Universidad de la República
Facultad de Ingeniería
Instituto de Computación

Proyecto de Grado

Implantación de Data Warehouse Open Free

13 de Diciembre del 2011
Montevideo - Uruguay

Autores:
Nicolás Gerolami
Esteban Revello
Germain Venzal

Tutores:
Nelson Calero
Lorena Etcheverry
Ariel Sabiguero

Tribunal:
Antonio López, Adriana Marotta, Raquel Sosa

Agradecimientos

Los resultados de este proyecto están dedicados a todas aquellas personas que, de alguna forma, son parte de su culminación. A Eduardo Bieñkowski, Gabriel Costa, Marcelo Lemos, Federico Ramos y Carlos Strozzi de ASSE por la colaboración brindada durante la realización de éste proyecto, a nuestros tutores y en especial a nuestros familiares, novias y esposas por el apoyo brindado todo este tiempo.

Resumen

Este proyecto surge como propuesta del área informática de la Administración de los Servicios de Salud del Estado (ASSE). Dicha institución tiene varios sistemas de información corporativos de los cuales interesa obtener reportes gerenciales. Hasta el momento dichos reportes no son obtenidos de forma automática sino que son realizados en forma manual por funcionarios de la institución, con los consiguientes problemas de calidad de datos que se pueden ocasionar, así como también la dificultad en la consolidación de los mismos.

El objetivo principal del proyecto es la implementación de un Data Warehouse corporativo utilizando tecnología Open Source que permita generar mensualmente y de forma automática dichos reportes. La primer etapa consistió en el análisis de prestaciones de las distintas soluciones disponibles en todas las capas de la solución, así como también un estudio de la calidad de los datos. La segunda etapa fue la implementación propiamente dicha utilizando la plataforma seleccionada en el punto anterior incluyendo la puesta en funcionamiento inicial, configuración, creación de trabajos de extracción, transformación y carga requeridos para el mantenimiento de la información necesaria. Se crearon reportes de gestión con los datos extraídos y se implementaron cuadros de mando e infografías en la capa de visualización de los datos. Para cumplir con éstos requerimientos se interactuó con distintos sistemas instalados en ASSE, por un lado el Sistema de Gestión de Afiliados y por otro lado el Sistema de Gestión de Salud para Atención Primaria (SGS-AP). Por último se realizó el traspaso de conocimiento a la institución en las herramientas utilizadas para que el sistema pueda ser mantenido a futuro por ellos mismos.

A modo de evaluación se concluye que se pudo cumplir con los objetivos planteados a pesar de que la implementación y el uso real de los sistemas para el soporte de decisiones no es una tarea sencilla y generalmente requiere enfrentar varios problemas de distinta índole. Uno de los principales en este proyecto fue la baja calidad de los datos de origen. A pesar de estos problemas se puede decir que es posible implementar una solución de Business Intelligence con herramientas Open Source en una institución pública y en el marco de un proyecto de grado.

Índice general

1. Introducción **9**
1.1. Contexto 9
1.2. Motivación 9
1.3. Objetivos 11
1.4. Resultados Esperados 11
1.5. Organización del documento 12
1.6. Público objetivo 13

2. Definiciones y características **15**
2.1. Introducción 15
2.2. Marco conceptual 15
2.2.1. Data Warehouse 16
2.2.2. OLAP y ROLAP 17
2.2.3. Cubos OLAP 18
2.2.4. Elementos que componen un cubo multidimensional 19
2.2.5. ETL 19
2.2.6. Business Intelligence 20
2.2.7. Dimensiones de Calidad de Datos 21
2.2.8. Técnicas para el desarrollo de un DW 22
2.2.9. Esquema Estrella y Copo de Nieve 24

3. Evaluación de herramientas de BI **27**
3.1. Primer acercamiento 27
3.2. Comparación de funcionalidades 31
3.2.1. Herramientas de soporte a ETL 32
3.2.2. Operaciones y componentes para análisis OLAP 33
3.2.3. Heramientas de Reporting 34
3.2.4. Tableros de mando 35
3.2.5. Seguridad 35
3.2.6. Documentación 36
3.2.7. Usabilidad 37

3.2.8. Aspectos generales . . . 37
3.3. Comparación final . . . 38
3.4. Prototipado . . . 39
3.5. Evaluación y selección final . . . 39

4. Diseño de la solución **41**
4.1. Descripción de la arquitectura . . . 41
4.2. Análisis de las fuentes de datos . . . 43
4.3. Calidad de datos . . . 46
4.3.1. Exactitud . . . 48
4.3.2. Completitud . . . 49
4.3.3. Frescura . . . 50
4.3.4. Consistencia . . . 50
4.3.5. Unicidad . . . 53
4.4. Modelado Multidimensional . . . 53
4.4.1. Esquema Multidimensional . . . 58
4.5. Diseño relacional de la base que soporta a los cubos . . . 60

5. Implementación del prototipo **63**
5.1. Componentes de la arquitectura de la suite de Pentaho . . . 63
5.2. Cubos y Análisis OLAP . . . 65
5.2.1. Estructura General . . . 66
5.2.2. Publicación . . . 68
5.3. Extracción, transformación y carga . . . 68
5.3.1. Introducción . . . 68
5.3.2. Tablas auxiliares para la carga de datos . . . 68
5.3.3. Mejora en calidad de datos . . . 69
5.3.4. Verificación de carga a realizar . . . 70
5.3.5. Diseño de procesos de carga y actualización . . . 72
5.4. Consultas Ad Hoc . . . 77
5.5. Reportes . . . 78
5.5.1. Herramientas para la elaboración de reportes . . . 78
5.5.2. Estructura de los reportes . . . 79
5.5.3. Visualización . . . 82
5.6. Cuadros de Mando . . . 82
5.6.1. Herramientas . . . 82
5.6.2. Estructura General . . . 83
5.6.3. Desarrollo(CDE) . . . 84
5.6.4. Creación de dashboards . . . 87
5.6.5. Publicación y Visualización . . . 89
5.7. Infografía . . . 90
5.7.1. Estructura general . . . 91
5.7.2. Creación y visualización de infografías . . . 91

5.8. Consola de Administración 93
5.8.1. Administración de Usuarios y Permisos 94
5.8.2. Administración de orígenes de datos 95
5.8.3. Herramientas administrativas 95
5.9. BI Server 96
5.10. Tiempos del proyecto 97

6. Conclusiones y Trabajo a futuro **101**
6.1. Resultados Obtenidos 101
6.2. Dificultades durante el proceso 101
6.3. Conclusiones Finales 102
6.4. Trabajo a Futuro 102

Anexo 1: Instalación en producción **105**
Descripción 105
Requisitos 105
Arquitectura 106
Instalación 108
Descarga de Archivos 108
Configuraciones iniciales 108
Configuraciones del esquema workbench 108
Instalación del Pentaho Data Integration 110
Publicación de esquema en Pentaho 112
Mantenimiento para efectuar cargas en forma manual 112
Como verificar una carga anteriormente realizada 112
Como realizar una carga 113
Como realizar un cambio de conexión 113
6.4.1. Configuración actual de la base de datos donde se encuentra la tabla de gestión de logs de carga 114
Problemas encontrados en las herramientas 114

Anexo 2: Complemento de la implementación de la solución **117**
Cubos y Analisis OLAP 117
Diseño de procesos de carga y actualización 120
Estructura general de los reportes 122

Anexo 3: Procesos de ETL para cubos de afiliados **127**
Introducción 127
Trabajo principal 127
Verificación de carga 128
Carga de la base de datos temporal 129
Carga de las dimensiones del Data Warehouse 131
Carga de la tabla de hechos del Data Warehouse 134

Anexo 4: Extensión del DW a partir del sistema AP-SGA **135**
Introducción . . . 135
Objetivos . . . 135
Relación entre requerimientos, dimensiones y medidas . . . 135
Diseño de la solución . . . 137
Naturaleza de los datos . . . 137
Modelado Multidimensional . . . 137
Diseño relacional de la base que soporta a los cubos . . . 143
Implementación de la solución . . . 145
Extracción, transformación y carga . . . 145
Consultas Ad Hoc . . . 154
Reportes . . . 155

Anexo 5: Tutorial de implementación de un DW **159**
Objetivos . . . 159
Construir un cubo con Schema Workbench . . . 159
Definir una conexión . . . 159
Crear un esquema . . . 160
Crear el cubo y asociar la tabla de hechos . . . 160
Agregar dimensiones . . . 161
Agregar jerarquías y niveles . . . 161
Agregar medidas . . . 161
Publicación . . . 162
Elaborar procesos de ETL's . . . 163
Análisis OLAP . . . 166
Pentaho Report Designer . . . 166

Glosario **167**

Bibliografía **169**

Capítulo 1

Introducción

En este capítulo se describe el alcance y los objetivos del proyecto de grado. En la sección 1.1 se presenta la definición del proyecto y su contexto. En la sección 1.2 se describe la motivación del problema. La sección 1.3 presenta los objetivos del proyecto. La sección 1.4 muestra los resultados esperados. Por último en la sección 1.5 se describe la organización de este informe.

1.1. Contexto

La Administración de los Servicios de Salud del Estado (de aquí en adelante ASSE) es una organización que busca mejorar el proceso de atención a la salud a los habitantes del país y que cuenta con una red de atención integral constituida por 66 Unidades Ejecutoras, de las cuales 51 se distribuyen en el interior del país, 18 son Centros Departamentales y 33 son Centros Auxiliares. De dicha institución dependen más de 228 policlínicas distribuidas en todo el territorio nacional [20].

Para la gestión de los datos de los afiliados en toda la red de atención a la cual brinda servicio, ASSE posee b hjumyvarios sistemas corporativos que facilitan dicha gestión. A partir de la información contenida en los mismos se desea obtener reportes gerenciales con la información que es manejada por estos sistemas. Actualmente en la institución no se dispone de un sistema de información gerencial del tipo Data Warehouse a nivel organizacional.

1.2. Motivación

Una de las principales motivaciones de este proyecto es brindar herramientas que den soporte a la toma de decisiones de negocio concretas, de forma sencilla y rápida. Los sistemas de información tradicionales que dan soporte a procesos transaccionales no almacenan la información en estructuras adecuadas para lograr estos objetivos en

forma eficiente.

La ausencia de información histórica es una de las limitaciones más notorias en los sistemas transaccionales dado que los datos almacenados en estos sistemas están diseñados para llevar la información de una institución al día pero no permiten contrastar claramente la situación actual con la de meses o años atrás.

Otro aspecto negativo de los sistemas transaccionales son los largos tiempos de respuesta, ya que las consultas de datos complejas usualmente implican uniones de tablas operacionales de gran tamaño, lo cual se convierte en incómodos retrasos que dificultan la fluidez del trabajo.

También se encuentra en estos sistemas una gran rigidez a la hora de extraer datos, de manera que el usuario muchas veces debe limitarse a los informes predefinidos que se configuraron en el momento de la implantación, y que no siempre responden a sus verdaderas necesidades, no pudiendo realizar reportes configurables en función de ciertos parámetros.

Los sistemas de tipo On-Line Analytical Processing (OLAP) surgen en la década de los 90 en contraposición a los sistemas On-Line Transaction Processing (OLTP) para satisfacer este tipo de necesidades. Estos dos términos se describen en detalle en la sección 2.2.2.

En el contexto particular del presente proyecto se observa que actualmente algunos de los reportes mencionados en la sección anterior son realizados manualmente por uno o más funcionarios experimentados en cada una de las áreas relevantes, lo cual es un trabajo sumamente tedioso ya que al ser una institución tan grande y con tantas unidades distribuidas se manejan grandes volúmenes de datos. Es por ésto que se pretende automatizar dichas tareas con el objetivo de optimizar recursos y a la vez poder tener encapsulada la información relevante de todos los sistemas participantes en uno solo que brinde un acceso uniforme y simplificado para la toma de decisiones.

Además de lo dicho anteriormente también se buscó mejorar la calidad de los datos manejados, evitando por ejemplo la información duplicada o inconsistente que se encuentra dentro de los distintos sistemas. La creciente necesidad de información para la toma de decisiones ha hecho que los sistemas decisionales cobren mayor importancia dentro de la institución, donde el objetivo principal sea la mejora en la calidad de atención a los afiliados.

1.3. Objetivos

El primer objetivo de este proyecto corresponde a la selección de una plataforma de Data Warehouse y Business Intelligence (DW/BI) con características Free/Open Source (FOSS) y que sea producto de un análisis profundo de prestaciones de las variadas soluciones disponibles actualmente en el mercado. Esta primer etapa implica un estudio del estado del arte tanto en soluciones de DW Open Source, como en bases de datos y herramientas de reporting. Estos conceptos se detallan en el capítulo de Definiciones y Características, en las secciones 2.2.1 y 2.2.6.

El segundo objetivo es implementar un prototipo de un sistema de DW corporativo utilizando la tecnología seleccionada en el punto anterior y que incluya la puesta en funcionamiento inicial, configuración y creación de los trabajos de Extracción Transformación y Carga (ETL, ver sección 2.2.5) requeridos para la carga de información necesaria. Además de implementar la actualización periódica del DW de forma automatizada, crear reportes de gestión por área de implicancia requeridos por ASSE con los datos extraídos, elaborar cuadros de mando e indicadores solicitados por la institución. Por último, es un objetivo del proyecto realizar el traspaso de conocimiento en las herramientas utilizadas al personal técnico designado para que el sistema pueda ser mantenido a futuro por ellos mismos.

1.4. Resultados Esperados

- Realizar un estudio del estado del arte en soluciones de DW Free/Open Source, tanto en bases de datos como en herramientas de reporting.
- Evaluar las prestaciones de cada una de las plataformas de DW Free/Open disponibles en el mercado.
- Se debe desarrollar un DW corporativo utilizando tecnología Free/Open Source junto con las herramientas recomendadas como software de ETL, BI, reportes y los que sean necesarios para contemplar una solución completa al problema planteado.
    - Se requiere el diseño de cubos que permitan cumplir los requerimientos de los reportes solicitados, así como también la implementación previa de los procesos de ETL necesarios para la carga de dichos cubos.
    - Se espera contar con un mecanismo de carga automática que se ejecute en forma periódica para obtener así los reportes actualizados para la toma de decisiones por parte de la institución.
    - Como último punto se requiere la transferencia de los conocimientos adquiridos sobre herramientas seleccionadas a la institución, con el fin de que puedan ser mantenidas por ellos mismos.

1.5. Organización del documento

El presente informe cuenta con seis capítulos diferenciados cuyo contenido se esboza a continuación.

En el capítulo 1 se presenta una introducción inicial al problema en el cual se detallan las características del problema a resolver, su contexto, motivación, objetivos del proyecto y los resultados esperados.

El capítulo 2 contiene las definiciones y características generales en donde se realiza una breve introducción a los conceptos necesarios para entender el trabajo realizado.

Luego, en el capítulo 3 se realiza una evaluación de herramientas de BI en donde se muestra el estudio del estado del arte desarrollado y se evalúan las distintas herramientas posibles para la resolución. Se comparan funcionalidades, se muestra el prototipo de dos plataformas seleccionadas. Se concluye el capítulo haciendo la evaluación y selección final de la plataforma a utilizar en el proyecto junto con la metodología utilizada en dicha selección.

En el capítulo 4 se detallan las decisiones tomadas para el diseño de la solución, se describe su arquitectura, naturaleza y análisis de los datos fuentes utilizados junto con un estudio de la calidad de los datos de los mismos. Se muestra el modelado multidimensional realizado y el diseño relacional de la base que soporta a los cubos.

El capítulo 5 presenta la implementación de la solución basada en el diseño antes mencionado. Se definen los cubos, análisis OLAP realizados, procesos de ETL implementados, consultas Ad Hoc, reportes, cuadros de mando e infografías.

Por último, en el capítulo 6 se evalúan los resultados alcanzados, las dificultades encontradas durante el proceso, conclusiones finales del proyecto y trabajo a futuro.

En forma adicional se entregan los siguientes anexos:

En el anexo 1 se describen los procesos de configuración e instalación del sistema puesto en producción.

El anexo 2 es un complemento del capítulo 5. En este anexo se describen los detalles de implementación que no están abarcados en el capítulo de Implementación por motivos de claridad en la lectura de este texto.

En el anexo 3 se presentan los procesos de ETL implementados para la carga de los cubos que se utilizan en el DW.

El anexo 4 detalla los indicadores implementados producto de nuevas solicitudes por parte de ASSE. Estos requerimientos implican nuevos reportes para la toma de decisiones por parte de los directivos. La información para estos requerimientos suplementarios es extraída a partir de los datos almacenados en la base de datos del SINADI - Asistencial (Sistema AP_SGA).

Por último el anexo 5 cubre el requerimiento correspondiente al traspaso de información, está dirigido al personal técnico y pretende ser una breve guía en la utilización de las herramientas comprendidas en la suite de la plataforma seleccionada.

Algunos términos específicos del contexto de trabajo de este proyecto han sido recopilados y definidos en un glosario luego de los anexos. Por último y luego del glosario se presenta la bibliografía utilizada en este proyecto.

1.6. Público objetivo

Este documento tiene por público objetivo personas relacionadas al área de Tecnología de la Información con conocimientos en sistemas operativos, base de datos y programación. También son aconsejables para el lector conocimientos relativos a calidad de datos, sistemas multifuentes, data warehousing y business intelligence para una mayor comprensión del documento. Estos últimos no son requisitos excluyentes dado que en el transcurso del documento son introducidos los conceptos básicos necesarios para la comprensión del mismo.

Capítulo 2

Definiciones y características

2.1. Introducción

En este capítulo se presentan las definiciones de los conceptos considerados relevantes sobre Data Warehouse (DW), On-Line Analytical Processing (OLAP), Extracción Transformación y Carga (ETL) y Business Intelligence (BI), además de otros relevantes al momento de analizar este proyecto.

2.2. Marco conceptual

Los sistemas de DW apuntan a la construcción y mantenimiento de estructuras destinadas al análisis de datos, transformando éstos en información y la información en conocimiento.

El concepto de DW, cuya traducción literal sería almacén o repositorio de datos, surge alrededor del año 1990 con la necesidad de recopilar información de datos acumulados durante años por los sistemas de gestión. Este concepto nace como producto de la evolución de los sistemas para dar soporte a la toma de decisiones.

Antes de la aparición de estos sistemas, la información que requerían las organizaciones se obtenía a partir de consultas y procesamientos sobre las bases de datos de los sistemas operacionales. A lo largo de los años las organizaciones han ido acumulando grandes cantidades de datos de todo tipo y con el tiempo estas consultas han sido insuficientes para resolver las necesidades analíticas.

Como ya se mencionó en el capítulo anterior, los sistemas de procesamiento transaccionales en línea (OLTP) usualmente no mantienen la información histórica requerida para la toma de decisiones en una organización. Las consultas gerenciales con información resumida y desde distintas vistas, demandan el procesamiento de importantes volúmenes de datos, requiriendo recursos y decrementando notablemente

el rendimiento de los sistemas operacionales.

Otro aspecto a tener en cuenta es la capacidad de las soluciones de BI es de lograr integrar datos desde distintas fuentes muy diversas.

En esta sección se denotan varios de los conceptos nombrados a lo largo del documento y que forman la base teórica necesaria para entender el dominio del problema que se abordó.

2.2.1. Data Warehouse

Un DW es una base de datos corporativa de apoyo a la toma de decisiones que se caracteriza por integrar datos crudos de una o más fuentes distintas, depurando y almacenando la información necesaria de forma organizada para luego procesarla, permitiendo su análisis desde múltiples perspectivas y con grandes velocidades de respuesta. Permite a los directivos que lo utilizan, tener una visión más completa e integral de los procesos dado que el resultado de su implementación es conocimiento acerca del funcionamiento de la organización [5].

Según la definición más tradicional del término DW, especificada por Bill Inmon [2] a principios de la década de los 90, los DW se caracterizan por ser:

- **Orientados al Sujeto:** Los datos almacenados brindan información sobre un sujeto o asunto en particular en lugar de concentrarse en la dinámica de las transacciones de la organización.
- **Integrado:** Los datos cargados en el DW pueden provenir de diferentes fuentes y son integrados para dar una visión global coherente.
- **Variables en el Tiempo:** El DW se carga con los distintos valores que toma una variable en el tiempo para permitir comparaciones, lo que implica que todos los datos deben estar asociados con un período de tiempo específico.
- **No volátiles:** Los datos son estables en el DW,se agregan y modifican datos, pero los datos existentes no son removidos.

La creación de un DW representa en la mayoría de las ocasiones uno de los primeros pasos, desde el punto de vista técnico, para implantar una solución completa y fiable de BI. Al no generar datos por sí mismos se dice que este tipo de sistemas son fuentes secundarias de información, alimentados desde fuentes de datos externas.

Las áreas o componentes básicos que conforman un sistema de DW son:

- **Sistemas de Datos Fuentes:** Donde se encuentra la información relevante que va a ser utilizada y cargada en el DW.

- **Área de almacenamiento temporal:** Aquí se guardan los datos limpios, combinados y estandarizados dentro de un sistema temporal de almacenamiento y sobre el cual se realizan procesamientos secuenciales de trabajos.
- **ETL:** Procesos que permiten obtener datos de distintas fuentes y depurarlos por medio de transformaciones para finalmente cargarlos en el DW.
- **Área de presentación de datos:** Es donde la información es organizada, almacenada y habilitada para la consulta directa de los usuarios finales.
- **Área de Herramientas de acceso a datos:** Corresponde a las herramientas de acceso a los datos del área de presentación y que permiten realizar análisis analíticos sobre los mismos.

Para cumplir con los requerimientos del proyecto, la plataforma elegida debe estar licenciada de tal manera que los usuarios pueden estudiar, modificar y mejorar su diseño mediante la disponibilidad de su código fuente, de ahí es que la solución aplicada es de tipo DW/FOSS.

2.2.2. OLAP y ROLAP

Los modelos de almacenamiento para procesamiento analítico online (OLAP) están diseñados y optimizados para guardar grandes volúmenes de datos. Esto se realiza de forma estructurada de manera de poder cumplir con la meta de funcionar en forma eficiente frente a los requerimientos del usuario por lo que están orientadas al procesamiento analítico. Para ello se utilizan estructuras multidimensionales denominadas Cubos OLAP.

Los sistemas OLAP requieren un procesamiento analítico que implica, generalmente, la lectura de grandes cantidades de datos para llegar a extraer algún tipo de información útil como ser tendencias de ventas, patrones de comportamiento de los consumidores, entre otros.

Los modelos para procesamiento transaccional online tradicionales (OLTP) a diferencia de los modelos OLAP, están orientados al procesamiento de transacciones, siendo éste típico de las bases de datos operacionales.

Los modelos multidimensionales pueden implementarse según diferentes estrategias. Las herramientas que implementan estas estructuras suelen utilizar algunas de las siguientes estrategias:

- **ROLAP:** Son los modelos en los cuales la organización física se implementa sobre tecnología relacional disponiendo de algunas facilidades para mejorar el rendimiento. Cuenta con todos los beneficios de una RDBMS Relacional a los cuales se les provee extensiones y herramientas para poder utilizarlo como un Sistema Gestor de DW.

- **MOLAP:** Son los modelos en los cuales la organización física de los datos se realiza en estructuras multidimensionales de manera que la representación externa y la interna coincidan. Disponen de estructuras de almacenamiento específicas y técnicas de compactación de datos que favorecen el rendimiento del DW.
- **HOLAP:** Son los modelos híbridos entre MOLAP y ROLAP, combinan estas dos implementaciones para almacenar algunos datos en un motor relacional y otros en una base de datos multidimensional.

2.2.3. Cubos OLAP

Los cubos o hipercubos OLAP son estructuras que representan los datos como una matriz en la cual sus ejes corresponden a los criterios de análisis y en los cruces se encuentran los valores a analizar. Estos cubos constan de dimensiones y medidas. Las dimensiones están relacionadas con los criterios de análisis de los datos, son variables independientes, representan los ejes del cubo y están organizadas en jerarquías. Las medidas son los valores o indicadores a analizar, se corresponden a datos asociados a relaciones entre los objetos del problema, son variables dependientes y se encuentran en la intersección de las dimensiones.

Existe la posibilidad de moverse dentro de las jerarquías de las dimensiones y observar de esta forma diferentes visiones de las medidas. Se puede seleccionar alguna de las dimensiones que se pretende analizar para realizar operaciones de agregación o desagregación, así como también fijar valores sobre algunas de estas dimensiones.

Para realizar estas acciones y otras se dispone de las siguientes operaciones sobre los modelos OLAP:

- **Slice:** Permite definir un subconjunto de dimensiones sobre las cuales interesa analizar las medidas.
- **Dice:** Se realiza un filtro estableciendo valores fijos para alguna de las dimensiones.
- **Pivot:** Permite seleccionar el orden de visualización de las medidas.
- **Drill-up y Drill-down:** Se realizan movimientos en la jerarquía de una dimensión agregando y desagregando respectivamente la misma. Estas operaciones pueden verse como ajustes en las escalas de los ejes.
- **Roll-up:** Calcula mediante una consolidación las medidas en función de agrupamientos. Estas medidas pueden ser resultado de operaciones como sumas o promedios.
- **Drill-across:** Es muy similar al funcionamiento de Drill-down, con la diferencia de que Drill-across no se realiza sobre jerarquías de una dimensión, sino que agrega como nuevo criterio de análisis una nueva dimensión.
- **Drill-through:** Con esta operación se accede desde un dato del cubo a datos en el DW de donde se deriva el cubo para acceder a datos descriptivos o seguir la traza de un dato.

Ver [3], [4], [5] y [6] por más detalles sobre estas operaciones.

2.2.4. Elementos que componen un cubo multidimensional

A continuación se presentan algunas definiciones de conceptos que corresponden a los componentes básicos de un cubo [22].

Medidas

Las medidas son atributos utilizados como unidades de medida sobre las entidades y que conforman los valores o indicadores a analizar. Son estandarizadas dentro de un mismo DW para que todas las fuentes de datos expresen sus valores de igual manera.

Dimensiones

Las dimensiones son los criterios principales de análisis de los datos de un DW y se desprenden de las entidades que forman parte del origen de datos del problema. Con las dimensiones se conforman los ejes de los cubos de información a ser utilizados en los sistemas OLAP. Pueden tener jerarquías internas para organizar los valores de datos que representan.

Hechos

Los hechos son criterios de análisis que contienen las medidas numéricas que serán utilizados por los analistas del negocio para apoyar el proceso de toma de decisiones. Contienen datos cuantitativos necesarios para el análisis.

Cubo Multidimensional

Los cubos multidimensionales son estructuras de datos a través de las cuales se representan los datos de un DW. Un cubo multidimensional representa o convierte los datos planos en una matriz de N dimensiones, está conformado por:

- Un conjunto de Dimensiones organizadas en jerarquías.
- Un conjunto de Medidas asociadas a cada coordenada.

La idea principal de estas estructuras es poder moverse en las jerarquías de las dimensiones y observar de esa forma diferentes visiones de las medidas.

2.2.5. ETL

El objetivo de los procesos de Extracción-Transformación-Carga (ETL) consiste en mantener cargado el DW con los datos correspondientes. La estructura general de estos

procesos consiste en operaciones de manipulación de datos que se realizan en un cierto orden comunicando entradas y salidas. El DW se carga inicialmente y luego se mantiene actualizado, normalmente involucra volúmenes de datos mucho mayores a los habituales en operaciones OLTP.

Los procesos ETL básicos son:

- **Extracción:** Ésta es la primera etapa y corresponde a la obtención de los datos que luego serán manipulados para ser cargados en el DW.
- **Transformación:** Una vez que la información es extraída hacia el área de datos temporales hay distintos pasos de transformación, como la limpieza de la información o selección de los campos necesarios para la carga del DW, también se pueden combinar distintas fuentes de datos y realizar otras operaciones.
- **Carga:** Al final del proceso de transformación, los datos están en forma para ser cargados dentro del DW. En ésta y en las anteriores etapas se pueden generar distintos tipos de logs.

Para la carga de los datos en el DW se suele emplear una herramienta que permita modelar visualmente los procesos de trabajo de ETL. En la extracción, las fuentes de datos pueden ser muy diversas, desde información dentro de páginas Web, archivos de texto, hojas de cálculo, hasta bases de datos relacionales. Para la transformación, se dispone de funciones para efectuar cálculos, análisis sintácticos de cadenas, definir formatos y enriquecimiento con información obtenida a partir de búsquedas externas. El proceso de carga ingresa en el DW los datos previamente transformados.

2.2.6. Business Intelligence

Una plataforma de BI es un conjunto multifuncional de metodologías, procesos y tecnologías. Brinda la posibilidad de generar estructuras de datos para realizar análisis de la información a partir de un conjunto de datos almacenados en uno o varios sistemas. Las plataformas de BI incluyen integración de datos, calidad de datos, data warehousing, gestión de datos maestros, textos, análisis de contenido y muchos otros. Por lo tanto, los DW son parte de la arquitectura de un sistema de BI por lo que están estrechamente vinculados y se complementan entre sí, el primero para la preparación y almacenamiento de los datos y el segundo para el análisis de estos mismos.

En un sistema de BI se suelen encontrar los siguientes componentes o funcionalidades:

- **Análisis OLAP:** Es una herramienta de visualización multidimensional que permite analizar los datos provenientes de un DW dando la posibilidad de realizar las operaciones antes mencionadas.
- **Reportes:** Generación de información con alto nivel de detalle orientado a un nivel gerencial que es utilizado para la toma de decisiones.

- **Dashboard:** Es una herramienta de visualización de datos que muestra el estado actual de las métricas e indicadores claves del rendimiento de una empresa o área de negocio.
- **Data Mining:** Se enfoca en intentar descubrir mediante el reconocimiento de patrones de datos información oculta que reside de manera implícita en los datos.

2.2.7. Dimensiones de Calidad de Datos

Las dimensiones de calidad que se introducen en esta sección son las que serán utilizadas en este proyecto. Estas dimensiones son Exactitud, Completitud, Frescura, Unicidad y Consistencia. [21] Dentro de las Dimensiones estudiadas, se abordarán algunos factores de calidad particulares a las mismas, los cuales se detallan en el siguiente listado.

Dimensión	Factor
Exactitud	Correctitud semántica Correctitud sintáctica Precisión
Completitud	Cobertura Densidad
Frescura	Frescura
Consistencia	Integridad de dominio Integridad intra-relación Integridad referencial
Unicidad	Duplicación Contradicción

Cuadro 2.1: Tabla de dimensiones analizadas junto con sus factores.

Exactitud

La exactitud indica que tan precisos, válidos y libres de errores están los datos utilizados. Dentro de la dimensión Exactitud existen varios factores, los que se abordaron son la *Correctitud semántica*, *Correctitud sintáctica* y *Precisión*

Completitud

La completitud describe el porcentaje en el que la información ingresada a las fuentes representa al mundo real. En esta dimensión, están incluidos los factores *Cobertura* y *Densidad*

Frescura

Con respecto a esta dimensión de calidad lo que se busca es determinar qué tan vigentes u obsoletos son los datos que estamos manejando dentro de nuestro DW, tomando en cuenta el tiempo transcurrido desde la ultima extracción de las fuentes.

Consistencia

Captura la satisfacción de reglas semánticas definidas sobre los datos. Como por ejemplo si los datos satisfacen las reglas de dominio, si las dependencias funcionales y referenciales se satisfacen, o si hay contradicciones entre los datos. Dentro de esta dimensión se destacan los factores *Integridad de dominio*, *Integridad Intra-Relación* e *Integridad referencial*.

Unicidad

La unicidad indica el nivel de duplicación entre los datos. Si los mismos están repetidos, o si hay datos contradictorios que representan la misma entidad. Esta dimensión abarca los factores *Duplicación* y *Contradicción*.

2.2.8. Técnicas para el desarrollo de un DW

Para llegar a la implementación de los reportes y análisis OLAP se debe pasar previamente por una etapa de diseño e implementación de los artefactos que serán utilizados en el DW desde los cuales los reportes a generar obtendrán la información necesaria para poder ser visualizada. La figura 2.1 muestra el proceso de diseño, desarrollo y carga del DW. Se pueden diferenciar varios pasos que conforman el proceso y se detallan a continuación.

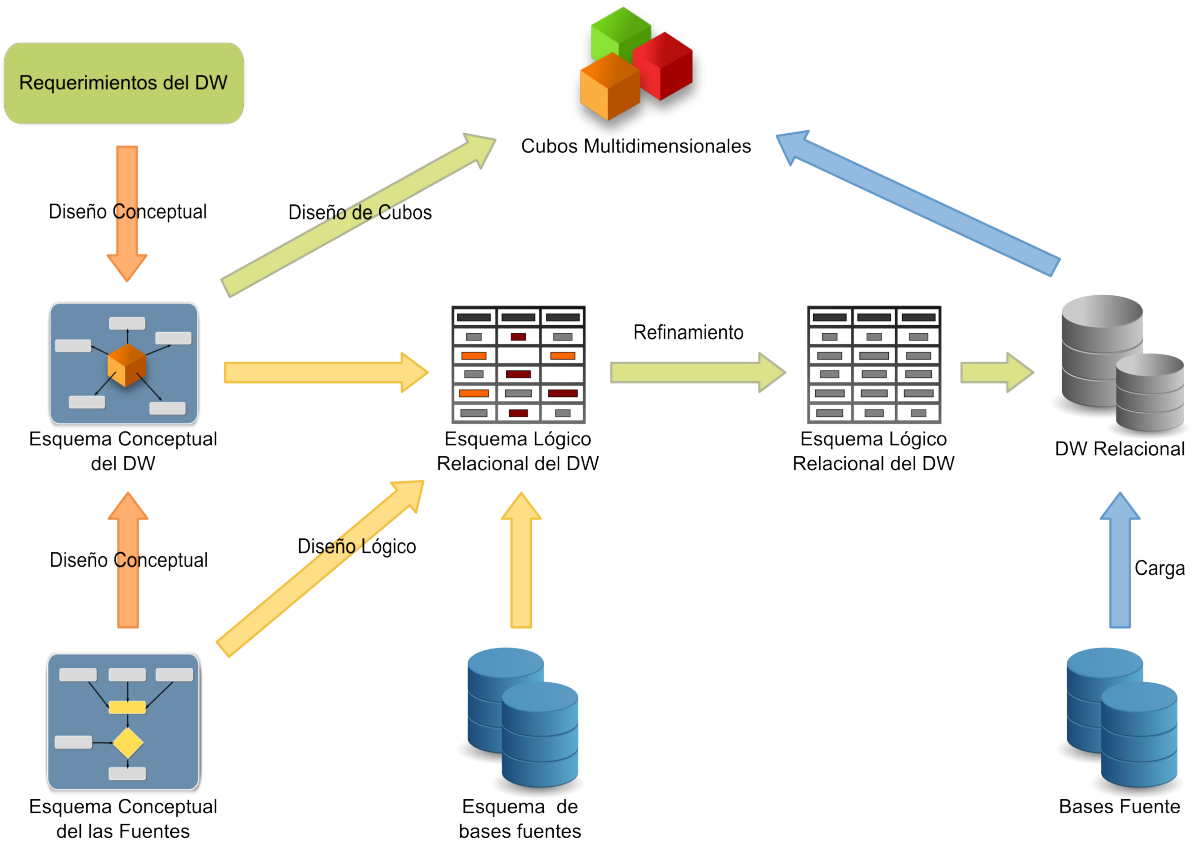


Figura 2.1: Diagrama del proceso de diseño del DW

El primer paso del proceso es el denominado *Diseño Conceptual Multidimensional* del sistema. A partir de los requerimientos funcionales iniciales, las estructuras de las bases de datos fuentes y las características de los datos se genera el esquema conceptual de las estructuras multidimensionales que son requeridas para cubrir los requerimientos del DW.

El segundo paso del proceso es llamado *Diseño Lógico Multidimensional* y es el pasaje del esquema conceptual multidimensional creado en el paso anterior al modelo lógico de las estructuras multidimensionales. Estas estructuras son generadas a partir de los requerimientos no funcionales del modelo conceptual, así como también, de la aplicación de los lineamientos de diseño multidimensional en este tipo de sistemas.

El tercer paso del proceso corresponde al *Refinamiento y Creación* y es la etapa en la cual, teniendo en cuenta las estructuras multidimensionales creadas en el paso anterior, las operaciones criticas y también los volúmenes de datos que se manejan, se depuran lo más posible las estructuras generadas anteriormente. En este paso se crean también lo cubos multidimensionales a partir de los objetos resultantes del diseño y de la

depuración.

La última etapa del proceso corresponde a la *Carga de datos* en la cual se cargan las tablas relacionales con la información que proviene de las bases de datos fuentes y que es previamente refinada y mejorada en términos de calidad.

Como se puede apreciar, el proceso de desarrollo del modelo dimensional es un proceso iterativo, en donde los modelos creados pueden ir siendo mejorados a medida que se va conociendo más sobre los requerimientos y los datos. A medida que se avanza en el diseño, se pueden descubrir nuevas dimensiones y medidas para los indicadores.

Otro punto importante a destacar en este proceso de diseño es que, la mayor comprensión del negocio permite definir de mejor manera las dimensiones y medidas del modelo dimensional. Los requerimientos de los usuarios como la realidad de los datos fuentes influyen directamente al momento de tomar decisiones durante los cuatro pasos descriptos anteriormente.

En [4] se presentan y detallan procesos de análisis típicos para la creación de DW y donde también se ejemplifican cuestiones de diseño de los mismos.

2.2.9. Esquema Estrella y Copo de Nieve

Al modelar las dimensiones, la implementación física de las mismas es un punto muy importante. Las dimensiones pueden contar con múltiples atributos y con jerarquías de varios niveles entre los mismos, lo que hace que se deba definir si se normalizan o no las tablas de dimensión diseñadas.

En algunos casos, la dimensión se puede representar en una única tabla donde reside toda la información al nivel más bajo, o también mediante un conjunto de tablas relacionadas que respeten la tercera forma normal. El primero de los casos corresponde a un esquema estrella (que consta de una tabla de hechos central y un conjunto de tablas de dimensión relacionadas al hecho). Al normalizar las dimensiones se dice que se transforma al modelo estrella en un copo de nieve, debido al gráfico que se desprende de su estructura como se denota en la figura 2.2.

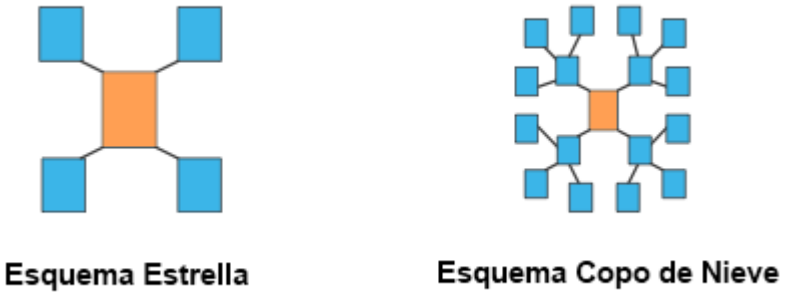


Figura 2.2: Esquemas estrella y copo de nieve

Las definiciones anteriormente presentadas nos dan una breve introducción al dominio de este proyecto. Para no extendernos del alcance de este capítulo, al final de este documento hacemos referencia a bibliografías que profundizan los temas introducidos anteriormente.

Capítulo 3

Evaluación de herramientas de BI

El primer objetivo de este proyecto consistió en realizar un relevamiento y una evaluación de las diferentes soluciones de Data Warehouse (DW) y Business Intelligence (BI) Free Open Source Software (FOSS) existentes en el mercado.

Los pasos seguidos para la elección de la herramienta a utilizar fueron los siguientes:

- Investigación de posibles herramientas a utilizar que cubran los requerimientos del proyecto.
- Preselección de un subgrupo de las herramientas investigadas sobre las cuales se realiza una evaluación y comparación de las características funcionales generales y características de arquitectura (ver [1]) midiendo con métricas creadas en cada uno de los puntos. Estos pasos se detallan en las secciones 3.1 y 3.2 respectivamente.
- Elección de dos herramientas del paso anterior sobre las cuales se realizan prototipos en ambientes disjuntos. Estos pasos se detallan en las secciones 3.3 y 3.4 respectivamente.
- Comparación de características y capacidades de los prototipos anteriores y selección final para la implantación. Este paso es detallado en la sección 3.5

Para la evaluación de las herramientas se utilizó un conjunto de métricas creadas para este fin que permiten clasificar las plataformas evaluadas y poder seleccionar una de ellas. En las siguientes secciones se detalla el proceso de selección mencionado.

3.1. Primer acercamiento

Se realizó un relevamiento inicial de herramientas existentes, considerando solo aquellas que proveen licenciamiento libre y de código abierto (Free/Open Source). Cuando nos referimos a licenciamiento libre estamos hablando de la libertad que provee el software utilizado para poder ser copiado, utilizado y/o modificado en caso de ser necesario. A continuación se presentan cada una de estas herramientas.

Pentaho

La plataforma Open Source Pentaho Business Intelligence[1] está basada en tecnología Java y con un ambiente de implementación también basado en Java lo que la hace una herramienta flexible y adaptable a varios ambientes. La plataforma posee módulos de reportes, análisis olap, cuadros de mando (Dashboards), extracción de datos (Data Mining), integración de datos (ETL), administración y seguridad. Posee una interfaz de usuario bastante amigable.

Características Generales:

- **Versión Evaluada:** Pentaho BI Suite Community Edition - 3.5.2 Estable , Junio 2010
- **Licenciamiento:** GPL2, LGPL, MPL (Mozilla Public Licence)
- **Versión Comercial:** Pentaho BI Suite Enterprise Edición (Mayor cantidad de funcionalidades)
- **Componentes Principales:** ETL, Job Designer, Conectores, Repositorio Visual, Análisis OLAP, Metadata, Data Mining, Reporting, Dashboards, BI Platform, Administration Server.

JasperSoft

La plataforma JasperSoft[2] es un conjunto de herramientas que componen un sistema de BI en el cual su característica predominante es ser unificador de datos de distintos orígenes, con capacidades de análisis de dichos datos de forma interactiva. Basado en tecnología Java, está formada por herramientas para generar informes, integración y análisis de datos, dashboards y herramientas para administración de la solución. Posee una interfaz amigable al usuario.

Características Generales:

- **Versión Evaluada:** JasperSoft BI Suite Community - 3.7.0 Estable , Junio 2010
- **Licenciamiento:** GPLv2
- **Versión Comercial:** JasperSoft BI Suite Express Edition, Professional Edition y Enterprise Edition (Mayor cantidad de funcionalidades)

[1] *Pentaho Community website - http://sourceforge.net/projects/pentaho/files/Data%20Integration/4.1.0-stable/-Ultimoacceso03/2011*

[2] *Jaspersoft project website - http://jasperforge.org/projects/jaspersoft - Ultimo acceso 06/2011*

- **Componentes Principales:** ETL, Job Designer, Conectores, Repositorio Visual, Análisis OLAP, Reporting, Dashboards, BI Platform, Administration Server.Tecnología: J2EE, iReport, Liferay.

SpagoBI

La Plataforma SpagoBI[3] es una plataforma de integración ya que se construye en torno a un conjunto de herramientas pre existentes. Provee varias funcionalidades tanto en términos de análisis y de gestión de datos como también de administración y seguridad. Ofrece soluciones para generación de informes, análisis OLAP, minería de datos, tableros de mando, consultas ad-hoc, KPI(Key Performance Indicators), integración de datos, así como también gestión para el control de versiones y la aprobación de flujos de trabajo de los documentos generados. Permite el uso de varios motores de análisis de forma concurrente y a su vez posee consolas para monitorizar procesos en tiempo real. Es una solución completa en términos de funcionalidades básicas y totalmente Open Source dado que no posee versiones comerciales.

Características Generales:

- **Versión Evaluada:** SpagoBI Studio - 2.6.0, Junio 2010
- **Licenciamiento:** LGPL (GNU Lesser General Public License)
- **Versión Comercial:** No existe, solo se cobra por Soporte a Usuarios, Proyectos y Mantenimientos.
- **Componentes Principales:** ETL, Reporting y Ad-Hoc Reporting, Análisis OLAP, BI Platform, Administration , Charting, Dashboard, Cockpits interactivos, GEO/GIS, Data Mining, Query By example, Smart Filters, Accesible reporting, Consola de monitoreo en tiempo real, Repositorio Visual, SDK integrado, Dossier Analítico .

OpenI

OpenI[4] es una herramienta de informes OLAP basada en tecnología JEE Web, es una solución de rápida instalación para la construcción y publicación de informes de fuentes de datos OLAP XMLA compatibles como Microsoft Analysis Services o Mondrian. Posee módulos para modelado dimensional, herramientas para diseño de cubos OLAP, modelos

[3] *SpagoBI - http://www.spagoworld.org/xwiki/bin/view/SpagoBI/ - Ultimo acceso 05/2011*
[4] *OpenI Project - http://openi.org/ - Ultimo acceso 06/2011*

estadísticos predictivos, herramientas para generación de reportes interactivos y creación de Dashboards. Posee un entorno de administración simple y directo.

Características Generales:

- **Versión Evaluada:** OpenI Suite - 2.0 RC2 , Julio 2010
- **Licenciamiento:** GPLv2 (GNU General Public License versión 2)
- **Versión Comercial:** No
- **Componentes Principales:** Análisis OLAP, Reporting y Dashboards para servidores ROLAP.

Palo

El paquete Palo BI[5] es una solución Open Source para la gestión del rendimiento, planificación, análisis, informes y ETL de datos e información. El paquete incluye un servidor OLAP, hojas de cálculo en línea basadas en Ajax y una herramienta Web de ETL. Bastante integrable con sistemas propietarios como SAP y Microsoft Office. Es una solución más inmadura en términos de soluciones en comparación a las antes mencionadas.

Características Generales:

- **Versión Evaluada:** Palo Suite, Mayo 2010
- **Licenciamiento:** GPLv2 (GNU General Public License versión 2)
- **Versión Comercial:** Palo Suite Premium Edition (garantía extendida del software y funcionalidades de soporte).
- **Componentes Principales:** Multidimensional OLAP Server (MOLAP) , Web server, Palo ETL Server y Palo para integración con Excel, Palo Modeler, Palo Report Manager.

---

[5] *Palo Project - http://www.jedox.com/es/producto/palo-suite.html - Ultimo acceso 06/2011*

3.2. Comparación de funcionalidades

Dada la cantidad de plataformas Open Source de Business Intelligence que se encuentran en el mercado es necesario poder diferenciar cada uno de sus componentes y sus características para tomar la opción que más se ajusta a los requerimientos de este proyecto. Las herramientas y características de las plataformas que se evaluaron fueron las siguientes:

- Herramientas de soporte a ETL
- Operaciones y componentes para análisis OLAP
- Heramientas de Reporting
- Tableros de Mando
- Seguridad
- Documentación brindada
- Usabilidad y amigabilidad con el usuario
- Aspectos generales

Para la comparación y evaluación de las plataformas se definieron ciertas medidas que permiten evaluar los distintos productos seleccionados y acercarse a la decisión más apropiada.

Se evaluaron y compararon cada uno de los componentes por separado tomando el siguiente rango de evaluación según sus funcionalidades:

Nivel A: El componente existe y posee una cantidad de funcionalidades superior a la media.

Nivel B: El componente existe pero posee una cantidad media de funcionalidades.

Nivel C: El componente existe pero posee escasa o pobre cantidad de funcionalidades con respecto a la media.

Nivel D: El componente no se encuentra como una característica de la plataforma pero puede ser reemplazado por otro o implementado fácilmente.

Nivel E: El componente no se encuentra como una característica en la plataforma y se desconoce si puede ser reemplazado por otro.

A partir de estos 5 niveles se genera una escala de valores numéricos, los cuales van desde A=5 (máximo) hasta E=1 (mínimo).
Es importante resaltar que al no disponer de instalaciones locales de todas las herramientas evaluadas, algunas de las calificaciones fueron realizadas en base a material encontrado en la web como revisiones de especialistas, videos explicativos y tutoriales.

3.2.1. Herramientas de soporte a ETL

En esta sección comparamos las diferentes herramientas de ETL provistas por las plataformas. Se evaluaron las capacidades de extracción de datos desde los distintos tipos de orígenes, su transformación, su transporte y la carga de los mismos hacia el DW. Para realizar las comparaciones de las herramientas se realizaron instalaciones locales de todas las herramientas provistas. Además, se analizaron video-tutoriales de las mismas para diferencias las distintas capacidades en cada una de ellas. La siguiente tabla muestra la comparación entre cada una de las plataformas y la evaluación del módulo.

Plataforma	**Herramienta**	**Evaluación**
Pentaho	Pentaho Data Integration (Kettle)	**B**
JasperSoft	Jasper ETL (Basado en TOS)	**A**
SpagoBI	TOS (Talend Open Studio)	**A**
OpenI	N/A	**D**
Palo	Palo ETL Server	**C**

Cuadro 3.1: Tabla de comparación de las capacidades de ETL de las distintas herramientas

Pentaho Data Integration (PDI) es provista por la plataforma Pentaho. La misma esta desarrollada con tecnología Java y utiliza Javascript como código de desarrollo de transformaciones. Tiene una interfaz de usuario muy simple y bastante intuitiva, permitiendo que con algunos pocos pasos se puedan crear procesos simples de ETL. Provee de una cantidad de componentes limitados pero bastante potentes, siendo éstos suficientes para realizar procesos complejos de ETL y de integración de datos. Posee una herramienta de debugging muy simple y se pueden activar los logs para analizar las ejecuciones de las transformaciones. Permite ejecuciones concurrentes de acciones lo cual puede mejorar los tiempos de ejecución.

De las herramientas analizadas, las dos más destacadas son Talend Open Studio (TOS) y Jasper ETL (por estar basada en TOS) y a pesar de que prácticamente brindan las mismas funcionalidades que PDI, éstas poseen algunas diferencias y mejoras en ciertas características. La mayor diferencia con respecto a PDI es que TOS es un generador de código, lo que hace que, cuanto más conocimiento se tenga de la tecnología generada mejor es el aprovechamiento y potencialidad de la herramienta. Al estar generando código se pueden utilizar componentes ya existentes o se puede integrar con otra herramienta de forma directa y también se puede realizar el debug directo sobre el código, lo cual es una diferencia importante con respecto a la herramienta de Pentaho. TOS está basado en Eclipse y posee una interfaz fácil de entender y completamente integrada que permite una visualización global de cada elemento utilizado, así como también del código generado. Otra diferencia notoria es que TOS posee ayuda

contextual bastante completa dentro de la herramienta. También cabe destacar que como funcionalidad extra posee una herramienta sencilla para modelado gráfico de procesos.

Ambas herramientas tienen liberaciones nuevas cada poco tiempo, pero TOS muestra una evolución bastante más constante y rápida con respecto a PDI.

Por fuera de la comparación inicial y como resultado de la experiencia obtenida en el proyecto utilizando PDI podemos agregar que, uno de los puntos negativos de la herramienta es la complejidad implicada en el pasaje de parámetros y variables entre los distintos trabajos y transformaciones de carga creados con la misma, donde para realizar la correcta parametrización se requiere de un tiempo considerable en comparación al diseño lógico de cada proceso. El tiempo requerido aumenta al momento de realizar modificaciones sobre los mismos. Otro problema que ocurrió durante la ejecución de los procesos de ETL desde la herramienta fue que se produjeron cortes de ejecución en algunos trabajos por consumo excesivo de memoria por parte de la aplicación, lo cual hizo que los procesos no se finalizaran con normalidad y obligándonos a realizar las ejecuciones de los trabajos desde la consola.

3.2.2. Operaciones y componentes para análisis OLAP

En este punto se evalúan las capacidades de análisis OLAP ofrecidos por cada plataforma. En particular las mismas deben contar con un motor o servidor de cubos y un componente visual que permita realizar las operaciones OLAP.

Estos componentes en conjunto proveen la capacidad de consultar grandes cantidades de datos en el DW utilizando estructuras multidimensionales (ó *Cubos OLAP*) y a su vez permiten interactuar visualmente con esta información.

En la tabla 3.2 se enumeran los diferentes motores OLAP de las plataformas y sus componentes de visualización de tablas dinámicas (ó *Pivot Tables*).

La evaluación de dichos componentes es realizada tomando en cuenta varias características, algunas de ellas son las operaciones OLAP disponibles al realizar los análisis, la cantidad de componentes visuales provistos, las posibilidades de filtrados en tiempo real en los análisis, la cantidad de combinaciones posibles entre motores de tablas dinámicas y los motores de cubos, así como también el grado de madurez de cada uno de los mencionados.

Para realizar esta evaluación se analizó en profundidad la documentación de las plataformas involucradas, se investigaron comparaciones en foros y se analizaron video-tutoriales de las mismas para la comparación de usabilidad.

Plataforma	Tablas Pivot/Motor	Evaluación
Pentaho	JPivot/Mondrian	**B**
JasperSoft	JPivot/Mondrian	**B**
SpagoBI	JPivot/Mondrian - JPalo/Mondrian - JPivot/XMLA Server	**A**
OpenI	JPivot/Mondrian - JPivot/XMLA Server	**A**
Palo	Palo (MOLAP)	**B**

Cuadro 3.2: Tabla de comparación de las capacidades de procesamiento analítico en línea

La única plataforma que utiliza su propio servidor de cubos y su propio visualizador integrado es Palo, las demás utilizan por defecto el motor Mondrian, aunque SpagoBI y OpenI pueden sustituirlo por otro (por ej: XMLA Server)

Lo mismo ocurre con los componentes visuales, todas las plataformas utilizan por defecto JPivot, salvo en el caso de SpagoBI y OpenI que pueden utilizar componentes visuales como JPalo.

Las plataformas mejor evaluadas se caracterizaron por tener la capacidad de poder intercambiar los componentes utilizados y esto fue evaluado como una capacidad extra importante dado que pueden adaptarse a múltiples motores y visualizadores en el caso de ser necesario.

3.2.3. Heramientas de Reporting

En esta sección se comparan las capacidades de las plataformas de creación y visualización de reportes, las cuales se muestran en las siguientes dos tablas. Algunas herramientas fueron descargadas, instaladas localmente y comparadas pero sin utilizar datos, simplemente como una forma de prueba de evaluación. La mayor comparación entre las mismas fue realizada según la especificación de la documentación de cada proveedor y de los videos tutoriales analizados en la web.

Plataforma	Creación de Reportes	Evaluación
Pentaho	Pentaho Report Designer,JasperReport, BIRT	**A**
JasperSoft	JasperReport	**B**
SpagoBI	JasperReport, BIRT	**B**
OpenI	N/A	**D**
Palo	Palo Report Manager	**C**

Cuadro 3.3: Tabla de comparación de las capacidades de creación de reportes

Plataforma	Visualización de Reportes	Evaluación
Pentaho	Web, email, web services, PDF, HTML, XLS, RTF, texto plano	**A**
JasperSoft	Web, email, PDF, HTML, XLS, CSV, RTF, XML, texto plano	**B**
SpagoBI	Web, email, PDF, HTML, CSV, RTF, XML, texto plano	**B**
OpenI	Web, PDF, HTML, XLS, RTF, texto plano	**C**
Palo	Web, PDF, HTML, CSV, RTF, XML, texto plano	**C**

Cuadro 3.4: Tabla de comparación de las capacidades de visualización de reportes

3.2.4. Tableros de mando

En esta sección evaluamos las capacidades de creación y visualización de los tableros de mando dentro de cada plataforma. En términos de herramientas de creación se evaluó la usabilidad de las mismas y en términos de resultados se evaluó la amigabilidad, la capacidad interactiva y la funcionalidad de los componentes obtenibles. Al igual que en las comparaciones anteriores, fueron muy útiles los videos tutoriales de la web y los foros de discusión que proveyeron de posibles capacidades de las herramientas evaluadas.

Plataforma	Creación/Visualización de Dashboards	Evaluación
Pentaho	CDF, JfreeChart, CCC Charts, Google Maps	**A**
JasperSoft	Dashboard Designer	**A**
SpagoBI	OpenLazlo	**A**
OpenI	OpenI	**B**
Palo	Palo	**B**

Cuadro 3.5: Tabla de comparación de las capacidades de creación y visualización de los tableros de mando

Todas las herramientas analizadas tuvieron una evaluación positiva, las mejor puntuadas se destacan en la cantidad de componentes que se pueden utilizar y las capacidades de integración con componentes gráficos externos.

3.2.5. Seguridad

En esta sección evaluamos las capacidades en temas de seguridad que proveen las plataformas estudiadas. Evaluamos la seguridad en términos de funcionalidades provistas para autenticación y perfiles de usuarios, acceso a documentos y carpetas, interfaces de autenticación con sistemas externos y seguridad en la transmisión de los

datos. La forma de evaluar cada uno de los puntos fue únicamente a través del análisis de la documentación de las plataformas y por comentarios en los foros de desarrolladores.

Plataforma	**Seguridad**	**Evaluación**
Pentaho	ACEGI,Por Usuario/Roles/Grupos por restricción de acceso a elementos.	**B**
JasperSoft	Single Sign-On, autorización externa via LDAP/MSAD, Por Usuario/Roles/Grupos sobre las filas y celdas de OLAP, Por Usuario/Roles/Grupos por restricción de acceso a elementos.	**A**
SpagoBI	Single Sign-On, autorización externa vía LDAP/MSAD, Por Usuario/Roles/Grupos por restricción de acceso a elementos.	**A**
OpenI	Single Sign-On, Por Usuario/Roles/Grupos por restricción de acceso a elementos.	**B**
Palo	Basado en la estructura de seguridad de J2EE y posee políticas de acceso a usuarios o grupos, permitiendo políticas jerárquicas de acceso a los datos.	**C**

Cuadro 3.6: Tabla de comparación de las capacidades en seguridad que brindan las distintas plataformas

3.2.6. Documentación

En esta sección evaluamos la cantidad de información que es provista por cada uno de los distribuidores sobre las plataformas, así como también las referencias externas obtenidas como por ejemplo foros, bibliografías, tutoriales, blogs, wikis, entre otros. Este punto fue uno de los más importantes dado que son las herramientas que van a utilizar los administradores futuros para el mantenimiento y gestión de la plataforma luego de terminado el proyecto.

Plataforma	Documentación	Evaluación
Pentaho	Documentación completa, Website, Foros, Papers, wiki, blogs externos, Bibliografía, tutoriales.	**A**
JasperSoft	Documentación, Website, blogs externos	**B**
SpagoBI	Documentación completa, Website, Foros, Papers, wiki, blogs externos, tutoriales	**A**
OpenI	Documentación parcial, Mail Advisors, blogs externos	**C**
Palo	Documentación escasa	**C**

Cuadro 3.7: Tabla de comparación de la cantidad de documentación provista en cada una de las distribuciones

3.2.7. Usabilidad

En esta sección evaluamos la usabilidad de los componentes de las plataformas. Lo que se busca es medir la facilidad o complejidad con que las personas pueden utilizar cada una de las herramientas de las plataformas así como también la amigabilidad que brindan las herramientas para el usuario. Para esto se separaron en tres puntos de vista dependiendo del tipo de acceso sobre la herramienta, los cuales son: el punto de vista del usuario, el de los administradores y el de los desarrolladores. Para este punto en particular fue necesario recurrir a las revisiones de especialistas, videos y/o tutoriales encontrados en la web, así como también comentarios de usuarios registrados en foros.

Plataforma	Usuarios	Administradores	Desarrolladores	Promedio
Pentaho	A	A	B	**A**
JasperSoft	B	B	B	**B**
SpagoBI	B	A	B	**B**
OpenI	A	B	B	**B**
Palo	B	B	B	**B**

Cuadro 3.8: Tabla de comparación de la usabilidad desde el punto de vista de los Administradores

3.2.8. Aspectos generales

La *Madurez* de una plataforma es una clasificación subjetiva creada en esta etapa de evaluación y lo que busca es clasificar una plataforma según sus capacidades en general, además de la calidad y cantidad de herramientas provistas por ésta en comparación con las demás. En esta sección se evalúan los aspectos no técnicos de las plataformas como por ejemplo la filosofía, sus tipos de licenciamiento, la disponibilidad de las ediciones

empresariales pagas, la madurez que ha alcanzado, así como también otros aspectos que influyen en la elección de una plataforma como son la cantidad de funcionalidades que provee, la capacidad de planificación de tareas automatizadas, capacidad de integración con otros sistemas, entre otros. La siguiente tabla muestra la comparación entre los aspectos mencionados y están agrupados en no técnicos y otros.

Plataforma	**No Técnicos**	**Otros**	**Promedio**
Pentaho	A	A	**A**
JasperSoft	A	A	**A**
SpagoBI	A	A	**A**
OpenI	B	B	**B**
Palo	B	B	**B**

Cuadro 3.9: Tabla de comparación de aspectos no técnicos y otros aspectos importantes de las plataformas.

3.3. Comparación final

La siguiente tabla muestra los resultados finales de cada uno de los puntos marcados anteriormente además del promedio final de cada una de las plataformas.

Plataforma	**Evaluación**
Pentaho	**A**
JasperSoft	**B**
SpagoBI	**A**
OpenI	**C**
Palo	**B**

Cuadro 3.10: Tabla de comparación final de las plataformas.

Dados los resultados anteriores sobre cada uno de los puntos tomados en cuenta para evaluar las plataformas, se tomó la decisión de probar dos de las distribuciones estudiadas, las cuales son Pentaho y SpagoBI. Se seleccionaron estas plataformas en lugar de JasperSoft ya que se encontró que este último contaba con menos documentación que el resto. Estas distribuciones seleccionadas son las que dieron las mejores evaluaciones generales en cada uno de los puntos pero además de que son las que poseen la mayor cantidad de funcionalidades, también poseen grandes diferencias en sus arquitecturas dado que Pentaho posee la mayoría de sus componentes de forma integrada y SpagoBI mantiene sus componentes distribuidos, una es totalmente FOSS (SpagoBI) y la otra lo es parcialmente (Pentaho). Además de estas diferencias, notamos que las dos distribuciones elegidas al momento de tomar la decisión eran las dos más versátiles de las estudiadas dado que eran las más integrables y podían

intercambiar herramientas de otras distribuciones en algunos casos, por ejemplo, con las herramientas de ETL que permitían obtener información de fuentes como Hadoop y otras NoSQL o como las herramientas de reporting.

3.4. Prototipado

Para poder tener una evaluación más real de las plataformas preseleccionadas decidimos realizar dos prototipos sobre cada una de las distribuciones seleccionadas, una con la de Pentaho y otra con la de SpagoBI, de modo de poder compararlas entre sí y poder elegir la mejor opción para aplicar en este proyecto. Para evaluar la configuración e instalación de las dos plataformas preseleccionadas se virtualizaron dos máquinas con idénticas características, y en donde en cada una se realizó el proceso de instalación. Las máquinas fueron instaladas con Ubuntu Linux 10.04 como sistema operativo base.

En una de las maquinas se instalaron los paquetes de herramientas de la distribución de Pentaho y en la otra las herramientas de SpagoBI hasta tener un pequeño sistema de DW en cada una de ellas.

Se aprovechó que Pentaho provee una pequeña base de datos de prueba con información de ventas de productos que fue utilizada como base de datos de los DW en el resto de la etapa de prototipado. Con estos DW simulados ya cargados se creó un cubo OLAP simple de 3 dimensiones y luego se lo publicó y ejecutó sobre cada plataforma, obteniendo como resultado una vista de análisis en cada una. Al tener en ambos prototipos la misma información con el mismo cubo, nos permitió realizar comparaciones de desempeño más reales entre ellos. También permitió realizar las evaluaciones de forma directa sobre el uso de las herramientas en cada una de las funcionalidades.

3.5. Evaluación y selección final

Se compararon los prototipos mencionados anteriormente tomando como métricas de evaluación las siguientes características:

1. Nivel de facilidad en la instalación de la plataforma.
2. Nivel de facilidad en el proceso de configuración.
3. Nivel de calidad y completitud de documentación brindada por la distribución.
4. Nivel de calidad y completitud de documentación brindada por fuentes externas a la distribución.
5. Nivel de accesibilidad y facilidad de uso de la herramienta.

Como rangos de evaluación se utilizaron valores enteros de 1 a 5, denotando que, a mayor valor mejor es la característica analizada en la plataforma referida y a menor

valor peor es dicha característica. La siguiente tabla muestra la comparación final junto con la valoración promedio de cada uno de los puntos listados anteriormente.

Plataforma	**1**	**2**	**3**	**4**	**5**	**Promedio**
Pentaho	4	4	4	5	5	**4,4**
SpagoBI	5	2	3	2	3	**3**

Cuadro 3.11: Tabla de comparación final de las plataformas.

A partir de los datos reflejados en la tabla luego de las evaluaciones se visualiza que Pentaho se diferencia bastante con respecto a SpagoBI básicamente en dos puntos principales, el primero es la cantidad de documentación e información que se encuentra disponible para consultar y el segundo la amigabilidad y facilidad de uso que brinda al usuario.

Con respecto a la documentación, pensamos que es un punto a favor muy importante de Pentaho. El hecho de no tener disponible soporte técnico sobre las herramientas obliga a contar con una amplia base de conocimientos y/o una comunidad de expertos activa al momento de realizar los desarrollos y mantenimientos sobre la plataforma.

También se considera importante cuan amigables sean las herramientas para los usuarios, a mayor facilidad de uso mayor aceptabilidad tendrá sobre estos y mejor será su utilización.

Por estos motivos, la plataforma que se seleccionó para la implantación de este proyecto fue Pentaho.

Capítulo 4

Diseño de la solución

En este capítulo se describe el diseño de la solución desarrollada para satisfacer el segundo objetivo del proyecto que corresponde a la implementación de un sistema de Data Warehouse (DW) corporativo utilizando la tecnología seleccionada. Este capítulo se organiza en seis partes. La Sección 4.1 presenta el diseño y la descripción de la arquitectura, en la Sección 4.2 se habla de la naturaleza de los datos, en la Sección 4.3 se detalla el análisis de los datos fuentes, la Sección 4.4 corresponde al análisis de la calidad de los datos, en la Sección 4.5 se presenta el modelado conceptual multidimensional y por último en la Sección 4.6 se detalla el diseño relacional de la base que soporta a los cubos.

4.1. Descripción de la arquitectura

Para tener una visión general del sistema y tomando en cuenta que la plataforma elegida para la implementación de la solución es Pentaho se explica a continuación la arquitectura típica utilizada en los sistemas de DW, detallando cada una de los sub-sistemas que conforman el sistema realizado y que se encuentra esquematizado en el diagrama (ver figura 4.1).

Se Diferencian seis grandes sub-sistemas en los cuales el sistema está estructurado:

- Fuentes de datos
- Extracción Transformación y Carga (ETL)
- On-Line Analytical Processing (OLAP)
- Presentación
- Seguridad
- Administración

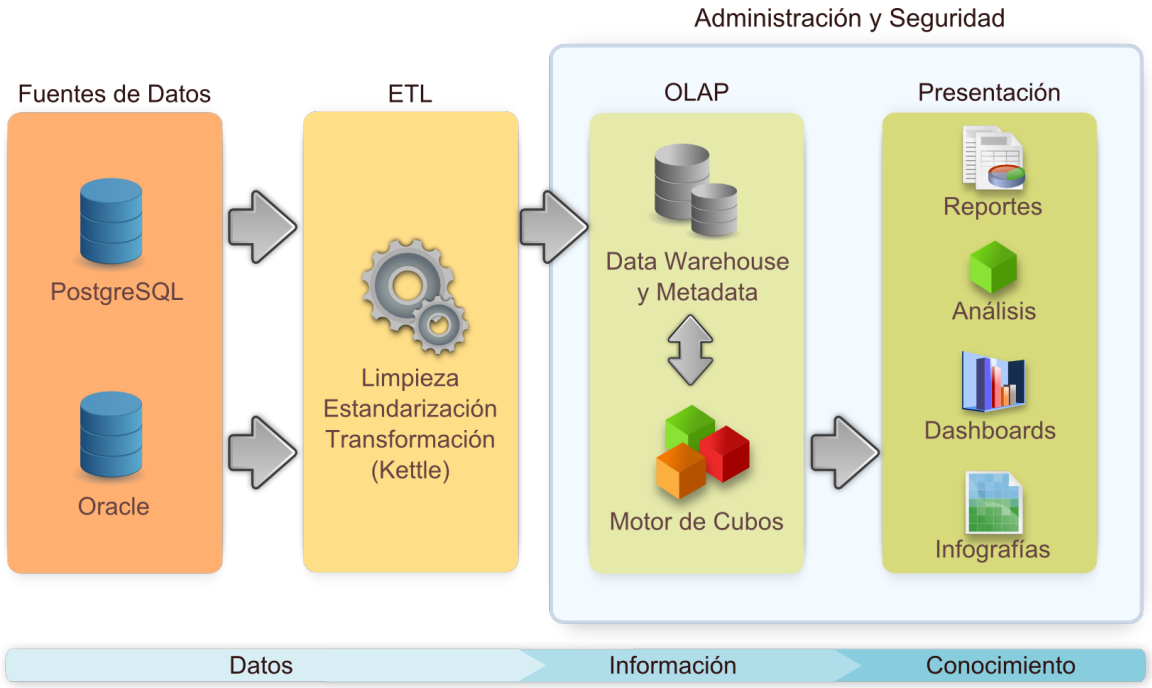


Figura 4.1: Arquitectura de la solución

El sub-sistema correspondiente a las fuentes de datos contiene las distintas fuentes que se utilizaron en la obtención de los datos que alimentan el sistema. En este proyecto solo se utilizaron como fuentes de datos dos bases de datos relacionales (Oracle y PostgreSQL), sin embargo, además de tener fuentes de tipo Relational Database Management System (RDBMS) también se podría obtener información desde otro tipo de fuentes como por ejemplo base de datos NoSQL (por ejemplo MongoDB), plataformas de búsqueda (por ejemplo Solr [1]), planillas de calculo, sistemas de archivos distribuidos (por ejemplo Hadoop [2]), archivos planos de texto, entre otros.

El área de ETL es la sección donde se agrupan una serie de sub-procesos que llevan a cabo tareas relacionadas con la extracción, manipulación, control, integración, limpieza de datos, carga y actualización del DW. Es decir, todas las tareas que se realizan desde que se toman los datos de las diferentes fuentes hasta que se cargan en el sistema para su utilización. En este sub-sistema se mantienen los datos obtenidos en una base de datos temporal (ver sección 5.3.2) que es usada para todos los procesos que ejecutan las tareas antes mencionadas, en nuestro caso se utilizó para este propósito una base de datos relacional PostgreSQL.

[1] *Plataforma de búsquedas distribuidas del proyecto Apache Lucene - http://lucene.apache.org/solr/-Ultimoacceso11/2011*

[2] *Framework para procesamiento distribuido de grandes cantidades de datos entre clusters de computadoras - http://hadoop.apache.org/-Ultimoacceso11/2011*

El sub-sistema OLAP es el núcleo del sistema que corresponde al repositorio central de información donde residen los datos actualmente utilizados. En el DW se almacenan los datos operacionales en estructuras multidimensionales que optimizan su acceso para las consultas y que son muy flexibles, además de contener la metadata de la información almacenada que ofrece información descriptiva sobre el contexto, la calidad, condición y características de los datos. En esta área se incluye el motor de cubos multidimensional que es el encargado de ejecutar las consultas realizadas por los componentes externos.

La Presentación es el área correspondiente a la interacción con el usuario, cuya finalidad es mostrar los datos almacenados de forma útil y transparente a través de las distintas herramientas. Este sub-sistema se comunica directamente con el servidor de cubos a través de consultas, las cuales retornan la información requerida donde ésta es transformada y presentada para la visualización final. Los reportes requeridos en el proyecto se encuentran en esta área.

En el área de Seguridad se encuentran definidas las restricciones de acceso a los objetos de la plataforma y a los diferentes recursos.

Por último, en el sub-sistema de administración se encuentran las herramientas administrativas de la plataforma. Gestión de usuarios, administración de conexiones de fuentes de datos, herramientas de limpieza de los diferentes cachés y el sistema de archivos interno del DW se encuentran en esta área.

Por mayor detalle de la arquitectura diseñada ver figura 2 en la Sección Anexo 1.

4.2. Análisis de las fuentes de datos

Los datos que se utilizaron para alimentar el DW corresponden al Sistema de Gestión de Afiliados de la Administración de los Servicios de Salud del Estado (ASSE). Este sistema tiene información del padrón de afiliados de ASSE y funciona como registro donde consta el conjunto de afiliados. Los datos son ingresados al sistema por funcionarios de ASSE desde los distintos centros de atención.

La base de datos no tiene suficientes restricciones referenciales ni método de verificación de la correctitud de los datos que se ingresan, pudiendo así ingresar información no solo incorrecta, sino también en diferentes formatos con significados no siempre interpretados de la misma forma.

Los reportes generados mensualmente en ASSE obtienen información del Sistema de Gestión de Afiliados que posee datos de las diferentes unidades asistenciales que maneja la institución. Este sistema es una de las principales fuentes de información que

se utilizaron a lo largo del proyecto. Puntualmente, es el proveedor de los datos a partir del cual se generaron los reportes mencionados anteriormente. Es por ello que se realizó un análisis exhaustivo de la base de datos para diferenciar cuales son las entidades relevantes para la solución del problema.

A partir del análisis de las tablas de la base de datos de origen correspondiente al Sistema de Gestión de Afiliados de ASSE se identifican más de treinta entidades. Luego de analizar los requerimientos y obtener información de cuales eran las entidades necesarias para los reportes solicitados se lograron identificar cuales eran las tablas que formarían parte de la solución. Estas son las que representan las entidades departamentos, personas, unidades ejecutoras, grupo familiar, afiliados y unidad asistencial. Para los primeros reportes solicitados se trabajó con las entidades departamentos, localidades y personas. Pensando en la creación de futuros cubos se incorporaron también las restantes entidades.

A continuación se describen dichas entidades y como éstas se relacionan dentro de la base de datos del Sistema de Gestión de Afiliados. Se adelantan también algunos problemas que serán abordados en la siguiente sección.

La entidad **Departamentos** corresponde a los diferentes departamentos que existen en Uruguay. Pese a que se cuenta con 19 departamentos, en esta tabla se identifican 20 registros. El registro extra se asigna cuando se desconoce el departamento del afiliado. Por otro lado se tiene la entidad **Localidades** que es conceptualmente una entidad débil de Departamentos. Luego de analizar los registros de la tabla correspondiente a este concepto se detectó que no existe ninguna localidad para el departamento 0 que corresponde al departamento "A Clasificar". Esto significa por ejemplo que si se tienen personas con una localidad asociada y que no tengan departamento, no se lograría tener una referencia a la tabla Localidades. Razonando de forma análoga, hay que agregar los casos en que el departamento exista y que no se tenga una localidad a la cual una persona haga referencia.

La entidad **Personas** representa a todas las personas que hay dentro del sistema de salud. Las mismas se identifican mediante el número de documento, el tipo de documento de la persona y el campo propiedad. Este último campo tiene por defecto el valor 0, en caso de que no sea 0 significa que la persona no cuenta con número de documento propio. Esto significa que el número documento encontrado en el registro es el número de documento de su madre y el campo propiedad tiene el valor más alto que está asociado al registro correspondiente a la madre del afiliado incrementado en uno.

La figura 4.2 presenta el modelo entidad-relación sobre la parte relevante de la base de datos correspondiente al Sistema de Gestión de Afiliados de ASSE.

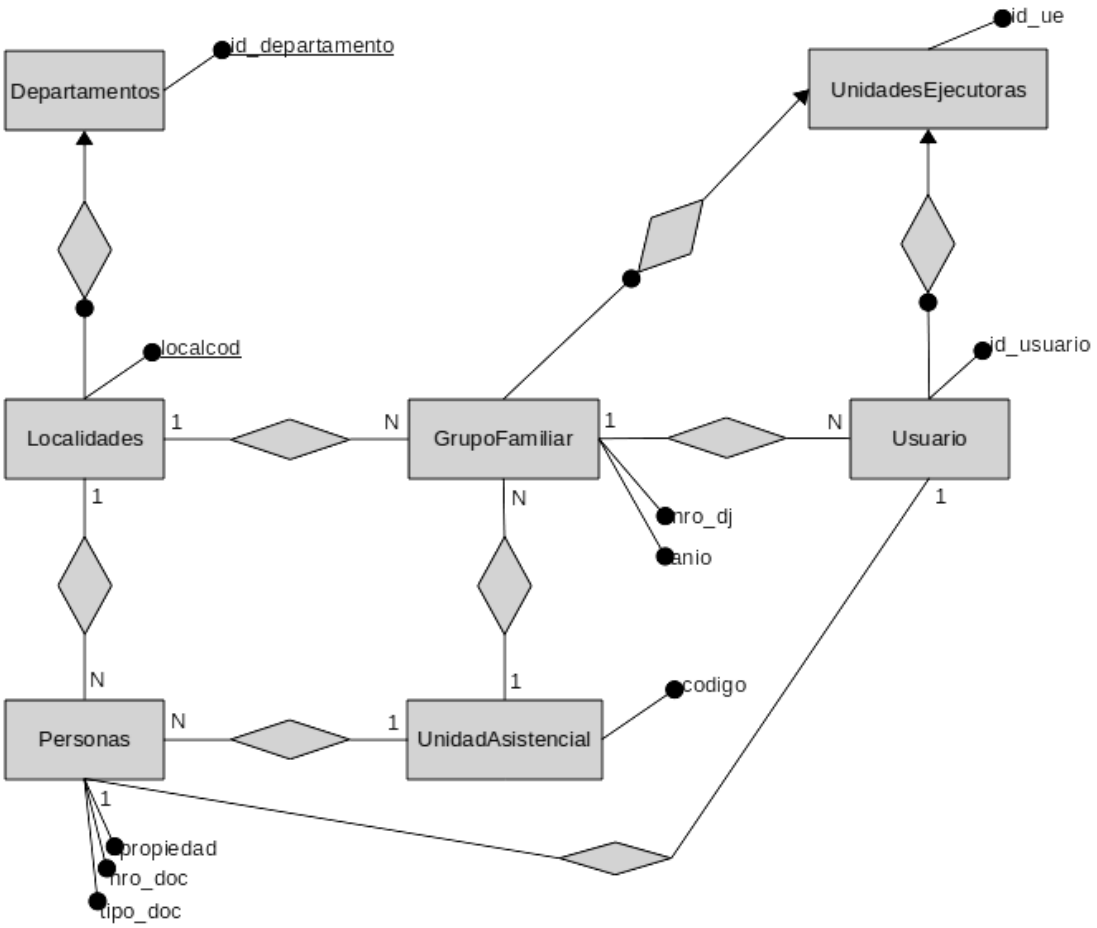


Figura 4.2: Diagrama MER de base de datos de gestión de afiliados

A continuación se presenta el modelo relacional de las tablas involucradas como fuentes de datos de nuestra solución [Figura 4.3].

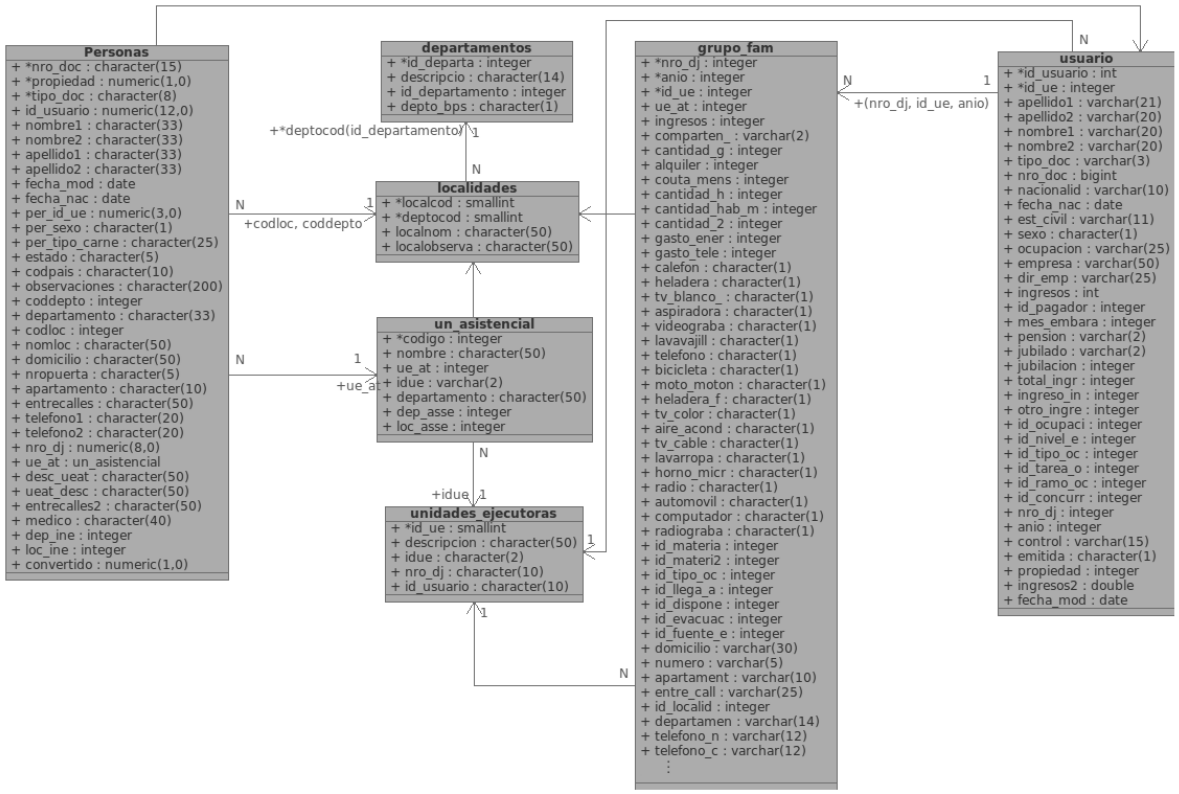


Figura 4.3: Modelo relacional de la base de datos de Gestión de AFiliados

4.3. Calidad de datos

Al analizar las fuentes de información se detectaron problemas con la calidad de los datos contenidos en las mismas, lo cual interfirió en la implementación del DW implementado. Si bien solucionar este problema no era un requerimiento del proyecto, fue necesario abordarlo para poder eliminar los inconvenientes que impedían avanzar con la carga de datos. La eficiencia en la toma de decisiones depende directamente de la calidad de los datos analizados, de modo que toda mejora en la exactitud de los mismos puede generar mejoras sustanciales en las decisiones tomadas. De la misma forma en que se mejora los niveles de exactitud de los datos se gana en credibilidad sobre la herramienta, haciendo que se fomente el uso de la misma por parte de los

usuarios.

Este aspecto se afrontó en dos etapas. La primer etapa tuvo como objetivo principal realizar un análisis que permitiera descubrir posibles valores inválidos, faltantes y/o mal representados, a los cuales se les pudiera aplicar acciones correctivas manuales o automatizadas a través de los procesos de carga. Esto permitió que los datos que sean integrados en el DW converjan a lo que representa la realidad. La segunda etapa consistió en notificar cuales son los problemas que no entran en la etapa anterior para que ASSE tenga la información de cuales son los puntos a mejorar en términos de calidad.

Con este análisis no se buscó tener una solución cien por ciento libre de defectos, sino que se trató de lograr la mejor correspondencia entre la estructura de la base de datos de origen y las tablas temporales del DW, a la vez se buscó recuperar la mayor cantidad posible de datos de manera de poder proveer un bloque de información robusto y confiable para la toma de decisiones.

Con respecto a la primer etapa, los tres grandes aspectos analizados que permitieron realizar la búsqueda de inconsistencias fueron separados en los siguientes puntos:

- **Análisis de metadata:** Mediante la comparación de los valores, las definiciones relacionales en las cuales los mismos estaban almacenados y los formatos definidos.
- **Análisis de contenido:** A través del análisis de las reglas de negocio y las propiedades de las columnas de los atributos, como por ejemplo dominios, rangos, conteos.
- **Análisis de relaciones:** Mediante verificaciones de integridad, relaciones entre claves foráneas y búsqueda de datos redundantes.

Una vez realizados estos análisis se pudo diferenciar un conjunto importante de problemas, los cuales fue necesario atacar y corregir. Los mismos se numeran a continuación:

1. Referencias a valores inexistentes de claves foráneas no implementadas como tales en la base de datos (ej: la tabla Personas debería tener una clave foranea a la tabla Departamentos).
2. Errores de integridad entre valores de diferentes tablas (ej: la edad de una persona es distinta a la diferencia entre la fecha de nacimiento y la fecha de realizado el registro).
3. Múltiples identificadores haciendo referencia a un mismo atributo (ej: el departamento correspondiente a Treinta y Tres estaba representado por muchos identificadores como 33, TT, TyT, Treinta y Tres).
4. Valores fuera de rangos posibles (ej: edades y fechas).

5. Redundancia de atributos en múltiples tablas (ej: la tabla personas contiene la descripción del departamento así como también una referencia a la tabla Departamentos).
6. Valores sintácticamente y/o semánticamente incorrectos usados como identificadores (ej: números de documento que incluían letras).
7. Valores de columnas con errores tipográficos (ej: este caso se pudo ver con las fechas ingresadas, ya que en vez de ingresar un nacimiento con el año 2010, se lo ingresaba con el año 2100).
8. Valores con múltiples formatos representando un mismo elemento (ej: celdas que podían contener *F* ó *1* y significaban el valor *Femenino*).

Para abordar de una manera más formal los problemas de calidad de datos, se identificaron las dimensiones de calidad presentadas en 2.2.7. Las dimensiones y sus factores de calidad fueron aplicados únicamente a los atributos y a las relaciones directamente relacionadas con nuestro problema.

Se pasan a describir los factores analizados junto con algunos ejemplos simples de problemas específicos encontrados, además de la acción tomada para cada caso.

4.3.1. Exactitud

Correctitud semántica

Con respecto a la correctitud semántica, se diferenció que tanto se corresponden los datos utilizados. Debajo se detallan algunos ejemplos de los problemas encontrados y las acciones tomadas para solucionarlos.

Problema: Valores mal definidos en campos de tablas

Por ejemplo, se encontraron personas que no tienen el valor sexo definido correctamente en la base de datos. Esto fue detectado al encontrar por ejemplo, personas con nombres correspondientes al sexo femenino y que tenían definido en la base de datos el sexo masculino.

Solución: La corrección de este tipo de errores no fue abordada por nosotros dado que involucra una comparación de los datos con el mundo real y la única forma de realizarlo era comparando el nombre de la persona contra un diccionario de nombres, lo cual excedía el alcance de este proyecto.

Correctitud sintáctica

Con respecto a la correctitud sintáctica, se verificó si los datos de las fuentes tienen errores sintácticos o de formato. Debajo se detallan algunos ejemplos de los problemas encontrados y las acciones tomadas para solucionarlos.

Problema: Errores de tipeo en valores de columnas

Se detectaron en la entidad personas errores de tipeo en registros como por ejemplo, la fecha de nacimiento correspondiente a una fecha futura, así como también con edades superiores a los 200 años.

Solución: Se pasaron todas las edades negativas o superiores a 125 años a un valor que denominamos *Indeterminado*, dado que tratar de calcular las edades reales utilizando las fechas de nacimiento podía agregar datos erróneos ya que gran cantidad de las mismas tenían errores o eran nulas.

Problema: Atributos identificados con múltiples identificadores

El atributo que corresponde al sexo de la persona no estaba en un mismo formato habiendo valores como *M*, *F*, *I*, *1*, *2* y *null*, los cuales representan a los sexos *masculino*, *femenino* e *indeterminado*.

Solución: Se realizaron tablas de mapeos para relacionar los múltiples valores a un único formato en el caso del sexo. Durante el proceso de carga de ETL se analizan los atributos de tipo *Sexo* y en caso de encontrar el valor "M" se le asignó el otro valor presente "1". En forma análoga se convirtió "F" en "2". Para los casos restantes se les asignó el valor "3", el cual corresponde a "No Definido".

Precisión

Con respecto a la precisión de los datos, existieron varios problemas de este tipo y que se daban en valores que requerían de buen nivel de detalles como por ejemplo las direcciones de los afiliados. Estos problemas no fueron tomados en cuenta dado que no influían directamente en los datos que utilizamos para la creación del DW.

4.3.2. Completitud

Con respecto a la completitud se intentó verificar si los orígenes de datos representan todos los objetos de la realidad y si los mismos se encuentran representados por todos los datos obtenidos.

Cobertura

Con respecto a la cobertura, se trató de verificar cuántas entidades de la realidad existen en los datos de origen. Un ejemplo de los problemas encontrados es el siguiente:

Problema: Falta de cobertura en filas de tablas referenciadas

Se detectaron problemas como en el caso presentado entre las entidades *localidades* y *personas*. En este caso se muestra como existen muchas personas que tienen una localidad asociada que no existe en la entidad localidades. En este caso se vio que la cobertura de los datos correspondiente a las localidades no es completa, ya que las localidades a las cuales se hace referencia no existen.

Solución: En este caso no se abordó el tema dado que requería de un tiempo excesivo de solución ya que se necesita crear una tabla de dominio de localidades con todas las localidades posibles del país, de manera que, al asociar una persona se lo haga por el identificador de la tabla de dominio.

Densidad

Con respecto a la densidad de los datos, se trató de verificar cuánta información existe y cuánta falta sobre las entidades de origen.

Problema: Alto porcentaje de valores nulos en atributos

Se detectaron casos sobre atributos de referencia en los cuales existía una gran cantidad de valores nulos.

Solución: Se pasaron todos los valores nulos al valor *Indeterminado* del referido correspondiente de forma de mantener una densidad completa sobre las referencias.

4.3.3. Frescura

En particular en este proyecto el problema de la frescura está acotado, dado que tomando en cuenta el grupo de entidades utilizado para la implementación, en el peor de los casos los datos más actuales tendrían un mes de antigüedad dentro del sistema, dado que la carga se realiza una vez al mes para todo el conjunto de valores.

4.3.4. Consistencia

Con respecto a la consistencia, lo que se trató de diferenciar es si se satisfacen las reglas semánticas definidas sobre los datos. En particular, si los datos cumplen las reglas del

dominio, si se satisfacen las dependencias referenciales.

Debajo se detallan los factores analizados y se explican algunos ejemplos de los problemas encontrados y las acciones tomadas para solucionarlos.

Integridad de dominio

En el caso de este factor de calidad se verificó si se satisfacían las reglas definidas sobre los contenidos de los atributos.

Problema: Valores fuera de rango

Como se mostró en la dimensión exactitud dentro del factor correctitud sintáctica, la edad está fuera del rango comprendido entre 0 y 120 años. En el contexto de la consistencia, este problema se categoriza dentro del factor de integridad de dominio. En la mayoría de los casos, los problemas de valores fuera de rangos ocurrían en campos con formato fecha.

Solución: En el caso de las edades se tomó la misma solución que en el ejemplo de correctitud sintáctica. En otros casos con campos que tenían formato de fechas fuera de rango no se tomaban como válidos valores que sobrepasaban las fechas de carga.

Integridad Intra-Relación

Con este factor se trató de verificar si se satisfacían las reglas definidas entre atributos de una misma tabla. Existieron varios problemas de este tipo pero no fueron analizados dado que no se encontraban en las tablas fuentes del DW.

Integridad Referencial

Relacionado a la integridad referencial se verificó si se satisfacían reglas entre los atributos de varias tablas. Analizando este factor, se detectaron muchos errores dentro de la base de datos. Esto sucede ya que hay muchas restricciones a nivel de afiliado que no están representadas como restricciones de integridad de la base de datos por medio de claves foráneas. Como ejemplo de esto, existen personas que no tienen departamento asociado, o que no tienen localidad asociada. Para categorizar estos casos se creo un registro con descripción "indeterminado" en las tablas que presentan el problema, con el fin de cambiar las referencias de las tablas que hacen referencia, colocándoles una referencia a dicho registro.

Otro ejemplo claro es el que ocurre en la tabla *personas*, la cual contiene referencias a la tabla *departamento* y donde las descripciones del departamento son distintas dentro de la tabla *personas*. Esto se ve representado en el cuadro 4.1, el cual se genera a partir

del Algoritmo 1.

Algoritmo 1 Selección de todos los códigos de departamento junto con su nombre y la cantidad de tuplas en la tabla de personas

```
SELECT coddepto,trim(departamento) as depto,count(*)
FROM personas
GROUP BY coddepto,trim(departamento) ORDER BY coddepto;
```

Cuadro 4.1: Resultados de la consulta

coddepto	depto	count
0		2
1	San Jose	1
1	Cerro Largo	1
1	Montevideo	510527
1		2
1		2
1	Rocha	1
1	Lavalleja	2
1	Canelones	3
1	Colonia	1
2	Artigas	56754
2	Canelones	28
3	Cerro Largo	10
3	Canelones	257406
4	Colonia	1
4	Cerro Largo	63003
5	Colonia	66863
6	Durazno	37798
7	Flores	17185
8	Florida	40178
8	Lavalleja	4
9	Maldonado	5
9	Lavalleja	33915
10	Maldonado	84446
10	Montevideo	1
11	Paysandu	58819
12	Rio Negro	37118
13	Rivera	70967
14	Rocha	40813
15	Salto	82417
16	Canelones	1

Continúa en la siguiente página

Cuadro 4.1 – continuado desde la página anterior

coddepto	depto	count
16	San Jose	63286
17	Soriano	43831
18	Tacuarembo	66708
19	Lavalleja	1
19	T.y Tres	35287
	Soriano	315
	Flores	297
	Maldonado	444
	Montevideo	805
		132658
	Durazno	844
	Rio Negro	283
	Colonia	2357
	Rocha	178
	Lavalleja	1150
	T.y Tres	111
	T	272
	Artigas	396
	Canelones	1973
	Cerro Largo	391

4.3.5. Unicidad

Al abordar esta dimensión de calidad se analizaron sus factores y no se detectaron problemas en las tablas que fueron utilizadas en la carga del DW.

4.4. Modelado Multidimensional

En esta sección se denotan cada uno de los elementos multidimensionales que forman parte de la solución. Para el modelado de la solución al problema planteado fueron analizados los siguientes reportes elaborados en forma mensual por ASSE, los cuales forman parte de los requerimientos iniciales:

- Total de afiliados a ASSE por tipo de cobertura según sexo y grupos de edad.
- Total de afiliados a ASSE por tipo de cobertura según departamento.
- Total de afiliados a ASSE por tipo de cobertura según departamento y unidad asistencial.

- Total de afiliados a ASSE por grupos de edad según departamento.
- Total de afiliados a ASSE por grupos de edad según departamento y unidad asistencial.

Estos requerimientos fueron ampliados por parte de ASSE con el fin de expandir la variedad de los reportes a presentar en nuestra solución. Dentro de los requerimientos figuraba el considerar diferentes grupos de edad y varios niveles en los tipo de cobertura. Otro punto que fue tomado en cuenta es que a ASSE le interesa tener datos en forma mensual únicamente. Esto llevó a que se tomara la decisión de hacer cargas incrementales en el DW para poder así obtener los datos correspondientes a cada mes. Las dimensiones que se desprenden de los requerimientos son las que se pasan a describir a continuación.

Dimensión Sexo

La dimensión ***Sexo*** es utilizada para el análisis de las afiliaciones según el género del afiliado. El único atributo que contiene esta dimensión es el nombre de sexo (ver figura 4.4).

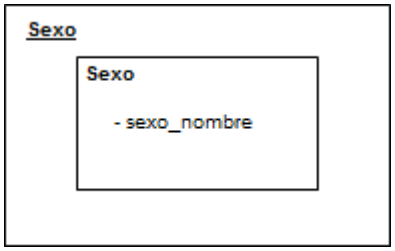


Figura 4.4: Dimensión Sexo

Dimensión Unidad Asistencial

La dimensión ***Unidad Asistencial*** es utilizada para el análisis de las afiliaciones según la unidad asistencial en la cual una persona fué afiliada a la institución. Para modelar esto se creó la dimensión con una jerarquía de dos niveles. En el nivel superior se representa la entidad departamentos, la cual está asociada a varias unidades asistenciales que se representan en el nivel de jerarquía inferior. Esto permite aplicar las operaciones de OLAP roll-up y drill-down para navegar por los niveles de la jerarquía dentro de la dimensión. (ver figura 4.5).

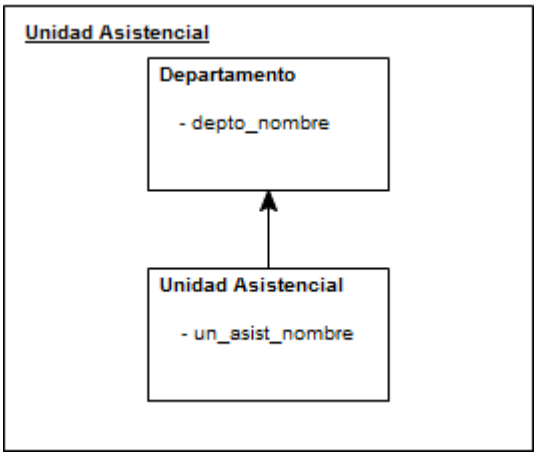


Figura 4.5: Dimensión Unidad Asistencial

Dimensión Tipo de Cobertura

Esta dimensión fue diseñada con una jerarquía de tres niveles, con el fin de obtener reportes con distinto nivel de granularidad dependiendo del nivel de jerarquía utilizado. Se detallan a continuación la composición de los diferentes niveles de la jerarquía:

- Carné
  - Asistencia
    - Bonificado 1
    - Bonificado 2
    - Carné de asistencia gratuito
    - Materno - Infantil
  - Vitalicio
- FONASA
  - Afiliados
  - De Oficio
- Cuotas
  - Convenios colectivos
  - Individuales
  - Núcleo familiar
- Convenio
  - Panes
  - Sanidad policial

Esta jerarquía fue brindada por ASSE para ser utilizada en los reportes solicitados (ver figura 4.6).

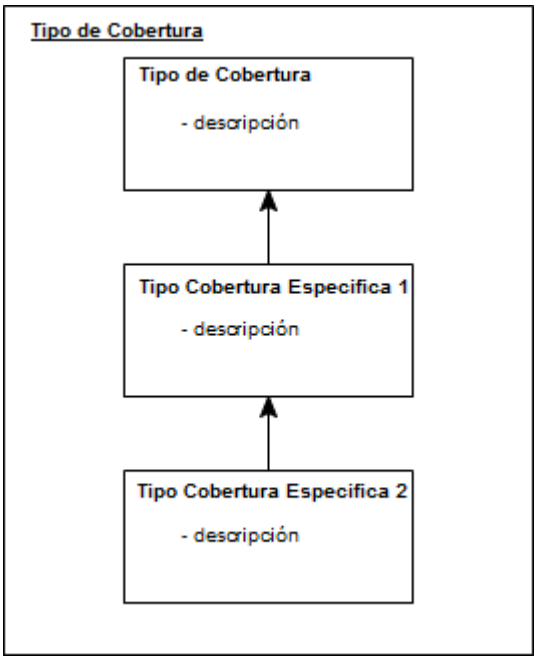


Figura 4.6: Dimensión Tipo de Cobertura

Dimensión Tiempo

Tiempo es la dimensión que determina a que mes y año pertenecen los datos cargados dentro del DW. Esto permite obtener reportes mensuales como lo pretende ASSE, siendo mes el nivel más bajo de la jerarquía y año el nivel más alto (ver figura 4.7). Dado que ASSE requiere que los datos sean cargados mensualmente no es necesario tener un nivel de granularidad más fino en esta dimensión.

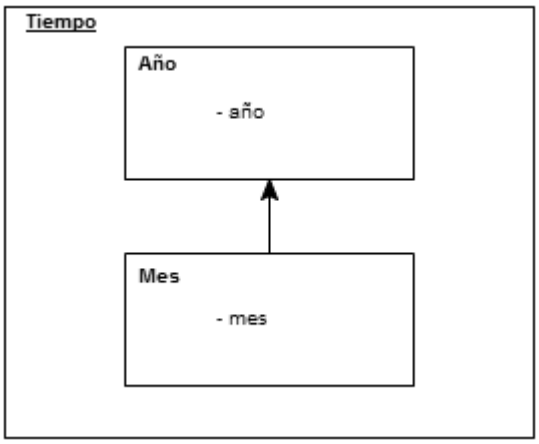


Figura 4.7: Dimensión Tiempo

Dimensiones Grupos de Edad

Existen dos criterios para clasificar los grupos de edad, uno surge del programa INFAMILIA (Infancia, Adolescencia y Familia), y el otro de FONASA (Fondo Nacional de Salud). Para poder analizar la información por cualquiera de los dos criterios se decidió modelarlos como dos jerarquías multidimensionales independientes y ambas jerarquías tienen un solo nivel.

Dimension Grupos de Edad INFAMILIA

Esta dimensión representa los grupos de edad llamado INFAMILIA. Agrupa los afiliados según este criterio, el cual permite a ASSE por medio de los reportes obtener información según estos grupos de edad. (ver figura 4.8).

Grupos de edad INFAMILIA:

- de 0 a 2
- de 3 a 5
- de 6 a 9
- de 10 a 11
- de 12 a 14
- de 15 a 19
- de 20 a 24
- de 25 a 29
- de 30 a 49
- de 50 a 64
- 65 y más

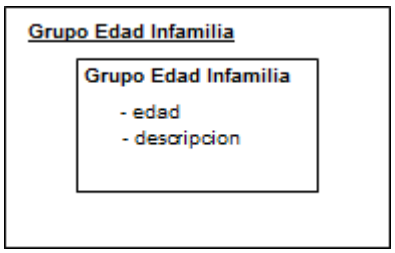


Figura 4.8: Dimensión Grupos de Edad INFAMILIA

Dimensión Grupos de Edad FONASA

Como en la dimensión representada en el punto anterior, los grupos de edad denominado FONASA agrupa los afiliados según el criterio FONASA. Esto permite a ASSE por medio de los reportes obtener información según grupos de edad FONASA (ver figura 4.9).

Grupos de edad FONASA:

- Menores de 1
- de 1 a 4
- de 5 a 14
- de 15 a 19
- de 20 a 44
- de 45 a 64
- de 65 a 74
- 75 y más

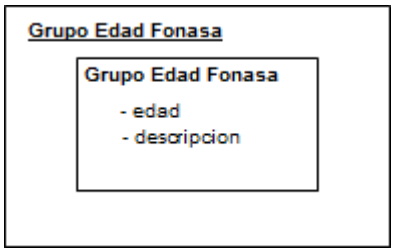


Figura 4.9: Dimensión Grupos de Edad FONASA

4.4.1. Esquema Multidimensional

En la figura 4.10 y 4.11 se muestran las dimensiones utilizadas para definir los cubos de las afiliaciones según grupo de edades INFAMILIA y grupo de edades FONASA respectivamente.

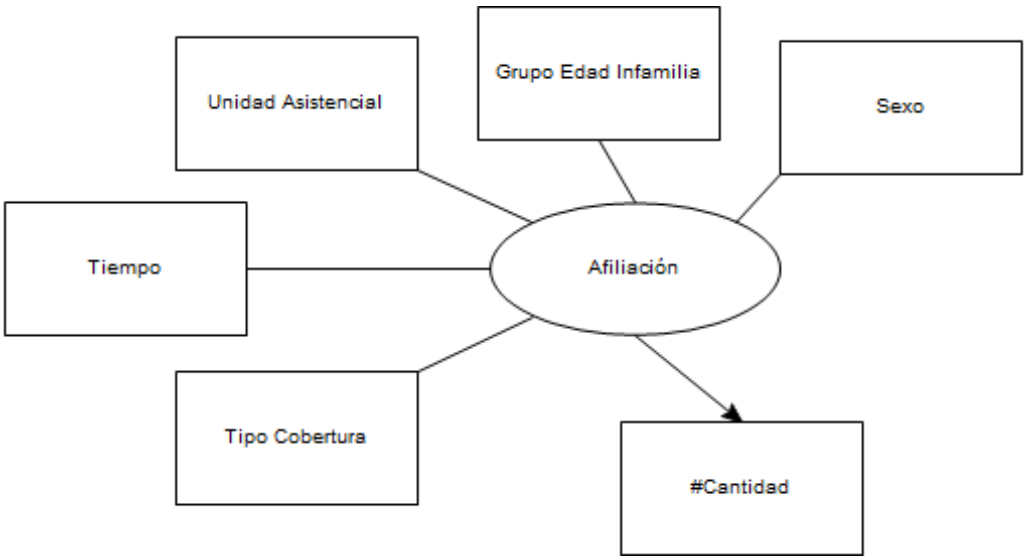


Figura 4.10: Relación dimensional para el cubo de afiliaciones según INFAMILIA

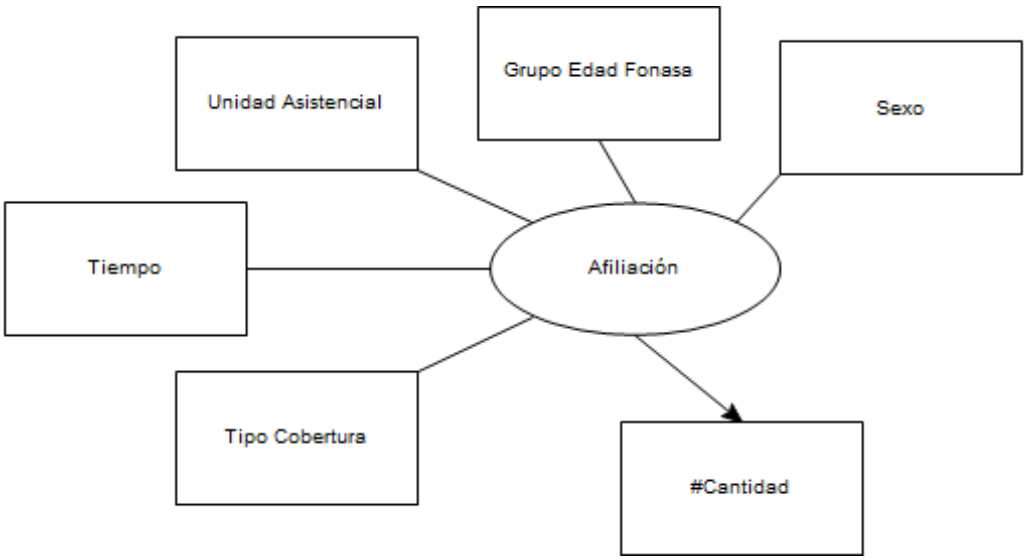


Figura 4.11: Relación dimensional para el cubo de afiliaciones según FONASA

4.5. Diseño relacional de la base que soporta a los cubos

La figura 4.12 representa el diagrama relacional de las estructuras relacionales usadas para la creación del modelo multidimensional usado para el manejo de las afiliaciones. Éste se deriva de las dimensiones y del esquema multidimensional. Se decidió utilizar un esquema estrella para el diseño en lugar de un esquema copo de nieve, ganando así simplicidad en el diseño y velocidad de acceso por tener las distintas jerarquías desnormalizadas. Como contrapartida al tomar esta decisión se tiene redundancia en los datos por tener los niveles de las jerarquías desnormalizados. Esto no afecta el volumen extra de datos que se pueda generar dado que la cantidad de registros de las dimensiones es relativamente pequeña.

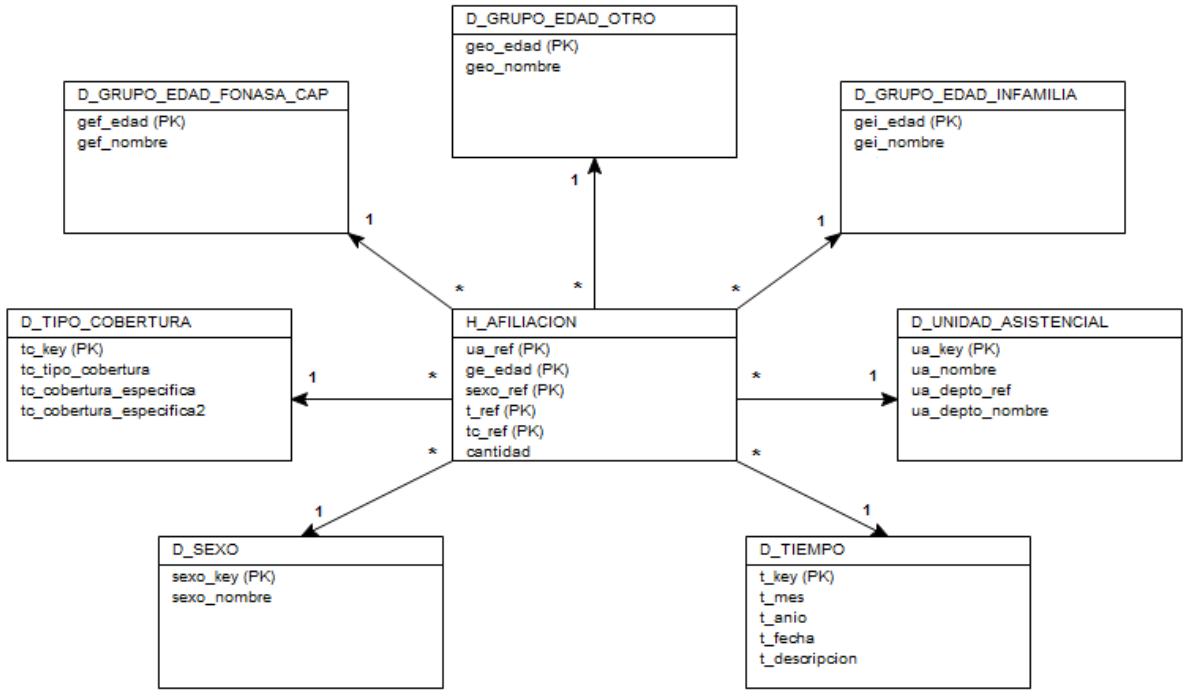


Figura 4.12: Modelo multidimensional en base a tablas relacionales para los cubos de afiliacion según FONASA e INFAMILIA

Hecho Afiliación

Este hecho modela la cantidad de afiliaciones según el cruzamiento de las dimensiones ya descritas como se puede ver en la figura 4.13. Cada cantidad se calcula contando las afiliaciones para un mes y año de afiliación, grupo de edad, sexo, unidad asistencial y tipo de cobertura de los afiliados.

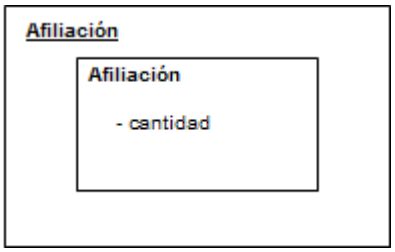


Figura 4.13: Hecho Afiliación

Concluyendo este capítulo, se destaca que se lograron diseñar los cubos capaces de soportar los requerimientos proporcionados por ASSE, así como también las tablas que representan los cubos. Para esto se abordaron también los problemas de calidad de datos y se buscaron soluciones en el diseño de tablas temporales para la posterior carga del DW.

Capítulo 5

Implementación del prototipo

En este capítulo se detalla el proceso de implementación de la solución diseñada. Se toman como ejemplo los reportes de *Afiliaciones* que fueron implementados para mostrar los artefactos utilizados en cada paso. En la sección 5.1 se describen los componentes utilizados de la plataforma seleccionada. La Sección 5.2 describe los objetos implementados luego de realizado el Diseño Lógico Multidimensional definido en el capítulo anterior. La Sección 5.3 detalla el proceso de Extracción Transformación y Carga (ETL) implementado para la carga de los datos dentro del Data Warehouse (DW), así como también, la etapa de limpieza realizada sobre los mismos. En las Secciones 5.4 a 5.7 se ven las herramientas de visualización utilizadas para la implementación. Dentro de las Secciones 5.8 y 5.9 se muestran las funcionalidades utilizadas de la herramienta implantada. Por último, en la Sección 5.10 se visualizan la distribución de los tiempos durante el proyecto.

5.1. Componentes de la arquitectura de la suite de Pentaho

La figura 5.1 muestra de forma estructurada la arquitectura funcional y los diferentes componentes que forman parte de la suite de Pentaho. Como se puede apreciar en dicha figura, Pentaho se puede separar en cuatro bloques principales que son:

- Orígenes de datos (capa de *3rd Party Applications* en la imagen). Donde se encuentran los sistemas desde los cuales se extrae la información.
- Integración de datos (capa *Data & Application Integration*). Es la capa en donde se encuentran la herramienta de ETL (Data Integration) y la de creación de metadata de cubos (Schema Workbench).
- Plataforma de Business Intelligence (BI) (capa *Business Intelligence Platform*). Es el conjunto de herramientas que permiten la administración y ejecución de los artefactos creados para realizar el análisis de los datos. En esta capa se encuentra el repositorio de archivos, la lógica de negocios, los sistema administrativos de la plataforma y los componentes que gestionan la seguridad sobre todos los artefactos.

- Capa de presentación (capa *Presentation Layer*) es donde se visualizan los resultados de las ejecuciones de los distintos artefactos creados para realizar el análisis. En la figura se detallan los distintos tipos de visualizaciones de datos que permite crear la herramienta y que son los reportes, los On-Line Analytical Processings (OLAPs) y los tableros de mando o Dashboards.

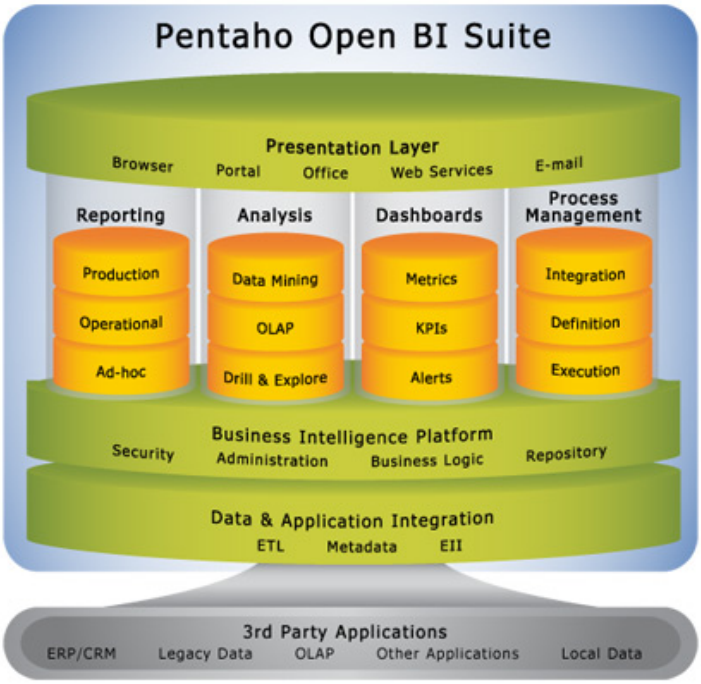


Figura 5.1: Arquitectura de la plataforma Pentaho [1]

En nuestra solución se utilizaron la mayoría de los componentes propuestos por la plataforma. Entre los más relevantes se encuentran el componente de ETL para la extracción transformación y carga de los datos y todos los componentes que conforman la capa que contiene la plataforma de BI. Sobre la capa de presentación de los datos se implementaron sobre el análisis, reportes y dashboards. Dentro de la capa de Integración de datos otra herramienta utilizada para crear la metadata de los cubos.

Los artefactos más importantes, dado que son utilizados en todas las capas de la plataforma son los cubos OLAP, que son las estructuras multidimensionales que guardan la información dentro de nuestro sistema y que permiten realizar operaciones sobre la misma.

[1] *Arquitectura de la plataforma Pentaho - http://www.pentaho.com/images/pentaho_functional_architecture.jpg-Ultimoacceso11/2011*

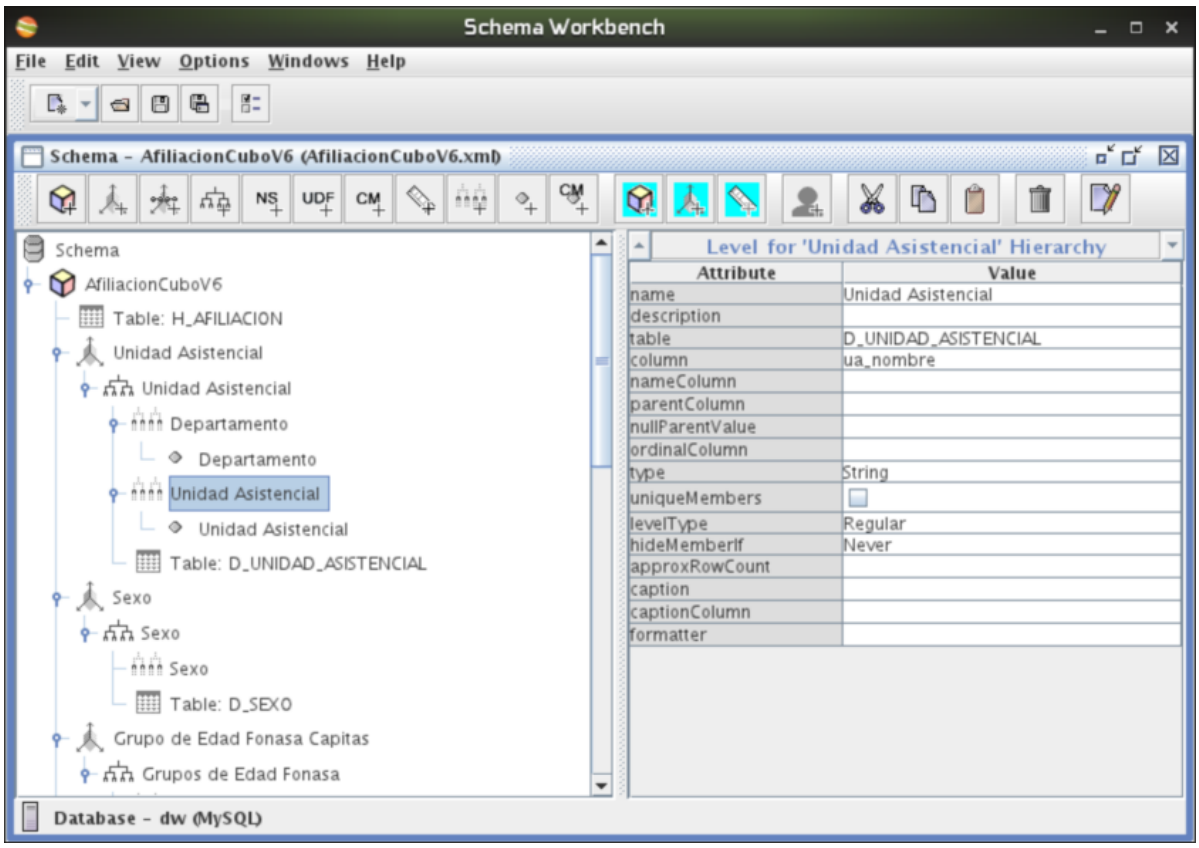


Figura 5.2: Captura de pantalla de la herramienta Schema Workbench

5.2. Cubos y Análisis OLAP

La definición del cubo, es el primero de tres pasos en la creación de éste y donde básicamente se debe decidir cuales son los atributos que son requeridos para realizar el análisis, así como también el diseño del tipo de estructura lógica que se va a requerir. Como segundo paso se debe especificar la estructura de las dimensiones diseñando las agregaciones (elementos precalculados de datos) junto con la creación de todas las tablas relacionales requeridas, y como ultimo se debe realizar la carga del cubo para procesarlo.

Para poder realizar análisis OLAP en Pentaho se deben tener primero los cubos de análisis multidimensional de información de Mondrian. Los cubos que utiliza este servidor son llamados esquemas y son archivos XML que definen las dimensiones que va a tener el cubo. Dichos esquemas pueden ser generados de forma manual o a través de la herramienta de Schema Workbench [14] provista en la plataforma. En este proyecto se utilizó la herramienta de Pentaho antes mencionada para la creación de los cubos, la figura 5.2 muestra una captura de pantalla de dicha herramienta en la cual se está editando un cubo de la implementación.

5.2.1. Estructura General

En Pentaho los cubos se especifican mediante un archivo ([12]) donde se describe la estructura de los mismos (dimensiones, jerarquías, niveles, hechos, medidas) así como se realizan los mapeos entre estas estructuras y las tablas donde se encuentran almacenados los datos que pueblan las mismas.

A continuación se presenta la estructura general de los cubos utilizados dentro del proyecto.

Como se puede ver en la figura 5.3 se denota que en la herramienta la estructura del cubo se detalla como una estructura arbórea en la cual a partir del nodo padre *Cubo* se desprenden cada uno de los componentes que lo definen. En el caso de la figura el cubo de afiliaciones de nombre *CuboAfiliacion* contiene un conjunto de dimensiones que se encuentran en el primer nivel, también la tabla de hechos de la misma (de nombre *H_AFILIACION*) y por último la medida que se desea registrar (en este ejemplo el campo de nombre *cantidad*).

Figura 5.3: Representación de cubo Afiliación

Desplegando la estructura correspondiente al diseño de la dimensión *Tipo de Cobertura*, se puede observar la representación de una jerarquía interna de tres niveles. Estos son el *Tipo de Cobertura*, *Tipo de Cobertura Específica* y *Tipo de Cobertura Específica 2*, lo cual permite desagregar el concepto de *Tipo de Cobertura Específica* en cada uno de los tipos de cobertura que lo componen así como también permite desagregar entre *Tipo de Cobertura Específica 2* y los anteriores. La figura 5.4 ilustra esta representación.

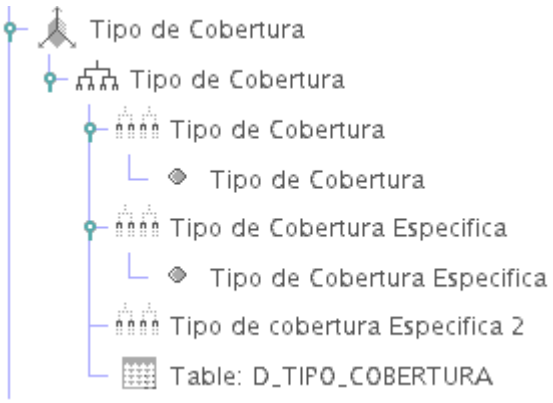


Figura 5.4: Tipo de Cobertura

En el siguiente XML se detalla la metadata correspondiente a la estructura referida en la figura 5.4. Se puede observar el mapeo entre la tabla *D_ TIPO_ COBERTURA* y la dimensión con la jerarquía Tipo de Cobertura. Debajo del nodo correspondiente a la jerarquía se encuentran también los tres niveles definidos con los elementos de nombre *Level*. En este lugar se hace el mapeo entre los campos de la tabla y el nivel de la jerarquía. Un ejemplo de esto puede ser visto en el tercer elemento *Level* con la etiqueta *Tipo de Cobertura Especifica* y que está mapeado con el campo *tc_ cobertura_ especifica2* de la tabla *D_ TIPO_ COBERTURA*.

```
<Dimension type="StandardDimension" foreignKey="tc_ref" highCardinality="false" name="Tipo de Cobertura">
  <Hierarchy name="Tipo de Cobertura" hasAll="true" allMemberName="Todos" primaryKey="tc_key">
    <Table name="D_TIPO_COBERTURA"></Table>
    <Level name="Tipo de Cobertura" table="D_TIPO_COBERTURA" column="tc_tipo_cobertura" type="String"
          uniqueMembers="false" levelType="Regular" hideMemberIf="Never">
      <Property name="Tipo de Cobertura" column="tc_tipo_cobertura" type="String"></Property>
    </Level>
    <Level name="Tipo de Cobertura Especifica" table="D_TIPO_COBERTURA" column="tc_cobertura_especifica"
          type="String" uniqueMembers="false" levelType="Regular" hideMemberIf="Never">
      <Property name="Tipo de Cobertura Especifica" column="tc_cobertura_especifica" type="String">
      </Property>
    </Level>
    <Level name="Tipo de cobertura Especifica 2" table="D_TIPO_COBERTURA" column="tc_cobertura_especifica2"
          type="String" uniqueMembers="false" levelType="Regular" hideMemberIf="Never">
    </Level>
  </Hierarchy>
</Dimension>
```

En el anexo correspondiente a la implementación de los cubos se puede observar el resto de la solución correspondiente a esta sección.

5.2.2. Publicación

Para que los cubos sean accedidos por el motor de cubos estos deben publicarse dentro de la plataforma Pentaho. Existen dos maneras de publicar los cubos en la aplicación, la primera es copiando los archivos XML de definición en el repositorio de esquemas del servidor de cubos *Mondrian* dentro de la aplicación y luego modificando los archivos de configuración que contienen los datasources del mismo. Este archivo de configuración le indica al motor de cubos donde se encuentra la metadata de los mismos, la cual le informa cuales son las tablas de la base de datos a utilizar al momento de la ejecución. Ésto debe realizarse cada vez que se crea un cubo lo cual lo hace bastante engorroso para el usuario final. La segunda manera es utilizando la funcionalidad de la herramienta, lo cual es bastante más directo y sencillo. Para realizar la publicación de un esquema en Pentaho desde el Schema Workbench ver el en el Anexo la sección de publicación de esquemas.[6.4]

5.3. Extracción, transformación y carga

En esta sección se explican los pasos seguidos para realizar la extracción, transformación y carga de los datos dentro del DW desde las fuentes de datos externas. De aquí en más se pasará a referir a la extracción, transformación y carga como ETL.

5.3.1. Introducción

Como ya se indicó anteriormente, en esta solución se decidió utilizar la herramienta *Pentaho Data Integration* de la plataforma Pentaho, también conocida como *Kettle*. Esta cuenta con varias aplicaciones para la manipulación de datos, entre las que se encuentran *Spoon*, *Pan* y *Kitchen*. Por medio de *Spoon*, se puede acceder a una interfaz gráfica que permite diseñar los elementos principales de un proceso de ETL, los cuales son : los trabajos y las transformaciones. *Pan* es una aplicación que interpreta y ejecuta transformaciones diseñadas con *Spoon*. De forma similar, *Kitchen* es una aplicación que interpreta y ejecuta trabajos diseñados con *Spoon*.
Mediante estas 3 herramientas se pueden cubrir todos los requerimientos de carga del Data Warehouse.

En el anexo de ETL en la página 127 se detallan los trabajos de ETL creados con las herramientas durante este proyecto.

5.3.2. Tablas auxiliares para la carga de datos

Las tablas auxiliares se crearon sobre un motor de base de datos PostgreSQL. Son utilizadas con el fin de cargar y actualizar algunas dimensiones del DW que no se derivan directamente de las fuentes externas, además se utilizan para registrar los

eventos de carga automática dentro del DW.

Las tablas utilizadas en la base de datos temporal son las que se presentan a continuación:

- log_carga_dw: Es utilizada para registrar los logs de carga dentro del DW, lo cual se explicará con mayor detalle en la sección 5.3.4.
- lkp_grupo_edad_fonasa_cap: Contiene la definición del grupo de edad fonasa, en caso de que la misma sea modificada se debe actualizar la dimensión D_GRUPO_EDAD_FONASA_CAP del DW.
- lkp_grupo_edad_infamilia: En forma análoga, esta tabla contiene la definición del grupo de edad infamilia.
- lkp_tipo_cobertura: En el Sistema de Gestión de Afiliados de ASSE se cuenta con dos registros que permiten clasificar a los afiliados según su tipo de cobertura, estos son el tipo de carné y el estado. Con esta información, y haciendo un mapeo se puede obtener el tipo de cobertura especifico definido para cada afiliado.

5.3.3. Mejora en calidad de datos

Como se vio en el capítulo anterior, en la sección 4.3 se aborda el problema de la calidad de los datos. El proceso comenzó con el análisis de los datos fuentes del Sistema de Gestión de Afiliados a ser utilizados en el DW. Este sistema está representado por treinta y cuatro entidades. Para que puedan ser cumplidos los requerimientos solicitados en la creación de los reportes se enfatizó en la depuración de las siguientes entidades:

- Departamentos
- Personas
- Unidad Asistencial

Teniendo en cuenta posibles reportes a futuro se decidió incorporar al proceso tres entidades suplementarias:

- Grupo Familiar
- Localidades
- Afiliados

Esto se logró copiando los datos correspondientes a las entidades dentro de una base de datos intermedia al DW. En estas tablas se implementó una estandarización en la nomenclatura que solo contienen los atributos necesarios para la posterior carga del DW.

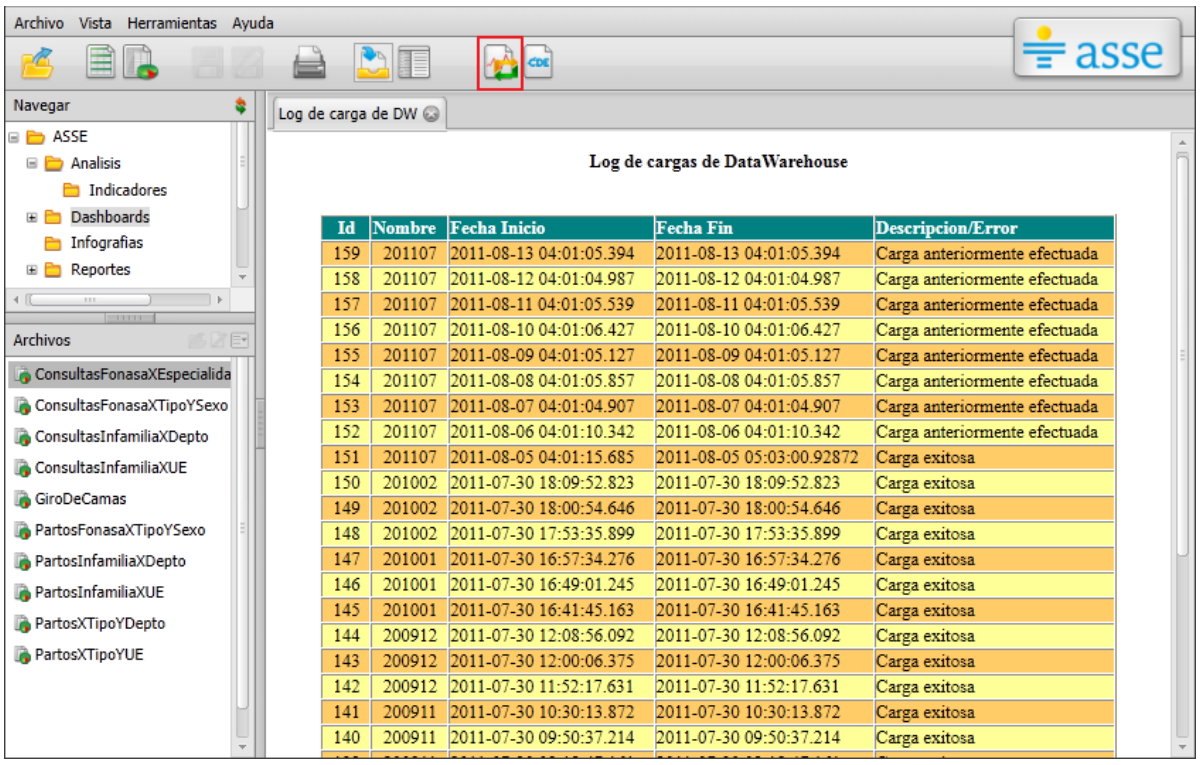
Los problemas encontrados y analizados en el capítulo 4.3, en algunos casos fueron corregidos y en otros fueron comunicados a ASSE, con el fin de que se puedan tomar medidas preventivas para mejorar los datos fuentes provenientes del Sistema de Gestión de Afiliados al momento de realizar futuros procesos de ETL.

5.3.4. Verificación de carga a realizar

Este proceso consiste en identificar si el DW necesita ser cargado. La verificación se hace diariamente y consiste en identificar si el esquema del mes anterior ya está disponible en un lugar predefinido para realizar la carga. En caso de que esté disponible y de que la carga no se haya hecho con anterioridad, se realiza el proceso de carga. La verificación de la carga dentro del DW es responsabilidad del proceso de ETL, lo cual permite que se realice en forma autónoma, sin necesidad de intervención de un administrador del sistema. En caso de que por algún motivo como la caída de la conexión a una de las bases de datos fuentes, deje a la tabla que se encarga de registrar las cargas en un estado inconsistente, el proceso puede recuperarse de ese estado. A la siguiente carga puede diagnosticar que la carga que se está actualmente efectuando, o definirla como inconsistente y arrancar una nueva. Cada ejecución del proceso genera un archivo de log que en caso de algún problema del cual el proceso no se pueda recuperar, un administrador del sistema puede verificar para detectar el origen del problema.
El proceso descrito se logra haciendo un chequeo en una tabla llamada *log_ carga_ dw*, que se encuentra en la base de datos temporal. En dicha tabla se genera un log que corresponde al comienzo de carga de cada trabajo ejecutado, el cual recibe dos parámetros de entrada. Estos corresponden al año y el mes que se quiere cargar, los cuales se ingresan en un formato "yyyyMM", y el segundo parámetro para el log es el proceso de carga que se ejecuta. En función de estos datos se puede saber si un la carga ya fue ejecutada, si se está ejecutando, o si se debe ejecutar. Los estados que se utilizan para el registro de los resultados de los trabajos son los siguientes:

1. cargando
2. carga exitosa
3. carga anteriormente efectuada
4. actualmente cargando en otra ejecución
5. esquema de base de datos de origen no existente
6. error en la carga
7. time out en la carga

La figura 5.5 corresponde a una captura del verificador de cargas que se implementó como complemento sobre el Bi Server de Pentaho y que es accedido desde la aplicación. El log muestra todas las cargas realizadas por el DW y el estado de cada ejecución.



Id	Nombre	Fecha Inicio	Fecha Fin	Descripcion/Error
159	201107	2011-08-13 04:01:05.394	2011-08-13 04:01:05.394	Carga anteriormente efectuada
158	201107	2011-08-12 04:01:04.987	2011-08-12 04:01:04.987	Carga anteriormente efectuada
157	201107	2011-08-11 04:01:05.539	2011-08-11 04:01:05.539	Carga anteriormente efectuada
156	201107	2011-08-10 04:01:06.427	2011-08-10 04:01:06.427	Carga anteriormente efectuada
155	201107	2011-08-09 04:01:05.127	2011-08-09 04:01:05.127	Carga anteriormente efectuada
154	201107	2011-08-08 04:01:05.857	2011-08-08 04:01:05.857	Carga anteriormente efectuada
153	201107	2011-08-07 04:01:04.907	2011-08-07 04:01:04.907	Carga anteriormente efectuada
152	201107	2011-08-06 04:01:10.342	2011-08-06 04:01:10.342	Carga anteriormente efectuada
151	201107	2011-08-05 04:01:15.685	2011-08-05 05:03:00.92872	Carga exitosa
150	201002	2011-07-30 18:09:52.823	2011-07-30 18:09:52.823	Carga exitosa
149	201002	2011-07-30 18:00:54.646	2011-07-30 18:00:54.646	Carga exitosa
148	201002	2011-07-30 17:53:35.899	2011-07-30 17:53:35.899	Carga exitosa
147	201001	2011-07-30 16:57:34.276	2011-07-30 16:57:34.276	Carga exitosa
146	201001	2011-07-30 16:49:01.245	2011-07-30 16:49:01.245	Carga exitosa
145	201001	2011-07-30 16:41:45.163	2011-07-30 16:41:45.163	Carga exitosa
144	200912	2011-07-30 12:08:56.092	2011-07-30 12:08:56.092	Carga exitosa
143	200912	2011-07-30 12:00:06.375	2011-07-30 12:00:06.375	Carga exitosa
142	200912	2011-07-30 11:52:17.631	2011-07-30 11:52:17.631	Carga exitosa
141	200911	2011-07-30 10:30:13.872	2011-07-30 10:30:13.872	Carga exitosa
140	200911	2011-07-30 09:50:37.214	2011-07-30 09:50:37.214	Carga exitosa

Figura 5.5: Log de carga del DW sobre el servidor de Pentaho y botón de acceso desde la herramienta

En el algoritmo 2 define el proceso realizado para el registro de las ejecuciones y sus cambios de estado.

Algoritmo 2 Registro de ejecuciones

Entrada: Parámetro *MES_ESQUEMA*

```
1: Se obtiene la referencia de la nueva tupla de la tabla log_carga_dw
2: Se obtiene el mes y el año de la variable MES_ESQUEMA correspondiente al
   esquema que se quiere importar
3: Se obtiene la fecha actual
4: Continuar = 0
5: si Ya fue cargado con ese mes y ese año entonces
6:   Estado = 2 //carga anteriormente efectuada
7: si no
8:   si Está cargando actualmente con ese mes y año entonces
9:     si no han transcurrido más de seis horas entonces
10:      Estado = 3 //actualmente cargando
11:      registrarLog(estado)
12:      return
13:    si no
14:      Estado = 6 //time out en la carga
15:    fin si
16:  si no
17:    si no existe esquema con ese mes y ese año entonces
18:      Estado = 4 //esquema no existente
19:    si no
20:      Estado = 1 //cargando
21:      Continuar = 1
22:    fin si
23:  fin si
24: fin si
25: registrarLog(estado)
26: devolver Continuar
```

5.3.5. Diseño de procesos de carga y actualización

Las dimensiones utilizadas por los cubos son cargadas y actualizadas de forma automática por procesos de carga dentro del ETL. Lo que se presenta a continuación son los algoritmos de carga y otro correspondiente a un algoritmo de actualización que utiliza procesos para realizar lo requerido. Se explicará también el proceso de carga correspondiente a la tabla de hechos de los dos cubos de afiliación.

A modo de ejemplo se presenta el proceso de mantenimiento de la dimensión *D_TIPO_COBERTURA*, se realiza insertando registros nuevos o actualizando registros ya existentes. El proceso consiste en recorrer cada Tipo de Cobertura definido en una tabla auxiliar, en caso de que uno de estos no exista, se obtiene una referencia nueva para luego insertar el nuevo elemento. En el segundo algoritmo se realiza el

chequeo inverso. Para cada Tipo de cobertura activo dentro de la dimensión, se verifica que esté presente en la tabla auxiliar. En caso de no existir, se lo define como inactivo dentro de la dimensión.

Algoritmo 3 Carga inicial de Dimensión D_TIPO_COBERTURA

```
1: para cada tipo de cobertura definido en LKP_TIPO_COBERTURA hacer
2:    si no existe o existe en estado inactivo en D_TIPO_COBERTURA entonces
3:       Se obtiene nueva referencia
4:       Se inserta en la tabla D_TIPO_COBERTURA
5:    fin si
6: fin para
```

Algoritmo 4 Actualización de Dimensión de D_TIPO_COBERTURA

```
1: para cada tupla de D_TIPO_COBERTURA en estado activo hacer
2:    si no existe en LKP_TIPO_COBERTURA entonces
3:       Se cambia el estado a inactivo
4:    fin si
5: fin para
```

En la figura 5.6 se muestra mediante un diagrama la carga de las dimensiones de nuestra solución. Las cargas están estructuradas en tres partes. Origen de los datos que se utilizan para cargar la dimensión, procesamiento interno de los datos, e inserción dentro de la dimensión correspondiente.

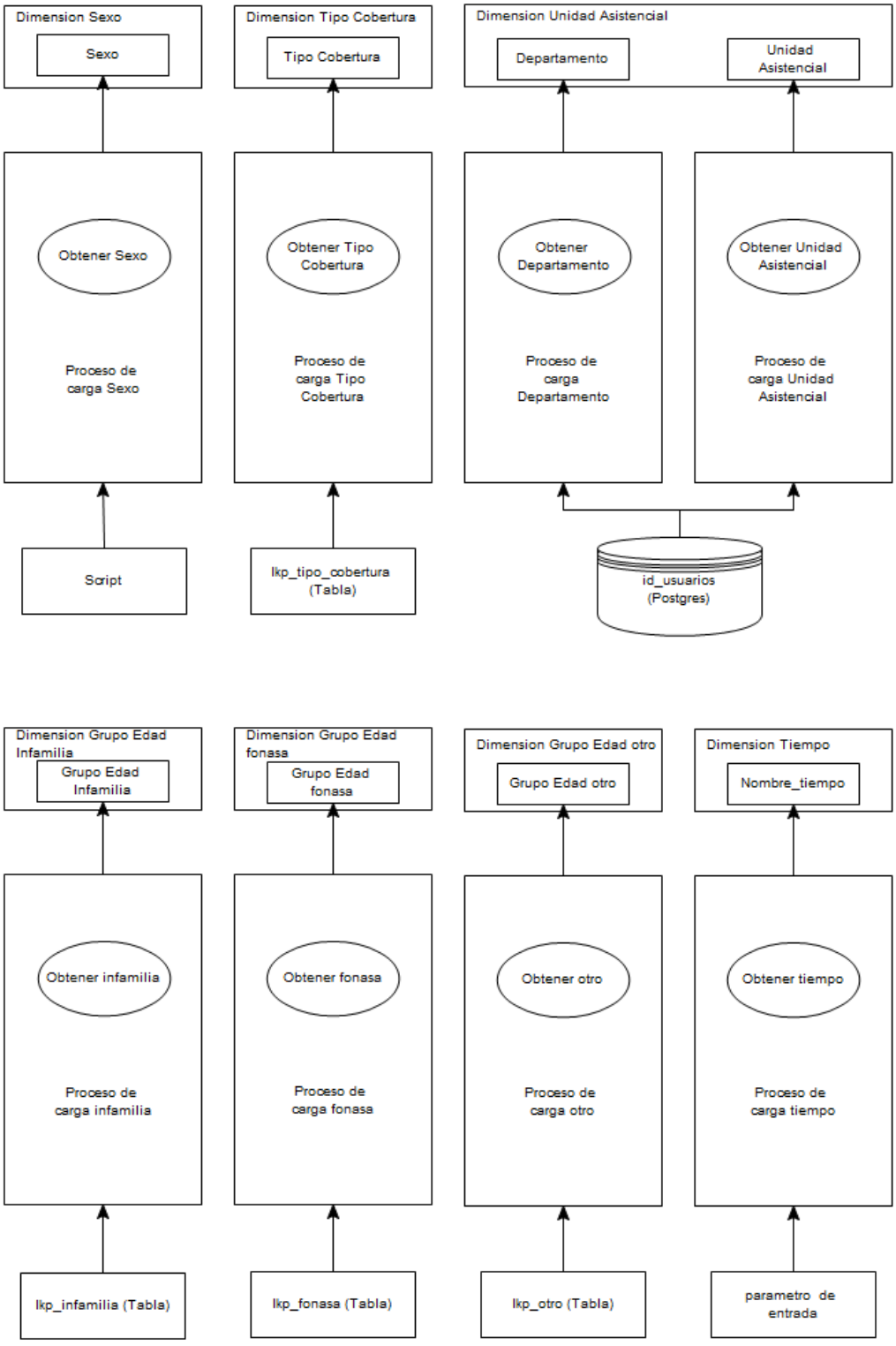


Figura 5.6: Diagrama de cargas de dimensiones

El algoritmo presentado a continuación, representa el proceso de carga de la tabla de hechos de los dos cubos de afiliaciones. El proceso consiste en obtener todas las personas junto con su correspondiente unidad asistencial y demás datos relacionados.

Estos datos son agrupados por su sexo, edad, tipo de cobertura, y unidad asistencial. Se aplican filtros con el fin de asignarle valores indeterminados a valores fuera del rango predefinidos correspondientes al sexo y a la edad de los afiliados. Se obtiene el mes y el año correspondiente a parámetro de entrada MES_ESQUEMA utilizados para relacionar la carga con la dimensión tiempo correspondiente. Se busca la referencia del tipo de cobertura del afiliado en la tabla de lookup auxiliar de los tipos de cobertura, y en caso de no encontrarlo se obtiene el tipo de cobertura definido como elemento por defecto dentro de la misma tabla auxiliar. Si la referencia es inexistente dentro de las dimensiones de tipo de cobertura se genera un error en el log de carga. Para finalizar el proceso, se ordenan las tuplas, se agrupan y se insertan en la tabla de hechos.

Algoritmo 5 Carga de la tabla de hechos H_AFILIACION

1: **para cada tupla de la tabla personas con unidad asistencial hacer**
2: (agrupados por sexo, la edad, el tipo de cobertura, la unidad asistencial)
3: para cada elemento de la tupla se clasifican los valores de sexo y edad
4: Se obtiene el parámetro MES_ESQUEMA que contiene la información del mes y del año de la dimensión tiempo
5: Se busca en LKP_TIPO_COBERTURA la clasificación correspondiente de los elementos encontrados en tipo de cobertura obtenidos de las personas
6: **si** no se encuentra tipo de cobertura **entonces**
7: Obtener clasificación del tipo de cobertura por defecto desde LKP_TIPO_COBERTURA para así asignárselo a la tupla
8: **fin si**
9: Se busca la referencia en la dimensión D_TIPO_COBERTURA correspondiente a la clasificación del tipo de cobertura de la tupla.
10: **si** no se encuentra referencia **entonces**
11: Se loguea el error
12: **si no**
13: Se ordena la tupla en función de las referencias de las dimensiones de la tabla de hechos
14: Se agrupan las tuplas
15: Se insertan en la tabla de hechos
16: **fin si**
17: **fin para**

En la figura 5.7 se ve representado el flujo de carga descripto en el algoritmo anterior. Este comienza en la parte inferior de la figura haciendo el join entre las tablas personas

y unidad asistencial de la base de datos fuente. Culmina insertando en la tabla *H_ AFILIACION* de la base de datos del cubo.

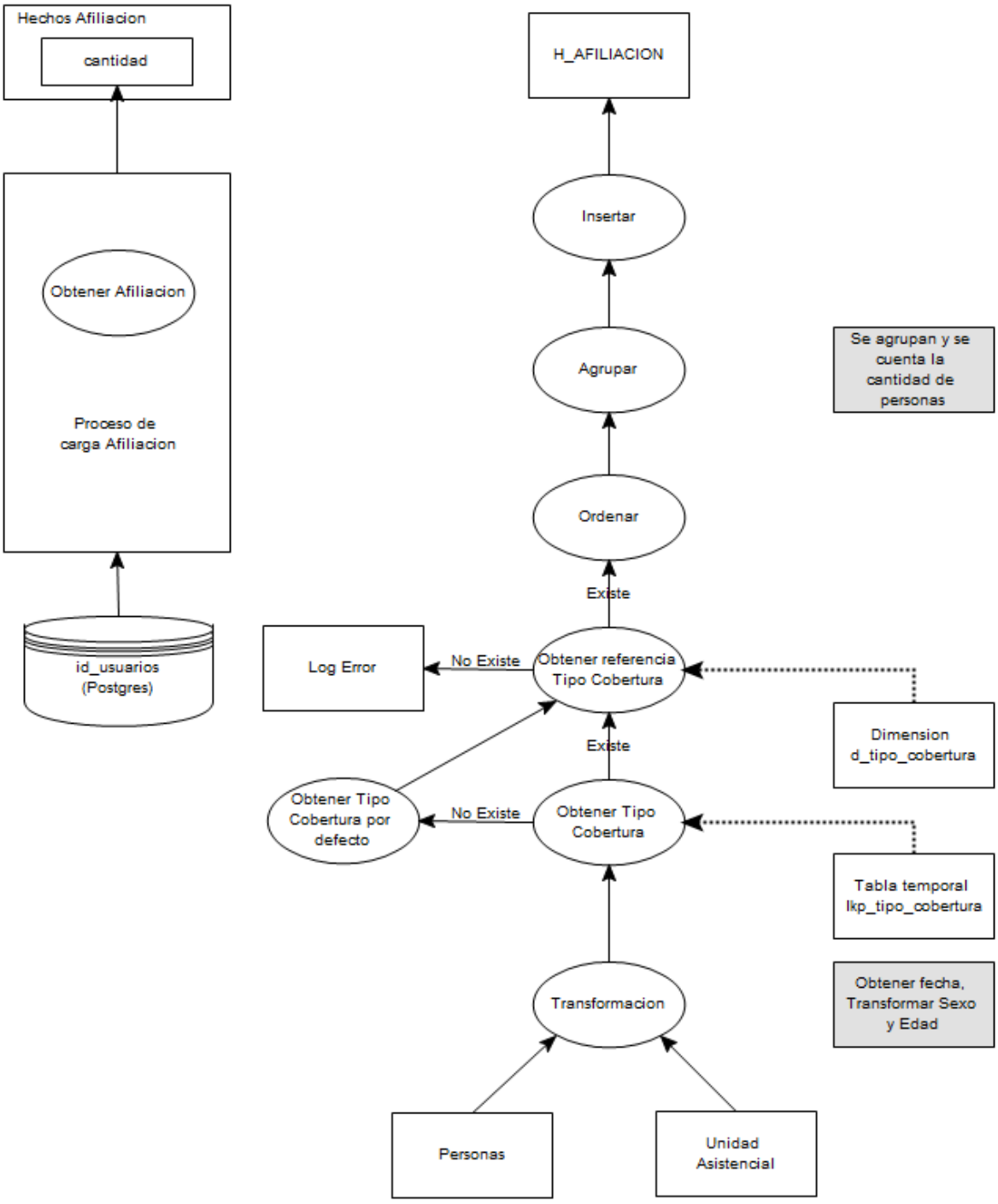


Figura 5.7: Diagrama de carga de hechos

5.4. Consultas Ad Hoc

Pentaho provee el servicio de consultas Ad Hoc a través de la librería de JSP para renderización de tablas OLAP llamado JPivot. Esta herramienta ya está integrada en el BI server de Pentaho y a su vez provee accesos directamente desde el portal de la suite como lo muestra la imagen siguiente 5.8.

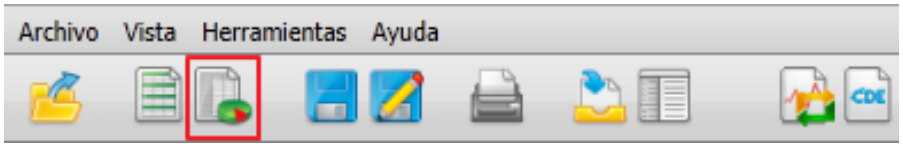


Figura 5.8: Acceso desde el BI Server a las tablas dinámicas de análisis

Utilizando esta herramienta se pueden crear tablas dinámicas, usando los cubos publicados en el Data Warehouse y modificando las dimensiones en tiempo real, usando el mouse o modificando las consultas MDX que controlan las consultas sobre el cubo.
La siguiente imagen [5.9] muestra un ejemplo de un reporte de análisis OLAP creado con Pentaho.

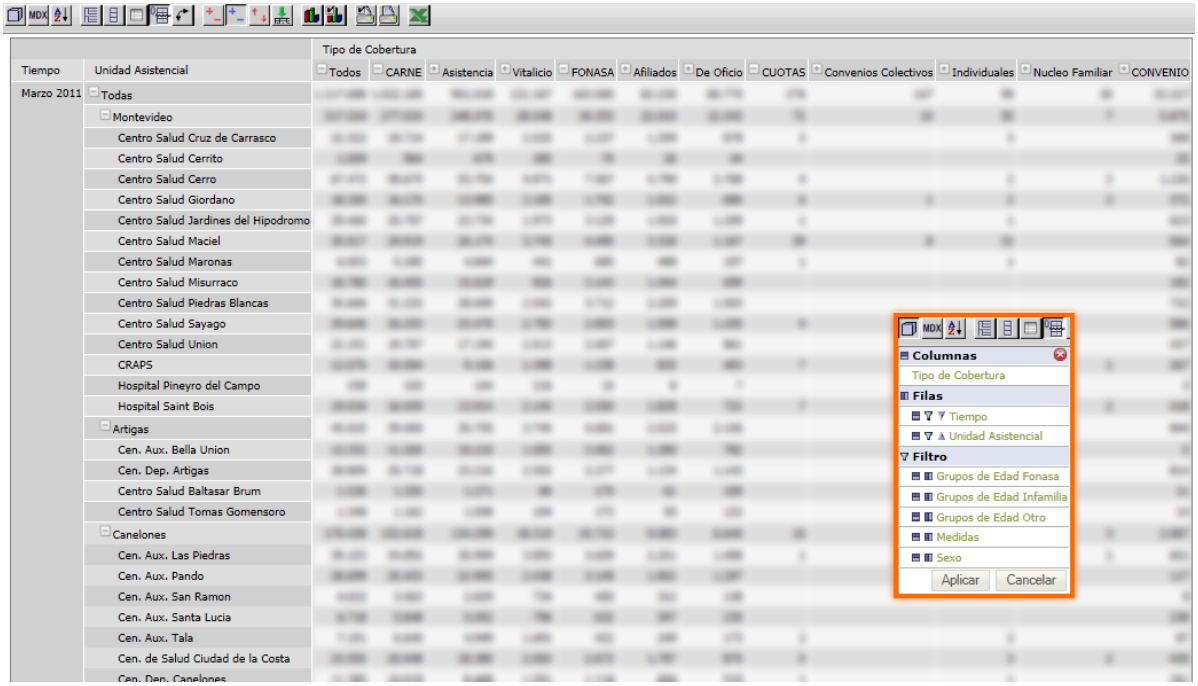


Figura 5.9: Ejemplo de análisis OLAP sobre Pentaho

5.5. Reportes

Uno de los problemas a resolver dentro de este proyecto fue la generación de reportes a partir de los datos registrados, de manera que se generaron varios reportes útiles para la dirección de ASSE que son los que se muestran a continuación:

Reportes relativos a los grupos de edad INFAMILIA:

- Total de afiliados a ASSE por grupo de edad INFAMILIA, según departamento
- Total de afiliados a ASSE por grupo de edad INFAMILIA, según departamento y unidad asistencial

Reportes relativos a los grupos de edad FONASA (Cápitas):

- Total de afiliados a ASSE, por tipo de cobertura según sexo y grupo de edad FONASA
- Total de afiliados a ASSE por tipo de cobertura, según departamento
- Total de afiliados a ASSE por tipo de cobertura, según departamento y unidad asistencial

5.5.1. Herramientas para la elaboración de reportes

Pentaho permite consultar los datos de distintas fuentes y ponerlos a disposición de los usuarios en diferentes formatos (HTML, PDF, Microsoft Excel, Rich Text format y texto plano). Además de generar reportes, las herramientas de Pentaho permiten distribuir los reportes a los usuarios interesados, como así también publicarlos en un portal donde cada usuario puede observar la información que necesita. Para poder realizar estas tareas Pentaho utiliza una serie de componentes integrados, los cuales se detallan a continuación:

JFreeReport: Es un motor de reportes basado en tecnología Java y es el encargado de generar los reportes a partir de la definición del mismo (archivo XML).

Report Designer: Es una herramienta que permite generar definiciones de reportes a través de una interfaz gráfica, en la cual se diseña de manera visual el reporte. La definición generada luego es interpretada por JFreeReport. Permite generar en pocos pasos una definición de reportes utilizando herramientas de creación paso a paso o *wizzards*. A diferencia de la anterior, solo permite definir algunos parámetros que generan rápidamente un reporte, pero limita ciertas acciones. Muy útil para generar reportes rápidamente. Al igual que la anterior, la definición generada luego es interpretada por JFreeReport.

Ad hoc Reporting: Es otra herramienta que permite generar reportes en base a una serie de parámetros, pero a diferencia de la anterior se ejecuta vía WEB en el portal de Pentaho.

Pentaho Design Studio: Es una herramienta basada en eclipse que permite hacer ejecuciones controladas de los reportes y crear secuencias de acciónes (Actions Sequences).

Pentaho Report Designer es el componente de Pentaho utilizado para la creación de los reportes. La versión que utilizada es Pentaho Reporting 3.8.0 CE Stable [13][18].
Para la creación de cada reporte al principio utilizamos Report Design Wizard donde definimos algunos parámetros necesarios para generar el reporte y se siguieron los siguientes pasos:

- Se seleccionó el diseño general o template a utilizar en nuestro reporte.
- Se configuró la fuente de datos en la cual se agregan las conexiones y demás configuraciones.
- En el Pentaho Analysis Schema File se asigna el archivo XML conteniendo el diseño del cubo a utilizar.
- Se asigna la conexión a la base de datos donde se encuentra cargado el DW
- De la vista de análisis se obtiene la consulta MDX utilizada para obtener los datos a presentar.
- Se seleccionan las columnas a mostrar.

Luego de culminar con el asistente proporcionado por la herramienta, se pasa a la etapa de personalización del reporte. En este paso se agregó cabezal y pie de página, así como la definición de la orientación que utilizada para que el reporte esté presentado en forma horizontal con el fin de que entren toda las columnas.

Otro de los puntos a definir para la generación de reportes es el de los parámetros requeridos para su generación. En este caso corresponde a la lista de los meses ya cargados en el DW. Para ello se debe definir una conexión que obtenga dicha lista mediante una consulta SQL ordenada por año y mes.

El ultimo paso corresponde a la publicación del reporte dentro del servidor para que pueda ser visualizado.

5.5.2. Estructura de los reportes

Una definición general de un reporte consiste en un conjunto de secciones que definen la disposición y contenido de la información dentro de éste. Estas secciones son:

- Cabecera y pie del reporte: Es impreso al comienzo y fin del reporte respectivamente.
- Cabecera y pie de página: Son impresos al comienzo y fin de cada página respectivamente.
- Cabecera y pie de grupo: Son impresos al comienzo y fin de cada grupo respectivamente. Un grupo, generalmente, contiene el nombre de una columna y su valor.
- Ítems o detalles: Contienen los datos obtenidos de la consulta. Estos valores se repiten tantas veces como filas devueltas en la consulta.
- Sección de funciones y expresiones: Permiten realizar cálculos de valores. Por ejemplo se podría calcular el total de un valor que pertenece a un grupo.

Los cinco reportes están agrupados en dos grandes grupos, por un lado los reportes relacionados con el grupo de edades INFAMILIA, y por otro lado los que tienen relación con el grupo de edades Fonasa. Estos son:

- Reportes relativos a los grupos de edad INFAMILIA
  - Total de afiliados a ASSE por grupo de edad INFAMILIA y según departamento
  - Total de afiliados a ASSE por grupo de edad INFAMILIA, según departamento y unidad asistencial
- Reportes relativos a los grupos de edad FONASA (Cápitas)
  - Total de afiliados a ASSE, por tipo de cobertura según sexo y grupo de edad FONASA
  - Total de afiliados a ASSE por tipo de cobertura, según departamento
  - Total de afiliados a ASSE por tipo de cobertura, según departamento y unidad asistencial

A continuación se detalla la estructura general de uno de los reportes relativos al grupo de edad Fonasa Cápitas presentados junto con un detalle de las filas y las columnas que éste posee. El resto de los reportes serán detallados en al anexo 6.4.1.

Total de afiliados a ASSE, por tipo de cobertura según sexo y grupo de edad FONASA

1. En este reporte las columnas corresponden a la cantidad de afiliados asociado a cada tipo de cobertura
   - Total
   - Tipos de cobertura
     - Carné
       - Asistencia

        - ◇ Bonificado 1
        - ◇ Bonificado 2
        - ◇ Carné de asistencia gratuito
        - ◇ Materno - Infantil
      - Vitalicio
    - FONASA
      - Afiliados
      - De Oficio
    - Cuotas
      - Convenios colectivos
      - Individuales
      - Núcleo familiar
    - Convenio
      - Panes
      - Sanidad policial

2. Las filas corresponden primeramente a los totales de afiliados discriminados por sexo
   - Total
   - Sexo Masculino
   - Sexo Femenino
   - Sexo no indicado
     - A su vez para cada categorización anterior correspondiente al sexo del afiliado se agrupa por los diferentes grupo de edades FONASA incluyendo un grupo de edad "No indicado"
       - Grupos de edades
         - ◇ Total
         - ◇ Menores de 1
         - ◇ de 1 a 4
         - ◇ de 5 a 14
         - ◇ de 15 a 19
         - ◇ de 20 a 44
         - ◇ de 45 a 64
         - ◇ de 65 a 74
         - ◇ 75 y más
         - ◇ No indicado

5.5.3. Visualización

La figura 5.10 muestra la visualización del reporte descripto anteriormente de afiliaciones en la herramienta.

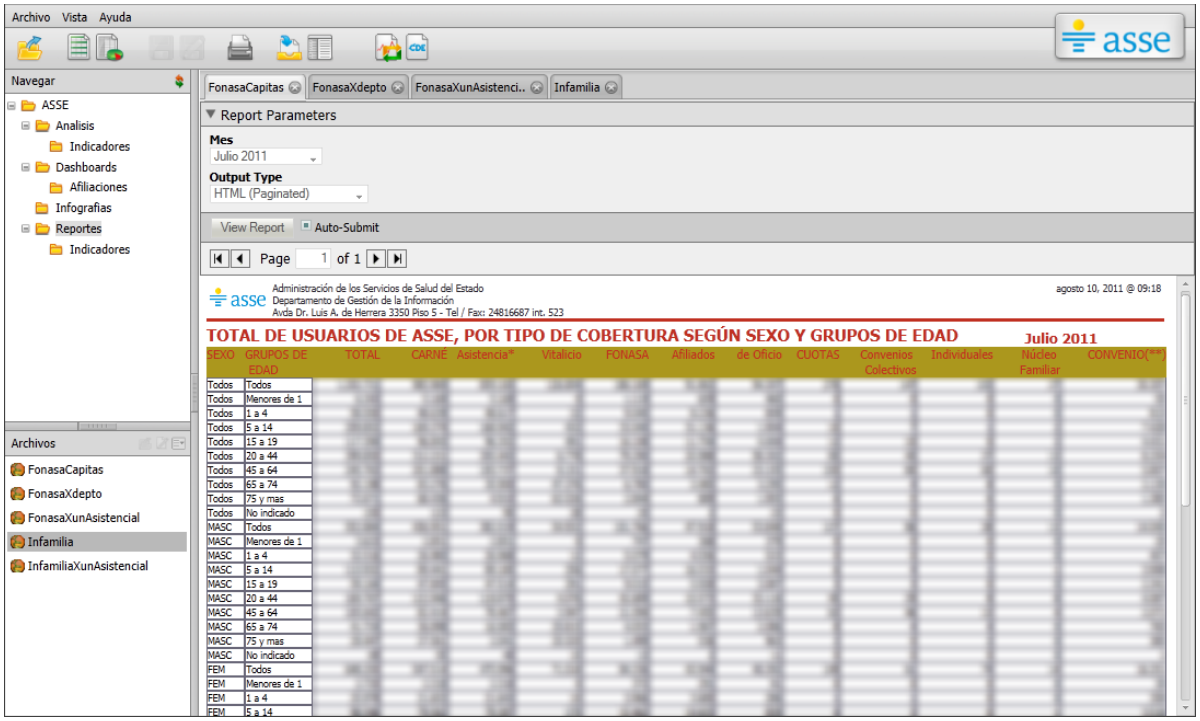


Figura 5.10: Figura que muestra la visualización de un reporte.

5.6. Cuadros de Mando

Por fuera de los requerimientos iniciales del proyecto, pero de gran importancia en un sistema de BI, se encuentran los cuadros de mando (de aquí en adelante los llamaremos Dashboards para utilizar la misma nomenclatura que Pentaho). Se decidió profundizar en la realización de dashboards que permitan de manera dinámica obtener la información que es manejada por los cubos de información y desplegarla al usuario de forma amigable e interactiva, tratando de posibilitar evaluaciones y análisis más detallados.

5.6.1. Herramientas

La plataforma de Pentaho provee de la tecnología necesaria para desarrollar cuadros de mando de forma amigable para el usuario. El Community Dashboard Framework (CDF)

es la herramienta que permite la ejecución de los cuadros de mando y posee las siguientes características:

- No requiere del desarrollo extra de páginas de JSP para la creación de los Dashboards generados.
- Los archivos que generan los componentes de los Dashboards son fácilmente manejables dentro de la estructura de archivos y carpetas de la aplicación.
- La creación de Dashboards dinámicos con Ajax integrado.
- Aplica el sistema de seguridad sobre los componentes a través del modelo de seguridad de Pentaho que es usado en toda la plataforma.
- Integración con OpenStreetMaps

Para la creación de los cuadros de mandos se requiere de muy poca programación JavaScript, además de contar con auto-generadores de objetos de formulario, una separación clara entre el diseño HTML y la definición de los componentes utilizados así como también la capacidad de gestionar eventos de cambios entre los objetos participantes solamente definiendo listeners sobre las variables utilizadas. Para la creación de estos dashboards Pentaho provee una herramienta que facilita la creación de los dashboards que se llama Pentaho Design Studio (ver [17]), pero que igualmente es bastante más compleja en términos de uso que CDE, el cual será explicado posteriormente.

5.6.2. Estructura General

Los dashboards de CDF son páginas web que contienen áreas llamadas componentes, en donde se visualiza la información obtenida desde los cubos (pueden ser gráficas, tablas, mapas, etc.).

Para definir un dashboard se deben crear dos archivos, uno de definición y otro de estructura, que son los que generan la página en donde se encuentran los componentes. El primero de ellos es un archivo en formato *XML* con extensión .xcdf y que se utiliza para identificar el dashboard dentro del repositorio de la solución de Pentaho. El segundo es un archivo *HTML* que es el que define la estructura visual y los diferentes componentes que lo conformarán.

Cuando el usuario a través del navegador realiza una petición de un dashboard al servidor, éste intenta localizar el fichero de definición asociado al dashboard solicitado. Luego cada componente ejecuta sus archivos de acciones (con extensión *.xaction*) utilizando la API de CDF, donde la misma es la encargada de comunicarse con el gestor de cubos y de retornar los resultados de vuelta a los componentes. Luego de ésto la librería de dashboards genera cada uno de los componentes y finalmente como resultado se obtiene una página que es renderizada en el navegador.

La figura 5.11 muestra un esquema de la arquitectura lógica y la interacción de las entidades que utiliza CDF para generar un dashboard.

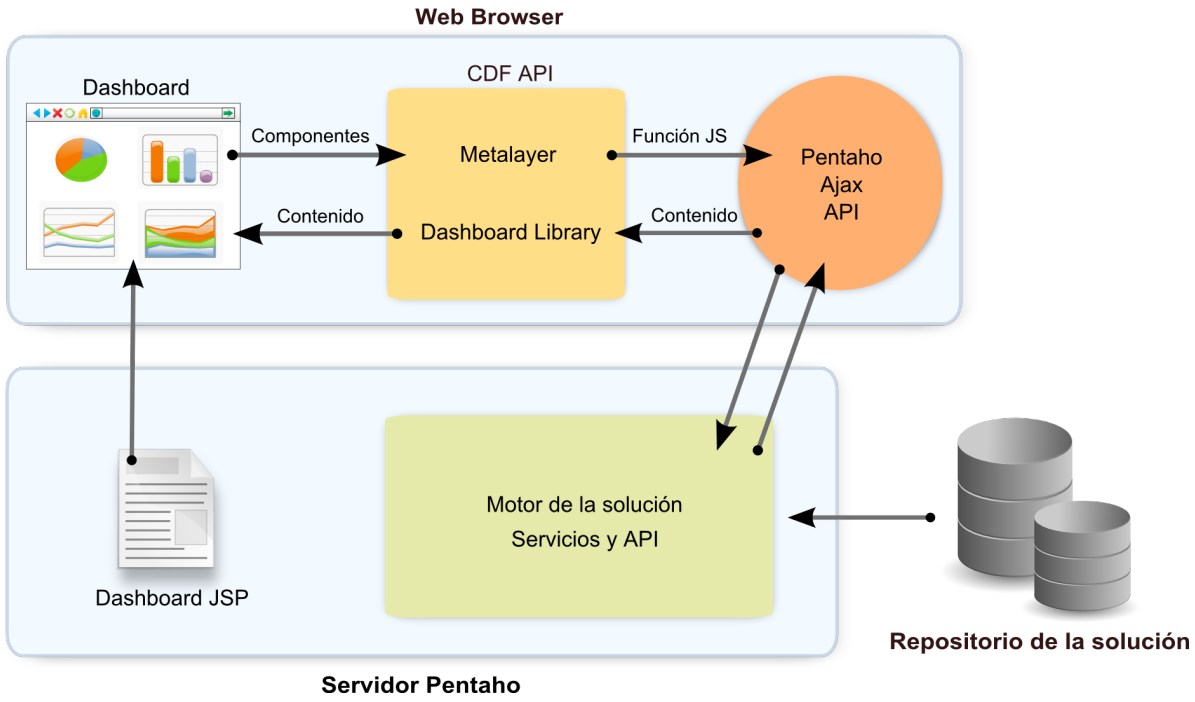


Figura 5.11: El siguiente es un esquema simplificado de la arquitectura de un dashboard en Pentaho.

5.6.3. Desarrollo(CDE)

Para el desarrollo de los dashboards en la aplicación fue utilizado el Community Dashboard Editor for Pentaho (CDE [9]), que es una herramienta externa a Pentaho, de caraterísticas Free/Open Source, y que provee integración con la herramienta a través de un plugin. El editor instalado es un desarrollo de integración de tecnologías Open Source externo y que, utilizando los componentes provistos por el CDF, además de otros componentes (como el Community Chart Component (CCC) y el Community Data Access Component (CDA)) permite el diseño y la generación de los cuadros de mando en tiempo real, desde la aplicación y de forma más amigable para el usuario, disminuyendo los tiempos de desarrollo de los artefactos buscados y con una mejor calidad visual.

Utiliza otros componentes como CCC que es una librería de gráficas para Pentaho y el CDA es un sub-projecto de la suite que permite la recuperación de datos en varios formatos desde la plataforma de BI utilizando llamadas simples a URLs (crea una API para las consultas que pueden ser accedidas remotamente) y obteniendo datos desde distintas fuentes, como pueden ser, consultas SQL, consultas MDX, Meta-datos, etc y devolviendo los datos requeridos en formatos estándares como JSON, XML, etc.

El CDE permite la generación de únicamente un archivo de configuración que es interpretado en tiempo de ejecución por el editor y que ejecuta un Dashboard en tiempo real luego de la compilación y previa publicación de dicho archivo. El archivo de configuración mencionado es un archivo en formato JSON (JavaScript Object Notation) que contiene todos los componentes, interacciones, triggers, listeners y componentes visuales que permiten la creación del Dashboard.

El CDE utiliza una estructura de capas de MVC (Modelo-Vista-Controlador) para la creación de los Dashboards que en nuestro caso lo denotamos por capas llamadas de Datos, de Visualización y de Controles, las cuales se definen de la siguiente manera:

- Capa de visualización: Es la capa encargada de definir los componentes visuales que tendrá el Dashboards, así como también de la estructura de HTML que conformará el diseño del mismo.

- Capa de controles: Es la capa encargada de definir los controles, variables y componentes dinámicos que formaran parte del Dashboard así como también la interacción entre cada uno de ellos. Aquí participan los componentes del CCC mencionado anteriormente.

- Capa de datos: Es la capa encargada de configurar los diferentes Datasources que serán utilizados para cargar los componentes del Dashboard. Básicamente lo que se genera es un archivo de paquete de CDA con las consultas necesarias para los datos requeridos.

En las siguientes imágenes [5.12 , 5.13 y 5.14] se muestran los editores de las diferentes capas que conforman el CDF-DE

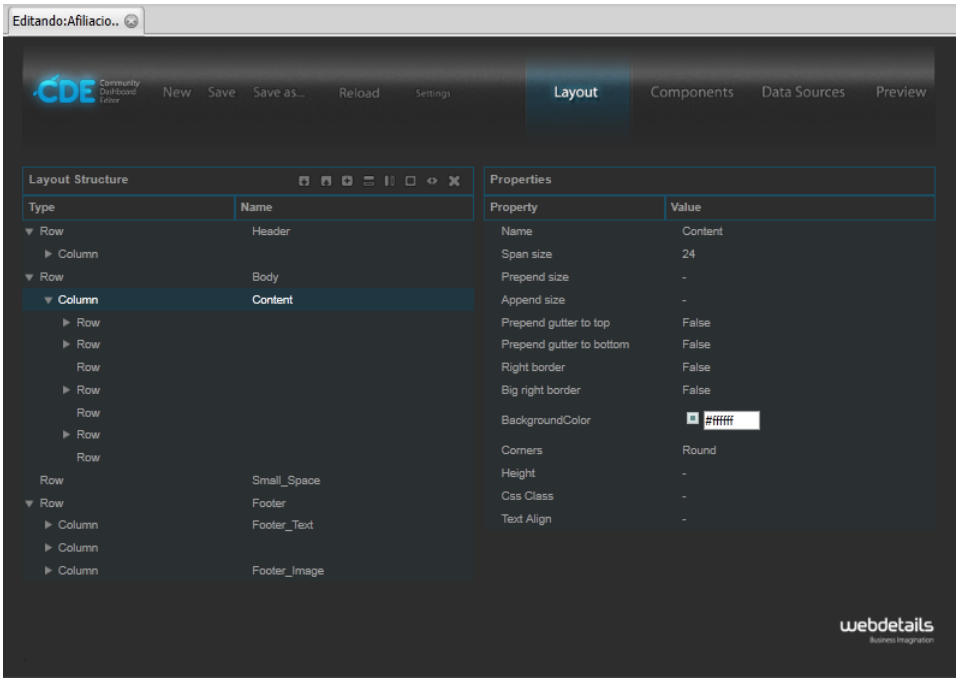


Figura 5.12: Figura que muestra la capa de Visualización.

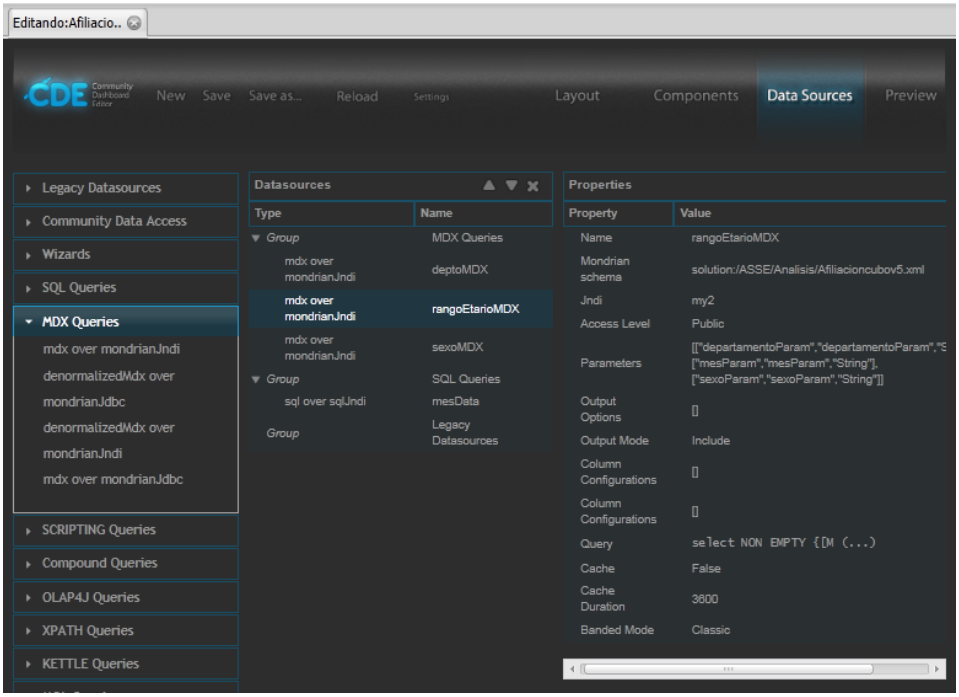


Figura 5.13: Figura que muestra la capa de Controles.

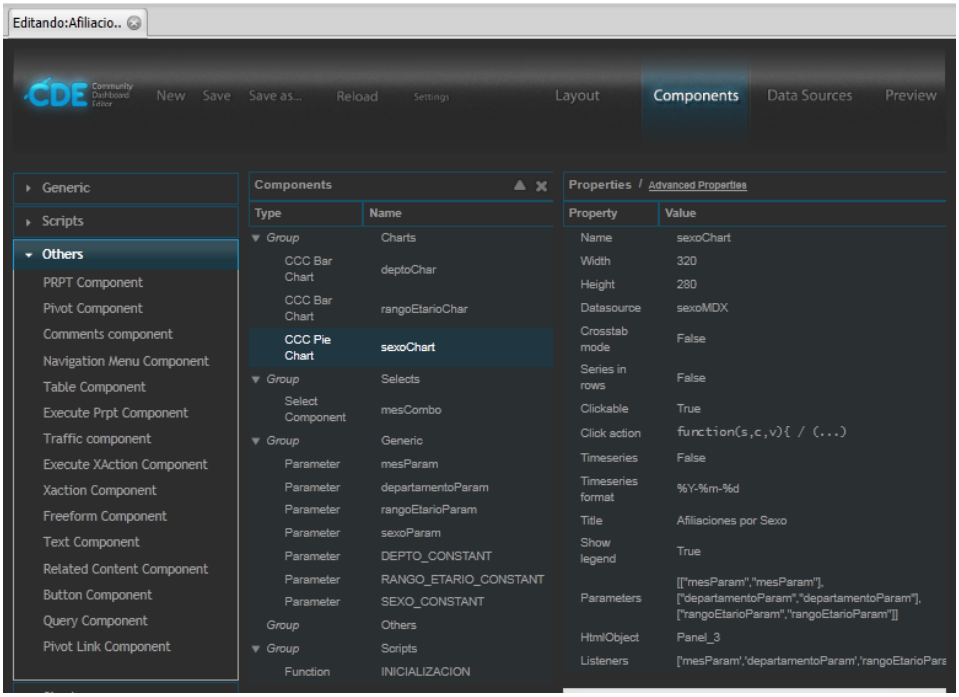


Figura 5.14: Figura que muestra la capa de Datos.

5.6.4. Creación de dashboards

Se toma como ejemplo de creación una gráfica del Dashboard de afiliaciones para explicar los pasos realizados en el desarrollo de dichos objetos sobre Pentaho.

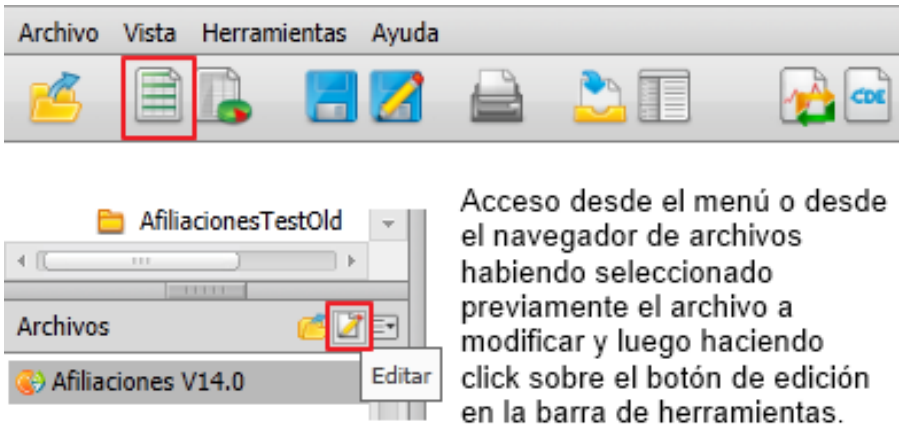


Figura 5.15: Figura que muestra los accesos al editor de dashboards.

El editor mencionado anteriormente, el cual se puede acceder desde la pantalla principal desde el botón "Nuevo Dashboard", posee varias secciones que se pasan a detallar.

En la parte superior izquierda del editor se aprecian los controles de administración del mismo. Como se menciona anteriormente, el editor realiza una separación de tres capas de los componentes que forman un Dashboard. Como se aprecia en la parte superior derecha de la ventana notamos que existen tres accesos a cada una de estas secciones que son:

- **Layouts:** Capa de estructura, en la cual se define el esqueleto de la página que contendrá los componentes que formaran el cuadro de mando. En esta capa se definen los estilos y recursos necesarios para la customización.
- **Components:** Capa de control, la cual define y provee los componentes visuales e interactivos requeridos para el Dashboard, como por ejemplo los Combos usados para la interacción con el usuario, los distintos componentes de gráficas a usar, botones, links, etc.
- **Datasources:** Capa de modelos de datos, la cual define los recursos de datos necesarios para cargan los componentes definidos en la capa anterior. Estos recursos pueden ser consultas SQL, consultas MDX sobre cubos de Mondrian, etc.

Para mostrar la creación de los dashboards tomamos como ejemplo de desarrollo de uno simple con una única gráfica. Se tomará en cuenta cada una de las capas definidas por la herramienta para explicar paso a paso los cambios realizados. Primero, se define la estructura de layout para el dashboard, la cual se realiza desde la sección de *Layouts* denotada anteriormente. La manera de estructurar la página es básicamente con bloques de contenido, en el caso del editor, éstos son traducidos como *filas* y *columnas* que en tiempo de ejecución son transformados en elementos DIV de HTML y en donde se van a cargar los componentes que conforman el dashboard. Cada fila definida en dicho layout deberá tener un identificador que identifica a la estructura dentro de los componentes que lo utilizarán para realizar su posterior renderizado.

Luego de tener definida la estructura donde se van a colocar cada uno de los componentes se prosigue creando los componentes que van a participar en el dashboard (como componentes se hace referencia a gráficas, combo boxes, imágenes, etc). Para crear una gráfica se accede a la pestaña de *Components* dentro del editor y se definen algunos atributos requeridos que son:

- Nombre de la gráfica (necesario también para referencias entre componentes)
- Datasource (Orígenes de datos que se utilizaran para cargar la gráfica con datos)
- Titulo (Titulo de la gráfica)
- Show legend (true/false, habilita o des habilita la leyenda de la gráfica)
- Parameters (parámetros usados para pasaje de parámetros)
- HtmlObject (componente visual en el cual va a ser renderizada la gráfica)

- Listeners (al definirse permiten escuchar por eventos externos a la gráfica, ej: que la gráfica se actualice en caso de que una variable sea modificada)

Notar que para cargar el datasource dentro de la gráfica se lo debe tener definido previamente, para realizar esto se debe cambiar a la pestaña de *Data Sources* y ahí crear un nuevo datasource (en este caso se utilizan consultas de tipo *MDX over mondrian Jndi*) que contienen los siguientes atributos definidos:

- Nombre (Necesario para poder referenciarlo desde la gráfica)
- Mondrian schema (Esquema del cubo sobre el cual se va a realizar la consulta)
- Jndi (Nombre de configuración de fuente de datos)
- Parameters (Parámetros en el caso de ser requeridos)
- Query (Consulta MDX sobre el cubo)

Ahora si se puede realizar la previsualización del dashboard creado accediendo a la pestaña *Preview*.

5.6.5. Publicación y Visualización

Luego de verificar que el resultado mostrado por la previsualización es correcto, se guarda el dashboard dentro de la carpeta en la cual se va a acceder para su visualización.

Para configurar la seguridad sobre el archivo, se hace click derecho dentro del link al dashboard en la ventana de navegación de archivos y en *Propiedades >Compartir* se accede a la lista de usuarios y roles a los cuales se les puede marcar los permisos de acceso que permiten la visualización y edición selectiva. Para visualizar el dashboard, luego de la previa publicación se debe actualizar el cache del repositorio de Pentaho desde la aplicación y acceder desde el árbol de carpetas al dashboard requerido. La figura 5.16 se puede ver como es la visualización de un reporte interactivo generado con la herramienta instalada.

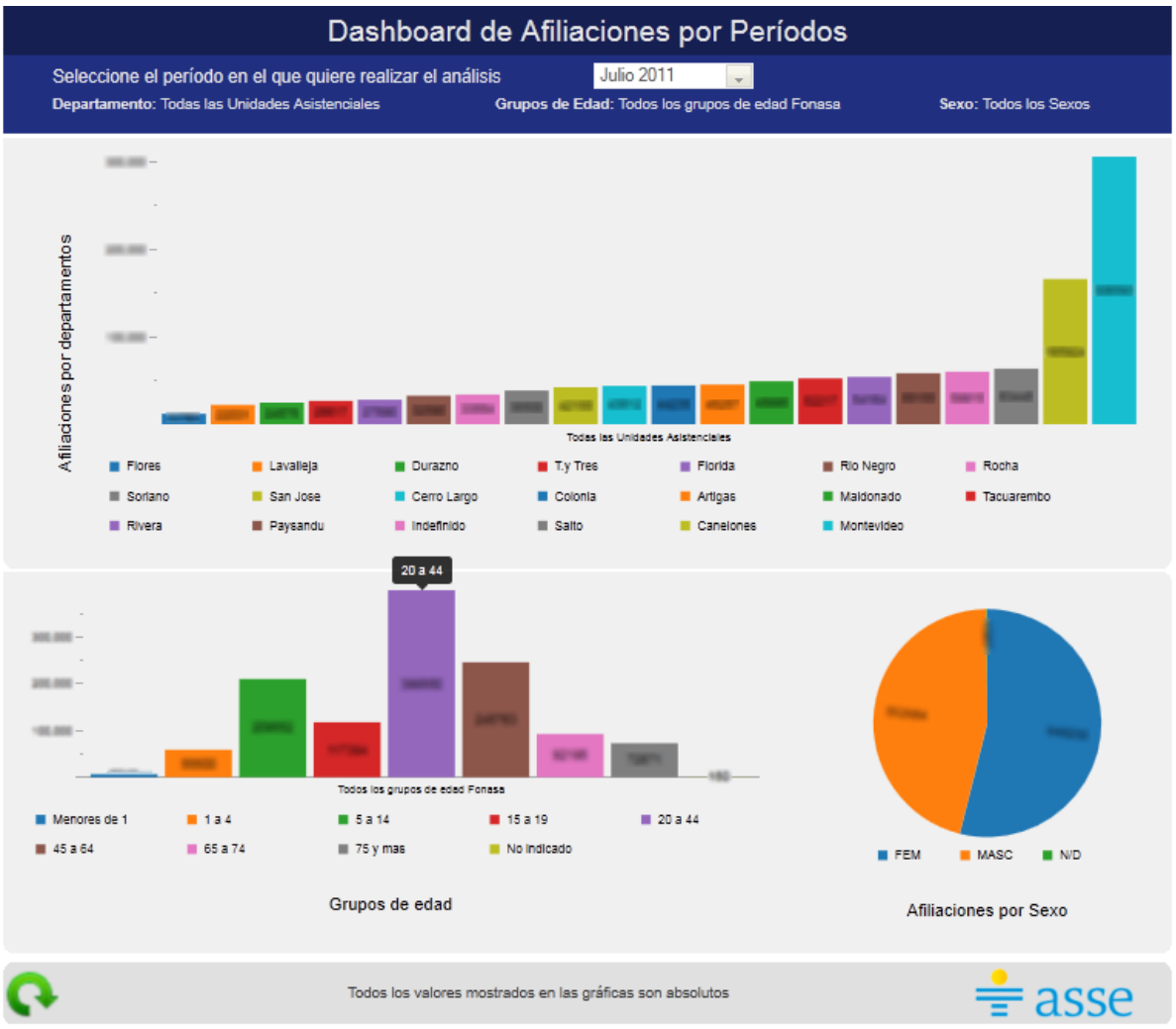


Figura 5.16: Captura del cuadro de mando de afiliaciones en la aplicación.

Cabe destacar que el dashboard creado en el proyecto permite la interacción entre todos los componentes creados, por ejemplo, al seleccionar un valor en una tabla se actualizarían las demás restantes filtrando por el dato que fue seleccionado anteriormente y actualizandose en el momento. Las gráficas son cargadas con consultas MDX directamente sobre los cubos creados al principio.

5.7. Infografía

Pentaho permite también la creación de otros medios para transmitir información visualmente como por ejemplo las infografías, las cuales son cargadas directamente con la información del DW. Pentaho no provee una herramienta que permita la edición

visual de las infografías sino que deben crearse de forma manual desde las imágenes, la estructura y la carga de los datos requeridos utilizando secuencias de acciones. A modo de ejemplo se muestra el desarrollo de la infografía de *Afiliaciones por departamento* creado para este proyecto.

5.7.1. Estructura general

Las infografías se componen básicamente de dos elementos básicos que son, una fuente de datos de la cual se obtienen los datos que van a ser mostrados al usuario, una imagen en la cual aplicar o graficar los datos obtenidos en el paso anterior.

5.7.2. Creación y visualización de infografías

Para este ejemplo se tendrán dos archivos que conforman la infografía y que se explican a continuación:

- **Archivo de Imagen Vectorial:** En este caso se utilizó la figura del mapa de uruguay definida como archivo vectorial en formato SVG (Scalable Vector Graphics) que es una especificación para describir gráficos vectoriales bidimensionales en formato XML. En este proyecto se usó la herramienta libre Inkscape Illustrator [23] para la generación del mapa de uruguay utilizado para el mencionado gráfico.
- **Archivo de Secuencia de Acción (Xaction):** En este archivo de ejecución de Secuencia de Acción (utilizados por Pentaho para definir objetos interactivos) se definen las configuraciones de las acciones que se realizan para realizar la carga y presentación.

El requerimiento principal para poder dejar accesible el gráfico es que dicha figura debe contener identificadores dentro del XML en los objetos gráficos que se quieren utilizar de manera de poder hacer referencia a través del archivo de secuencia de acciones y poder realizar las modificaciones necesarias para presentar la información. El siguiente ejemplo marca el atributo que debe ser agregado para poder hacer los objetos reverenciables, en este caso se le agrega el identificador *tacuarembo* al tag de path que define gráficamente al departamento de Tacuarembó.

```
<path
   ...
   id="tacuarembo"
   ...
/>
```

El archivo de secuencia de acción es un XML que contiene las estructuras y las acciones que ejecuta la infografía. Se enumeran debajo las acciones definidas en dicho archivo:

1. Tag de titulo de la infografía

2. Tag de versión

3. Tag de logging-level: Define el tipo de error que se logueará a través de log4j.

4. Tag de Autor, Descripción e ícono.

5. Tag de inputs, en este caso no es necesario.

6. Tag de outputs, define el formato de salida de nuestro componente.

7. Tag de recursos, un ejemplo se define debajo

```
<resources>
    <template>
      <solution-file>
        <location>mapa_uruguay.svg</location>
        <mime-type>text/xml</mime-type>
      </solution-file>
    </template>
    <catalog>
      <file>
        <location>/usr/Pentaho/Data/xml/Afiliacioncubov5.xml</location>
        <mime-type>text/plain</mime-type>
      </file>
    </catalog>
  </resources>
```

8. Tag de acciones, éste es el tag principal dado que en él se definen todas y cada una de las acciones necesarias para la ejecución, por ejemplo se declara la consulta OLAP sobre el cubo de donde se va a obtener la información y se crea el javascript que colorea los departamentos dependiendo del porcentaje de afiliados obtenidos por departamentos.

La siguiente es un ejemplo de ejecución de la infografía generada con los archivos detallados anteriormente.

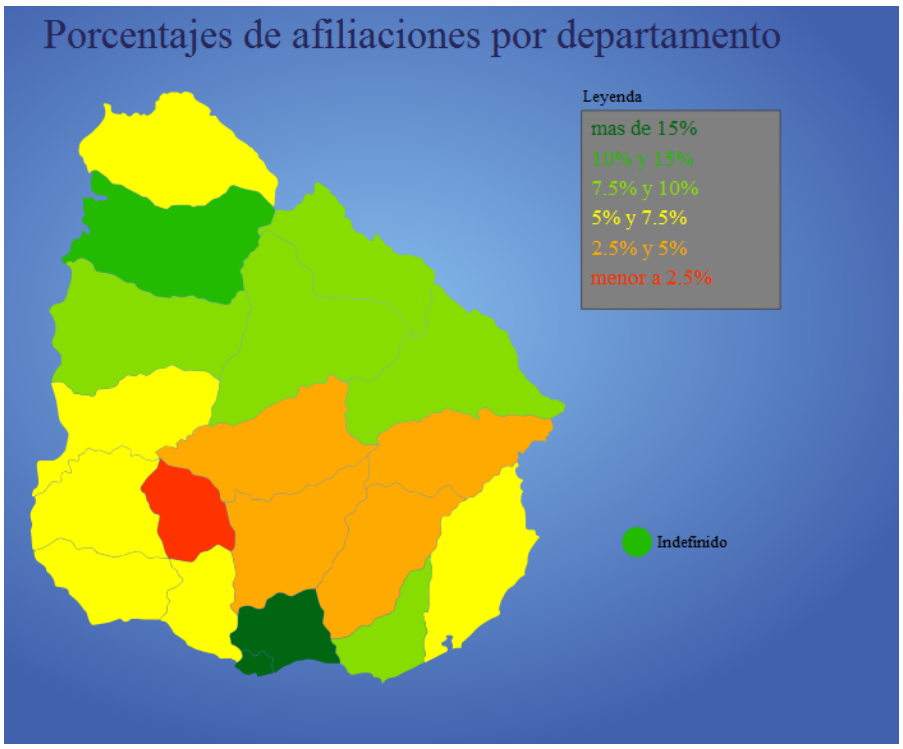


Figura 5.17: Infografía de Afiliaciones por departamento.

Para publicar la infografía dentro de la aplicación se deben copiar los archivos dentro de una carpeta en la estructura de carpetas de la solución, actualizar el cache del repositorio de Pentaho desde la aplicación y acceder desde el árbol de carpetas a la infografía.

5.8. Consola de Administración

La consola de administración de Pentaho es el administrador central de Pentaho. Los componentes agregados en la consola simplifican muchas de las tareas administrativas de la plataforma (que en versiones anteriores se hacían de forma manual) como ser el manejo de usuarios y permisos, la gestión de las conexiones de datos, la programación de trabajos de ejecución y la gestión de servicios.

La seguridad en los componentes de Pentaho están basados en Listas de control de acceso o ACL. Un ACL por defecto es aplicado a todos los directorios contenidos de la solución al momento del inicio del servicio, luego son administrados desde la consola. Cuando un usuario solicita una operación sobre un objeto en un modelo de seguridad basada en ACL el sistema comprueba en la lista que exista una entrada aplicable para decidir si la operación solicitada está autorizada sobre ese objeto y por ese usuario. Es posible integrar Pentaho con Single Sign-On usando CAS (Central Authentication

Service) o sino también con Microsoft Active Directory. Como nota se agrega que Pentaho BI soporta todos los métodos de autenticación soportados por el Spring Secutiry Framework.

Cabe destacar que para dicha aplicación se debe de tener permisos de administrador para poder levar a cabo las tareas administrativas sobre los objetos.

5.8.1. Administración de Usuarios y Permisos

En la consola de administración es donde se gestionan los usuarios y los roles de cada uno de ellos. Antes de la configuración de los roles se debe planificar como se van a realizar los accesos a los objetos, se debe determinar que roles van a tener sentido dentro de la aplicación, si al integrar con otro sistema que también usa ACL tratar de realizar un mapeo entre los roles heredados, se debe determinar cuales roles van a tener acceso a las URL particulares y determinar si algunos de ellos van a tener capacidad de ejecutar secuencias de acción en el repositorio de soluciones.

La siguiente figura [5.18] muestra en la consola de administración la pestaña de gestión de usuarios en la aplicación.

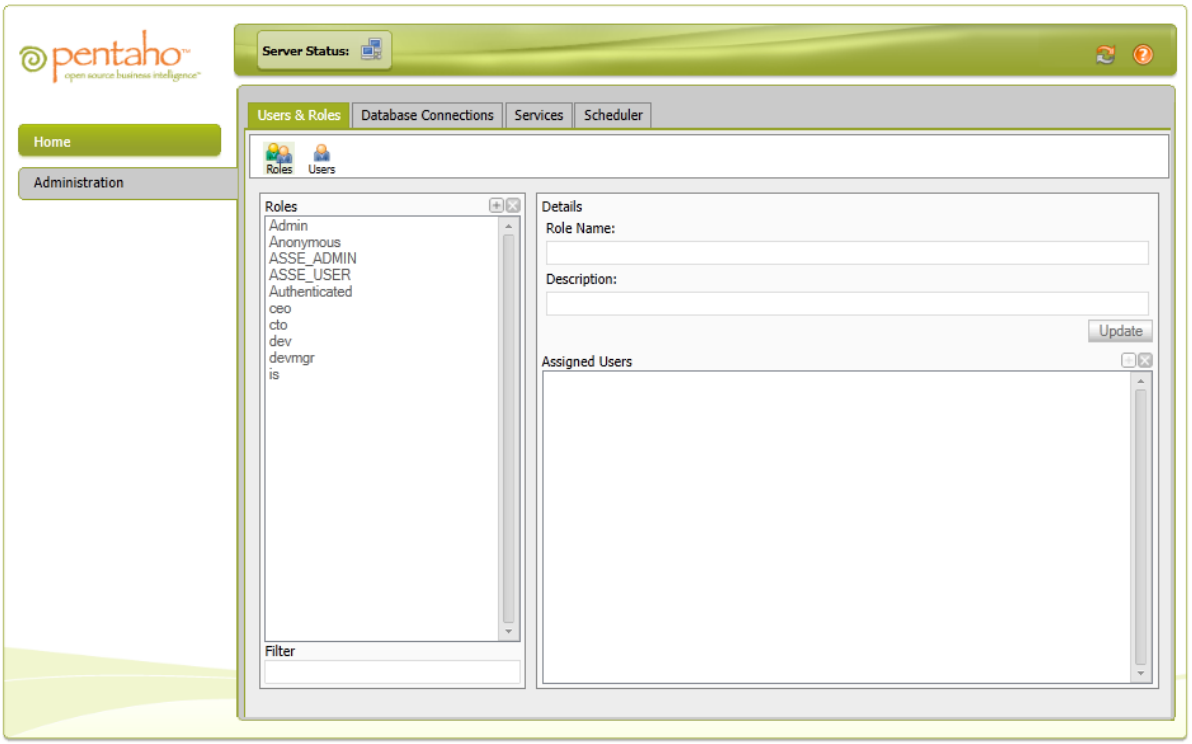


Figura 5.18: Pestaña de gestión de usuarios en la consola de administración

5.8.2. Administración de orígenes de datos

La consola de administración permite administrar también las distintas fuentes de datos o *Data Sources* a las cuales queremos acceder para obtener los datos necesarios para nuestros cubos. En la pestaña de "Data Sources" se pueden administrar los mismos y básicamente los datos requeridos para definir un origen de datos son: un nombre de clase de JDBC para el driver de la base de datos, una URL de data source (nombre de servidor, numero de puerto, nombre de la base de datos) y también el usuario y password de acceso.

5.8.3. Herramientas administrativas

La consola de administración posee algunas herramientas administrativas que son útiles al momento de desarrollar y agregar nuevas funcionalidades a la plataforma.El siguiente listado muestra algunas de las acciones posibles.

Refresh Solution Repository: Actualiza el repositorio de archivos de la solución cuando se agrega manualmente o editan archivos en la solución principal dentro de el sistema local de archivos.

Refresh BI Server: En esta herramienta se encuentran varias acciones como por ejemplo la actualización de configuraciones del sistema, la de las variables de sistema, la de los modelos de metadatos y la eliminación de la del cache del servidor Mondrian, esta última por ejemplo elimina los datos de las consultas utilizadas por los cubos.

Content Repository Elimina los archivos creados en el repositorio de contenido que se encuentran en la ruta /pentaho-solution/system/content y que tienen más de 180 días. (para cambiar el número de días, se debe editar el archivo de solución clean_repository.xaction que se encuentra en /pentaho-solution/admin)

La siguiente figura 5.19 muestra los disparadores de acciones de los servicios administrativos.

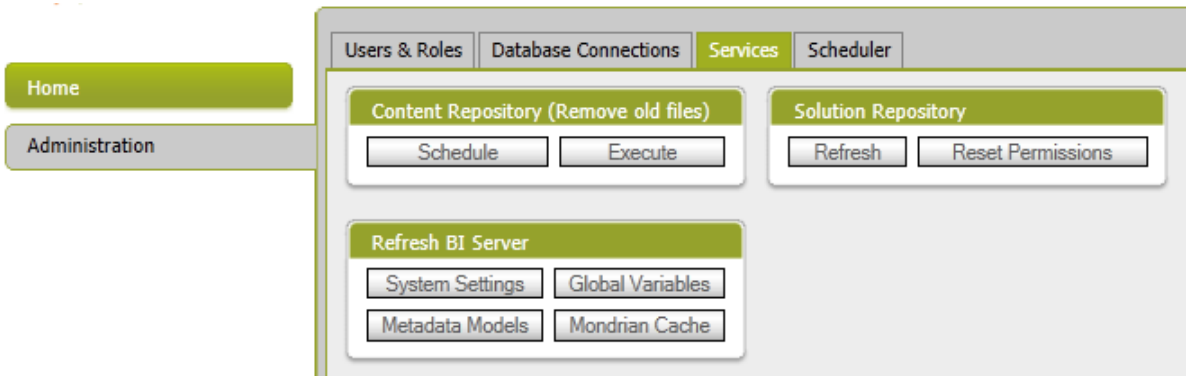


Figura 5.19: Botones de acciones de las herramientas administrativas

5.9. BI Server

EL Pentaho BI Server [15] es el componente principal que provee Pentaho Community [11] a los usuarios y es una aplicación Web J2EE que corre sobre un servidor de apache embebido dentro del paquete y que incluye los sub-paquetes de Pentaho Reporting, Pentaho Analysis y Pentaho Dashboard Framework. En esta aplicación los datos son publicados, almacenados, compartidos y administrados por todos los usuarios.

Al ingresar a la aplicación web se diferencian 4 sectores en los cuales la información va a ser desplegada. La figura [5.20] muestra cada uno de los sectores que pasamos a explicar a continuación. El sector marcado con el numero uno es el escritorio de trabajo y es en donde se visualizarán los objetos creados por la herramienta (reportes, dashboards, OLAP, etc). El sector marcado con el numero dos es el explorador de carpetas de la plataforma y que se muestran con una estructura de árbol. El tercer sector es una vista de los archivos que se encuentran en las carpetas del segundo sector. El sector marcado con el cuatro es el menú de la aplicación y desde el mismo se puede realizar la administración básica de los componentes, así como también acciones de limpiezas de caché, cambio de idioma de la solución, acceder a la documentación, etc.

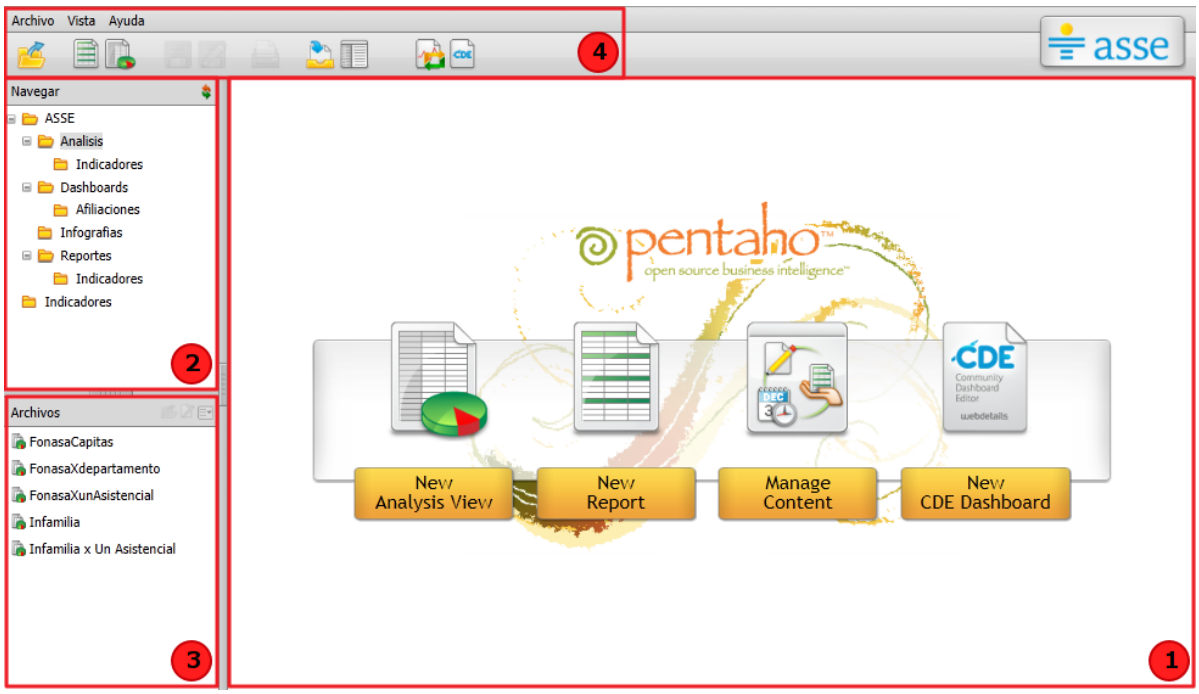


Figura 5.20: Distintos sectores del ambiente de trabajo del BI Server

5.10. Tiempos del proyecto

Esta sección está centrada en la evolución temporal de este proyecto y se intenta ilustrar el tiempo dedicado a cada una de las etapas que lo conformaron.

Al comienzo del proyecto se establecieron las principales fases y los hitos a alcanzar pero las actividades que se realizaron no se planificaron antes de comenzar el proyecto, sino que se fueron definiendo durante el transcurso del mismo.

Cabe destacar que una de las particularidades de este proyecto de grado es que fue realizado de forma remota ya que por motivos laborales teníamos dificultades para trabajar directamente en ASSE, lo que generó retrasos de tiempo en algunas etapas del mismo. La causa de esto fue particularmente por tratarse del manejo de un volumen importante de datos personales y privados de los afiliados a ASSE lo cual impidió que se extrajeran a un ambiente de desarrollo fuera del recinto de ASSE. Además de que para las pruebas de ETL y de respuestas era necesario contar con una gran cantidad de datos para poder evaluar performance y posibles errores, que lo hacía aún más difícil de replicar en un ambiente local de desarrollo.

Al inicio del proyecto realizó un estudio del estado del arte de las tecnologías

involucradas como ser DW, Diseño Multidimensional y BI, entre otras.

Al mismo tiempo se evaluaron los primeros requerimientos brindados por ASSE que llevaron al análisis de los orígenes de datos que se utilizarían para generar los reportes gerenciales que se produjeron. Para ésto se comenzaron a analizar las bases de datos de las cuales debíamos obtener la información. Se debió realizar una ingeniería inversa sobre las bases de datos, revisando las estructuras de las mismas así como también los datos que contenían y se comenzó a crear un diagrama de entidad relación de las partes que estuvieron involucradas en la implementación del DW dado que no se contaba a priori con diagramas de entidad que nos facilitaran la tarea. Se realizaron entrevistas con las personas encargadas del mantenimiento del Sistema de Gestión de Afiliados con el fin de entender mejor el dominio en le que nos encontrábamos trabajando.

En el mismo momento que se estudiaba el estado del arte de las tecnologías se comenzaron a realizar las distintas evaluaciones sobre las diferentes plataformas que fueron investigadas, luego se procedió a realizar las comparaciones descriptas en el Capítulo 3 y finalmente se comenzó a implementar los dos prototipos también ahí mencionados. En esta etapa se termino de decidir cual iba a ser la plataforma a ser implementada durante el resto del proyecto.

Luego de decidir cual iba a ser a la plataforma a utilizarse se paso a instalar y configurar cada una de las herramientas comprendidas en dicha plataforma sobre los servidores virtualizados alocados dentro de la institución y así poder tener las bases necesarias al momento de crear los artefactos que iban a ser utilizados en el DW.

Tres etapas que estuvieron muy relacionadas fueron las etapas de Calidad de Datos, Diseño Multidimensional y la etapa de ETL. Éstas fueron las más importantes del proyecto dado que fueron las etapas en las que se sentaron las bases de las estructuras a utilizarse, se extrajeron, depuraron y cargaron los datos que formarían parte de los reportes y se guardaron en el DW utilizando los modelos de datos multidimensionales creados. Durante gran parte del proyecto se realizaron trabajos de mejoras de calidad (etapa de Calidad de datos) de los datos los cuales requieren un tiempo importante de desarrollo dado que se deben detectar y corregir los problemas de calidad antes de ser ingresados al DW en el caso de ser posible. La etapa de Diseño Multidimensional requirió más tiempo del supuesto dado que, al no poseer un conocimiento profundo del tema los cubos creados se fueron optimizando y mejorando con el tiempo y a la vez se fueron modificando y adaptando para generar otros reportes con las mismas estructuras multidimensionales, soportando en algunos casos la información de otros requerimientos. La etapa de ETL dependió mucho de la evolución de la etapa de diseño dado que para poder cargar los datos en los cubos fue requerido tener definidas las estructuras previamente. Estos procesos de carga necesitaban muchas horas para correr dado que manejaban una gran cantidad de información, en el caso de que fallaran o no cumplieran los cambios que se pretendían deberían modificarse y ejecutarse

nuevamente generando atrasos de tiempo no deseados.

La etapa de OLAP y reportes utiliza los cubos de información cargados de la etapa de ETL. Los reportes fueron los componentes de esta etapa que llevaron más tiempo de desarrollo dado que se utilizó una herramienta de la plataforma para generar las estructuras que era visual y a la cual accedimos de forma remota.

La etapa de dashboards e infografías fue una etapa totalmente aparte de las demás y a pesar de que no fue tan extensa en comparación con la de los reportes requirió de un tiempo extenso de desarrollo, debido a que no era intuitiva la creación de los componentes necesarios para la creación de los dashboards y no se tenía documentación extensa provista por el proveedor del framework.

Otra de las etapas del proyecto que llevó un tiempo considerable fue la documentación, la cual no solo sirve para la asignatura en sí sino que también sirve como una de las formas de traspaso de conocimiento a los interesados y afiliados dentro de ASSE.

En la figura 5.21 se detallan las estimaciones de tiempo requerido para la realización de las distintas etapas y tareas comprendidas dentro del proyecto. La posición de cada tarea a lo largo del tiempo hace que se puedan identificar las relaciones de precedencia entre las mismas y más aún la duración en esfuerzo de cada una de las mismas.

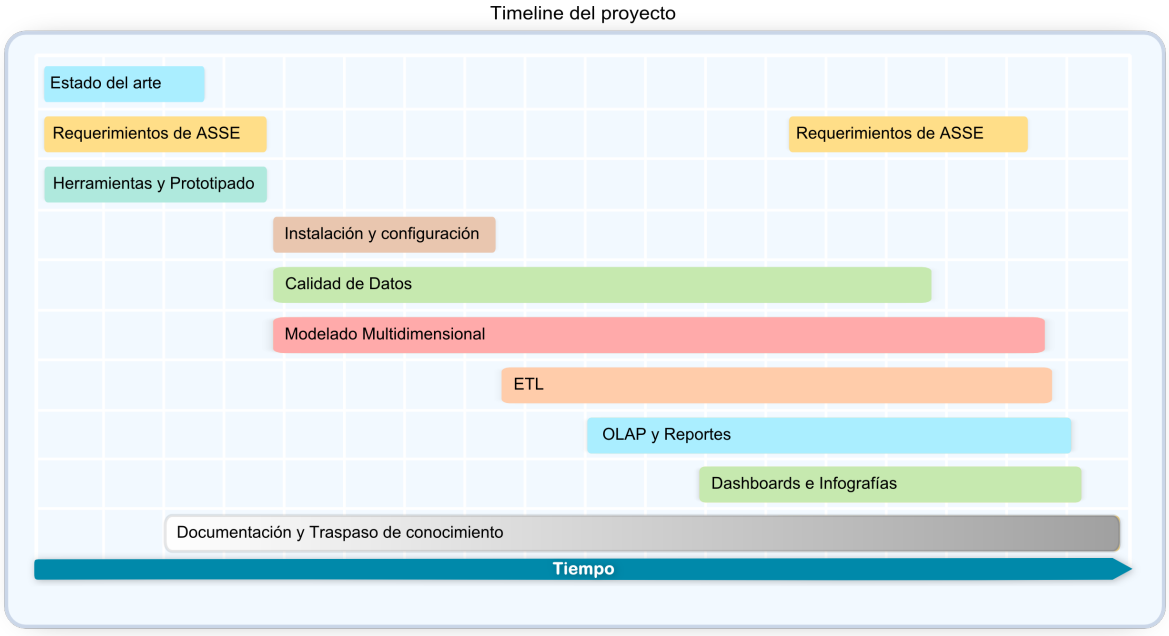


Figura 5.21: Distribución del tiempo requerido para la realización de las distintas etapas del proyecto

Se estima que es posible ejecutar este tipo de proyectos en un plazo menor al transcurrido si se toman en cuenta los siguientes puntos:

- Trabajo desde el lugar de la implantación
- Contando con la documentación y apoyo técnico disponible por parte de los interesados
- Contando con un ambiente de trabajo estable y con los recursos de hardware necesarios

Para concluir este capítulo, se lograron implementar los procesos de carga, consultas Ad Hoc, reportes, cuadros de mando, infografías, automatización de la carga y la configuración de los usuarios en la consola de administración.

Capítulo 6

Conclusiones y Trabajo a futuro

En este capítulo se presentan las conclusiones del proyecto. Para ello se exponen algunas de las dificultades encontradas en el desarrollo del trabajo, los resultados, aportes obtenidos respecto a los objetivos planteados y posibles trabajos a futuro.

6.1. Resultados Obtenidos

Se realizó un estudio del estado del arte en soluciones de Data Warehouse (DW) Free Open Source Software (FOSS), tanto en base de datos como en herramientas de reportes que sirven como base para la elección de herramientas en futuros proyectos de similares características. Por otro lado se instalaron las herramientas recomendadas, base de datos, software de Extracción Transformación y Carga (ETL) y software de Business Intelligence (BI). Se diseñaron los cubos que permiten obtener los reportes definidos y se desarrollaron los programas de ETL necesarios para la carga del DW. Se implementó el DW solicitado y se programó la actualización periódica del mismo de forma automatizada con manejo de errores ante eventuales fallas del proceso. Se implementaron los reportes solicitados con la herramienta de reportes seleccionada y se realizaron cuadros de mando e infografías. Se destaca como resultado obtenido la importante experiencia adquirida en temas de business intelligence a lo largo del proyecto. Por último, el proyecto fue seleccionado por la Asociación de Informáticos del Uruguay (AsIAP) para ser presentado en las Jornadas Informáticas de la Administración Pública (JIAP) 2011, representando para nosotros una experiencia interesante ya que se notó gran interés sobre el proyecto por las personas que concurrieron a la presentación.

6.2. Dificultades durante el proceso

Dentro de las dificultades encontradas durante el proceso se destaca el incremento continuo de los requisitos de los usuarios una vez que las soluciones son implementadas.

Otro aspecto importante a destacar es que no se pudo trabajar en un ambiente de desarrollo con datos generados por nosotros mismos. Esto se debe a que era necesario poder trabajar directamente con los datos que íbamos a cargar dentro de nuestro DW. Esto permitió enfrentarnos directamente tanto con los problemas de calidad de los datos así como también con el importante volumen de estos mismos. La opción de tener estos datos en un ambiente de desarrollo local no era viable, ya que estos datos son confidenciales. Se nos brindó entonces la posibilidad de trabajar en forma remota, trayendo los problemas de caídas de los servidores remotos, así como también bajos tiempos de respuesta para realizar las tareas.

La calidad de los datos es otra de las dificultades que se tuvieron que enfrentar. Para que puedan ser utilizados los datos deben ser lo mas confiables que se pueda, y estar disponibles y completos en la mayor parte del tiempo posible.

Por otro lado el hecho de trabajar con una herramienta open source trae algunos problemas como por ejemplo la falta de documentación y soporte, los constantes cambios de versión en las diferentes herramientas de la plataforma y la existencia de bugs en los componentes utilizados son algunos de los inconvenientes del software libre.

6.3. Conclusiones Finales

A pesar de las dificultades mencionadas en la sección anterior una de conclusiones más importantes luego de finalizado el proyecto es que se puede implementar una solución de BI FOSS en el tiempo que abarca un proyecto de grado y en el marco de una institución estatal. La baja calidad de los datos de origen no fue un impedimento para lograr el objetivo aunque si se le tuvo que dedicar mucho tiempo a intentar mejorar lo máximo posible este aspecto.

6.4. Trabajo a Futuro

Como trabajo a futuro sería deseable poder mejorar aún más la calidad de los datos de manera que el conocimiento brindado por el DW sea aún más confiable y tenga mayor valor al momento de la toma de decisiones.

Sería interesante también extender el DW creado según las distintas necesidades de la Administración de los Servicios de Salud del Estado (ASSE), esto podría implicar la creación de nuevos cubos y reportes, así como también nuevos cuadros de mando e indicadores gerenciales que brinden aún más información institucional. Una manera de organizar los componentes de la solución podría ser mediante la utilización de Data Marts, logrando así subgrupos lógicos del DW.

Desde el punto de vista técnico y funcional se podrían incorporar modificaciones a las soluciones actuales y desarrollar nuevos componentes dentro del sistema para realizar nuevos análisis de información, como por ejemplo integración con sistemas de información geográfica y geolocalización.

Finalmente se podría incluir en el sistema interno de ASSE el acceso a los reportes generados automáticamente por el DW para la toma de decisiones.

Anexo 1: Instalación en producción

Descripción

El día 28 de enero de 2011 se realizó la instalación de un ambiente de producción donde para el cual se utilizaron los últimos paquetes estables que componen la herramienta Pentaho.

Se instalaron todos los componentes del sistema de DWOF y las herramientas de BI en un mismo sistema como un bloque de herramientas cooperativas.

Las características de ésta instalación se detallan a continuación.

Requisitos

- Ram: 1gb
- Disco: 10gb
- SO: Open Suse 11.3 64
- BD: MySql 5.1.54
- Pentaho BiServer ce 3.7.0 stable
- Pentaho Data Integration 4.1.0 stable
- Pentaho Report Designer 3.7.0 stable
- Pentaho Schema Workbench 3.2.1.13885 (inestable por defecto)
- Design Studio 3.7.0
- Pentaho Dashboard Editor CDE bundle 1.0 RC3

Arquitectura

La arquitectura del sistema se separó en dos capas, la capa de datos estaría localizada en los servidores de ASSE, los cuales cuentan con una base de datos MySQL ya instalada y en las cuales se creó un esquema de datos para que sea utilizado por Pentaho y otro esquema para las tablas temporales que son cargadas desde producción.

La capa de negocio y de presentación estarían contenidas en una maquina virtual, con sistema Operativo Open Suse 11.3 antes mencionado en la cual se instalaron todos los servidores requeridos por la aplicación así como también las herramientas de ETL.

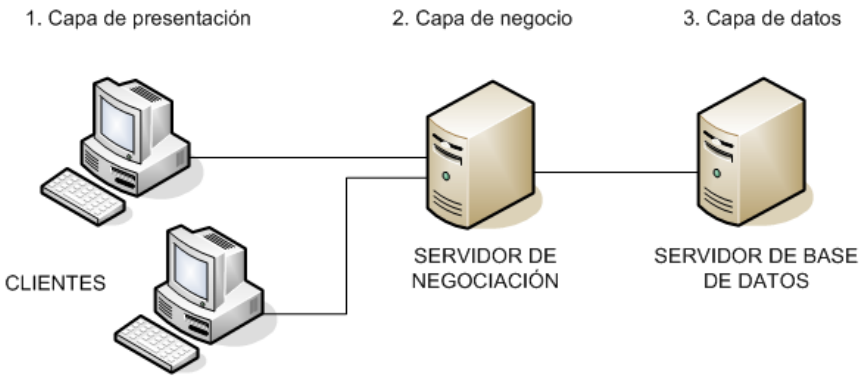


Figura 1: Ambiente típico de Data Warehouse en el cual, de diferentes orígenes de datos se almacenan en un ambiente centralizado

Luego de varias pruebas y problemas, se decidió cambiar la arquitectura, jerarquizando la capa de negocio en dos capas diferenciadas según sus funcionalidades. Para ésto se copiaron las herramientas de ETL (Pentaho Data Integration) a una maquina separada de la de los servidores, de manera de poder contener en un mismo sistema las bases de datos temporales para el cubo y las herramientas de ETL por un lado y por el otro los servidores de Mondrian para OLAP, el servidor de la consola de administración para la gestión de los data sources y el servidor de BI de JPivot junto con los dashboards y reportes. Ésta separación logra un mayor nivel de abstracción de los componentes funcionales, permitiendo que dicha separación lógica aumente la seguridad de la estructura de todo el sistema, dado que si por algún motivo, uno de los sistemas se vuelve inestable o se bloquea, se puede continuar trabajando con las otras herramientas hasta que se logre recomponerlo.

La segunda arquitectura quedó de la siguiente manera:

Maquina 1

- Ram: 1.5gb
- Disco: 10gb
- SO: Open Suse 11.3 64
- Pentaho BiServer ce 3.7.0 stable (descargado desde [15])
- Pentaho Data Integration 4.1.0 stable (descargado desde [16])
- Pentaho Report Designer 3.7.0 stable (descargado desde [13])
- Design Studio 3.7.0 (descargado desde [17])
- Pentaho Dashboard Framework CDF bundle 3.2 (descargado desde [10])
- Pentaho Dashboard Editor CDE bundle 1.0 RC3 (descargado desde [8])

Maquina 2

- Ram: 1gb
- Disco: 10gb
- SO: Ubuntu 10.04 Lucid Lynx 64
- MySql 5.5
- Pentaho Schema Workbench 3.2.1.13885 RC1 (inestable, descargado desde [14])

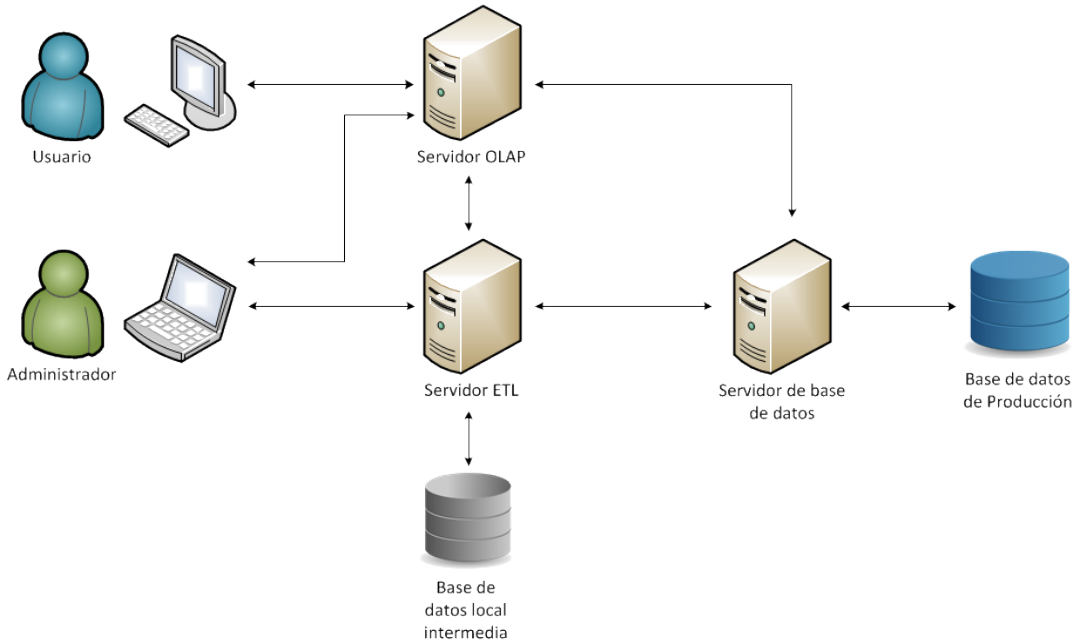


Figura 2: Configuración de la arquitectura luego de la separación de la capa de negocio

Luego de tener varios problemas que no pudieron ser definidos en el Hardware que soportaba la arquitectura distribuida, se decidió volver a la arquitectura inicial relocalizando en un hardware único todos los componentes requeridos.

Instalación

Se detalla a continuación los pasos para realizar las instalaciones en los respectivos servidores.

Descarga de Archivos

Se descargan los paquetes mencionados anteriormente disponibles desde el sitio de Pentaho [11]

Configuraciones iniciales

Configuraciones del esquema workbench

Descargar schema workbench "Schema Workbench 3.2.1-stable (2010-12-01)"desde el sitio de descarga [14]. En la ruta /usr/Pentaho"se debe descomprimir el contenido de "psw-ce-3.2.1.13885.tar.gz"mediante el comando :

```
tar -zxvf psw-ce-3.2.1.13885.tar.gz
```

Para la configuración se debe ejecutar el Pentaho Administration console para así poder configurar la conexión al cubo. Esto permite tener a disposición una conexión para poder así asignarla al cubo al momento de hacer la publicación.

```
cd /usr/Pentaho/administration-console
./start-pac.sh
```

Posteriormente se ingresa la dirección http://localhost:8099 en el navegador con usuario: admin y password: password, lo cual nos permitirá ingresar a la consola de administración.
En Administration ->Database Connections, se configuran los datasource. Se creará uno para la base de datos MySQL del cubo, lo cual permite armar el cubo. La configuración de la conexión se pasa a detallar:

```
Name: my2
Driver Class: com.mysql.jdbc.Driver
User Name: root
password: nomaschucu
url: jdbc:mysql://my2:3306/dw
```

Antes de ejecutar el Schema Workbench, se debe copiar el driver de MySQL para así configurar una conexión. Como el Pentaho ya cuenta con él se lo puede copiar desde el administration-console.

```
cp /usr/Pentaho/administratiion-console/jdbc/mysql-connector-java-5.1.10.jar  /usr/Pe
```

Una vez copiado se ejecuta el Schema Workbench:

```
cd /usr/Pentaho/schema-workbench
./workbench.sh
```

En Options ->Connection, se configura la conexión a la base de datos que será usada para poder armar el cubo. Para armar el cubo, recomendamos la lectura del documento "Pasos para crear Cubos con Mondrian Schema Workbench"[7].

Para la publicación de un cubo se debe configurar el password en Pentaho para publicar schemas.

Se debe editar el archivo publisher_config.xml que se encuentra en: biserver-ce/pentaho-solutions/system.

Se busca:

```
<publisher-config>
   <publisher-password></publisher-password>
</publisher-config>
```

En ese lugar se define la contraseña que será requerida al momento de publicar el cubo. Otro requerimiento al momento de publicar el cubo es un usuario y contraseña de Pentaho (joe/password por defecto).

Si se desea hacer la publicación del cubo manualmente se debe editar el archivo datasource.xml que se encuentra en biserver-ce/pentaho-solution/system/olap Dentro del tag <Catalogs>se agregan los catálogos correspondientes a los cubos que queremos publicar. Se crea un tag <Catalog>que incluye una propiedad "name"la cual define el nombre del cubo que será utilizado desde el Pentaho (<Catalog name="Nombre del Cubo»). Dentro de este tag se definen 2 tags suplementarios:

- <DataSourceInfo>el cual contiene las propiedades separadas por ";":
  - Provider=mondrian , el cual corresponde al motor de OLAP

- DataSource=my2, el cual corresponde al nombre de la conexión anteriormente definida
- EnableXmla=False, para que no sea un esquema expuesto a un webservice XMLA

- <Definition>que contiene la propiedad siguiente:
  - solution, que corresponde el lugar donde se encuentra el cubo

m

La sintaxis se describe a continuación mediante un ejemplo :

```
<Catalog name="Nombre del Cubo">
   <DataSourceInfo>
      Provider=mondrian;DataSource=my2;EnableXmla=False
   </DataSourceInfo>
   <Definition>
      solution:/ASSE/Analisis/miCuboMondrian.xml
   </Definition>
</Catalog>
```

Instalación del Pentaho Data Integration

Los pasos a seguir para la instalación de la herramientas de Data Integration son los siguientes. Luego de descargar el Kettle "Stable build of Kettle 4.1 released (2010-11-30)"desde el sitio de descarga [16] ,se procede a crear la ruta /usr/Pentahoz descomprimir dentro de este el contenido del "pdi-ce-4.1.0-stable.tar.gz"descargado con el comando :

```
tar -zxvf pdi-ce-4.1.0-stable.tar.gz
```

Luego copiar las siguientes carpetas a "/usr/Pentaho/data-integration" con sus correspondientes contenidos :

```
- main
- temporal
- dimensiones
- logCarga
```

Crear la carpeta .kettle en el directorio home. Copiar dentro de ésta carpeta los archivos:

```
- kettle.properties
- shared.xml
```

El archivo "shared.xml" contiene las conexiones globales a las tres bases de datos a las que se conecta el proceso de carga, éstas son:

- Base de datos postgres que corresponde a la fuente de datos (base id_usuarios)
- Base de datos postgres temporal (dw)
- Base donde se encuentra el data warehouse sobre mysql de nombre dw

En caso de querer cambiar alguna de las conexiones a las bases de datos utilizadas en los procesos de ETL, se debe modificar el archivo "shared.xml".
Otra forma consiste en ejecutar spoon.sh en "/usr/Pentaho/data-integration". En la pestaña "View"de la vista del arbol de carpetas, bajo la carpeta "Trabajos.ªl desplegar la carpeta Çonexiones a base de datos", se ven las conexiones configuradas actualmente. Las que están en letra negrita, corresponden a conexiones globales. Haciendo doble click a la conexión que se desea editar, se puede cambiar los parámetros de conexión deseados. Se cuenta con un botón "Probarçon el fin de verificar que todo está bien configurado. Para finalizar con los cambios, se debe "Validar"para luego cerrar el Spoon.

Para programar el cron se deberá copiar dentro de "/usr/Pentaho" el archivo "cron_data_updater_executer.sh" el cual corresponde al script a ejecutar.

La programación del cron se efectúa mediante el comando :

```
sudo crontab -e
```

ya que el usuario que estamos utilizando es root Al momento de abrirse el editor se puede utilizar la configuración siguiente :

```
#m h dom mon dow   command
01 04 * * * sh /usr/Pentaho/cron_data_updater_executer.sh
```

En este caso se ejecutara el ETL de carga todos los días a las 04:01. Para cambiar la configuración se puede ver por ejemplo en el sitio de cron [19] donde se indica como configurar el cron

Para que los cambios sean tenidos en cuenta se debe reiniciar el servicio mediante: sudo service cron restart

Para evitar posibles problemas al momento de la carga de la base de datos temporal, también configuramos una tarea programada con cron para reiniciar el postgres local. Se decidió reiniciar una hora antes de que se haga la carga :

```
#m h dom mon dow   command
01 03 * * * /etc/init.d/postgresql restart
```

Publicación de esquema en Pentaho

Para permitir las publicaciones de esquemas dentro de Pentaho es necesario configurar previamente los archivos de seguridad de publicaciones del servidor. Para ésto se debe modificar el archivo del servidor de nombre 'publisher_config.xml' que se encuentra dentro de la carpeta "./biserver-ce/pentaho-solutions/system/".

Dentro de éste archivo, en el tag de "publisher_password"se debe de agregar la contraseña de acceso de publicación, como se muestra en el siguiente ejemplo.

```
<publisher-config>
<publisher-password>nuevo_password</publisher-password>
</publisher-config>
```

En el momento de realizar una publicación de un cubo desde el schema-workbench, la herramienta pedirá una cuenta de acceso a Pentaho además del password de publicación que se acaba de setear, se lo ingresa y se tiene una ventana nueva en donde se pueden navegar los archivos, aquí se busca el directorio del datasource necesario y se continúa. Luego de ésto el esquema queda publicado en el sistema.

Mantenimiento para efectuar cargas en forma manual

Como verificar una carga anteriormente realizada

Antes de efectuar una carga se debe revisar que esta no fue previamente hecha en forma exitosa. Para dicha acción, solo basta con ejecutar una consulta en la base de datos local PostgresSQL que contiene el log de carga, asignándole como parámetro el mes y el año en formato el yyyyMM, y el nombre del cubo que se quiere revisar. Si quisiéramos verificar la carga de noviembre del año 2010 en el cubo AFILIACIÓN ejecutaríamos la siguiente consulta:

```
select * from log_carga_dw
where lcdw_resultado = 1 and lcdw_nombre = '201011'
and lcdw_tipo = 'AFILIACION';
```

Los distintos cubos son los que se detallan a continuación:

- AFILIACION
- NUMERO PARTOS
- TOTAL CONSULTAS
- GIRO DE CAMAS

En caso de que la consulta a la base de datos retorne un registro y que se quiera realizar la carga nuevamente, se debe de borrar este registro para que la verificación de carga no lo impida.

Como realizar una carga

Existen dos métodos para efectuar esta acción. El primero puede ser efectuado desde el Spoon. Para ejecutarlo:

```
cd /usr/Pentaho/data-integration
./spoon.sh
```

Se debe abrir el trabajo correspondiente al cubo que se quiera cargar. Partiendo desde la carpeta donde se encuentra el spoon.sh, los diferentes trabajos se los encuentra en:

```
(Trabajo encargado del cubo Afiliación)
./main/Main.kjb

(Trabajo encargado del resto de los cubos)
./carga\ indicadores/Traboajo\ Indicadores.kjb
```

Luego de tener el trabajo abierto en el Spoon, se debe ejecutar el trabajo con el botón verde que simboliza el Play en la parte superior izquierda de la interfaz. Se ingresa el parámetro del mes y del año de la carga que se quiere realizar asignándole a la variable MES_ESQUEMA dicho valor con el formato yyyyMM. En nuestro ejemplo deberíamos asignarle el valor 201011 para representar la carga de noviembre del año 2010. El segundo método consiste en ejecutar desde un terminal asignándole al parámetro MES_ESQUEMA el mes y el año en el formato yyyyMM como se muestra a continuación al ejemplificar el caso de la carga del cubo Afiliación para noviembre del 2010:

```
sh /usr/Pentaho/data-integration/kitchen.sh -norep
   -file=/usr/Pentaho/data-integration/main/Main.kjb
   -level=Detailed -param:MES_ESQUEMA="201010"
   >> /usr/Pentaho/data-integration/logCarga/logSalida201010.txt
```

En caso de querer efectuarla para los otros cubos hay que ejecutar :

```
sh /usr/Pentaho/data-integration/kitchen.sh -norep
   -file=/usr/Pentaho/data-integration/
   carga\ indicadores/Traboajo\ Indicadores.kjb
   -level=Detailed -param:MES_ESQUEMA="201010"
   >> /usr/Pentaho/data-integration/logCarga/logSalida201010.txt
```

Como realizar un cambio de conexión

La forma de realizar este procedimiento es a partir del gestor de conexiones del Spoon. Para realizar dicha acción:

```
cd /usr/Pentaho/data-integration
./spoon.sh
```

Abrir alguno de los trabajos para que aparezcan las conexiones. En la pestaña "View"de la vista del árbol de carpetas, bajo la carpeta "Trabajos"desplegar la carpeta Çonexiones a base de datos", ahí se verán las conexiones configuradas actualmente. Estas están en letra negrita, lo cual significa que son conexiones globales. Hacer doble click a la conexión que se desea editar, y cambiar los parámetros de conexión debidos. Se cuenta con un botón "Probarçon el fin de verificar que está todo bien configurado. Validar para luego cerrar el Spoon.

6.4.1. Configuración actual de la base de datos donde se encuentra la tabla de gestión de logs de carga

```
Host Name : 10.202.2.45
Database Name: dw
Port Number: 5432
User Name: postgres
Password: postgres
```

Problemas encontrados en las herramientas

Uno de los problemas encontrados corresponde a un error existente entre Pentaho y las versiones de Firefox posteriores a 4.0. Luego de loguearse a Pentaho BI Server aparece un popup indicando el siguiente error :

```
Error generating XUL: Failed to parse:
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="chrome://global/skin/" type="text/css"?>
<window width="400" height="275" title="Placeholder"...
```

El problema es que luego de este mensaje no se despliega el menú superior con los botones de la consola. El reporte de este error se puede seguir en la siguiente url de la página de Pentaho:

"http://jira.pentaho.com/browse/BISERVER-5282"

Se encontró una forma de solucionar esto que funciona al menos para cualquier versión de Firefox ejecutando sobre Windows:

1. Apagar BI Server y Pentaho Enteprise Console
2. En todos los archivos .xul ubicados en la carpeta ”/pentaho/server/biserver/” y en todas sus subcarpetas remplazar todas las instancias del siguiente string: ”http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xul” por: ”http://www.pentaho.org/keymaster/gatekeeper/there.is.only.xul”

3. Reiniciar BI Server y Enterprise Console

Anexo 2: Complemento de la implementación de la solución

Cubos y Analisis OLAP

En esta etapa se realizó el análisis OLAP, así como la creación de los cubos. En la figura 3 se ve reflejada la estructura del *CuboAfiliación*.

Figura 3: Representación de cubo Afiliación

Se la puede ver representada la metadata del cubo de afiliación en el siguiente XML.

```
<Schema name="AfiliacionCuboV6">
  <Cube name="AfiliacionCuboV6" cache="true" enabled="true">
    <Table name="H_AFILIACION">
    </Table>
```

```
<Dimension type="StandardDimension" foreignKey="ua_ref" highCardinality="false" name="Unidad Asistencial">
  <Hierarchy name="Unidad Asistencial" hasAll="true" allMemberName="Todas" primaryKey="ua_key">
    <Table name="D_UNIDAD_ASISTENCIAL">
    </Table>
    <Level name="Departamento" table="D_UNIDAD_ASISTENCIAL" column="ua_depto_nombre" type="String"
          uniqueMembers="false" levelType="Regular" hideMemberIf="Never">
      <Property name="Departamento" column="ua_depto_nombre" type="String">
      </Property>
    </Level>
    <Level name="Unidad Asistencial" table="D_UNIDAD_ASISTENCIAL" column="ua_nombre" type="String"
          uniqueMembers="false" levelType="Regular" hideMemberIf="Never">
      <Property name="Unidad Asistencial" column="ua_nombre" type="String">
      </Property>
    </Level>
  </Hierarchy>
</Dimension>
<Dimension type="StandardDimension" foreignKey="sexo_ref" highCardinality="false" name="Sexo">
  <Hierarchy name="Sexo" hasAll="true" allMemberName="Todos" primaryKey="sexo_key">
    <Table name="D_SEXO">
    </Table>
    <Level name="Sexo" column="sexo_nombre" type="String" uniqueMembers="false" levelType="Regular"
        hideMemberIf="Never">
    </Level>
  </Hierarchy>
</Dimension>
<Dimension type="StandardDimension" foreignKey="ge_edad" highCardinality="false"
        name="Grupo de Edad Fonasa Capitas">
  <Hierarchy name="Grupos de Edad Fonasa" hasAll="true" allMemberName="Todos" primaryKey="gef_edad">
    <Table name="D_GRUPO_EDAD_FONASA_CAP">
    </Table>
    <Level name="Grupos de Edad" column="gef_nombre" type="String" uniqueMembers="false"
        levelType="Regular" hideMemberIf="Never">
    </Level>
  </Hierarchy>
</Dimension>
<Dimension type="StandardDimension" foreignKey="ge_edad" highCardinality="false"
        name="Grupo de Edad Infamilia">
  <Hierarchy name="Grupos de Edad Infamilia" hasAll="true" allMemberName="Todos" primaryKey="gei_edad">
    <Table name="D_GRUPO_EDAD_INFAMILIA">
    </Table>
    <Level name="Grupos de Edad" column="gei_nombre" type="String" uniqueMembers="false"
        levelType="Regular" hideMemberIf="Never">
    </Level>
  </Hierarchy>
</Dimension>
<Dimension type="StandardDimension" foreignKey="ge_edad" highCardinality="false" name="Grupo de Edad Otro">
  <Hierarchy name="Grupos de Edad Otro" hasAll="true" allMemberName="Todos" primaryKey="geo_edad">
    <Table name="D_GRUPO_EDAD_OTRO">
    </Table>
    <Level name="Grupos de Edad" column="geo_nombre" type="String" uniqueMembers="false"
        levelType="Regular" hideMemberIf="Never">
    </Level>
```

```
      </Hierarchy>
    </Dimension>
    <Dimension type="StandardDimension" foreignKey="tc_ref" highCardinality="false" name="Tipo de Cobertura">
      <Hierarchy name="Tipo de Cobertura" hasAll="true" allMemberName="Todos" primaryKey="tc_key">
        <Table name="D_TIPO_COBERTURA">
        </Table>
        <Level name="Tipo de Cobertura" table="D_TIPO_COBERTURA" column="tc_tipo_cobertura" type="String"
             uniqueMembers="false" levelType="Regular" hideMemberIf="Never">
          <Property name="Tipo de Cobertura" column="tc_tipo_cobertura" type="String">
          </Property>
        </Level>
        <Level name="Tipo de Cobertura Especifica" table="D_TIPO_COBERTURA" column="tc_cobertura_especifica"
             type="String" uniqueMembers="false" levelType="Regular" hideMemberIf="Never">
          <Property name="Tipo de Cobertura Especifica" column="tc_cobertura_especifica" type="String">
          </Property>
        </Level>
        <Level name="Tipo de cobertura Especifica 2" table="D_TIPO_COBERTURA" column="tc_cobertura_especifica2"
             type="String" uniqueMembers="false" levelType="Regular" hideMemberIf="Never">
        </Level>
      </Hierarchy>
    </Dimension>
    <Dimension type="StandardDimension" foreignKey="t_ref" highCardinality="false" name="Tiempo">
      <Hierarchy name="Tiempo" hasAll="true" allMemberName="Todos" primaryKey="t_key">
        <Table name="D_TIEMPO">
        </Table>
        <Level name="Descripcion" column="t_descripcion" type="String" uniqueMembers="false"
             levelType="Regular" hideMemberIf="Never">
        </Level>
      </Hierarchy>
    </Dimension>
    <Measure name="cantidad" column="cantidad" aggregator="sum" visible="true">
    </Measure>
  </Cube>
</Schema>
```

Diseño de procesos de carga y actualización

En esta sección se presentan los algoritmos que no fueron mencionados en el documento principal, intentando así no ser repetitivos presentándolos en ese lugar.

Algoritmo 6 Carga Inicial de Dimensión de D_GRUPO_EDAD_FONASA_CAP

1: **para** $i = -1 \rightarrow 150$ **hacer**
2: Obtener rango etario en LKP_GRUPO_EDAD_FONASA_CAP
3: Insertar rango etario en D_GRUPO_EDAD_FONASA_CAP
4: **fin para**

Algoritmo 7 Actualización de Dimensión de D_GRUPO_EDAD_FONASA_CAP

1: **para** $i = -1 \rightarrow 150$ **hacer**
2: Obtener rango etario en LKP_GRUPO_EDAD_FONASA_CAP
3: Actualiza rango etario en D_GRUPO_EDAD_FONASA_CAP
4: **fin para**

Algoritmo 8 Carga inicial de Dimensión de D_GRUPO_EDAD_INFAMILIA

1: **para** $i = -1 \rightarrow 150$ **hacer**
2: Obtener rango etario en LKP_GRUPO_EDAD_INFAMILIA
3: Insertar rango etario en D_GRUPO_EDAD_INFAMILIA
4: **fin para**

Algoritmo 9 Actualización de Dimensión de D_GRUPO_EDAD_INFAMILIA

1: **para** $i = -1 \rightarrow 150$ **hacer**
2: Obtener rango etario en LKP_GRUPO_EDAD_INFAMILIA
3: Actualiza rango etario en D_GRUPO_EDAD_INFAMILIA
4: **fin para**

Algoritmo 10 Carga Inicial de Dimensión de D_GRUPO_EDAD_OTRO

1: **para** $i = -1 \rightarrow 150$ **hacer**
2: Obtener rango etario en LKP_GRUPO_EDAD_OTRO
3: Insertar rango etario en D_GRUPO_EDAD_OTRO
4: **fin para**

Algoritmo 11 Actualización de Dimensión de D_GRUPO_EDAD_OTRO

1: **para** $i = -1 \rightarrow 150$ **hacer**
2: Obtener rango etario en LKP_GRUPO_EDAD_OTRO
3: Actualiza rango etario en D_GRUPO_EDAD_OTRO
4: **fin para**

Algoritmo 12 Carga inicial de Dimensión de D_SEXO

1: Se insertan los tres sexos (M, F, I)

Algoritmo 13 Carga inicial de Dimensión de D_TIEMPO

1: Se inserta el Mes, Año, clave generada con Mes y Año correspondiente al esquema del mes y del año que se cargan en el data warehouse

Algoritmo 14 Carga inicial de Dimensión D_TIPO_COBERTURA

1: **para** cada tipo de cobertura definido en LKP_TIPO_COBERTURA **hacer**
2: **si** no existe o existe en estado inactivo en D_TIPO_COBERTURA **entonces**
3: Se obtiene nueva referencia
4: Se inserta en la tabla D_TIPO_COBERTURA
5: **fin si**
6: **fin para**

Algoritmo 15 Actualización de Dimensión de D_TIPO_COBERTURA

1: **para** cada tupla de D_TIPO_COBERTURA en estado activo **hacer**
2: **si** no existe en LKP_TIPO_COBERTURA **entonces**
3: Se cambia el estado a inactivo
4: **fin si**
5: **fin para**

Algoritmo 16 Carga inicial de Dimensión de D_UNIDAD_ASISTENCIAL

1: **para** cada tupla del Join entre las tablas Unidad Asistencial y Departamentos **hacer**
2: **si** no existe en D_UNIDAD_ASISTENCIAL **entonces**
3: Se inserta en D_UNIDAD_ASISTENCIAL
4: **fin si**
5: **fin para**

Estructura general de los reportes

Reportes relativos a los grupos de edad Infamilia:

1. Este reporte corresponde al total de afiliados a ASSE por grupo de edad Infamilia y según departamento.
   a) Columnas:
      - La primer columna corresponde a los totales de afiliados por departamento
      - Las columnas restantes detallan las cantidades de afiliados para cada grupo de edad Infamilia, los grupos de edades de este tipo son los siguientes y están expresados en años:
        - de 0 a 2
        - de 3 a 5
        - de 6 a 9
        - de 10 a 11
        - de 12 a 14
        - de 15 a 19
        - de 20 a 24
        - de 25 a 29
        - de 30 a 49
        - de 50 a 64
        - 65 y más
        - No indicado

   b) Filas:
      - Las filas corresponden a los 19 departamentos + 1 valor correspondiente a los afiliados que no tienen un departamento definido

2. Total de afiliados a ASSE por grupo de edad Infamilia, según departamento y unidad asistencial
   a) Columnas
      - La primer columna corresponde a los totales de afiliados por departamento y por unidades asistenciales
      - Las columnas restantes detallan las cantidades de afiliados para cada grupo de edad Infamilia, los grupos de edades son los mismos que los del reporte anterior:

- de 0 a 2
- de 3 a 5
- de 6 a 9
- de 10 a 11
- de 12 a 14
- de 15 a 19
- de 20 a 24
- de 25 a 29
- de 30 a 49
- de 50 a 64
- 65 y más
- No indicado

b) Filas

- Los 19 departamentos + Departamento indefinido
  - A su vez dentro de cada departamento se detallan las cantidades de afiliados para cada unidad asistencial correspondiente a su respectivo departamento. El total de unidades asistenciales es 110.

Reportes relativos a los grupos de edad Fonasa (Cápitas):

1. Total de afiliados a ASSE, por tipo de cobertura según sexo y grupo de edad Fonasa
   a) En este reporte las columnas corresponden a la cantidad de afiliados asociado a cada tipo de cobertura
   - Total
   - Tipos de cobertura
     - Carné
       - Asistencia
         - Bonificado 1
         - Bonificado 2
         - Carné de asistencia gratuito
         - Materno - Infantil
       - Vitalicio
     - Fonasa
       - Afiliados
       - De Oficio

- Cuotas
  - Convenios colectivos
  - Individuales
  - Núcleo familiar
- Convenio
  - Panes
  - Sanidad policial

b) Las filas corresponden primeramente a los totales de afiliados discriminados por sexo

- Total
- Sexo Masculino
- Sexo Femenino
- Sexo no indicado
  - A su vez para cada categorización anterior correspondiente al sexo del afiliado se agrupa por los diferentes grupo de edades Fonasa incluyendo un grupo de edad "No indicado"
    - Grupos de edades
      - Total
      - Menores de 1
      - de 1 a 4
      - de 5 a 14
      - de 15 a 19
      - de 20 a 44
      - de 45 a 64
      - de 65 a 74
      - 75 y más
      - No indicado

2. Total de afiliados a ASSE por tipo de cobertura, según departamento

a) Aquí las columnas corresponden a los totales de afiliados correspondientes a los distintos tipos de cobertura

- Total
- Tipos de cobertura
  - Carné
    - Asistencia
      - Bonificado 1
      - Bonificado 2

        - ◇ Carné de asistencia gratuito
        - ◇ Materno - Infantil
      - Vitalicio
    - Fonasa
      - Afiliados
      - De Oficio
    - Cuotas
      - Convenios colectivos
      - Individuales
      - Núcleo familiar
    - Convenio
      - Panes
      - Sanidad policial

  b) Las filas de este reporte corresponden a los totales de afiliados en cada departamento incluyendo "Departamento indefinido"
    - Los 19 departamentos + Departamento indefinido

3. Total de afiliados a ASSE por tipo de cobertura, según departamento y unidad asistencial

  a) Las columnas corresponden a los totales de afiliados que están asociados a los diferentes tipos de cobertura
    - Total
    - Tipos de cobertura
      - Carné
        - Asistencia
          - ◇ Bonificado 1
          - ◇ Bonificado 2
          - ◇ Carné de asistencia gratuito
          - ◇ Materno - Infantil
        - Vitalicio
      - Fonasa
        - Afiliados
        - De Oficio
      - Cuotas
        - Convenios colectivos
        - Individuales
        - Núcleo familiar

- Convenio
  - Panes
  - Sanidad policial

b) En la agrupación primaria tenemos en cada fila los diferentes departamentos
   - Los 19 departamentos + Departamento indefinido
     - A su vez para cada departamento se especifican las cantidades de afiliados correspondiente a cada unidad asistencial, la cual a su vez pertenece a un departamento. El total de unidades asistenciales es 110.

Anexo 3: Procesos de ETL para cubos de afiliados

Introducción

Como fué mencionado en el capítulo 5.3, el proceso ETL consiste en la extracción, transformación y carga de datos como lo describe su sigla en inglés (Extract, Transform and Load), desde una o distintas fuentes de datos hacia el Data Warehouse. También como fue mencionado, se utilizó la herramienta llamada *Spoon* [1], provista por la plataforma de Pentaho para diseñar, implementar y ejecutar los procesos de necesarios para las cargas de las tablas requeridas que son utilizadas por los cubos. En éste anexo se detallan los procesos de trabajos creados con ésta herramienta para cada una de las etapas de la capa de ETL para la carga de los cubos OLAP que contendrán la información para las Afiliaciones.

Trabajo principal

El trabajo principal de carga de las afiliaciones de usuarios es el esqueleto principal de proceso de la carga del DW para ésta sección de la información. El proceso gestiona las transformaciones y trabajos intermedios a ejecutarse y que realizan la carga ordenada de los datos requeridos. La primer transacción realizada por el trabajo es la verificación de las cargas anteriores y es la que decide si se debe realizar la carga o no del DW. Luego de realizada la verificación se comienza con la ejecución del trabajo de actualización donde se borran las tablas temporales y se las carga con los datos limpios y mejorados de las fuentes. Luego de ésto se ejecuta el trabajo de carga de las dimensiones de los cubos y posteriormente se ejecuta la tranformación de carga de las tablas de hechos involucradas. En caso de existir una falla en cualquiera de las etapas anteriores se aborta la ejecución y se añade una entrada en la tabla de logs con la información del fallo y la fecha y hora en la ocurrió, en caso de suceso se agrega una entrada en la tabla de logs con la fecha

[1] *Spoon se encuentra contenido dentro del paquete de Data Integration de Pentaho - http://sourceforge.net/projects/pentaho/files/Data%20Integration/4.1.0-stable/ - Ultimo acceso 03/2011*

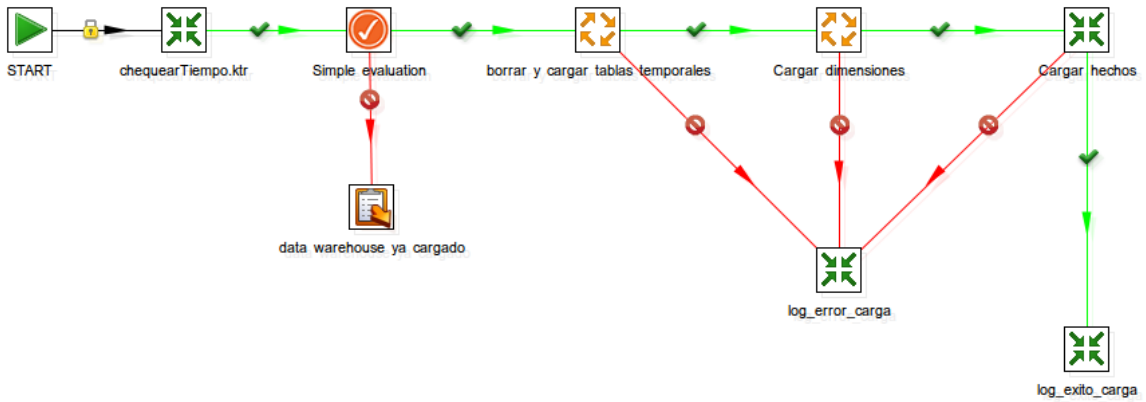


Figura 4: Trabajo principal de carga de los datos de Afiliaciones de usuarios

y hora de fin de la carga realizada. La figura 4 ilustra el diagrama de trabajo de carga detallado anteriormente.

Verificación de carga

Como fue mencionado en la sección anterior, previo a todos los procesos de carga de las distintas tablas de los cubos, se realiza una verificación que permite abortar los trabajos de carga si el DW ya fue cargado correctamente para el mes corriente. Para ésto se realizó una transformación de verificación que es la primera que se ejecuta en el trabajo principal de carga. Ésta transformación verifica primero que el mes que se desea cargar no este cargado en el sistema, luego verifica que actualmente no se este realizando una carga en el momento de la ejecución y por ultimo verifica que el si se está realizando una carga en el momento de la verificación el tiempo transcurrido no sea mayor al período permitido definido por un *Timeout* configurado en el proceso, si alguna de las verificaciones se cumplen entonces se actualiza el log de carga con los datos del intento realizado, sino se procede a ejecutar el inicio de la carga y se actualiza el log de cargas con la fecha y hora en la cual se inicio la ejecución del proceso. La figura 5 muestra el diagrama de Spoon de dicha transformación.

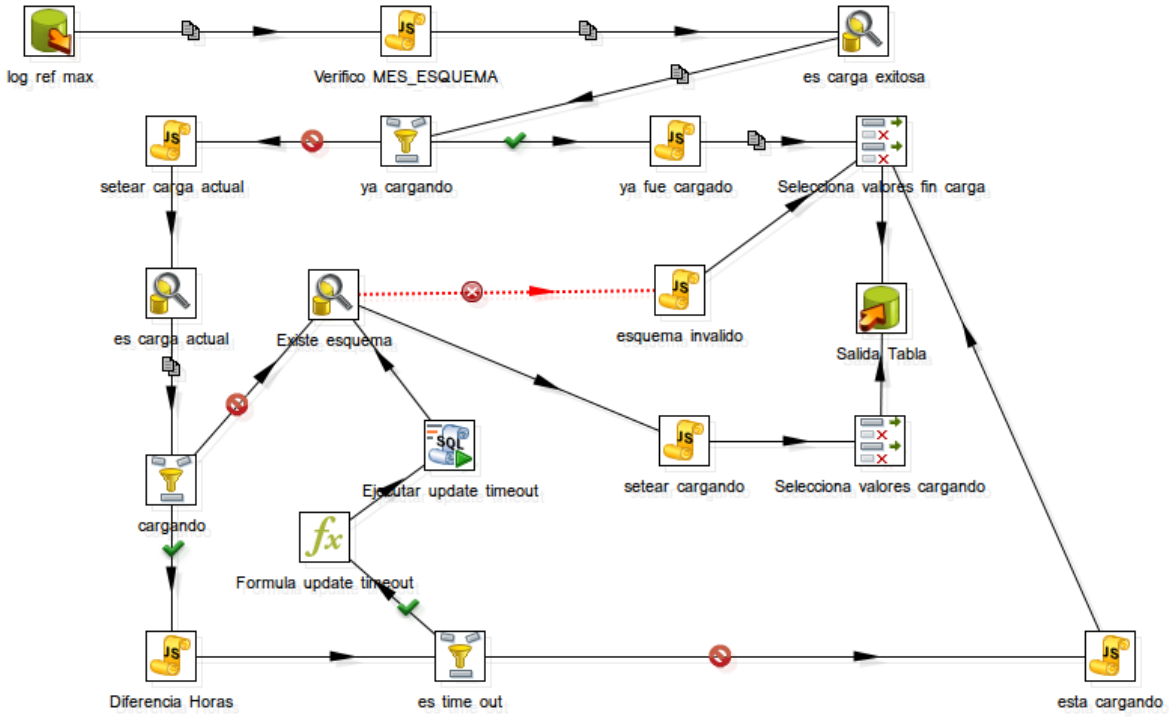


Figura 5: Trabajo de verificación de carga del DW

Carga de la base de datos temporal

Para la carga de datos desde la base de datos fuente *id_usuarios* a la base de datos destino temporal se creó un trabajo en *Spoon* para ejecutar así las distintas transformaciones, la siguiente imagen muestra una captura de éste trabajo.

Figura 6: Trabajo de carga general de la base temporal

La ejecución comienza en el nodo *Start* para luego continuar con el siguiente paso del tipo SQL, el cual será el encargado de eliminar todas las entradas de todas las tablas de la base de datos temporal. Las siguientes transformaciones del trabajo consisten en limpiar los datos de origen cuando es necesario y hacer el correspondiente mapeo en las nuevas tablas. Las transformaciones se componen de pasos y saltos de control o de flujo entre dichos pasos. Estos están clasificados en varios grupos, a modo de ejemplo, citamos: entrada, salida, transformación, flujo, scripting, lookup, validadores, estadísticas, etc.

En la transformación de los *Departamentos* hacemos el mapeo correspondiente a la nueva base de datos y la fila que tiene en la tabla origen el valor *null* le asignamos el valor 0 y a la descripción correspondiente el valor a clasificar (ver imagen 7).

Figura 7: Trabajo de transformación de tabla Departamentos

En forma análoga se carga *Localidades* como se ilustra en la siguiente figura.

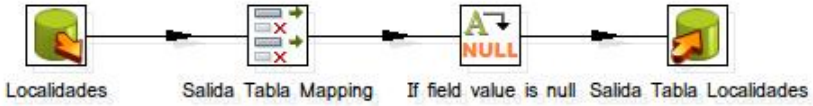


Figura 8: Trabajo de transformación de tabla Localidades

En la carga de los datos correspondientes a la tabla *Personas*, se han encontrado algunas inconsistencias en los datos de origen que no respetan su integridad. Para evitar esto, se creó un archivo de salida en caso de error, el cual nos marca el registro que no cumple con la integridad de la base de datos, y el proceso de carga no se detiene (ver imagen 9).

Figura 9: Trabajo de transformación de tabla Personas

Para la carga de la tabla *Unidad Asistencial*, se llevó a cabo un casteo de la columna correspondiente a las *Unidad Ejecutora* que era del tipo texto a entero. Para esta acción se utiliza el paso JavaScript (ver imagen 10).

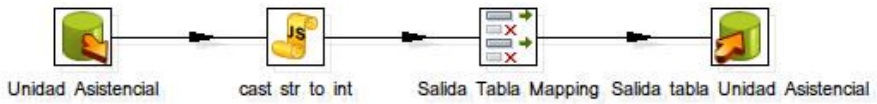


Figura 10: Trabajo de transformación de tabla Unidad Asistencial

La tabla *Unidad Ejecutora*, también contiene un paso JavaScript. En este caso se buscaba hacer el casteo de texto a entero de los campos correspondientes al número de declaración jurada, y al identificador del usuario (ver imagen 11).

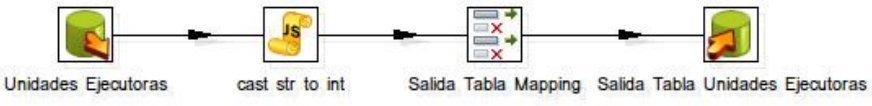


Figura 11: Trabajo de transformación de tabla Unidades Ejecutoras

En las tablas *Grupo Familiar* y *Usuarios* se hace simplemente un mapeo entre las tablas de la base de datos origen *id_ usuarios* a la base de destino temporal (ver imagen 12).

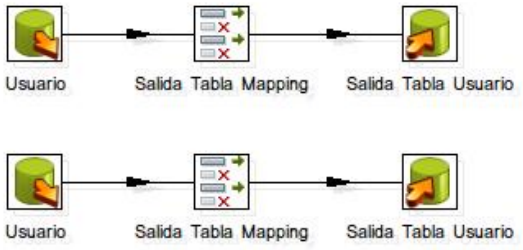


Figura 12: Trabajos de transformación de las tablas de Grupo Familiar y de Usuarios

Carga de las dimensiones del Data Warehouse

La segunda parte de nuestro proceso de ETL consiste en cargar o actualizar las tablas de dimensión de nuestro Data Warehouse según se requiera, para finalmente cargar la tabla de hechos. Para lograrlo, se tomará como fuente de datos del proceso de ETL la base temporal descrita en la sección anterior, y nuestra base de datos de destino será la del Data Warehouse. A continuación se especifica el trabajo que comprende a las transformaciones encargadas de esta tarea, así como la explicación detallada de las mismas.

El trabajo encargado de la carga de las dimensiones comienza en el paso *Start*, para luego ocuparse de la carga de las dimensiones *Tipo Cobertura*, *Unidad Asistencial*, *Grupo de Edad*, *Sexo*, para finalmente cargar la dimensión *Tiempo*. En la figura 13 se ilustra el trabajo de carga de todas las dimensiones.

Figura 13: Trabajo de carga de cada una de las dimensiones

El proceso de carga del *Tipo de Cobertura* se decidió hacer en dos transformaciones: la primera encargada de la actualización de los registros existentes en la dimensión, y la segunda centrada en la carga de los elementos faltantes en dicha dimensión.

La primera transformación comienza obteniendo los registros de la tabla de dimensión correspondiente al *Tipo de Cobertura*. En el siguiente paso correspondiente al lookup se hace una búsqueda sobre la tabla *lkp_tipo_cobertura* de la base temporal. Esto nos permite saber si alguno de los registros de la dimensión cambió, y en dicho caso se actualiza el registro de la dimensión en un estado eliminado ingresándole una fecha de finalización al registro.

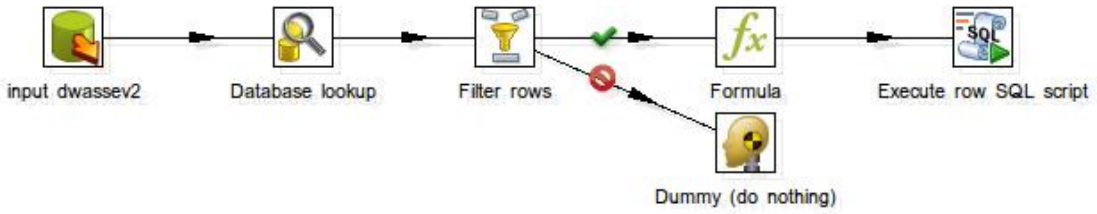


Figura 14: Trabajo de carga de la tabla de Tipo de Cobertura

Para la inserción de nuevos registros en la tabla de dimensión del *Tipo de Cobertura*, se procede a verificar desde la tabla *lkp_tipo_cobertura* si existen sus correspondientes en la dimensión *Tipo de Cobertura* del Data Warehouse. En caso de no existir o que el registro cuente con fecha de finalización dentro de la dimensión, se inserta un nuevo registro.

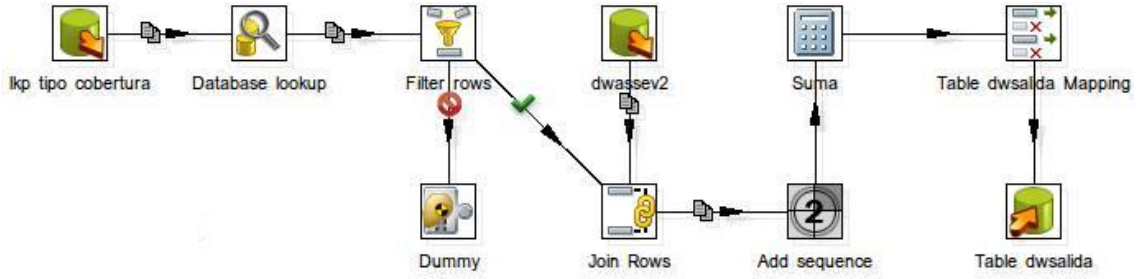


Figura 15: Trabajo de carga de la tabla de Tipo de Cobertura

La siguiente transformación dentro de nuestro trabajo es la de la carga de la dimensión *Unidad Asistencial*. Para esto, se obtienen los registros de la tabla temporal y se le agrega un registro más para tener así la unidad asistencial con valor indefinido para poder clasificar los hechos que no tengan *Unidad Asistencial* asociados. Luego se realiza una verificación de las unidades asistenciales de nuestro Data Warehouse, y en caso de que el registro no exista, lo insertamos.

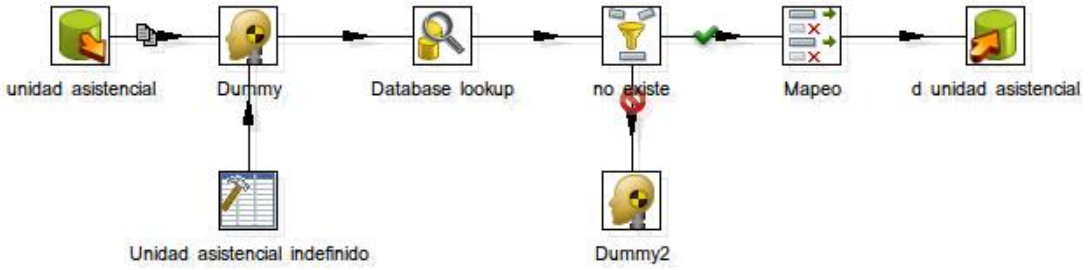


Figura 16: Trabajo de carga de la tabla de Unidad Asistencial

En la carga de las tres dimensiones correspondientes al *Grupo de Edad*, se genera una secuencia de 150 números comenzando en el -1, el cual corresponde a edades fuera de rango o indefinidas en la tabla *Personas*. Para cada número generado se hace un chequeo de su existencia en cada dimensión, y en caso de no existir, se hace una búsqueda de la descripción correspondiente en las tablas que definen los rangos de cada grupo edad en la base de datos *temporal*, y se insertan en la dimensión correspondiente (ver figura 17).

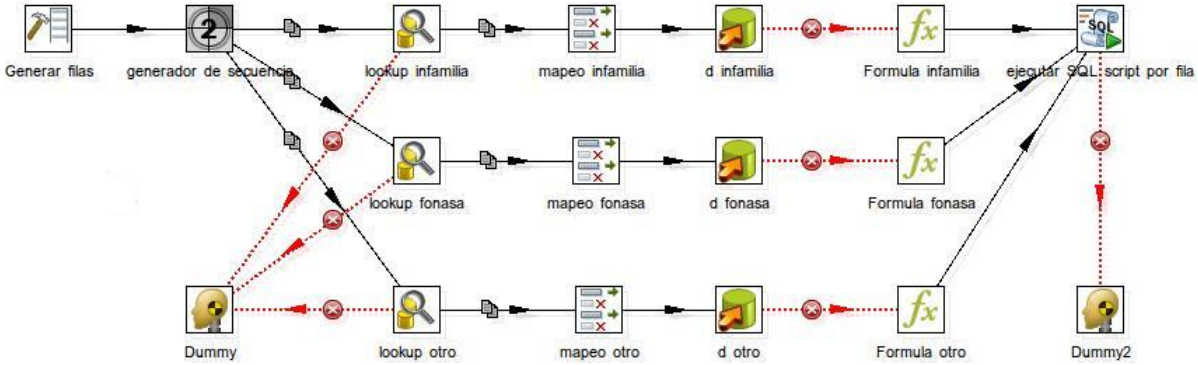


Figura 17: Trabajo de carga de las dimensiones de Grupo de Edad

La dimensión *Sexo* es cargada mediante la ejecución de un script que inserta los tres registros correspondientes a masculino, femenino e indefinido. En las ejecuciones posteriores a la carga inicial, estas ejecuciones darán error y por eso se agregó el flujo a una etapa dummy para que no haga nada en dicho caso.

Figura 18: Trabajo de carga de la dimensión Sexo

Para finalizar la carga de las dimensiones, pasamos a cargar la dimensión correspondiente al *Tiempo*. Para ello generamos una fila, la cual es cargada mediante código JavaScript en el paso. La información cargada en cada ejecución es el número del año, del mes anterior a la fecha cargada, la descripción correspondiente a dichos mes y año, y una clave generada con estos dos datos (ver figura 19).

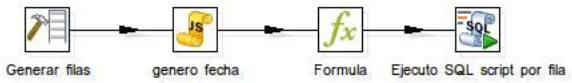


Figura 19: Trabajo de carga de la dimensión Tiempo

Carga de la tabla de hechos del Data Warehouse

Para la transformación correspondiente a la carga de la tabla de hechos, obtenemos los datos mediante una consulta a la base de datos temporal. Posteriormente a los datos de entrada le agregamos el mes y el año provenientes del parámetro de entrada MES_ESQUEMA con el formato ya descrito yyyyMM. Comenzamos verificando que los registros tengan asociados un tipo de cobertura, y en caso de no tenerlo, le asignamos el tipo de cobertura por defecto. Posteriormente se realizan dos pasos que aumentan enormemente el tiempo de carga, los cuales corresponden al ordenamiento de los datos y su posterior agrupamiento. Finalmente (ver figura 20) se insertan los datos en la tabla de hechos de afiliaciones del Data Warehouse .

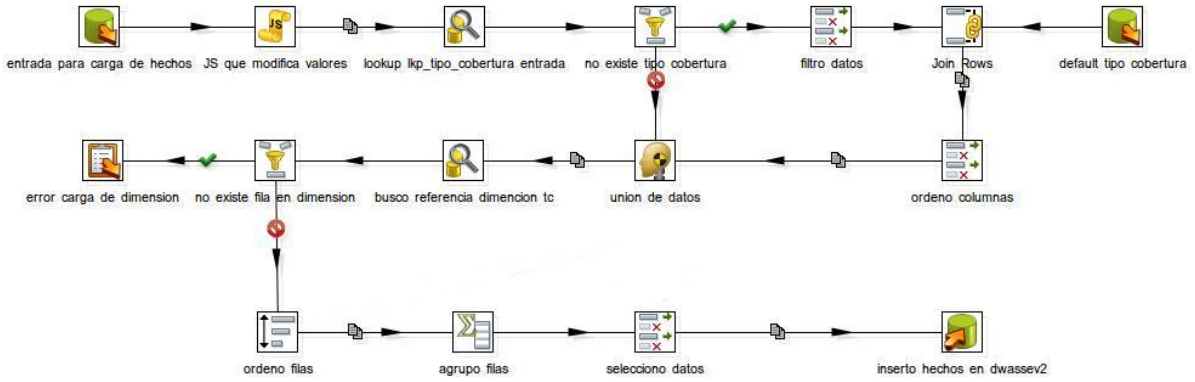


Figura 20: Trabajo de carga de la tabla de Hechos de Afiliaciones

Anexo 4: Extensión del DW a partir del sistema AP-SGA

Introducción

Tras la necesidad de ASSE de tener nuevos reportes e indicadores, nos fueron entregados nuevos requerimientos para poder así facilitar la tomas de decisiones por parte de los directivos mediante estas herramientas. Los datos para cubrir estos requerimientos provienen del Sistema de Gestión Asistencial (AP-SGA), el cual es el software utilizado por ASSE para la atención en la red de asistencia primaria. El sistema está desarrollado en Genexus, generando código Java. La base de datos que utiliza es Oracle, con un volumen de datos superior a 60 GB.

Objetivos

De los requerimientos suministrados, se negoció la implementación de cinco de la lista pedida ya que la implementación de todos los que se pedían excedía el alcance de este proyecto. Estos requerimientos consisten en un conjunto de reportes que no existen en la actualidad en la institución.

Relación entre requerimientos, dimensiones y medidas

Los requerimientos que se decidieron implementar son presentados a continuación:

Total de consultas médicas y no médicas

- Dimensión:
  - Departamento Indicadores: Departamento donde vive el afiliado que recibe la prestación
  - Grupo Edad Infamilia: Grupo de edad correspondiente a los rangos de edad de Infamilia

    - Grupo Edad Fonasa: Grupo de edad correspondiente a los rangos de edad de Fonasa
    - Servicios Médicos: Permite clasificar la prestación ente los distintos servicios brindados así como saber a que tipo de servicio pertenece
    - Sexo: Determina el sexo del afiliado que recibió la prestación
    - Tiempo: Permite saber en que mes y que año se realizo la prestación
    - Unidades Ejecutoras: Se logra saber en que unidad ejecutora y en que departamento se realizó
- Medida:
    - Cantidad: Esta medida representa la cantidad de consultas realizadas

Total de consultas médicas

Este requerimiento se satisface haciendo un filtrado sobre la dimensión que corresponde los Servicios Médicos al seleccionar los que son del tipo 'consultas médicas'

Total de consultas no médicas

En forma análoga, este requerimiento se satisface haciendo un filtrado sobre la dimensión que corresponde los Servicios Médicos al seleccionar los que son del tipo 'consultas no médicas'

Giro de camas

- Dimensión:
    - Departamento Indicadores: Departamento donde vive el afiliado que recibe la prestación
    - Grupo Edad Infamilia: Grupo de edad correspondiente a los rangos de edad de Infamilia
    - Grupo Edad Fonasa: Grupo de edad correspondiente a los rangos de edad de Fonasa
    - Sexo: Determina el sexo del afiliado que recibió la prestación
    - Tiempo: Permite saber en que mes y que año se efectuó el egreso del afiliado
    - Unidades Ejecutoras: Permite saber en que unidad ejecutora y en que departamento se efectuó el egreso del afiliado
- Medida:
    - Cantidad: Esta medida representa la cantidad de egresos de afiliados que estaban ocupando una cama
    - Cantidad por camas: Esta medida representa la cantidad de egresos de afiliados que estaban ocupando una cama por la cantidad de camas disponibles en dicha unidad ejecutora en el lapso de un mes

Numero de partos

- Dimensión:
  - Departamento Indicadores: Departamento donde vive el afiliado que recibe la prestación
  - Grupo Edad Infamilia: Grupo de edad correspondiente a los rangos de edad de Infamilia
  - Grupo Edad Fonasa: Grupo de edad correspondiente a los rangos de edad de Fonasa
  - Sexo: Determina el sexo del nacido
  - Tiempo: Permite saber en que mes y que año se produjo el nacimiento
  - Tipo de Parto: Determina si un parto es de tipo óbito, aborto y nacido vivo
  - Unidades Ejecutoras: Permite saber en que unidad ejecutora y en que departamento se efectuó el parto
- Medida:
  - Cantidad: Esta medida representa la cantidad de nacimientos de los tres tipos

Diseño de la solución

Naturaleza de los datos

Los datos utilizados provienen del sistema APSGA El motor de base de datos utilizado para este sistema es un ORACLE. Los datos utilizados para cubrir los requerimientos son las siguientes dos tablas:

- Movil
- Prestación

La tabla Movil, es la que proporciona la información de las camas según su centro. Para obtener las camas sobre las cuales se va a hacer la carga del DW se deben seleccionar únicamente las sanas. El problema encontrado es que existe un registro de camas muy limitado, y esto no refleja la realidad actual de los centros.
La tabla prestación centraliza el registro de todas las prestaciones del sistema APSGA. Esto hace que esta tabla tenga mas de 350 columnas, con muchos campos en null y mas de 60 millones de registros. Para los distintos requerimientos, se necesitan obtener información a partir de columnas distintas de la tabla, y cualquier filtro aplicado sobre la información se obtienen los datos con un tiempo de respuesta largos que llegan alcanzar algunos minutos dependiendo de la complejidad de la consulta.

Modelado Multidimensional

En esta sección se modelan las dimensiones y los hechos necesarios para cumplir con los requerimientos que no han sido aun definidos en la sección 4.4.

Dimensión Departamento Indicadores

La dimensión ***Departamento Indicadores***, tiene como fin hacer un filtrado por el departamento donde vive el afiliado registrado en el sistema. Solo cuenta con el atributo correspondiente al nombre del departamento (ver figura 21).

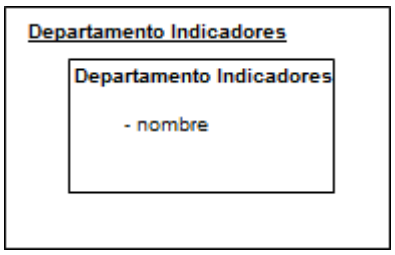


Figura 21: Dimensión Departamento Indicadores

Dimensión Servicios Médicos

Esta representa a los servicios médicos, no médicos y servicios que no aplican en ninguna de las dos categorías según el listado que nos fue proporcionado por ASSE. Los atributos con los que cuenta la dimensión son el nombre del servicio, y el tipo que clasifica a cada uno de estos (ver figura 22).

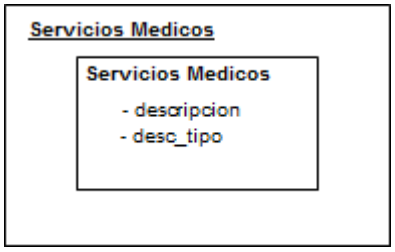


Figura 22: Dimensión Servicios Médicos

Dimensión Tipo Parto

La dimensión ***Tipo de parto*** es utilizada en el cubo Numero de Partos. Permite filtrar los numeros de parto según los valores 'Óbito', 'Aborto' y 'Nacido Vivo'. Estos

corresponden a los valores posibles al atributo nombre (ver figura 23).

Figura 23: Dimensión Tipo Parto

Dimensión Unidades Ejecutoras

Esta dimensión es utilizada para el filtrado de las afiliaciones según la unidad ejecutora o mediante el departamento en la cual hizo la consulta el afiliado. Para modelar esto se creó la dimensión con una jerarquía de dos niveles. En el nivel superior se representa la entidad departamentos, la cual está asociada a varias unidades ejecutoras que se representan en el nivel de jerarquía inferior. Esto permite aplicar las operaciones de OLAP roll-up y drill-down, navegando así por los niveles de la jerarquía dentro de la dimensión (ver figura 24).

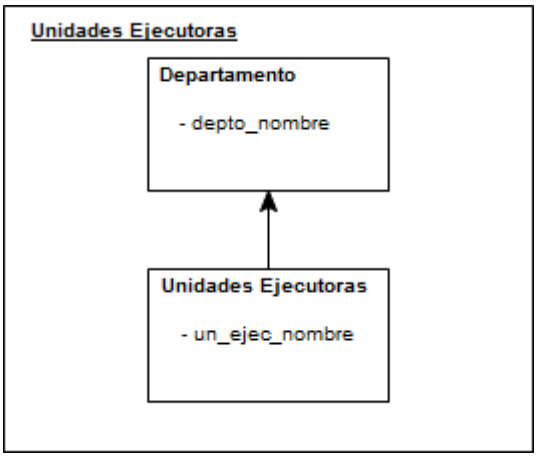


Figura 24: Dimensión Unidades Ejecutoras

Hecho Consultas

Este hecho modela la cantidad de consultas según el cruzamiento de las dimensiones: servicios médicos, unidades ejecutoras, grupo edad infamilia, grupo edad fonasa, departamento indicadores, sexo y tiempo. Cada cantidad se calcula contando las consultas efectuadas en un lapso de un mes según las dimensiones especificadas anteriormente. En la figura 25, se presenta la tabla de hechos llamada Consultas.

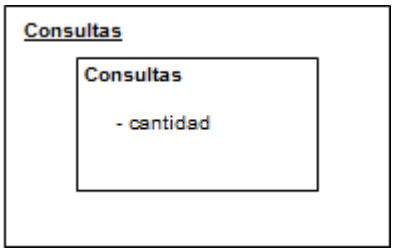


Figura 25: Hecho Consultas

Hecho Giro de Camas

Giro de camas modela la cantidad de egresos de afiliados por unidad ejecutora en un lapso de un mes así como también lo hace según la cantidad de egresos por cantidad de camas de esta unidad ejecutora en el mismo período. Se realiza también un cruzamiento con las dimensiones: grupo edad infamilia, grupo edad fonasa, departamento indicadores y sexo (ver figura 26).

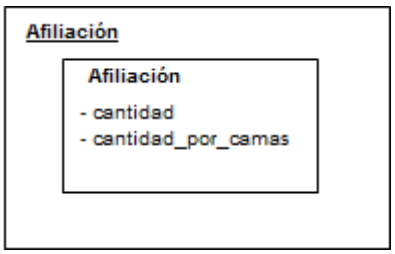


Figura 26: Hecho Giro de Camas

Hecho Numero de Partos

Este modela la cantidad de prestaciones correspondientes a partos según el cruzamiento de las dimensiones: tipo parto, unidades ejecutoras, grupo edad infamilia, grupo edad fonasa, departamento indicadores, sexo y tiempo (ver figura 27).

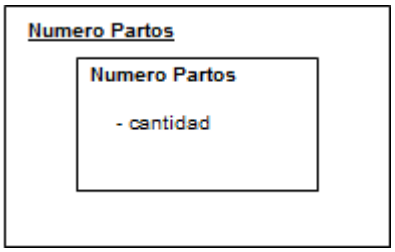


Figura 27: Hecho Numero de Partos

Esquema Multidimensional

En la figura 28 se muestran las dimensiones utilizadas para definir las consultas médicas, no médicas y las que no aplican a ninguna de las dos categorías. La figura 29 representa las dimensiones utilizadas para el requerimiento correspondiente al giro de camas.

Las dimensiones correspondientes al requerimiento de número de partos se puede ver en la figura 30

Se decidió utilizar un esquema estrella para el diseño de los tres cubos en lugar de un esquema copo de nieve, ganando así simplicidad en el diseño y velocidad de acceso por tener las distintas jerarquías desnormalizadas.

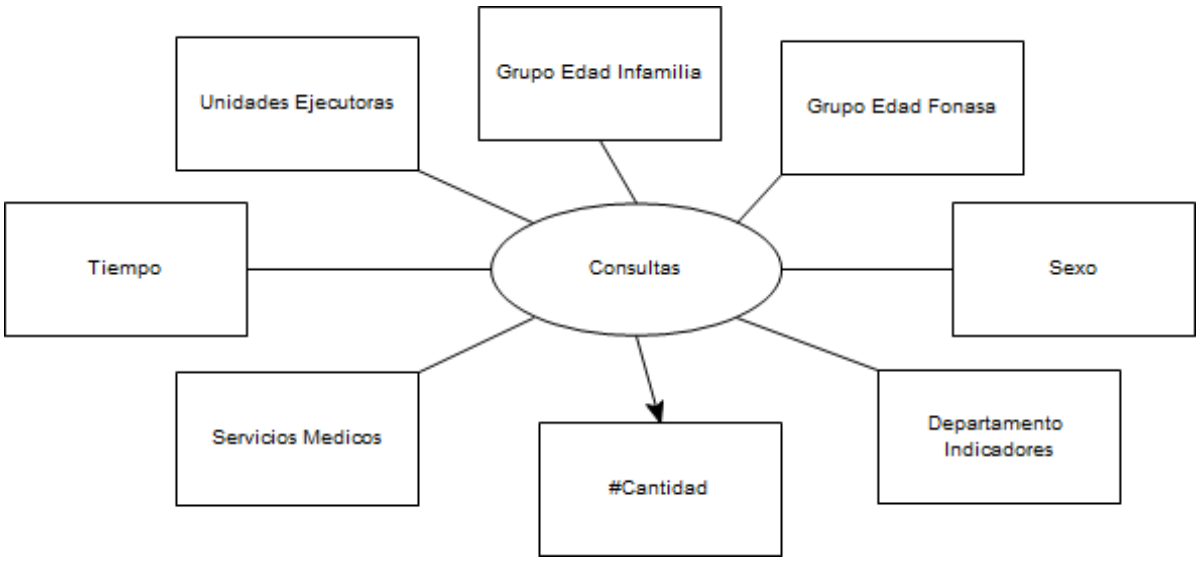


Figura 28: Relación dimensional para el cubo de consultas

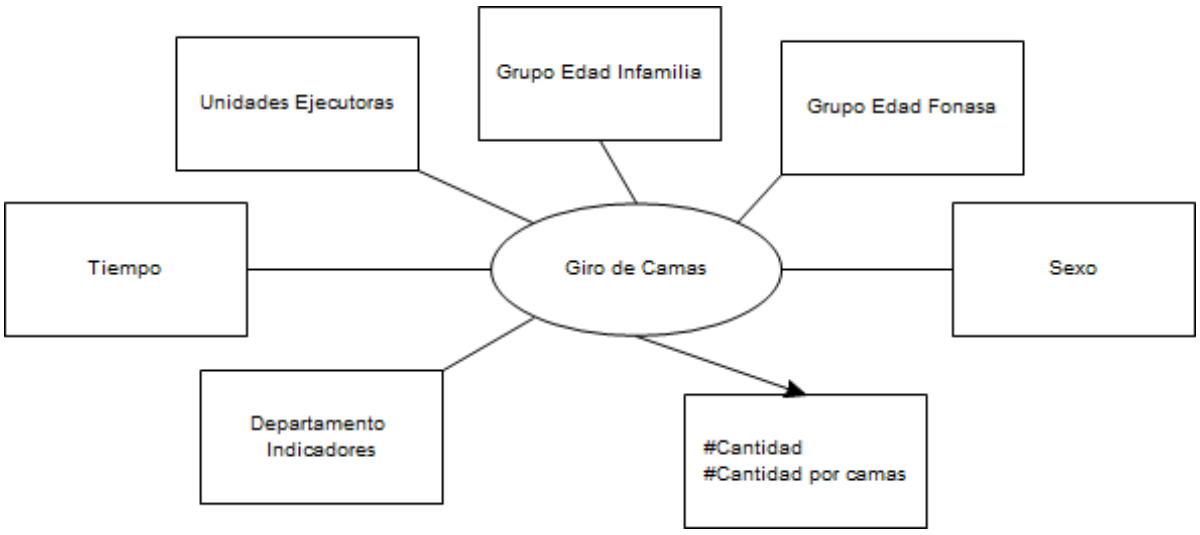


Figura 29: Relación dimensional para el cubo de giro de camas

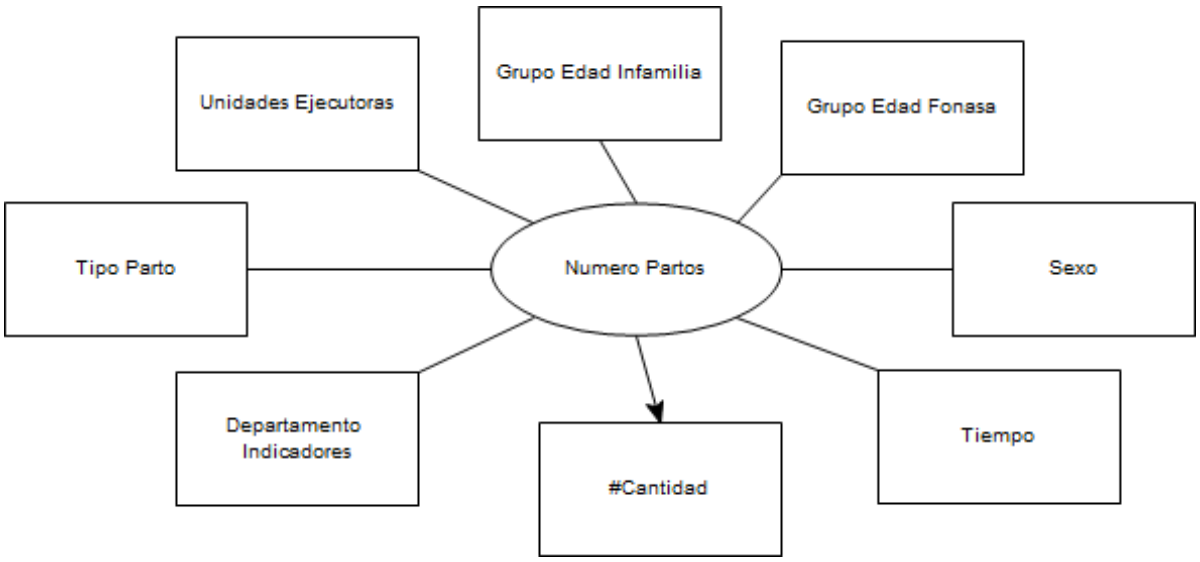


Figura 30: Relación dimensional para el cubo de numero de partos

Diseño relacional de la base que soporta a los cubos

En las figuras 31, 32 y 33 se encuentran las estructuras relacionales usadas para la creación de los modelos multidimensionales usado para los cubos creados para esta parte. Estos se derivan de las dimensiones y de los esquemas multidimensionales.

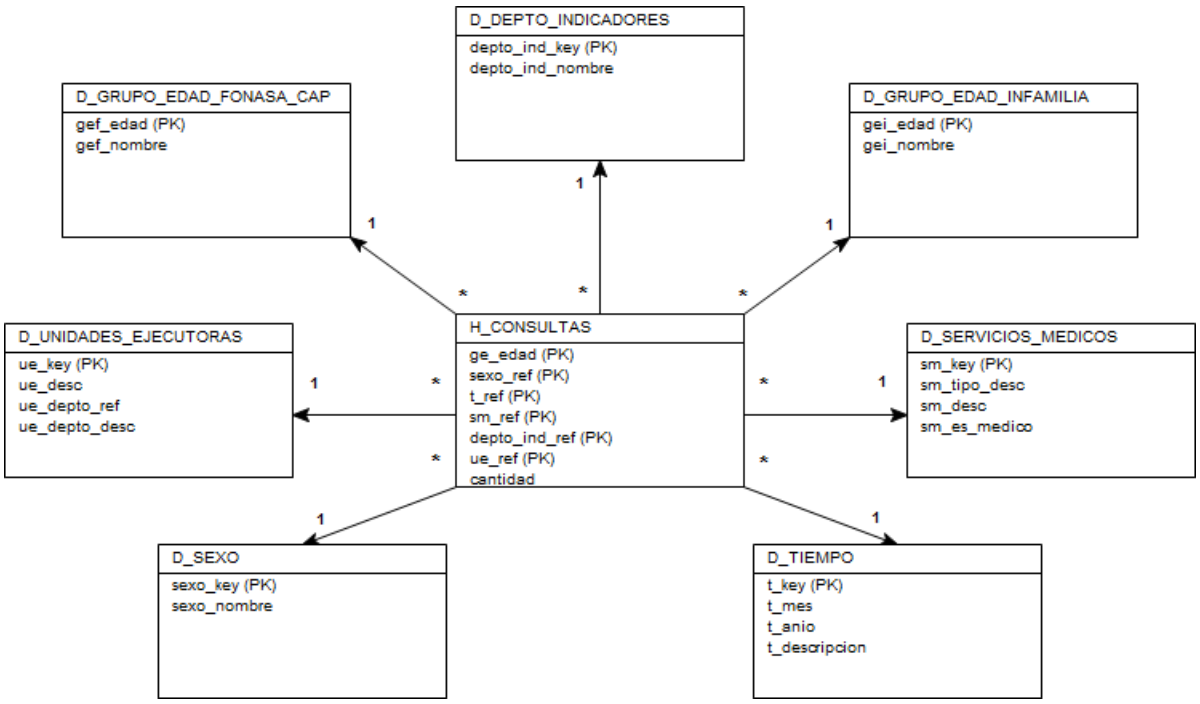


Figura 31: Modelo multidimensional en base a tablas relacionales para el cubo de consultas

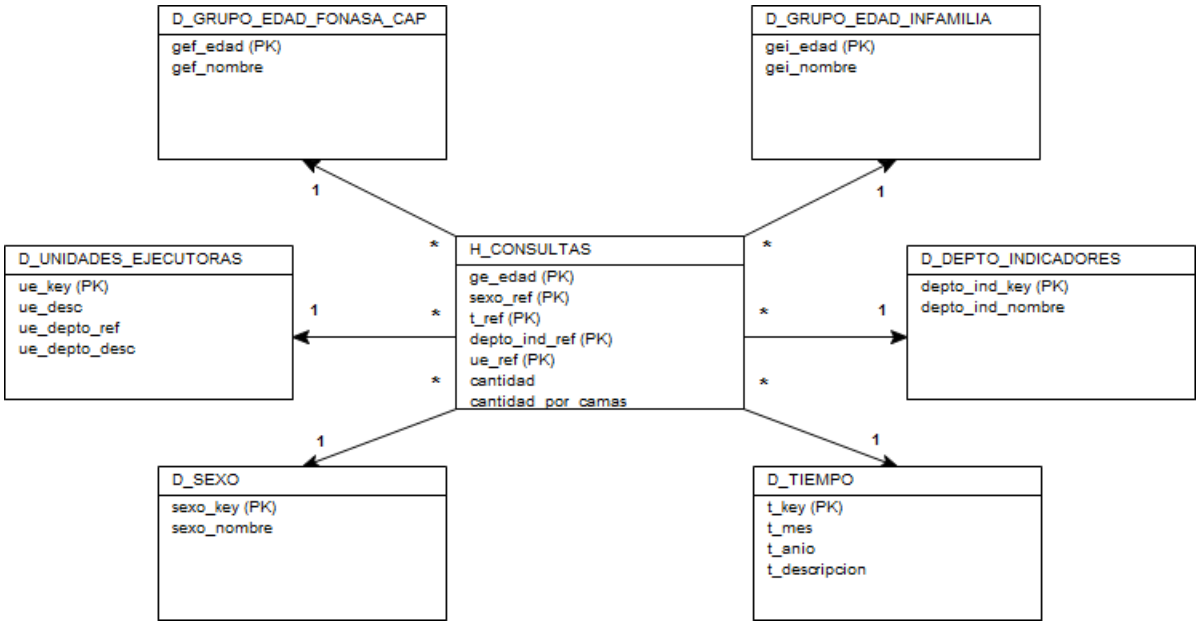


Figura 32: Modelo multidimensional en base a tablas relacionales para el cubo de giro de camas

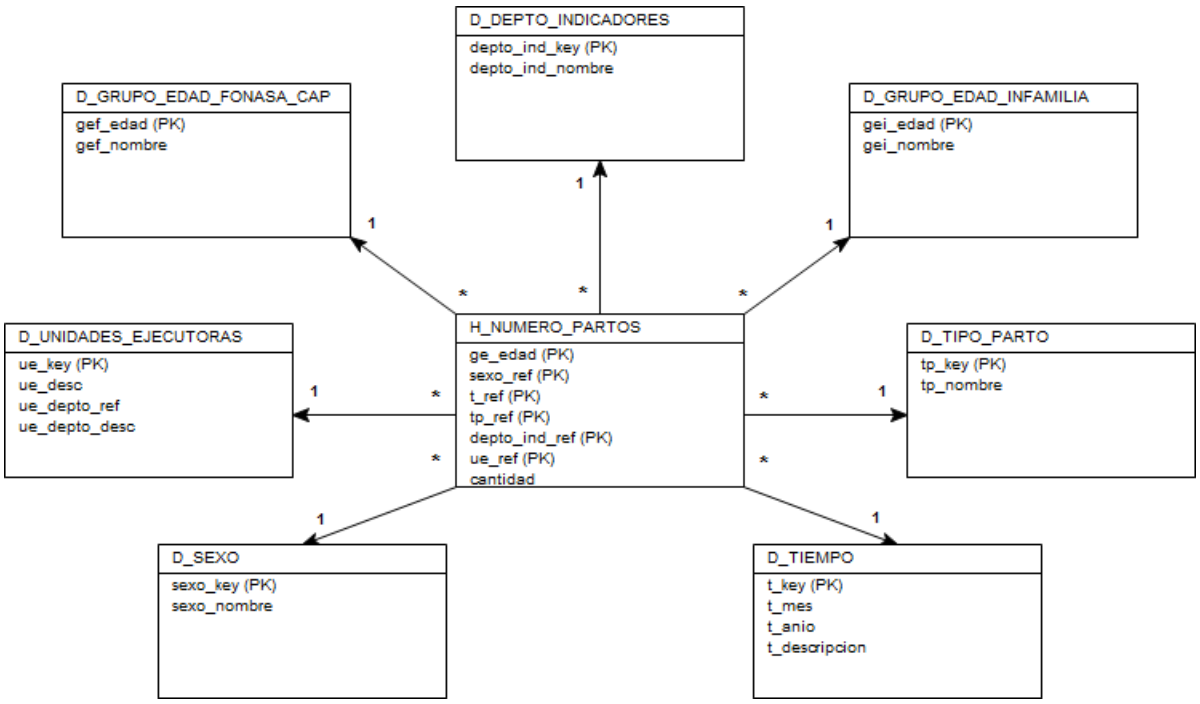


Figura 33: Modelo multidimensional en base a tablas relacionales para el cubo de numero de partos

Implementación de la solución

Extracción, transformación y carga

Mejora en calidad de datos

En esta parte del trabajo se encontró problemas en las edades de los afiliados provenientes de las fuentes de datos, se contó además del dato de su edad con la de su fecha de nacimiento. Mediante estos dos datos si uno no entraba dentro de las edades aceptables, se verificaba el otro dato con el fin de obtener el valor correcto de este parámetro.

Diseño de procesos de carga y actualización

Los procesos de ETL se encargan de la carga y actualización de las dimensiones tablas de hechos. A continuación se describen los procesos que fueron agregados para cubrir los requerimientos de esta sección.

Algoritmo 17 Carga inicial y actualización de dimensión D_SERVICIOS_MEDICOS

```
1: para cada servicio médico en LKP_SERVICIOS_MEDICOS hacer
2:   si no existe en D_SERVICIOS_MEDICOS entonces
3:     Se inserta en la tabla D_SERVICIOS_MEDICOS
4:   si no
5:     Se actualiza descripción y estado en la tabla D_SERVICIOS_MEDICOS
6:   fin si
7: fin para
```

Algoritmo 18 Carga inicial y actualización de dimensión D_UNIDADES_EJECUTORAS

```
1: para cada servicio médico en LKP_UNIDADES_EJECUTORAS hacer
2:   si no existe en D_UNIDADES_EJECUTORAS entonces
3:     Se inserta unidad ejecutora y su correspondiente departamento en la tabla
       D_UNIDADES_EJECUTORAS
4:   si no
5:     Se actualiza descripción y departamento en la tabla
       D_UNIDADES_EJECUTORAS
6:   fin si
7: fin para
```

Algoritmo 19 Carga y actualización de LKP_CANTIDAD_CAMAS

```
1: para cada centro médico de la tabla origen MOVIL hacer
2:   si cantidad de camas no rotas del centro >0 entonces
3:     unidad ejecutora = obtenerUnidadEjecutora(centro medico)
4:     estructuraTemporal.agregar(unidad ejecutora, cantidad camas)
5:   fin si
6: fin para
7: estructuraTemporal.agruparPorUnidadEjecutora()
8: para i=0; estructuraTemporal-1; i++ hacer
9:   si existe estructuraTemporal[i].unidadEjecutora en LKP_CANTIDAD_CAMAS
     entonces
10:    actualizar cantidad de camas en LKP_CANTIDAD_CAMAS
11:  si no
12:    insertar unidad ejecutora y cantidad de camas en LKP_CANTIDAD_CAMAS
13:  fin si
14: fin para
```

En las figuras 34 y 35 se ejemplifican con diagramas la carga de las dimensiones mencionadas.

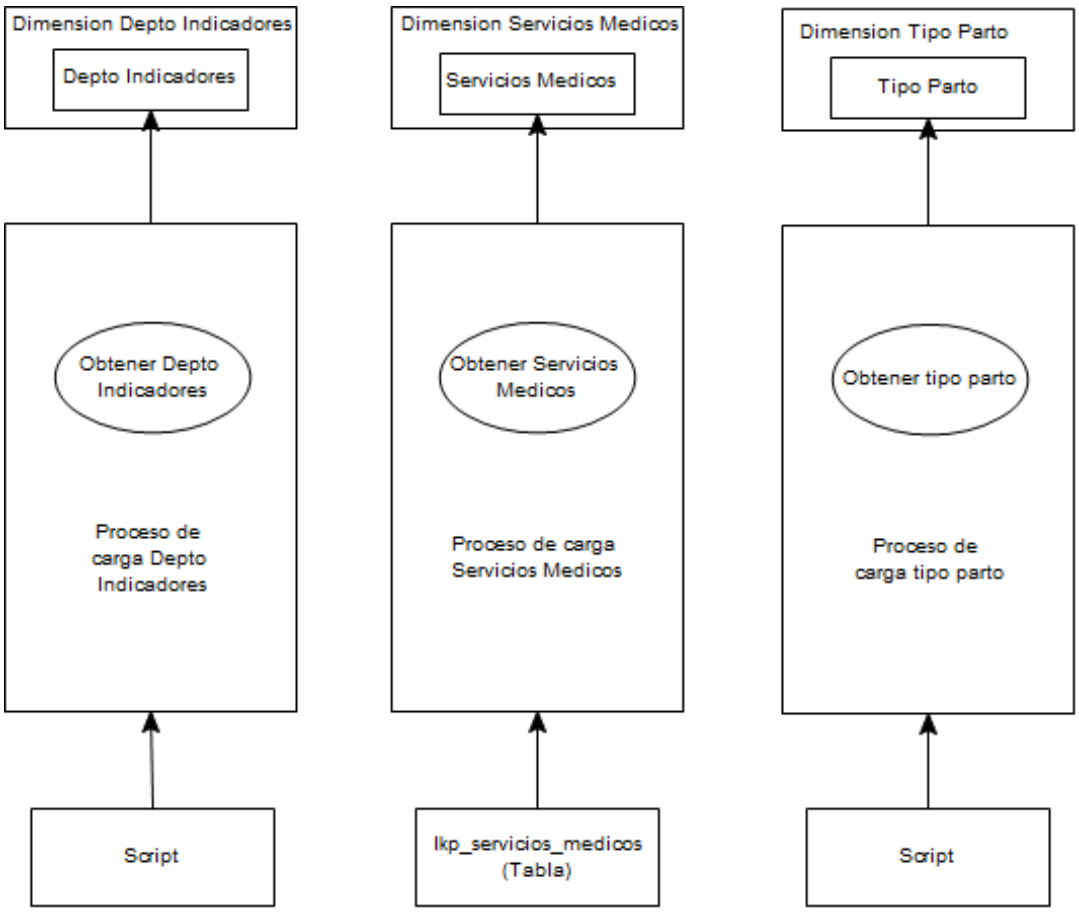


Figura 34: Diagrama de cargas de dimensiones

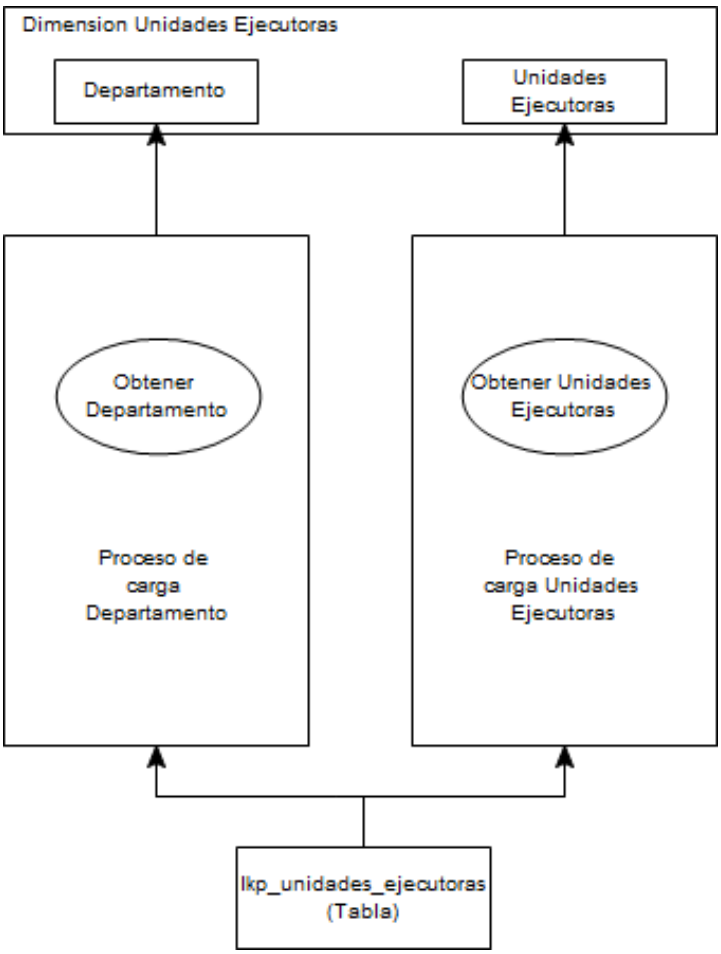


Figura 35: Diagrama de cargas de dimensiones

Algoritmo 20 Carga de la tabla de hechos H_NUMERO_PARTOS

1: **para cada tupla de la tabla prestación hacer**
2: (agrupados por sexo, la edad, el tipo de parto, centro y departamento)
3: para cada elemento de la tupla se clasifican los valores de sexo y edad
4: Se obtiene el parámetro MES_ESQUEMA que contiene la información del mes y del año de la dimensión tiempo
5: Se busca a partir del centro en LKP_CENTRO_UNIDADES_EJECUTORAS la unidad ejecutora correspondiente
6: Se ordena la tupla en función de las referencias de las dimensiones de la tabla de hechos
7: Se agrupan las tuplas
8: Se insertan en la tabla de hechos
9: **fin para**

Algoritmo 21 Carga de la tabla de hechos H_CONSULTAS

1: **para cada tupla de la tabla prestación correspondiente a una consulta hacer**
2: (agrupados por sexo, la edad, centro, departamento y consulta)
3: para cada elemento de la tupla se clasifican los valores de sexo y edad
4: Se obtiene el parámetro MES_ESQUEMA que contiene la información del mes y del año de la dimensión tiempo
5: Se busca a partir del centro en LKP_CENTRO_UNIDADES_EJECUTORAS la unidad ejecutora correspondiente
6: Se busca a partir de la consulta en LKP_SERVICIOS_MEDICOS si es servicio médico, no médico, u otro
7: **si** existe servicio de algún tipo **entonces**
8: Se ordena la tupla en función de las referencias de las dimensiones de la tabla de hechos
9: Se agrupan las tuplas
10: Se insertan en la tabla de hechos
11: **fin si**
12: **fin para**

Algoritmo 22 Carga de la tabla de hechos H_GIRO_CAMAS

1: **para cada tupla de la tabla prestación correspondientes a egresos desde los centros que tengan camas hacer**
2: Se busca a partir del centro en LKP_CENTRO_UNIDADES_EJECUTORAS la unidad ejecutora correspondiente
3: (agrupados por sexo, la edad, centro y departamento)
4: para cada elemento de la tupla se clasifican los valores de sexo y edad
5: Se obtiene el parámetro MES_ESQUEMA que contiene la información del mes y del año de la dimensión tiempo
6: Se ordena la tupla en función de las referencias de las dimensiones de la tabla de hechos
7: Se agrupan las tuplas
8: Se obtiene la cantidad de camas para la unidad ejecutora
9: Se hace el cociente entre la cantidad de prestaciones sobre la cantidad de camas
10: Se insertan en la tabla de hechos
11: **fin para**

A continuación se prensentará en las figuras 36, 37 y 38 los diagramas de carga de las tablas de hecho de numeros de parto, consultas y giro de camas.

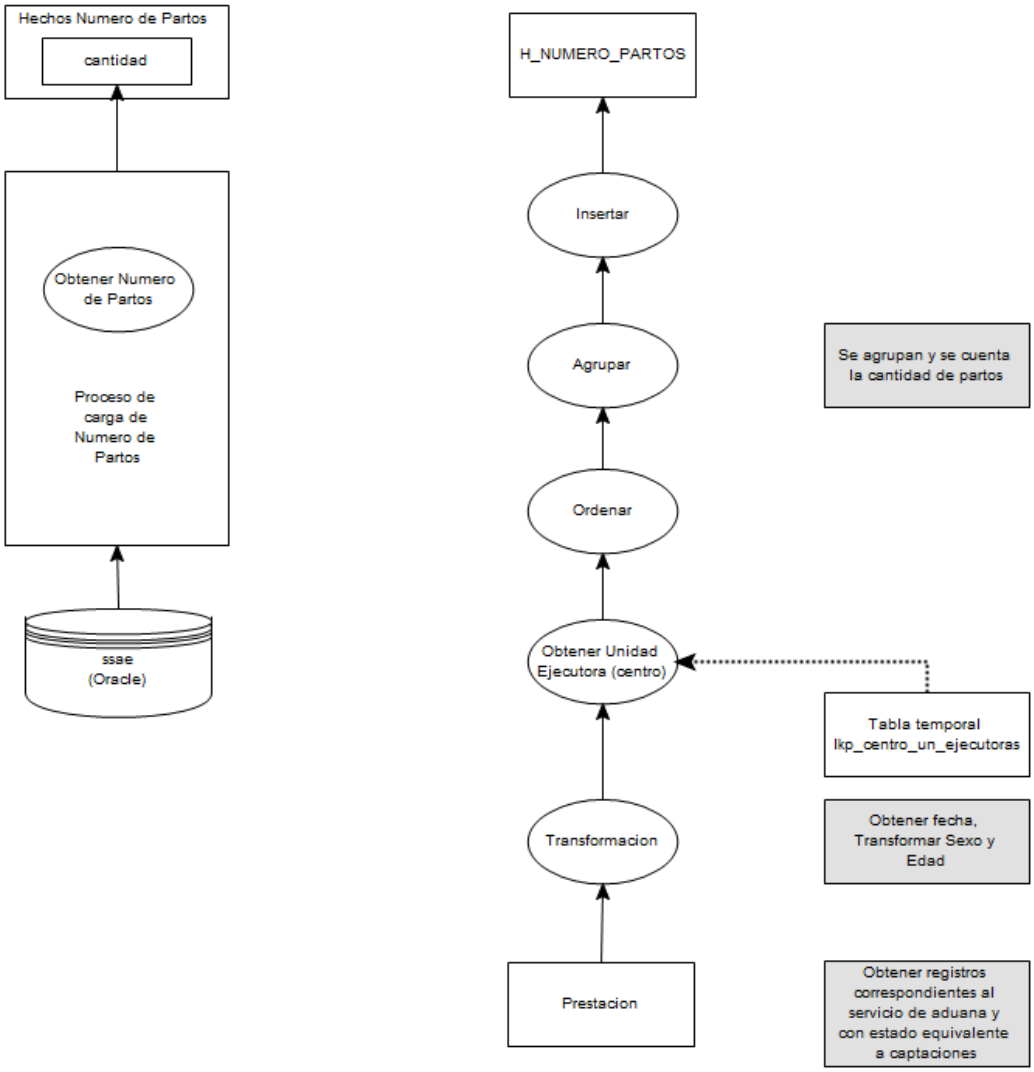


Figura 36: Diagrama de carga de hechos numero de partos

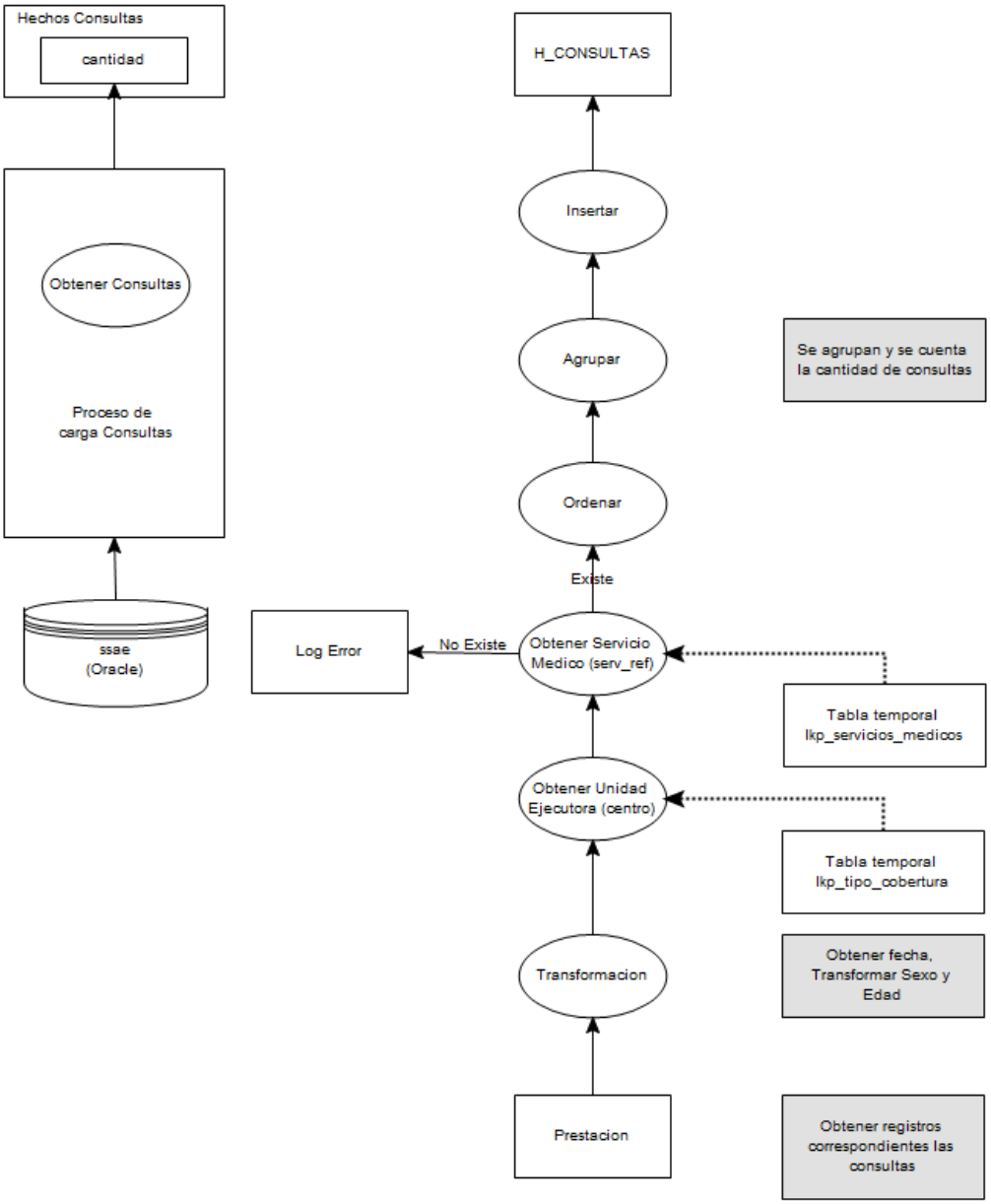


Figura 37: Diagrama de carga de hechos consultas

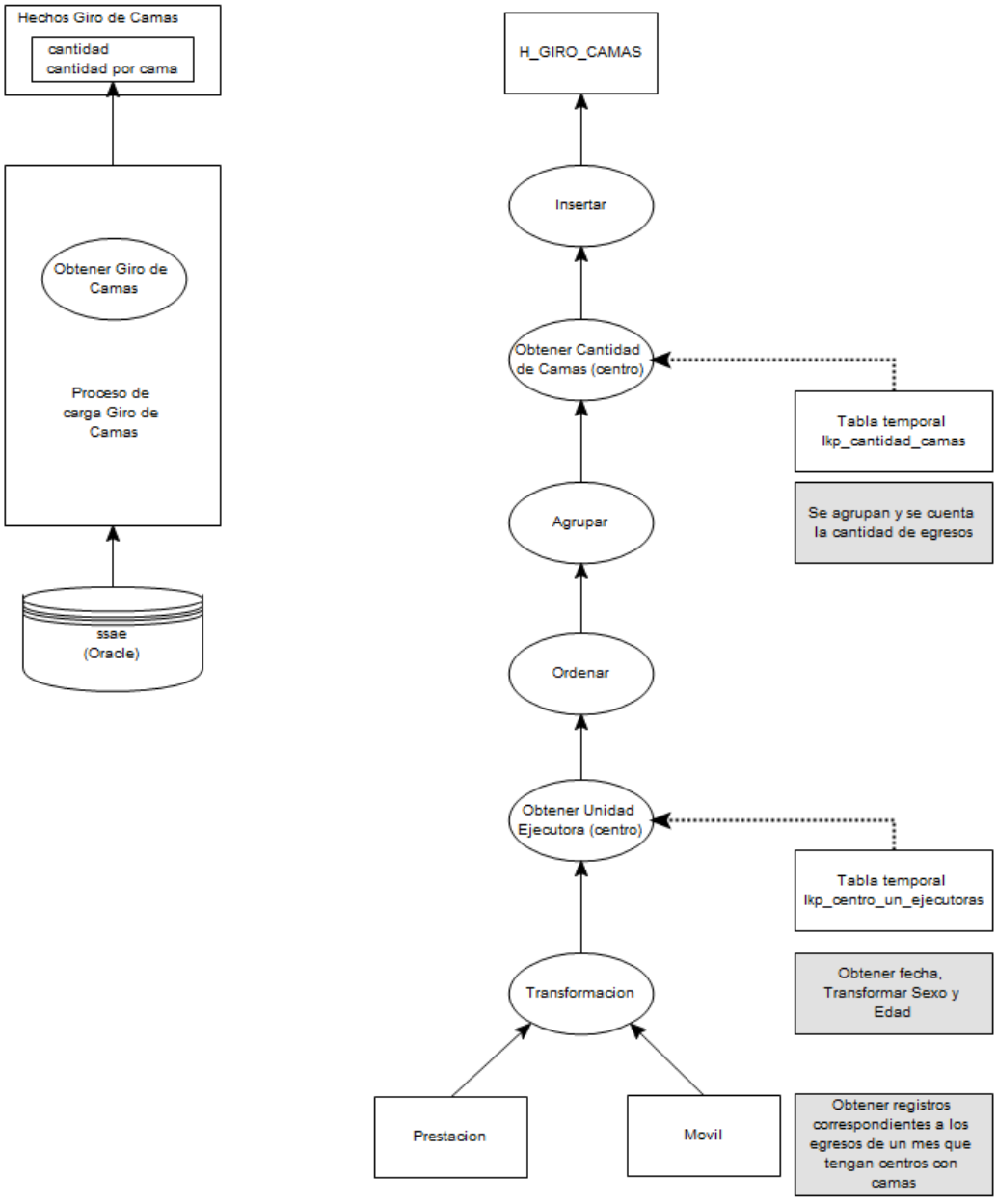


Figura 38: Diagrama de carga de hechos giro de camas

ETL

En las figuras que se presentan a continuación representan los diagramas de carga implementados en la herramienta Pentaho Data Integration. Se observa en 39 el trabajo que verifica el estado de carga de cada una de las tablas de hecho de los indicadores en un mes y año dado. Esto permite manejar tres estados en la tabla donde se controla la carga: 'Carga exitosa', 'Carga anteriormente efectuada', y 'Error en la carga'. Se encarga también de la carga y actualización de las dimensiones utilizadas, así como también de las carga de las tablas de hecho.

En las transformaciones 40, 41 y 42, se representan los algoritmos descritos anteriormente.

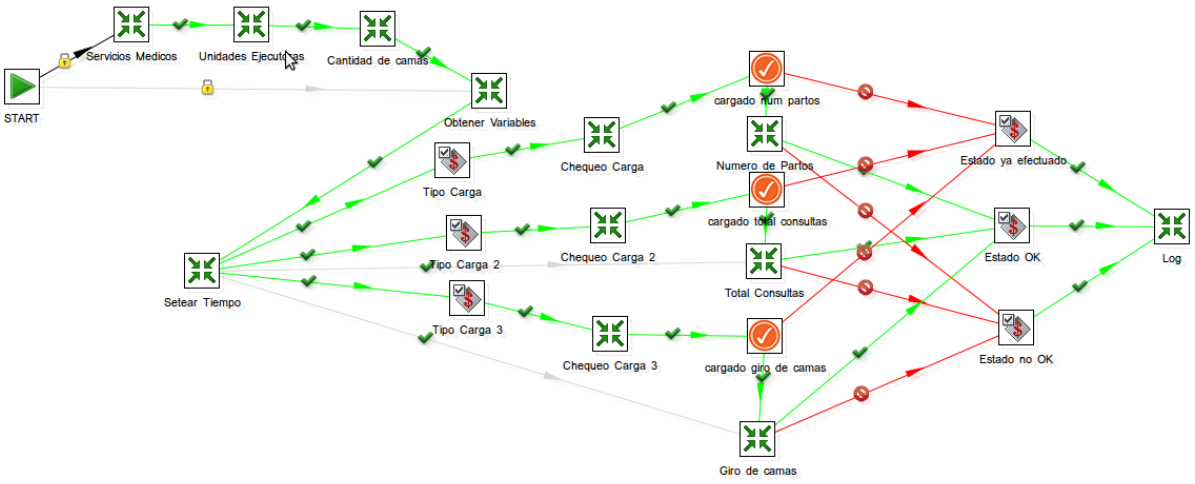


Figura 39: Trabajo de carga de los indicadores

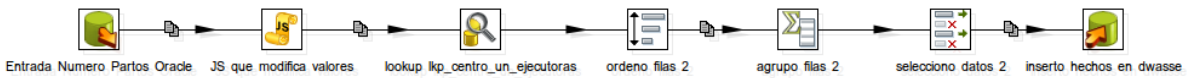


Figura 40: Transformación de la carga del numero de partos

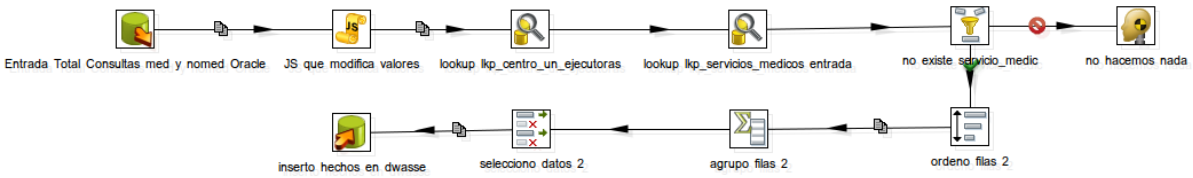


Figura 41: Transformación de la carga de las consultas

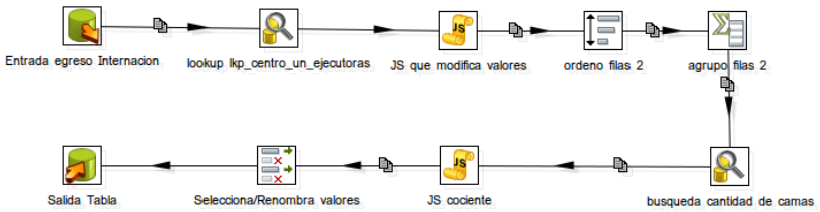


Figura 42: Transformación de la carga de los giros de cama

Tablas auxiliares

- lkp_cantidad_camas: Se carga a partir de la base de datos correspondiente al sistema AP-SGA. El fin de esta tabla es el de lograr minimizar el tiempo de ejecución del proceso de carga de la tabla de hechos H_GIRO_CAMAS perteneciente al DW. Este método permite reducir el tiempo de carga de cuatro horas a cuarenta minutos.
- lkp_centro_un_ejecutoras: Al querer relacionar los centros existentes del sistema AP-SGA con las unidades ejecutoras del sistema de Sistema de Gestión de Afiliados, no se pudo realizar ningún mapeo. Esta tabla soluciona el mapeo entre los dos sistemas.
- lkp_servicios_medicos: En el sistema AP-SGA, no está reflejado en forma correcta lo que es un servicio médico de lo que no lo es, lo cual es importante para tener un buen criterios en la toma de decisiones. El fin de esta tabla es brindar esa información, la cual es actualizada en el DW en caso de que se cambie o se agregue en un futuro un servicio médico.

Consultas Ad Hoc

Para ésta sección se implementaron varias consultas dinámicas que se corresponden con cada uno de los reportes que se detallan en la siguiente sección, éstas consultas son

vistas de análisis en donde se puede agregar y desagregar dimensiones así como agregar o quitar filtros de diferentes tipos.

Reportes

Para esta sección se implementaron varios reportes que se consideraron podían ser de utilidad dados los requerimientos, para una mayor comprensión se agruparon en 3 grupos, los reportes relativos a las consultas médicas y no médicas, los reportes relacionados con los números totales de partos y los reportes de giros de camas:

- Reportes relativos a los totales de consultas médicas y no médicas
  - Número total de consultas médicas y no médicas por especialidad, según grupo de Edad Fonasa
  - Número total de consultas médicas y no médicas por tipo de consulta, según sexo y grupo de edad Fonasa
  - Número total de consultas médicas y no médicas por Grupo de Edad Infamilia, según departamento de residencia del afiliado
  - Número total de consultas médicas y no médicas por Grupo de Edad Infamilia, según unidad ejecutora correspondiente a la prestación

- Reportes relativos a los totales de egresos del mes sobre el total de días
  - Número total de egresos del mes sobre el total de días

- Reportes relativos al número total de partos
  - Número total de partos informados y registrados por tipo de parto según sexo del hijo y Grupo de Edad Fonasa de la madre
  - Número total de partos informados y registrados por Grupo de Edad Infamilia, según departamento de residencia de la madre
  - Número total de partos informados y registrados por Grupo de Edad Infamilia, según unidad ejecutora correspondiente a la prestación
  - Número total de partos informados y registrados por tipo de parto, según departamento de residencia de la madre
  - Número total de partos informados y registrados por tipo de parto, según unidad ejecutora correspondiente a la prestación

A continuación se listan algunas de las consultas médicas y no médicas que aparecen en estos reportes:

- Consultas médicas
  - Alergista

- Anestesista
- Cardiología
- Cirugía
- Cirugía Pediátrica
- Cirugía Plástica
- Cirugía Vascular
- Dermatología
- Diabetología
- Endocrinología
- Fisiatría
- Gastroenterología
- Genética
- Geriatría
- Ginecología
- Hematología
- Internista
- Medicina General
- Médico de Area
- Médico de Familia
- Microcirugía
- Nefrología
- Neumología
- Neurocirugía
- Neurología
- Neuropediatría
- Neuropsiquiatría
- Otorrinolaringología
- Pediatría
- Psiquiatría
- Psiquiatría Infantil
- Reumatología
- Traumatología
- Urología

- Consultas no médicas
  - Asistencia Integral
  - Asistente Social

- Curaciones
- Enfermería
- Fisioterapia
- Fondo de Ojos
- Fonoaudiología
- Higienista Dental
- Holter
- Nutricionista
- Odontología
- Partera
- Podólogo
- Psicología
- Psicomotricista

- Mal definido / no aplica
  - Actividad Comunitaria
  - Actualización de Saldos
  - Adolescencia
  - Adquisición de Insumos Médicos
  - Colposcopía
  - Cont. Quimicos
  - Despacho
  - Eco Doppler
  - Ecografía
  - Electrocardiograma
  - Electroencefalograma
  - Emergencia
  - Emergencia en Puerta
  - Ergometría
  - Funcional Respiratorio
  - Identificación de Usuarios
  - Internación
  - Laboratorio
  - Oftalmología
  - Portal Amarillo
  - Programa Aduana
  - Programa Infancia y Familia

- Programa Seguimiento Nutricional
- Radiología
- Reposición de Farmacias
- Suministro a Crónicos
- Suministro a Servicios
- Tabaquismo
- Toxicología

Anexo 5: Tutorial de implementación de un DW

Objetivos

El objetivo de este capítulo es trasmitir nuestros conocimientos adquiridos a lo largo del proyecto con el fin de que este DW pueda ser mantenido a futuro por personal técnico de ASSE, de manera que puedan extenderlo agregando nuevos cubos, procesos de ETL, reportes, cuadros de mando, etc.

Construir un cubo con Schema Workbench

En esta sección se intenta mostrar paso a paso como construir un cubo con el editor de cubos de Pentaho: Schema Workbench 3.2.1.13885. El motor de cubos de Pentaho se llama Mondrian, es el encargado de traducir las consultas MDX a SQL.

Tanto Mondrian como Schema Workbench están disponibles en sourceforge. En este tutorial se utilizará una base de datos MySQL instalada en el servidor "my2".

Definir una conexión

Lo primero que hay que hacer es definir una conexión. Para esto vamos al menú Options, damos click en Connection y vamos a ver las propiedades de la conexión a la base de datos. En la pestaña General definimos las siguientes propiedades con sus respectivos valores:

- **Connection Name:** "my2"
- **Connection Type:** "MySQL"
- **Host Name:** "my2"

- **Database Name:** "dw"
- **Port Number:** 3306
- **User Name:** "root"
- **Password:** "admin1234"
- **Access:** Native(JDBC)

Se puede testear la conexión dando click en el botón "Test". En caso de que la conexión haya quedado funcionando ya podemos acceder a la base de datos para ver la estructura de las tablas, para esto vamos al menú File, opción New - JDBC Explorer y ahí vemos la estructura de la base de datos que acabamos de configurar.

A modo de ejemplo vamos a crear un cubo que contenga la siguiente información: Cantidad de afiliados de ASSE, agrupados por Mes, Sexo y Grupo de Edad.

Vamos a construir un cubo utilizando como tabla de hechos H_ASSE y como dimensiones las tablas D_TIEMPO, D_SEXO y D_GRUPO_EDAD_INFAMILIA.

Crear un esquema

Los cubos se definen dentro de un esquema. Un esquema es capaz de contener varios cubos pero solo los cubos que pertenecen a un mismo esquema son capaces de compartir dimensiones.

Para realizar esto vamos al menú File, opción New - Schema, damos click en "Schema" y le ponemos un nombre. Con esto nos ha quedado creado el esquema donde luego agregaremos el cubo.

Las dimensiones con información temporal y geográfica suelen ser buenas candidatas para dimensiones compartidas. Un esquema puede contener cubos y dimensiones, pero también podrá contener cubos y dimensiones virtuales. Un cubo virtual es un cubo construido en base a otros cubos, al igual que una dimensión virtual se construye en base a otras dimensiones ya existentes. Dentro de un cubo pueden definirse dimensiones privadas que no serán compartidas por otros cubos, dimensiones públicas, métricas y miembros calculados.

Crear el cubo y asociar la tabla de hechos

Para crear el cubo damos click derecho sobre "Schema" y seleccionamos "Add cube". Le ponemos un nombre al cubo, por ejemplo "CuboASSE". Luego damos click derecho en "CuboASSE" y seleccionamos "Add Table". En el cubo hay que especificar cual va a ser

la tabla de hechos, es decir, la tabla que va a contener las variables sobre las cuales queremos obtener información. En el campo name ponemos el nombre de la tabla que usaremos como tabla de hechos, en este ejemplo pondremos "H_ASSE".

Ahora tenemos que agregar las dimensiones al cubo. Las dimensiones son los posibles ejes de análisis o las variables por las que vamos a querer filtrar los datos del cubo a ser mostrados.

Agregar dimensiones

Comenzaremos creando la dimensión Sexo. Para esto damos click derecho sobre el cubo y seleccionamos "Add Dimension". Se agregará un nuevo nodo dimensión asociado al cubo que creamos. Le ponemos un nombre a la dimensión y luego tenemos que especificar el campo "foreignKey" correspondiente a la clave foranea a ser utilizada en esta dimensión, en el ejemplo pondremos "sexo_ref", este campo es el atributo ubicado en la tabla "H_ASSE" y es el campo correspondiente al género del afiliado en la tabla de hechos.

Agregar jerarquías y niveles

Luego crearemos la jerarquía asociada a la dimensión, para esto damos click derecho sobre la dimensión y seleccionamos "Add Hierarchy", le ponemos un nombre a la jerarquía y seteamos un valor en el campo primaryKey, este valor hará referencia a la clave primaria de la tabla de dimensión que estamos asociando, en nuestro caso pondremos el valor "sexo_key". Ahora tenemos que agregar un nivel a la jerarquía, nuevamente damos click derecho sobre la jerarquía y seleccionamos "Add Level", con esto quedará creado el nivel que irá asociado a la jerarquía.

Una vez creado el nivel le ponemos un nombre y un valor en el campo "column", este valor corresponde a la descripción de la dimensión que queremos que se muestre en el cubo, escribimos en este campo el nombre atributo, en nuestro caso será "sexo_nombre". Finalmente debemos asociar la tabla dimensión al nivel de la jerarquia, así que damos click sobre el nodo Table que está asociado a la jerarquía y escribimos en el campo "name" el valor "D_SEXO" que corresponde a la tabla de dimensión. Posteriormente será necesario crear las dimensiones restantes a ser utilizadas en el cubo.

Agregar medidas

Luego que tenemos creada las dimensiones resta agregar la medida de nuestro cubo. Para esto damos click derecho sobre el cubo y seleccionamos "Add Measure", con esto

se agregará un nuevo nodo debajo del cubo la cual le tendremos que poner un nombre, por ejemplo ”cantidad”. Este nodo corresponde al hecho en sí mismo y será lo que querrá ser contado en el cubo, en nuestro ejemplo es la cantidad de afiliados. En el campo ”aggregator” seleccionamos la operación que se realizará sobre esta cantidad, en nuestro ejemplo será ”sum” y por último en el campo ”column” ponemos el nombre del atributo en la tabla de hechos.

Publicación

Finalmente, cuando se han creado las dimensiones y las medidas necesarias procedemos a guardar el archivo generado el cual es un archivo xml, así que vamos al menú File, opción Save, le ponemos un nombre y lo guardamos en algún lado del disco. A continuación haremos la publicación del cubo, en caso de que aún no se haya hecho hay que configurar un password en Pentaho para publicar schemas, para esto debemos editar el archivo ”./biserver-ce/pentaho-solutions/system/publisher_config.xml”.

Acá buscamos lo siguiente:
<publisher-config>
<publisher-password></publisher-password>
</publisher-config>

Definimos el password que queramos entre ”<publisherpassword>” y ”</publisher-password>”

Por ejemplo: ”<publisher-password>password</publisher-password>”

Guardamos el archivo .xml.

Ahora vamos al menú File opción Publish, nos va a pedir el password que acabamos de configurar arriba y una cuenta de acceso a Pentaho, si no se modificaron las cuentas por defecto se puede usar como usuario ”Joe” y contraseña ”password”.

Luego seleccionamos la carpeta donde queremos realizar la publicación, en el campo ”Pentaho or JNDI Data Source” escribimos el nombre del datasource que creamos antes, en nuestro caso sería ”my2”, damos click en ”Publish” y si nos dice ”Publish Successfull” quiere decir que tenemos el cubo exitosamente publicado.

Elaborar procesos de ETL's

En ésta sección se pretende describir a grandes rasgos como crear un sencillo proceso de ETL para cargar el cubo que creamos en la sección anterior.

Para empezar debemos ejecutar Pentaho Data Integration. Luego crearemos la transformación principal que será la encargada de realizar la carga del DW. Para esto vamos al menú Fichero - Nuevo, opción Transformación, escribimos un nombre para la Transformación, por ejemplo "CargaDW". Empezaremos por insertar un nodo de tipo Entrada para la lectura de los datos de origen, para esto seleccionamos el nodo "Entrada Tabla" y lo insertamos en la nueva Transformación.

Una vez que tenemos el nodo de Entrada damos doble click sobre el mismo y vamos a configurar los parámetros de conexión, para esto damos click en el botón "Nuevo" que se encuentra a la derecha del combo "Conexión". Se nos abrirá una ventana donde configuraremos la conexión a la base de datos de entrada. Le ponemos un nombre a la conexión, por ejemplo "Entrada" y configuramos los siguientes valores en los respectivos campos:

- **Connection Type:** "PostgreSQL"
- **Host Name:** "localhost"
- **Database Name:** "dw"
- **Port Number:** 5432
- **User Name:** "postgres"
- **Password:** "postgres"
- **Access:** Native(JDBC)

Una vez configurados estos parámetros damos click en el botón "Probar", si la prueba es satisfactoria significa que ha quedado bien configurada la nueva conexión de entrada. Damos click en "Ok" y en el combo de Conexión seleccionamos la nueva conexión que acabamos de crear.

Luego de haber creado la conexión a la base de datos de origen debemos escribir la consulta SQL que utilizaremos para obtener los datos que luego serán cargados en el cubo. A continuación mostramos un ejemplo de consulta SQL muy sencilla que puede ser utilizada para éste propósito:

```
select trim(per_sexo) sexo,
coalesce(FLOOR(((DATE_PART('YEAR',current_timestamp)
-DATE_PART('YEAR',p.per_fecha_nac))* 372
+ (DATE_PART('MONTH',current_timestamp)
- DATE_PART('MONTH',p.per_fecha_nac))*31
+ (DATE_PART('DAY',current_timestamp)
-DATE_PART('DAY',p.per_fecha_nac)))/372),-1)
```

```
edad, count(*) cantidad
from personas p
group by sexo,edad
order by sexo,edad;
```

Como se puede ver en ésta consulta, se selecciona el sexo de cada persona, se calcula la edad, se agrupa por estos 2 criterios y se calcula la cantidad de personas que entran en cada categoría que corresponde a cada agrupación posible de sexo y edad.

A continuación vamos a insertar un nodo "Valor Java Script Modificado" de la carpeta Scripting en la pestaña Desing. Luego de insertar el nodo lo conectamos a la entrada de la tabla. Éste nodo lo utilizaremos para transformar y agregar algunos datos previos a la carga del cubo. Luego de haberlo insertado en nuestra Transformación le damos doble click y se nos abre una ventana donde escribiremos el siguiente código JavaScript:

```
var tiempo = 201107;
if (sexo == 'M' || sexo == 1)
sexo = 1;
else if(sexo == 'F' || sexo == 2)
sexo = 2;
else
sexo = 3;
var sexo = sexo;

if (edad < 0 || edad > 115)
edad = -1;
else
    edad = edad;
var edad = edad;
```

Con éste código primeramente lo que hacemos es poner un valor cualquiera para el mes y para el año simplemente a modo de ejemplo, en realidad este valor debería ser tomado del esquema seleccionado en la base de datos de origen. Por otro lado se establece un formato único para el sexo de la persona, los únicos valores que insertamos en el campo sexo son 1, 2 o 3 para los casos Masculino, Femenino y no indicado respectivamente. Finalmente filtramos los casos en los cuales por error se haya calculado que la edad de la persona es menor a 0 o mayor a 115 en cuyos casos se le pone valor -1.

Luego de escribir ese código en el campo "Java script:" presionamos el botón "Obtener Variables", esto inferirá las variables utilizadas en el java script junto con el tipo asociado. Por último presionamos "Ok" y volvemos a la transformación en la cual estabamos trabajando.

A continuación vamos a insertar un nodo de tipo "Transformar" llamado "Ordenar filas" y lo conectamos al nodo anterior Java Script, éste nodo ordenará las filas según nuestro criterio de agrupación, así que lo abrimos y para nuestro ejemplo seleccionamos

en la sección "Campos" los atributos sexo y edad en ese mismo orden y presionamos "Ok" para volver a la transformación.

Ahora agregaremos un nodo "Agrupar filas" y lo conectamos con el nodo "Ordenar filas" anterior. En éste nodo agruparemos las filas según el criterio que estamos utilizando por lo tanto abrimos el nodo y en la sección "Campos que forman la agrupación" agregamos los atributos "sexo", luego "edad" y por último "tiempo". En la sección inferior llamada "Agregados" insertamos el atributo "cantidad" y en tipo de operación seleccionamos "Suma" dado que lo que queremos realizar es la suma de la cantidad de afiliados que se encuentra en cada agrupación de sexo, edad y mes. Presionamos "Ok".

Luego vamos a crear la conexión de Salida que será la que insertará los datos en el cubo, para esto agregamos un nodo "Salida Tabla" a nuestra Transformación, le ponemos un nombre, por ejemplo "Salida" y configuramos los siguientes parámetros:

- **Connection Type:** "MySQL"
- **Host Name:** "my2"
- **Database Name:** "dw"
- **Port Number:** 3306
- **User Name:** "root"
- **Password:** "nomaschucu"
- **Access:** Native(JDBC)

Luego presionamos el botón "Probar" para chequear que haya quedado bien configurada la conexión y presionamos "OK". A continuación debemos especificar en el campo "Tabla destino" la tabla en la cual se insertarán los datos y corresponderá a la tabla de hechos que en nuestro caso es H_ASSE.

Finalmente insertamos un nodo llamado "Selecciona/renombra valores" ubicado en la carpeta "Transformar". Luego de insertarlo en la transformación lo conectamos entr el nodo "Agrupar filas" anteriormente insertado y el nodo de salida a la base de datos donde está ubicado el cubo. Abrimos éste último nodo y realizamos el siguiente mapeo entre los campos que estamos utilizando y los atributos de la tabla destino:

```
sexo   --> sexo_ref
edad   --> ge_edad
tiempo --> t_ref
```

Una vez que hemos terminado de agregar los nodos vamos a correr la transformación. Para esto presionamos el botón Play y luego damos click en "Ejecutar". Esto ejecutara la transformación que realiza la carga del cubo.

Análisis OLAP

Una vez finalizada la carga del DW ya se puede realizar un análisis OLAP para probar el cubo. Para esto entramos en un navegador a la dirección http://localhost:8080/. Nos aparecerá la pantalla de bienvenida a Pentaho con un botón que nos permitirá loguearnos. Damos click en "Pentaho User Console Login" y podemos acceder con algún usuario que esté definido, por ejemplo "adminasse1" contraseña "adminasse1". Luego de loguearse nos aparecerán varios botones, damos click en "Nueva Vista de Análisis", seleccionamos el esquema y el cubo que hemos publicado, en nuestro caso sería esquema "CuboASSE" y cubo "CuboASSE" y presionamos "Ok".

A continuación se nos cargará una vista de análisis con los filtros por defecto, los cuales podremos cambiar y reorganizar como deseemos. En la barra de herramientas superior se presentan varios botones, en particular hay uno que dice "Abrir navegador OLAP", éste nos permite manipular las dimensiones en filas y columnas según nuestra conveniencia, por ejemplo una vista deseada puede ser mostrar los grupos de edades en columnas y los sexos en filas de forma que se muestren las afiliaciones por sexo según grupo de edad. Luego de armar el cubo presionamos "Aplicar" y podemos ver la vista de análisis creada. Esto se puede guardar fácilmente para un análisis futuro.

Pentaho Report Designer

En ésta sección presentaremos una forma de diseñar reportes con Pentaho Report Designer. Lógicamente lo primero que debemos hacer es ejecutar Pentaho Report Designer. Una vez abierto vamos al menú File - Report Wizard, seleccionamos un template y presionamos Next, configuramos el data source apuntando a nuestro cubo. A continuación debemos seleccionar los items que deseamos mostrar en el reporte y clickeamos en Next nuevamente. Por último tenemos la opción de cambiar los titulos de los campos seleccionados así como el ancho de cada columna y la alineación.

Para finalizar damos click en "Finish". Se nos mostrará a continuación un esquema gráfico del reporte donde podremos editar las filas y las columnas, colores, fuente y tamaño de letra, título del reporte y otras configuraciones. Una vez que tenemos el reporte deseado vamos al menú File y seleccionamos "Publish". Escribimos la url del servidor que en nuestro caso será "http://localhost:8080/pentaho". En "Pentaho Credentials" ponemos el mismo usuario y contraseña que utilizamos para loguearnos en el sitio y damos "OK". Seleccionamos nombre de archivo, título, descripción, ubicación y tipo de salida para el reporte. En el campo "Publish Password" pondremos la contraseña que utilizamos para la publicación del cubo. Finalmente presionamos "Ok" y quedará publicado el reporte el cual podremos ver entrando a la consola en la ubicación que hayamos seleccionado anteriormente.

Glosario

BI Business Intelligence

DW Data Warehouse

DWOF Data Warehouse Open Free

OLAP Procesamiento Analítico en Linea (eng: Online Analytical Processing)

ROLAP Procesamiento Analítico en Linea Relacional (eng: Relational Online Analytical Processing)

MOLAP Procesamiento Analítico en Linea Multidimensional (eng: Multidimentional Online Analytical Processing)

HOLAP Procesamiento Analítico en Linea Híbrido (eng: Hybrid Online Analytical Processing)

FOSS Free/Open Source Software

ETL Extracción, Transformación y Carga (eng: Extraction-Transformation-Load)

DS Data Source, orígen de datos.

RDBMS Relational Database Management System

Data Marts Subconjunto físico o lógico de datos dentro del data wearehouse.

Bibliografía

[1] Matteo Golfarelli. Open source bi platforms: A functional and architectural comparison. *Lecture Notes in Computer Science*, 5691/2009, 2009.

[2] William H.Inmon. *Building the data warehouse.* Wiley, fourth edition edition, 2005.

[3] Joe Caserta Ralph Kimball. *The data warehouse ETL Toolkit: Practical techniques for extracting, cleaning and delivering data.* Wiley Computer Publishing, first edition edition, 2004.

[4] Margy Ross Ralph Kimball. *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling.* Wiley Computer Publishing, second edition edition, 2002.

[5] Umeshwar Dayal Surajit Chaudhuri. An overview of data warehousing and olap technology. Volume 26 Issue 1, March 1997.

[6] Alan R.Simon Thomas C. Hammergren. *Data Warehousing for Dummies.* Wiley, second edition edition, 2009.

[7] `http://api.ning.com/files/RLZMtjtGdOkmyjoovQhnUaxUO5sC7AwcuaBAYNQO5HE_/howtopentaho3.5ycubomondrianengnulinux.pdf`. Pasos para crear cubos con mondrian schema workbench. Ultimo acceso 05/2011.

[8] `http://cdf-de.googlecode.com/files/CDE-bundle-1.0-RC3.tar.bz2`. Pentaho dashboard editor cde bundle 1.0 rc3. Ultimo acceso 03/2011.

[9] `http://code.google.com/p/cdf-de/`. Referencia a community dashboard editor. *WebDetails - www.webdetails.pt.* Ultimo acceso 05/2011.

[10] `http://code.google.com/p/pentaho-cdf/downloads/detail?name=pentaho-cdf-bundle-3.2.zip`. Pentaho dashboard frmawork cdf bundle 3.2. Ultimo acceso 03/2011.

[11] `http://community.pentaho.com/`. Pentaho community web site. Ultimo acceso 03/2011.

[12] `http://mondrian.pentaho.com/documentation/schema.php`. Mondrian documentation. Ultimo acceso 05/2011.

[13] `http://sourceforge.net/projects/jfreereport/files/04.%20Report%20Designer/`. Pentaho report designer 3.7.0 stable. Ultimo acceso 03/2011.

[14] `http://sourceforge.net/projects/mondrian/files/schema%20workbench/3.2.1-stable/`http://sourceforge.net/projects/mondrian/files/schema%20workbench/3.2.1-stable/psw-ce-3.2.1.13885.tar.gz/download. Schema workbench 3.2.1-stable. Ultimo acceso 03/2011.

[15] http://sourceforge.net/projects/pentaho/files/Business%20Intelligence%20Server/3.7.0-stable/. Pentaho bi platform 3.7.0 estable. Ultimo acceso 03/2011.

[16] http://sourceforge.net/projects/pentaho/files/Data%20Integration/4.1.0-stable/. Pentaho data integration 4.1.0 estable. Ultimo acceso 03/2011.

[17] http://sourceforge.net/projects/pentaho/files/Design%20Studio/3.7.0-stable/pds-ce-linux-64-3.7.0-stable.tar.gz/download. Pentaho design studio 3.7.0 stable. Ultimo acceso 03/2011.

[18] http://wiki.pentaho.com/display/Reporting/Report+Designer. Pentaho report designer documentation. Ultimo acceso 05/2011.

[19] http://wiki.pentaho.com/display/Reporting/Report+Designer. Referencia a cron de unix. Ultimo acceso 05/2011.

[20] http://www.asse.com.uy/uc_2113_1.html. Objetivos institucionales de asse. 2011.

[21] http://www.fing.edu.uy/inco/cursos/caldatos/teorico.php. Calidad de datos 2010, fing, udelar. 2010.

[22] http://www.fing.edu.uy/inco/cursos/disDW/. Diseño y construcción de dw 2011, fing, udelar. 2011.

[23] http://www.inkscape.org/. Inkscape 0.48 illustrator. Ultimo acceso 05/2011.