Almacenes de datos – Análisis de becas Erasmus

Ana Belén Morales del Castillo
Grado de Ingeniería Informática

Juan Vidal Gil

15/01/16

FICHA DEL TRABAJO FINAL

Título del trabajo:	Data warehouse
Nombre del autor:	**Ana Belén Morales del Castillo**
Nombre del consultor:	**Juan Vidal Gil**
Fecha de entrega (mm/aaaa):	01/2016
Área del Trabajo Final:	**Ingeniería del Software**
Titulación:	*Grado de Ingeniería Informática*

Resumen del Trabajo (máximo 250 palabras):

Actualmente, la información de becas concedidas a estudiantes en el programa Erasmus se encuentra dispersa en diversos ficheros que además tienen distintos formatos. Esto hace que el análisis de la información sea complicado, ya que no se pueden hacer agregaciones, comparaciones y búsquedas de una forma rápida y sencilla. Además, estos ficheros contienen datos erróneos, por lo que no se pueden obtener conclusiones certeras a través de su análisis.

En este Trabajo Fin de Grado, se ha desarrollado un sistema que integra toda la información del programa en una única ubicación. Tras desarrollar este almacén de datos, se ha procedido a la carga de la información de becas en dicho almacén desarrollando un proceso ETL que valida y homogeneiza dichos datos para obtener información de calidad a través de los cuales poder inferir decisiones correctas.

Para que la exploración de los datos y el análisis de los mismos sea más sencillo y rápido, se ha diseñado un cubo OLAP que explota todas las dimensiones del esquema de datos, así como una serie de informes libres que permite a los usuarios pertenecientes a la alta dirección del programa, explorar los datos de manera libre. Por último, también se proporcionan una serie de informes estáticos que ofrece la información que necesitan los usuarios gestores del programa.

En conclusión, mediante la utilización de este sistema se simplifica la gestión de la información con objetivos de análisis y se ofrece un interfaz ágil y amigable para que los usuarios puedan explotar dicha información.

Abstract (in English, 250 words or less):

Nowadays, the Erasmus program provides the information about students grants distributed in several files in different formats. This makes it difficult to analyze this information because common operations such as aggregations, comparisons or searches, can not be easily performed. Furthermore, these files can contain wrong data. Therefore, analyzing this data will not ensure an accurate conclusion.

During this Final Project a system that integrates all the information of the Erasmus program, in one single location, has been developed. Once the data warehouse has been implemented, we proceeded to load the information into it. This has been achieved using an ETL process that validates and formats the data, providing accurate data to populate the data warehouse and, therefore, allowing accurate conclusions.

An OLAP cube has also been implemented, so browsing and analyzing the data can be done fast and easy. In this OLAP cube all the dimensions implemented in the data schema are present. In addition, several dynamics reports, that allow executives to explore data freely, are available. Finally, several static reports are provided so managers can analyze the information they need.

In conclusion, this system simplifies the management of data for analytic purposes and provides and agile and friendly interface so that users can explore the information.

Palabras clave (entre 4 y 8):

Data warehouse, almacén de datos, business intelligence, BI, ETL, reporting

Índice

Lista de figuras

1. Introducción

A continuación se proporciona una explicación a alto nivel de los objetivos y metodología utilizados para realizar este proyecto.

1.1 Contexto y justificación del Trabajo

Actualmente el programa **Erasmus** forma parte de la comunidad **Open Data**, de manera que proporcionan la información del programa al público en general. Esta información se encuentra diseminada en distintos ficheros que se ofrecen en diversos formatos (excel y CSV):

- un fichero con la información de movilidad de los estudiantes por curso académico
- un fichero con la información de las organizaciones
- un fichero con la información de los países que participan en el programa
- otro con las áreas de estudio

Además, falta alguna información necesaria para interpretar los datos de movilidad de estudiantes (como los idiomas estudiados, por ejemplo).

Así pues, al encontrarse la información tan diseminada y con diferentes formatos de un curso académico a otro, se hace muy difícil navegar por la misma, hacer búsquedas, comparaciones, etc.

El sistema que se desea obtener en este proyecto, deberá ofrecer una herramienta que centralice toda la información disponible y proporcione un interfaz sencillo de utilizar y amigable que permita explotar los datos, de manera que los profesionales que trabajan en el programa Erasmus puedan obtener conclusiones interesantes a partir de ella con poco esfuerzo.

1.2 Objetivos del Trabajo

El **objetivo general** de este Trabajo Fin de Grado es simplificar las tareas que realizan los profesionales del programa Erasmus ofreciendo una herramienta que permita explotar los datos de distintas formas, adaptándose a sus necesidades particulares.

Para ello, es necesario unificar los datos y almacenarlos en un único almacén. Para que las conclusiones inferidas a través de estos datos sean correctas, dichos datos deben de someterse a un proceso de refinamiento y homogeneización mediante un proceso ETL.

Así mismo, se deben de proporcionar dos modalidades de análisis de la información:

- exploración de los datos de manera libre: mediante informes dinámicos que proporcionen análisis multidimensional de la información

- informes estáticos rápidos de cargar con información predefinida y que se consultarán periódicamente.

Adicionalmente, se pueden identificar los siguientes **objetivos de aprendizaje**:

- Aprender a diseñar e implementar un base de datos en estrella orientada al análisis OLAP.

- Aprender a utilizar el software de Pentaho para la creación del proceso ETL y de los informes.

- Repasar y aplicar conceptos sobre gestión y planificación de proyectos informáticos estudiados durante el grado.

- Repasar y aplicar conceptos de análisis y diseño de aplicaciones estudiados durante el grado.

1.3 Enfoque y método seguido

Como hemos visto, actualmente se dispone de distintos archivos con la información del programa Erasmus. En cada uno de estos archivos se encuentra información de distintos aspectos del proyecto.

Para conseguir el objetivo general del presente Trabajo Fin de Grado, se pueden plantear varias alternativas: crear una **base de datos operacional** o bien, un **almacén de datos**. A continuación se presenta una tabla comparativa de ambas tecnologías y se comenta cómo se ajustan éstas a las necesidades de nuestro proyecto. Además, se han sombreado las opciones que son más interesantes para nuestro proyecto para poder identificarlas más fácilmente.

Característica	Sistemas operacionales	Almacén de datos	Comentarios
ALMACENAMIENTO, DISEÑO Y ESTRUCTURA DE LOS DATOS			
Temporalidad de los datos	1 a 2 años	5 a 10 años	Necesitamos almacenar datos durante largos períodos de tiempo para poder detectar tendencias a lo largo del tiempo.
Volumen de las transacciones	Relativamente pequeño	Alto	En nuestro caso el volumen de datos de las transacciones será alto, ya que se harán consultas sobre grandes cantidades de datos, y normalmente se solicitarán agregados.
Nivel de agregación	Bajo	Puede haber varios niveles	Distintos niveles de datos nos permite agrupar los datos según distintos criterios, lo cuál es también útil para realizar comparativas y detectar tendencias.
Actualización	Constante	Periódica	Ya que no necesitamos tener los datos actualizados hasta el último momento, basta con establecer unas actualizaciones periódicas. Además, al tener una gran cantidad de datos, estas actualizaciones pueden tardar mucho (si hay que recalcular tablas agregadas, etc.), por lo que no tiene sentido hacerlas de manera constante.
Estructuración de los datos	Relacional	Visión multidimensional	En particular, para los usuarios con roles estratégicos (alta dirección) es muy interesante disponer de una visión multidimensional que favorezca la exploración de los datos desde distintos puntos de vista.
TRATAMIENTO DE LA INFORMACIÓN			

Característica	Sistemas operacionales	Almacén de datos	Comentarios
Explotación de la información	Aplicaciones predefinidas	Permite también consultas imprevistas.	Al igual que en el apartado anterior, para los roles estratégicos de la organización es muy interesante poder realizar consultas imprevistas en el sistema (y no tener que ceñirse a los informes predefinidos diseñados en el sistema).
Tiempo de respuesta	Instantáneo	Rápido (pero no instantáneo)	El tiempo de respuesta no es crítico, por lo que es suficiente con que éste sea rápido.
FUNCIONALIDAD			
Actividades	Operatividad para el funcionamiento de la empresa	Análisis y decisión estratégica.	El objetivo de nuestro sistema es favorecer el análisis y la decisión estratégica.
Importancia de los datos	La importancia residen en el dato actual.	La importancia residen en los datos históricos.	Disponer de datos a lo largo del tiempo nos permite detectar tendencias.
Usuarios	Muchos. De la estructura media-baja de la empresa.	Pocos usuarios. De la estructura alta de la empresa.	El sistema estará disponible para gestores y directivos del programa Erasmus.

Así pues, se creará un almacén de datos para disponer de manera centralizada de los datos de nuestro sistema. Para cargarlos en el mismo, se creará un proceso ETL (*Extract, Transform, Load*) que validará los datos y los homogeneizará, asegurándonos que los datos utilizados en la toma de decisiones son de calidad.

Este paso es especialmente importante debido a las características de los datos proporcionados por el programa Erasmus: datos contenidos en distintos archivos con distintos formatos e introducidos por personas de distintas nacionalidades ubicadas en distintos países (lo cual hace muy difícil establecer procedimientos de introducción de datos, además de que se deberán de tener en cuenta aspectos culturales (deberemos tener cuidado con los distintos idiomas, utilización de distintos formatos de fechas, etc..).

Una vez se hayan cargado los datos en el almacén de datos mediante el proceso ETL, se procederá a diseñar el cubo que permitirá realizar el análisis multidimensional.

Por último, se procederá a la creación de los informes estáticos y dinámicos que resuelven las necesidades de nuestros usuarios.

1.4 Planificación del Trabajo

El proyecto comenzó el 18 de septiembre de 2015 y finalizará el 15 de enero de 2016. En total se compone de 18 tareas a realizar en 95 días.

Fase	Duración (en días)	Inicio	Fin
Inicio de proyecto	10	18/08/15	01/10/15
Análisis y Diseño	25	01/10/15	29/10/15
Implementación	41	29/10/15	17/12/15
Entrega final	19	21/12/15	15/01/16
Total	**95 días**		

La jornada laboral se ha establecido de la siguiente manera:

- De lunes a sábado: 3 horas de trabajo diarias
- Domingos: descanso

A continuación se presenta la **estimación detallada** de cada una de las fases presentadas:

1.4.1. Inicio del proyecto

Fecha de inicio: 18/09/15
Fecha hito: 01/10/15

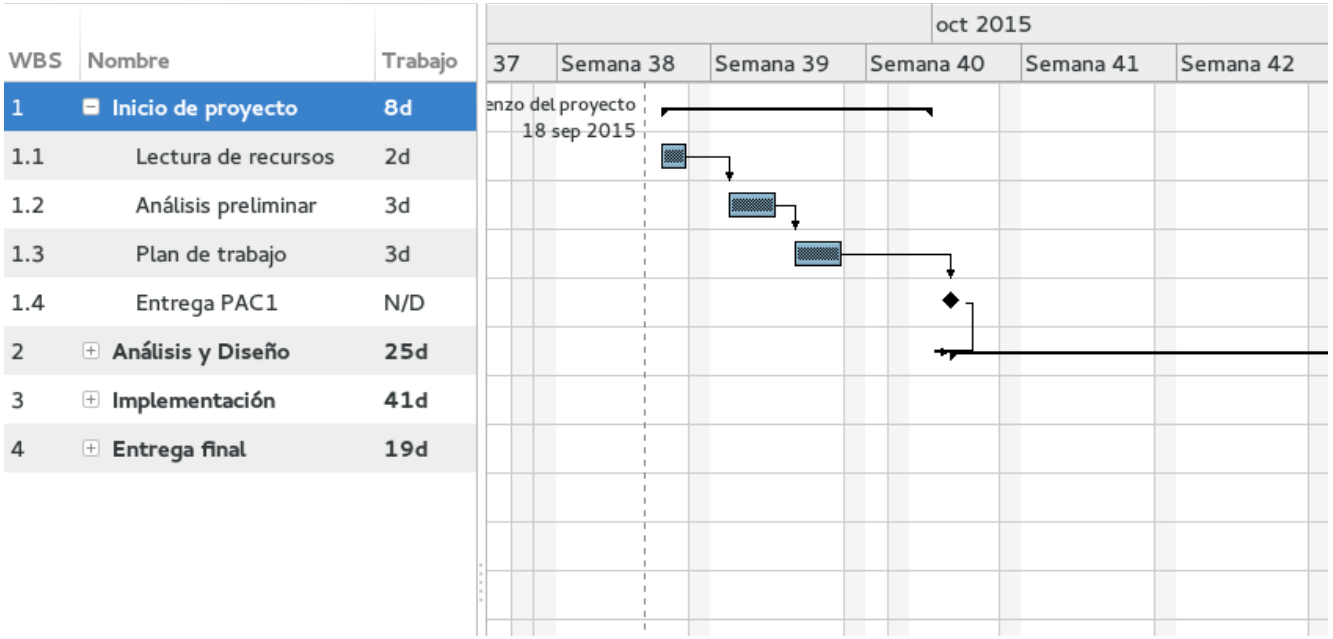
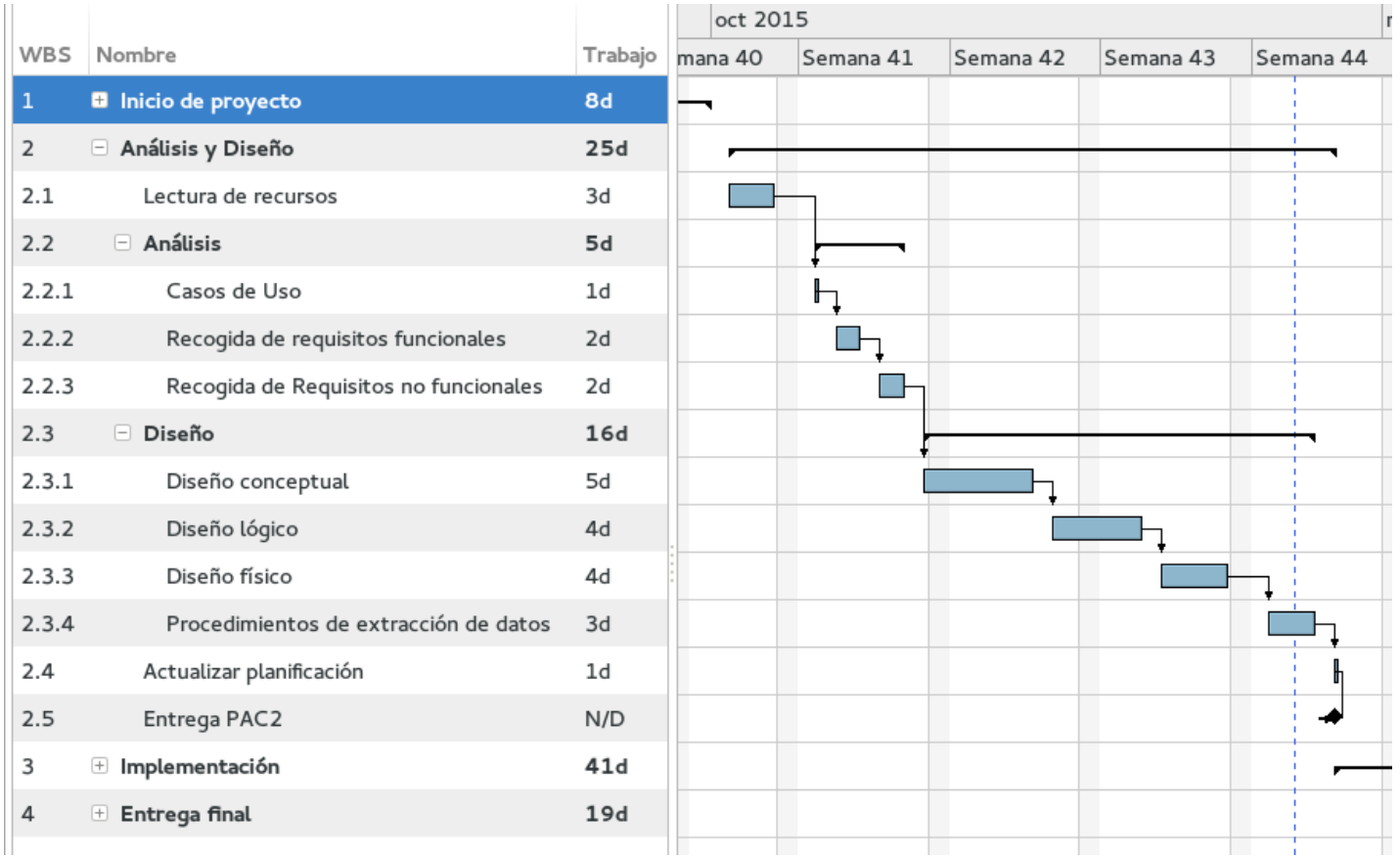


Ilustración 1: Gantt - Inicio del proyecto

1.4.2. Análisis y Diseño

Fecha de inicio: 01/10/15
Fecha hito: 29/10/15

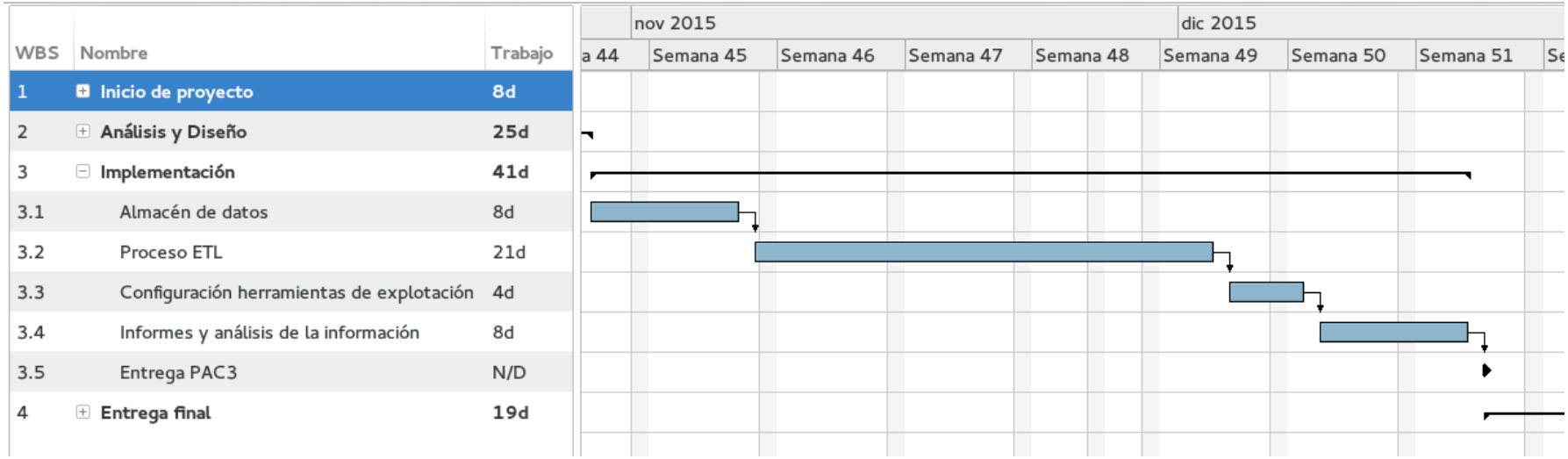


WBS	Nombre	Trabajo
1	**Inicio de proyecto**	**8d**
2	**Análisis y Diseño**	**25d**
2.1	Lectura de recursos	3d
2.2	**Análisis**	**5d**
2.2.1	Casos de Uso	1d
2.2.2	Recogida de requisitos funcionales	2d
2.2.3	Recogida de Requisitos no funcionales	2d
2.3	**Diseño**	**16d**
2.3.1	Diseño conceptual	5d
2.3.2	Diseño lógico	4d
2.3.3	Diseño físico	4d
2.3.4	Procedimientos de extracción de datos	3d
2.4	Actualizar planificación	1d
2.5	Entrega PAC2	N/D
3	**Implementación**	**41d**
4	**Entrega final**	**19d**

Ilustración 2: Gantt - Análisis y Diseño

1.4.3. Implementación

Fecha de inicio: 29/10/15
Fecha hito: 17/12/15



WBS	Nombre	Trabajo
1	**Inicio de proyecto**	**8d**
2	**Análisis y Diseño**	**25d**
3	**Implementación**	**41d**
3.1	Almacén de datos	8d
3.2	Proceso ETL	21d
3.3	Configuración herramientas de explotación	4d
3.4	Informes y análisis de la información	8d
3.5	Entrega PAC3	N/D
4	**Entrega final**	**19d**

Ilustración 3: Gantt - Implementación

1.4.4. Entrega final

Fecha de inicio: 17/12/15
Fecha hito: 15/01/16

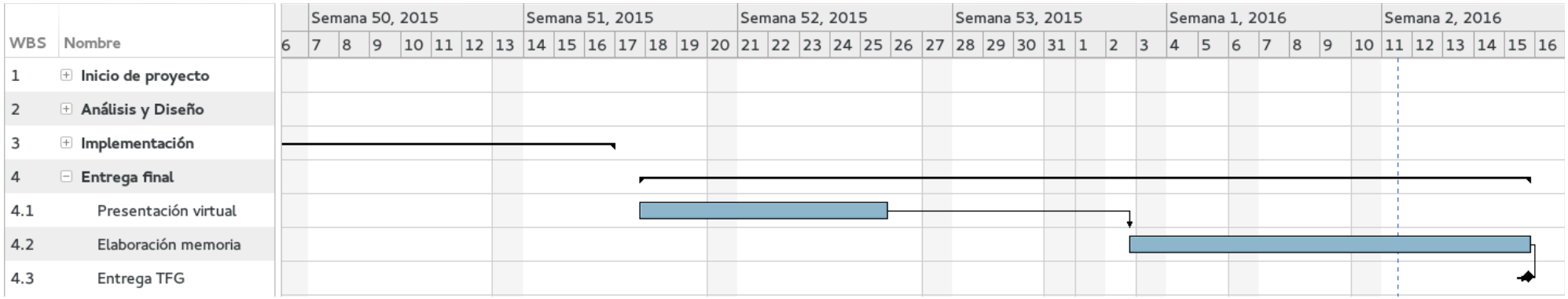


Ilustración 4: Gantt - Entrega final

1.5 Breve sumario de productos obtenidos

A continuación se listan los productos obtenidos como resultado de este Trabajo Fin de Grado::

i. Esquema de base de datos (MySQL)
ii. Proceso ETL (desarrollado con Pentaho Kettle).
iii. Cubo OLAP (desarrollado con Pentaho Mondrian Schema Workbench).
iv. Informes estáticos (desarrollado con Pentaho Reporting).
v. Informes dinámicos (desarrollados con Saiku Analytics)

1.6 Breve descripción de los otros capítulos de la memoria

En los siguientes apartados, se describen los aspectos más destacados de las fases de diseño e implementación del proyecto, haciendo énfasis en las dificultades encontradas y cómo se han resuelto.

2. Diseño

En este apartado se describen los aspectos más relevantes de la fase de diseño. Para ello, se comenzará con una breve introducción a la arquitectura *hardware* y *software* del sistema y continuaremos con el modelo de datos, el diseño lógico y el físico. Por último, también se introduce el proceso ETL a alto nivel.

2.1. Arquitectura hardware y software

Sobre la arquitectura software, cabe destacar que los **informes estáticos** (realizados con Pentaho Report Designer) se han ejecutado sobre el almacén de datos directamente. Se podrían haber ejecutado también sobre el cubo, no habiendo prácticamente ninguna diferencia entre ambas posibilidades, ya que estos informes sólo se ejecutarán de manera ocasional para tener una versión del informe con los datos actualizados (se espera que esto sea anualmente). Sin embargo, en este proyecto se ha decidido hacerlo sobre el almacén de datos directamente para mostrar una solución más realista. El cubo está orientado al análisis multidimensional realizado por los usuarios finales pertenecientes a la alta dirección de Erasmus y, por tanto, debe tener un tiempo de respuesta muy bueno. Con esta solución, evitamos sobrecargar el cubo de peticiones, lo que favorece que las consultas sobre el cubo se ejecuten rápido. Además, debido a que el número de *joins* que se necesitaban hacer en los informes no era muy extenso, el desarrollo de los mismos no ha resultado lento y no ha presentado problemas.

De manera opuesta, los **informes dinámicos** (realizados con Saiku) se ejecutan sobre el cubo para aprovechar la rapidez y potencia que proporciona en las consultas, de forma que la experiencia del usuario (que está realizando consultas libres sobre el cubo) sea rápida y potente. Si se hubiera establecido otra fuente de datos (distinta a un cubo), las consultas hubieran tardado mucho en realizarse debido a la cantidad de *joins* que pueden ser necesarios y al volumen de datos utilizado.

2.2. Diseño conceptual

En este apartado se presentan los aspectos más importantes del modelo de datos diseñado para dar soporte a este sistema.

2.2.1. Diseño del modelo de datos

Sobre el diseño del modelo de datos, cabe destacar el refinamiento realizado sobre el modelo de datos diseñado inicialmente, y que podemos ver en la siguiente figura:

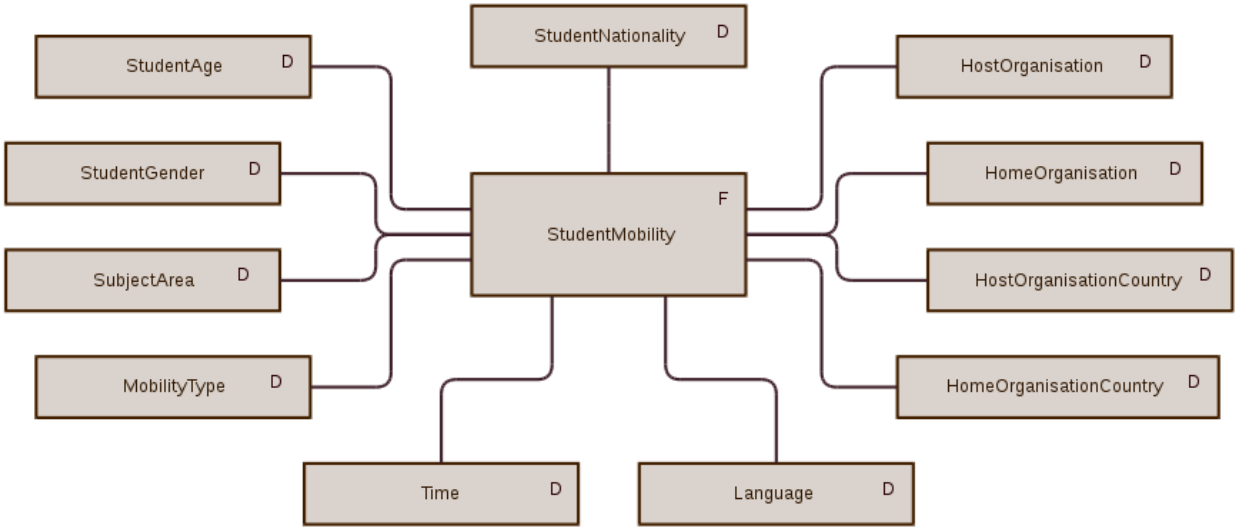


Ilustración 5: Diseño conceptual inicial

Como se puede observar, apreciamos varias dimensiones que hacen referencia a un mismo concepto:

- Las dimensiones *HomeOrganisation* y *HostOrganisation* hacen referencia a organizaciones, por lo que se han agrupado en una única dimensión (Organisation), que juega dos roles distintos (uno para cada dimensión identificada inicialmente).

- Las dimensiones *HomeOrganisationCountry*, *HostOrganisationCountry* y *StudentNationality* se han agrupado en la dimensión *Country*, que juega tres roles distintos para cada una de estas tres dimensiones identificadas inicialmente.

Una vez realizado este refinamiento, el diagrama conceptual final es el siguiente:

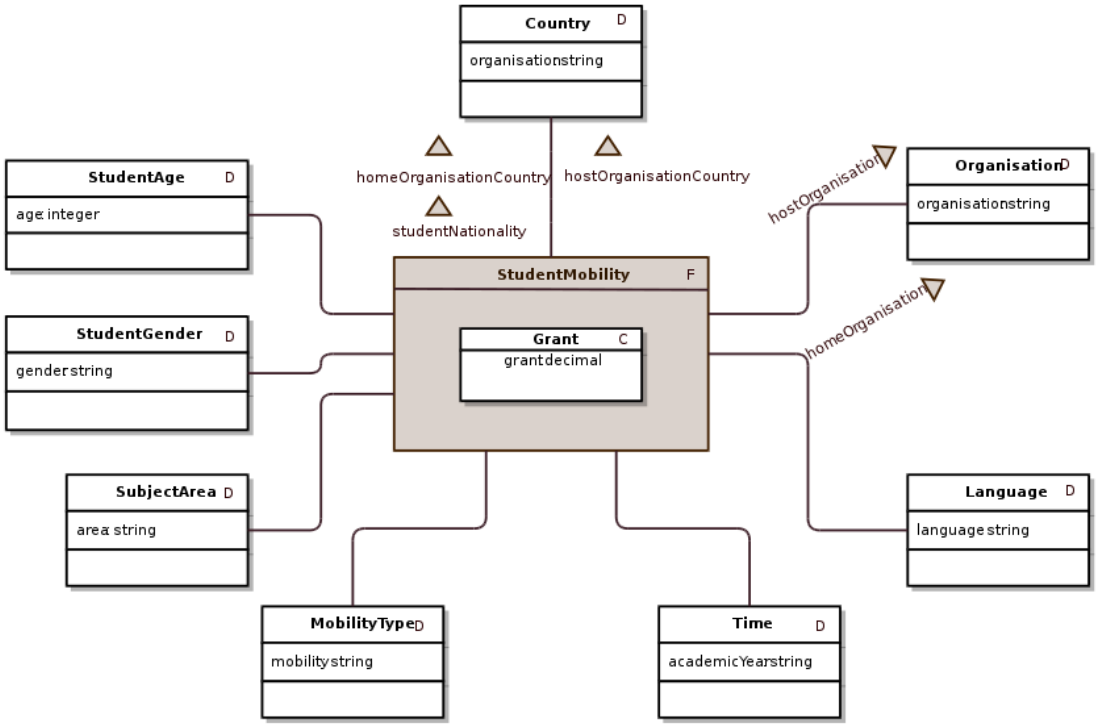


Ilustración 6: Diseño conceptual final

2.3. Diseño lógico

Como sabemos, no es recomendable utilizar un esquema en forma de copo de nieve, ya que cada vez que utilizáramos una dimensión, se debería de hacer un gran número de *joins*. Así pues, se ha decidido seguir una estructura de datos en forma de estrella. Ya que en nuestro sistema sólo vamos a analizar un hecho (la movilidad de estudiantes), sólo será necesario crear una estrella.

Durante esta fase, cabe destacar los **campos** que se han añadido a cada una de las **dimensiones**:

- utilización de claves subrogadas (rowId) que nos ayudarán a identificar cada fila de cada dimensión
- cuando ha estado disponible, se ha añadido un campo con el atributo identificador que proviene del sistema operacional
- un campo que indicará la versión de la instancia (version): este diseño nos permitirá registrar los cambios que se realicen en las dimensiones sin tener que borrar ningún dato

- un campo que indica la fecha de inserción del registro (insertTimeStamp). Aunque no es necesario para la ejecución del proceso ETL, se ha considerado que es un *metadato* importante para el mantenimiento de la base de datos.

Las referencias a las dimensiones desde la tabla de **hechos** apuntan a los *rowId* de las dimensiones. De esta forma, evitamos problemas si se modifican los identificadores de las tablas de dimensiones en los sistemas operacionales y, además, reducimos el tamaño de la tabla de hechos (en comparación a si almacenáramos las descripciones completas).

2.4. Diseño físico

En el documento de diseño se ha propuesto aplicar técnicas estándar de mejora del rendimiento a nivel físico sólo en el caso de que haya problemas de rendimiento. En particular, se han propuesto:

- Índices de mapas de bits
- Técnicas de preagregación

3. Diseño técnico

En este apartado se describe los aspectos más importantes del diseño del **proceso ETL**. Este proceso se ha dividido en dos partes: por una parte están los subprocesos que generan la **carga inicial de datos** en el almacén de datos, y por otros los **procesos de actualización de las dimensiones** (como veremos más adelante, la tabla de hechos es tratada de manera especial).

3.1. Carga inicial de dimensiones

Prácticamente todas las dimensiones siguen un mismo algoritmo general para realizar la carga inicial. Este **algoritmo general** se puede resumir de la siguiente manera:

- Creación de la tabla correspondiente a la dimensión en base de datos si esta no existe
- Leer la fuente de datos indicada para la dimensión
- Mapear los campos de la tabla del almacén de datos y encontrar el campo al que corresponde en la fuente de datos
- Validar los datos leídos en la fuente de datos y aplicar el formato adecuados
- Crear el campo *versión* y poner su valor a 1.0
- Insertar la fila en base de datos

En caso de que se detecte un **error en las validaciones**, el proceso ETL abortará la inserción de la fila que ha provocado el error y guardará dicha fila en un fichero de log específico para cada dimensión (o para la tabla de hechos), de manera que el administrador del sistema pueda comprobar el motivo del error.

Las dimensiones que no siguen este algoritmo son St*udentAge*, *StudentGender*, *Time* y *MobilityType,* ya que generan los datos directamente en el mismo proceso ETL. Se ha decidido hacer así debido a que son dimensiones con muy pocos valores y que se modificarán en ocasiones excepcionales.

3.2. Actualizaciones de dimensiones

En este apartado, cabe destacar la **gestión de las versiones** diseñada para las dimensiones. En las fuentes de datos recuperaremos el identificador del sistema operacional y el campo descripción (nombre del país u organización, etc). Si en la fuente de datos la descripción correspondiente a dicho identificador es distinta a la del almacén de datos, se añadirá una fila nueva en la dimensión con la nueva descripción y se incrementará el valor del campo *version* en 1.

3.3. Tabla de hechos

La tabla de hechos tiene un tratamiento especial. En primer lugar, antes de cargar la tabla de hechos se deberán de actualizar las dimensiones. En segundo lugar, se deberán de borrar los datos del curso académico que se va a cargar (en caso de que exista alguno). Esto nos permitirá realizar reprocesados del curso académico de manera sencilla (por ejemplo, si el proceso aborta antes de terminar).

3.4. Tratamiento de errores

Si se encuentran errores en el almacén de datos, se realizará un proceso diseñado a medida en el que se procederá a validar los nuevos datos a introducir y se realizará una fusión destructiva modificando los registros afectados por el error. Además, se incrementará el número de versión del registro en 0.1.

Además, se deberá corregir el error en las fuentes de datos para evitar que se vuelva a propagar en el futuro.

4. Implementación

En este apartado se comentan las decisiones de implementación más importantes, así como algunas dificultades que se han encontrado y cómo se han solucionado. Para ello, este apartado se divide en los distintos productos en los que se ha trabajado.

4.1. Creación del almacén de datos

En un principio había planteado utilizar el motor **InfiniDB** de MySQL, ya que es un motor de base de datos basado en columnas con excelentes resultados para sistemas analíticos. Ya que en nuestro caso se va a implementar un cubo OLAP, un motor de base de datos basado en columnas proporcionaría un mayor rendimiento. Sin embargo, este motor no se encuentra disponible en la máquina Amazon (esto lo podemos comprobar con el comando *show engines* de MySQL), por lo que al final se ha decidido trabajar con el motor **InnoDB**.

Para mejorar el rendimiento de las consultas, se han creado **índices B-Tree** en los campos descriptivos de las dimensiones y en el campo *grantAmount* de la tabla de hechos. Sin embargo, no ha sido necesario crear índices de mapas de bits ni tablas preagregadas, como se sugería inicialmente.

Por último, a la hora de crear las tablas se ha decidido utilizar una **nomenclatura** que facilite la comprensión de la estructura en estrella utilizada en el esquema de datos. Las tablas correspondientes a dimensiones comienzan por “Dim_” y la tabla correspondiente a los hechos comienza por la cadena “Fact_”.

4.2. Creación del ETL

La implementación de los procesos ETL ha sido bastante similar en todos los casos, adaptando el proceso a las particularidades de cada dimensión. Ésta se ha realizado con **Pentaho Kettle** versión 5.4.

Un caso excepcional ha sido la carga inicial de la **dimensión *Language***, ya que en la fuente de datos puede haber más de una fila con el mismo código (por ejemplo, la ISO-639 que es la norma seguida, indica que tanto el Catalán como el Valenciano tienen el mismo código “ca”). Para solventarlo, se ha decidido crear un único registro en el que el código será “CA” y la descripción será “Calatan/Valencian”. Así pues, en el proceso ETL se han agrupado las filas por código y se ha indicado que se concatenen los idiomas separados por el carácter “/”.

Merecen mención especial también las dimensiones ***MobilityType, StudentAge, StudentGender*** **y *Time***, ya que los **datos** se **generado en el propio ETL**. Se ha decidido proceder de esta manera debido a

que son dimensiones con muy pocos datos y propensos a pocas modificaciones.

En el caso de las **actualizaciones**, se sigue el mismo proceso que en las cargas iniciales, pero una vez tenemos los datos listos, se comparan con los datos actuales del almacén de datos mediante el operador *Merge Rows (diff)*. Esta operación busca en el almacén de datos el registro indicado por el campo clave de la fuente de datos, recupera el campo descriptivo y lo compara con el obtenido en la fuente de datos. Posteriormente añade una columna al flujo de datos en el que indica si la fila se encuentra repetida, eliminada o es nueva. Para las filas nuevas, se establecerá la versión a 1.0 y en el caso de las modificadas la versión se incrementará en 1.0.

En el caso de la **tabla de hechos**, se ha creado un proceso ETL por curso académico, ya que los nombres y orden de las columnas en los ficheros de cada año son distintos. Cabe destacar el uso del paso *Database Value Lookup* para recuperar los *rowId* almacenados en el almacén de datos para cada código proveniente del sistema operacional. Este paso no utiliza una tabla hash como se había propuesto en el diseño (ya que aún no conocía en profundidad Pentaho Kettle no conocía este paso), pero utilizando la opción de precargar todos los datos en caché se obtiene un rendimiento adecuado.

Para finalizar, se calcula el total de la beca concedida al estudiante. Esto se hace mediante un paso *Calculator*, en el que se suman los importes concedidos para los conceptos *studyGrant* (beca de estudios) y *placementGrant* (beca de trabajo).

4.3. Creación del Cubo

Para implementar el cubo se ha utilizado **Pentaho Schema Workbench** versión 3.10. La herramienta ha resultado fácil de utilizar y aprender a manejar. Existe una amplia y detallada documentación sobre la misma, que ha sido de gran ayuda.

Para crear el cubo, se ha procedido a añadir todas las **dimensiones** definidas en nuestro problema, indicando para cada una de ellas la jerarquía y el nivel (ya que todas las dimensiones tienen un sólo nivel más además de *All*). También se ha indicado la tabla de base de datos de la que deben de obtener los datos, y cómo hacer los joins con la tabla de hechos.

Se ha indicado también la tabla de la que leer los hechos (tabla Fact_StudentMobility), y se ha añadido las **medidas** *Grant* (donde podemos consultar los importes medios de las becas concedidas) y *StudentAmount* (que sirve para contabilizar el número de estudiantes).

En la siguiente imagen podemos ver el aspecto del cubo desde Mondrian Schema Workbench:

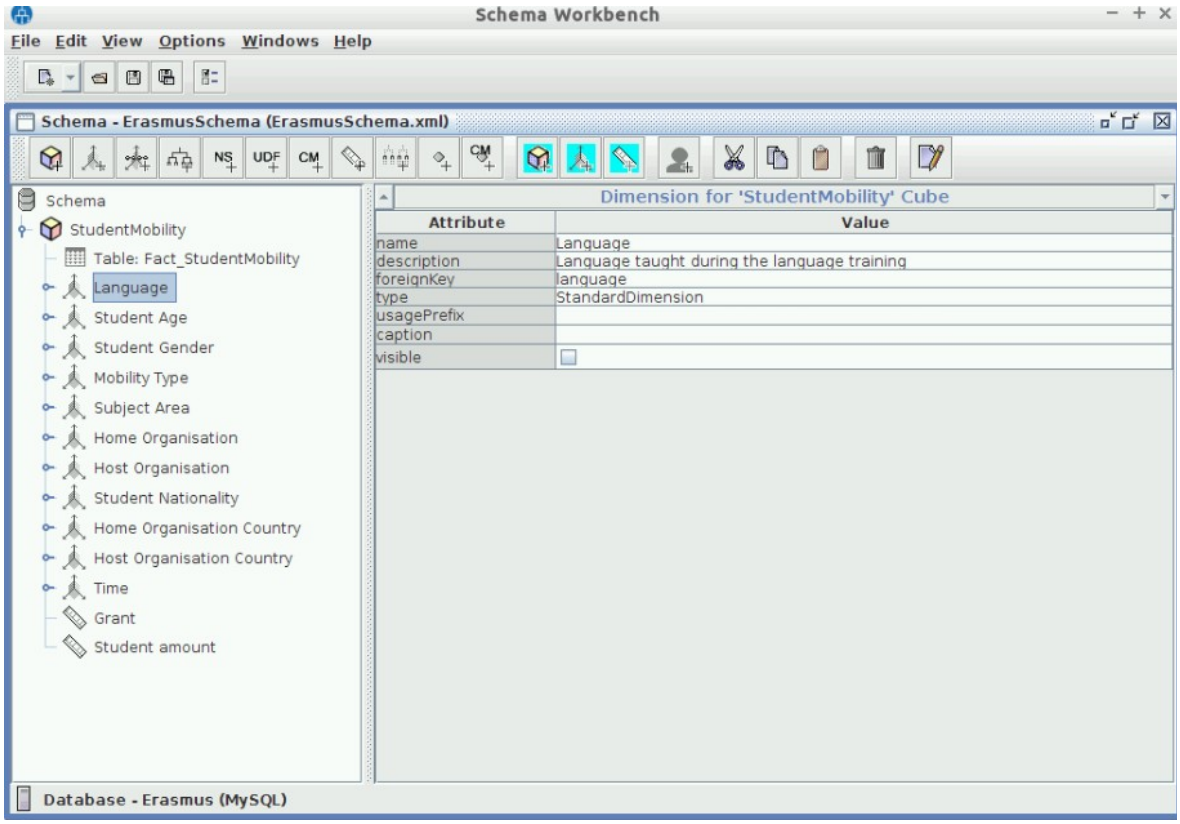


Ilustración 7: Mondrian Schema Workbench

Se han establecido como invisibles las dimensiones que no han sido solicitadas, desmarcando la casilla *visible* de las dimensiones que no queremos mostrar. De esta forma, tenemos el cubo completo, y podremos ofrecer distintas vista del mismo cambiando la visibilidad de las dimensiones según las necesidades del usuario final (será necesario publicar cada una de estos cubos).

Además, se han completado los campos *description* de las dimensiones y medidas para que aparezca un *tooltip* con una descripción del elemento seleccionado. De esta forma, el usuario que está diseñando el informe dinámico desde Saiku dispondrá de una descripción de los elementos disponibles.

Durante el diseño del cubo, se ha ido probando poco a poco cada dimensión y medida creando una **query MDX** básica que accediera a la dimensión (o medida) probadas. Estas consultas siguen el formato siguiente:

```
select {[Dim1].[All Dim1s]} on rows,
{[Measures].[Meas1]} on columns
from [CubeName]
```

Esta consulta, seleccionaría todos los elementos de la dimensión *Dim1* en las filas, y en las columnas aparecería la medida *Meas1*.

Una vez que el cubo está listo, se debe de **publicar** mediante la opción *File* > *Publish* e indicar la dirección en la que se encuentra el servidor BI.

4.4. Creación de los informes dinámicos

Los informes dinámicos se han creado utilizando el complemento de Pentaho BI **Saiku Analytics** versión 3.3. Una vez iniciado el servidor BI, abrimos el navegador y entramos en http://localhost:8080/pentaho/Home y seleccionamos la opción *Create New* > *Saiku Analytics* > *Create new query.*

Durante la implementación del informe dinámico, se decidió hacer un cambio respecto al diseño que se había planteado. En el análisis, en un principio se indicó que se debería crear un único informe dinámico con todas las dimensiones y medidas indicadas en el enunciado del trabajo. Sin embargo, una vez que se comenzó la implementación del informe, se detectó que no tenía sentido añadir tantas dimensiones en un único informe, ya que resultaba una tabla (o gráfico, según la visualización seleccionada) muy difícil de entender, con demasiados valores. Así pues, se ha decidido crear tres informes dinámicos (*Free-Report1*, *Free-Report2* y *Free-Report3*) que muestren sólo unas cuantas dimensiones y resulten más fáciles de comprender.

La implementación de los informes con Saiku ha sido sencilla, ya que como se puede apreciar en la captura de pantalla siguiente, sólo hay que pinchar y arrastrar las dimensiones y medidas a las cajas mostradas con las etiquetas *Measures, Columns* y *Rows.* Una vez hecho esto, se guarda el informe.

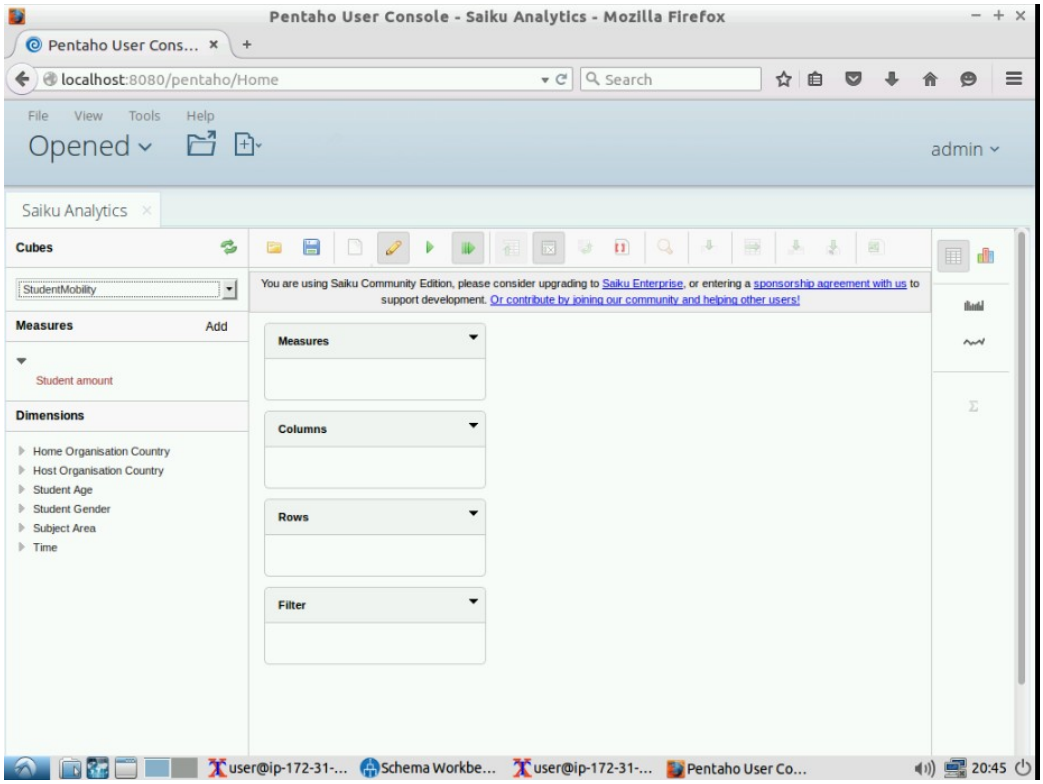


Ilustración 8: Saiku Analytics

Se ha decidido mostrar los informes en forma de tabla para que el usuario pueda navegar por los datos de manera sencilla. Una vez que haya encontrado el nivel de detalle deseado, él mismo puede crear las gráficas que más le interesen simplemente pinchando en el icono de gráficas y seleccionando el tipo de gráfico deseado.

Ilustración 9: Gráfico en Saiku Analytics

4.5. Creación de los informes estáticos

Para crear los informes estáticos se ha utilizado **Pentaho Report Designer** versión 5.0, instalado en el portátil de trabajo personal. Para poder realizar la conexión con la base de datos MySQL ubicada en la máquina AWS se utilizó un túnel SSH, ya que no podía realizarse la conexión directa a base de datos porque el puerto de MySQL no estaba abierto a Internet. En concreto, el comando utilizado fue el siguiente:

```
ssh -L 3306:localhost:3306 user@<IP_máquina_en_AWS>
```

Una vez configurada la fuente de datos, se ha procedido a crear un informe básico que servirá de **plantilla** para realizar los otros. Se le han añadido la cabecera y el pie de página, padding a las distintas secciones del informe y se ha decidido la gama de colores a utilizar (en caso de usar pocos colores, éstos están basados en los colores del logotipo del programa Erasmus para crear una imagen corporativa homogénea).

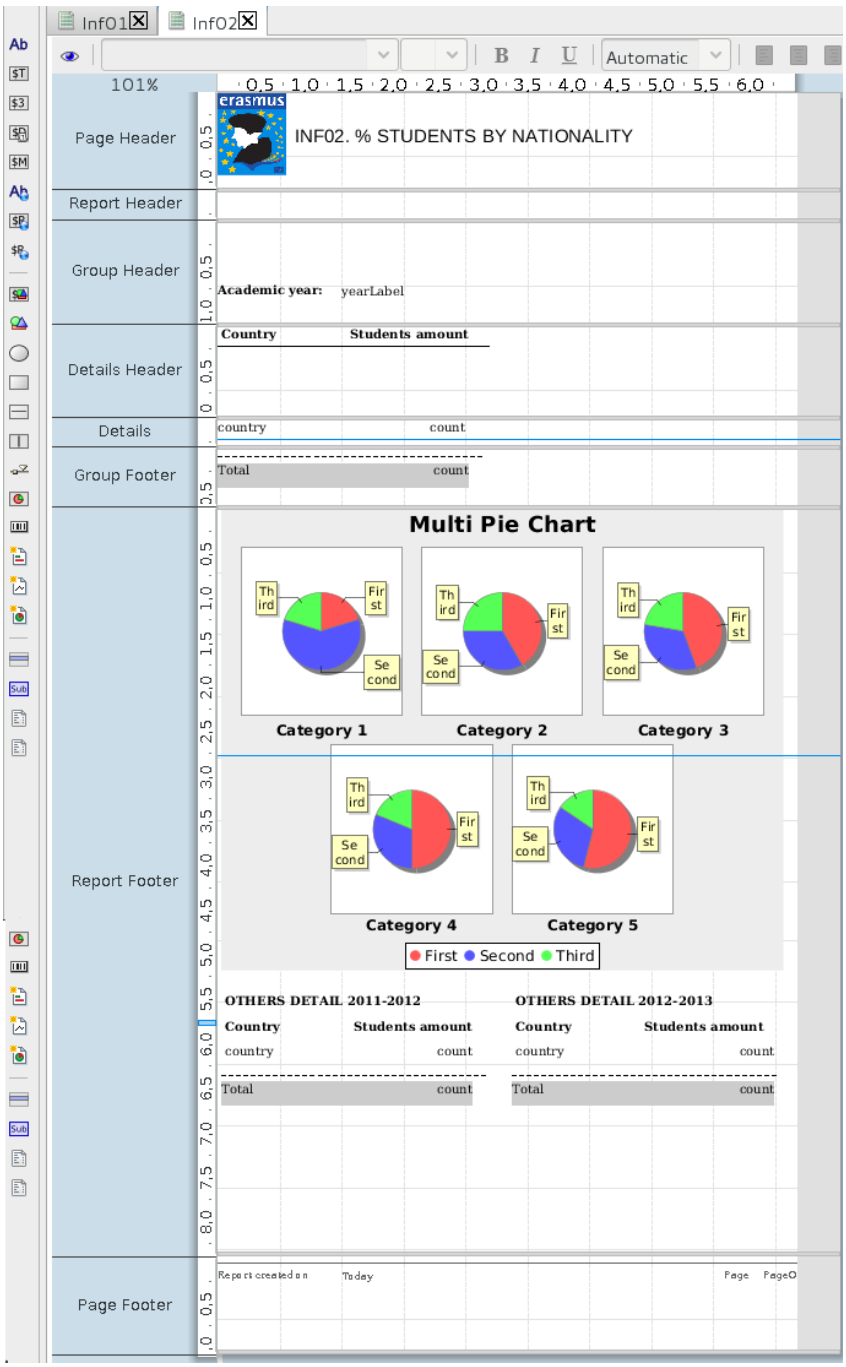


Ilustración 10: Informe con gráfico de sectores y sección "Others"

5. Valoración económica del trabajo

En este apartado se desglosa una valoración económica del trabajo realizado y se hace un estudio de viabilidad del producto.

5.1. Coste de desarrollo

Para realizar la valoración económica del desarrollo del presente trabajo, se han de tener en cuenta los siguientes elementos:

5.1.1. Mano de obra

En la siguiente tabla se detalla el número de días de trabajo necesarios para realizar el proyecto:

Fase	Número de días
Inicio del proyecto	8
Análisis y diseño	25
Implementación	41
Entrega final	19
Total	**93 días**

Si establecemos que el precio del día es de 90€ (3 horas diarias a 30€ cada una), tendremos un total de 8.370€ en mano de obra.

5.1.2. Coste software

En la siguiente tabla se muestra el software utilizado para desarrollar el presente proyecto y se indica el precio del mismo:

Software	Precio
Base de datos MySQL	Gratuito
Pentaho BI Suite Comunity Edition	Gratuito
Pentaho Data Integration (Kettle)	Gratuito
Mondrian Schema Workbench	Gratuito
Pentaho Report Designer	Gratuito
Sistema operativo Fedora	Gratuito
Cliente VNC Vinagre	Gratuito
Libre Office	Gratuito

Software	Precio
Navegador Firefox	Gratuito
TOTAL	**0€**

Como se puede apreciar, se ha utilizado software gratuito para llevar a cabo todas las fases del proyecto.

5.1.3. Coste hardware y otros servicios

Se ha hecho uso de una máquina de AWS para desarrollar el producto y dejarlo instalado en la misma.

Se desconoce el precio acordado por la UOC, pero suponiendo que se ha utilizado una instancia bajo demanda optimizada para informática, el precio por hora es de 0.11$.

Si multiplicamos este precio por las horas utilizadas (93 días * 3 horas diarias = 279 horas) obtenemos un importe de 30,69$.

El euro se encuentra hoy a 1,09$, por lo que el total es de **33,45€**.

5.1.4. Coste de desarrollo total

Así pues, teniendo en cuenta estos factores, el coste total del desarrollo del producto es el siguiente:

Concepto	Precio (en euros)
Mano de obra	8.370€
Coste software	0
Coste hardware y otros servicios	33,45
TOTAL	**8.403,45€**

5.5. Coste de mantenimiento

Como sabemos, el almacén de datos deberá ser actualizado una vez al año con los nuevos datos generados por el programa Erasmus. Se deberá realizar una actualización de los datos almacenados (se añadirán más datos en la tabla de hechos y puede ser necesario añadir nuevos registros en las dimensiones).

Se estima que este trabajo puede ser realizado por una persona en dos jornadas de trabajo de 8 horas. Si multiplicamos las horas necesitadas por 30€ la hora, tenemos un total de 480€ para la **mano de obra**.

El **software** utilizado es gratuito, por lo que no incurrirá en ningún coste.

Por último, el día de trabajo para el mantenimiento anual supondrá un coste de 8 horas * 0,11$ = 0,88$, es decir, **0,95€** a día de hoy. Además, según el uso que se haga del sistema por parte de los usuarios se deberá de abonar una cantidad distinta. Se estima que de los 251 días laborales que tiene un año, los usuarios acceden al sistema el 60% de los días y cada conexión con el sistema dura una media de 1 hora (150,6 horas). Si tenemos un total de 200 usuarios repartidos en toda la unión europea, el total será de 30.120 horas. Con el plan actual en AWS, esto supondría un coste de 3313,20$, es decir 3.611,38€.

Así pues, el total del mantenimiento del producto para un año sería:

Concepto	Precio
Mano de obra	480
Coste software	0
Coste hardware y otros servicios	3.611,38
TOTAL	**4.091,38€**

Este importe podría ser aceptado por el programa Erasmus, ya que los beneficios que proporciona el análisis en profundidad de los datos puede revertir en grandes mejoras para programa. Sin embargo, sería interesante contactar con otros proveedores de servicios de computación en la nube , como Windows Azure de Microsoft[1] o Joyent [2] para ver si ofrecen precios más competitivos.

[1] MICROSOFT. Windows Azure (2016). [en línea]. https://azure.microsoft.com/en-us/ [fecha de consulta: 7 de enero de 2016].

[2] JOYENT. Joyent (2016). [en línea]. https://www.joyent.com/ [fecha de consulta: 7 de enero de 2016].

6. Conclusiones

Como se ha indicado en la introducción de la memoria, el **objetivo principal** de este Trabajo Fin de Grado era el de centralizar la información ofrecida por el programa Erasmus en un único lugar que facilite la exploración de los datos de dos formas distintas: de manera libre, para que el usuario pueda navegar por los datos según necesite y, por otra parte, ofreciendo unos informes estáticos que pueden ser consultados periódicamente de forma sencilla.

En mi opinión el objetivo principal de este proyecto ha sido conseguido, ya que los informes dinámicos ofrecen una navegación libre por los datos y los estáticos ofrecen una representación de los datos que puede ser consultada de manera rápida en cualquier momento. Así mismo, el proceso ETL diseñado gestiona la actualización del almacén de datos para que se puedan añadir o actualizar elementos en las dimensiones y en la tabla de hechos.

Además, también he conseguido familiarizarme y profundizar en algunos aspectos del diseño e implementación de un sistema BI, que era otro **objetivo importante a nivel personal**.

En líneas generales, se ha comprobado que la dificultad de este sistema a nivel de análisis y diseño reside en comprender los datos disponibles (muchas veces será complicado obtener todas las fuentes de datos, no tendremos una descripción de las mismas, ni sabremos qué significa cada campo, etc). Así mismo, también ha entrañado cierta complejidad proporcionar una representación gráfica de los datos que facilite la obtención de conclusiones.

A nivel técnico, el producto que ha entrañado una mayor complejidad ha sido el proceso ETL, ya que se trata de un proceso laborioso que exige un trabajo cuidadoso y detallista para que los resultados obtenidos sean los esperados. Ya que el volumen de datos utilizado no ha sido demasiado grande, con el diseño planteado no se han encontrado problemas de rendimiento, aunque si dicho volumen aumentara drásticamente, sería necesario aplicar técnicas de mejora del rendimiento a nivel físico.

La **planificación** se ha conseguido seguir, aunque ha habido que realizar un esfuerzo especial en la fase de implementación, ya que el proceso ETL ha consumido más tiempo del inicialmente planificado; sin embargo, aumentando las horas de trabajo durante la semana (de 2 horas diarias a 3) y con unas cuantas horas de trabajo extra los fines de semana se ha podido solventar.

Cabe destacar que debido a la inexperiencia en este ámbito y tecnología, ha sido necesario cambiar algunos requisitos inicialmente planteados en el análisis, como la presentación de los datos en los

informes, ya que los gráficos indicados en el análisis no resultaban adecuados una vez que se han implementado.

Sin lugar a dudas, existen muchos detalles que se pueden depurar y mejorar en el presente trabajo y que abren líneas de trabajo futuro que pueden mejorar sustancialmente el trabajo presentado. Algunos ejemplos son los siguientes:

- **Mejoras en el proceso ETL**:
  - Creación de un *job* en el proceso ETL que llame a cada proceso ETL en su debido orden. De esta forma automatizamos aún más la carga inicial y actualización del almacén de datos (podríamos llamar incluso a este job mediante un *script*, simplificando aún más el proceso).

  - Incluir el año académico como un parámetro en el job, facilitando la ejecución del proceso ETL en futuros años académicos.

  - Utilización de variables para definir las rutas de los ficheros de entrada y salida en el proceso ETL, de forma que la exportación del proceso a otras máquinas sea trivial.

- **Mejoras en el cubo**:
  - Para mejorar el rendimiento de las consultas hechas con el cubo, se podrían crear dimensiones degeneradas (*degenerate dimensions*) para las dimensiones *Time, StudentGender* y *MobilityType*, de forma que nos podríamos ahorrar algunos joins en base de datos al referenciar a dichas tablas y por tanto mejorar la velocidad de las consultas.

  - En las dimensiones HomeOrganisation y HostOrganisation, se podría añadir un nivel más en la jerarquía que indique el país de la organización, facilitando la navegación por los datos.

  - Añadir una medida que también indique la cuantía de las becas concedidas, pero que utilice como operación de agregación la suma, de forma que podríamos consultar las medias y los totales en el cubo.

  - En la dimensión Tiempo, se podría añadir el mes de inicio de la beca.

- **Mejoras en los informes**:
  - Instalación de Pentaho Report Designer en la máquina Amazon, de manera que se puedan publicar los informes directamente en la plataforma BI y así estén disponibles en línea.

- Configuración de usuarios en Pentaho de forma que los usuarios sólo puedan acceder a los informes que les interesan.

4. Glosario

Almacén de datos (o data warehouse): colección de datos orientado a un ámbito determinado, integrado, no volátil y variable en el tiempo, que ayuda en la toma de decisiones en la entidad en la que se utiliza.

Amazon Web Services (AWS): conjunto de servicios de computación remotos que ofrece una plataforma de computación en la nube.

Business Intelligence (BI): categoría de soluciones software que permite a las organizaciones obtener conocimiento sobre sus operaciones críticas a través de aplicaciones de *reporting* y herramientas de análisis analítico.

InfiniDB: motor de base de datos escalable, basado en columnas construido para ser utilizados por aplicaciones de BI, almacenes de datos, big data y otras aplicaciones con lecturas intensivas. Permite consultas muy rápidas.

Informe estático: informe generado con los datos disponibles en el momento de ejecutarlo y que no permitirá ningún tipo de interactuación con el mismo.

Informe libre (o dinámico): informe generado con los datos disponibles en el momento de ejecutarlo y que permitirá cambiar el nivel de detalle en el que se ven dichos datos.

InnoDB: motor de base de datos para MySQL. Proporciona transacciones que cumplen con el estándar ACID y proporciona soporte para claves foráneas.

Multimensional Expressions (MDX): lenguaje de consulta para bases de datos OLAP.

Open Data: comunidad en la que se ofrecen los datos de manera libre a todo el mundo para que los puedan utilizar como deseen, sin restricciones de copyright u otros mecanismos de control.

Proceso ETL: las siglas provienen del inglés “Extract, transform and load” (extracción, transformación y carga). Proceso que permite mover datos desde múltiples fuentes de datos, formatearlos y limpiarlos, y cargarlos en otra base de datos (o data warehouse) para analizar, o en otro sistema operacional para apoyar a un proceso de negocio.

Reporting: creación de informes sobre los datos.

5. Bibliografía

ABELLÓ GAMAZO, A. (2015). Diseño multidimensional. Material Docente UOC, TFG Datawarehouse.

AMAZON WEB SERVICES. Precios de Amazon EC2 (2016). [en línea]. https://aws.amazon.com/es/ec2/pricing/ [fecha de consulta: 4 de enero de 2016].

FFMEPG. (2016). [en línea].http://www.ffmpeg.org/ [fecha de consulta: 13 de enero de 2016].

FUNDÉU BBVA. Buscador urgente de dudas (2016). [en línea].http://www.fundeu.es/ [fecha de consulta: 5 de enero de 2016].

JOYENT. Joyent (2016). [en línea]. https://www.joyent.com/ [fecha de consulta: 7 de enero de 2016].

LINGUEE. Diccionario inglés-español (2016). [en línea]. http://www.linguee.es [fecha de consulta: 5 de enero de 2016].

MICROSOFT. Windows Azure (2016). [en línea]. https://azure.microsoft.com/en-us/[fecha de consulta: 7 de enero de 2016].

REAL ACADEMIA DE LA LENGUA. Buscador urgente de dudas (2016). [en línea]. http://www.rae.es/ [fecha de consulta: 5 de enero de 2016].

RIUS GAVÍDIA, A.; SERRA VIZERN, M.; CURTO DÍAZ, J. (2015). Módulo 2: Introducción al almacenamiento de datos. Material Docente UOC, TFG Datawarehouse.

SÁENZ HIGUERAS, N.; VIDAL OLTRA, R. (2015). Redacción de textos científico-técnicos. Material Docente UOC, TFG Datawarehouse.

WIKIPEDIA. Almacén de datos (2016). [en línea]. https://es.wikipedia.org/wiki/Almac%C3%A9n_de_datos [fecha de consulta: 3 de enero de 2016].

WIKIPEDIA. InfiniDB (2016). [en línea]. https://en.wikipedia.org/wiki/InfiniDB [fecha de consulta: 3 de enero de 2016].

WIKIPEDIA. InnoDB (2016). [en línea]. https://en.wikipedia.org/wiki/InnoDB [fecha de consulta: 3 de enero de 2016].

WIKIPEDIA. Open Data (2016). [en línea]. https://en.wikipedia.org/wiki/Open_data [fecha de consulta: 3 de enero de 2016].