



INVESTIGATION ON LOWCOST.BPI

GRADUATION REPORT

Author
Georgiana Codreanu





Fontys University of Applied Sciences
HBO-ICT (English Stream)

GRADUATION REPORT
Investigation on LOWCOST.BPI

Data student	
Initials:	G.
Name:	Codreanu
Studentnumber:	2114482
Data company	
Name company/organisation:	Vanderlande Industries
Visiting adress :	Vanderlandelaan 2 5466 RB Veghel
Company mentor	
Initials:	G. T. C. M.
Name:	Jaspers
Department/position	VI Software House - product development/BPI project leader
University mentor	
Initials:	N. H. J. M.
Name:	Aarts
Final report	
Title:	Investigation on LOWCOST.BPI
Date:	08.06.2011

Approved and signed by the company tutor:

Date: 08.06.2011

Signature:



Preface

This graduation report marks the end of the work I performed within the Software House department of Vanderlande Industries (VI) and is written with the goal to fulfill the requirements of the Bachelor program in Information & Communication Technology at Fontys University of Applied Sciences, in Eindhoven.

The assignment was to investigate the options of further reducing the cost of a Business Intelligence (BI) product called BPI, and create the basis for the implementation of a LOWCOST.BPI product. The activities identified for this assignment were to define a set of requirements and a minimum set of reports to be implemented, and to investigate the low cost reporting tools options.

First of all, I would like to thank my company supervisor Mr. Gonard Jaspers for initiating this graduation project, for his support during the last six months and for giving me the opportunity to work under his supervision. Secondly, I would like to thank my school tutor Mr. Henk Aarts for his guidance and advises. Furthermore I would like to thank my colleagues Lifiester Hosana and Ze Wen Oh for their support.

Georgiana Codreanu, June 2011



Table of contents

PREFACE	3
SUMMARY	7
GLOSSARY	8
INTRODUCTION.....	9
ABOUT VANDERLANDE INDUSTRIES	10
2.1. SOFTWARE HOUSE - PRODUCT DEVELOPMENT.....	10
2.2. TARGETED MARKET SEGMENTS	10
2.1.1. <i>Baggage Handling</i>	10
2.1.2. <i>Distribution</i>	11
2.1.3. <i>Parcel & Postal</i>	12
OVERVIEW OF THE BPI PRODUCT	13
3.1. BPI WAY OF WORKING.....	13
3.2. BPI FEATURES.....	14
3.3. VITAL.BPI	15
3.4. QLIKVIEW	16
ASSIGNMENT OVERVIEW	17
4.1. PROBLEM DEFINITION	17
4.2. ASSIGNMENT DESCRIPTION AND OBJECTIVES	17
4.3. PROJECT PHASES	17
4.3.1. <i>Planning – Create project plan</i>	18
4.3.2. <i>Project management – Write graduation report</i>	18
4.3.3. <i>Start-up phase – Requirements Analysis and Reporting tool Investigation</i>	18
4.3.4. <i>Iteration 1 – KPI's Requirements, Design, Implementation</i>	18
4.3.5. <i>Iteration 2 – KPI's Requirements, Design, Implementation</i>	18
4.3.6. <i>Iteration 3 – KPI's Requirements, Design, Implementation</i>	18
PROJECT APPROACH.....	19
5.1. APPROACH OF THE PLANNING PHASE	19
5.2. APPROACH OF THE PROJECT MANAGEMENT PHASE.....	19
5.3. APPROACH OF THE START-UP PHASE	20
5.4. APPROACH OF THE FIRST ITERATION.....	20
5.5. APPROACH OF THE SECOND ITERATION.....	20
5.6. APPROACH OF THE THIRD ITERATION	20
5.7. APPROACH OF THE EVALUATION	21
REQUIREMENTS ANALYSIS	22
6.1. REQUIREMENTS CATEGORIES.....	23
6.2. KPI's	24
6.3. CONCLUSIONS AND REMARKS.....	24
RESEARCH	25
7.1. CREATING THE LONG LIST	26
7.2. CREATING THE SHORT LIST	26
7.1. JASPERREPORTS	27



7.1.1. JasperServer	27
7.1.2. iReports	28
7.2. BIRT	29
7.2.1. BIRT report designer	29
7.2.2. BIRT deployment environment	30
7.3. OPENREPORTS	30
7.4. CONCLUSIONS AND REMARKS	31
IMPLEMENTATION	32
8.1. NUMBER OF PARCELS PER INFEED	36
8.2. NUMBER OF PARCELS PER CHUTE	37
8.3. LENGTH OF STAY OF A PARCEL IN THE SYSTEM	38
8.4. CONCLUSIONS AND REMARKS	39
RESULTS	40
9.1. REQUIREMENTS LIST	40
9.2. RANKING OF THE REPORTING TOOLS	43
9.3. COSTS BREAKDOWN	45
9.3.1. Engineering effort	45
9.3.2. License costs	45
9.3.3. Hardware costs	45
9.3.4. Costs comparison	46
CONCLUSIONS	48
REFERENCES	49
APPENDIX A – ORIGINAL ASSIGNMENT	50
APPENDIX B – PROJECT PLAN (FONTYS)	52
APPENDIX C – PROJECT PLANNING	70
APPENDIX D – REPORTING TOOLS COMPARISON	71
APPENDIX E – RESEARCH LOG-BOOK	76
APPENDIX F - PACKAGE EXAMPLE	79
APPENDIX G – REPORT FILE (JASPERREPORTS)	86
APPENDIX H - REPORT FILE (BIRT)	89



List of tables

TABLE 1 – REPORTING TOOLS FEATURES	27
TABLE 2 – WAREHOUSE CORE TABLE WCT_7FSC_SORTREPORT	35
TABLE 3 – DM_LOWCOST_INFEED FIELDS	36
TABLE 4 – DM_LOWCOST_CHUTE FIELDS	37
TABLE 5 – DM_LOWCOST_PARCEL FIELDS	38
TABLE 6 – MoSCoW CHART	43
TABLE 7 – REPORTING TOOL RANKING BASED ON FEATURES	43
TABLE 8 – REPORTING TOOL RANKING BASED ON REQUIREMENTS	44
TABLE 9 – COSTS BREAKDOWN	45
TABLE 10 – HARDWARE CONFIGURATION	46
TABLE 11 – COSTS COMPARISON	46
TABLE 12 – COSTS COMPARISON - EXTRA KPI	47

List of figures

FIGURE 1– BAGGAGE HANDLING	11
FIGURE 2 – DISTRIBUTION	11
FIGURE 3 – PARCEL & POSTAL	12
FIGURE 4 – BPI ARCHITECTURE	13
FIGURE 5 – BPI CORE AND BPI APPLIANCES	14
FIGURE 6 – VITAL.BPI	15
FIGURE 7 – QLIKVIEW MAIN SCREEN	16
FIGURE 8 - LOWCOST.BPI STAKEHOLDERS	22
FIGURE 9 - JASPERSERVER – LOGIN PAGE	28
FIGURE 10 - JASPERSERVER – REPOSITORY PAGE	28
FIGURE 11 – IREPORTS MAIN WINDOW	29
FIGURE 12 – BIRT REPORT DESIGNER	30
FIGURE 13 - OPENREPORTS – REPORT ADMINISTRATION WINDOW	31
FIGURE 14 – DATAMART	33
FIGURE 15 – DM_LOWCOST_INFEED FLOWCHART	36
FIGURE 16 – DM_LOWCOST_CHUTE FLOWCHART	38
FIGURE 17 – DM_LOWCOST_PARCEL FLOWCHART	39



Summary

The BPI product implemented by Vanderlande Industries (VI) is a reporting/analysis server which provides analyzing, diagnosing and reporting capabilities that can help customers make better decisions regarding their business processes. The BPI is applied in the three market segments targeted by VI, which are Baggage Handling (VIBES.BPI), Parcel & Postal (VITAL.BPI) and Distribution (VISION.BPI).

An investigation regarding the development of a LOWCOST.BPI appliance is proposed in this report. The problem is raised by the Parcel & Postal (P&P) stakeholders who consider that is necessary to have a low cost version of VITAL.BPI. The ways in which the price of VITAL.BPI could be reduced are to cut the license costs, limit the number of engineering hours which are necessary for installation and configuration of the BPI and implementation of reports, and decrease the hardware costs.

The first step of this investigation was to interview the stakeholders. During these interviews the LOWCOST.BPI requirements were defined based on which the research for a new reporting tool was started. To choose this reporting tool, different alternatives were compared and the best two were picked.

The next step was to implement reports and rank the reporting tools based on their functionalities. The implementation part was seen as a way to investigate the reporting tools more thorough and discover possible incompatibilities with the defined requirements.

The final step of this investigation was to create the results documents:

- Requirements list
- Reporting tools ranking
- Costs breakdown

Each of these documents is very important for the future development of the LOWCOST.BPI appliance. The requirements list contains data aligned with both VITAL.BPI and VISION.BPI stakeholders and will be a reference point in the next phase of the LOWCOST.BPI project.

The ranking of the reporting tools shows the important differences between the two technologies and can have an essential role when the final selection will be made.

The most expected deliverable of the project is the costs breakdown document which shows that the results of this investigation satisfy the main requirement of the stakeholders – to have the proof that low cost BPI appliance can be implemented.



Glossary

Term	Description
BI	Business Intelligence
BPI	Vanderlande Industries Business Process Intelligence.
DS	Destination Server
FAT	Factory Acceptance Test
FSC	Flow System Controller
GL	Group Leader
HLC	High Level Controls
KPI	Key Performance Indicator (reports)
LSG	Logistic Software GmbH
N/A	Not Applicable
PLC	Programmable Logic Controller
PR	Problem Report
PSR	Project Status Report
P&P	Parcel & Postal
SAC	Sort Allocation Computer
SW	Software
SWH	Software House
VI	Vanderlande Industries
VIBES.BPI	Vanderlande Industries BPI product for the Baggage Handling market segment
VISION.BPI	Vanderlande Industries BPI product for the distribution market segment
VITAL.BPI	Vanderlande Industries BPI product for the Parcel & Postal market segment



1

Introduction

During the last decade there has been a powerful and rapid rise of information technologies coming from the need of companies to control enterprise data and convert it into useful information. This technique, known as Business Intelligence (BI) includes technologies such as Data Warehousing, Data Analysis, Data Mining, Data Backup and Data Recovery. BI helps companies increase their business value by applying all its technologies in analyzing and organizing data.

Having a BI system in place can help organizations get answers to questions related to their operational processes. This type of system provides analyzing, diagnosis and reporting functionalities which will help companies improve their operational efficiency. Independent of the company size, the capacity to gather, analyze, diagnose and report operational data is essential.

A Business Process Intelligence (BPI) product has been implemented by VI and it's used for analyzing, diagnosing and reporting data that goes through the systems they manufacture. Through reports, defined specifically for each market segment (Baggage Handling, Distribution, and Parcel & Postal) the BPI product helps VI's customers improve their business processes.

Offering a large number of functionalities and a dynamic reporting environment the BPI proves to be expensive for some of the smaller clients of VI. For this reason the VITAL.BPI stakeholders have requested to the product development team to investigate options to further reduce the costs. Therefore, this thesis introduces several aspects regarding this subject.

As a first step an introduction about VI and the BPI product is given in chapters 2 and 3. Chapter 4 presents the assignment overview and details about the project phases and in Chapter 5, 6 and 7 the requirements analysis, research and implementation phases are described. The results of the project are presented in Chapter 9 and the final conclusions and recommendations are given in Chapter 10.



2

About Vanderlande Industries

Vanderlande Industries is a Dutch company founded as a family business, in 1949 by Eddie Van Der Lande and is best known as a supplier and manufacturer of airport baggage handling systems. The company targets the markets for Baggage Handling systems, Distribution centers and P&P sorting facilities. The systems that VI implements are of all sizes, from local airports (Eindhoven Airport), distribution centers and sorting depots, to the world's largest facilities (London Heathrow Airport). In its field the company ranks among top 5 and in baggage handling is one of the leading suppliers.

The company possesses core competences in all relevant disciplines, ranging from system design, engineering, and manufacturing to supply management, information technology, project management and customer services. VI also offers a lot of opportunities for students, through a large number of internships and graduation projects.

The company employs over 2000 people of whom more than 50% have a college or university degree. Vanderlande Industries is a global player with a presence in all key regions of the world. The company has subsidiaries in the Netherlands, Belgium, Germany, France, Great Britain, Spain, Canada, PR China, South Africa and the USA. These Customer Centers handle all key business functions and maintain direct contacts with customers.

2.1. Software House - Product Development

The Software House (SWH) department, was started on April 1st, 2007 by VI, as a project-supporting department with a specific focus on High Level Control (HLC) – engineering. The objectives of SWH as a new department are to strengthen VI's position as service provider, to enlarge total sales and profit, and to attract, develop and retain software engineers.

Product Development is the part of SWH that focuses on developing and maintaining software products like BPI, CM, SAC, and DS.

2.2. Targeted market segments

2.1.1. Baggage Handling

VI designs, builds and services leading baggage handling systems for airports of all sizes. The solutions the company offers combine operational effectiveness, short connection times and high



conveyability. They deliver the highest availability, reliability and lowest costs per bag. Some of the solutions that VI offers in the baggage handling segment are shown in the figure below.

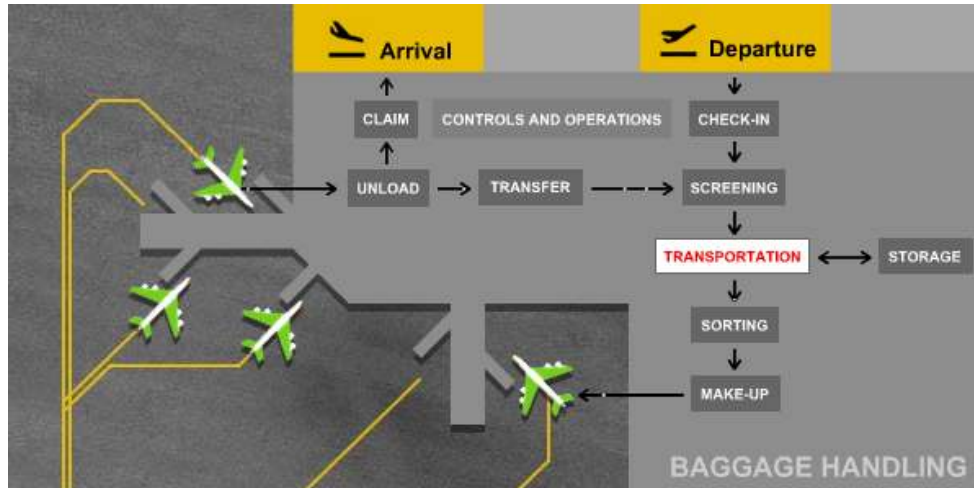


Figure 1– Baggage Handling

2.1.2. Distribution

VI is one of the leading suppliers of integrated logistics systems for automation of warehouse/ distribution centers. The focus is on goods flow, from goods receiving, storage, order picking up to shipping and the related flow of information.

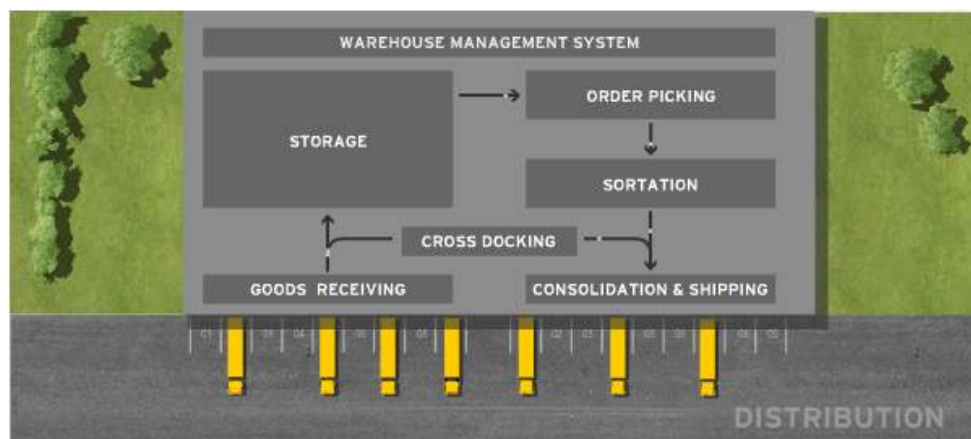


Figure 2 – Distribution

The VI logistic solution includes:

- Warehouse Management Systems;
- Order picking
- Automated storage and retrieval systems
- Sorting systems
- Conveyor systems



Customer that have automated their warehouse and distribution centers vary from care, food and fashion to automotive, components and parts and retail.

2.1.3. Parcel & Postal

Increasing labor costs and expanding volumes make automated sorting systems attractive for both express delivery and postal operators. VI provides a wide range of automated parcel and postal sorting solutions for hubs and depots of all sizes. The automated solutions that VI offers for hubs and depots are shown in the figure below.

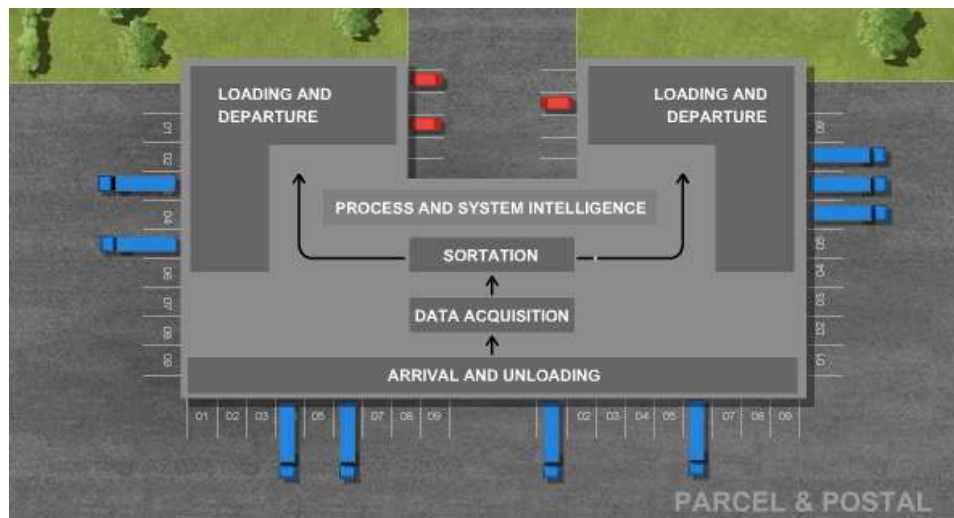


Figure 3 – Parcel & Postal



Overview of the BPI product

The BPI product is a reporting/analysis server used to gather, analyze, diagnose and report data in order to help customers take better business decisions. BPI collects relevant data from material handling systems under production. The gathered information is processed in order to report the system and process performance, and is used as an input for the improvement of the operational processes.

Internally, the BPI is used for problem solving and maintenance purposes. The service department uses BPI to get relevant information about the components that need to be serviced.

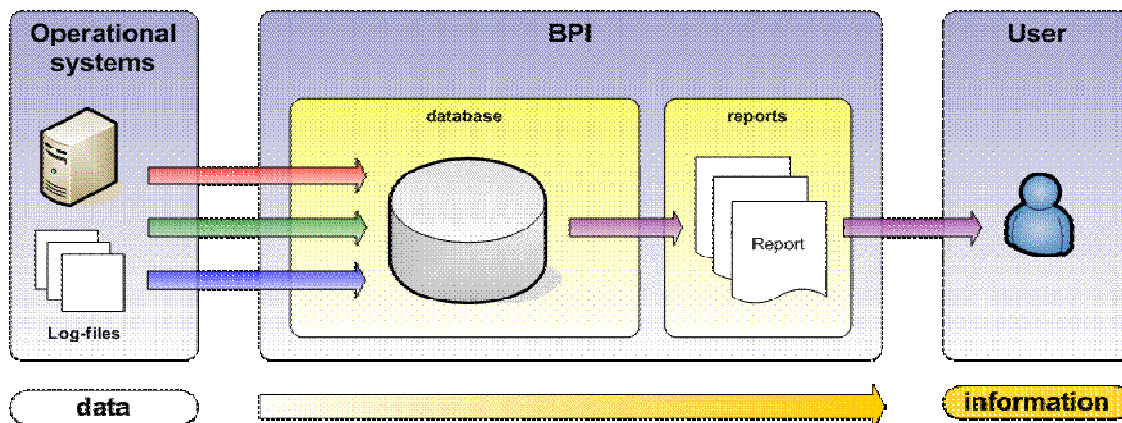


Figure 4 – BPI architecture

BPI is a non critical analyzing tool that focuses on mid to long term data. If for some reason the BPI is not working properly, the daily operations of a business will not be affected. On average the collected data is available for analyzing and reporting in 15 – 30 minutes, depending on the amount of data. The advantage of this way of working is that the data is transformed into a data structure created to fit analyzing and reporting, but the disadvantage is that BPI is not real-time (it's almost real time).

The detailed data collected by the BPI product is aggregated, summarized, and stored over time.

3.1. BPI way of working

The BPI way of working is determined by its 4 architectural layers:

- **Data acquisition** - is done by a set of application programs which extract data from several data sources (FSC, PLC, DS, SAC, log-files), transform this data and load it into the data storage. In the current architecture this is done by using ETL (Extract, Transform, Load) layer.



- Data storage – the data coming from the data sources represents events, message data (fact) and reference data (dimension). The fact and dimension data is stored in the BPI data warehouse, specially designed to handle large amounts of data. The BPI data warehouse contains an Oracle database that holds the historical and archived fact and dimension data.
- Data delivery – is the process that moves fact and dimension data from the BPI data warehouse into BPI data marts. The data marts are subsets of the data warehouse designed to serve a specific purpose.
- Data presentation and analysis – the summarized and order data stored in the BPI data marts is reported and exported for further analysis.

3.2. BPI features

BPI is applied for all three market segments mentioned in earlier sections and targeted by VI and for life cycle management.

- **VIBES.BPI** – Baggage Handling
- **VITAL.BPI** – Parcel & Postal
- **VISION.BPI** – Distribution
- **LCM.BPI** – Life Cycle Management

VISION.BPI and the BPI Core (Figure 5) are developed by the VI product development team from VI LSG in Dortmund, Germany. The picture below shows the top level architecture of the BPI core and its appliances.

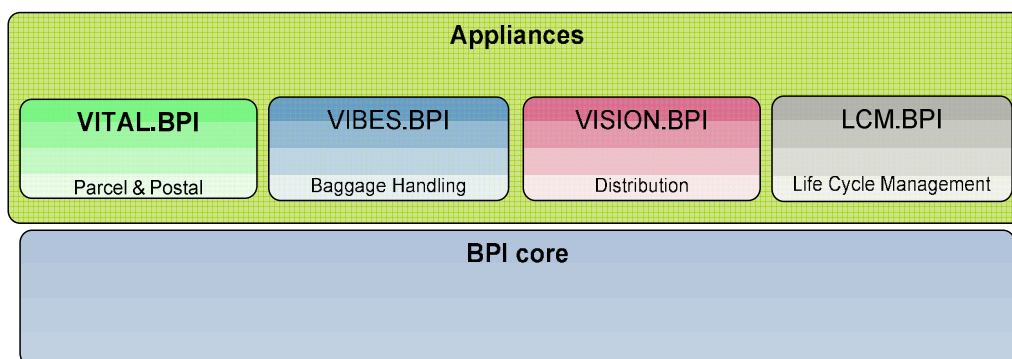


Figure 5 – BPI core and BPI Appliances

Some of the BPI features are:

- ETL - Extract, transform and load fact and dimension data from several VI and external data sources and log-files into a data warehouse (BPI data warehouse).
- Store data for a long period of time (history).
- Backup and recovery of data (archiving).
- The appliances listed above which can be used off the shelf or as a starting point (template) for project specific development.
- Example of basic reports: parcel overviews, sorting overviews, equipment reports, error overviews, running hours overviews, capacity reports, maintainability reports, response times reports.
- Create user-defined and project specific reports on top of historical data.
- Support of dynamic analysis and trend analysis of historical data.



➤ Schedule a report and send the result automatically to a user or group of users.
The starting point of this investigation is the VITAL.BPI appliance hence the next section will present a few details about it.

3.3. VITAL.BPI

This BPI appliance was developed to fit the requirements coming from the P&P market segment. It has an important role in gathering, optimizing, storing and reporting the data coming from sorting systems.

The KPI's reported by VITAL provide information about the process related performance as well as equipment performance. A few of this KPI's are listed below.

- Number of parcels that enter the system;
- Number of successfully sorted parcels;
- Number of lost parcels;
- Number of parcels with errors;
- Number of failure and disturbances in the system;
- Number of recirculated parcels;



Figure 6 – VITAL.BPI



3.4. QlikView

The KPI's are presented to the users as charts and graphs. All BPI appliances use as a reporting /analyzing tool the solution offered by the BPI core product which is using QlikView. QlikView is a BI reporting and analysis tool which provides fast, powerful and visual business analysis and offers numerous possibilities of creating dashboards

By creating dynamic reports QlikView gives the possibility to change parameters while a report is still running, hence focusing more on the analysis part and not just simple reporting.

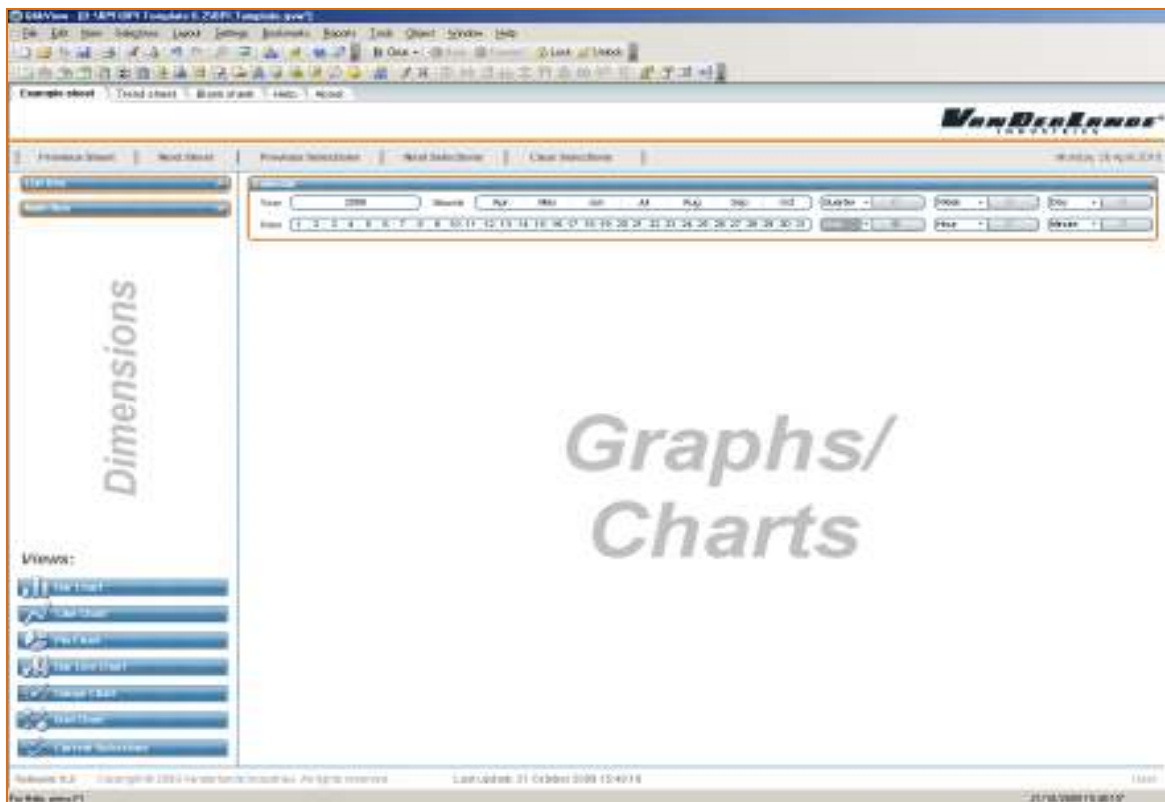


Figure 7 – QlikView main screen



4

Assignment Overview

4.1. Problem definition

The cost for the VITAL.BPI appliance, which is around 30000 €, is considered to be high by some of the VI customers, in the P&P market segment. Some of these customers are small businesses and don't want to pay for all the functionalities that VITAL offers. The P&P stakeholders find it necessary to have a low cost version of this product, which should offer a minimum set of functionalities and reports. An investigation is needed in order to identify what are the possibilities to reduce the costs of VITAL.BPI and implement a low cost product (LOWCOST.BPI).

4.2. Assignment Description and Objectives

The BPI product team defined a graduation assignment based on the problem defined in section 4.1. The graduation assignment is to investigate the options to further reduce the cost of VITAL.BPI and implement a proof of concept for the LOWCOST.BPI product. The focus of this assignment is on the following activities:

- Define a minimum set of requirements for the LOWCOST.BPI;
- Define a minimum set of KPI's (reports) that will be implemented;
- Select one or more new technologies to use for reporting (reporting tools), taking into account the requirements discussed with the P&P stakeholders and the goal to further reduce the costs of VITAL.BPI.
- Implement a proof of concept based upon the defined requirements, KPIs and chosen reporting technology.

The final deliverables of this project should consist out of a final list of requirements, a cost breakdown for the LOWCOST.BPI and the ranking of the selected reporting tools, based on the defined requirements. These documents should provide the essential information to help the BPI product development team in next phases of the LOWCOST.BPI project.

4.3. Project Phases

In order to make sure that the project is successful, a thorough planning has to be created. This planning divides the project into phases. The sequence and a short description of the project phases are presented below.



4.3.1. Planning – Create project plan

The planning phase is the most important phase in managing a project. In this case it should start with defining a list of project activities which will have to be organized into phases, and should continue with creating the project deliverable list, a detailed planning and the project plan document.

4.3.2. Project management – Write graduation report

This phase will handle activities related with managing the project: update documents (project plan, planning, and deliverables list), create and update the graduation report, training sessions. Special hours are planned for every week in order to undertake the activities mentioned above.

4.3.3. Start-up phase – Requirements Analysis and Reporting tool Investigation

During the start-up phase the preparation for the requirements phase will take place.

A first step in creating a proof of concept for the LOWCOST.BPI will be to gather all the requirements and other information necessary to start investigating the possibility of implementing a LOWCOST.BPI.

The activities planned for this phase are:

- Gather requirements through meetings with the involved stakeholders;
- Investigate the options for the new reporting tool;
 - Make a first selection and create a long list;
 - Create short list;

4.3.4. Iteration 1 – KPI's Requirements, Design, Implementation

The activities related with requirements, design and implementation will be scheduled in three iterations. The purpose of the first iteration is to test how a report can be implemented with the new reporting tool(s). For this purpose, two already existing reports (reports that were created for VITAL.BPI) will be used. Following the same requirements, design and implementation method from VITAL these two reports will be developed in the LOWCOST environment.

4.3.5. Iteration 2 – KPI's Requirements, Design, Implementation

During the second iteration it is planned to implement a new report in order to measure the engineering hours needed to develop it. The number of the engineering hours, to install the reporting tool, configure and develop reports is very important for the cost breakdown and implementation of the LOWCOST.BPI.

4.3.6. Iteration 3 – KPI's Requirements, Design, Implementation

The third iteration is scheduled for implementing a more complex report, or to be used as extra time in case there is not sufficient time scheduled for the first two iterations.



5

Project Approach

As mentioned in the previous chapter this project was divided into seven phases. Each phase had its own activities and deliverables. These deliverables were important for determining when the phase can be completed and when more time is needed. The next section presents the approach that was chosen for each phase.

5.1. Approach of the planning phase

The objective of the planning phase was to create the project plan, deliverable list and planning. These three documents represented an important step in managing the project. After identifying all the deliverables and the activities required for producing them, the project was organized into the following phases:

- Planning phase;
- Project management;
- Project start-up;
- Requirements analysis;
- Iteration 1;
- Iteration 2;
- Iteration 3;
- Evaluation;

Each of these phases will be described in more details in the next sections.

The purpose of the project plan was to serve as a guide. Besides description of phases and deliverables it also describes the graduation assignment in details, contains information about project risks, constraints and management plan,

In order to have a good overview of the project activities, two planning documents were created. The first planning is more general concerning the schedule of each phase. The second one is a weekly planning and it describes the activities per each week, including number of hours worked. The two documents were updated weekly in order to be able to follow the project status and identify the current situation when needed.

5.2. Approach of the project management phase

Beside the deliverables directly involved with solving the assignment, some documents required by the university had to be created. One of them and the most important was this graduation thesis. The management phase was created to keep track of the activity of writing this document as well as creating a communication plan, and taking part in training activities that helped in gaining knowledge about the BPI product and VI software development processes.



5.3. Approach of the start-up phase

Two of the most important tasks of this graduation assignment were to define the requirements for the Lowcost.BPI and investigate what are the reporting tool alternatives.

The approach for the requirements analysis part of this phase was to interview the stakeholders and together with them define the requirements. An important thing was to have these requirements prioritized. To do that the MoSCoW chart method was used. More details about the approach used for gathering the requirements are given in chapter 6.

The second part of this phase was about finding a suitable reporting tool that could be used in the development of the Lowcost.BPI. Knowing the subject and purpose of this investigation it was easy to create a planning and define the steps that had to be followed in order to get to a final result. A description of all this steps is given in chapter 7.

5.4. Approach of the first iteration

The next step after defining the requirements and selecting the reporting tool was to implement the reports. The implementation phase was divided into three iterations. The purpose of each iteration was to develop one or two KPI's in order to test the functionalities of the tool and measure engineering hours.

The KPI's for the first iteration were chosen together with the stakeholder for the P&P market segment. The implementation of the KPI's started with writing the functional and non-functional requirements, continued with design and finished with the actual coding and creating reports. The details of the implementation are described more thorough in chapter 8.

5.5. Approach of the second iteration

The purpose of the second iteration was to implement one or two new defined KPI's. If in the first iteration the objective was more related with the functionalities of the reporting tools, this time the focus was on measuring the number of hours used to implement one report, starting from requirements till deployment.

The implementation part of the first iteration required more time than expected in order to finish the KPI's and present them to the stakeholders. Hence the activities scheduled for the second iteration were delayed.

Another activity that had to be re-planned was to alignment of the requirements. This required preparing a presentation for the stakeholders which took about one week. The high priority of this two activities resulted in postponing the implementation of the new KPI's till the third iteration.

The most challenging part of this iteration was to deal with the continuous change of opinions from the stakeholders regarding the Lowcost.BPI requirements.

5.6. Approach of the third iteration

The third iteration was used to finish and improve the KPI's implemented in the first two iterations, and to implement a new one. This phase was also used to update requirements and design documents, and continue the investigation of the reporting tools.



5.7. Approach of the Evaluation

During the evaluation phase the results of this graduation project were produced. This resulted in three very important documents:

- Aligned requirements list;
- Ranking of the reporting tools based on functionalities and requirements.
- LOWCOST.BPI costs breakdown;

Also during this phase all the documents that had to be delivered to the company have been prepared and archived.



6

Requirements Analysis

Requirements analysis was a critical factor in the success of this project encompassing the tasks that helped to determine the conditions to meet for the Lowcost.BPI and taking account of the possibly conflicting requirements of the stakeholders. The main objectives during this phase were to gather requirements, determine whether they are unclear, incomplete, ambiguous or contradictory and solve these issues. To define the requirements it was necessary to identify all the stakeholders, consider their needs and ensure they understand the scope of this graduation project.

The investigation for the Lowcost.BPI was requested by the VITAL stakeholders. Hence interviewing them was an important step in creating the requirements list. First meeting was with Mr. Bas Oversier, group leader in the engineering department at VI and representative of the P&P customers, and it had more of an introduction purpose. The goal established for this meeting was to make a start with the Lowcost.BPI project, make a step forward in defining the minimal set of KPI's (reports) and define the requirements for the user-interface of the reporting tool (for example: web-based, navigation options, minimal license costs, ease of use). It was essentially to define the basic reporting requirements as soon as possible, in order to start the investigation for the reporting tool.

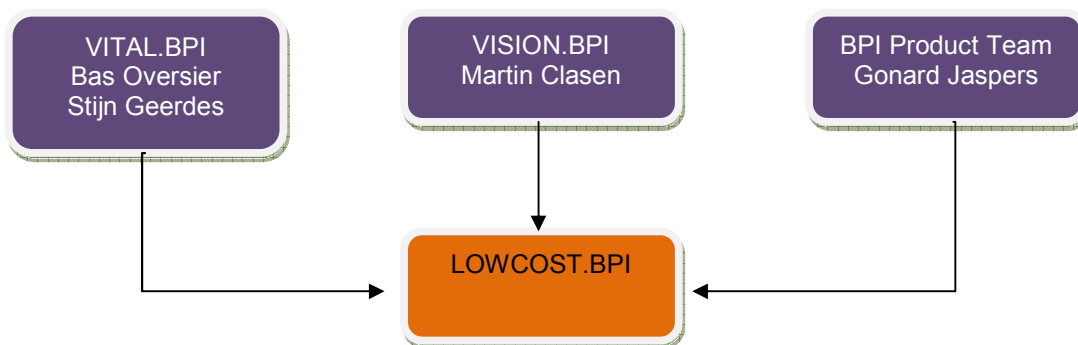


Figure 8 - Lowcost.BPI stakeholders

Following the initial one, additional meetings with VITAL stakeholders were planned throughout the duration of this project. The results were each time summarized in a Minutes-Of-Meeting document. Even though initially the investigation was focusing on VITAL.BPI, the stakeholder of VISION.BPI, Mr. Martin Clasen, also showed his interest for a Lowcost.BPI appliance. Hence it was very important to meet him and see what his exact demands are. As a representative of VI customers for the Distribution market segment he had a very strong opinion regarding how a Lowcost.BPI should look like. The meetings with Mr. Clasen took place at VI LSG in Dortmund. During these meeting the requirements for low cost version of VISION.BPI were defined and discussed.



The next step in defining the requirements was to make them SMART (Specific-Measurable-Attainable-Realizable-Traceable). Therefore it was necessary to schedule additional interviews with both VITAL and VISION stakeholders. During these meetings there were a few problems in making the stakeholders understand what the exact purpose of this project is. It was important for them to know that at the end they will not receive a product which they can sell, but just a proof that it can be implemented, a list with clear defined requirements, and an overview of the reporting tools that can be used for developing the product.

The gathering process became more complicated once a decision was made to align the requirements from both VITAL and VISION stakeholders, and make the LOWCOST.BPI suitable for both BPI appliances. At this moment in the project it became more important to have a list of requirements that will satisfy all the stakeholders than implement a large number of reports. For this purpose a requirements alignment meeting was scheduled. The meeting was started with a PowerPoint presentation which outlined the requirements defined at that point and the differences of opinion between VITAL and VISION stakeholders. The goal of this meeting was achieved once the stakeholders agreed on the requirements. Another positive output of this meeting was that they were interested in going further with the LOWCOST.BPI and that they were already considering what will be the next step to take, once this graduation project will be over.

6.1. Requirements categories

The MoSCoW method was used for reaching a common understanding with the stakeholders of both VITAL and VISION regarding the importance they place on the delivery of each requirement. Based on this method the requirements were categorized in MUST have, SHOULD have, COULD have, WON'T have. The MoSCoW chart for all the requirements is presented in Chapter 9 – Results.

In software development the requirements are categorized as either functional or non-functional. The non-functional requirements of the Lowcost.BPI are about the BPI system itself and how well it will perform its functions.

Example of non-functional requirements

✎ Installation and configuration:

- Number of engineering hours to develop a KPI;

✎ Costs:

- Costs of engineering hours 1% Max of project costs;

✎ Reporting:

- Web-based;

✎ Hardware:

- Able to run on different hardware configurations;

✎ Performance:

- Generation of datamart configuration in 15 minutes;

✎ Data management:

- Data transfer from the production environment to BPI has to be speed-up to a few seconds/half of minute;



6.2. KPI's

The functional requirements are a description of what the Lowcost.BPI should be able to do and are defined per KPI. The KPI is a measure of the process related performance as well as the equipment performance and its role is to help customers improve their business processes. During the implementation phase the requirements document is updated with references to the appropriate dimensions and user presentation requirements for each implemented KPI.

In order to understand better what kind of reports a customer needs to have, one of the VITAL stakeholders, Mr. Stijn Geerdes organized an onsite visit to DPD Best (worldwide parcel Delivery Company). The operational manager of DPD Best indicated that the most important for him is to know when something goes wrong with the system and to have an overview on entrance and exit point's usage. Such reports can help him supervise the operational process.

Defined KPI's

👉 Identify non-efficient loading:

- Number of recirculated parcels;
- Number of parcels per infeed;
- Number of parcels prior to a certain time;

👉 Identify output usage:

- Compare the output for all chutes (which chutes are more used and which not);
- Number of parcels that missed their chutes;

👉 Missed capacity due to issues:

- Number of error parcels;
- Number of lost parcels;
- Barcode scanner performance;
- Weighing scale performance;
- Number of chutes full;
- Top 10 equipment issues;
- Length of stay of parcel in the system;

From this list only three KPI's were chosen for implementation:

- 👉 Number of parcels per Infeed;
- 👉 Number of parcels per chute;
- 👉 Length of stay of parcel in the system;

6.3. Conclusions and remarks

The most challenging part of the requirements phase was to deal with the stakeholders. Every meeting they had a different opinion about how the reports should look like and how should the data be presented. To solve this issues it was necessary to remind them the purpose of the project and that the functionalities of a low cost BPI cannot be the same with what a full-blown BPI has to offer. Gathering requirements was more complicated than expected, but it was a very interesting process that in the end finished successfully.



7

Research

This chapter describes the activities performed during the research part of the project. To make sure that the investigation process was conducted correctly the questions listed below were addressed. The answers for these questions are given in the next sections.

Research questions

- What is it investigated?
- Why is it investigated?
- What methods are used?
- What are the results?
- What are the conclusions?

As mentioned earlier in this thesis the aim of the research was to investigate the existing low cost/open source reporting tools and find which of them will fit the Lowcost.BPI. Because the main requirement for the Lowcost.BPI was to be low cost, there was not an option to use QlikView, which license costs were around 7,000 €, therefore a new reporting tool was needed. To find the most suitable one, the following steps were applied.

Research steps

- Make research plan:
 - Create planning;
 - Identify sub-deliverables: long list, short list, research log-book, ranking of the reporting tools based on the requirements;
- Interview stakeholders in order to gather all the information necessary for the research;
- Identify the Business Intelligence reporting tools available on the market;
- Study existing documentation (previous results of similar research);
- Create a log-book where all the information related with the reporting tools has to be stores: installation problems, functionalities, bugs, solutions to problems;
- Make a first selection by matching the price of the tools with the costs requirements (License costs must be < 1,000 €) – hence create the long list;
- Compare the functionalities of the remaining tools and based on that make the selection for the short list;
- Install the tools from the short list and test them further by creating reports;
- Rank the tools based on their performance;



7.1. Creating the long list

Based on the license costs requirement, which says that the cost of the reporting tool has to be smaller, than 1,000 €, a first list of tools was created:

- JasperReports – open source;
- BIRT(Business Intelligence Reporting Tool) – open source;
- Pentaho – open source;
- CrystalReports – license costs around 800 €;

From this list it results that most of the Business Intelligence tools existent on the market have a license price bigger than 1,000 €, an exception making CrystalReports, which is very close to this limit, and three open source reporting tools, JasperReports, BIRT and Pentaho.

Because VI had some bad experience with support from Business Objects, the company that produces CrystalReports, this was not an acceptable option for the stakeholders. Therefore was removed from the list. The remaining three tools were investigated further.

7.2. Creating the short list

JasperReports, BIRT and Pentaho have more or less the same features. To find their weak and strong points, and to identify the differences between them, a comparison of all their functionalities was done. A small example of the result is given in the following table and the detailed comparison can be found in Appendix D.

	BIRT	JasperReports	Pentaho
Website	www.birt-exchange.com/www.eclipse.org/birt	www.jasperforge.com	reporting.pentaho.com
License	Eclipse Public License	GNU Lesser General Public License	GNU Lesser General Public License
Report designer	BIRT Report Designer 2.6.0	Jasper iReports 4.0	Pentaho Report Designer 3.5.0 stable
Designer Platforms	Windows, Linux, Mac OS X	Windows, Linux, Mac OS X	Windows, Linux, Mac OS X
Standalone Java Client Available	Y	Y	Y
Report Format	XML (.RPTDESIGN files are pure XML)	.JRXML report design files are compiled into .JASPER report files, which are Java Byte Code. You then deploy the .JASPER file.	XML (.PRPT report file is a ZIP that contains the XML file plus other resources)
Sub-reports	Y	Y	Y
Tables	Y	N	N



Cross-tabs	Y	Y	“Experimental” in Pentaho 3.5
Hyperlinks within a report	Y	Y	Y
Cascading Style Sheets (CSS controlled format)	Y	N	N
Conditional Formatting	Y	Partial	Y
Multiple data sources and queries per report	Yes, plus support for joining them	Only via sub-reports (one data source per sub-report)	Only via sub-reports (one data source per sub-report) or in charts
Support for joining multiple data sources in the Designer	Y	N	N

Table 1 – Reporting tools features

Based on this list and the defined requirements, BIRT and JasperReports were chosen as the two best possible options for the Lowcost.BPI. The next two sections describe in details the functionalities of these tools.

7.1. JasperReports



JasperReports is a java based, open source reporting tool. It is able to use data coming from any kind of data source (including JDBC, log files) and produce reports that can be exported, printed, emailed in a large variety of document formats including PDF, Excel, Open Office, HTML, Word. The installation package of JasperReports comes together with JasperServer (the deployment environment for jasper reports) and iReports (the most known report designer for jasper reports). The investigation of JasperReports included the following activities:

- Test the reporting functionalities and make sure they coincide with the requirements of stakeholders.
- Test the design functionalities by investigating all the options of iReports;
- Test server functionalities by investigating how a report is deployed, scheduled and printed with JasperServer.

7.1.1. JasperServer

JasperServer is the open source server for JasperReports used for deploying reports. This server has a web GUI which makes it easier to run reports and provides the ability to manage them efficiently and securely, interact with reports including entering parameters and drilling on data, scheduling reports for distribution via email and storage in the repository. The repository can be accessed from the report designer and reports can be directly uploaded on the server.



Figure 9 - JasperServer – Login page

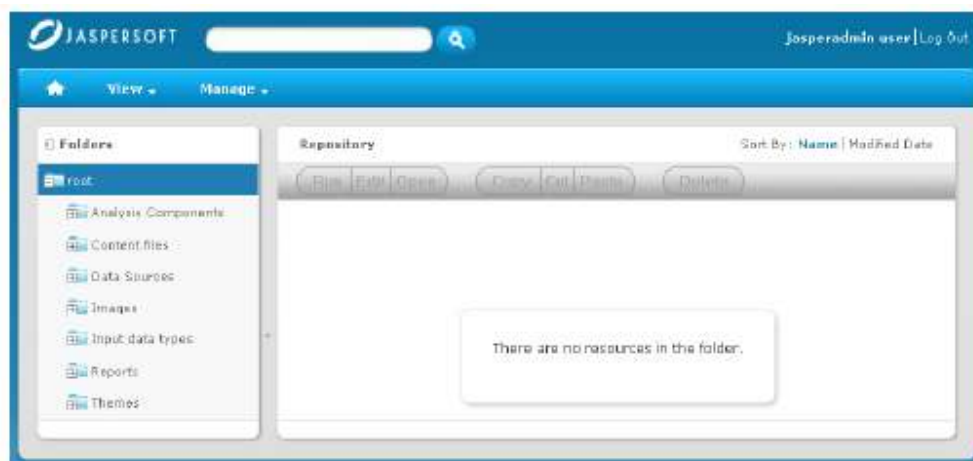


Figure 10 - JasperServer – Repository page

7.1.2. iReports

iReports is a free, open source report designer for JasperReports. With the help of this application the reports can have very sophisticated layouts containing charts, images, sub-reports, crosstabs and much more. The data can be accessed by creating new data-sources (JDBC, TableModels, JavaBeans, XML, Hibernate, CSV, and other custom sources). An important functionality of iReports is the direct connection with the repository from JasperServer. This way, once a report is created it can be uploaded from the designer to the server immediately.

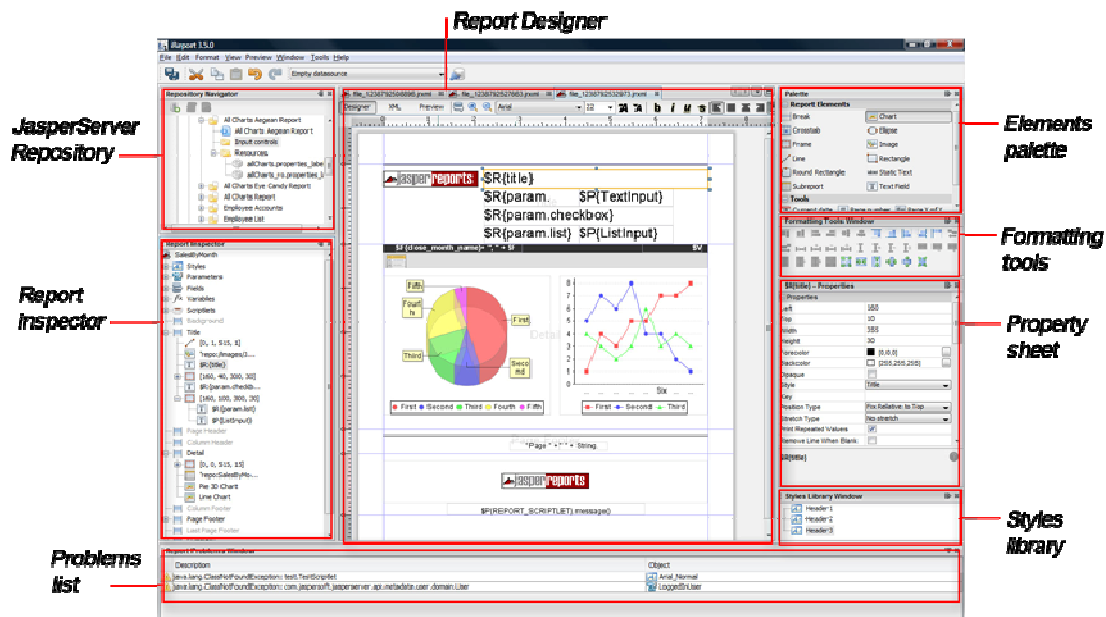


Figure 11 – iReports main window

7.2. BIRT



BIRT is the second tool selected during the research phase. It is an open source Eclipse-based reporting tool that integrates with Java applications to produce compiling reports. BIRT also comes with a report designer which is very easy to install and configure. The report designer is simple to use and offers a large variance of charts, supports tables, crosstabs and can read data from any data-source. One disadvantage of BIRT is that it doesn't offer a deployment environment for reports. Even though the BIRT web-viewer (an example of how to deploy the reports into web applications) is supplied, because it only offers the possibility to print and export reports, but not to schedule or email them, it is not sufficient to comply with the reporting requirements.

7.2.1. BIRT report designer

BIRT report designer is an eclipse integrated set of plug-ins that offers a variety of tools:

- Layout view – and editor that allows the user to design reports by using drag and drop (fields, charts, tables).
- Data explorer – Organizes database connections and is also used for designing queries and report parameters.
- Palette - Contains BIRT reports design elements such as labels, tables, charts.

Some other functions that BIRT designer provides are: reports preview, script editor, outline, chart builder and expression builder. Basically it offers the same functionalities for BIRT reports as iReports offers for jasper reports, having the advantage that it is easier to configure and use.

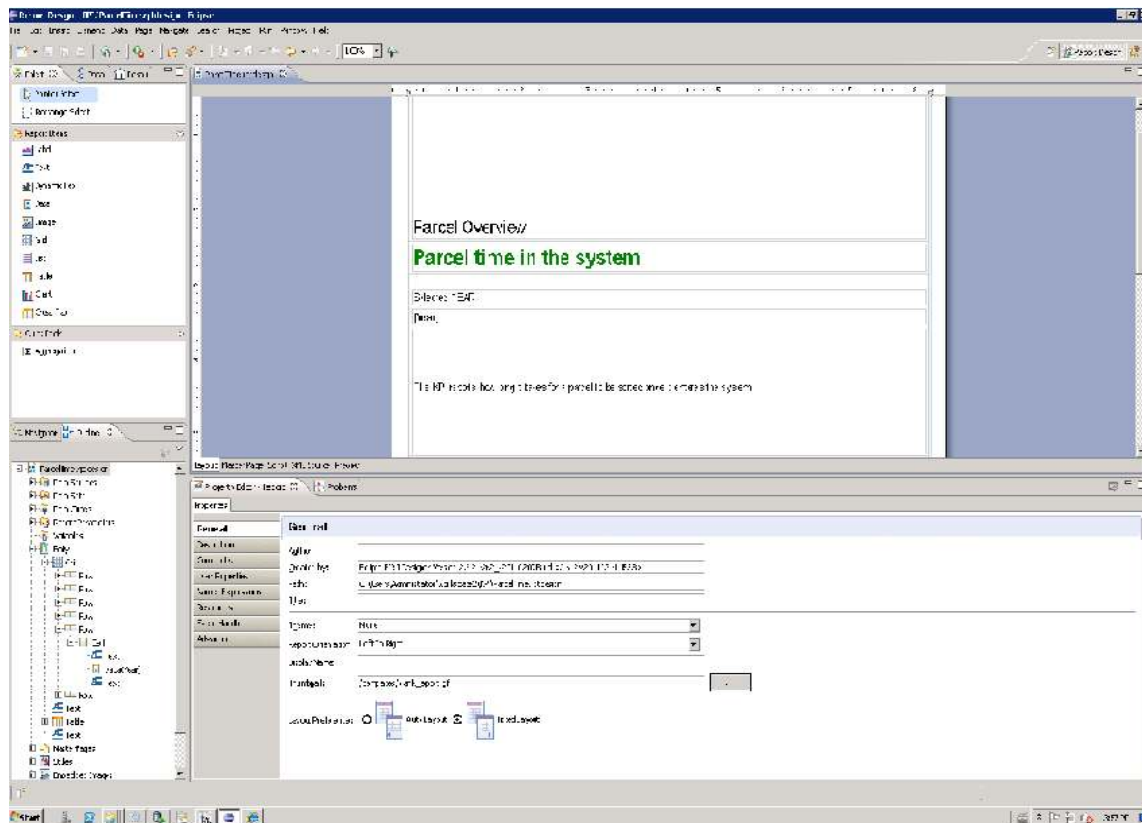


Figure 12 – BIRT report designer

7.2.2. BIRT deployment environment

BIRT designer did not come with an integrated deployment environment but with the BIRT web-viewer example which shows how a BIRT report can be deployed in a web application. This viewer offers the options to print and export reports in pdf format but does not offer the possibility to schedule and email reports. Looking for other options to use as a deployment environment, the OpenReports application was found. This application supports many types of reports, including JasperReports and BIRT. More details about OpenReports are given in the next section.

7.3. OpenReports

OpenReports is an open source web reporting tool that provides web based, report generation, printing, emailing and scheduling capabilities. Comparing with JasperServer it only provides minimum functionalities and it is much simple to use. A few features of OpenReports are:

- Supports PDF, HTML, CSV, XLS, RTF and Image export formats;
- Web based administration of Users, Groups, Reports, Parameters and Data-sources;
- Reports scheduling;
- Parameter supports;
- Support for drill down reports;
- Security control access to Reports, Scheduling and Administration;



One disadvantage is that the documentation coming with this application is not very useful and the installation steps were pretty hard to follow. But being that JasperServer also has a difficult installation process. Hence OpenReports can be taken into account as a good option for deploying both jasper and BIRT reports.

Figure 13 - OpenReports – Report administration window

7.4. Conclusions and remarks

In this chapter the research phase of the Lowcost.BPI is described, investigation steps are mentioned and the selected reporting tools are introduced. All the information gathered about the tools is stored in the research log-book. The results of this research represent a condition for the implementation phase which is described in the next chapter.



8

Implementation

The objective of this phase was to implement the reports defined during the requirements phase, as a proof of concept for the Lowcost.BPI. Another purpose was to measure the engineering hours necessary to create reports in order to calculate the costs per KPI.

The conditions for carrying this action successfully to the end were to have the requirements and KPI's defined, and the reporting tool(s) selected.

An overview of the structure of the implementation phase is given below.

👉 Purpose

- Investigate the functionalities of the reporting tools in more detail;
- Measure engineering hours needed for installation, configuration and implementation of reports;
- Implement the Lowcost.BPI proof of concept;

👉 Conditions

- Align requirements with all stakeholders;
- Define the set of reports that have to be implemented;
- The reporting tool is selected;

👉 Planning

- Iteration 1;
- Iteration 2;
- Iteration 3;

👉 Methods

- Requirements -> Design -> Implementation;
- Technologies: Oracle, PL/SQL, Java;
- Create Package -> Procedure -> Datamart ;
- Design report -> Deploy report -> Test report;

👉 Evaluation

- Check the results of the reports;
- Acceptance demo;

As mentioned earlier the implementation was done in three iterations.

The goal for the first iteration was to implement two KPI's, "Number of parcels per chute" (how many parcels go through one exit point of the system at one moment in time) and "Number of parcels per infeed" (how many parcels go through one entrance point of the system at one moment in time). For the second iteration a new KPI was defined: "Length of stay for a parcel in the system". During the third iteration, the implementation of the three KPI's above mentioned was finalized.

The first step in the implementation of each of the selected KPI's was to update the requirements and design documents. The requirements document contains information about the functional and



non-functional requirements of the LOWCOST.BPI and detailed description about the implemented reports. In the design document details about the design structures used in the development of the Lowcost.BPI are given.

Next step was to create the data mart table which would contain the data ready to be reported. To fill this data mart the package and procedure that extract the data from the warehouse core table, are created in Oracle SQL developer using PL/SQL.

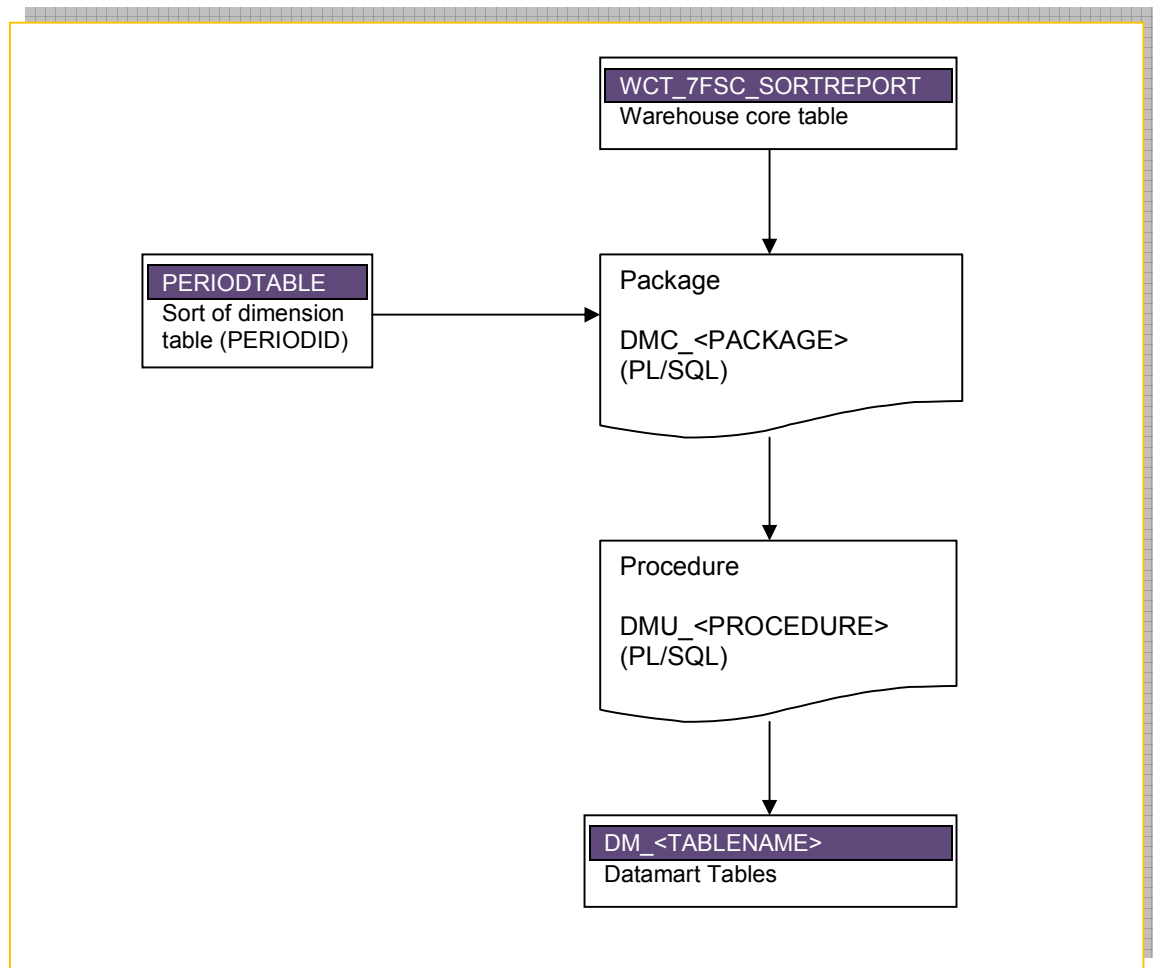


Figure 14 – Datamart

When a parcel leaves the system a Sort Report is generated. One Sort Report represents one parcel and it means that if one physical parcel for any reasons leaves the system and enters the tracking system again later, it will be counted as two parcels. The information from the Sort Report is stored in the Warehouse Core table of the BPI schema and contains the following fields. The fields in red were used in the calculations of the three reports implemented for the Lowcost.BPI.



Column name	PK	Format	Description
IDEVENT	X	NUMBER(19,0)	The primary key, a sequence number generated by the LOWCOST.BPI.
EVENTTS		TIMESTAMP(6)	The timestamp when the event occurred.
INSERTTS		TIMESTAMP(6)	The timestamp when the event is inserted to the warehouse core table.
<i>The rest of the fields are the attributes from the FSC_SortReport event.</i>			
ALIBI_ID		VARCHAR2(2000)	Parcel entrance time in the system.
BARCODE		VARCHAR2(2000)	Parcel scanner data 1 (from FSC parcel data structure).
BARCODE_2		VARCHAR2(2000)	Parcel scanner data 2 (from FSC parcel data structure).
BARCODE_3		VARCHAR2(2000)	Parcel scanner data 3 (from FSC parcel data structure).
BARCODE_4		VARCHAR2(2000)	Parcel scanner data 4 (from FSC parcel data structure).
BARCODE_5		VARCHAR2(2000)	Parcel scanner data 5 (from FSC parcel data structure).
CHECKSUM		VARCHAR2(2000)	Checksum (from FSC parcel data structure).
DTS		VARCHAR2(2000)	Destination Translate State (from FSC parcel data structure).
FSC		VARCHAR2(2000)	The FSC from where the LOWCOST.BPI received the event.
HEIGHT		VARCHAR2(2000)	Parcel height (mm) (from FSC parcel data structure).
HOST_DATA		VARCHAR2(2000)	Host data (from FSC parcel data structure).
HOST_PIC		VARCHAR2(2000)	Host PIC (from FSC parcel data structure).
LENGTH		VARCHAR2(2000)	Parcel length (mm) (from FSC parcel data structure).
MSG_ID		VARCHAR2(2000)	The Message ID of the FSC Sort Report.
NROFWNDWS		VARCHAR2(2000)	Number of windows (from FSC parcel data structure).
ODS		VARCHAR2(2000)	Original Destination State (from FSC parcel data structure).
PAD		VARCHAR2(2000)	Parcel Actual Destination (from FSC parcel data structure).
PALTD_1		VARCHAR2(2000)	Parcel Alternative Destination 1 (from FSC parcel data structure).
PALTD_2		VARCHAR2(2000)	Parcel Alternative Destination 2 (from FSC parcel data structure).
PALTD_3		VARCHAR2(2000)	Parcel Alternative Destination 3 (from FSC parcel data structure).
PALTD_4		VARCHAR2(2000)	Parcel Alternative Destination 4 (from FSC parcel data structure).
PALTD_5		VARCHAR2(2000)	Parcel Alternative Destination 5 (from FSC parcel data structure).
PEP		VARCHAR2(2000)	Parcel Entrance Point (from FSC parcel data structure).
PES		VARCHAR2(2000)	Parcel Entrance State (from FSC parcel data structure).
PIC		VARCHAR2(2000)	Parcel Identification Code (from FSC parcel data structure).



POD		VARCHAR2(2000)	Parcel Original Destination (from FSC parcel data structure).
PPD		VARCHAR2(2000)	Parcel Physical Destination (from FSC parcel data structure).
PXP		VARCHAR2(2000)	Parcel Exit Point (from FSC parcel data structure).
PXS		VARCHAR2(2000)	Parcel Exit State (from FSC parcel data structure).
RECIRCULATIONS		VARCHAR2(2000)	Number of recirculations (from FSC parcel data structure).
SCANNER_ID_1		VARCHAR2(2000)	Parcel scanner id 1 (from FSC parcel data structure).
SCANNER_ID_2		VARCHAR2(2000)	Parcel scanner id 2 (from FSC parcel data structure).
SCANNER_ID_3		VARCHAR2(2000)	Parcel scanner id 3 (from FSC parcel data structure).
SCANNER_ID_4		VARCHAR2(2000)	Parcel scanner id 4 (from FSC parcel data structure).
SCANNER_ID_5		VARCHAR2(2000)	Parcel scanner id 5 (from FSC parcel data structure).
SDS		VARCHAR2(2000)	Scanner Data State 1 (from FSC parcel data structure).
SDS_2		VARCHAR2(2000)	Scanner Data State 2 (from FSC parcel data structure).
SDS_3		VARCHAR2(2000)	Scanner Data State 3 (from FSC parcel data structure).
SDS_4		VARCHAR2(2000)	Scanner Data State 4 (from FSC parcel data structure).
SDS_5		VARCHAR2(2000)	Scanner Data State 5 (from FSC parcel data structure).
UPDATE_STATE		VARCHAR2(2000)	Update State (from FSC parcel data structure).
WEIGHT		VARCHAR2(2000)	Parcel weight (gram) (from FSC parcel data structure).
WEIGHT_ID		VARCHAR2(2000)	Parcel weight identification (from FSC parcel data structure).
WEIGHT_SCALE_ID		VARCHAR2(2000)	Parcel weighing scale identification (from FSC parcel data structure).
WIDTH		VARCHAR2(2000)	Parcel width (mm) (from FSC parcel data structure).
WNDWINFO		VARCHAR2(2000)	Windows info (from FSC parcel data structure).

Table 2 – Warehouse core table WCT_7FSC_SORTREPORT

Once the data mart is populated the report file has to be created. To create a report in both JasperReports and BIRT the following steps are followed:

- Create a database connection;
- Create a new report file;
- Add fields from the database report;
- Define report parameters;
- Deploy report on server;



8.1. Number of parcels per infeed

This KPI reports the number of parcels which enter the system through each entrance point. The calculation was done based on the PEP* field of the Sort Report. The requirements and design parts of the implementation lasted for one week, time in which the Business Rule for the KPI was defined, and the design of the data mart was created. The data mart contains the data ready for populating the report.

Bellow some details are given regarding the datamart created for this report. For more information the requirements and design documents are attached at the end of this thesis.

- Datamart : DM_LOWCOST_INFEED
- Procedure : DMU_LOWCOST_INFEED
- Package: DMC_LOWCOST_INFEED

Column Name	PK	Format	Description	Formula
IDEVENT	X	NUMBER(10,0)	The primary key of DM_LOWCOST_INFEED	The IDEVENT of the WCT_7FSC_SORTREPORT.
INSERTTS		TIMESTAMP	The timestamp where the event is inserted into WCT_7FSC_EVENT.	The INSERTTS of the WCT_7FSC_SORTREPORT.
EVENTTS		TIMESTAMP	The timestamp when the parcel is sorted.	The EVENTTS of the WC_2FSC_RECIRCREPORT.
INFEED_ID		NUMBER(10,0)	The ID from DIM_LAYOUT (based on the entrance point of the parcel)	The ID from DIM_LAYOUT based on the PEP of the WCY_7FSC_SORTREPORT with the type "INFEED".
PERIODID		NUMBER(20,0)	The ID from PERIODMINUTELY based on the day timestamp where the event occurred.	The ID from PERIODMINUTELY based on the EVENTTS of the WCT_7FSC_SORTREPORT.

Table 3 – DM_LOWCOST_INFEED fields

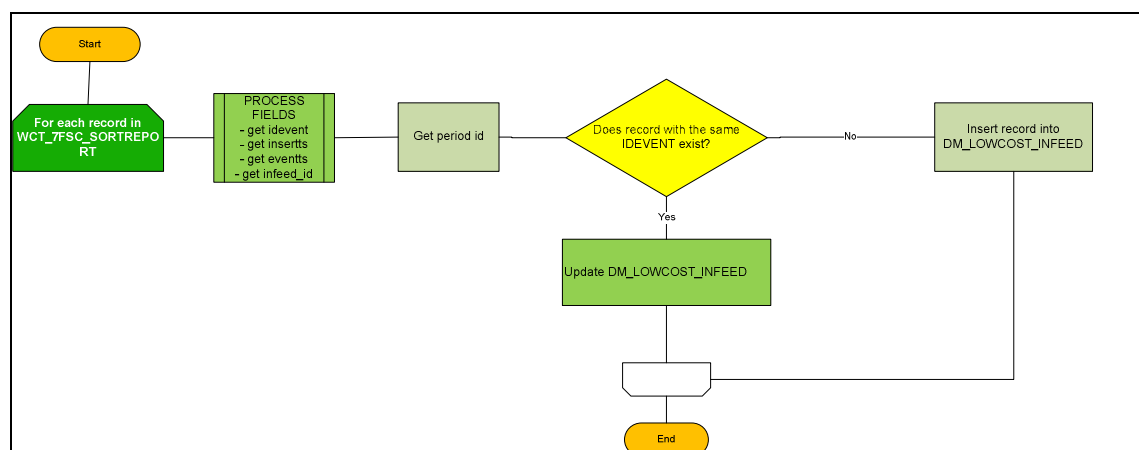


Figure 15 – DM_LOWCOST_INFEED flowchart

Following the requirements, design and implementation of the package and procedure was the creation of the report using iReports, the report designer from JasperReports. The first step in creating the report was to connect from iReports to this BPI schema and gain access to the



information stored in the DM_LOWCOST_INFEED data mart. After the connection was established the fields from the data mart were retrieved using an SQL query. The most challenging part of implementing this first KPI was to design and run the report with iReports and JasperServer. The lack of experience and the difficult way in which iReports is organized made it difficult to understand how everything is working. The issues discovered during each step of the implementation were recorded into the log-book together with the solutions found.

8.2. Number of parcels per chute

This KPI reports the number of parcels which are successfully sorted for a specific chute (exit point). The dimensions supported by this KPI are the layout dimension – chute and the time dimension day. An example of how this report can be formulated is: Calculate the number of parcels per chute *from start date – to end date*, where start date and end date are parameters selected by the user.

The calculation for this KPI was done based on the chute_id field of the Sort Report. The data mart created for this report was DM_LOWCOST_CHUTE and it contains the following fields.

Column Name	PK	Format	Description	Formula
IDEVENT	X	NUMBER(10,0)	The primary key of DM_LOWCOST_CHUTE	The IDEVENT of the WCT_7FSC_SORTREPORT.
INSERTTS		TIMESTAMP	The timestamp where the event inserted into WCT_7FSC_EVENT.	The INSERTTS of the WCT_7FSC_SORTREPORT.
EVENTTS		TIMESTAMP	The timestamp when the parcel is sorted.	The EVENTTS of the WCT_7FSC_SORTREPORT.
ORIG_CHUTE_ID		NUMBER(10,0)	The ID from DIM_LAYOUT based on the original destination of the parcel.	The ID from DIM_LAYOUT based on the POD of the WCT_7FSC_SORTREPORT with the type 'CHUTE'.
CHUTE_ID		NUMBER(10,0)	The ID from DIM_LAYOUT based on the actual destination of the parcel.	The ID from DIM_LAYOUT based on the POD of the WCT_7FSC_SORTREPORT with the type 'CHUTE'.
PERIODID		NUMBER(20,0)	The ID from PERIODMINUTELY based on the day timestamp where the event occurred.	The ID from PERIODMINUTELY based on the EVENTTS of the WCT_7FSC_SORTREPORT.

Table 4 – DM_LOWCOST_CHUTE fields

The package and procedure created to populate this data mart are DMC_LOWCOST_CHUTE and DMU_LOWCOST_CHUTE. For the presentation layer the BIRT reporting tool was used. To create a report with the BIRT designer prove to be a very simple process. Even though the steps are almost the same as creating a report with iReports the development time was almost half.

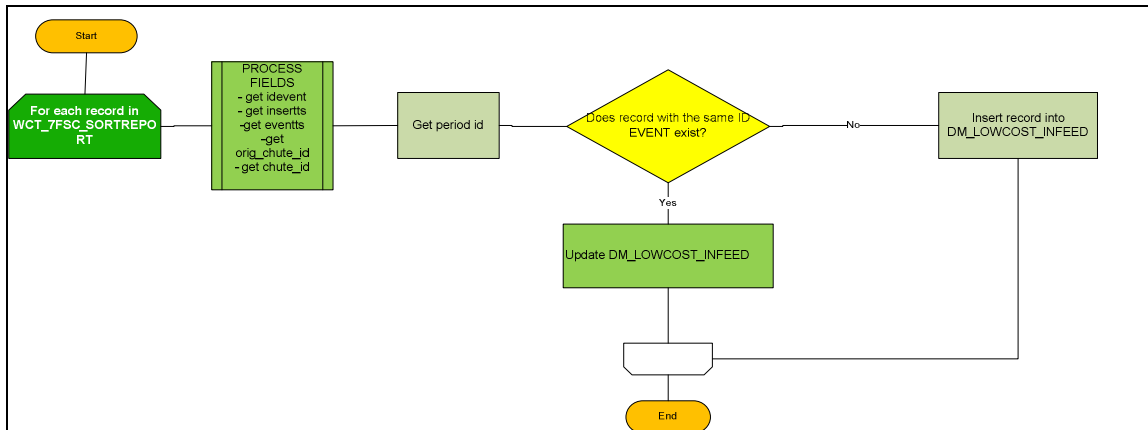


Figure 16 – DM_LOW_COST_CHUTE flowchart

8.3. Length of stay of a parcel in the system

This KPI reports how long a parcel stays in the system and is calculated on daily time dimension. In order to be able to make this calculation the EVENTTS and ALIBI_ID fields of the Sort Report are used.

The data mart used to create this report was named DM_LOW_COST_PARCEL and the package and procedure that populate this data mart are named DMC_LOW_COST_PARCEL and DMU_LOW_COST_PARCEL.

The presentation part of this KPI was done in both JasperReports and BIRT. This offered the possibility to compare the output of the two files and see which one is more close to what the stakeholders want.

Column Name	PK	Format	Description	Formula
IDEVENT	X	NUMBER(10,0)	The primary key of DM_LOW_COST_INFEED	The IDEVENT of the WCT_7FSC_SORTREPORT.
INSERTTS		TIMESTAMP	The timestamp where the event inserted into WCT_7FSC_EVENT.	The INSERTTS of the WCT_7FSC_SORTREPORT.
EVENTTS		TIMESTAMP	The timestamp when the parcel is sorted.	The EVENTTS of the WC_2FSC_RECIRCREPORT.
ALIBI_ID		STRING	The ID from DIM_LAYOUT (based on the entrance point of the parcel)	The ID from DIM_LAYOUT based on the PEP of the WCT_7FSC_SORTREPORT with the type "INFEED".
PERIODID		NUMBER(20,0)	The ID from PERIODMINUTELY based on the day	The ID from PERIODMINUTELY based on the EVENTTS of the WCT_7FSC_SORTREPORT.
PARCELT		INTERVAL	The length of stay of parcel in the system based on entrance and exit points of parcels.	

Table 5 – DM_LOW_COST_PARCEL fields

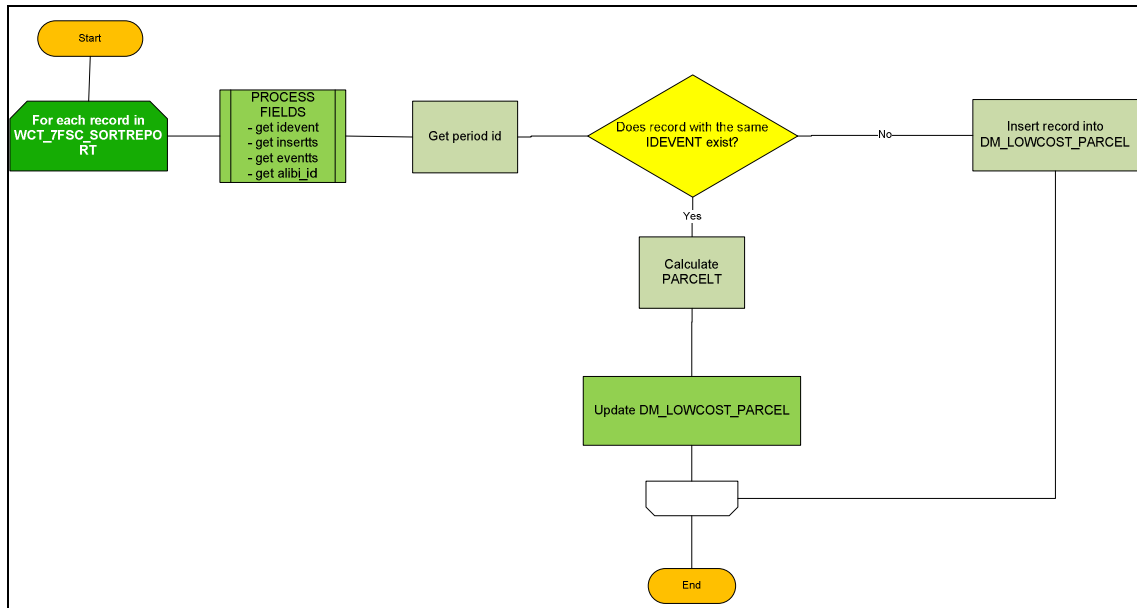


Figure 17 – DM_LOW COST_PARCEL flowchart

8.4. Conclusions and remarks

This chapter describes the implementation of the three KPI's defined together with the stakeholders. Details about the structure of the implementation phase, the requirements and design of the KPI's are given. The most challenging part of this phase was to deal with the reporting tools issues and identify all their functionalities in order to create reports that will satisfy the stakeholders. The next chapter describes the deliverables of this graduation project which are going to be used as a base for the implementation of the Lowcost.BPI product.



9

Results

The purpose of this graduation project was to define the requirements for the Lowcost.BPI, make a reporting tool selection and implement a proof of concept. The defined deliverables were the aligned requirements list, the ranking of the reporting tools and the costs breakdown for the Lowcost.BPI. These three documents are presented in the next sections.

9.1. Requirements list

The requirements were defined together with VITAL and VISION stakeholders. As it can be seen below the MoSCoW chart was used to prioritize them. This requirements list was an input in the reporting tool investigation and the implementation phase.

Installation and configuration	MUST	SHOULD	COULD	WONT
Engineering hours (to develop an average KPI) - max 24 hours				
Duration of configuration – 4h (maximum one working day)				
Installation and configuration on-site installing a pre-installed/configured BPI can be done by the IT department or LLC (FSC) engineering within the standard setup of the production system.				
<i>Pre-requisite: Data sources are ready to provide required data.</i>				
Costs				
Costs of engineering hours 1% MAX (10k) of project costs				
License cost of maximum 1k (if applicable no costs for the reporting tool)				
No maintenance costs				
Reduce hardware costs (to be handled together with the IT department)				
If the BPI is not sold to a customer the LOWCOST becomes part of the standard VISION (VITAL) environment to:				
- support service activities, support commissioning, provide real time statistics;				
The additional price has to be smaller than 5k including BPI commissioning, on-site configuration, in-house installation &				



configuration				
Re-configuration of service environment must be standardized and part of BPI pricing				
Switch from Lowcost.BPI to sold BPI				
User Management/Access Rights have to be taken over automatically.				
Statistics environment moves from LOWCOST.BPI to sold BPI with QlikView				
Data transfer from Lowcost.BPI to sold BPI must be standardized and part of the BPI pricing				
Reporting				
Web-based				
Multi-users (5 users simultaneous)				
User access rights and user right management				
Internationalization (language support)				
Single-sign-on				
Reports can be adapted by the end user				
Printing				
Navigation - dropdown menu				
Basic report design (Start page, Header, Body, Footer, End page)				
Export to pdf				
Export to excel				
Dynamic reports				
Schedule reports				
Email scheduled reports to group of users				
Integration with web applications (BIRT is already integrated in WEB-SVS)				
User interface should be simple and have the same look and feel with SCADA or DS				
Automatic refresh				
Minimum time frame between 5 – 15 minutes				
Reports can be run based on shift selection				
Reports can be run based on start date - end date selection				
Time resolution				
Graph type (at least bar chart, pie chart, line chart)				
Arithmetic operations				
Comparison of multiple graphs (up to four graphs in one view) that can be freely positioned.				
Reports can be grouped.				



When the report is loaded it should first show the graph. A click on the graph should show the grid view with all details related to the graph.				
If the user wants to see the graphical representation and the grid representation of a KPI he has to open the report twice.				
Alerting (depends on scheduling)				
User can define alert levels (based on the alert level he can define actions)				
Hardware				
It should be able to run on different hardware configurations				
Should run in the environment of the VISION server solution, e.g. on the FCA server - additional accepted costs: max 1000 PKP) larger HD, larger RAM)				
Should have an external network access point that is independent on the customer network but: A person can access the system with a laptop that has an installed QlikView. QVW-File can be created without having direct access on the system and without using the customer network.				
Performance requirements				
Fast loading time of reports (normal intranet speed)				
Fast switching time between reports				
Generation of data mart - configuration in 15 minutes				
Data management				
Time to store data:				
3 months only for detailed data required by service				
15 years for LCM data (daily base - min. Resolution)				
Max # GB to store: 800				
Data transfer from the production environment to the BPI has to be speed-up to a few seconds/half of minute.				
More than one BPI can be connected to one production environment				
Easy transfer of collected data from a small system to a large system (1/2 day)				
Configurable extrapolation of data to an hour or a day + presentation of the result and online updates.				
Hierarchical sampling/cascaded data.				
Data collection function should be the same with the one for the full-blown BPI				
A report is generated by configuring a data mart				



A report can have up to 20 dimensions/attributes that can be used to drill down				
---------------------------------------------------------------------------------	--	--	--	--

Table 6 – MoSCoW chart

9.2. Ranking of the reporting tools

The ranking of the reporting tool was done based on the investigation and implementation of the reports. During the investigation the functionalities of the reporting tools were discovered, and their weak and strong points were identified. The implementation of reports was another good way to find issues with the reporting tools and decide if they are still suitable with the requirements define for the Lowcost.BPI.

The first ranking was done based on the reporting tools features and the second one was done based on the defined reporting requirements.

	BIRT	JasperReports
Installation		
👉 Report Designer	++	+
👉 Server (deployment environment)	-	-
Configuration		
👉 Installing driver for datasource	+	+
👉 Connection to the datasource	+	+
Creating reports		
👉 SQL Query	+	+
👉 Design report	+	-
👉 Run report	+	+
👉 Deploy report on server	+	++
👉 Create reports with parameters	+	+
👉 Create reports with sub-reports	-	+
👉 Create templates	+	+
👉 Chart support	++	-
👉 Create reports with charts	++	-
👉 Tables	+	-
👉 Cross-Tabs	+	-
👉 Response time (designer)	++	+
👉 Response time when running a report	-	+

Table 7 – Reporting tool ranking based on features

Reporting tools – ranking (based on requirements)

	BIRT	JasperReports
Web-based	+	+
Multi-users (5 users simultaneous)	+	+



User access rights and user right management	+	++
Internationalization (language support)	+	+
Single-sign-on	-	-
Reports can be adapted by the end user	-	-
Printing	+	+
Navigation - dropdown menu	+	+
Basic report design (Start page, Header, Body, Footer, End page)	-	+
Export to pdf	+	+
Export to excel	+	+
Dynamic reports	-	-
Schedule reports	+	++
Email scheduled reports to group of users	+	++
Integration with web applications (BIRT is already integrated in WEB-SVS)	++	+
User interface should be simple and have the same look and feel with SCADA or DS	++	+
Automatic refresh	-	-
Minimum time frame between 5 – 15 minutes	-	-
Reports can be run based on shift selection	+	+
Reports can be run based on start date - end date selection	+	++
Time resolution	+	+
Graph types (at least bar chart, pie chart, line chart)	++	+
Arithmetic operations	+	+
Comparison of multiple graphs (up to four graphs in one view) that can be freely positioned.	-	-
Reports can be grouped.	+	++
When the report is loaded it should first show the graph. A click on the graph should show the grid view with all details related to the graph.	-	+
If the user wants to see the graphical representation and the grid representation of a KPI he has to open the report twice.	-	+
Alerting (depends on scheduling)	-	-
User can define alert levels (based on the alert level he can define actions)	-	-

Table 8 – Reporting tool ranking based on requirements



9.3. Costs breakdown

The costs breakdown it's the most important deliverable of this project, and is based on the reporting tools costs and functionalities, on the engineering hours and hardware costs. The rate for one engineering hour is 98 €. The number of hours spent on the implementation of one KPI depends on its complexity. The following overview is done based on the implementation of the KPI "Length of stay for a parcel in the system".

9.3.1. Engineering effort

Activity	Actual Hours	
	BIRT	JasperReports
Installation and configuration	12h	16h
Installation of the report designer	1h	1h
Installation of the deployment server	3h	3h
Configuration of the reporting tool (make connection to the database, add data source driver)	2h	4h
Commissioning	2h	4h
Training	4h	4h
Implementation of a KPI	38h	40h
• Documentation	2h	2h
• Design report and data model	6h	6h
• Build package and procedure	24h	24h
• Deploy and test report	4h	4h
• Commissioning	2h	4h

Table 9 – Costs breakdown

9.3.2. License costs

Both BIRT and JasperReports are open-source reporting tools, so no price has to be paid in order to be able to use them. However if additional features are needed both of them offer the possibility to by a commercial version.

9.3.3. Hardware costs

The configuration of the BPI hardware has to be established together with the IT department. The available configurations and their prices are listed below.



	BPI Hardware	Costs	IT-hours
See IT hardware catalogue	Light configuration	€ 1.000	12
	Configuration A (basic server)	€ 3.500	28
	Configuration B (basic server + rack)	€ 6.500	28
	Configuration C (cold standby, 2 basic servers + rack)	€ 10.500	30
	Configuration D (warm standby)	€ 13.500	30
	Configuration E (hot standby)	€ 27.500	55
	Extra memory harddisk	€ 500	
	Extra memory RAM (4GB)	€ 175	

Table 10 – Hardware configuration

9.3.4. Costs comparison

In Table 11 an example of the cost differences between VITAL.BPI and LOWCOST.BPI is presented. This examples shows that based on hardware costs, software costs and engineering effort the difference is around € 10,000. As it can be seen the main difference is made by the reporting tool license price which dropped from almost € 7,000 to 0.

Costs comparison		
	VITAL.BPI	LOWCOST.BPI
Hardware costs	€ 6500	€ 3500
Software costs	€ 7490	€ 1010
• Microsoft Windows 2003 server 32/64	€535	€535
• Oracle License (9i or 10g)	€475	€475
• Reporting Tool License	€6480	€0
BPI core	€ 7448 (76h* € 98)	€ 7448 (76h* €98)
• Installation of BPI hardware and software	28h	28h
• Installation of BPI	8h	8h
• BPI configuration	8h	8h
• BPI testing	12h	12h
• Commissioning	20h	20h
Total hours	76h	76h
BPI reporting	€ 2548 (26h * € 98)	€ 1568 (16h * € 98)
• Installation	4h	4h
• Configuration	2h	2h
• Testing	8h	2h
• Commissioning	8h	4h
• Training	4h	4h
Total hours	26h	18h
TOTAL	€ 23986	€ 13526

Table 11 – Costs comparison



During VI projects sometimes is necessary to add new KPI's besides the ones that come by default with VITAL.BPI, VISION.BPI or VIBES.BPI. This can also be the case for LOWCOST.BPI in the future. The next example shows the costs differences between a VITAL KPI and a LOWCOST one.

Costs comparison – Extra KPI		
	VITAL.BPI	LOWCOST.BPI
Implement KPI		
• Documentation	4h	2h
• Design report and data model	8h	6h
• Build package and procedure	24h	24h
• Test report and ETL procedures	8h	4h
• Commissioning	2h	2h
• User manual	4h	2h
Total hours	50h	40h
TOTAL	€ 4900	€ 3920

Table 12 – Costs comparison - Extra KPI



10

Conclusions

The ultimate aim of the work described in this thesis is to create a base for the implementation of the LOWCOST.BPI product, more precisely to create a blueprint.

Requirements analysis was an essential part of this project. The most challenging aspect of this phase was to deal with the stakeholders. In the first few meetings they were changing their minds quite often about how the LOWCOST.BPI should look like and what kind of information should the reports provide. To solve these issues it was necessary to remind them that there is only a limited time for this graduation project and that the functionalities of a low cost BPI are limited. Even though gathering requirements was more complicated than expected, trying to deal with the stakeholders and align the requirements was a very interesting process.

As a result of the research phase two open source reporting tools were selected, JasperReports and BIRT. The reporting features offered by these two tools are more or less the same, having both strong and weak points. The conclusion reached is that both of them are strong candidates for being selected as the reporting tool for the LOWCOST.BPI, with BIRT having a slight advantage, because it has already been integrated with the web-interface of the BPI.

Trying to understand how the reporting tools are working and find the easiest way to create reports was the most difficult part of the implementation phase.

The costs breakdown shows the positive outcome of this investigation and proves that it is possible to have a low cost BPI appliance. As a result of this graduation project the LOWCOST.BPI gained a lot of attention. The obtained results should be used as an input for further discussions and the implementation of the LOWCOST.BPI product.



References

Documents

Gonard Jaspers, Product Application Note BPI, VI Software House, 2007
Gonard Jaspers, Product Architectural Outline BPI, VI Software House 2008
Gonard Jaspers, BPI Product Plan, VI Software House 2010
Georgiana Codreanu, LOWCOST.BPI Release Plan, VI Software House, 2011
Georgiana Codreanu, LOWCOST.BPI Component Requirements Specification, VI Software House, 2011
Georgiana Codreanu, LOWCOST.BPI Component Design Specification, VI Software House, 2011

Websites

www.birt-exchange.com/www.eclipse.org/birt
www.jasperforge.com

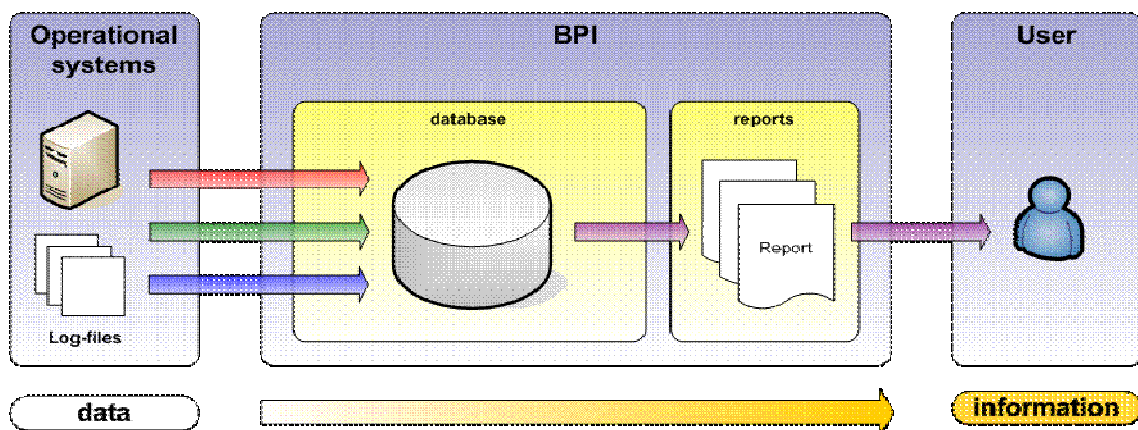


Appendix A – Original Assignment

Introduction

Within the Vanderlande Industries (VI) Software House the BPI product is developed. The BPI (Business Process Intelligence) product is a reporting/analysis server which makes diagnosis, analysis and reporting possible.

BPI is a category of application programs and technologies for gathering, storing, analyzing, and providing access to data to help users make better business decisions. This definition also applies to what is commonly known as Business Intelligence (BI). BPI includes the activities of querying and reporting, analytical processing and diagnosing.



The BPI product is applied in all three VI market segments; Parcel & Postal (VITAL.BPI), Baggage Handling (VIBES.BPI) and Distribution (VISION.BPI). In the Parcel & Postal market segment the costs of the BPI product (VITAL.BPI) are to be further reduced by decreasing the license costs and limiting the number of engineering hours.

Assignment

The graduation assignment is to investigate the options to further reduce the costs of the VITAL.BPI product and to specify/define the blue print for a low cost BPI product. The graduation assignment will focus on the following 3 areas:

- The definition/specification of the minimal set of KPI's (reports) and functionality to be delivered within the VITAL.BPI product in consultation/cooperation with our Parcel & Postal stakeholder(s).
- The selection of the technology to use for reporting (reporting tool) taking into account the requirements from stakeholder(s) and the goal to further reduce the (license) costs of the VITAL.BPI product.
- The implementation of a 'proof of concept' based upon the defined minimal set of KPI's (reports) and the chosen technology for reporting (reporting tool).

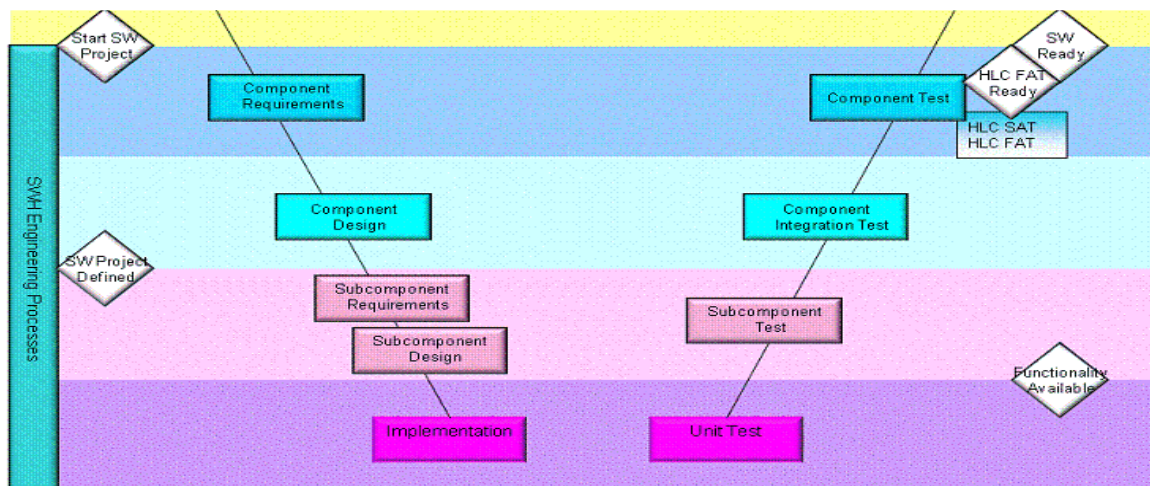


Research component

The research component consists of the selection of the technology to use for reporting (reporting tool). A reporting tool shall be selected/chosen which fulfills the requirements of our stakeholder(s); low cost (minimum to no license costs), web-based user-interface, KPI's.

METHODS

The graduation assignment shall follow the software development processes defined within the VI Software House; the V-model (see below).



The 'proof of concept' shall be based upon the current VITAL.BPI product development tools; Oracle database, PL-SQL, java, apart from the reporting tool which is to be selected.

The student will be guided by:

Gonard Jaspers

Project leader BPI (BPI product development)

Vanderlande Industries, Veghel, the Netherlands

+31-(0)413-495290

□ +31-(0)413-362910

□ gonard.jaspers@vanderlande.com

□ www.vanderlande.com

The student becomes (temporarily) member of the BPI product development team within the VI Software House.

The most important topics of the graduation assignment are; the definition/specification of the minimal set of KPI's (reports) for Parcel & Postal and the selection of the technology to use for reporting (reporting tool). Therefore the most important field of study is information analysis.



Appendix B – Project Plan (Fontys)

Graduation Project - LOWCOST.BPI

Project Plan

Author: Georgiana Codreanu

Supervisor: Gonard Jaspers

Version: 1.0

Date: 28.02.2011



Definitions and Abbreviations

Term	Description
BPI	Vanderlande Industries Business Process Intelligence.
FAT	Factory Acceptance Test
GL	Group Leader
HLC	High Level Controls
LSG	Logistic Software GmbH
N/A	Not Applicable
PR	Problem Report
PSR	Project Status Report
P&P	Parcel & Postal
SW	Software
SWH	Software House
VI	Vanderlande Industries
VIBES.BPI	Vanderlande Industries BPI product for the Baggage Handling market segment
VISION.BPI	Vanderlande Industries BPI product for the distribution market segment
VITAL.BPI	Vanderlande Industries BPI product for the Parcel & Postal market segment



Table of Contents

1. INTRODUCTION	55
2. PROJECT STATEMENT	55
2.1. FORMAL CLIENT	56
2.2. STUDENT	56
2.3. CURRENT SITUATION	56
2.4. PROJECT JUSTIFICATION	57
2.5. PROJECT PRODUCT	57
2.6. PROJECT DELIVERABLES AND NON-DELIVERABLES	57
2.7. PROJECT CONSTRAINTS	58
2.8. PROJECT RISKS	59
3. PROJECT PHASING	60
3.1. PLANNING PHASE	60
3.2. PROJECT MANAGEMENT	61
3.7. TESTING PHASE	64
3.8. EVALUATION PHASE	64
3.9. MONEY	65
3.10. SKILLS	65
3.11. QUALITY	65
3.12. INFORMATION	66
3.13. TIME	67
3.14. ORGANIZATION	68
REFERENCES	68
APPENDIX A - MEETINGS	69



1. Introduction

Vanderlande Industries (VI) is best known as a supplier and manufacturer of airport baggage handling systems, package sorting parcels for industrial and warehousing storage and order picking systems for logistics centers. In the field of baggage handling the company occupies the second place in the world (in 2007 there were around 400 airports using Vanderlande baggage handling systems). In terms of goods sorters and automated warehouses, the company is among the top five.

This document represents the project plan of my graduation project within Vanderlande Industries, Software House department (SWH) and it covers the project description, phases and planning followed by a brief management plan. This plan serves as a guidance for the company and university tutors during the graduation period and reflects my approach in accomplishing the assigned tasks and deliver the final result at the expected level.

The assignment of this graduation project concerns the Business Process Intelligence (BPI) product which is developed within VI SWH. This product is a diagnosis and performance management tool which makes analysis and reporting possible in order to help customers improve their business processes.

The BPI product is applied on all VI market segments. For the Parcel & Postal (P&P) market the product is called VITAL.BPI. VIBES.BPI is applied on the baggage handling segment and VISION.BPI is developed for the distribution part. VISION.BPI is implemented by the product development team from VI Germany. Another version of the BPI, LCM.BPI is developed to fit the needs of Life Cycle Management. The picture below shows the top level architecture of the BPI core and its appliances.

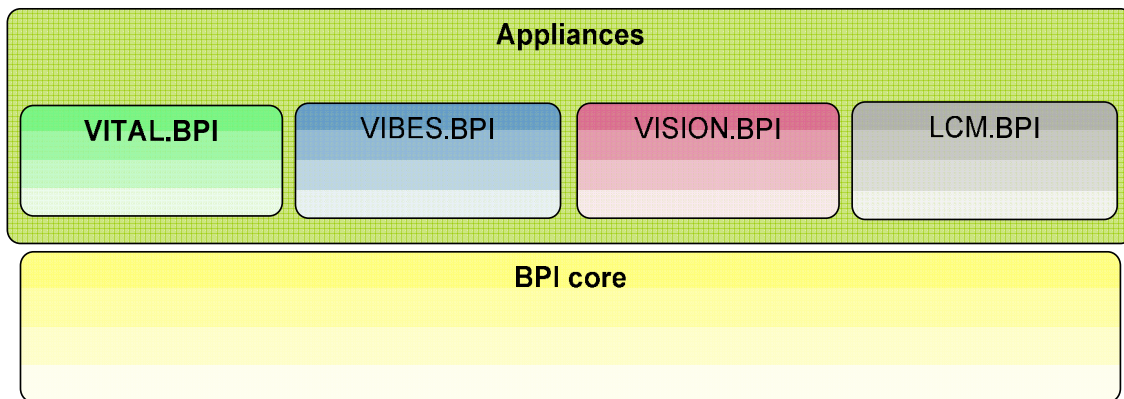


Figure 18 - BPI appliances overview

2. Project Statement

The subject of this graduation project is to make a feasibility study in order to demonstrate that it is possible to implement a low cost BPI on the P&P market segment. The new product will resemble VITAL.BPI but will have less functionalities (as mentioned before VITAL.BPI is the BPI appliance for Parcel & Postal).

This chapter introduces the different people involved in the project and gives detailed information about the current situation and the project justification. In later sections the project



product is defined followed by a list of deliverables and non-deliverables. The project constraints and the project risks can be found at the end of this chapter.

2.1. Formal Client

Gonard Jaspers

Company:	Vanderlande Industries
Department:	Software House - Product Development Team
Position:	BPI Project leader
Address:	Vanderlandelaan 2 5466 RB Veghel Building 10
Email:	gonard.jaspers@vanderlande.com
Phone number:	+31 (0)413-495290

2.2. Student

Georgiana Codreanu

University	Fontys University of Applied Sciences
Department:	Information and Communication Technology – English stream
Email:	g.codreanu@student.fontys.nl
Phone number:	+31 (0)648-721316

2.3. Current Situation

As mentioned earlier BPI is a category of application programs and technologies for gathering, storing, analyzing, and providing access to data in order to help users make better business decisions. BPI includes the activities of querying and reporting, analytical processing and diagnosing.

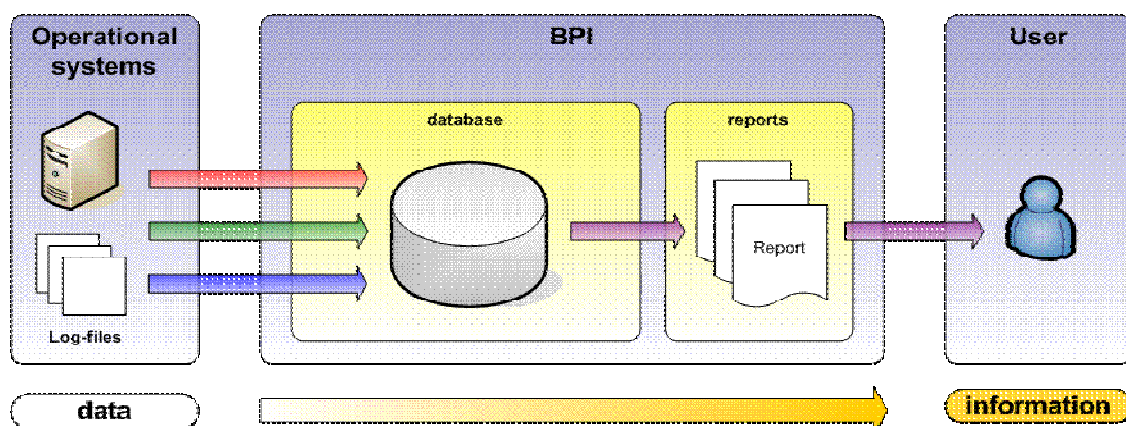


Figure 19 - BPI appliances overview



BPI's way of working is determined by its four architectural components:

- Data acquisition (ETL)
- Data storage
- Data delivery
- Data presentation and analysis

Some features of the BPI are listed below:

- Extract, transform and load fact and dimension data from several sources: low level controls, high level controls and log-files into a data warehouse.
- Store data for a long period of time.
- Backup and recover data.
- Multiple packages of basic reports: parcel overviews, sorting overviews, equipment reports, error overviews, availability reports, capacity reports, maintainability reports.
- Create user-defined and project specific reports on top of historical data.
- Support of dynamic analysis and trend analysis of historical data.
- Alert when the sampled/measured value of a variable exceeds a predefined minimum or maximum boundary.
- Schedule a report and send the result automatically to a user or group of users by email.

At this moment the BPI product for Parcel & Postal (VITAL.BPI) market segment provides all the functionalities listed above. VITAL.BPI offers a lot of reports and a dynamic user interface for reporting, using QlikView as the reporting technology. In order to please all their customers, the stakeholders of VITAL requested for a low cost version of the product. The cost of VITAL needs to be reduced by finding a way to decrease the license costs of the reporting tool, and limit the number of engineering hours.

2.4. Project justification

In order to be able to implement the low cost BPI VI needs to make sure that it is possible. The research and realization of a proof of concept for the low cost product, serves as the subject of my graduation assignment. The steps I need to follow in order to come up with a result are:

- Analyze the requirements of VITAL.BPI and select together with the stakeholder, a set of minimum requirements that will be implemented.
- Research for an open source (low cost) reporting tool that could successfully replace QlikView and will be able to provide all the functionalities required by the stakeholders.
- Implement the proof of concept based on the defined requirements and the chosen reporting technology.

2.5. Project product

Name: Lowcost.BPI (proof of concept)

Description: The final product of this project will be a blue print of the Lowcost.BPI.

Product information: The product will be delivered together with the deliverables listed in the next section.

2.6. Project deliverables and non-deliverables

The delivery of the proof of concept for the Lowcost.BPI will include the following document.



Deliverable	Recipients	Delivery Date	Delivery Method	Comments
Communication Plan Project Plan	Gonard Jaspers Internship Supervisor	11.02.2011	Hand in	Describes the communication means
	Henk Aarts Internship Tutor		Email	Includes: Project definition, Organization, Phasing, Management Plan
Component Requirements Specification (CRS)	Gonard Jaspers Internship Supervisor	11.03.2011	Hand in	Describes requirements development and related test activities. Includes details about system capabilities, interface requirements and other requirements
Component Design Specification (CDS)	Gonard Jaspers Internship Supervisor	25.03.2011	Hand in	Describes the component design and related activities (functional design, detailed design)
Component test plan	Gonard Jaspers Internship Supervisor	16.05.2011	Hand in	Describes the test cases
Component test report	Gonard Jaspers Internship Supervisor	25.05.2011	Hand in	Describes the result of the test cases.
Acceptance demo	Gonard Jaspers Internship Supervisor	03.06.2011	Hand in	

Non-deliverables

- 👉 Full LOWCOST.BPI product

2.7. Project constraints

The following constraints are applicable:

- 👉 The duration of the project is depending on the graduation period (01.02.2011 – 01.07.2011)
Start Date: 01.02.2011
Estimated End Date: 03.06.2011
Maximum End Date: 09.06.2011 (Hand in graduation report)



- Apart from the reporting tool which is to be selected the 'proof of concept' shall be based upon the current VITAL.BPI product development tools;:
- Oracle database
 - PL-SQL
 - Java

2.8. Project risks

Three essential factors in managing the risks are probability, impact and categories.

Id	Risk description	Risk effect	Probability	Impact	Risk categories	Preventive measure	Corrective measure	Owner
	The found reporting tools don't match with the requirements.		Improbable	Critical	M			
	The graduation assignment is not completed		Improbable	Critical	M	Stick to the schedule	Less functionalities	
	Illness		Probable	Marginal	M	The project schedule takes into account possible illness		
	Low level of skills		Improbable	Critical	M	Prepare ahead by studying in depth every subject involved in the implementation of the application.		
	Difference of opinions		Improbable	Negligible	L	Establish good communication with internship supervisor and stakeholder(s). Organize meetings and discuss till a common point is reached.		
	Computer crash		Probable	Marginal	M	Archive all documents and files in Synergy		
	Technical failure of the application					Make sure that when the project is finished there is still sufficient time planned to debug and test the application.		
	Changes in the user requirements		Improbable	Marginal	L	Meet with the client and discuss the requirements in details.		



	Project size		Improbable	Marginal	L	The project schedule has to be done carefully and has to take into account the possible risks.		
--	--------------	--	------------	----------	---	------------------------------------------------------------------------------------------------	--	--

Table 13 Risks Table

Legend:

Risk Category	Impact →	Catastrophic	Critical	Marginal	Negligible
Probability ↓	Value	S	A	B	C
Almost Certain	4	H	H	M	M
Very Likely	3	H	H	M	M
Probable	2	H	M	M	L
Improbable	1	M	M	L	L

Risk category	Value
L	Low
M	Medium
H	High

3. Project Phasing

This chapter contains the project work breakdown into phases. The activities and deliverables are listed for each phase. The project plan will be updated periodically with information concerning the working hours needed to finalize each activity (Process document section).

3.1. Planning phase

Main objective:

The main objective of this phase is to get acquainted with the assignment and create a compact and clear planning to guide the execution of the project.

<u>Activities:</u>	<u>Sub-deliverables:</u>
<ul style="list-style-type: none"> ➤ Create/maintain project plan ➤ Create/maintain project planning ➤ Create communication plan ➤ Create deliverables list 	<ul style="list-style-type: none"> ➤ Project plan ➤ Planning ➤ Communication plan ➤ Deliverables list

Estimated time: **2 weeks**

Estimated end date: **11th February 2011**

Process document(This section will be updated at the end of the phase)

Activity	Man-hours	Actual time
Project plan	20	28
Planning	9	9
Communication plan	2	2



Deliverables list	4	4
Review project plan	3	6

3.2. Project management

Main objective:

This phase includes activities that are not directly related to the final product. It refers more to how the project is managed, the writing of the graduation report and other activities.

Activities:	Sub-deliverables:
<ul style="list-style-type: none"> ➤ Create/maintain graduation report ➤ Create report structure/write introduction ➤ Update 1 (project planning and startup) ➤ Update 2 (Specification) ➤ Update (Design) ➤ Update (Implementation - 1) ➤ Update (Implementation - 2) ➤ Update (Review and testing) ➤ Final Version ➤ Other activities 	<ul style="list-style-type: none"> ➤ Graduation report ➤ Project status report

Estimated time: **18 weeks**

Estimated end date: **3rd June 2011**

Process document(This section will be updated at the end of the phase)

Activity	Man-hours	Actual time
➤ Create/maintain graduation report	39	
➤ Create report structure/write introduction	2	
➤ Update 1 (project planning and startup)	1	
➤ Update 2 (Specification)	3	
➤ Update (Design)	4	
➤ Update (Implementation - 1)	2	
➤ Update (Implementation - 2)	3	
➤ Update (Review and testing)	8	
➤ Final Version	16	
Other activities	130	



3.3. Project start-up

Main objective:

During the start-up phase the preparation for the requirements phase takes place. This includes requirements analysis and research for the reporting technology. Besides this, it also includes the training program.

Activities:	Sub-deliverables:
<ul style="list-style-type: none">➤ BPI/Synergy/SWH processes training➤ Requirements analysis➤ Reporting Tool - research	<ul style="list-style-type: none">➤ List with defined requirements➤ Document containing the reason why a certain reporting technology will be chosen.

Estimated time: **3 weeks**

Estimated end date: **4th March 2011**

Process document(This section will be updated at the end of the phase)

Activity	Man-hours	Actual time
➤ BPI/Synergy/SWH processes training	9	
➤ Requirements analysis	36	
➤ Reporting Tool - research	32	

3.4. Iteration 1

The activities related to requirements, design and implementation were scheduled in three iterations. The objective of the first iteration is to test how a report can be implemented with the new reporting tool(s). For this purpose, two already existing reports were used.

The requirements phase is an important part of the project. Is the time when the minimum set of requirement has to be defined. This phase serves as a foundation for the remaining phases of the project and it's most important deliverable is the Component Requirements Specification (CRS) document.

During the design phase the component design document is created. This document will contain information about the functional design, details about the design of each report (KPI) and the user presentation design.

The implementation phase is the part of the project where the proof-of-concept will be created.

Requirements	Sub-deliverables
<ul style="list-style-type: none">➤ Define requirements(for the two chosen reports)➤ Create CRS	<ul style="list-style-type: none">➤ CRS document➤ Process document



Estimated time: 2 weeks Estimated end date: 18th March 2011	
Design	
<ul style="list-style-type: none"> ➤ Create/maintain component design 	<ul style="list-style-type: none"> ➤ Component design specification document ➤ Process document
Estimated time: 3 weeks Estimated end date: 25st March 2011	
Implementation	
<ul style="list-style-type: none"> ➤ Develop first KPI: Number of Parcels per Infeed ➤ Develop second KPI: Number of Parcels per Chute 	<ul style="list-style-type: none"> ➤ Report files ➤ Process document
Estimated time: 5 weeks Estimated end date: 1st April 2011	

3.5. Iteration 2

During the second iteration it is planned to implement a new report. This activity will measure the engineering hours needed to develop it.

Requirements	Sub-deliverables
<ul style="list-style-type: none"> ➤ Update requirements(for the new report) 	<ul style="list-style-type: none"> ➤ CRS document ➤ Process document
Estimated time: 1 week Estimated end date: 8th April 2011	
Design	
<ul style="list-style-type: none"> ➤ Update design document 	<ul style="list-style-type: none"> ➤ Component design specification document ➤ Process document
Estimated time: 1 weeks Estimated end date: 8th April 2011	
Implementation	
<ul style="list-style-type: none"> ➤ Develop the new KPI (to be defined later) 	<ul style="list-style-type: none"> ➤ Report files ➤ Process document
Estimated time: 2 weeks Estimated end date: 22nd April 2011	

3.6. Iteration 3

The third iteration is scheduled for implementing a more complex report, but it can also be used to improve previous reports and/or find solution to use the reporting tools more efficiently.



Requirements	Sub-deliverables
<ul style="list-style-type: none"> Update requirements <p>Estimated time: 1 week Estimated end date: 29th April 2011</p>	<ul style="list-style-type: none"> CRS document Process document
Design	
<ul style="list-style-type: none"> Update design document <p>Estimated time: 1 weeks Estimated end date: 29th April 2011</p>	<ul style="list-style-type: none"> Component design specification document Process document
Implementation	
<ul style="list-style-type: none"> Develop the new KPI (to be defined later) <p>Estimated time: 1 weeks Estimated end date: 6th May 2011</p>	<ul style="list-style-type: none"> Report files Process document

3.7. Testing phase

Main objective:

During the testing phase all the functionalities developed during the implementation phase, will be tested.

Activities:	Sub-deliverables:
<ul style="list-style-type: none"> Create/maintain test plan Execute tests Write test report Prepare/perform stakeholder(s) acceptance demo 	<ul style="list-style-type: none"> Test plan; Test report; Process document; Acceptance demo

Estimated time: **3 weeks**
Estimated end date: **27th May 2011**

Process document(This section will be updated at the end of the phase)

Activity	Man-hours	Actual time
<ul style="list-style-type: none"> Create/maintain test plan 	62	
<ul style="list-style-type: none"> Execute tests 	36	
<ul style="list-style-type: none"> Write test report 	10	
<ul style="list-style-type: none"> Prepare/perform stakeholder(s) acceptance demo 	16	

3.8. Evaluation phase

Main objective:

The purpose of this phase is to determine whether the project was a success.



Activities:	Sub-deliverables:
<ul style="list-style-type: none"> ➤ Create final presentation ➤ Meeting with university tutor (practice presentation) ➤ Create close down project report 	<ul style="list-style-type: none"> ➤ Close down project report

Estimated time: **3 weeks**
Estimated end date: **3rd June 2011**

Process document(This section will be updated at the end of the phase)

Activity	Man-hours	Actual time
➤ Create final presentation	3	
➤ Meeting with university tutor (practice presentation)	8	
➤ Create close down project report	6	

3.9. Money

Man hours	712
Software costs	-----
Housing	-----
Wage/month	450 €
Total costs	2250 €

3.10. Skills

The skills required for this project are:

- Good writing skills
- Good communication skills - the ability to extract information from the parties involved in the project.
- Organizational skills – the capability of making accurate plans and designs.
- Programming skills - PL/SQL, Java;

3.11. Quality

The quality of the project at hands depends heavily on the functionality and the performance of the final product and the clarity of the documentation. With the help of use cases, users' behavior will be simulated and runtime errors will be visible. To make sure that the quality of this project is up to specification, the implemented functionalities will be tested and an acceptance demo will be created for the stakeholders.



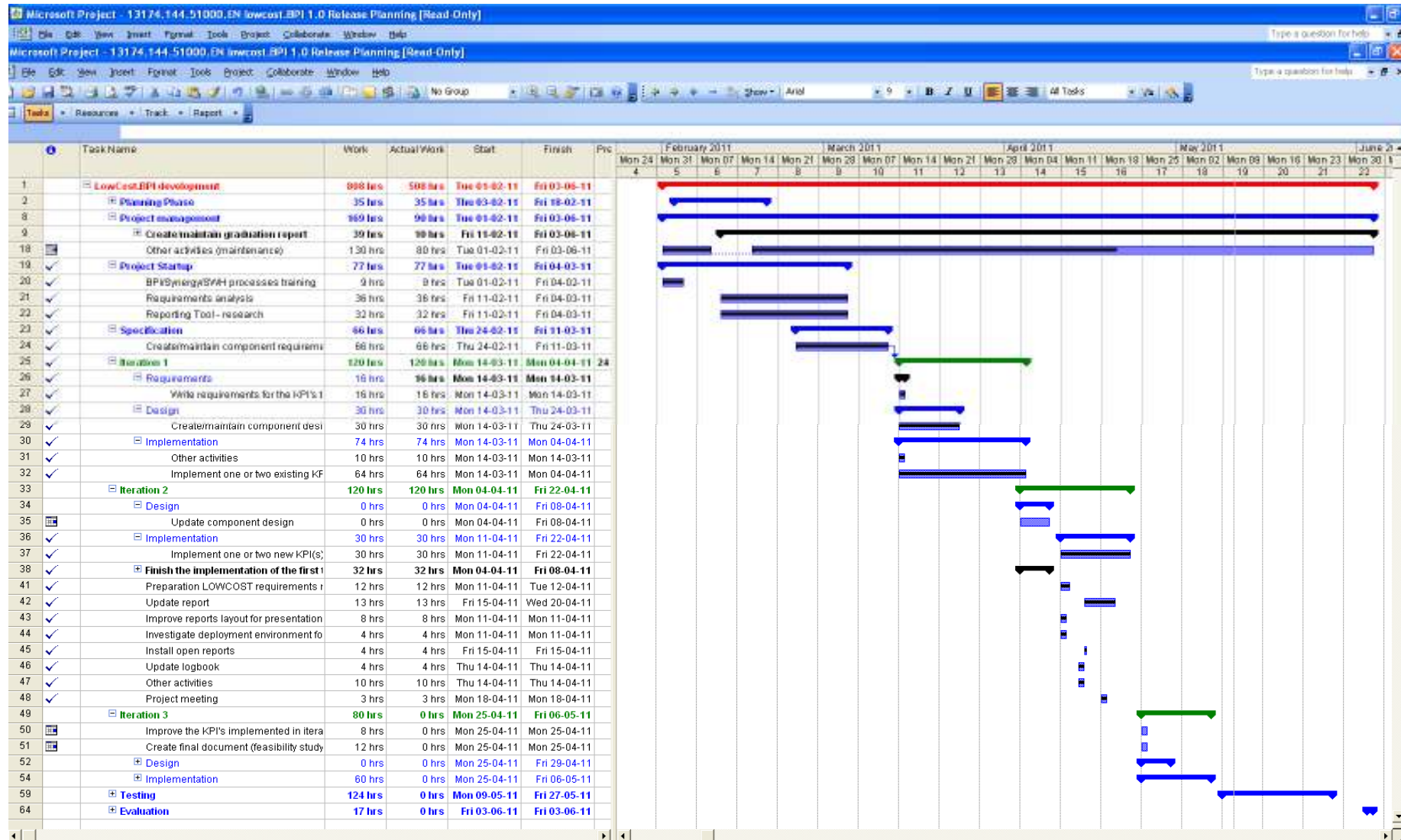
3.12. Information

The information matrix below shows the flow of information between all parts involved in the project:

Documents Author	Project Plan	Component Requirements	Component Design	Test plan	Test Report	Graduation Report
Gonard Jaspers Company Supervisor	C A	C A	C A	C A	C A	E A
Henk Aarts School Tutor	C	C	C	C	C	E A
Student	M S Ach	M S Ach	M S Ach	M S Ach	M S Ach	M S Ach

Legend: **A**- Approve | **Ach** – Archive | **C** – Check | **E** – Evaluate | **M** – Make | **S** – Send

3.13. Time



3.14. Organization

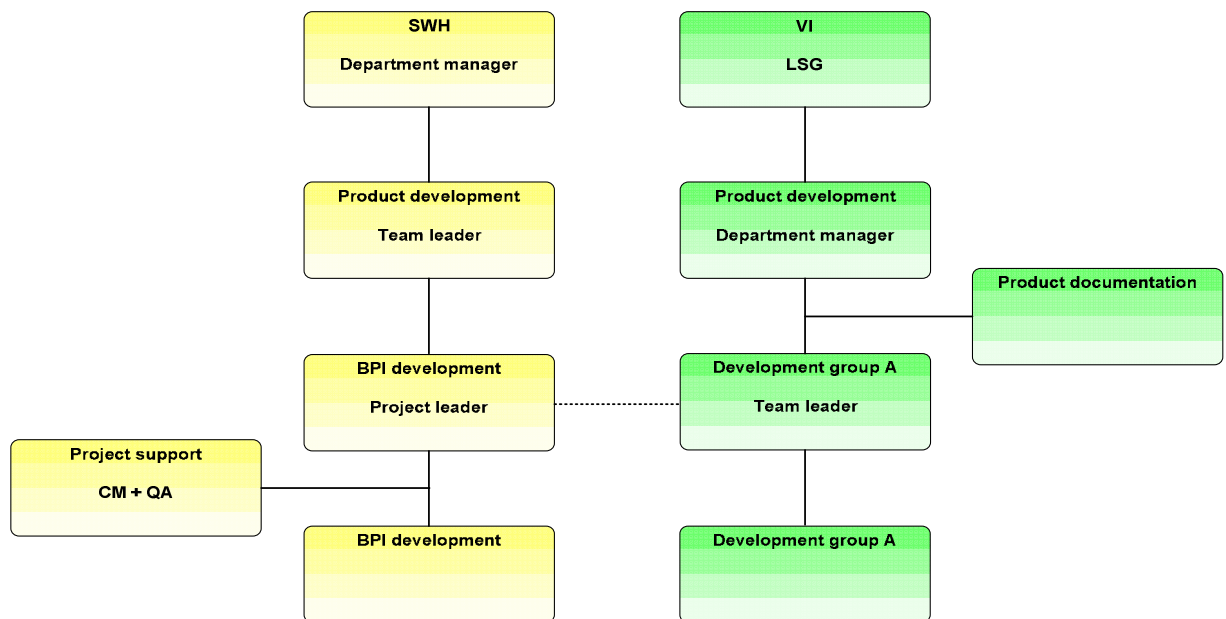


Figure 20 - The product organisation

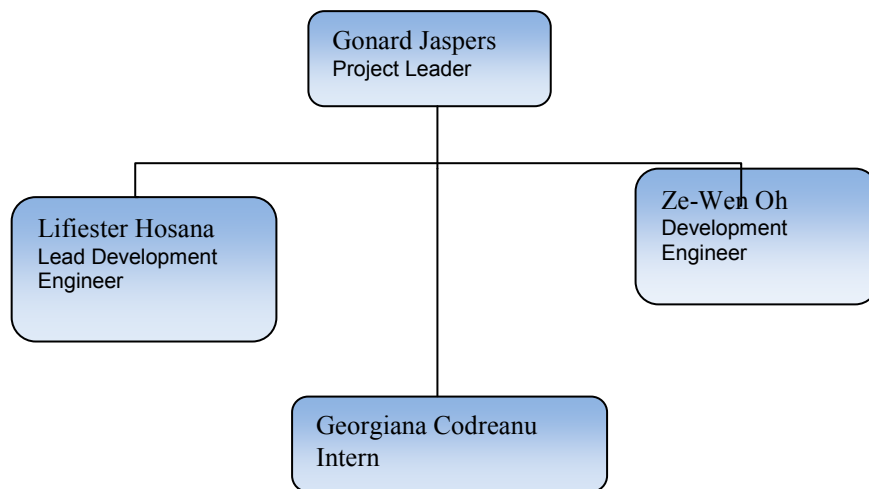


Figure 21 - Product development team

References

Reference	Title	Version	Author	Location	Remarks
[BPI product plan]	Global BPI Project Plan	005	Gonard Jaspers	BPI file server	
[Lowcost.BPI release plan]	Lowcost.BPI release plan	001	Georgiana Codreanu		
[Lowcost.BPI]	Lowcost.BPI 1.0 Deliverable List + WBS		Georgiana Codreanu		
[Lowcost.BPI 1.0 PSR]	Lowcost.BPI 7.0 Project Status Report		Gonard Jaspers		



Appendix A - Meetings

Name	Purpose	Participants	Frequency	Y/N	Minutes
					Distribution list
Project meeting	Discuss project status	Gonard Jaspers Lifiester Hosana Ze-Wen Oh Georgiana Codreanu	Weekly		Participants
Meetings with the stakeholder(s)	Define KPI's	Lowcost.BPI Stakeholder(s) Gonard Jaspers (Project Leader) Georgiana Codreanu	Start of the project		
Introduction meeting	Discuss current situation and project plan	Lowcost.BPI Stakeholder(s) Gonard Jaspers(Project Leader) Henk Aarts (University tutor) Georgiana Codreanu	Start of the project		Participants
Acceptance demo	Present prototype		End of the project		
Evaluation meeting			End of the project		
Final Presentation	Graduation presentation	Gonard Jaspers Henk Aarts Graduation Comitee	End of the project		Participants



Appendix C – Project Planning

	i	Task Name	Work	Actual Work	Start	Finish
1		▢ LowCost.BPI development	712 hrs	696 hrs	Tue 01-02-11	Mon 27-06-11
2		▢ Planning Phase	35 hrs	35 hrs	Thu 03-02-11	Fri 18-02-11
3	✓	Create/maintain project plan	20 hrs	20 hrs	Thu 03-02-11	Fri 11-02-11
4	✓	Create/maintain project planning	9 hrs	9 hrs	Fri 04-02-11	Tue 08-02-11
5	✓	Create communication plan	2 hrs	2 hrs	Fri 04-02-11	Mon 07-02-11
6	✓	Create deliverable list	4 hrs	4 hrs	Mon 07-02-11	Tue 08-02-11
7	📅	Estimation meeting	0 hrs	0 hrs	Fri 18-02-11	Fri 18-02-11
8	✓	▢ Project management	113 hrs	113 hrs	Tue 01-02-11	Fri 03-06-11
9	✓	▢ Create/maintain graduation report	63 hrs	63 hrs	Fri 11-02-11	Fri 03-06-11
18	✓	Other activities (maintenance)	50 hrs	50 hrs	Tue 01-02-11	Fri 03-06-11
19	✓	▢ Project Startup	77 hrs	77 hrs	Tue 01-02-11	Fri 04-03-11
20	✓	BPI/Synergy/SWH processes training	9 hrs	9 hrs	Tue 01-02-11	Fri 04-02-11
21	✓	Requirements analysis	36 hrs	36 hrs	Fri 11-02-11	Fri 04-03-11
22	✓	Reporting Tool - research	32 hrs	32 hrs	Fri 11-02-11	Fri 04-03-11
23	✓	▢ Specification	66 hrs	66 hrs	Thu 24-02-11	Fri 11-03-11
24	✓	Create/maintain component requireme	66 hrs	66 hrs	Thu 24-02-11	Fri 11-03-11
25	✓	▢ Iteration 1	120 hrs	120 hrs	Mon 14-03-11	Mon 04-04-11
26	✓	▢ Requirements	16 hrs	16 hrs	Mon 14-03-11	Mon 14-03-11
28	✓	▢ Design	30 hrs	30 hrs	Mon 14-03-11	Thu 24-03-11
30	✓	▢ Implementation	74 hrs	74 hrs	Mon 14-03-11	Mon 04-04-11
33		▢ Iteration 2	120 hrs	120 hrs	Mon 04-04-11	Fri 22-04-11
34		▢ Design	0 hrs	0 hrs	Mon 04-04-11	Fri 08-04-11
36	✓	▢ Implementation	30 hrs	30 hrs	Mon 11-04-11	Fri 22-04-11
38	✓	▢ Finish the implementation of the first	32 hrs	32 hrs	Mon 04-04-11	Fri 08-04-11
41	✓	Preparation LOWCOST requirements r	12 hrs	12 hrs	Mon 11-04-11	Tue 12-04-11
42	✓	Update report	13 hrs	13 hrs	Fri 15-04-11	Wed 20-04-11
43	✓	Improve reports layout for presentation	8 hrs	8 hrs	Mon 11-04-11	Mon 11-04-11
44	✓	Investigate deployment environment fo	4 hrs	4 hrs	Mon 11-04-11	Mon 11-04-11
45	✓	Install open reports	4 hrs	4 hrs	Fri 15-04-11	Fri 15-04-11
46	✓	Update logbook	4 hrs	4 hrs	Thu 14-04-11	Thu 14-04-11
47	✓	Other activities	10 hrs	10 hrs	Thu 14-04-11	Thu 14-04-11
48	✓	Project meeting	3 hrs	3 hrs	Mon 18-04-11	Mon 18-04-11
49	✓	▢ Iteration 3	125 hrs	125 hrs	Mon 25-04-11	Fri 20-05-11
50	✓	Improve the KPI's implemented in itera	32 hrs	32 hrs	Mon 25-04-11	Thu 28-04-11
51	✓	Create final document (feasibility study	12 hrs	12 hrs	Mon 25-04-11	Mon 25-04-11
52	✓	▢ Design	2 hrs	2 hrs	Mon 25-04-11	Fri 29-04-11
54	✓	▢ Implementation	79 hrs	79 hrs	Mon 25-04-11	Fri 20-05-11
59		▢ Testing	40 hrs	40 hrs	Tue 01-02-11	Fri 27-05-11
65		▢ Evaluation	16 hrs	0 hrs	Fri 10-06-11	Mon 27-06-11
66	📅	Meeting with university tutor	2 hrs	0 hrs	Mon 20-06-11	Mon 20-06-11
67	📅	Create final presentation	12 hrs	0 hrs	Fri 10-06-11	Mon 20-06-11
68	📅	Close down project	2 hrs	0 hrs	Mon 27-06-11	Mon 27-06-11



Appendix D – Reporting Tools Comparison

	BIRT	JasperReports	Pentaho
Website	www.birt-exchange.com/www.eclipse.org/birt	www.jasperforge.com	reporting.pentaho.com
License	Eclipse Public License	GNU Lesser General Public License	GNU Lesser General Public License
Report designer	BIRT Report Designer 2.6.0	Jasper iReport 4.0	Pentaho Report Designer 3.5.0 stable
Designer Platforms	Windows, Linux, Mac OS X	Windows, Linux, Mac OS X	Windows, Linux, Mac OS X
Standalone Java Client Available	Y	Y	Y
Report Format	XML (.RPTDESIGN files are pure XML)	.JRXML report design files are compiled into .JASPER report files, which are Java Byte Code. You then deploy the .JASPER file.	XML (.PRPT report file is a ZIP that contains the XML file plus other resources)
Common Report Designer Components: Report Editor Palette Data Explorer Property Editor Outline view report structure Report Preview Expression Builder Report Problems Chart Builder Script Editor	Y	Y	Y
Sub-reports	Y	Y	Y
Tables	Y	N	N
Cross-tabs	Y	Y	“Experimental” in Pentaho 3.5
Hyperlinks within a report	Y	Y	Y
Actionable charts	Y	Partial – hyperlinks	N
Cascading Style Sheets (CSS controlled format)	Y	N	N
Conditional Formatting	Y	Partial	Y
Multiple data sources and queries per report	Yes, plus support for joining them	Only via sub-reports (one data source per sub-report)	Only via sub-reports (one data source per sub-report) or in charts
Support for joining multiple data sources in the Designer	Y	N	N



Report can further re-sort, filter, or group data returned from a query.	Y	Partial – Jasper can further manipulate data sets before building cross-tabs, but not for regular tables.	N
Data Sources Types	- Database JDBC	- Database JDBC	- Database JDBC
	- XML File	- XML File	- XML files
	- Web Services	- CSV File	- Table
	- Flat files: CSV,SSV,PSV,TSV	- Microsoft Excel	- OLAP MDX
	- Custom data source	- JavaBeans	- Pentaho Metadata
	- OLAP MDX	- Hibernate HSQL	- Pentaho Data Integration
	- Scripted Data Source: POJO,EJB,Hibernate,XM L Stream	- Spring Hibernate EJBQL	
		- XMLA Server	- Scripted Data Source: POJO,EJB,Hibe rnative
		- Mondrian OLAP	
		- Text File	
		- NetBeans JDBC	
		- POJOs	
		- Custom	
		- Remote XML files	
Query Designer	Y	Y	Y
Graphical Query Designer	Prototype only	Y (SQL Leonardo)	Y (SQL- Leonardo)
Scripting	- JavaScript	- JavaScript	- JavaScript
	- Java Event Handlers	- Groovy	- Bean Script Framework (BSF)
		- Java	- Bean-Script Host (BSH)
			- Single Value Query
PDF	Y	Y	Y
Excel (XLS)	Y	Y	Y
XML	Y	Y	Y
Plain Text	Y	Y	Y
RTF	Y	Y	Y
Powerpoint (PPT)	Y	N	N
CSV	Y	Y	Y
Postscript	Y	Y	Y
OpenOffice report types (document & spreadsheet)	N	Y	N



Microsoft Office 2007 report types (DOCX, XLSX)	N	Y	N
Custom Formats	Y	Y	Y
Geometric shapes & lines	Y	Y	Y
Barcodes	Y	Y	Y
Chart Wizard	Y	Partial two-step wizard, rest is a big dialog box	N
Chart Interactivity	Y	Y	N
	mouse-over, tool tips, hyperlinks, etc.	hyperlinks only	
Chart themes	Y	Y	N
Precise control over format of all control elements	Y	N	N
Charts types that all have: 2D 3D Pie Multi-pie Bar Stacked Bar Bar XY Line Line XY Area Area XY Stacked Area Bar Line Bubble Scatter Plot Multi-Axis	Y	Y	Y
Ring Chart	N	N	Y
Tube chart	Y	N	N
Cone chart	Y	N	N
Pyramid	Y	N	N
Time Series	Y	Y	N
Meter / Gauge	Y	Y	N
Waterfall	N	N	Y
Step	N	N	Y
Step Area	N	N	Y
Difference	Y	N	Y
Radar	Y	N	Y
Candlestick / Stock Chart (High/Low)	Y	Y	N
Gantt	Y	Y	N
Survey Scale	N	N	Y
Bar Sparkline	N	N	Y
Line Sparkline	N	N	Y
Pie Sparkline	N	N	Y
Static Parameters select parameter values from a hard-coded list of values	Y	Y	Y
Dynamic Parameters users select parameters from a list	Y	Y	Y



of values that came from a database		Cascading input controls are report independent	
Cascading parameters	Y	Y	Y
linked data driven prompts e.g.			
- Country			
- State			
- City			
Calendar date-picker for parameters of type date.	Y	Y	Y
Can specify default values	Y	Y	Y
Drop-down list boxes	Y	Y	Y
Radio buttons	Y	Y	Y
Check boxes	Y	Y	Y
Combo boxes	Y	Y	Y
Common Aggregations	- Average	- Average	- Average
	- Count	- Count	- Count
	- Distinct Count	- Distinct Count	- Count by Page
	- First	- Sum	- Group Count
	- Is-Bottom-N	- First	- Sum
	- Is-Bottom-N-Percent	- Lowest (Minimum)	- Minimum
	- Is-Top-N	- Highest (Maximum)	- Maximum
	- Is-Top-N-Percent	- Standard Deviation	- Sum Quotient
	- Last	- Variance	- Sum Quotient Percent
	- Max	- System	- Calculation
	- Median		- Count for Page
	- Min		- Sum for Page
	- Mode		- Sum (Running)
	- Moving Ave		- Count (Running)
	- Percentile		- Group Count (Running)
	- Percent-Rank		- Count Distinct (Running)
	- Percent-Sum		- Average (Running)
	- Quartile		- Minimum (Running)
	- Rank		- Maximum (Running)
	- Running Count		- Percent of Total (Running)
	- Running Sum		
	- Standard Deviation		



	- Sum		
	- Variance		
	- Weighted Average		
User Defined Functions / Expressions	Y	Y	Y
	Java, JavaScript	Java, JavaScript, or Groovy	OpenFormula (Excel-like), Java
User Define Aggregates	Y	Y	Y
Designer provides “code hooks” (aka callbacks, event handlers. ...) for each report element.	Y	Partial – only the report itself has callbacks – not at the per-element level.	Not in Designer
Templates			
(custom report starting points)	Y	Y	Y
User-defined Libraries			
(reusable report pieces)	Y	N	N
CSS	Y	N	N



Appendix E – Research Log-book

Reporting tools		JasperReports	BIRT
Download		JasperServer 4.0.0	Eclipse framework
		iReports 4.0	OpenReports
		JasperReports Library	BIRT library
License		Open source	Open Source
		Offers the possibility to buy more functionality with a commercial version;	Offers the possibility to buy more functionality with a commercial version;
Installation	Server	JasperServer install package comes with MySQL database and TOMCAT server	BIRT does not have a server where to deploy reports, but it comes with an example of how to deploy them in a web application.
		If you already have MySQL and TOMCAT installed on your computer, you can choose the option "use already installed MySQL" or "use already installed TOMCAT".	The example is called the BIRT viewer and in order to run it has to be installed in the TOMCAT server.
		During the installation I encountered some problems when I tried to use bundled MySQL and already installed TOMCAT.	The installation of the BIRT viewer consists in just copying the BIRT viewer folder into WEB-APP folder of Apache TOMCAT, starting the TOMCAT manager. The TOMCAT manager will open and you can see the BIRT viewer in the application list.
		The only way I could make it work at that time was to first install MySQL 5.1 and use the bundled TOMCAT.	Another deployment option for BIRT reports (as well as for Jasper reports) is the OpenReports application. This application offers the minimum functionalities necessary for deploying, exporting and scheduling reports.
	Designer	iReports - standalone graphical program that provides designer capabilities	Install Eclipse framework and start a new Business Intelligence project (the Eclipse BIRT designer will be open)
		Install iReports designer and create a connection with JASPERSERVER. The designed reports can be uploaded directly from the report on to the server.	
		Easy to install and configure, but not easy to use.	Good report designer and charting support
		odbc driver needs to be added in the class path of iReports: tools->options->classpath	When adding a new data source the specific odbc driver needs to be added (More details are given in the BIRT user manual)
		Both tools have more or less the same functionalities when it comes to creating reports. First impression was that the BIRT designer is easier to understand. After just one hour I was already able to create a simple report.	



		Easy to integrate and flexible for a developers point of view	Good support for reports with multiple data sources
			Harder to integrate with OpenReports then other report engines.
Create Reports	Connection with database	Add a new database connection.	Add a new database connection.
	Query	Define the report query to get the necessary fields from the database	Create dataset - a query has to be defined for each dataset
		Drag the resulted fields to the details section of the report page.	When using CROSSTABS, cubs have to be created.
	Designing	JasperReports have a well defined structure: Title, Page Header, Column Header, Details section, Column Footer, Page Footer. The disadvantage is that you have to create the reports based on that structure and it takes more time when you want to deviate from it.	BIRT designer does not have a well defined structure for reports. Using grids you can create a the layout you want for your report (ex. add break page when necessary). A page break will not be visible in the designer, but only if the report is uploaded in BIRT viewer of OpenReports.
	Parameters		
		Report templates can be created in order to reduce the implementation time.	Templates can be created.
	Simple Reports (First report)	16 hours	8 hours
	Complex Reports	32 hours	16 hours
	KPIs	48 hours	32 hours
	<i>including the time to learn how the two reporting tools are working, investigate functionalities, how the parameters are used, and how the reports can be deployed</i>	<i>Every time I was designing a KPI in iReports, I had a lot of issues in finding the functionality I wanted to use.</i>	
	Charts	iReports uses the jfreechart library for charts	BIRT charts are easy to design. They range from line, tub charts, to 3D charts, time charts..
		no 3D charts	3D charts
		In preview mode not all chart labels are displayed.	
Deployment		JasperServer	BIRT viewer / OpenReports
		The reports can be uploaded right from iReports of through the JasperServer web application.	To add a new report in the BIRT viewer it is necessary to add a link to the page where you want your report to be displayed.
		For each report parameter created in iReports, an input control has to be defined on the JasperServer.	The report parameters don't have to be configured for the BIRT viewer.



		The input parameter has to have the exact name as the iReports parameter.	To add a report in OpenReports it is necessary to upload a new file through the web user interface.
		Includes scheduling, email, user authorization and user management functionalities.	Includes scheduling, email, user authorization and user management functionalities.
		For emailing the js.quartz.properties file has to be configured.	
Maintenance		No maintenance costs	No maintenance costs
		For information about the next releases of JasperServer and iReports check the jasperforge website.	For information about the next releases of BIRT check the birt-organization website.
Support		Jasperforge community forum	BIRT forum
		Jaspersoft website	
		JasperServer - Installation manual, user manual, release notes	



Appendix F - Package Example

A package has two parts, Package Header and Package Body. The Package Header is only used for declaration of types, variables, constants, functions. The Package Body is used to provide the implementation for the functions declared in the Package Header.

Package Header

```
create or replace
PACKAGE DMC_LOWCOST_INFEED AS

-----

-- INSERT

-----

FUNCTION insert_dm_lowcost_infeed
(i_dm          IN          dm_lowcost_infeed%ROWTYPE)
RETURN BOOLEAN;

-----

-- UPDATE

-----

FUNCTION update_dm_lowcost_infeed
(i_dm          IN          dm_lowcost_infeed%ROWTYPE)
RETURN BOOLEAN;

-----

-- PROCESS_FIELDS

-----

FUNCTION process_fields
(i_wc          IN          wct_7fsc_sortreport%ROWTYPE
,i_dm          IN          dm_lowcost_infeed%ROWTYPE
,i_io_dm       IN OUT      dm_lowcost_infeed%ROWTYPE
,i_io_ctl      IN OUT      definitions.contrl
)
RETURN BOOLEAN;

-----

-- IDEVENT

-----

FUNCTION is_idevent
(i_wc          IN          wct_7fsc_sortreport%ROWTYPE
,i_dm          IN          dm_lowcost_infeed%ROWTYPE
,i_io_dm       IN OUT      dm_lowcost_infeed%ROWTYPE
,i_io_ctl      IN OUT      definitions.contrl
)
RETURN BOOLEAN;
```



```
-- INSERTTS

FUNCTION is_insertts
  (i_wc          IN      wct_7fsc_sortreport%ROWTYPE
  ,i_dm          IN      dm_lowcost_infeed%ROWTYPE
  ,io_dm         IN OUT  dm_lowcost_infeed%ROWTYPE
  ,io_ctl       IN OUT  definitions.contrl
  )
  RETURN BOOLEAN;

-- EVENTTS

FUNCTION is_eventts
  (i_wc          IN      wct_7fsc_sortreport%ROWTYPE
  ,i_dm          IN      dm_lowcost_infeed%ROWTYPE
  ,io_dm         IN OUT  dm_lowcost_infeed%ROWTYPE
  ,io_ctl       IN OUT  definitions.contrl
  )
  RETURN BOOLEAN;

-- PEP_ID (INFEED_ID)

FUNCTION is_pep_id
  (i_wc          IN      wct_7fsc_sortreport%ROWTYPE
  ,i_dm          IN      dm_lowcost_infeed%ROWTYPE
  ,io_dm         IN OUT  dm_lowcost_infeed%ROWTYPE
  ,io_ctl       IN OUT  definitions.contrl
  )
  RETURN BOOLEAN;

-- PERIODID

FUNCTION is_periodid
  (i_wc          IN      wct_7fsc_sortreport%ROWTYPE
  ,i_dm          IN      dm_lowcost_infeed%ROWTYPE
  ,io_dm         IN OUT  dm_lowcost_infeed%ROWTYPE
  ,io_ctl       IN OUT  definitions.contrl
  )
  RETURN BOOLEAN;

/* TODO enter package declarations (types, exceptions, methods etc) here */

END DMC_LOWCOST_INFEED;
```




Package Body – The implementation of insert and update, process fields, is_idevent and is_insertts functions, is given as an example below.

```
create or replace
PACKAGE BODY DMC_LOWcost_INFEED AS

-- INSERT

FUNCTION insert_dm_lowcost_infeed
(i_dm IN dm_lowcost_infeed%ROWTYPE)
RETURN BOOLEAN
IS
  c_proc varchar2(100) := 'dmc_lowcost_infeed.insert_dm_lowcost_infeed';
  funcname      TAtt;
BEGIN
  INSERT INTO dm_lowcost_infeed
  VALUES i_dm;
RETURN TRUE;

  EXCEPTION
    WHEN OTHERS THEN
      logger.logerror(c_proc, 'insert dm_lowcost_infeed record: ' || SQLERRM, TAlisT(funcname));
      RETURN FALSE;
END insert_dm_lowcost_infeed;

-- UPDATE

FUNCTION update_dm_lowcost_infeed
(i_dm IN dm_lowcost_infeed%ROWTYPE)
RETURN BOOLEAN
IS
  c_proc varchar2(100) := 'dmc_lowcost_infeed.update_dm_lowcost_infeed';
  funcname      TAtt;
BEGIN
RETURN TRUE;
EXCEPTION
WHEN OTHERS THEN
  logger.logerror(c_proc, 'update dm_lowcost_infeed record: ' || SQLERRM, TAlisT(funcname));
  RETURN FALSE;
END update_dm_lowcost_infeed;
```



```
-- PROCESS FIELDS

FUNCTION process_fields
(i_wc      IN      wct_7fsc_sortreport%ROWTYPE
,i_dm      IN      dm_lowcost_infeed%ROWTYPE
,io_dm     IN OUT  dm_lowcost_infeed%ROWTYPE
,io_ctl    IN OUT  definitions.contrl)
RETURN BOOLEAN
IS
    c_proc      varchar2(100) := 'dmc_lowcost_infeed.process_field';
    funcname     TAtt;
    bresult_ok   boolean;
BEGIN
    bresult_ok := is_idevent
        (i_wc      => i_wc
        ,i_dm      => i_dm
        ,io_dm     => io_dm
        ,io_ctl    => io_ctl);

    IF bresult_ok THEN
        bresult_ok := is_insertts
            (i_wc      => i_wc
            ,i_dm      => i_dm
            ,io_dm     => io_dm
            ,io_ctl    => io_ctl);
    END IF;

    IF bresult_ok THEN
        bresult_ok := is_eventts
            (i_wc      => i_wc
            ,i_dm      => i_dm
            ,io_dm     => io_dm
            ,io_ctl    => io_ctl);
    END IF;

    IF bresult_ok THEN
        bresult_ok := is_pep_id
            (i_wc      => i_wc
            ,i_dm      => i_dm
            ,io_dm     => io_dm
            ,io_ctl    => io_ctl);
    END IF;

    IF bresult_ok THEN
        bresult_ok := is_periodid
            (i_wc      => i_wc
            ,i_dm      => i_dm
            ,io_dm     => io_dm
            ,io_ctl    => io_ctl);
    END IF;

    RETURN bresult_ok;

EXCEPTION
WHEN OTHERS THEN
    logger.logerror(c_proc, 'process dm_lowcost_infeed: '||SQLERRM, TList(funcname));
    RETURN FALSE;
END process_fields;
```



```
-- IDEVENT
-----
FUNCTION is_idevent
(i_wc          IN      wct_7fsc_sortreport%ROWTYPE
,i_dm          IN      dm_lowcost_infeed%ROWTYPE
,io_dm         IN OUT  dm_lowcost_infeed%ROWTYPE
,io_ctl       IN OUT  definitions.contrl
)
RETURN BOOLEAN
IS
  c_proc varchar2(100) := 'dmc_lowcost_infeed.is_idevent';
  funcname      TAtt;
BEGIN
  io_dm.idevent := i_wc.idevent;
  RETURN TRUE;

EXCEPTION
WHEN OTHERS THEN
  logger.logerror(c_proc, 'get idevent: '||i_wc.idevent||' '||SQLERRM, TAlst(funcname));
  RETURN FALSE;
END is_idevent;
-----
-- INSERTTS
-----
FUNCTION is_insertts
(i_wc          IN      wct_7fsc_sortreport%ROWTYPE
,i_dm          IN      dm_lowcost_infeed%ROWTYPE
,io_dm         IN OUT  dm_lowcost_infeed%ROWTYPE
,io_ctl       IN OUT  definitions.contrl
)
RETURN BOOLEAN
IS
  c_proc varchar2(100) := 'dmc_lowcost_infeed.is_insertts';
  funcname      TAtt;
BEGIN
  io_dm.insertts := i_wc.insertts;
  RETURN TRUE;

EXCEPTION
WHEN OTHERS THEN
  logger.logerror(c_proc, 'get insertts: '||i_wc.idevent||' '||SQLERRM, TAlst(funcname));
  RETURN FALSE;
END is_insertts;
```



Appendix G – Procedure Example

```
create or replace
PROCEDURE DMU_LOW_COST_INFEED
(
  frominsertts IN DATE,
  untilinsertts IN DATE
)
IS
  c_proc          VARCHAR2(100) := 'bpi.dmu_lowcost_chute';
  funcname        TAtt;
  dlastinsertts   TIMESTAMP      := NULL;

  bresult_ok      BOOLEAN        := FALSE;
  nrec_count      NUMBER(10)     := 0;

  ctl             definitions.ctrl;

  in_dm           dm_lowcost_infeed%ROWTYPE;
  out_dm          dm_lowcost_infeed%ROWTYPE;

  CURSOR cparcels(b_frominsertts IN DATE, b_untilinsertts IN DATE)
  IS
    SELECT *
    FROM wct_7fsc_sortreport wc
    WHERE wc.insertts > b_frominsertts
    AND wc.insertts < b_untilinsertts
    ORDER BY wc.idevent;

  r_dm            wct_7fsc_sortreport%ROWTYPE;

BEGIN
  SAVEPOINT spdmu_lowcost;
  funcname := TAtt('DB_JOB', 'DM_LOW_COST_INFEED');
  --logger.logdebug ('DB_JOB', 'Enter', talist (funcname));

  --Step 1: Loop through the records of WCT_7FSC_SORTREPORT through 'cparcels'.
  FOR r_dm IN cparcels(frominsertts, untilinsertts)
  LOOP

    bresult_ok := dmc_lowcost_infeed.process_fields
      (
        i_wc => r_dm
      , i_dm  => in_dm
      , io_dm => out_dm
      , io_ctl => ctl
      );
  END LOOP;
END;
```



```
IF (bresult_ok) THEN
    bresult_ok := dmc_lowcost_infeed.insert_dm_lowcost_infeed
                (i_dm => out_dm);
ELSE
    bresult_ok := dmc_lowcost_infeed.insert_dm_lowcost_infeed
                (i_dm => out_dm);
END IF;

IF (r_dm.insertts > NVL(dlastinsertts,to_timestamp('01012010','DDMMYYYY'))) THEN
    dlastinsertts := r_dm.insertts;
END IF;

nrec_count := nrec_count + 1;

IF (dlastinsertts IS NOT NULL) THEN
    IF (MOD(nrec_count,2000) = 0) THEN
        bpidamarts.setuntilinsertts (datamartname_ => 'DMU_LOW_COST_CHUTE'
                                    ,lasteventts_ => dlastinsertts
                                    );
        COMMIT;
    END IF;
ELSE
    ROLLBACK TO SAVEPOINT spdmu_lowcost;
END IF;
END LOOP;

IF (dlastinsertts IS NOT NULL) THEN
    bpidamarts.setuntilinsertts (datamartname_ => 'DMU_LOW_COST_CHUTE'
                                ,lasteventts_ => dlastinsertts
                                );
    COMMIT;
ELSE
    ROLLBACK TO SAVEPOINT spdmu_lowcost;
END IF;

EXCEPTION
WHEN OTHERS THEN
    ROLLBACK TO SAVEPOINT spdmu_lowcost;
    logger.logerror('DB_JOB', SQLERRM || ': last exception ' || dbms_utility.format_error_backtrace(), TAlst(funcname));
END DMU_LOW_COST_INFEED;
```



Appendix G – Report File (JasperReports)

The report files created with JasperReports are automatically saved with the extension .jrxml. This files can be opened with iReports or deployed in JasperServer. To show the output of the KPI Number of Parcels Per Infeed the following report was created.

Number of parcels per infeed

VAN DER LANDE
INDUSTRIES

Parcel Overview

Number of parcels per infeed


Tuesday 10 May 2011

This KPI reports the number of parcels per infeed. This number is calculated based on the report parameters which in this case are YEAR and MONTH.

Report parameters

Selected Year: **2010**

Selected Month: **11**



Parcel & Postal

1



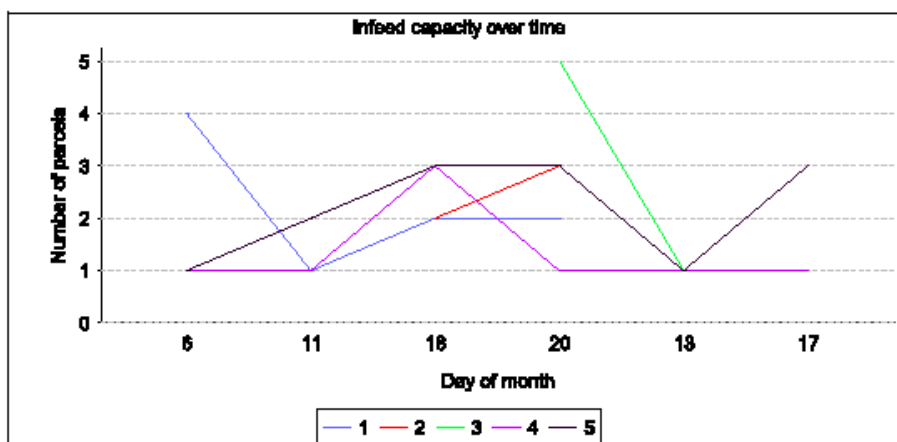
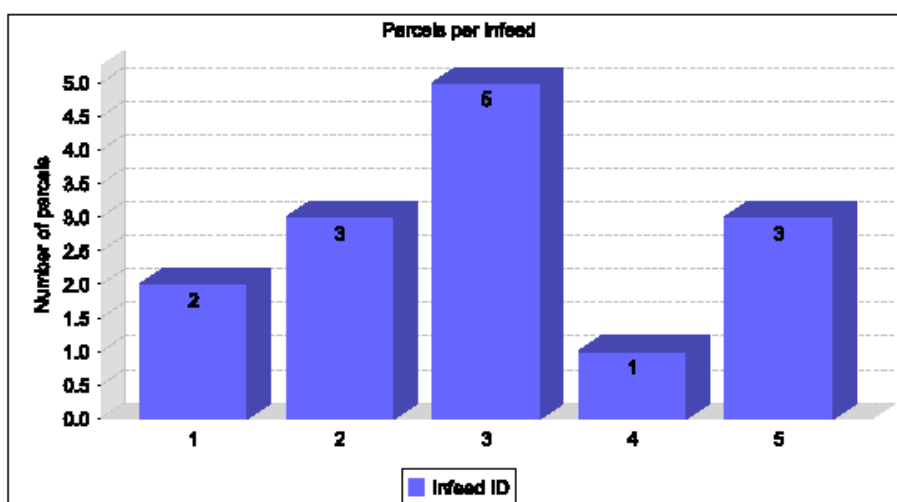
Number of parcels per infeed



Report parameters

Selected year: **2010**

Selected month: **11**





Number of parcels per infeed



Report parameters

Selected year: **2010**

Selected month: **11**

Detailed data

Day of month	6	11	13	17	18	20	Total per infeed
Infeed ID							
1	1	1	0	0	1	1	4
2	1	0	1	0	1	1	4
3	1	0	1	0	0	1	3
4	1	1	1	1	1	1	6
5	1	1	1	1	1	1	6
Total per day	5	3	4	2	4	5	23



Appendix H - Report File (BIRT)

The report files created with BIRT are automatically saved with the extensions .rtpdesign and .rtpconfig. These files can be opened with the BIRT designer or deployed in Apache TOMCAT server using the BIRT viewer. To show the output of the KPI Number of Parcels per Infeed the following report was created.

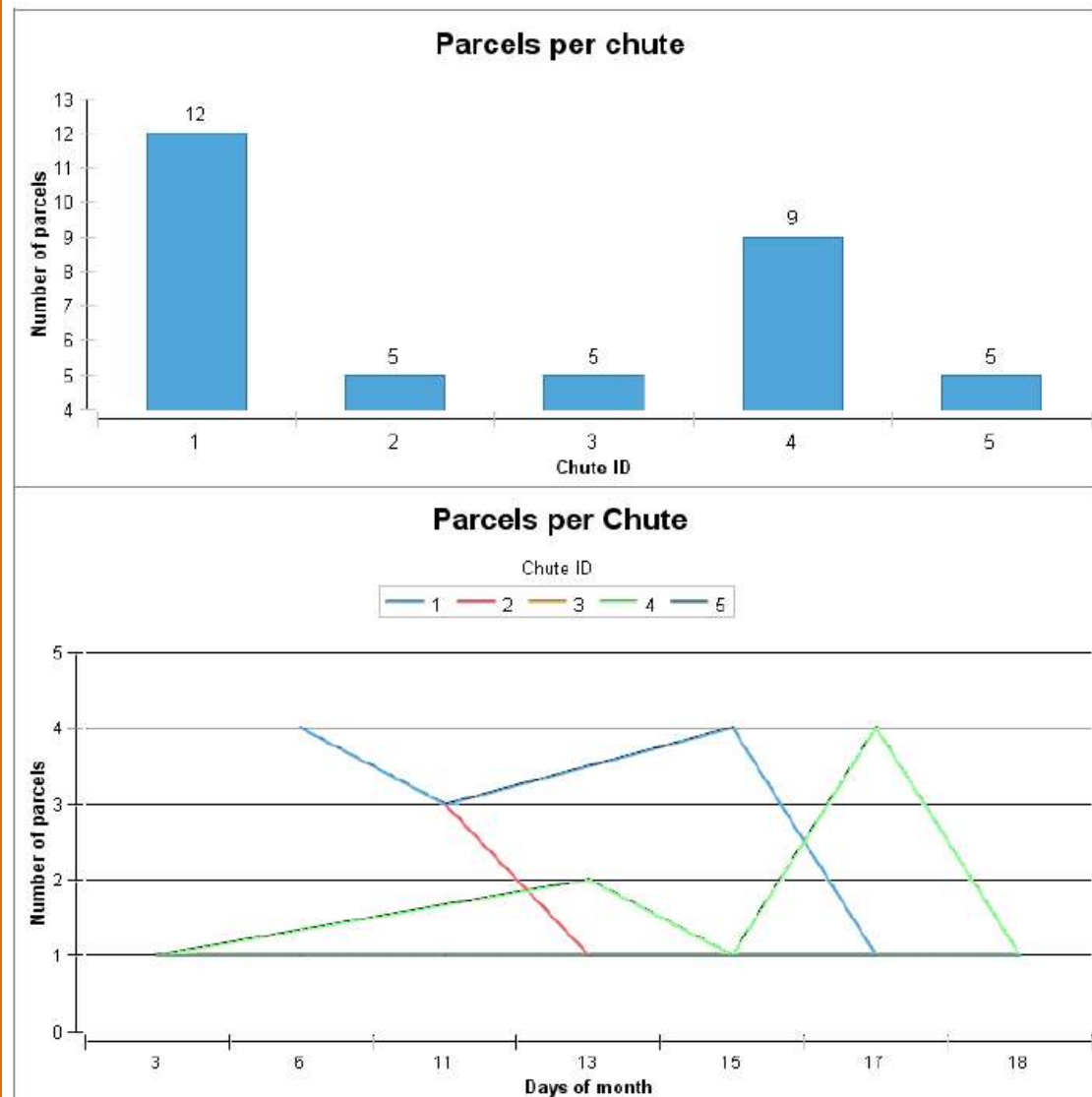




Charts

Report data is based on the selected year.

2010





Detailed data representing the number of parcel per chute.
Report data is based on the selected year.
2010

Day	3	6	11	13	15	17	18
1		4	3		4	1	
2			3	1		1	
3	1	1	1	1			1
4	1			2	1	4	1
5	1	1	1	1			1