# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTRODUCTION</td>
<td>3</td>
</tr>
<tr>
<td>AMADEUS</td>
<td>3</td>
</tr>
<tr>
<td>History</td>
<td>3</td>
</tr>
<tr>
<td>Localization</td>
<td>4</td>
</tr>
<tr>
<td>System</td>
<td>5</td>
</tr>
<tr>
<td>THE DEPARTMENT</td>
<td>7</td>
</tr>
<tr>
<td>PREVIOUS KNOWLEDGE</td>
<td>9</td>
</tr>
<tr>
<td>THE PROJECT</td>
<td>10</td>
</tr>
<tr>
<td>PRODUCT DEFINITION</td>
<td>10</td>
</tr>
<tr>
<td>Target of the project</td>
<td>10</td>
</tr>
<tr>
<td>Existing and purposed tools</td>
<td>13</td>
</tr>
<tr>
<td>SQL Plus and SQL Developer</td>
<td>13</td>
</tr>
<tr>
<td>Oracle tools</td>
<td>15</td>
</tr>
<tr>
<td>Schema Spy</td>
<td>17</td>
</tr>
<tr>
<td>Statspack</td>
<td>18</td>
</tr>
<tr>
<td>Oracle Designer</td>
<td>18</td>
</tr>
<tr>
<td>Conclusion</td>
<td>19</td>
</tr>
<tr>
<td>DEVELOPING</td>
<td>22</td>
</tr>
<tr>
<td>Plan of work</td>
<td>23</td>
</tr>
<tr>
<td>Oracle Designer working</td>
<td>25</td>
</tr>
<tr>
<td>Previous tool</td>
<td>25</td>
</tr>
<tr>
<td>Functional Requirements</td>
<td>25</td>
</tr>
<tr>
<td>Technologies used</td>
<td>27</td>
</tr>
<tr>
<td>Navigation flow</td>
<td>28</td>
</tr>
<tr>
<td>Previous design considerations</td>
<td>42</td>
</tr>
<tr>
<td>Developing of functionality one &quot;Patch Delivery&quot;</td>
<td>43</td>
</tr>
<tr>
<td>Previous tool</td>
<td>43</td>
</tr>
<tr>
<td>Functional Requirements</td>
<td>44</td>
</tr>
<tr>
<td>Technologies used</td>
<td>45</td>
</tr>
<tr>
<td>Navigation flow</td>
<td>46</td>
</tr>
<tr>
<td>Developing of functionality two &quot;Publish Data Models&quot;</td>
<td>52</td>
</tr>
<tr>
<td>Previous tool</td>
<td>52</td>
</tr>
<tr>
<td>Functional Requirements</td>
<td>53</td>
</tr>
<tr>
<td>Technologies used</td>
<td>55</td>
</tr>
<tr>
<td>Navigation flow</td>
<td>55</td>
</tr>
<tr>
<td>Developing of functionality three &quot;Developing tuning support tool&quot;</td>
<td>63</td>
</tr>
<tr>
<td>Previous tool</td>
<td>63</td>
</tr>
<tr>
<td>Functional Requirements</td>
<td>67</td>
</tr>
<tr>
<td>Technologies used</td>
<td>68</td>
</tr>
<tr>
<td>Navigation flow</td>
<td>68</td>
</tr>
<tr>
<td>Join tools together</td>
<td>71</td>
</tr>
<tr>
<td>Real plan of work done</td>
<td>74</td>
</tr>
<tr>
<td>FUTURE AND UPDATES</td>
<td>76</td>
</tr>
<tr>
<td>PROJECT COST</td>
<td>77</td>
</tr>
<tr>
<td>THANKS</td>
<td>78</td>
</tr>
<tr>
<td>BIBLIOGRAPHY</td>
<td>79</td>
</tr>
<tr>
<td>LETTER FROM THE DIRECTOR</td>
<td>80</td>
</tr>
</tbody>
</table>
Introduction

Amadeus

Before my arrival in Amadeus, I did not know much about the company, the only I knew was that Amadeus was an IT company dedicated to design, develop, and implement software for flight companies and airport management, but it was just what I knew... At that moment I had just arrived in “Côte d’Azur”, the mediterranean french riviera, ready to improve myself as a computer scientist, learn about the company, about new technologies, and definitely do my final degree project.

La Promenade des Anglais, Nice.

History

In the late 60s, the main US airline companies started developing and implementing CRSs (Computerized Reservation Systems). Years later, in mid seventies, they decided to install terminals in travel agencies, giving access only to limited data. But as a result of the pressure of travel agencies and different official organizations, the CRSs started moving towards neutral systems showing all competitors’ information. In the 80s, US CRSs were increasing in sophistication and they began to look overseas for opportunities to expand.

In the meanwhile, in Europe, many of the national airlines had developed their own reservation systems and distribution networks. These, however, only served their respective national markets. Imminent deregulation of the European travel industry made it imperative to create distribution systems able to serve the European and global market.

In 1987, responding to these needs and business opportunities, Air France, Iberia, Lufthansa and SAS pooled their resources in a project called Amadeus. In 1989 Amadeus was the first non-US GDS (Global Distribution System) to offer a neutral flight availability display and became fully operational in 1992.

Nowadays, Amadeus has become the world leader of Global Distribution Systems (GDS) serving the needs of the world’s travel and tourism industries. It provides the computer
network, the terminals, the software and sometimes the content that allows airlines, hotel chains, car rental firms, ferry and cruise lines, train operators and insurance agents to distribute travel products to travel agents, corporations or end-user customers over the world. Amadeus operates everything that enables travel agents and travel service providers to market and to sell travel in over 200 markets around the globe.

![Graph showing billable bookings by GDS from 1994 to 2004](image)

Amadeus has become a pervasive technology partner for all the players of the leisure and travel industry, by investing on travel IT and e-travel.

Localization

The main corporate, development and operational activities are split among three main locations across Europe, responding to the origin of the companies who mainly founded the company (Air France from France, Iberia from Spain and Lufthansa from Germany):

The main three locations of Amadeus are:

- Headquarters, commercial and marketing: Madrid, Spain
- Product development: Sophia Antipolis, France
- Data processing centre: Erding, Germany

Additional IT services centers are located in London (UK), Miami and Boston in the USA and Sydney in Australia. Also made according to the new airlines who joined Amadeus, or new companies who became customers of Amadeus.

Amadeus has subscribers in more than 215 markets worldwide, covering the local needs of those different markets with over 70 National Marketing Companies (NMCs). Over 5,300 people work in the Amadeus worldwide group, which represents 95 different nationalities.
System

Today, Amadeus processes over 400 million bookings annually. Through the Amadeus System some 67,000 travel agency locations and some 10,000 airline sales offices around the world are able to make bookings with:

- some 490 airlines, representing more than 95% of the world's scheduled airline seats
- 51,000 hotel properties
- some 45 car rental companies, serving over 29,000 locations
- other travel provider groups (ferry, rail, cruise, insurance companies and tour operators)

![Diagram](image)

*Amadeus is connecting providers with points of sales*

The Amadeus data center (in Erding, Germany) is one of the Europe's largest civilian data processing centers. Today, it records data on a total amount of 22 terabyte of data storage disk capacity.

The core of the Amadeus' System are 6 IBM computers with a total of 30 processors running the IBM Real Time Operating System TPF (Transaction Processing Facility). TPF is a specialized operating system, offering very high throughput, very fast response times and very high availability. It is only used by about 50 companies worldwide, most of them in the travel industry.
- The network is attached to the Front End computer, which handles all the communication software. The message rate handled by the Front End computer exceeds 3,000 transactions per second.
- 5 computers form the Back End complex. This is where the application software runs (more than 20,000 programs) and to which the main database is attached. More than 70 millions of end-user requests are processed in these every day.

Also, clustered Unix-based systems support sophisticated access to data base systems. For example, one of the Unix clusters handles a wide range of communication protocols and conversions that are endemic to the travel industry. It is designed to complement the Global Core Front End and allows to expand rapidly market needs in terms of different communication protocols. Some new applications developed by Amadeus are based on the Linux operating system, which is widely used in the server area, and on Oracle databases and on C++.

Currently, the company is in a process of migration from de IBM-TPF Back Ends to the “Open Back Ends” with Unix-SUSE management system, Oracle Database core and Open source applications, as far as possible.
The Department

The department and the team I join in order to do my project is a Product Development team, which is physically located in the Sophia Antipolis site. Within Sophia Antipolis site, it is in a Global Core huge department (The development teams in charge of develop, basically, the processes of the domain layer linking with Data tier teams in Erding, Germany, and with the presentation development departments located in Sophia Antipolis too). From Global Core, I work in the Development Support department and specifically in the Database support, in fact, our customers are the Amadeus developers.

![Diagram of Global Core and its departments]

Definitely, the team I join (DBS) is the team that helps and develops tools to make things easier to all the developers who work with databases and it is, furthermore the administrator of all the databases.

The following diagram shows the work process of databases in Amadeus. First of all, the customers gain access to the system through a multiplexer that redirects to the right Back End application, which is, at the same time, supposed to access data.
One of the problematics of the team and all global core is that the process of migration started to change from IBM-TPF Back End applications to Open Back End applications. This is, in fact, one of the most important reasons to develop the tool/project I was requested to.
Previous knowledge

First of all one of the main reasons for applying this internship was because I have always had a special interest in Database. When studying my university studies in FIB (Facultat d'Informàtica de Barcelona), I coursed as many Database related subjects as possible. However, because of a change in the curriculum I was not allowed to study BDA "Advanced Databases" subject. After that background in Databases using SQL/PL Language, hosting SQL languages like SQLJ and JDBC, overviewing relational Database Management Systems like Informix, SQL-Server and Oracle, the internship offered in "Database Model Publishing" has other skill and technical requirements.

The other knowledge in computer science required are, a good base in html and all the other involved web and net technologies and languages like php, javascript, css, cgi... as in scripting languages like Pearl, Kom shell... and of course in C/C++ programming language, cause it is the language used to develop applications in the Global Core department in Amadeus. Before start the internship I had a superficial knowledge of most of this languages and technologies that I learn as a student in Barcelona and Italy in my computer science degree, and Amadeus and the internship are a great chance to improve myself in these fields, and learn deeply about these languages too.

Of course, is also required to work/know either of Unix/Linux and Microsoft Windows operating systems.

It is also mandatory to has a good English level, and I have now the chance to improve my English and get some basic skills in French langueage, good for me as a person and for my C.V.
The Project

Product Definition

The current environment of Database Support team and, in fact, of Amadeus, is the coexistence of two kinds of Back Ends to connect and manage data. The first one is TPF (Transaction Processing Facility) from IBM that provides to developers all functions and applications needed to access data. The other is called Open Back End, and it is a set of tools and applications, mainly in opensource, which have to provide the same functionalities to the developers as TPF. In the following years, TPF would disappear and Open-Back-End system would be the only one, therefore, it has to be improved to be totally accurate. The migration plan has already started.

Target of the project

As a part of the migration plan, all the developers will need new tools to work with the Database in the Open Back End applications. Some of these tools should be tested and developed (if it is necessary) by development support teams. Therefore, this project concerns the development of a new tool to help the database developers in the new environment.

The target of the tool would help developers in the following tasks:

- Developing of new Data Models.
- Understanding & Maintenance of existing Data Models.
- Developing of patches for the existing Data Models.
- Support to the tuning support.

However, nowadays, the company is already working with Open Back End applications, and these functionalities are covered by existing tools.

Then, what is the real target of the project?

Well, the new Open Back End environment is providing all these tools to the developers but these tools should be improved since they have two problems mainly

The first one is, the not user friendly tools or absence of GUIs in the tools.

Developers have to type in SQL code all the patches and all the new creations, and it is, sometimes, a heavy work for developers. that could be changed for some mouse clicks in a graphical developing tool.

To execute the tools that help them (the database developers) in understanding & maintenance of Data Models, or generating the delivery of a patch, they must type long commands in the CLI tools as, for example:

```java
java -jar
/projects/obedelde/OracleTools/latest/bin/Java/schemaSpy.jar -cp /nastools/oracle/products/9.2.0.7/jdbc/lib/classes12.zip -t $1 -host $2 -port $3 -db $4 -u $5 -s $6 -p $7 -o ./schema_spy/patch -cid -i "REF_CSH_XXX|REF_CSH_XXX"
```

The second one is the isolation of the tools. First, developers have to develop a tool, after this, they work with a linux box to create the patches' deliveries, and execute the understanding and maintenance tool, then look for the results in html format, and finally go to the tuning support tools in the intranet of Global Core.

Consequently, according to what has previously said, the real target of the project is:

- Find and test user friendly tools, and develop interfaces for the existing tools.
- Join all the tools in the same framework, develop this framework as a website.

Therefore, the first step in my project is to analyse all the tools already working, test and work with new suggested tools (which have been proposed/suggested by my team leader as a director of this project) in order to take advantage of all the work previously done for other members of the team, keeping the useful tools, and only adding a GUI (Graphic User Interface) above them.

It is also important to know how are the developers working with the Data Models because it will be necessary to understand future functionalities and explanations. Having an overview
of that and so as to simplify things, we could say that all developers in Nice site work on rehearsal databases, after developing and testing patches and new Data Models in these databases, they develop a delivery to Germany site (in Earding), where every patch will be applied on every concerned database.

Develop the patches for every distinct application  
Run and test the patch on a Rehearsal Database (with empty schemas)  
Execute the patch on the database

Nice  
Erding (Germany)

After that, the developers work on applications and inside there are schemas representing the Data Models, every instance of this schema is a replica of this schema but, of course, filled with its own information. For example, we can have a schema called AIR_COMPANY and all the air companies (instances) have its own replica of this schema.

Then, the behaviour is to work with generic schemas, in the example AIR_COMPANY (a.k.a AIR), and after that, and using synonymous, apply everything to all the air companies' schemas.
Existing and purposed tools

Nowadays, databases developers in Open Back End side work with several not joined tools as we can see in the diagram below:

![Diagram showing relationships between tools like SQL Plus, SQL Developer, Oracle tools, Transform script, View script, Schema spy, Global Core Intranet, and Statspack.]

First of all, we will try to understand the existing final applications, because they are already used by developers and could be perfect to take the most profit from them.

**SQL Plus and SQL Developer**

**SQL*Plus** is an Oracle command line utility which allows users to run SQL and PL/SQL commands interactively or from a script. But as the developers have to deliver exactly the scripts updated with the synonyms to Erding (Amadeus in Germany), is not necessary to have the chance to run them.

SQL*Plus is a simple tool with a basic command line interface, however it is commonly used as it is almost always available in any Oracle software installation.

SQL*Plus's predecessor was called UFI (User Friendly Interface). UFI was included in the first Oracle database releases up to Oracle v4.
The UFI interface was extremely primitive and, in today's terms, anything but user friendly. If a statement was entered incorrectly, UFI issued an error and rolled back the entire transaction.

Complete documentation for the SQL*Plus interpreter can be found on the Oracle website, is not attached in the annexe cause one of the targets of this project is to find and work with another tool (user friendly) replacing SQL*Plus, for this specific framework.

SQL Developer is Oracle's free graphical (But also in SQL scripting way of work) database development tool. While it is aimed at Oracle developers, it does include connectivity to MySQL, SQL Server and Access databases, mostly for the purposes of migration.
It is a Java-based tool, and it is therefore available for a wide range of platforms including Windows, Linux and Mac OSX. Its major competitors are commercial products, including TOAD from Quest Software and PL/SQL Developer from Allround Automations. It is a relatively recent entrant into the market and is not as full-featured as the commercial products.

It is packaged as part of the 11g database, with the intention of becoming an adjunct to SQL*Plus.

As SQL*Plus, the user guide is not attached in the annexe for the same reasons.

Oracle tools

First of all, as an overview, the Oracle Tools are a set of tools enabling to manage SQL scripts in a multi-schema context. They can be used:

- To create SQL scripts dedicated to each schema
- To apply each script to the correct schema
- To perform operations like managing the version of each schema, dropping the schema, generating a script to recreate the schema...
- Allow the display of all schemas in a graphically way.
- ...

These tools have been developed by OBE developers, and they are command line interface tools. The transform _sql would be necessary/interesting since the tools allow applying each script on the correct schema, and view_schema, which provides graphically representations of schemas, and moreover they are a useful help in the framework developing.

Transform tool

The “transform_sql” module of the Oracle Tools is used by the development team to generate SQL scripts specific to each schema type. Joined/put together in a “tar” file, these generated SQL scripts constitute the “data-model delivery” for ADP.

From some inputs, mainly the scripts generated in generic schemas, transform_sql generates using the synonymous in configuration.xml file and in distribution folders, the specific scripts for each schema, and make them ready to be executed in the definitive database.
View tool

The "view_schema" module generates html pages providing a graphical view of the schemas defined in the configuration.xml file as input.

In order to generate the html pages, view_schema tool is linked to Schema Sky application and it is explained below.
SchemaSpy is a Java-based tool that analyzes the metadata of a schema in a database and generates a visual representation of it in a browser-displayable format. It lets you click through the hierarchy of database tables via child and parent table relationships.

The browsing through relationships can occur through HTML links and/or through the graphical representation of the relationships. It is also designed to help resolve the obtuse errors that a database sometimes gives related to failures due to constraints.

The execution of SchemaSpy is by command line, with a command like "java -jar schemaspy.jar -t dbType -db dbName [-s schema] -u user [-p password] -o outputDir [-nohtml][-noimplied]" and the output is a website like follows.

---

### SchemaSpy Analysis of REH5.DBANGIFPDM_P

**Generated by SchemaSpy on Wed Mar 19 16:54 MET 2008**

**Database Type**: Oracle - Oracle9i Enterprise Edition Release 9.2.0.8.0 - 64bit Production With the Partitioning, OLAP and Oracle Data Mining options Solaris 9.2.0.8.0 - Production

104 Tables:

<table>
<thead>
<tr>
<th>Table</th>
<th>Children</th>
<th>Parents</th>
<th>Rows</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALTNL_NAME</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>ALTNL_NAME.TableName</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>ALTNL_NAME.TableName.TableName</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>AIRCARRIER_NAME</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>AIRCARRIER_NAME.TableName</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>AIRCARRIER_NAME.TableName.TableName</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>AIRPORT_NAME</td>
<td>4</td>
<td>3</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>AIRPORT_NAME.TableName</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>AIRPORT_NAME.TableName.TableName</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>AIRLINE_NAME</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>AIRLINE_NAME.TableName</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>AIRSIZE_NAME</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>AIRSIZE_NAME.TableName</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>AIRSIZE_NAME.TableName.TableName</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>AIRWAY_NAME</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>AIRWAY_NAME.TableName</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

### SchemaSpy Analysis of REH5.DBANGIFPDM_P - Relationships Graph

Generated by SchemaSpy on Wed Mar 19 16:54 MET 2008

- Include implied relationships
- Compact

---
SchemaSpy is a free application, all the documentation is available in http://schemaspy.sourceforge.net with the download, samples, and an online brief User Guide.

It could be useful if instead of displaying all schemas, would be possible to display just some of them (a few tables and their relationships, columns...), to make it clearer to the developers. To make this possible is another target of this project.

Statspack

Statspack is a set of performance monitoring and reporting utilities by Oracle. Statspack provides improved BSTAT/ESTAT functionality, though the old BSTAT/ESTAT scripts are still available.

In Amadeus the tool and the output raports can be displayed in the Global Core intranet, there is a huge number of raports and snapshots to run availables.

Oracle Designer
Now, in order to replace (for this framework) the SQL scripting tools like SQL Developer or SQL*Plus, the director of my project, and team leader too, proposed/suggested to test deeply the Oracle Designer application, according to the relationships with Oracle company and the database of the Open Back Ends.

**Oracle Designer** is Oracle’s CASE tool for designing an information system and generating it. After generating the information system one is able to edit the generated code.

Then, the analysis of this tool is another target of this project.

**Conclusion**

After overviewing the existing tools and the proposed/suggested graphically tool, the specific project targets, so as to satisfy the targets seen above, are the following:

- Test and verify Oracle Designer tool in order to get proper outputs.
- Develop a tool in order to link the Oracle Designer outputs to inputs in the Oracle Tool transform_sql.
- Develop a tool in order to link the Oracle Designer outputs to inputs in Oracle Tool view_schema and use also Oracle Designer outputs as SchemaSpy inputs directly in order to display just few parts of the schema.

- Purpose an alternative way to access to the snapshots of Statspack, because the current way by the intranet is not appropriate.

- Create a website/interface to involve all the tools and functionalities in the same framework.

Then, from the current environment, neither SQL*Plus and SQL Developer nor the intranet Statspack flow will be not used in this project.

Maybe a local repository could be the solution for the Statspack snapshots.
Developing

After an overview of the company, the environment, and the subjects and targets of the project, everything is ready to start with developing.

I have about six months to develop the framework, because my internship deal is that long. I have already spent three weeks just getting a background of the company and I will be one week at home for Christmas, then, in fact, I have 5 months to develop the tool.

To begin with, I have to spend time getting to know deeply how to work with Oracle Designer, install it, import data from the developers’ databases, modify data, create outputs... After that I will be able to start working with that outputs, and develop the tools to transform the Oracle Designer outputs in Oracle Tools inputs, and also in SchemaSpy inputs.

Then, at that moment I will develop the tool to make a patch delivery from the OD (Oracle Designer) outputs (aka “Patch Delivery”), after that, also with OD outputs, I will develop a tool to create the displays of the schemas with the view_schema Oracle Tool as it is currently working, and also to display parts of these schemas directly with the SchemaSpy application (aka “Publish Data Models”). The last point will be the design of some interesting tool to link directly to a repository where the appropriate snapshot Statspack reports will be stored. During all the process of these tools development, I will also be designing the Website Interface and integrating the tools already finished.

The technologies I will use to do the project will be known tools as a background, but it will give me the chance to know about them deeply, and the prevision is something like:

- For the WebSite: Html, php, javascript, css.
- Script languages like Korn shell, and C language to work with Oracle Designer outputs.
- Cgi's to connect the server side where the WebSite will be, and the developers' side.
- Also xml, to get information from the configuration file, used in oracle tools or from db_topology file to get the information of databases and applications from Amadeus.
Plan of work

I used the GanttProject tool (free tool) to create the gantt diagram of the project, which has been devided in four main tasks: Oracle Designer, Script designer (new aka for "Patch Delivery"), Publish Designer (new aka for "Publish Data Models") and Snapshots tool. I have planned it with accurated times to satisfy every task and every delivery in the foreseen dates.

After every main task would be done, I will take some days so that documentation, slide presentation and document writing could be done/carried out.

In order to satisfy with my full time contract (37 hours per week), my working timetable is as follows: from Monday to Thursday (from 9:00h till 17:30h with one our for lunch) and Fridays (from 9:00h till 17:00h with the same lunch break), but the time flexibility of the company, allows me to manage the starting and ending times as I need.

I start with this planning having already spent three weeks getting a company and department backgroung.

I am the only resource/worker to satisfy that project timing, but I feel free to ask for support to all the members of my team including, of course, my team leader.
<table>
<thead>
<tr>
<th>Task definition</th>
<th>Start</th>
<th>End</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Knowledge of company environment / overview of the</td>
<td>1-10-07</td>
<td>27-10-07</td>
<td>Get some background of the company department and project.</td>
</tr>
<tr>
<td>project</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Installation/configuration of Oracle Designer</td>
<td>29-10-07</td>
<td>1-11-07</td>
<td>Install tool.</td>
</tr>
<tr>
<td>Oracle Designer testing</td>
<td>1-11-07</td>
<td>26-11-07</td>
<td>Working with the tool.</td>
</tr>
<tr>
<td>Script Designer tool developing</td>
<td>26-11-07</td>
<td>21-12-07</td>
<td>Start with the first tool, Design + Developing.</td>
</tr>
<tr>
<td>Test</td>
<td>21-12-07</td>
<td>26-12-07</td>
<td>Test.</td>
</tr>
<tr>
<td>Delivery-Document</td>
<td>21-12-07</td>
<td>27-12-07</td>
<td>Prepare slides and documentation to my project director.</td>
</tr>
<tr>
<td>Publish Designer tool developing</td>
<td>27-12-07</td>
<td>18-1-08</td>
<td>Start with the second tool, Design + Developing.</td>
</tr>
<tr>
<td>Test</td>
<td>18-1-08</td>
<td>23-1-08</td>
<td>Test.</td>
</tr>
<tr>
<td>Delivery-Document</td>
<td>18-1-08</td>
<td>25-1-08</td>
<td>Prepare slides and documentation to my project director.</td>
</tr>
<tr>
<td>Snapshot tool Developing</td>
<td>25-1-08</td>
<td>22-2-08</td>
<td>Start with the last tool, Design + Developing.</td>
</tr>
<tr>
<td>Test</td>
<td>22-2-08</td>
<td>27-2-08</td>
<td>Test.</td>
</tr>
<tr>
<td>Delivery-Document</td>
<td>22-2-08</td>
<td>28-2-08</td>
<td>Prepare slides and documentation to my project director.</td>
</tr>
<tr>
<td>Website Developing</td>
<td>21-12-07</td>
<td>8-2-08</td>
<td>Develop the interface integrating the already finished tools</td>
</tr>
<tr>
<td>Join test</td>
<td>28-2-08</td>
<td>5-3-08</td>
<td>Test everything together.</td>
</tr>
<tr>
<td>Demo</td>
<td>5-3-08</td>
<td>19-3-08</td>
<td>Prepare a demo to my project director and manager of the department.</td>
</tr>
<tr>
<td>Final Report</td>
<td>5-3-08</td>
<td>4-4-08</td>
<td>Join information and write the current report.</td>
</tr>
</tbody>
</table>

In a six month stage there are no holidays and only 1st of November, 25th of December and 1st of January are free days.
Oracle Designer working

Previous tool

There are not existing tools in the current environment, in order to work with Data Models graphically. In fact, one of the targets of this project is to find, test, and definitely validate a tool (Oracle Designer) to join the other OBE (Open Back End) applications and make the database developers work easier. This tool has to be able to display tables, relationship from a data model, and it has to permit develop all kind of patches and new structures. The current way to develop patches and create new data models is SQL scripting using programs as Sql*Plus or SQL Developer.

Now, to develop patches, the developers type them in SQL code, and after that they have the Oracle Tools mentioned before in order to create the patch for all the synonyms and display the resulting Database Models.

The close relationship between the leaders of my department with Oracle, and with the software involved, made my team leader (also director of my project) suggest me to look for the tools in Oracle software website (specifically about Oracle Designer). Once the main tool would be tested, as long as it would be successfully resolved, it would probably be the definitive tool.

Functional Requirements

The requirements needed to make Oracle Designer as a designing tool, are the following:

- It must to be a user friendly tool. Work with Database Models graphically.
- Allow the developer to generate SQL scripts as outputs with all possible modifications and creations (patches).
- Allow the developer to generate outputs as Graphic files of parts selected of the Database Models.

After installing and configuring properly the Oracle Designer, and after few days of testing, it is easy to know if the program will satisfy perfectly all the functional requirements.

For example, just having a look we can discover that the application displays always the entity-relationship diagram, and most of the modifications can be easily made in the diagram with the mouse.
It is also possible to generate outputs with the modifications in the models (the program connects to the database and compare the existing models with the new/modified models on Designer and only generates the patches for the differences in the matching).

Every kind of modification is permitted by Oracle Designer (Columns, relationships, constraints, indexes...).

It is also allowed to create outputs as a graphic file, with the representation of the desired parts of a model.
Technologies used

This section will try to explain briefly how is the internal way of work of Oracle Designer (aka OD).

Oracle Designer is an application with its own repository, then, the way of work of OD is to import all the data from a real Database to its repository (an Oracle database too). All the modifications are made in the internal OD database, and after that, in order to create patches, OD connect again to the real database matching the differences between the data from its own repository and the data from the target database.
The fact of working with a copy of the data, and not with the data itself, may cause problems of concurrency, but in Amadeus, each database developer is involved only in a set of tables or database models, and he/she is the only one in charge of these models.

**Navigation flow**

This section is a compilation of use cases in Oracle Designer, using screenshots for an easy understanding.

The use cases for our developers in Oracle Designer are the following:

- Creation of the internal Repository.
- Importing database data to the Oracle Designer Repository.
- Working with Database Models.
- Generating outputs for “Patch Delivery”.
- Generating outputs for “Publish Data Models”.

**Creation of the internal Repository**

To work with Oracle Designer it is mandatory to have an internal Oracle database and create it in a repository. Each developer can create as repositories as he/she wants, and it could be
useful to store different releases of the same model. After that, the developer will create the Oracle database inside.

Starting with the Oracle Designer application, it also starts this use case:

**1st step**

![Oracle Designer GUI](image1)

The developer should create a new folder in the Oracle Designer Repository in order to store all the data. Then, it would be possible, just clicking on the **Process Modeler** application button from Oracle Designer main menu.

**2nd step**

![Process Modeler GUI](image2)

Once inside the application, click on the new file icon as you can see at the picture above.
3rd step

It will appear a new dialog window as shown on the picture, clicking on the marked icon is enough to continue with the use case.

4th step

At this moment, create a new folder. Just clicking in the marked icon from the picture above.
5th step

And click OK when the Application Systems option is selected. At this moment you can close this application, because the folder on the Oracle Designer Repository is already created.

Importing database data to the Oracle Designer Repository

In order to work with the real data from Amadeus databases, the developers should import this data from Amadeus Databases to Oracle Designer Repository.

1st step

First of all, when Oracle Designer is just started, choose the Designer Editor section in the main menu.
2nd step

As a subsection of this use case, is mandatory to create the Oracle Database inside the folder created in the use case before that. In order to satisfy this prerequisite, just choosing the DB Admin tab and then opening the folder before created, and clicking, with the right button, on the inner Oracle Databases folder. After that, just follow the menus to create the Database.

3rd step
In order to import all the data, and with the database created (REHE0 in the picture), click the Generate top menu, choose the Capture Design of option and click on the Server Model submenu.

4th step

A new dialog window will appear to manage the import data process. On this window, you must fill in:

**Source of Design Capture:** Fill in the Username, the Password and the connect string in the field Connect, then, fill also in the Owner field.

After that, choose the repository folder and the database created before to fill in the following fields: Target Container, Capture Implementations Info and Database User. Then select the Objects tab.
Here, select all the objects from the database to be imported. The useful objects (for Amadeus developers) to be imported are: All the Packages, Relational Tables and the Sequences, then the User, and the Tablespaces related with the Owner field, and also the Role. Then, just pushing Start button the use case goes ahead.

After that everything will be ready to start working. As you can see in the picture below.
Generating outputs for "Patch Delivery"

The Oracle Designer application provides to developers the generation of Script files with SQL code representing the differences found between the Oracle Database from the repository of OD and the real Amadeus database where the patches will be definitively applied.

1st step

The use case starts when a developer has modified something in a Database Model, and wants to create a SQL script (used later in the "Patch Delivery" tool) the developer selects everything modified or created (using the mouse or the "Ctrl" key). When everything is selected, just click the mouse right button (on one of the selected objects) and choose the option Generate.

2nd step

A new dialog window will appear, here, choose the Database as Target for Generation and fill in the fields Username, Password and Connect, with the connect string to the database. Fill also in the File prefix and Directory fields, with the name of the patch and the name of the folder where the Oracle Designer scripts must to be stored for a properly future use of "transform" tools to generate the ADP delivery. Then clicking on Start button the generation goes ahead. It is very important to fill in the database option inside the Target for Generation, because you need to have a target to compare with the Oracle Designer models, and generate scripts only for the changed objects.
Working with Database Models

Working with Database Models is easy because, as a target of this project, we have looked for a user-friendly tool with easy Graphic interface.

Then, to work with Database Models the developers have a set of tools (in the picture, inside the circle), and you can store parts of the schema as digrams (in the picture, pointed by the arrow) to manage easily everything.

Before start working with Oracle Designer, the 2 first use cases are mandatory to be done at least once. After that, and after working with Oracle Designer, another important thing will be the creation of outputs in order to satisfy the other functionalities included in the framework. To create these outputs, just follow the next use cases explanations.
At this step, the application is running, it proceeds asking the developers to confirm the removed objects and to warn of other objects which might have been modified due to previous modifications. With the dialog window shown in the above diagram, you can confirm or deny the changes.