forman parte del producto, estos campos no pueden ser modificados por el cliente.

Un formulario tiene distintas secciones, cada sección se le asigna a una o varias unidades del sistema. Una unidad representa un grupo de usuarios del sistema. Cada unidad tiene un número, código y descripción.

Los usuarios tienen un nombre, una contraseña, un mail asociado y un certificado digital que permite firmar digitalmente cada sección en la que él realice cambios.

Cada usuario tiene distintos roles en cada unidad para cada tipo de formulario. Esto se configura en la estructura organizacional de la plataforma iGDoc y son comunes para todos los módulos del sistema.

Un tipo de formulario es la entidad que caracteriza cada formulario. Un tipo de formulario tiene un nombre, configuración general (motivos de finalización, si requiere firma o no, plazos por sección, notificaciones, cantidad de secciones, etc.) y un flujo.

El flujo está dado por un elemento inicio, un conjunto de secciones conectadas por: relaciones directas, condicionales, acciones (envío de mails, consumo de ws, modificaciones de campos, etc.), y un elemento final.

Cada sección del flujo tiene asociada una unidad. Estas secciones deberán estar ordenadas en forma ascendente entre el elemento inicio y fin, pudiendo estar alguna sección ausente.

En cada sección del formulario se podrán ejecutar acciones sobre el mismo (siempre y cuando se tengan los roles necesarios en la unidad), por ejemplo: editar, guardar, finalizar, archivar, dar pase, reservar, recibir, etc.

Un documento en la base de datos documental es una instancia de un tipo de formulario que representa un trámite en particular, ya sea en ejecución o finalizado.

Para ilustrar los conceptos definidos y modelar lo que se pudo relevar del sistema del cliente iGDoc en la etapa de análisis.

ANEXO 2: Relevamiento de herramientas OLAP

	Características y requerimientos técnicas					
Herramienta	**Usabilidad/facilidad instalación/configuración**	**Tecnología**	**Integración**	**Reportes**	**Requerimiento de hardware**	**Requerimiento de software**
Pentaho Community	Fácil instalación.	Java	Es fácil de integrar con módulos de otros productos.	Provee una herramienta de escritorio para crear fácilmente reportes complejos. (Pentaho report designer)	Disco: 100Gb. CPU: Pentium dual core. Ram: 2Gb	Windows: XP, Windows Server 2003, Windows Server 2008. 32 o 64 bits. Linux: todas sus distribuciones. Mac: 10.4 en adelante
				Pentaho Reporting Software Developer's Kit es un paquete de herramientas para desarrollar reportes		Sun JRE 5.0 +, Internet explorer 6.0 +, Mozila Firefox 3.5+ , Safari 2.0.3 +
Palo	Fácil instalación.					Windows: XP, Windows Server 2003, Windows Server 2008. 32 o 64 bits. Linux: todas sus distribuciones. Mac: 10.4 en adelante
Spago BI	Fácil instalación.	Java	No se integran módulos con otras herramientas	Tiene un componente para hacer reportes. Utiliza las herramientas: JasperReport, BIRT, Accessible report, BO generando reportes en formatos: HTML, PDF, XLS, XML, TXT, CSV, RTF	Disco: 160Gb. CPU: 2 cores. Ram: 4Gb	Cualquier sistema operativo que soporte JVM 1.5
						Manejador de bases de datos: MySQL, Postgres, Oracle, HSQL, Ingres, MS SQL Server
						Servidor de aplicaciones: Tomcat 6.0.18, JBoss, WebSphere
						Firefox 2+, Chrome 11+, IE 8+

BMP-Conseil Vanilla		Java	No se integran módulos con otras herramientas	FreeWebRepor aplicación web para la creación de reportes. (link a video)	Disco: 100Gb. CPU: 2 cores. Ram: 4Gb (u 8Gb depende de para que se use)	Cualquier versión/distribución de Windows o Unix que soporten JVM 1.6. Se sugiere Windows server 2008 R2 o Debian 6.
				Plugin de BIRT para generar reportes. (link)		Manejador de bases de datos: MySQL, Postgres,
				FreeDashboard. Para crear cuadros de mandos.		Servidor Tomcat incluido en el paquete de instalación
						Firefox 2+, Chrome 11+, IE 8+
Jaspersoft Community	Fácil instalación	Java	Se puede integrar a otras aplicaciones de forma embebida y cuenta con varias API's para comunicarse con el servidor	Posee una herramienta Open Source llamada Jaspersoft Studio la cual es basada en eclipse para diseñar reportes	No específica, ver link	Windows Vista/XP/7/2003/2008*, Red Hat Enterprise Linux (RHEL)*, Novell SUSE Linux*, Sun Solaris, OpenSolaris*, Fedora Enterprise Linux, Apple Mac OS X, Debian Linux, HP-UX, FreeBSD, IBM AIX, CentOS, Ubuntu.
				Posee además una biblioteca enteramente Java la cual es capaz de obtener datos de fuentes diferentes y generar reportes		Servidores de aplicaciones: Apache Tomcat*, JBoss, IBM WebSphere, Oracle WebLogic, Sun GlassFish, SpringSource tc Server
						Cualquier RDBMS compatible con JDBC 2.1 y SQL-92. Data sources: Bean, JNDI, JasperAnalysis (Mondrian), XML/A, Custom (ej. Hibernate,

						XML, etc.) Microsoft Internet Explorer, Mozilla Firefox, Apple Safari, Google Chrome
Links						
Pentaho				http://wiki.pentaho.com/display/Reporting/Report+Designer		
				http://reporting.pentaho.com/classic_engine.php		
Spago BI						
BPM Vanila				http://www.dailymotion.com/video/xc5fsa_freewebreport-report-creation_tech		
				http://www.bpm-conseil.com/downloads/documents/4.2/EN/BPM_Vanilla_BIRTPlugins_v4.2_EN.pdf		

				http://www.bpm-conseil.com/downloads/documents/4.2/EN/BPM_Vanilla_FreeDashboard_v4.2_EN.pdf		
Jaspersoft Community			http://www.youtube.com/watch?feature=player_embedded&v=jMZdcziltdo#!		http://community.jaspersoft.com/wiki/hardware-requirements-jasperreports-server	http://www.jaspersoft.com/supported-platforms

	Características y requerimientos no técnicos			
Herramienta	**Experiencias**	**Soporte**	**Licenciamiento**	**Evolución**
Pentaho Community		Foros activos y documentación de todos los módulos de la herramienta	Libre	Activo hasta la fecha. Última liberación 29/11/2012
Palo				

Spago BI		El soporte es pago. Tiene tres niveles de soporte de usuario(se detallan en el link) y soporte de mantenimiento. El precio de éste último, varía dependiendo de los servicios brindados, con un valor mínimo de 5000 euros por año.	Libre, Open Source.	Activo. Última liberación en febrero del 2012.
BMP-Conseil Vanilla		Soporte pago. Tiene tres niveles de servicio, se adjuntan en el link.	Libre.	Activo. Última versión 4.2, liberada en enero del 2013
Jaspersoft Community		Soporte a través de la comunidad. La versión comercial posee soporte por parte de Jaspersoft Corporation	Libre. Open Source	Archivo. Última versión 5.1.0, liberada el 9 de mayo del 2013
Links				
Pentaho		http://wiki.pentaho.com/display/COM/Community+Wiki+Home		

Spago BI		http://www.spagoworld.org/xwiki/bin/view/SpagoBI/SpagoBIUserSupport		
		http://www.spagoworld.org/xwiki/bin/view/SpagoBI/SpagoBIMaintenance		
BPM Vanila		http://www.bpm-conseil.com/services.html#div_support_platinum		
Jaspersoft Community				

ANEXO 3: Ejecución del proyecto

En este anexo se pretende documentar cómo fue la ejecución del proyecto según lo planificado, qué problemas tuvimos que causaron el no cumplimiento de la planificación inicial, cómo fuimos re planificando a medida que los plazos se nos extendían, y cómo gestionamos internamente los cambios en el plan inicial.

Planificación inicial y ejecución

El proyecto se divide en dos fases. La primer fase es el estudio y análisis de herramientas Open Source para el análisis de información. La segunda fase es la implementación de una herramienta para poder estructurar información, que proviene de bases de datos no relacionales, de forma tal que la misma se pueda analizar mediante una herramienta de análisis de información, gestionando indicadores de negocio.

Cada una de estas fases del proyecto se puede ver en detalle en la documentación entregada. En este anexo se quiere hacer foco en cómo se planificaron y cuando se ejecutaron cada una de las actividades que forman cada fase del proyecto, ver donde tuvimos más retraso, y el porqué de los mismos.

Fase I – Análisis de herramientas de BI

Desde la primer parte del proyecto decidimos dividir el equipo de forma tal de comenzar a paralelizar el trabajo desde las primeras fases. Por este motivo dividimos el trabajo de esta fase en cuatro partes. Para esto seleccionamos las ocho herramientas más utilizadas y de mejor referencia que encontramos en una investigación inicial, y designamos el estudio y análisis de dos herramientas por persona.

Definimos reuniones semanales para ver el avance de cada uno e ir juntando la información en un documento guía. Estas reuniones fueron tomando carácter y metodología con el correr del tiempo, llegando a ser reuniones de corto tiempo y de una intensidad de trabajo interesante.

Luego de que teníamos el documento guía con suficiente información, dividimos nuevamente el equipo para designar dos personas a la confección del documento de análisis final. Este trabajo consistió en darle formato y unificar la redacción, tiempos verbales, etc. Para que el documento realizado por cuatro personas distintas quede como un documento único y coherente en sus tiempos verbales y formas de redacción.

Las otras dos personas del equipo fueron designadas al análisis del producto iGDoc documentado en el Anexo 1. Si bien esta tarea está en la Fase II del proyecto, fue abordada desde el inicio del mismo.

Para graficar un poco el avance de la primer parte del proyecto se realizó el diagrama de Gantt inicial mostrado en la Figura A3.1.

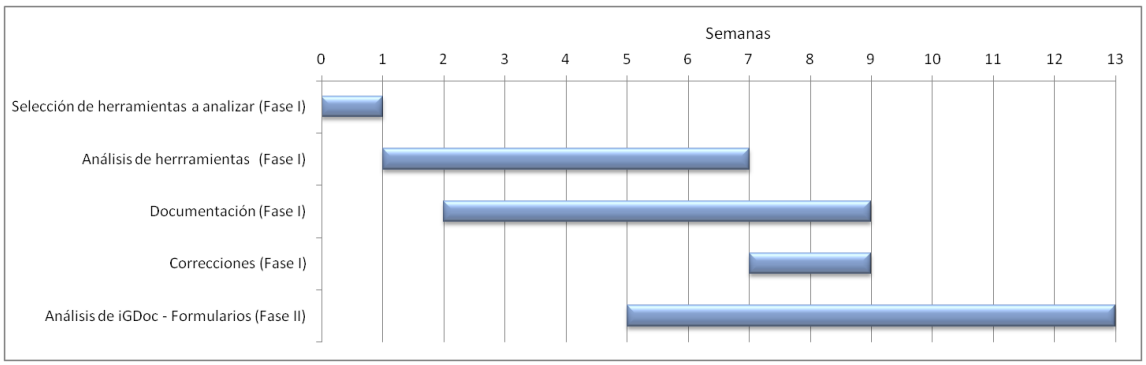


Figura A3.1 – Diagrama de Gantt planificación Fase I

Esto fue lo que se planificó en el inicio de la fase. Estimamos que en dos meses y una semana podríamos: seleccionar las herramientas a estudiar, analizar dichas herramientas y documentarlas.

Para este trabajo se estimó que necesitábamos dos semanas para realizar correcciones una vez que el documento se entregara al cliente del proyecto.

Por último en esta parte ya se anexó la tarea de la fase dos que es el análisis de iGDoc Formularios. Cómo se puede ver el Análisis se solapa en cuatro semanas con la documentación y las correcciones. Esto se debe a la división del equipo en dos grupos de dos personas.

En la ejecución de las tareas planificadas hubieron retrasos en dos de las cuatro planificadas en esta fase. Se adjunta el diagrama de Gantt en la Figura A3.2 del tiempo dedicado en la ejecución de manera de comparar gráficamente donde tuvimos el retraso.

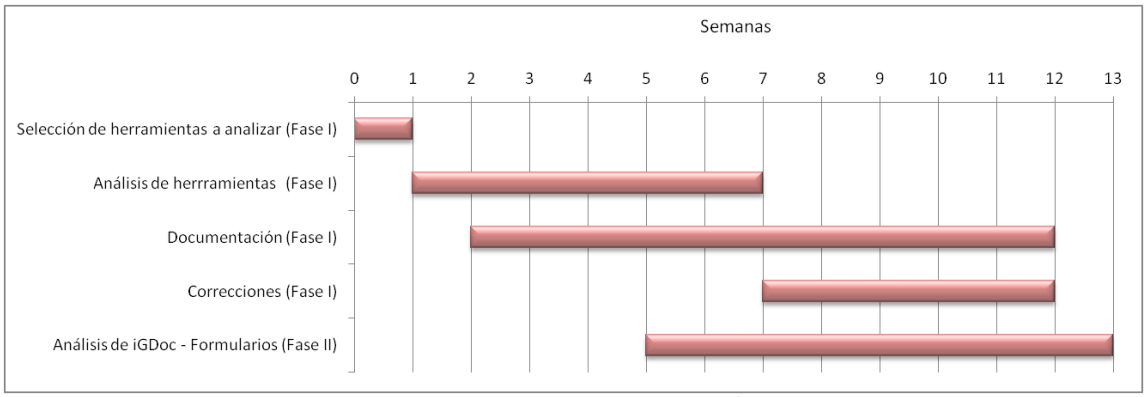


Figura A3.2 – Diagrama de Gantt Ejecución Fase I

En conclusión, la Fase I del proyecto nos llevó doce semanas, tres meses. Se comenzó la ejecución de dicha fase la semana dos del mes de abril, culminando dicha fase la semana dos de julio.

Para el cierre de la fase se entregó la documentación al cliente Isa Ltda, documentación que se anexa al documento corriente en ANEXO 2: Relevamiento de herramientas OLAP.

Fase II – Implementación del problema a resolver

En esta fase el trabajo se podría dividir en tres grandes etapas, cómo en toda implementación de un sistema tenemos una etapa de análisis, una etapa de diseño y una etapa de implementación.

En nuestro proyecto el análisis, el diseño y la implementación fueron de tres subsistemas: ETL, Diseñador de cubos, indicadores del cliente.

En la etapa de planificación se estimaron estas tres etapas, con sus actividades correspondientes según el diagrama de Gantt visto en la Figura A3.3.

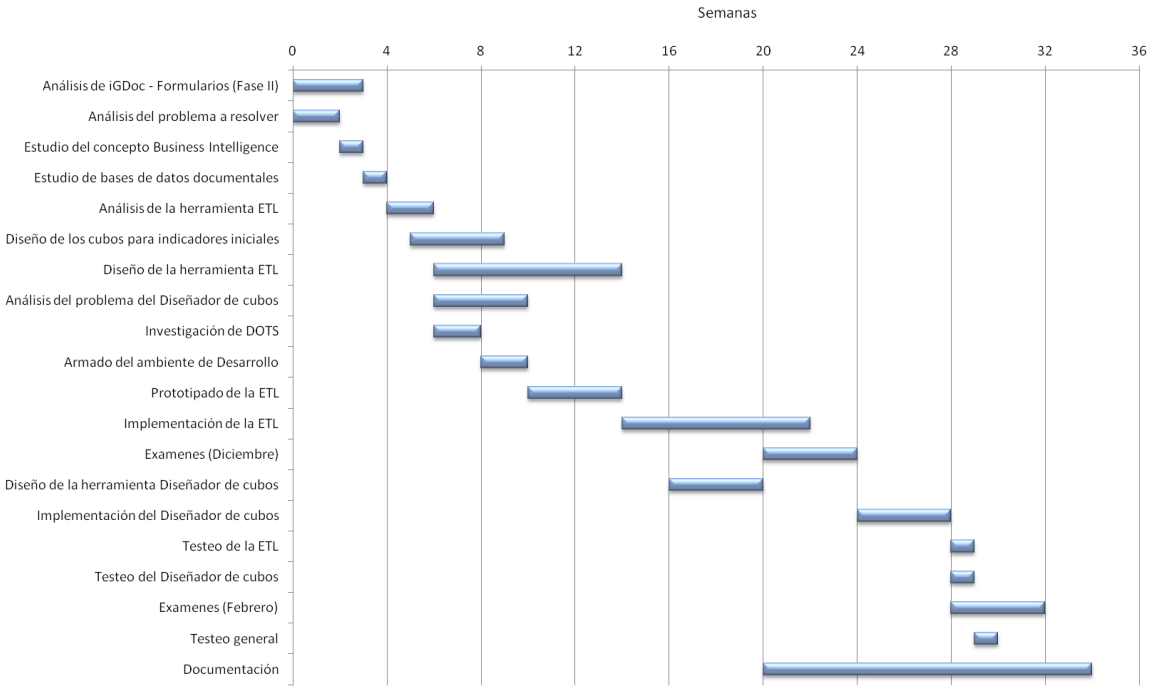


Figura A3.3 – Diagrama de Gantt planificación Fase II

Cómo se puede ver en el diagrama, más de la tercer parte del tiempo estimado es para el análisis de los subsistemas y la investigación de las herramientas utilizadas. Esto lo planificamos así debido a la gran incertidumbre sobre la posibilidad de desarrollar el sistema, y la cantidad de conocimiento que deberíamos adquirir para la implementación del mismo. Para clarificar: debíamos conocer el funcionamiento de bases de datos documentales, el sistema del cliente, el concepto de BI, OSGI y DOTS para correr tareas en servidores Domino, y la biblioteca para acceder a las bases de datos documentales, solo para poder implementar la ETL.

Otra de las observaciones que podemos realizar es la cantidad de tiempo estimado para la documentación. Sabíamos que la documentación iba a ser un pilar en el proyecto debido a que la primer fase del mismo era básicamente la documentación de las herramientas de BI, y para la segunda fase íbamos a tener que utilizar herramientas no del todo conocidas en Uruguay, por ende era fundamental la correcta documentación de las mismas.

En la planificación inicial el proyecto lo íbamos a entregar en Marzo del 2014. Es evidente el retraso grande que tuvimos para la implementación del mismo. Se adjunta

en la Figura A3.4 el diagrama de Gantt que representa los tiempos que nos llevaron la ejecución de las actividades planificadas.

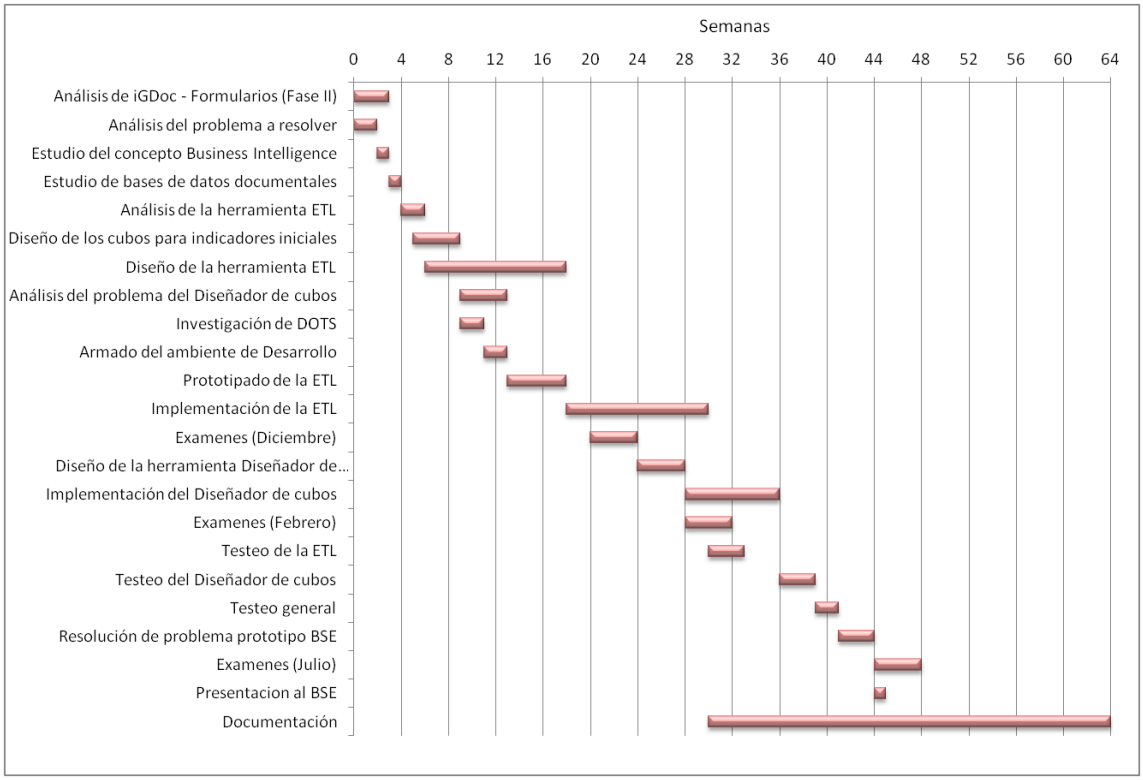


Figura A3.4 – Diagrama de Gantt ejecución Fase II

Se puede ver que la gran mayoría de las actividades se cumplieron en lo estimado, salvo por algunas actividades puntuales. He aquí nuestro mayor retraso y la justificación del por qué estamos entregando casi diez meses más tarde.

Si bien hubieron actividades que llevaron más de lo planificado, cómo lo fueron: el diseño de la herramienta ETL, la implementación de la herramienta ETL y la implementación del Diseñador de cubos. El gran error en la planificación fue estimar tan poco tiempo para la documentación.

También hubo tres cambios en el alcance que impactaron en el tiempo que nos llevó el proyecto. Estos fueron: el abordado del problema de la extracción, transformación y carga de datos de forma genérica, la implementación de un diseñador de los contextos y cubos, y la implementación de un prototipo de indicadores para el Banco de Seguros del Estado.

Estos cambios de alcance fueron consensuados por el equipo del proyecto, ya que nosotros siempre entendimos que el proyecto de grado no debería ser un trabajo más, sino que debería ser un trabajo que deje una marca y que pueda ser útil para alguien o que pueda ser la guía para un desarrollo a futuro.

Dichos cambios fueron los que retrasaron las actividades de diseño e implementación antes mencionadas, además de tener que agregar las actividades (final del Gantt en la Figura A3.4) con el BSE.

Por último concluir que para el cierre de la fase II del proyecto, hubo una demostración con el cliente Isa Ltda mostrando la herramienta funcionando y con el agregado de la flexibilidad no requerida por ellos desde el inicio.

También se organizó una demostración con el BSE y se les mostró como se implementaron los indicadores definidos en su prototipo, el potencial y la flexibilidad que tiene la herramienta para el desarrollo de estos indicadores. Se detalla en la sección 7.3 el retorno que nos hizo el BSE de la demostración indicada.

Por último, se entrega como cierre de la fase II y cierre del proyecto, la documentación corriente, dejando constancia de lo implementado durante el tiempo que nos llevó el proyecto.