CHALMERS

Dynamic Report Views

Implementation of Business Intelligence and Reporting Tools

Master of Science Thesis in the Programme Software Engineering and Technology

MD. Naimuzzaman

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Göteborg, Sweden, June 2009

Dynamic Report Views
Implementation of Business Intelligence and Reporting Tools

Naimuzzaman

Examiner: Katarina Blom
Supervisor: Magnus Carlsson

Department of Computer Science and Engineering
Chalmers University of Technology
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Department of Computer Science and Engineering
Göteborg, Sweden, June 2009

Table of Content

List of Figures and Tables

Abstract

English

The purpose of this study is to find out the suitable Business Intelligence and Reporting Tools (BIRT) for a stand alone system that can be implemented with it. It should also be able to generate the reports according to the user requirements. From the market research, Eclipse BIRT (EB) has been found as the best BIRT to be used among the twenty four open sources, free, and commercial BIRT. EB is an open source and free BIRT. Different technical details, strengths, weaknesses, features were analyzed to of all BIRT to select the best one. Eclipse BIRT has been used to design and develop reports. The designed report was implemented and integrated successfully with the GUI, which implies that third party BIRT can be used with the stand alone system. This paper also contains description of different critical challenges faced during project time and their solutions. Finally, Couple of charts also was implemented to test how it works; the developing time was estimated of the report and suggested some improvements in the project as well as in the EB.

Keywords: Business Intelligence and Reporting Tools, Eclipse BIRT, Reporting tools, Dynamic Report, EKO

Swedish

Syftet med den här uppsatsen är att hitta ett lämpligt beslutsstödsystem och rapporterings verktyg (Business Intelligence and Reporting Tools – BIRT) för att sedan utveckla ett fristående system med hjälp av detta. Dessa system och verktyg bör också kunna generera dynamiska rapporter enlig användarens behov. Efter att ha analyserat tjugofyra kommersiella produkter som finns tillgängliga med öppen källkod; jämfört dess styrkor, svagheter och funktionalitet, fann jag att Eclipse BIRT (EB) var lämpligast. EB har använts vid ett flertal tillfällen för att utveckla rapporteringssystem som integrerats med grafiska användargränssnitt framgångsrikt. Vilket indikerar att EB är en bra tredje parts BIRT som kan användas vid utveckling och integration av fristående system. Den här uppsatsen beskriver även flera problemställningar och dess lösningar som påträffats och genomarbetats under utvecklingens gång. Slutligen har jag bifogat exempel på tabeller för att beskriva rapportfunktionerna, tiduppskattningar för utveckling och förslag till förbättring av funktionalitet i systemen samt i EB.

Nyckelord: Business Intelligence and Reporting Tools, Eclipse BIRT, Reporting tools, Dynamic Report, EKO

Acknowledgment

First, I would like to convey my thanks to my supervisor Mr. Magnus Carlsson of Logica for his enthusiasm and endless time to time support throughout the thesis. I am grateful for his valuable supervision, interesting ideas and never-ending optimism.

I would also like to thank Lecturer Katarina Blom. She is my examiner from Chalmers University of Technology. She gave her valuable guidance and supported me during the project work and also in writing this thesis report.

Furthermore, I would like to express my gratitude to Mr. Lars Lundberg, Mrs. Ing-Marie Andrén for their valuable supports and comments during the development of the system. I also like to spread my thanks to Mr. Ivar Hegen for his endless supports to solve the problems regarding the setup of the system environment and other technical problems.

I would like to spread my thanks to those people who helped and supported me time to time during my thesis work whose names are not mentioned here.

Finally, my deepest gratitude goes to Almighty Allah for giving me such mental strength and support to complete the thesis work smoothly on time.

Introduction

Company Introduction

This thesis work was carried out in the Logica premises in Gothenburg, Sweden. The work has been done with the EKO project group under the close observation of the supervisor Mr. Magnus Carlsson. He is the project manager of EKO. He also works in different other projects (as project leader, architect and programmer), mostly from the office in Gothenburg.

Logica is one of the biggest and well known IT and Business Services Company not only in Europe but also in the world. It has almost 40,000 employees in 36 countries. Their main focus is on business consulting, systems integration along with IT and business process outsourcing [1].

Background

EKO is a solution to handle and organize all external financial contributions for Chalmers University of Technology, University of Gothenburg and recently acquitted by Uppsala University as well. EKO is developed with Oracle's PL-SQL keeping Oracle Database (ODB) in background. The whole application is running on Oracle Application Server (OAS). EKO has its own reporting system which fulfills almost all the requirements of users. It is also possible to add or modify any input fields by the users.

Currently, the maintenance team is facing many problems when a request of change comes from the user, especially on the report of the EKO. Change is a time consuming and complex process. The functionality of the system is huge, complicated and inter-related. Moreover, out of the main three developers of the system, only one is currently working at Logica. Due to these reasons, it is very difficult to meet any kind of change requests from the customers very quickly. Considering all these issues from both the users' and the developers' sides, Logica has decided to replace the existing reporting module with the best alternative solution.

First goal was to find out the best alternative solution to replace the existing reporting tool. After doing some market research and other project analysis, BIRT has been chosen to use. Traditional BIRT tools are designed to help self-sufficient power users to access, message and present information to management. Today's BIRT technologies are built to serve the needs of power users. These power users comprise less than 5% of the enterprise; therefore BIRT addresses the need of external audiences as poor or not at all. The first step in establishing a successful BIRT strategy is to recognize the need to facilitate the participation of every user. Then comes the task of acknowledging their roles and their analytic skills that influence the types of reporting applications they receive. BIRT tools are designed for power users for writing reports after completing significant training. This limits the widespread adoption of this technology across the enterprise. Current BIRT tools support multiple users and output formats.

As organizations expand their use of mission-critical reporting applications for large user populations, dependability and high performance become crucial business requirements. Information applications serve all users in an enterprise with mission-critical information. So, the platform should be developed to make it dependable, easy to manage, constant 24x7 availability and no single point of failure. An information application solution must be able to

access a variety of real-time data sources including relational databases, enterprise applications, flat files and legacy systems. Traditional BIRT technologies are built around data warehouse projects. These projects add ever-evolving scope, ETL complexities and storage of non-real-time data [2].

The problem with traditional reporting environments is that the ownership of report maintenance ebbs back toward the most skilled users: IT. The result is self-service shelf ware, reports that stop evolving and lose their luster and the return of tension between IT, analysts and the users. Considering the merits of the following iterative report development process that includes IT supplying the basic reporting template, a business user building ad-hoc reports from this template and consumers reviewing the subsequent interactive report:

a) It starts the report definition process by creating simple report object components, such as table designs, chart designs or internal report logic.

b) Reusable report components that are saved as templates or composite designs.

c) Business users then combine these templates with data to assemble interactive business reports that can immediately be deployed to any consumer.

d) As consumers' needs evolve or become clearer, they can make simple changes themselves, such as font, format or grouping adjustments without asking for an entirely new report.

e) In cases where end users seek more sophisticated changes, such as column additions or new charts

f) Users can make these requests to the business user who supplied them with the original.

g) Once made, the report design can be delivered back to the end user for consumption and further refinement.

h) Larger changes, such as new composite graphics, can be added to the same design by it and subsequently be returned to the business users and consumers downstream.

Notice that each participant in this process owns responsibility for only a small part of the design—just their specification—and that the environment they use to interact with the report is appropriate for their skills. Through sharing, each user can focus on delivering his or her report components without getting stuck with the responsibility of maintaining the entire report [3].

Problem Formulation

First problem of this project was to find out the best alternative solution to replace the existing reporting system. As a solution one of the markets available Business Intelligence and Reporting Tools (BIRT) has been chosen. BIRT can reduce the cost and the time needed for required changes. Understanding the application and the users' requirement is the most important aspect while selecting BIRT technologies. The assumptions and technical decisions shape the capabilities, performance, reliability and scalability of products [2]. Traditional Business Intelligence technology is still used by only 15 percent of employees within

enterprises. This technology is rarely used by the organizations' customers and partners. Organizations spend extraordinary amounts of time and energy negotiating reporting requirements among IT, business analysts and end users [3]. Considering all these issues it was quiet difficult to find out the most suitable, cost effective, user friendly BIRT. Chosen BIRT will also have to fulfill all the requirements of the customer and the developers. Finding such Business Intelligence and Reporting tool was one of the most important and difficult problems to be solved.

After choosing the BIRT it was required to consider whether it will really support the existing stand alone system. By looking at the features one may say that this BIRT can be integrated with the existing system. But there are huge differences between theoretical explanations and practical experience. So, the second problem was to design at least one of the reports using the selected tool. After that it was required to examine whether it really works according to the requirements. There are many critical conditions and functionality in the reports that are needed to be implemented. If these conditions and functionality were solved then it can be said that this BIRT can be used to design the required reports.

Designed report was supposed to be accessed from the existing system by passing parameters from the user interface. Hence, the third part was to integrate the designed report with the system. This was the most difficult and ultimate problem that was to be solved in this project.

According the above discussion the following research questions:

- What is the best solution to replace a stand alone reporting system?
- Which BIRT tool is the most suitable to fulfill the customers' and company's requirement?
- Is it possible to integrate a third party Business Intelligence and Reporting Tool (BIRT) with the existing stand alone system?
- How can a third party BIRT be implemented and integrated with a stand alone system to replace the existing reporting system?

Purpose

The purpose of this thesis work was to find out the most effective and efficient BIRT among all available BIRT by comparison of different features, strengths and weaknesses.

By implementing the chosen BIRT, it was desired to find out whether it supports a stand alone system as a third party tools. The main point of this paper was to implement the BIRT with a stand alone system. This implementation will prove that the chosen BIRT can support the system more efficiently and effectively according to users' requirements. Also to prove that it was user friendly and cost effective; also that it enhances the power of the end user in making managerial decisions.

Scopes and Delimitation

The scope of the thesis work was to find out the best Business Intelligence and Reporting Tools by doing market research. Second scope was to Design and develop a prototype of one of the reports by using the chosen BIRT. Last scope was to implement and integrate that report from the existing system by passing users given parameters.

Related Research

Methodology

The work in this thesis was based on a literature study and an empirical study. The purpose of this chapter is to describe the methods used for data collection and finding the answers to the research questions. Due to broad spectrum of the problems covered in the study, combination of both qualitative as well as quantitative approach was used to collect and analyze the data. Maxwell comments that collecting information using a variety of sources is referred to as triangulation in the literature [4]. This strategy actually reduces the risk that conclusions will reflect the limitations of a specific method or source, and allows the researcher to get broad understanding of the issues under investigation.

The thesis was performed by using both of these research approaches in parallel to the same goals but also to answer different dimensions of the same problem.

Qualitative Approach

Qualitative approach explores attitudes, behavior and experiences [5]. The most vital part of the study was to understand the whole process of EKO system and then exploring the complexity of the problem. Moreover, proper understanding of the functionality in the project was essential to comprehend the analysis. For this purpose, direct and thorough information was needed that could be obtained suitably by the methods in qualitative approach.

Due to the unstructured nature of qualitative inquiry, it offers the benefit of flexibility with more open ended research strategy [6]. The main steps of qualitative research strategy are given in figure 2.1

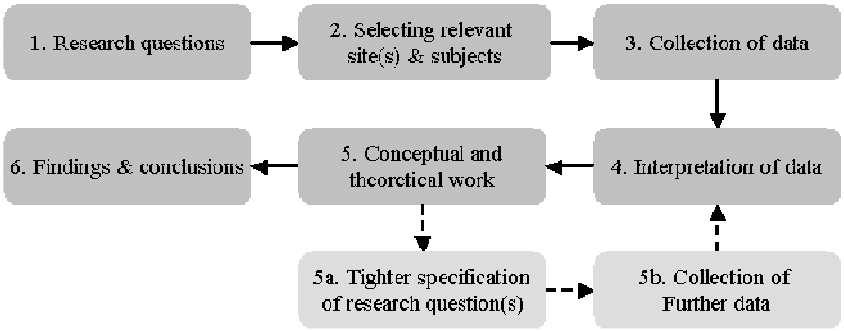


Figure 2.1: An outline of the main steps of qualitative research [7]

The methods chosen for the data collection were semi-structured interviews, documents and literature study, internet and focus group.

Interview

Semi structured interviews were conducted with people of different departments in the organization (see Appendix I). Due to the nature of the research questions, broader view of the subject was required and this method substantially helped for this purpose. Choice of this method was made to get instant and thorough understanding of the system.

Selection of the interviewees was made according to their knowledge in required areas. The project manager, developers of existing system, individuals who are responsible for the maintenance and support were interviewed multiple times. These interviews were done to get information about the features, data flow, related functionality, and database tables. They were interviewed throughout the study and notes were recorded for each of these interviews to learn about the requirements for the new dynamic reporting part i.e. the features, designs, outputs will be there in the reports.

Focus Group

The focus group is a method of interviewing more than one person and emphasizing on a specific topic explored in depth [6]. During this study three focus group interviews were conducted.

First was conducted to get the 'voice of the customer' with two interviewees. Project manager (PM) and one maintenance and support person were invited. Selection of the interviewees was made to clarify what is expected in this study and the reason behind it.

The second focus group consisted of the project manager and the developer of the current system. This was done to understand the system's development and setup environment. This interview helped to select the proper BIRT, which will sustain the environment of the current system. This method also helped to set up the environment for the selected BIRT and to be integrated with the current system. This method helped to get knowledge, which is helpful for BIRT in this project.

The third focus group consisted of project manager and the two individuals from maintenance and support group of the current system to understand the system's data flow, functionality and database.

The reason for conducting focus group was to get a wider range of vision from experts of different domains. In addition, focus groups offer the opportunity to study the ways in which participants create a sense of phenomenon and meaning [6].

Document and Application Study

A variety of documents were studied in the organization to assist the data collection and analysis. Documents including project description, description of the functions, Systemdokumentation EKO 2.7 are among the main list.

Quantitative Approach

Study in this thesis demanded the collection and analysis of different business intelligence and reporting tools to select the most effective and efficient Business Intelligence and Reporting Tool (BIRT). For this purpose, a detail analysis was done of different BIRT on the internet to collect different data. These data were about the detail and different features, strengths and weaknesses of different reporting tools. Main purpose was to find the best BIRT and to investigate whether any third party BIRT can be possible to integrate with the current system.

Benchmarking

Benchmarking is a method of improving performance by comparing your own operations with those of others who perform better in some respects [8]. Moreover, this is a long term process where the focus is not just on what is done best but also on how it is done.

Internal benchmarking was conducted in this study with the existing reporting system along with the EKO. The purpose was to understand the inputs for the report parameters, the flow of the data, and the design of the report. There are three main report categories in the system and each of this has several formats of report views. So, using this method was very easy to understand the design and make the reports according to that.

Another reason of choosing this method was that the existing system is already running successfully. Hence, it did not require too many times or workload to redesign the reports and to ensure whether the generated reports were acceptable or not.

Moreover, this method was applicable to relatively large organizations where different units or departments can be compared and evaluated against each other.

Empirical Study and Analysis

This chapter discusses about the requirement collection and specifications. Here it is discussed the analyzing of the existing Business Intelligence and Reporting Tools (BIRT) and how it has been chosen. It also explains how analyzing of the existing system has done and what the outcome from that analysis was. This study and analysis helped to design and develop reports in the selected BIRT tool and also to implement and integrate it with the system.

Managing the Requirements

The purpose of this section is to clarify the customer's expectations and to show how this study was organized accordingly.

Voice of Customer

Understanding and fulfilling the customer needs was the foremost objective of this study. A focus group with three participants representing the end users and customers was arranged to clarify the expectations and scope of the study. A questionnaire was discussed whose detail along with participant's information is provided in Appendix I. In the light of this discussion, scope and purpose was designed.

There was already an existing reporting system which was supposed to be rebuilt using the most suitable BIRT. On the other hand the existing system was too complex to understand easily, specially the data flows on the reports which got clear after discussing with the customers. Decisions regarding the development environment, BIRT, further improvements in the reports also came out from this discussion.

Creating Requirement Specification

A step by step breakdown of the requirements, decisions about data, analysis required for each requirement and corresponding solution proposals were created as a result of multiple discussions with the stakeholders. This detailed requirement specification explains it in detail and the requirements are presented below:

R1. This project in tales finding and developing a more dynamic and user friendly interface for presenting data according to user specification.
R2. During working on this project it is required to analysis different aspects of the data and the existing software
R3. According to the findings, it is required to design suitable solution for dynamic report viewing
R4. According to the design it is required to make a prototype of the project
R5. The total cost (developer licenses and end user licenses) need to be low, is preferred if it is free. The customers prefer not paying more
R6. The purpose is to see if we can replace current report production in EKO with a more cost efficient way
R7. We also need to have an idea of the amount of time it takes to replace the current code and the amount of time it takes to implement new features and new reports in future
R8. New features should take less time to implement and test that it takes at present

Functional SRS:

R9. Designed Report has function in the EKO environment, which uses Oracle Database 10g (or later), Oracle Application Server 10g (or later), PL-SQL, Java 1.5 (or later)
R10. It is Preferable to be supported by Java and developed by using Eclipse
R11. For the end user there needs to be no reduction of functionality
R12. The selection of the reports and the inputs need to have the same features as the features used currently and should also be handled as it is handled today
R13. The result needs to be the same regarding the content, number of columns and grouping
R14. New features as export to Excel and other programs, layout possibilities etc are required to be implemented

Later during the Project run time:

R15. Implement graphical presentation of the data to test whether it works properly with the system and how it looks
R16. Show details of the data in different places where aggregated functions are implemented

Analyzing Existing Business Intelligence and Reporting Tools

This section is about finding out the best Business Intelligence and Reporting Tool (BIRT) to implement which is my second research question.

There are many BIRT tools available in the market. Some of them are free, some of them are open source and some of them are commercial. It was very time consuming and hard to find out proper information of all reporting tools. It was also hard to do the analysis and find out the most effective and efficient BIRT. Twenty four BIRT tools like Eclipse BIRT Project both by Actuate and Eclipse, JasperReports, Crystal reports, Activate reports, Pentaho, Oracle reporting tools, Agata and so on were observed.

In the middle of the analysis, the customer demanded that the BIRT will have to be free and open source. This helped to narrow down the focus. Subsequently it was only focused on those BIRT which are both open source and free. Four BIRT named Eclipse BIRT project, JasperReports, DataVision Reports and RAQ Report were analyzed in details. It was not that easy to analyze these tools in details as most of them are not well known and widely used. As a result, user comments and different specifications like features, weaknesses, strength, architectures were not that much available. There was also no proper guidance or source of information to find out about these BIRT tools. Only source was internet where there is a lack of information for not well known and not widely used tools.

During the analyzing of these tools it was focused on the architecture of the tools, security, report server, connectivity regarding database, integration, application development, report types, database field types, formatting, grouping and sorting and output types. All these very specific and detailed data were very tough to find out for different tools. Because these tools are neither that much used nor popular. But it was tried to find out as much data as possible. According to that one BIRT was chosen which is the most effective, efficient and fruitful to

implement as third party reporting tool along with the existing stand alone system. Comparison among different tools are given below in Table 3.1

Features	Eclipse BIRT Project	JasperReports	DataVision Reports	RAQ Report
Version	BIRT 2.3.2	JasperReports 3.5.1	DataVision 1.2.0	RAQ Report 4.1.92
1. Architecture				
Platform Support	Microsoft Windows, Linux, Sun SOLARIS, Mac OS X, AIX	Windows, Linux	Microsoft Windows, Linux, Sun SOLARIS, Mac OS X, BSD,	RAQ Report IDE: Windows; RAQ Report Server: Windows, Linux, Solaris, Unix
Client	RCP Designer: for report developers on the Windows platform. Java 2 platform Java Development Kit (JDK) 1.5 Eclipse Platform Runtime Binary 3.4 Graphical Editor Framework (GEF) Runtime 3.2 Eclipse Modeling Framework (EMF) 2.3 Data Tools Platform Project 1.6 (DTP) Web Tools Project (WTP) 3.4	JRE 1.4, JAXP 1.1 XML Parser, JDBC 2.0 Driver, iText - 1.3.1, JExcelApi 2.6, JFreeChart 1.0.0, Jakarta Commons BeanUtils Component 1.7, Jakarta POI 3.0., JCommon	DataVision, iText, the Java-PDF library by Bruno Lowagie, JCalendar, the calendar Swing widget by Kai Toedter, BSF, the Bean Scripting Framework from the Apache Jakarta Project, JRuby, the Java implementation of Ruby by the JRuby Project, and	JRE 1.4, XML, JDBC, iText
Report processing	Report Engine API	JasperServer	JDBC	Report Server
Web Server support	Microsoft IIS 5.0/6.0, Apache 6.x	Microsoft IIS 5.0/6.0, Apache 6.x	Microsoft IIS 5.0/6.0, Apache 6.x	Microsoft IIS 5.0/6.0, Apache 6.x
2. Suggested tools				
Report Development	Design Engine is built in Eclipse BIRT	iReport	DataVision Report Designer	RAQ Report IDE
Report Server	Report Engine API is built it Eclipse BIRT	JasperServer	DataVision Server Application version 6.x	RAQ Report Server
Report publishing	Windows and Web Application	Windows and Web Application	Windows and Web Application	Windows and Web Application
3. Security				
Authentication, Object Security and Data Access Security	Strong; Several Authentication	Medium; User based	Medium; User based	Medium; User based
4. Report Server				
Pblishing	Eclipse BIRT Report Engine	JasperServer	DataVision Server Application	RAQ Report Server

Report scheduling	Possible even without using Apache	Possible using JasperServer	Possible using external and optional tools	Data not Available
Export formats	PDF, Excel, PowerPoint, Word, CSV, HTML, XML	PDF, HTML, XLS, CSV, Word, XML	HTML, XML, PDF, Excel, LaTeX2e	Word, PDF, Excel
Delivery channels	Run and view reports on-screen	Run and view reports on-screen	Run and view reports on-screen	Run and view reports on-screen
Printer selection	Available	Available	Available	Available
5. Connectivity				
SQL Commands	Can be edited	Available	Cannot be edited	Available
Open Connectivity	Available	Available	Available	Available
JavaBeans	Supports	Supports	Does not support	Supports
OLAP Engines	Supports multi dimensional OLAP engines	Supports	Supports	Supports
Native connections	Flat file, Scripted, XML, Oracle, SQL server, MySql, DB2, Java object, CSV, JDBC, POJO, EJB	JDBC, XML, POJO, EJB, MDX, CSV, and custom	Oracle, PostgreSQL, MySQL, Informix, hsqldb, Microsoft Access, DB/2, Progress	Oracle, SQL Server, Sybase, DB2, Informix, XML, file system, Java object
6. Integration				
6.1. Report Development				
Report creation at execution time	creating XML	creating XML	Creating an XML file	creating XML
Report development inside development tool	Design engine of Eclipse BIRT	iReport	DataVision GUI	RAQ Report IDE
Ad-hoc Reporting	Possible	Possible	Not possible	Not possible
6.2. Application Development (Client) (Report Server not required)				
Java	Java SDK supported	Java SDK supported	Java SDK supported	Java SDK supported
API	Design Engine, Report Engine, Chart Engine and many more	Not properly defined. Separate chart engine, report server and report designer used	DataVision GUI, DataVision testdata	Report Server, Report designer
7. Report Types (view object types for details)				
Drill-Down Reports	n depth level	Supports	Does not Support	Not available in free edition

Report with sub reports	n depth level	n level of depth	Available	Available
Cross-tab reports	Available	Available	Does not Support	Available
OLAP reports	Available	Supports	Does not Support	Data not Available
multi-column reports	In details and group	In details and group	Available	Available
Labels	User Defined or standard	User Defined or standard	Standard	Standard
8. Objects				
8.1. Object Explorers				
Field Explorer	Shows all report objects: database fields, formulas, parameters, SQL expressions, totals, and special fields	Shows all report objects	Shows all report objects	Shows all report objects
8.2. Object types				
Graphics	27 types of 2D and 3D charts	12 types of graphical charts	Data Not Found	25 types of graphical charts
Cross reference tables	Fully customizable. Information is showed in rows and columns; rows and columns can be grouped in 0, 1 o n levels. We can also apply templates	Fully customizable. Information is showed in rows and columns; rows and columns can be grouped in 0, 1 o n levels. We can also apply templates	Does not Support	Data not Available
Sub reports	Supports up to n depth level	Supports up to n depth level	Available	Available
OLE Objects	supports OLE objects	supports OLE objects	supports OLE objects	supports OLE objects
Formulas	Many basic mathematical and logic functions are provided	Implemented in JasperReports 3.0.0	Supports	supports
SQL Expressions	Available	Available	Supports but has many bugs	Available
Summary fields or aggregated functions	Possible	Possible	Possible	Possible
Parameters	Both default and dynamic parameter are supported	Both default and dynamic parameter are supported	Both default and dynamic parameter are supported	Built in parameter template; dynamic correlation does not support in free version
Cascading parameters	Supports	Supports	Does not Support	Does not Support

Pictures, lines and figures	Possible to apply condition	Possible to apply condition	Does not Support	Does not Support in free version
User Defined Functions	Possible	Possible	Possible	Possible as Macro is available
Custom Functions	Functions created inside one report (called "internal functions"). Can be exported to the Repository for global use	Functions created inside one report (called "internal functions"). Can be exported to the Repository for global use	Does not Support	Macro
Hyperlinks	Any report object can be linked to an external file (rpt or file system), to a web or e-mail address. Links can be conditional.	Any report object can be linked to an external file (rpt or file system), to a web or e-mail address. Links can be conditional.	Does not Support	Not available in free version
Lists	Available	Available	Does not Support	
Tables or Grids	Available	Available	Does not Support	Available
9. Database Field Types				
Basic Fields (varchar, integer etc)	Total Control	Total Control	Total Control	Total Control
Blob Fields (memo, image etc)	Supports	Supports	Supports	Supports
Stored Procedures	Supports	Does not support directly	Supports	Does not support
10. Formatting				
Fixed and conditional format	Available	Available	Available	Available
Field format	Available	Available	Available	Available
Object Format	Available	Available	Available	Available
Section Format	Available	Available	Available	Available
Highlighting Wizard	Available	Available	Available	Available
Conditional page breaks and sections	Available	Available	Available	Available
Templates	Available	Available	Available	Available
11. Grouping and sorting				
11.1. Grouping				
Hierarchical groups	Possible	Possible	Data Not Available	Possible

Top-N, Bottom-N, Top %, Bottom %	Possible	Possible	Data Not Available	Possible
Specified grouping	Possible	Possible	Possible	Possible
11.2. Sorting				
Group sorting, including Top-N, Bottom-N, Top and Bottom %	Possible	Possible	Possible	Possible
Specified sort	Possible	Possible	Data Not Available	Possible
12. Printing Option				
Local Printing	Possible	Possible	Possible	Possible
Server Printing	Possible	Possible	Not Possible	Not Possible

Table 3.1: Technical Comparison of Eclipse BIRT, JasperReports, DataVision and RAQ Report [22] [48] [49] [50]

Besides doing this detail comparison the weaknesses and strengths of these tools were also looked for, from the internet given by the users and report developers. The following discusses the features, weaknesses, strengths and architectures of these tools.

Eclipse BIRT is an open source reporting tool for web applications supporting Eclipse IDE. Especially for Java and J2EE based systems. It has design engine and report engine to design and run the report. It also has chart engine to support charting to the application.

There are four main parts of Eclipse BIRT. First one is Data which uses Open Data Access (ODA) framework. ODA allows anyone to build new UI. It also a runtime support for any kind of tabular data. Eclipse BIRT supports JDBC, XML, web services and flat files. Second is Data Transforms which supports sorting, summarizing, filtering and grouping of data according to user's requirements. It also allows sophisticated operations such as grouping on sums, percentages of overall totals and more. Third one is Business Logic which supports converting raw data into information which are useful for the user. JavaScript can be used to script the logic or logic can be called in to the existing Java code. Final one is Presentation which supports tables, charts, text and more to present the ready data to the users. A single data set can appear in multiple ways, and a single report can present data from multiple data sets [9] [25].

Strength:

- It is an Open Source reporting tool [9]
- It doesn't require a frame, it can use a reportlet, and it will only render the element in that reportlet [10]
- The chart builder can be used in standalone mode [10]
- Data can also be bound from the DB using `IGenerator.bindData(sql.resultset)` [10]
- BIRT drops into a web application much easier [11]
- BIRT has a nicer (eclipse based) designer, does sub reports better [12]
- Uses JavaScript (Rhino) for scripting, so does not require any compilation from templates [13]

- BIRT has excellent report designer and charting support [13]
- BIRT has great support for reports with multiple data sources [13]

Weakness:

- BIRT is harder to embed in your application [5]
- It is harder to integrate with OpenReports than other engines [5]

Architecture:

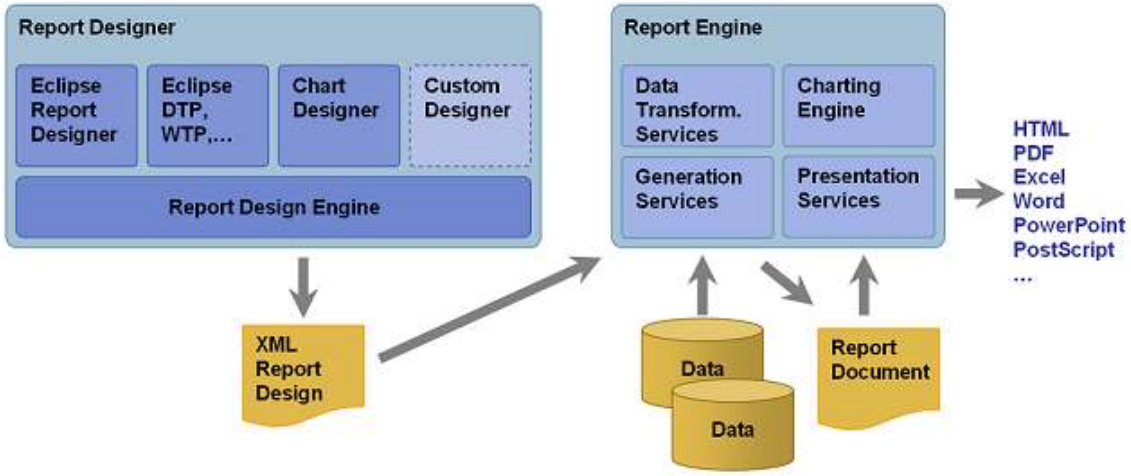


Figure 3.2: BIRT Architecture [2]

JasperReports is open-source reporting tool. By using JasperReports page oriented and ready to print reports can be developed very easily. JasperReports is written in Java and can be embedded in any Java application. It uses separate report engine, chart engine and report designer tool. JasperReports export reports in different formats like PDF, HTML, XLS, CSV, XML and so on [30].

Strength:

- JasperReports has Java report library
- It supports dashboards, tables, cross tabs, and charts
- Pixel-perfect, complex layouts for screen or print
- It supports flexible and extensible data sources along with different types of output formats like PDF, Excel, HTML and so on
- World's largest, most active community of report designers and developers
- This means JasperReports can perform page oriented operations (such as page totals, etc) that are useful in printed output, comes with a page view that can be embedded in Swing apps, etc [2]
- JasperReports gives the user complete control over the data [2]
- Jasper supports free-form layout [14]
- JasperReports has a number of built in functions, including totals, averages, min, max, etc [15]
- JasperReports includes support for the Groovy scripting language, and support for other languages can be easily plugged in [15]
- JasperReports can be imported into JasperServer and managed from there
- Open source, pure Java reporting, for superior portability [2]

- Support for bar codes, charting and paging [2]
- Easy to configure and install [2]

Weakness:

- JasperReports is a library and it uses separate server named JasperServer [2]
- To get complete control over the data have to write more code [2]
- Jasper itself does not offer a designer [14]
- Jasper does not have direct support for charts, though you can integrate the JFree Charts package [14]
- JasperReports in general doesn't support multistage queries [16]
- Stand alone JAVA based IDE [2]
- Not eclipse based [2]
- Limited data access, supports only one data source per report [2]
- Branding, customization and personalization difficult [2]
- Limited documentation and help [2]
- No error checking or debugging functionality [2]
- No page-level security or large-audience features [2]
- Difficult to generate templates outside the normal work flow [2]

Architecture:

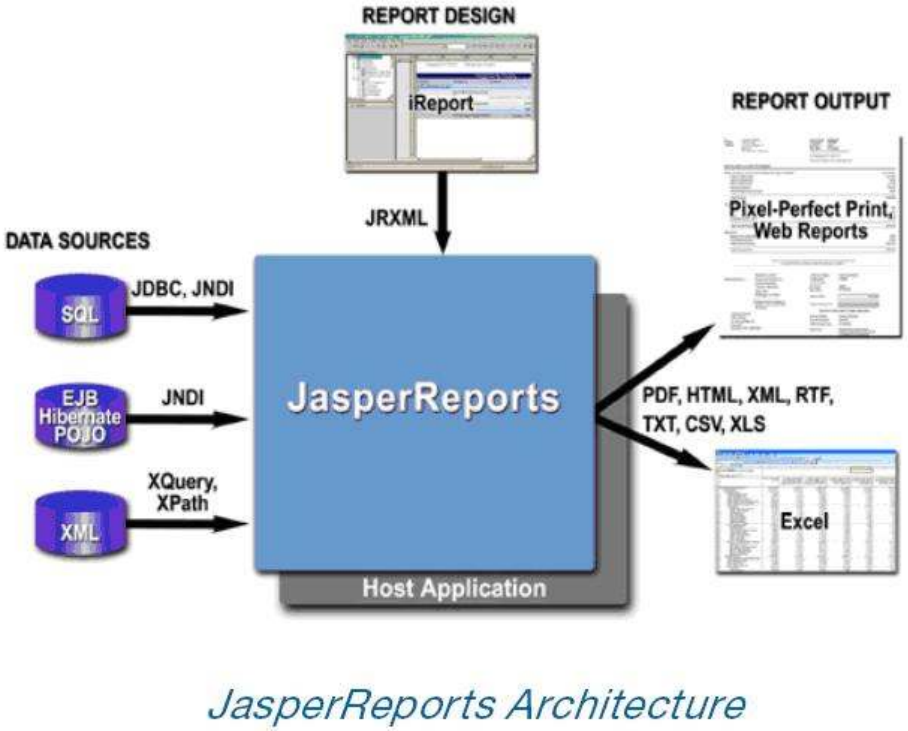


Figure 3.3: JasperReports Architecture [17]

DataVision is another free and open source reporting tool. As it is written in Java, IT supports all environments. It supports both database and text data file to retrieve data. It only supports

JDBC supported databases. The output form of the report is HTML, XML, PDF, Excel, LaTeX2e and Word. It is similar to the crystal reports.

Strengths:

- Supports visual design of simple reports accessing SQL data for end users [47]
- Supports optional output of results to a Web page, PDF file, text file, XML [47]
- Allows Java web applications to run server-based reports as part of applications [47]
- Has a flexible software license agreement [47]
- GUI supports eleven Languages [48]
- Report exports in LaTex format besides others [48]
- Supports both databases and text files to access data [48]

Weakness: [48]

- DataVision depends upon JDBC drivers to communicate with databases. You must have the correct driver for each type of database (Oracle, PostgreSQL, MySQL, hsqldb, ODBC, etc.) that you wish to use.
- Though there is a built-in class path that Java already knows about, that class path does not include the whereabouts of your JDBC drivers or DataVision itself.
- If any of the character of any language does not display properly then you will need to upgrade the version of the Java
- If the layout engine does not support a font then you will not be able to see that font in the report output
- You might get some unnecessary and harmless error messages of fonts in Mac OS X and Linux. But those are fixable if you are the root.
- You cannot select fields by clicking and dragging a selection box with the mouse
- Printing or viewing an exported output file from the command line is system dependent
- There are still many bugs in the sub reports.

RAQ Reports is open source business intelligence and reporting tool for web applications to generate reports.

Strengths: [19]

- No freeze table header
- Does not support enhance printing, hyper links and image uploads
- Does not support drop-down calendar, tree and dynamic correlation

Weaknesses: [19]

- Export to PDF, excel and word files
- Supports two dimensional report, group and cross reports
- Supports preformatted paper printing, style sheets and format painter
- Supports macro, parameter template, chart and page breaking
- Independent multiple data sets and sub report options are available
- Supports windows and Linux servers

Architecture:

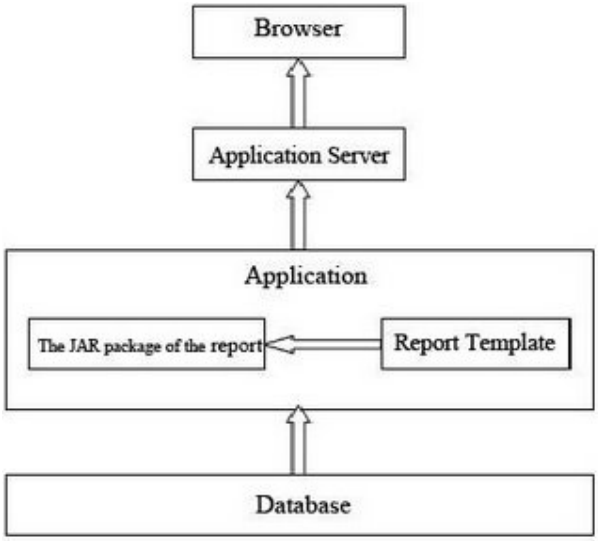


Figure 3.5: RAQ Soft application Architecture [20]

After analyzing all features, weaknesses and strengths of these tools, Eclipse BIRT and JasperReports were found as the two most efficient and effective tools. These tools can be used to implement reports and integrate with the system. Both of the tools has many similarities like both supports scripting language and are java based; supports XML, cross tabs, charts, sub reports and open source. The biggest advantage of Eclipse BIRT is that the report designer and report engine are included with Eclipse IDE. These engines can also be included just by adding some plug-ins to the IDE. The customer also preferred a tool which can be used or be supported by eclipse IDE while developing the reports. Eclipse BIRT is included with report designer, DTP, WTP, chart designer, custom designer, report engine, chart engine, generation and presentation service. Regarding the documentation and help, Eclipse BIRT has their own development community and web site with all necessary documentations and helps. Moreover, they have published two books one is BIRT: A Field Guide to Reporting and another one is Integrating and Extending BIRT. Both of these books cover all the necessary information regarding designing, generating and integrating reports.

In spite of having so many advantages for both tools there are some major disadvantages for JasperReports compared to Eclipse. The biggest disadvantage of JasperReports is that it is stand alone Java based IDE rather than eclipse based. JasperReports is not embedded with report, design and chart engines. It does not have its own report designer. It uses JasperServer as a separate server, JFree chart packages for charting. JasperReports also does not allow multilevel queries. It has less error checking and debugging options and there is no page-level security. JasperReports has very little documentation and help supports.

The outcome from this analysis was

- finding out different BIRT tools
- pointing out the differences among all these reporting tools
- comparing their features with the customers' requirements and
- selecting Eclipse BIRT reporting tool as the most effective and efficient BIRT tool.

Analyzing Existing System

Using benchmark method and having a system has both advantages and disadvantages. Advantage is that it does not require thinking much about the designing of the outputs most of the cases unless users want to change it. Again it can be justified whether the new system is working properly according to the successful benchmark. On the other hand it sometimes very difficult to follow and understand the existing system which is used as benchmark. There are also very little scope to do different things apart from the existing system. It has limitations of redesigning if the users want to do it. It has also limitations of creativity and innovation if the existing system does not fulfill the requirements.

The purpose of this analysis was to find out how the reports have been generated during run time; to find out which database tables, columns and data have been used to generate the reports; to find out what inputs of the reports will take as report parameters, validation and verification of these inputs; to find out the SQL query, conditions and joining in the query; to find out the validations and verifications have done during retrieving data from database tables and to find out other critical functionality to generate different reports.

It was not that easy to analyze the existing reporting system in this study. EKO itself is a huge system with functionality such as inserting, deleting, updating, report generating. It was developed with mixed PL / SQL and dynamic SQL. They are structured into three logical layers that are implemented in two physical layers. After analyzing the system and codes for three weeks thoroughly it was figured out the flow of the functionality and understood how the report is generating and the data are coming from database.

The reporting section has three packages of which each of the packages has several procedures and functions (see Appendix II). Most of the packages are about more than 10000 lines of codes. It was very difficult to find out the proper packages, procedures and functions developed for the reporting system and to understand the code along with the flow of data. All packages were inter-related to each other very closely.

The reports were generating dynamically on run time, having no predefined templates. They had many validations checking. The following describes how it works on the report delkontraktssummor / year. The principle is identical for all reports, except the Excel report described in the next paragraph. `rpt_delktr_sum_form.sok` package and procedural search and present data on the screen. `rpt_delktr_sum_form.ctrl` package and the procedure takes care of the data entered on the screen and validate it before it passed on down to the enforcement authorities. `rpt_delktr_sum_res.build` package handles all the enforcement of data from the database. The issue first built up with dynamic SQL in a single procedure and executed. Depending on the sort order selected for the report, so called after the procedure, which handles the logic of the specific sort order. The package is also building the report with HTML tags and presents it. This package is heavily modularization [21].

There are three kinds of reports in existing system where each of the reports has different formats as output. They are Income/Project and year report, Income/activity and year report and Income/payment plan and year. Income/Project and year report takes ten inputs as report parameters from the users and ten different formats of report outputs. Income/activity and year report takes eleven inputs for only one format of report output. And Income/payment plan and year takes eight parameters for only four different formats of report outputs. Since it

was supposed to develop the prototype of one type of report so Income/Project and year report and Fin.kod/Orgenhet format has been chosen for developing the prototype.

Thorough analysis of the packages, procedure and functions for this report was done to understand the work flow. This report format has used `sok` procedure from `EKOTRUNK.Rpt_Delktr_Sum_Form` package for the main report input page. Procedure `ctrl`, `validate`, `build`, `finansiarstyp_kst`, `init_arrays` were used to generate the main report. `get_column_values`, `make_sel_param` functions were also used to generate the main report. `EKOTRUNK.Rpt_Delktr_Sum_Res` package was used to read the parameters from the report input page. All these packages, procedures and functions are given in Appendix II.

The outcome from this analysis was

- the understanding of the existing system
- understanding how the reports have been designed and generated during run time
- how the critical validations and verifications are done while showing the data in the reports
- understanding which database tables, columns and data are used for the reports
- understanding different functionality which are required for reports and
- understanding the data flow of the reports.

Implementation

In this chapter it has been discussed about the implementation of Eclipse BIRT reporting tool. It is also discussed about the deployment setup, report designing and development, Graphical User Interface (GUI) designing, report integrating. Finally, it has been discussed about different problems that were faced during different stages of the development and their solutions.

Environment Setup

Setting up the environment for both development and deployment is at times very easy and others very difficult. It is easy when you know what to do and if you have all the necessary components and documentation to set up. On the other hand, it's very difficult because for a very small mistake you have to suffer a lot to fix that problem. In this study it was to fix the development environment for Eclipse BIRT (EB). For this project EB 2.3.2 All-in-all was employed. This BIRT project includes different frameworks like DTP 1.6.2, EMF 2.4.2+XSD, GEF 3.4.2 runtime and WTP 3.0.4 along with Eclipse 3.4.2 SDK. It was also possible just attaching mentioned frameworks with the Eclipse IDE. As Java 1.5 JDK/JRE is the pre-requisite to use EB, so the JDK was installed first. In addition to the installation of EB web tool project was also installed which gives the option to build reports for web application easily. Orai18n.jar and classes12_g.jar JAR files were attached to get JDBC connection with the databases. To implement the GUI as servlet the JAR file named javax.servlet.jar was also included [22].

On the other hand the deployment environment was also set up. Apache Tomcat v5.5 with the port number 8008 was installed to deploy the reports to run from the web application. For EB, It was also required to deploy the BIRT viewer into the webapps directory of Apache tomcat. First, EB report engine runtime version 2.3.1 was downloaded. Inside the birt-runtime directory there was a folder named Web Viewer Example which was deployed in to the webapps directory of Apache. That folder was renamed as "birt-viewer" (recommended). It also needed to adjoin `itext`-1.3.jar, iTextAsian.jar and JDBC drivers inside the birt viewer to access the databases during runtime from the web application [23].

As the project was done on the existing database so it was not necessary to configure the environment for the database. The existing system is using Oracle database 10g. The web application for the existing reporting system is deployed in with Oracle Application Server (OAS). During setting up the deployment environment many problems with Apache and OAS were faced. These problems and their solutions are discussed in section 4.5.

Designing Report

As this study was about to develop reports with the Eclipse BIRT (EB) so designing the report was one of the major and critical work. It was supposed to design the report in such a way which will fulfill all features of the current reports along with additional user requirements. It was also supposed to consider simplicity, usability, and cost effectiveness of newly developed report. In this study designing new reports was easy as there was the existing system as benchmark. The output report format of the existing system helped to develop the skeleton of the report in the EB. Moreover, EB has a very efficient Design Engine API plugged in with it. By using this API, it is very easy to design the report and execute them.

First of all it is required to switch to Report Design perspective in eclipse IDE to design and develop the report. Then the connection with the database using JDBC driver was created and generated some datasets of required database tables. One dataset named EKOTRUNK with the "master query" was constructed where all the data are retrieved from different database tables to be shown in the report. "Master query" in this case number or rows, column and data to be shown in the report depends on this query. Master query dynamically changes during the runtime of the report just before opening the EKOTRUNK dataset. This dynamic change in query has been done by implementing scripts on the dataset using JavaScript. All the validation and verification for the query are done in this script written for the dataset. The master query (copyrighted by Logica; not for use) is as follows:

```
select t180.kontrakt$id, t180.andamal$id, t210.ar, nvl(trim(t210.belopp),
0) BELOPP, t210.varav_natura, t250.beskrivning, t110.kst, t110.beskrivning
t110_beskrivning, t140.id, t160.nr, t160.finansiarsgrupp$nr,
t160.beskrivning t160_BESKRIVNING, sum(t420.inbetalt_belopp)

from t210_delkontrakts_period t210, t250_sektion t250, t110_kostnadsstalle
t110, t160_finansiarstyp t160, t140_finansiar t140, t100_kontrakt t100,
t180_delkontrakt t180, t420_rekv_fakt t420

where       t180.kontrakt$id = t100.id AND
            t180.finansiar$id = t140.id AND
            t140.finansiarstyp$nr = t160.nr AND
            t180.kostnadsstalle$kst = t110.kst AND
            t110.sektion$nr = t250.nr AND
            t180.kontrakt$id = t210.kontrakt$id AND
            t180.id = t210.delkontrakt$id AND
            t420.avser_ar (+) = t210.ar AND
            t420.KONTRAKT$ID (+) = t210.KONTRAKT$ID AND
            t420.delkontrakt$id (+) = t210.delkontrakt$id


group by    t180.kontrakt$id, t180.andamal$id, t210.ar, t210.belopp,
t210.varav_natura, t250.beskrivning, t110.kst, t110.beskrivning, t140.id,
t160.nr, t160.beskrivning, t160.finansiarsgrupp$nr, t180.projekt$nr

order by t160.nr, t110.kst, t210.ar
```

EB has a Palette where different report components are given to deploy in the report. These report components are table, cross table, grid, list, chart, data, label, aggregation and more. Then the template of the report was designed just by dragging and dropping different report components on the template. Initially, it was started with blank template then put in a table which gives the option to have header, footer and detail rows separately. Then different dataset rows and labels were inserted in different places according to the existing system's report format.

In the report it was supposed to summarize the amount in two places. One was based on the "financial code" to summarize the amount at the end of each financial code. This "financial code" is a parameter given by users. Another one was the "grand total" to summarize the whole amount at the end of the table. This summarizing and grouping was very easy to add to the template by just dropping the aggregation module on the template. The first grouping was done on the financial code and then for the grand total the grouping was done on the table.

In the report there were few columns which were supposed to be shown up on the report according the use inputs. For example, starting year and ending year were obvious to show on the report but if the range of the years is more than two similar data of five years will be shown then it will append additional columns during runtime. For the prototype these columns were implemented for up to ten years. These columns for years were hidden according to some conditions on the runtime checking. Ändamål (group of different sources of money) columns for each year also implemented the same way like hiding the columns unless users want them to be shown.

All the report parameter fields were also created and defined that will be utilized in the report. These fields will take inputs from the Graphical User Interface (GUI). CSS could be implemented for the outlook of the report but as it was a prototype and time shortage did not enable to consider it for the time being.

After implementing the report till this point, the company added two more requirements. First one was to implement charts. This was not in the requirement specification initially but it was proposed to implement. But after first demonstration of the project they agreed to put into operation it. The main target of implementing chart was to test whether it is really possible to implement charts according to users need or not. The customer also wanted to see how it works out with the report. Estimating the time to implement charts from the scratch was another target.

Three charts were put into service. Two of them were bar charts and one was area chart. First bar chart was designed and developed based on Totalt Belopp (Total Amount) vs. Finansiärskod (name of the companies or departments) grouped on Finansiär (name of the groups). Second bar chart was designed and developed based on Totalt Belopp (Total Amount) vs. ÅR (year) grouped on Finansiärskod (name of the companies or departments). The area chart was also designed and developed based on Totalt Belopp (Total Amount) vs. ÅR grouped on Finansiärskod (name of the companies or departments). Bar charts were made as 2D chart and the area chart as 3D. Both the bar chart (second one) and the area chart were made with the same data to show different looks of the data using different templates of charts. It was easy to implement the charts as EB has a very efficient Chart engine as plug in.

Chart Engine API is embedded with the EB which is very efficient to develop chart more easily and effectively. This API has a very user friendly interface to define different settings for the chart, though some problems were faced to figure out how to set the values in it in the in the beginning. There were three options: one was to select the chart type like bar, pie, line, cone and so on; Second option was to select the vales for X and Y axis. From this option data could be selected from the data sets or the tables. There was another option to group the data on the Y axis. Then the third option was to format the charts where the chart areas, title, plot, legend, axes could be formatted like the label, font size, font color, legend color, border and so on. It was very user friendly to implement a chart using almost all the features. To implement a chart it took approximately 20 minutes including selecting chart type, values and formatting.

Second requirement was to estimate the time to append an additional field to the system or to put into operation a new report. It has been discussed in section 4.6

Designing Graphical User Interface

Regarding the Graphical User Interface (GUI) there was no specific requirements from the users. In this project main focus was on the implementation of the report rather than beautifying the GUI. But as the target was to develop the whole project in java so the GUI was redesigned and implemented in Java. JSP was used to implement the GUI where the user gives the inputs and from that page all data goes to the report page as parameters and the report generates.

I was new at JSP so I was not sure whether it will work properly with the Eclipse BIRT (EB) or not. That is why before integrating with the developed report; the GUI was developed with JSP along with two java class files to handle the inputs and one more JSP page to show the output. `Index.jsp` file was the main GUI where it was implemented the form to submit the inputs to generate the report. CSS styles were also added to give it a better look. `EkoController.java` was the java class file which was supposed to read the parameters and send them to `FormBean.java` class file. `FormBean.java` was used to as bean file to handle all the Gets and Sets functions for the inputs. Then finally the outputs were sent to `success.jsp` file. Like the `index.jsp` file other three files were not used in the integration part as it was not required. It was just implemented to make sure that it is getting data properly and newly GUI is working perfectly. All the files are available in Appendix II

Designing of the template of the GUI was not that difficult. The template was same as it is in the existing system. So the benchmark was just followed. To do the designing and developing the GUI, Macromedia Dreamweaver 8 was used. It was easy to design using this tool rather than doing it by coding.

One of the main reasons why it did not focus on the GUI because in the long run they will integrate the reporting system with the existing system's GUI. In that case it was kind of waste of time to focus on looks and design of the new GUI. Moreover, according to Logica the end users are fairly satisfied with the looks and usability of the GUI.

Integration of GUI and Report

This section answers last two questions of my research: is it possible to integrate a third party Business Intelligence and Reporting Tool (BIRT) with the existing stand alone system and how.

Integrating developed report with the GUI was the one of the most difficult works of the project. It was emphasized more on integration than adding extra features besides the basic requirements. Initially lots of problems were faced to integration as that much examples could not be found and facts how to integrate. Most of the documents that has been found were about implementing the report engine to able to integrate with the report from the GUI.

Eclipse BIRT (EB) reports can be run directly from Java code by using Report Engine API. This API is essential to integrate reports into a stand alone java application and also to deploy as part of a servlet. It also allows integrating the run-time part of EB into the application. The engine reads the set of parameters defined for a report, run the report. It also creates the report as HTML, Word, Excel, PDF and CSV output. This API also fetches images and charts in the

report. The report engine supports extensions for custom report items, data access, and custom output formats.

Initially, it was focused on how the application integrates with the report engine and then it generates the report through the engine for the user. The first step was to download and install plug-in of the report engine. Then it was put in the JDBC drivers to the `ReportEngineInstall/birt-runtime-version/ReportEngine/plugins/org.eclipse.birt.report.data.oda.jdbc_version/drivers` directory. This configuration is done to access the database during the runtime of the application. The following key steps are required to use the API:

- Creating an instance of `EngineConfig` to set options for the report engine
- Setting the Engine Home and starting the Platform
- Creating an instance of the `ReportEngine` class. This object can be used to perform multiple tasks
- Opening a report design using one of the `openReport()` methods of `ReportEngine`
- Information about report parameters are obtained using `IGetParameterDefinitionTask`. The `IRunTask` and the `IRunAndRenderTask` interfaces have methods for setting parameter values
- Running and rendered the report using `IrunAndRenderReportTask` or `IRunTask` and `IRenderTask`
- When done, calling `shutdown()` on engine instance [44]

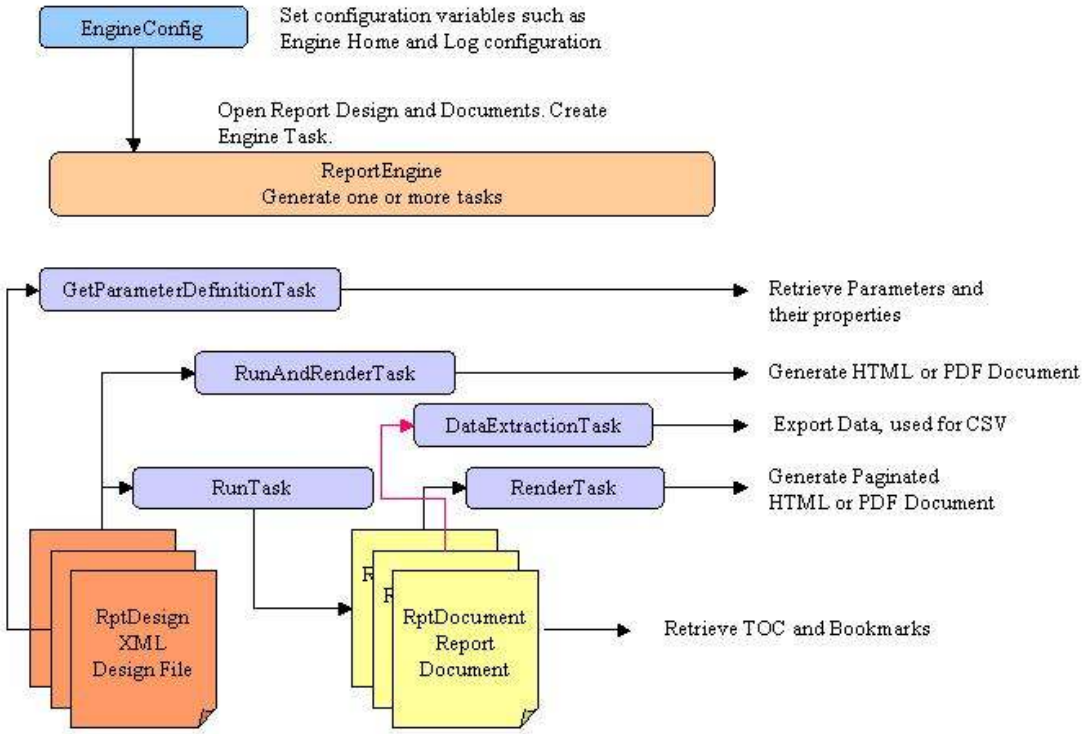


Figure 4.4.1: Overview of the classes needed to accomplish a given task [44]

`EngineConfig` class is used to set global options for the report engine as a whole. It also specifies the location of engine plug-ins, the location of data drivers, and adds application-wide scriptable objects. The engine home is set to `installedlocation/birt-runtime-version/ReportEngine` when deploying it as a stand alone Java Application.
The `ReportEngine` class represents the BIRT Report Engine. There is a significant cost associated with creating an engine instance, due primarily to the cost of loading extensions. Therefore, each application created just one `ReportEngine` instance and used it to run multiple reports. The report engine is created through a factory supplied by the Platform. Before creating the engine, it starts the Platform, which loads the appropriate plug-ins. This is done by calling `Platform.startup(config)` that takes an `EngineConfig` object as argument. After using the engine, `Plaform.shutdown()` function `is called` to do clean up work, which includes unloading the extensions [44].

EB report designs are stored as XML files. By default the extension is `.rptdesign`. The report is first loaded using one of the `openDesign()` methods in the `ReportEngine` class to be worked with the report design in the engine The report design opened methods return a `IreportRunnable` instance that represents the engine's view of the report design. Note that EB supplies many options for opening reports such as the file name or an input stream. EB optionally can store reports in an intermediate format, after generation and before rendering `.rptdocument`, can be manipulated with the `IReportDocument` interface. The Engine creates this document when the `runTask` is used. The EB viewer uses this format to do pagination, TOCs, CSV extraction, bookmarks, etc. When finishes with an IReportDocument, it is closed with the `close()` method. The Engine's `openDocument` method returns an `IreportDocument` that represents the intermediate report document.

EB tasks provide the framework for managing the scripting context, report locales and so on. In general, if an operation requires neither a script context nor a locale, it will appear as a method on the engine or the report design. However, if the operation does require these items, then a task class represents the operation. Tasks are created to support the scripting using the factory methods on the `ReportEngine` class. These are

- `engine.createDataExtractionTask();`
- `engine.createGetParameterDefinitionTask();`
- `engine.createRenderTask();`
- `engine.createRunTask();`
- `engine.createRunAndRenderTask();`

`IDataExtractionTask` class was used to extract report data into CSV format. This class supports extracting data from the report document by specifying the result set and columns that have to be extracted using `dataTask.selectResultSet(resultSetName); dataTask.selectColumns(columnNames); dataTask.setLocale(locale);`

`IGetParameterDefinitionTask` class is used to obtain information about parameters. The `IparameterGroupDefn` and `IscalarParameterDefn` interfaces provide information about parameter groups and individual parameters.

`IRenderTask` class is used to render a report document to a specific output (eg, HTML, PDF etc.). To use this class it is required that the document to should exist, which means it has

been generated with the `RunTask` engine task. This class renders the report, based on the supplied page range, page number or all if no page is specified [44].

`IRunTask` class is used to run a report and generate a report document, which is saved to disk. The report document is used with the `IrenderTask` to support features such as paging.

`RunAndRenderTask` class is used to run a report and output it to one of the supported output formats. The `IrunAndRenderReportTask` takes parameter values as a `HashMap`. The `IrunAndRenderReportTask` provides the `validateParameters()` method to validate the parameter values before it runs the report [44].

All these classes are implemented in the BIRT Report Engine API which is a plug in of Eclipse BIRT. Initially it was thought that it will have to be implemented all these classes and will have to develop a report engine API. But later after doing many research on this engine it was found out that Report engine API is included with the BIRT and it was just required to add the plug-in to generate the report from the GUI by passing parameters. For this purpose it was just required to give some additional report param Tag Attributes like `parameterPage id, reportDesign, name, pattern, height, width, format, title, showTitle, showToolBar, showNavigationBar` and etc along with the report design page name. These parameters are used to inform the engine how the report should be generated and which output formats will be available and so on. There are also other attributes to integrate and generate the report from the GUI named the viewer Tag Attributes, The report Tag Attributes, The parameterPage Tag Attributes and The paramDef Tag Attributes. In the reference number [51] the list of all Tag Attributes and examples for different engine classes are provided. As the integration was successfully completed and worked so it can easily be said that it is possible to integrate the BIRT with the stand alone system and the way to do this is to using Eclipse BIRT report engine API [44] [51].

Evaluation

During the whole project work several problems were faced which is natural for a research work. On the side, it was also tried to solve all of them. Some of the problems were small and some of them were critical and time consuming. First problem was to find out the best suitable BIRT tool for this project to implement. As mentioned earlier, there are many tools and finding out the best tool among them was really difficult. But then the best solution was found which was narrowing down the scope just to Java based open source and free tools. It helped to safe more time and easy to find out the best tool.

Secondly, problems were faced while going through the existing system to understand the whole functionality of it. First problem was that to understand the PL/SQL codes. There were too many packages, procedure and functions and most of them are big and written in a complex way. It seemed more complex as most of them were interrelated and to track them was quiet hard. On the other hand there was not much support from the company as the developers of the system left the company, due to the business for different projects and lack of proper knowledge. Due to having some knowledge in PL/SQL in advance it was not that hard to understand and track the codes. Moreover, full support was got from the supervisor and he did his best to make everything clear about the functionality and codes.

Thirdly, the problem was faced when setting up the environment for development. Initially when the GUI was developed, was trying to run it on OAS 10g along with JDK 1.5. But it

was giving Java runtime exception which was like it cannot run the page. Then After doing long time research it was found that OAS 10g and JDK 1.5 are not compatible together. Either it needs to increase the version of OAS to version 11i or downgrade the JDK to version 1.3. This solution was not that much suitable for the project as JDK 1.3 is too old to support all features and OAS 11i was too expensive for the customers. Then finally OAS 10g was replaced with Apache Tomcat 6.0. When the GUI was tested which was developed in JSP it was working perfectly on Tomcat 6.0. Afterward when it tried to test reports on Tomcat 6.0 it did not work for some unidentified reasons. Then it was downgraded to Tomcat 5.5 and both the GUI and the report files worked properly. So, the best solution to tackle this problem was to use Apache Tomcat 5.5 web server along with Oracle Database 10g. But OAS 10g was used later just for the database.

Fourthly, the problem regarding the design and output of the reports was faced. All the parameters given to the report as input are used as the condition for the SQL query. On the other hand all the parameters were not given or will not be given all together any time to run the report. The problem faced was that it needs to keep blank the condition for the query as the query was made static. Then as a solution "LIKE" was used in the condition of the query rather than using "=". Because by using "LIKE" keyword and "%" keyword as input it is possible to get all the data if there is no specific condition given. But that was not as effective and efficient. Lastly, the best solution was found which was to develop java script for the query to be generated on run time. This script will run just before creating the master dataset and will go through required verification and validation and will generate the query according to the given parameters.

Finally, The most critical and time consuming problem was to integrate the system to developed reports. It took majority of the time allotted for the project work to find the solution. Initially, by reading different references from the web sites and books it was thought that it will have to develop a report engine using several classes. It was even tried to develop one engine but was very confusing about how should it be developed. Then at last the solution was that report engine is already given with Eclipse BIRT. It was just required to install that plug in. In addition it was needed to pass some special parameters to the report engine from the GUI to run the reports. Although the solution was easy, the problem was critical and the best thing was that the total structural and functional knowledge of the BIRT report engine was gathered.

These were the main problems were faced and the most effective solutions were found. There were also many small and less critical problems during the project work and those were also solved. Having many small, big, critical, non critical problems were challenging in every step of this work and best thing is to accept those challenges and solved.

Estimation of time to develop

It was asked to estimate how much time it might take to add an additional field to the system or to put into operation a new report. When the project has started it was not possible to estimate the time as it was a very new this tool and had almost no idea about it. But after completing the report, the time was estimated to build up a new report or different formats of a report based on the knowledge. To develop a report from the scratch like Income/Project and year report, it will take more or less 85 hours including analysis and designing the wire-frame of the report, developing the GUI of the report and integrating the report with the GUI.

The analysis includes finding out the datasets, parameters, conditions, validations and verifications for the report. The division of the time estimation is given in the following table.

Section	Time (Minutes)	Notes
Analysis and Designing in wire frame	600	
Data Source	10	
Data Sets (General)	96	There are 8 general data sets
Data Set (Master)	30	
Report Parameters	154	22 report parameters can be taken by the report in different times
Basic Layout of the Report	60	Basic Layout as the skull of the report table except the columns and the headings of the report page, coloring, formatting, CSS adding.
Column Design	3660	There are 122 columns included in the report but most of the columns are almost same like Ändamål (group of different sources of money) columns with different data and conditions for different years
Script writing	60	This script is written just for the master dataset to make the query dynamic basis on different conditions
Chart implementation	60	Three different charts implemented and each of them took 20 minutes
Static GUI Design	300	Designing, formatting, coloring
Integration	30	
Total estimated time in Minutes	**5060**	
Total estimated time in Hours	**84.33**	

Table 4.6.1: Time estimation for a report to develop from scratch

On the other hand if the other report formats for Income/Project and year report will be developed then the time estimation of the GUI design, Data Source, Data sets, Report parameters and script writing can be omitted as they will be already developed by that time. In that case it will take approximately 29 hours. But if there are any changes in these sections or need any extension to datasets and in other sections then that time should be additional. The division of the time estimation is given in the following table.

Section	Time (Minutes)	Notes
Analysis and Designing in wire frame	100	Will just need to analysis the format of the report and data fields
Basic Layout of the Report	60	Basic Layout of the report like the headings of the report page, grouping, sorting, coloring, formatting, CSS adding.
Column Design	1464	Here time for designing the column has been estimated less as the fields will only be rearranged and the conditions rather than adding the columns in addition
Chart implementation	60	Three different charts implemented and each of them took 20 minutes

Integration	30	
Total estimated time in Minutes	**1714**	
Total estimated time in Hours	**28.57**	

Table 4.6.2: Time estimation to develop other report formats

It was also asked to estimate the time to add one column with data field in the report. It will depend on whether any new dataset will be required to add to the report or any data field from the existing dataset will be used in the column. If it requires adding new data set or changing in the master dataset to the report and apply all the conditions and insert the column to the report then it will take approximately 1.5 hours. On the other hand if it does not require creating new dataset or changing in any dataset then it will take almost 1.33 hours. More over these hours will vary depending on other issues like scripting, validations and verifications and so on which may append more 0.5 to 1 hour. The time estimation for one column is given bellow.

Section	Time (Minutes)	Notes
Analysis and Designing in wire frame	20	
Data Sets	15	either new dataset or changing in master dataset
Column Design	30	
Integration	30	
Total estimated time in Minutes	**95**	
Total estimated time in Hours	**1.58**	

Table 4.6.3: Time estimation to add one column with data field (s)

Discussion

Researchers usually tend to forget that it is time consuming and difficult to collect the proper data and measures for analysis. This study covered wide spectrum of different issues related to the problem where few issues are not investigated thoroughly. However, these issues can be further investigated in the future as an improvement work.

Due to lack of proper information on the different Business Intelligence and Reporting Tool (BIRT) it was very hard to analyze and find out the best one. Though there were few user reviews, examples on the Eclipse BIRT to find out different solutions for different problem, though finally two books named BIRT: A Field Guide to Reporting and Integrating and Extending BIRT helped a lot to design the report more effectively and run the report by integrating with the GUI.

Eclipse BIRT has design engine API which makes the report designing very easy and user friendly. It also helps to focus on the usability along with fulfilling user requirements. It has a very good report engine API which makes the integration of designed report with any stand alone system very easy and effective. It also has the chart engine API as plug-ins which helps the user to design different charts and also run them according to the requirements. Another usability feature of Eclipse BIRT is that user does not require to re-run the GUI to generate the report with different parameters. It can be done from the report page on the run time.

The developed report fulfilled almost all the requirements of the users and it runs efficiently. It was tested by the users and also provided very satisfactory feedback. There was one demo session with the customer just before the integration which helped to correct some functionality. It also helped to add some more features as user requirements on the system. The second presentation was done after the integration which satisfied the users with some additional future improvement suggestions.

As this was a prototype project there were some functions and features which are kept static for the time being in order to check whether it will work on real time or not. All the drop down lists are kept static where it is dynamic in the real system, no CSS has been added to the report template, columns for the years are kept static and limited to ten and also same for the Ändamål (group of different sources of money) columns.

Future Research and Improvements

The developed report was a prototype to test and make sure that the third party tool Eclipse BIRT can be used to enhance the usability of the reports and can also be integrated with the existing system. Naturally, there are many scopes of future improvements by finding out different user requirements. Possible improvements are adding CSS to the report templates so that the end users can change the outlook when ever they want. All static data fields can be made to dynamic. Year columns need to be dynamic according to the user's input and also the Ändamål (group of different sources of money) columns. Designing and developing other reports and integrate them with the system according to their different sections. There are also scopes to include the Charts and other useful functionality and features like sub reports for the summarized fields where the amounts of the summarized field will be shown. Extra text box can be added to make it more flexible for the users to input the amount which will be use to divide different amounts.

Eclipse BIRT is an open source java based BIRT tool which has a dedicated development community. There are many scopes in future to develop the design engine API which can make the designing of the report more users friendly, effective and efficient by adding more functionality. There is also scope to improve the report engine API to run the report with less specified report parameters. Chart engine API can also be improved to support different more types of bar, pie and line charts and also different types of other charts. Designing the tables for the reports can be more improved by making them dynamic on run time without the help of user written java scripts. Dynamic option should be more implemented as different palates in the report designing than just making them by writing java scripts. There are many scopes to research and improve the features and functionality of BIRT by getting different feedback from the users. Because, different users face different problems and they want those problems to be solved very easily than just writing codes.

In the middle of the project it was thought to implement the same reports using JasperReports. By implementing the reports using both tools it would be possible to compare which one performs better. It would also be possible to proof why Eclipse BIRT is better than JasperReports. So, in future this task can be done.

Conclusions

Eclipse BIRT is an effective business intelligence tool with user friendly attributes. It serves in developing effective and efficient dynamic reports according to the requirements of the user. It helps the user make reports accurately and efficiently in short time period because of its Report Engine, Design Engine and Chart Engine APIs. The open source quality of the software enables it to be updated regularly, which can enhance the reports and make them more effective. To put it more effectively, the third party tool namely eclipse birt fulfilled the customer's requirement and met the user satisfaction. In addition, charts were developed for testing purpose to further enhance the software according to the end user's needs that can be implemented in the real system.

Glossary

Tools

Apache

Apache is a free and open source licensed web server. Most of the Unix-based operating systems, Windows 2000 and XP supports Apache version 2.0. According to a web server survey 60% of all web sites are using Apache web servers. Apache is developed by the software development group named "Apache Group" [37].

CSV

Comma Separated Values (CSV) is a common file type used to import data from one software application to another, with commas separating the values in each field. It is often used to transfer data between databases or spreadsheet tables [41].

DTP

The Data Tools Platform project is an open source project of eclipse.org, overseen by a Project Management Committee (PMC) and project leaders. Data Tools Platform is a vast domain. A developer is interested in an environment that is easy to configure, one in which the challenges of application development are due to the problem domain, not the complexity of the tools employed. Data management, whether by a developer working on an application, or an administrator maintaining or monitoring a production system, should also provide a consistent, highly usable environment that works well with associated technologies.

Such an environment starts with key frameworks designed both for use and extensibility. Examples include location and management of data source drivers, and configurations for access to particular data source instances. Once a connection is successfully made, the next task often is to explore the data source, making changes as required. Some of these operations might be carried out by GUI actions, others directly through commands [28].

Eclipse BIRT

Eclipse BIRT is an open source reporting tool for web applications supporting Eclipse IDE. Especially for Java and J2EE based systems. It has design engine and report engine to design and run the report. It also has chart engine to support charting to the application.

Currently there are two releases of Eclipse BIRT. One is 2.2.2 which is included in most commercials and another one is 2.3.2 the latest. By using Eclipse BIRT a rich variety of reports can be added to the application like Lists, Charts, Cross tabs, Letters & Documents, and Compound Reports etc.

There are four main parts of Eclipse BIRT. First one is Data which uses Open Data Access (ODA) framework. ODA allows anyone to build new UI. It is also a runtime support for any kind of tabular data. Eclipse BIRT supports JDBC, XML, web services and flat files. Second is Data Transforms which supports sorting, summarizing, filtering and grouping of data according to user's requirements. It also allows sophisticated operations such as grouping on sums, percentages of overall totals and more. Third one is Business Logic which supports converting raw data into information which are useful for the user. JavaScript can be used to script the logic or logic can be called in to the existing Java code. Final one is Presentation which supports tables, charts, text and more to present the ready data to the users. A single

data set can appear in multiple ways, and a single report can present data from multiple data sets [9] [25].

EKO

EKO is a solution to handle and organize all external financial contributions for Chalmers University of Technology, University of Gothenburg and now recently also Uppsala University. EKO is developed with Oracles PL-SQL that requires Oracle DB and are running on Oracle Application Server (OAS).

EKO utilizes a web application to present data from the database. The report in EKO is very static and requires code changes to change the view according to the users needs. Each report has a number of different selections that can be combined and the user can chose how the result will be ordered. EKO has five different reports.

EMF

To build tools and other application it is required to model the framework and to generate codes. This facility is given to an application by the Eclipse Modeling Framework (EMF). EMF provides tools and runtime support to produce a set of Java classes for the model, along with a set of adapter classes that enable viewing and command-based editing of the model, and a basic editor. It also includes XML Schema Definition (XSD) which is a component of Model Development Tools (MDT) project [27].

GEF

The Graphical Editing Framework (GEF) allows developers to create a rich graphical editor from an existing application model. There are two plug-ins of GEF. The `org.eclipse.draw2d` plug-in provides a layout and rendering toolkit for displaying graphics. The developers can then take advantage of the many common operations provided in GEF and/or extend them for the specific domain. GEF also uses MVC (model-view-controller) architecture to enable simple changes to apply to the model from the view [26].

HTML

HTML (Hypertext Markup Language) is the set of markup symbols or codes inserted in a file intended for display on a World Wide Web browser page. The markup tells the Web browser how to display a Web page's words and images for the user. Each individual markup code is referred to as an element.

The current version of HTML is HTML 4.0. Web developers using the more advanced features of HTML 4 may have to design pages for both browsers and send out the appropriate version to a user. Significant features in HTML 4 are sometimes described in general as dynamic HTML. This is sometimes referred to as HTML 5 is an extensible form of HTML called Extensible Hypertext Markup Language (XHTML) [39].

JAR

Java Archive (JAR) is a file format which bundles all components required by a Java applet. JAR files create package of .class files, images, sounds, etc into one file. It also supports data compression. So, it is very easy to download as well. Extension of Jar files are `.jar` [34].

JasperReports

JasperReports is open-source reporting tool. By using JasperReports page oriented and ready to print reports can be developed very easily. JasperReports is written in Java and can be embedded in any Java application. It uses separate report engine, chart engine and report designer tool. JasperReports export reports in different formats like PDF, HTML, XLS, CSV, and XML and so on [30].

JasperServer

JasperServer is part of Business Intelligence (BI) family to support JasperReports. It is the report server to generate reports developed in JasperReports. It provides robust static and interactive reporting, report server, and data analysis capabilities. JasperAnalysis also runs seamlessly on JasperServer [31].

JDBC

To access databases from Java programs JDBC is used. Java database connectivity (JDBC) is a standard application programming interface (API). All the interfaces and classes of JDBC are written in Java. These standard interfaces and classes are used to connect to databases, write queries in SQL, and process the results. JDBC API is similar to the core Java interfaces and classes like `java.lang` and `java.awt` [35].

JDK

JDK or Java Development Kit is a Java platform which consists of different software development kits. It contains different API classes, one Java compiler and the JVM interpreter to compile the applets and applications. J2SE is the latest version of the JDK distributed by the Sun Microsystem [46].

JFreeChart

JFreeChart is a free Java chart library which is used along with JasperReports. It is used to facilitate the developers to develop professional quality charts in their applications. JFreeChart is a well-documented API which supports many different types of charts. It supports both server-side and client-side applications with many output types. These output types are Swing components, image files and vector graphics file formats. JFreeChart is open source and free software. It supports pie charts, bar charts, line charts, scatter plots, time series charts, high-low-open-close charts, candlestick plots, Gantt charts and so on [43].

JRE

Java Runtime Environment (JRE) contains different Java libraries, different necessary core classes and also Java Virtual Machine. It works as the subset of different SDK. This JRE is required to run all Java application and applets on the system. JRE is available either separately or with other environments like J2EE, J2SE and J2ME [44].

JSP

There are two types of text in a Java Server Page (JSP) document. First one is static data which are like HTML, SVG, WML, and XML. Second one is dynamic content which are different JSP elements. The extension of JSP files is `.jsp`. JSP can include the files at top which might be a complete JSP page or a fragment of a JSP page [33].

Oracle Application Server 10g

The Oracle Application Server (OAS) 10g is an integrated, standards-based software platform. It forms part of Oracle Corporation's Fusion Middleware technology stack. Oracle HTTP Server (Apache based) and OC4J (J2EE based) are the heart of OAS. It deploys J2EE-based applications. The latest version of OC4J is compatible with the J2EE 1.4. It also supports Service Oriented Architecture (SOA) [38].

Oracle Database10g

Oracle Database 10*g* is the first database designed for grid computing. This is the most flexible and cost-effective database to manage enterprise information along with the highest quality of service. Oracle Database 10*g* significantly reduces the costs of managing the IT environment, with a simplified install, greatly reduced configuration and management requirements, and automatic performance diagnosis and SQL tuning [32].

PDF

Portable Document Format (PDF) is Adobe Acrobat format. PDF is a proprietary format which allows reading electronic documents either online or off line. This format provides a page-by-page view of documents, exactly as they appear in their printed form, as well as allowing keyword search [40].

PL/SQL

PL/SQL is a procedural language extension to SQL for the Oracle database management system. This is the combination of database language and procedural programming language. There are three parts of PL/SQL. These are the declarative part, executable part, and exception-building part.

SQL statements can be used in PL/SQL blocks and subprograms while executing them in the Oracle. Response time can be improved by compiling PL/SQL blocks once and storing them in executable form. A PL/SQL program that is stored in a database in compiled form and can be called by name is referred to as a stored procedure. A PL/SQL stored procedure that is implicitly started when an INSERT, UPDATE or DELETE statement is issued against an associated table is called a trigger [36].

WTP

Web Tools Platform Project is an open source project of eclipse.org, overseen by a PMC and project leaders. The PMC Planning Council coordinates identified resources of the project against a Project Development Plan. The work is done in sub projects working against a CVS repository. The Eclipse Web Tools Platform Project Charter describes the organization of the project, roles and responsibilities of the participants, and top-level development process for the project [29].

XML

XML is Extensible Markup Language like HTML. The purpose of the XML is to dealing with data rather than displaying them. It is like carrying data. All the tags used in XML are user defined rather than being default. It is self descriptive and store data with focus on what the data is. The latest version of XML is 2.0 now [42].

Abbreviation

BIRT	Business Intelligence and Reporting Tool
CSV	Comma Separated Value
DTP	Data Tools Platform
EMF	Eclipse Modeling Framework
GEF	Graphical Editing Framework
GUI	Graphical User Interface
HTML	Hypertext Markup Language
JAR	Java Archive
JDBC	Java Database Connectivity
JDK	Java Development Kit
JRE	Java Runtime Environment
JSP	Java Server Pages
PDF	Portable Document Format
PL	Procedural Language
PMC	Project Management Committee
SQL	Structured Query Language
WTP	Web Tools Platform
XML	Extensible Markup Language

References

1. http://www.logica.com/
2. Actuate Corporation,(2007) A Tale of Two Technologies: Business Intelligence s Rich Internet Applications, Published by Actuate Corporation, https://www.redhat.com/solutions/intelligence/collateral/actuate_RIAvsBI-08.pdf
3. Actuate Corporation,(2006) Technical White Paper: Collaborative Reporting Architecture, Published by Actuate Corporation, https://www.redhat.com/solutions/intelligence/collateral/acutate_collaborative_reporting_architecture_whitepaper.pdf
4. Maxwell, Joseph A. (2005) *Qualitative Research Design: An Interactive Approach*. Sage publication, Inc. USA
5. Dawson, C., (2002) *Practical Research Methods*. Published by British library catalogue, United Kingdom
6. Bryman, A., (2004) *Social Research Methods*. Second Edition, published by Oxford University Press Inc., New York
7. Bryman, A. (2001). *Social Research Methods*. New York, Oxford University Pres
8. Karlöf, B., (1995) *Benchmarking Workbook*. John Wiley & Sons Ltd
9. Diana Peh, Nola hague, Jane Tatchell, (2008) *BIRT: A Field Guide to Reporting*, 2nd edition, Published by Actuate Corporation
10. David Michonneau, (2008) *Pros and Cons of different ways to insert chart into JSP,* Published by Eclipse Newsgroup, http://dev.eclipse.org/newslists/news.eclipse.birt/msg25550.html
11. Paul Rogers, (2005) *Comparing Jasper,FOP and BIRT and others,* Published by Eclipse Newsgroup, http://dev.eclipse.org/newslists/news.eclipse.birt/msg00702.html
12. Phil Zoio, (2008) *Open Source reporting libraries*, Published by Javawug forum, http://groups.google.com/group/javawug/browse_thread/thread/d8b025059ea221c2?pli=1
13. Jaspersoft, (2009) *White Paper: JasperReports*, Published by Jaspersoft Corporation, http://www.jaspersoft.com/JasperSoft_JasperReports.html
14. Paul Rogers, (2005) *Comparing Jasper,FOP and BIRT and others,* Published by Eclipse Newsgroup, http://dev.eclipse.org/newslists/news.eclipse.birt/msg00618.html
15. Barry Klawans, (2005) *Comparing Jasper,FOP and BIRT and others,* Published by Eclipse Newsgroup, http://dev.eclipse.org/newslists/news.eclipse.birt/msg00678.html
16. Enholm Heuristics, (2007) *IReport for JasperReports*, Published by Enholm Heuristics, http://enholm.net/modules/wordpress/?p=12
17. Hossam, (2008) *Java Reporting With Jasper Reports,* Published by Javalobby http://java.dzone.com/articles/java-reporting-part-2
18. *Business Intelligence with Pentaho*, Published by Linails, www.linalis.com/index.php/fr/content/download/341/2475/file/Seminaire%20Linalis%20BI%20suite%20Oct%202007.pdf
19. RAQ Reports, (2008) RAQ Reports *Features and Benefits*, Published by Raqsoft http://www.raqsoft.com/product/key-features/features/

20. Zhou Feng, (2008) *The Integration Solution of WEB Report—with RAQ Report*, Published by TheServerSide.COM, Enterprise Java Community, http://www.theserverside.com/discussions/thread.tss?thread_id=51661

21. EKO, *Systemdok EKO 2.7*, Logica

22. BIRT Project, (2008) *Business Intelligence and Reporting Tools*, Published by Eclipse Foundation, http://www.eclipse.org/birt/phoenix/

23. BIRT Installation, (2008) *Installing the BIRT Viewer in Tomcat*, Published by Eclipse Foundation, http://www.eclipse.org/birt/phoenix/deploy/viewerSetup.php

24. BIRT Deploy, (2008) *BIRT Integration*, Published by Eclipse Foundation, http://www.eclipse.org/birt/phoenix/deploy/

25. BIRT Intro, (2008) *BIRT Overview*, Published by Eclipse Foundation, http://www.eclipse.org/birt/phoenix/intro/

26. *GEF Overview*, Publishd by Eclipse Foundation, http://www.eclipse.org/gef/overview.html

27. Eclipse Modeling Framework Project, Publishd by Eclipse Foundation, http://www.eclipse.org/modeling/emf/

28. Eclipse Data Tools Platform (DTP) Project, Publishd by Eclipse Foundation http://www.eclipse.org/datatools/

29. *About the Web Tools Platform Project*, Publishd by Eclipse Foundation, http://www.eclipse.org/webtools/about.php

30. JasperAssistant (2008) *JasperAssistant User Guide*, (Version 3.1.0), Published by Infologic SA http://www.jasperassistant.com/docs/guide/index.html

31. JasperServer 2.1.0*,* (2008) *JasperServer - Reporting, Analysis*, Published by SourceForge, Inc, http://sourceforge.net/project/shownotes.php?release_id=553759&group_id=162962

32. *Oracle Database 10g Tchnical Product Information*, Published by Oracle Technology Network, http://www.oracle.com/technology/products/database/oracle10g/index.html

33. J2EE 1.4 Tutorial, *JavaServer Pages Technology - Documentation*, Publishd by Sun Microsystem, http://java.sun.com/j2ee/1.4/docs/tutorial/doc/JSPIntro2.html

34. http://www.webopedia.com/TERM/J/JAR.html

35. (2004) *IBM Red Brick Warehouse 6.3*, IBM Corporation, http://publib.boulder.ibm.com/infocenter/rbhelp/v6r3/index.jsp?topic=/com.ibm.redbrick.doc6.3/ciacg/ciacg33.htm

36. Yu-May Chang and Jeff Ullman, (1997) *Using Oracle PL/SQL,* http://infolab.stanford.edu/~ullman/fcdb/oracle/or-plsql.html

37. Christopher Kowalsky, (2008), *Apache*, Published by Techtarget, http://searchcio-midmarket.techtarget.com/sDefinition/0,,sid183_gci211576,00.html

38. *Oracle Application Server*, Published by Wikipedia, http://en.wikipedia.org/wiki/Oracle_Application_Server

39. Andrew Trubac, (2001) *HTML*, Published by TechTarget, http://searchsoa.techtarget.com/sDefinition/0,,sid26_gci212286,00.html

40. *What is PDF?*, Published by Bar-Ilan University, http://www.biu.ac.il/acrobat.html

41. Shelley Elmblad, *What is CSV?*, Published by About.com, http://financialsoft.about.com/od/glossaryindexc/f/CSV.htm

42. (2009) *What is XML?*, Published by Refsnes Data, http://www.w3schools.com/XML/xml_whatis.asp

43. David Gilbert, Thomas Morgner, (2007) *JFreeChart*, Published by JFree , http://www.jfree.org/index.html

44. *Report Engine API,* Published by Eclipse Foundation, http://www.eclipse.org/birt/phoenix/deploy/reportEngineAPI.php

45. javapeople, (2005) *What is JRE?*, Published by JAVA Rocks, http://javapeople.blogspot.com/2005/12/what-is-jre.html

46. *What id JDK?*, Published by programmers Heaven, http://www.programmersheaven.com/2/FAQ-JAVA-What-Is-JDK

47. Jim Mason, (2004) *Web-based report writing with DataVision*, Published by Search400.com, http://search400.techtarget.com/tip/0,289483,sid3_gci990723,00.html

48. (2008) *DataVision Home*, Published by Jim Menard, http://DataVision.sourceforge.net/

49. (2008) *RAQ* Repor, Published by RAQ Soft, http://www.raqsoft.com/product/raq-report/raq-report/

50. (2009) *JasperReports Project Home*, Published by Jaspersoft Corporation, http://jasperforge.org/plugins/project/project_home.php?group_id=102

51. *Using the BIRT Report Viewer*, Published by Eclipse Foundation, http://www.eclipse.org/birt/phoenix/deploy/viewerUsage2.2.php

Appendices

Appendix I: Voice of Customer

Questionnaire

1. What requirements / functionalities should the EKO reports fulfill?
2. What kind of data/information should it represent and produce to be effective?
3. How would you like to use it and who might be the end user of it?
4. What business advantages/added values should it produce for the whole organization?
5. Which kind of Business Intelligence and Reporting Tools do you want to use like free and open source or just free or commercials?
6. Which platform do you want to transfer the whole reporting system of EKO?
7. In what language the reporting tool has been developed in EKO?
8. How many reports are there and which one is the most important?
9. Which packages, procedures and functions are used to generate the reports?
10. Which database tables are involved to generate the reports?
11. Are all the inputs given at a time while generating the report?
12. Which inputs are mandatory and which are optional to generate the reports?
13. What is the purpose of Ändamål (group of different sources of money) columns?
14. What is the purpose of Skapa rapport?
15. Does Skapa rapport changes the format of the reports with the same inputs or it takes different data fields while generating the report?
16. How about adding graphical charts to the reports?
17. How many years a user can choose to see the data?
18. Do you have any other suggestions about this project and its scope?

Participants:
PM: Magnus Carlsson
1st Person from Maintenance and Support: Mr. Lars Lundberg
2nd Person from Maintenance: Mrs. Ing-Marie Andrén
Developer: Ivar Hagen

Appendix II: Source Code

Packages, Procedures and Functions for the existing system

The source code of the report designing and developing is copyrighted by Logica and is of 2,801 lines. So, in request and by the permission of Logica this code will only be available to see but not to use.

Report Designing and Developing Code

The source code of the report designing and developing is copyrighted and is of 21,557 lines. So, in request this code will only be available to see but not to use.

Graphical User Interface Code

The source code of the report designing and developing is copyrighted and is of 508 lines. So, in request this code will only be available to see but not to use.

Examples from different report engine class

EngineConfig

```
EngineConfig config = new EngineConfig( );
config.setEngineHome( "put engine path here" );
```

ReportEngine

```
try{
        final config = new EngineConfig( );
        config.setEngineHome( "C:\birt-runtime-2_1_0\birt-runtime-
2_1_0\ReportEngine" );
        config.setLogConfig(c:/temp, Level.FINE);

        Platform.startup( config );  //If using RE API in Eclipse/RCP
application this is not needed.
        IReportEngineFactory factory = (IReportEngineFactory) Platform
                        .createFactoryObject(
IReportEngineFactory.EXTENSION_REPORT_ENGINE_FACTORY );
        IReportEngine engine = factory.createReportEngine( config );
        engine.changeLogLevel( Level.WARNING );

}catch( Exception ex){
        ex.printStackTrace();
}
// Run reports, etc.
...

// destroy the engine.
try
{
        engine.destroy();
        Platform.shutdown();
}
catch ( EngineException e1 )
{
    // Ignore
}
```

IReportRunnable

```
IReportRunnable report = engine.openReportDesign( name );
```

IReportDocument

```
IReportDocument ird =
engine.openReportDocument("c:/work/test/TOCTest.rptdocument");
```

```
//get root node
TOCNode td = ird.findTOC(null);
List children = td.getChildren( );
//Loop through Top Level Children
if ( children != null && children.size( ) > 0 )
{
  for ( int i = 0; i < children.size( ); i++ )
  {
    TOCNode child = ( TOCNode ) children.get( i );
    System.out.println( "Node ID " + child.getNodeID());
    System.out.println( "Node Display String " + child.getDisplayString());
    System.out.println( "Node Bookmark " + child.getBookmark());

  }
}
```

IEngineTask

Create tasks using the factory methods on the `ReportEngine` class. The supported Tasks are shown below:

- `engine.createDataExtractionTask();`
- `engine.createGetParameterDefinitionTask();`
- `engine.createRenderTask();`
- `engine.createRunTask();`
- `engine.createRunAndRenderTask();`

IDataExtractionTask

```
//Open previously created report document
IReportDocument iReportDocument =
engine.openReportDocument("c:/work/test/TOCTest.rptdocument");

//Create Data Extraction Task
IDataExtractionTask iDataExtract =
engine.createDataExtractionTask(iReportDocument);

//Get list of result sets
ArrayList resultSetList = (ArrayList)iDataExtract.getResultSetList( );

//Choose first result set
IResultSetItem resultItem = (IResultSetItem)resultSetList.get( 0 );
String dispName = resultItem.getResultSetName( );
iDataExtract.selectResultSet( dispName );

IExtractionResults iExtractResults = iDataExtract.extract();
IDataIterator iData = null;
try{
    if ( iExtractResults != null )
    {
      iData = iExtractResults.nextResultIterator( );

      //iterate through the results
      if ( iData != null  )
      {
        while ( iData.next( ) )
```

```
        {
          Object objColumn1;
          Object objColumn2;
          try
          {
            objColumn1 = iData.getValue(0);
           }catch(DataException e)
            {
               objColumn1 = new String("");
             }
             try{
               objColumn2 = iData.getValue(1);
             }catch(DataException e){
               objColumn2 = new String("");
              }
              System.out.println( objColumn1 + " , " + objColumn2 );
          }
          iData.close();
        }
      }
}catch( Exception e){
 e.printStackTrace();
}

iDataExtract.close();
```

IGetParameterDefinitionTask

```
//Open a report design
IReportRunnable design =
engine.openReportDesign("C:/work/test/parameters.rptdesign");

IGetParameterDefinitionTask task = engine.createGetParameterDefinitionTask(
design );
Collection params = task.getParameterDefns( true );

Iterator iter = params.iterator( );
//Iterate over all parameters
while ( iter.hasNext( ) )
{
  IParameterDefnBase param = (IParameterDefnBase) iter.next( );
  //Group section found
  if ( param instanceof IParameterGroupDefn )
  {
   //Get Group Name
   IParameterGroupDefn group = (IParameterGroupDefn) param;
   System.out.println( "Parameter Group: " + group.getName( ) );

   //Get the parameters within a group
   Iterator i2 = group.getContents( ).iterator( );
   while ( i2.hasNext( ) )
   {
     IScalarParameterDefn scalar = (IScalarParameterDefn) i2.next( );
     System.out.println("      " + scalar.getName());
   }

  }
  else
  {
   //Parameters is not in a group
```

```
   IScalarParameterDefn scalar = (IScalarParameterDefn) param;
   System.out.println(param.getName());

   //Parameter is a List Box
   if(scalar.getControlType() ==  IScalarParameterDefn.LIST_BOX)
   {
    Collection selectionList = task.getSelectionList(param.getName());
    //Selection contains data
    if ( selectionList != null )
    {
      for (Iterator sliter = selectionList.iterator(); sliter.hasNext(); )
      {
        //Print out the selection choices
        IParameterSelectionChoice selectionItem =
                            (IParameterSelectionChoice) sliter.next( );
        String value = (String)selectionItem.getValue( );
        String label = selectionItem.getLabel( );
        System.out.println( label + "--" + value);
       }
      }
     }
    }
   }
   task.close();
```

IRenderTask

```
//Open a report document
IReportDocument iReportDocument =
engine.openReportDocument("c:/work/test/Pages.rptdocument");
//Create Render Task
IRenderTask task = engine.createRenderTask(iReportDocument);
//Set parent classloader report engine
task.getAppContext().put(EngineConstants.APPCONTEXT_CLASSLOADER_KEY,
RenderTaskExample.class.getClassLoader());

IRenderOption options = new RenderOption();
options.setOutputFormat("html");
options.setOutputFileName("output/resample/eventorder.html");


if( options.getOutputFormat().equalsIgnoreCase("html"))
{
  HTMLRenderOption htmlOptions = new HTMLRenderOption( options);
  htmlOptions.setImageDirectory("output/image");
  htmlOptions.setHtmlPagination(false);
  //set this if you want your image source url to be altered
  //If using the setBaseImageURL, make sure to set image handler to
  //HTMLServerImageHandler
  htmlOptions.setBaseImageURL("http://myhost/prependme?image=");

  htmlOptions.setHtmlRtLFlag(false);
  htmlOptions.setEmbeddable(false);
}else if( options.getOutputFormat().equalsIgnoreCase("pdf") ){

  PDFRenderOption pdfOptions = new PDFRenderOption( options );
  pdfOptions.setOption(IPDFRenderOption.FIT_TO_PAGE, new Boolean(true) );
  pdfOptions.setOption(IPDFRenderOption.PAGEBREAK_PAGINATION_ONLY, new
Boolean(true) );
```

```
}
//Use this method if you want to provide your own action handler
options.setActionHandler(new MyActionHandler());

//file based images
//options.setImageHandler(new HTMLCompleteImageHandler())
//Web based images.  Allows setBaseImageURL to prepend to img src tag
options.setImageHandler(new HTMLServerImageHandler());

IRenderTask task = engine.createRenderTask(document);
task.setRenderOption(options);
task.setPageRange("1-2");
task.render();
iReportDocument.close();
```

IRunTask

```
//Open a report design
IReportRunnable design =
engine.openReportDesign("C:/work/test/MyOrders.rptdesign");

//Create task to run the report - use the task to execute the report and
save to disk.
IRunTask task = engine.createRunTask(design);
//Set parent classloader for engine
task.getAppContext().put(EngineConstants.APPCONTEXT_CLASSLOADER_KEY,
RunTaskExample.class.getClassLoader());

//run the report and destroy the engine
task.run("c:/work/test/MyOrders.rptdocument");

task.close();
```

IRunAndRenderTask

```
//Open the report design
IReportRunnable design =
engine.openReportDesign("Reports/TopNPercent.rptdesign");


//Create task to run and render the report,
IRunAndRenderTask task = engine.createRunAndRenderTask(design);
//Set parent classloader for engine
task.getAppContext().put(EngineConstants.APPCONTEXT_CLASSLOADER_KEY,
RunAndRenderTaskExample.class.getClassLoader());

//Set parameter values and validate
task.setParameterValue("Top Percentage", (new Integer(3)));;
task.setParameterValue("Top Count", (new Integer(5)));
task.validateParameters();

//Setup rendering to HTML
HTMLRenderOption options = new HTMLRenderOption();
options.setOutputFileName("output/resample/TopNPercent.html");
options.setOutputFormat("html");
//Setting this to true removes html and body tags
options.setEmbeddable(false);
```

```
task.setRenderOption(options);
//run and render report
task.run();
task.close();
```

Appendix III: User Interface

GUI of existing system

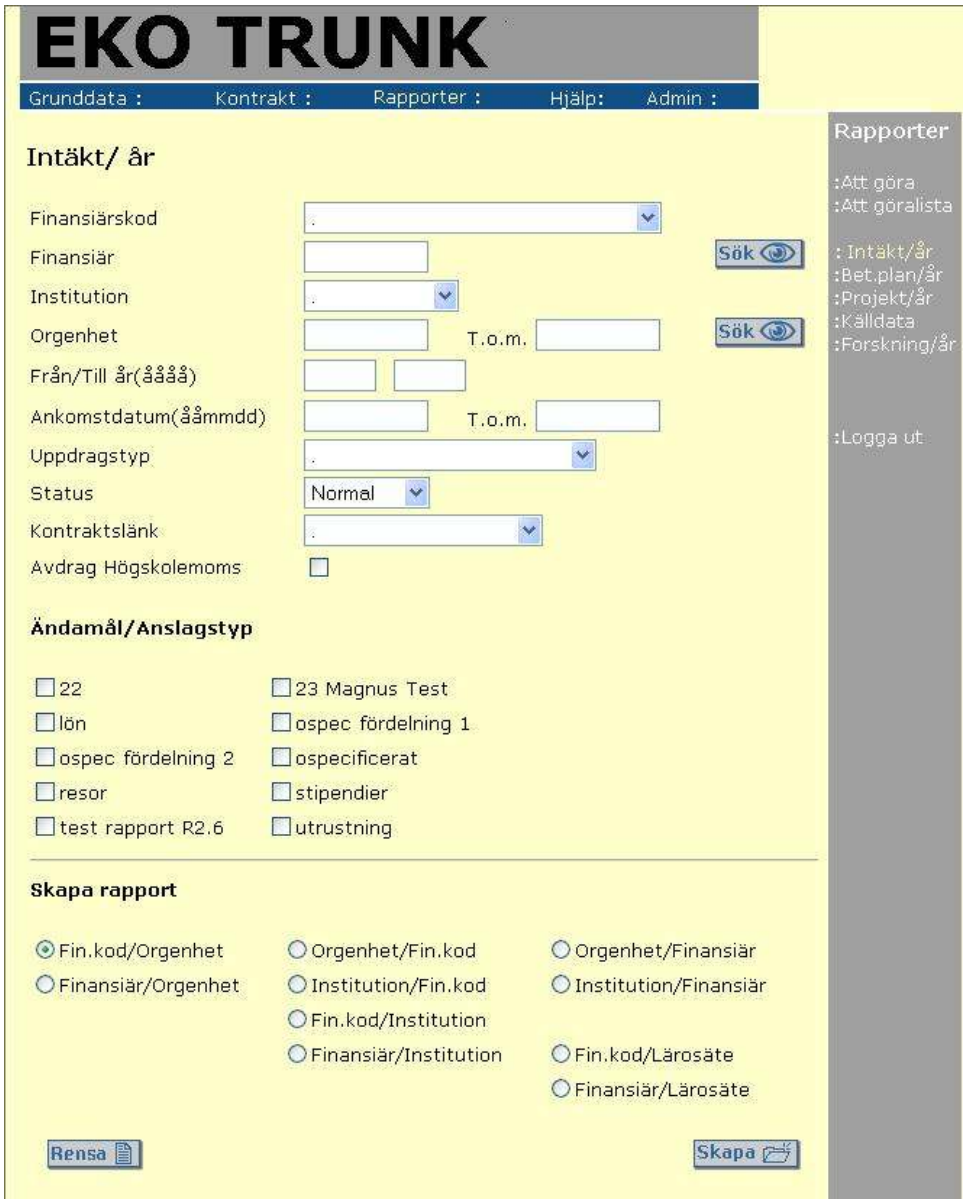
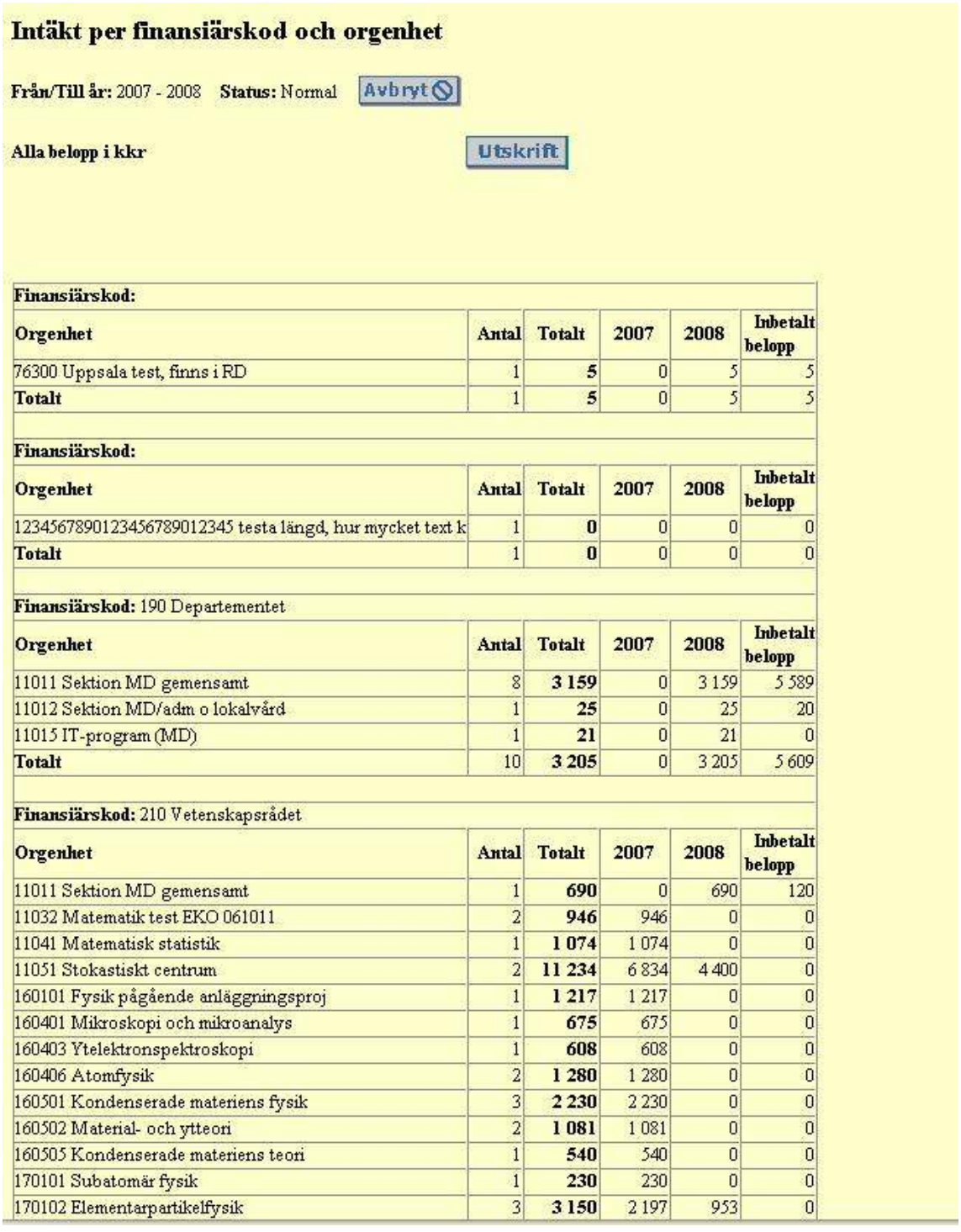


Figure AIII.1: Graphical User Interface of existing system

Report from existing system



Intäkt per finansiärskod och orgenhet

Från/Till år: 2007 - 2008 **Status:** Normal Avbryt

Alla belopp i kkr Utskrift

Finansiärskod:					
Orgenhet	**Antal**	**Totalt**	**2007**	**2008**	**Inbetalt belopp**
76300 Uppsala test, finns i RD	1	**5**	0	5	5
Totalt	1	**5**	0	5	5
Finansiärskod:					
Orgenhet	**Antal**	**Totalt**	**2007**	**2008**	**Inbetalt belopp**
12345678901234567890123345 testa längd, hur mycket text k	1	**0**	0	0	0
Totalt	1	**0**	0	0	0
Finansiärskod: 190 Departementet					
Orgenhet	**Antal**	**Totalt**	**2007**	**2008**	**Inbetalt belopp**
11011 Sektion MD gemensamt	8	**3 159**	0	3 159	5 589
11012 Sektion MD/adm o lokalvård	1	**25**	0	25	20
11015 IT-program (MD)	1	**21**	0	21	0
Totalt	10	**3 205**	0	3 205	5 609
Finansiärskod: 210 Vetenskapsrådet					
Orgenhet	**Antal**	**Totalt**	**2007**	**2008**	**Inbetalt belopp**
11011 Sektion MD gemensamt	1	**690**	0	690	120
11032 Matematik test EKO 061011	2	**946**	946	0	0
11041 Matematisk statistik	1	**1 074**	1 074	0	0
11051 Stokastiskt centrum	2	**11 234**	6 834	4 400	0
160101 Fysik pågående anläggningsproj	1	**1 217**	1 217	0	0
160401 Mikroskopi och mikroanalys	1	**675**	675	0	0
160403 Ytelektronspektroskopi	1	**608**	608	0	0
160406 Atomfysik	2	**1 280**	1 280	0	0
160501 Kondenserade materiens fysik	3	**2 230**	2 230	0	0
160502 Material- och ytteori	2	**1 081**	1 081	0	0
160505 Kondenserade materiens teori	1	**540**	540	0	0
170101 Subatomär fysik	1	**230**	230	0	0
170102 Elementarpartikelfysik	3	**3 150**	2 197	953	0

Figure AIII.2: Report Interface of existing system

Report designing interface

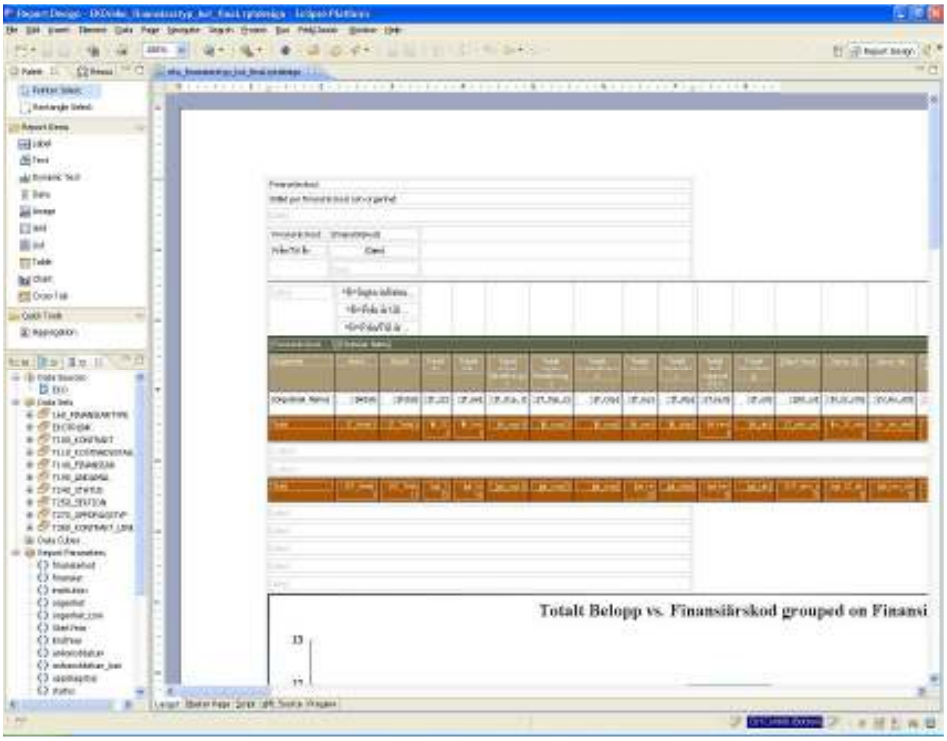


Figure AIII.3: Report Deigning Interface

Chart Designing wizard

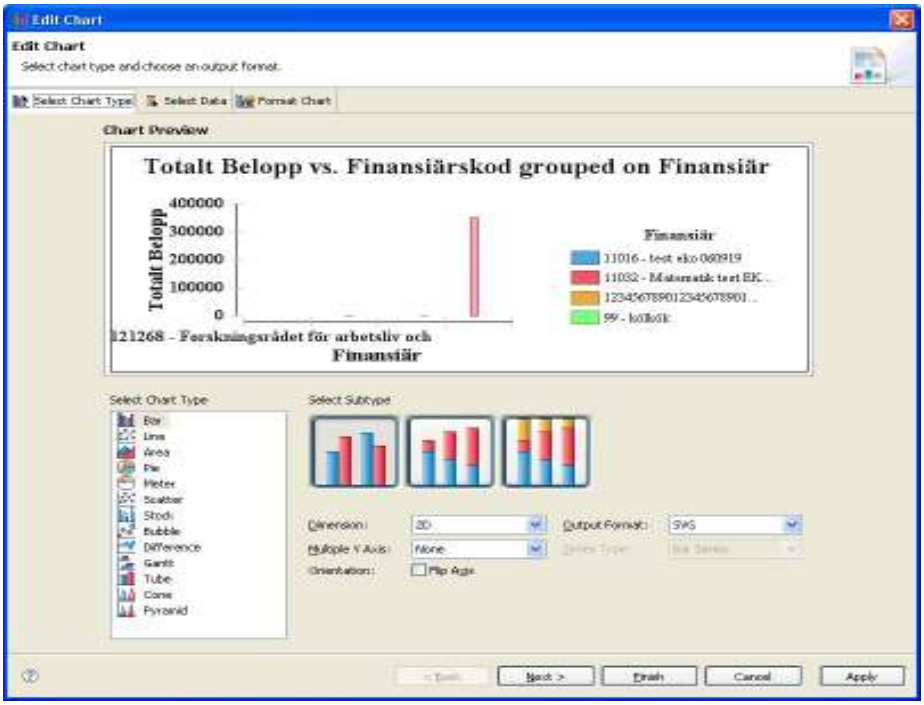


Figure AIII.4: Chart Designing Wizard-chart type selection tab

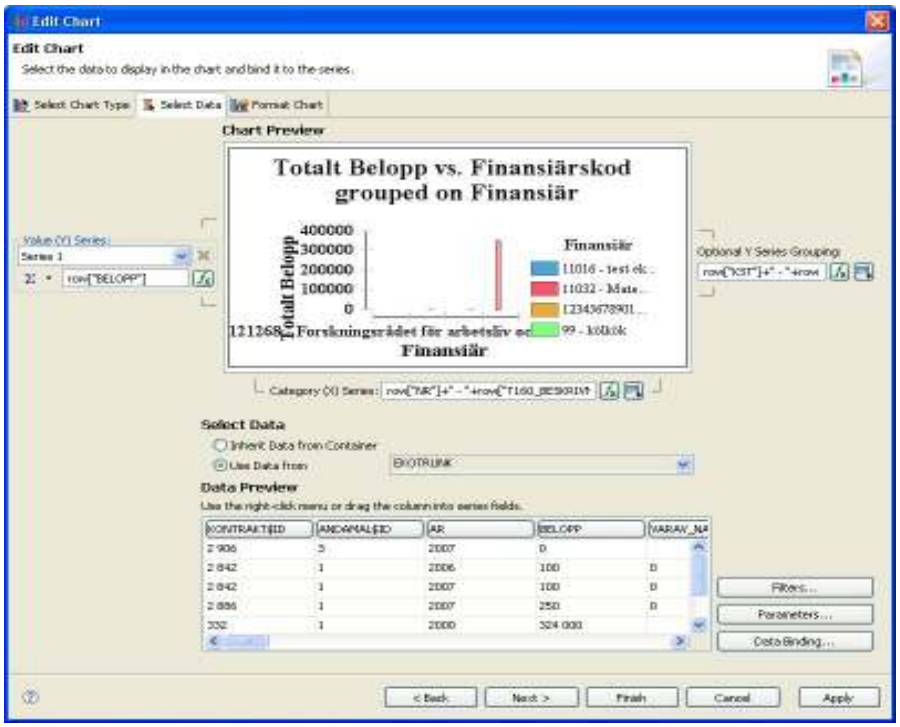


Figure AIII.5: Chart Designing Wizard-data selection tab

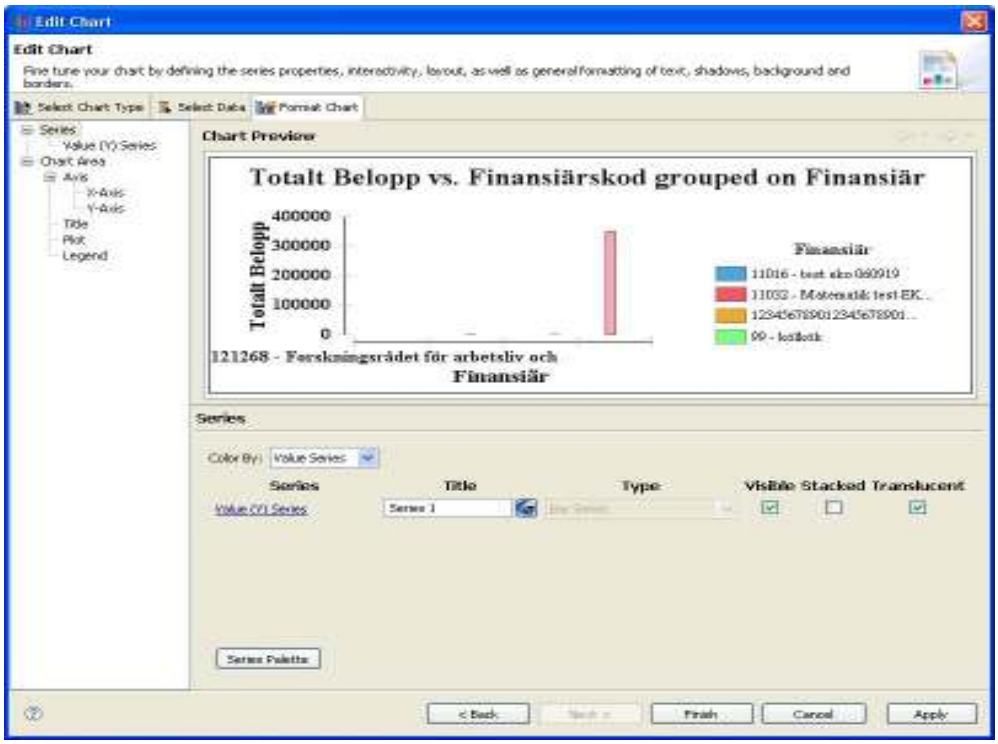


Figure AIII.6: Chart Designing Wizard-chart formatting tab

Charts

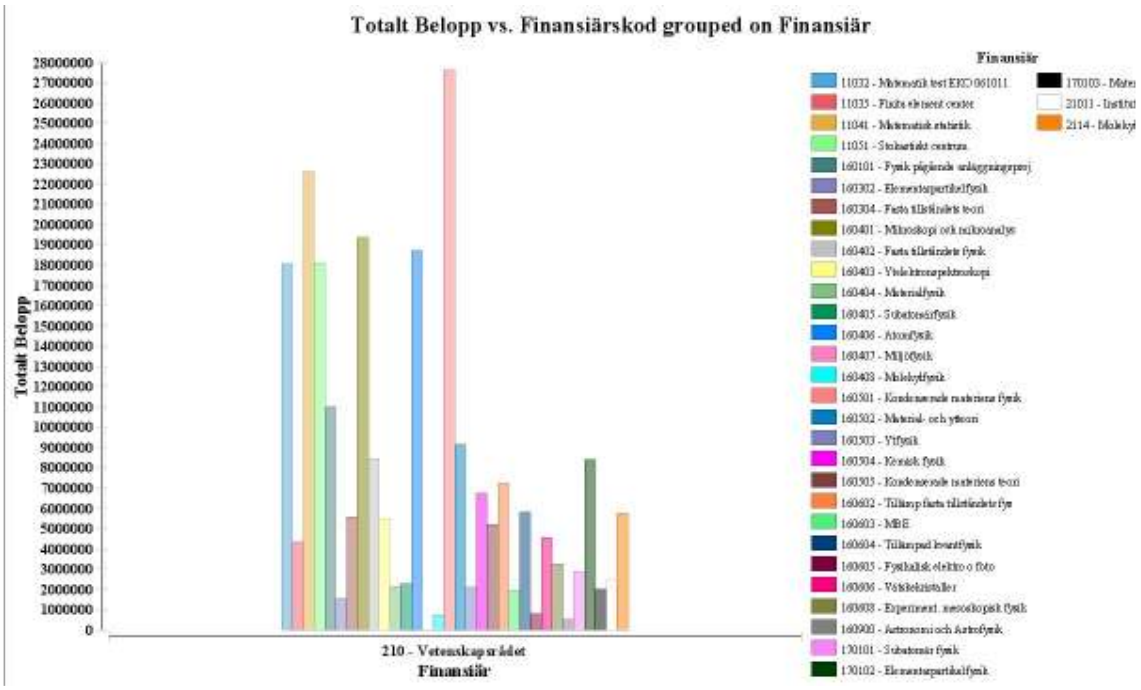


Figure AIII.7: Bar Chart- Total Amount vs. Company grouped by department of the company

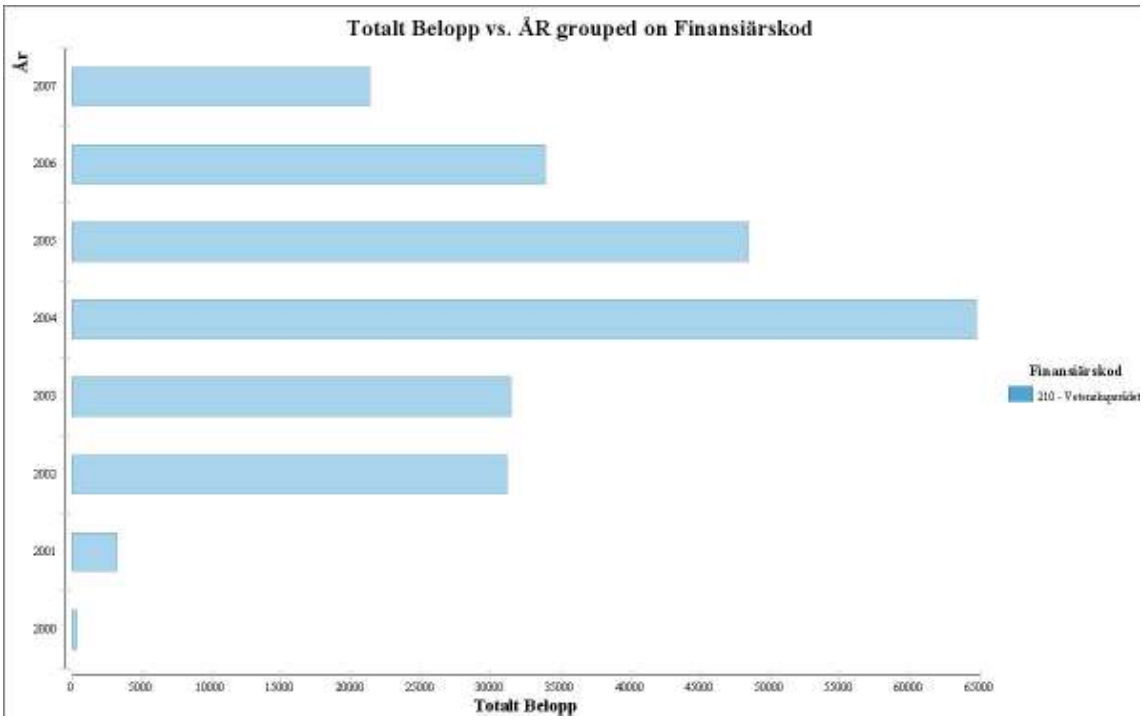


Figure AIII.8: Bar Chart- Total Amount vs. Year grouped by Company

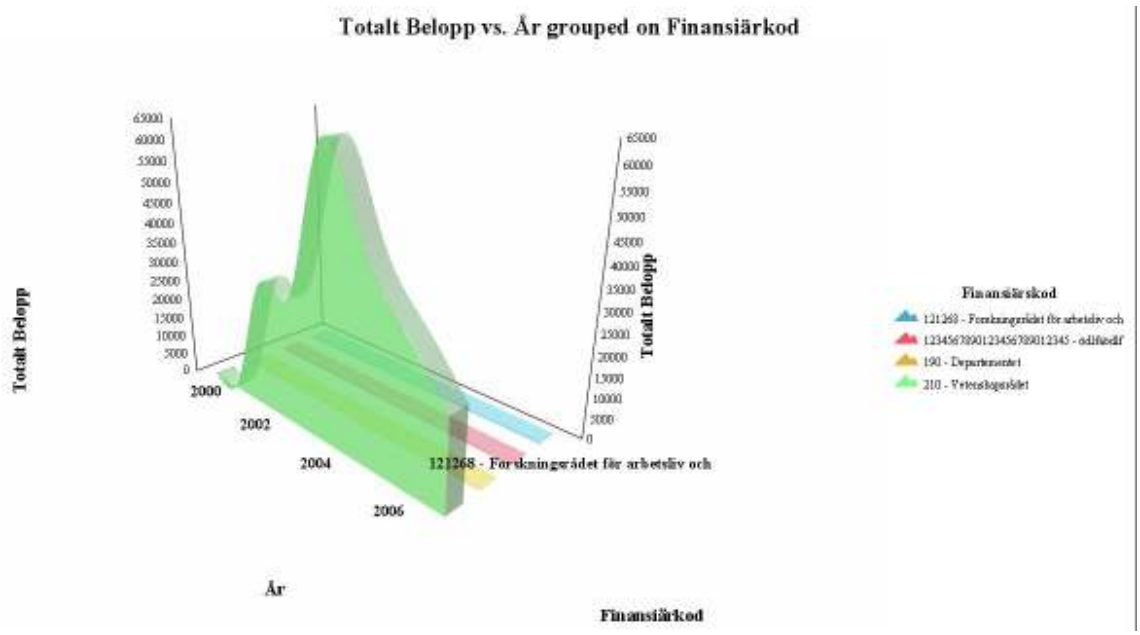


Figure AIII.9: Area Chart- Total Amount vs. Year grouped by Company

Report view

Finansierkod/Orgenhet Report

Showing page 1 of 1 Go to page:

Finansianskod:
Intäkt per finansiärskod och orgenhet

Finansianskod:
Från/Till år: 2003 - 2008

Finansianskod:	121268 Forskningsrådet för arbetsliv och																	
Orgenhet	Antal	Totalt	Totalt 22	2003	Varav 22	2004	Varav 22	2005	Varav 22	2006	Varav 22	2007	Varav 22	2008	Varav 22	Inkortalt belopp	Varav 22
11016 test eko 060919	2	7												7		0	
75300 Uppsala test, finns i RD	1	5												5		5	
Total	3	12												12		5	

Finansianskod:	12345678901234567890123456 odifasdf																	
Orgenhet	Antal	Totalt	Totalt 22	2003	Varav 22	2004	Varav 22	2005	Varav 22	2006	Varav 22	2007	Varav 22	2008	Varav 22	Inkortalt belopp	Varav 22
12345678901234567890123456 testa längd, hur mycket text k	3	0								0		0		0		0	
Total	3	0								0		0		0		0	

Finansianskod:	190 Departementet																	
Orgenhet	Antal	Totalt	Totalt 22	2003	Varav 22	2004	Varav 22	2005	Varav 22	2006	Varav 22	2007	Varav 22	2008	Varav 22	Inkortalt belopp	Varav 22
11011 Sektion MD gemensamt	13	3481	0											3 481	0	5 938	0
11012 Sektion MD/adm o lokalvård	1	25												25		20	
11015 IT-program (MD)	1	21												21		0	
99 kolkok	2	0										0		0		0	
Total	17	3527	0									0		3527	0	5 958	0

Figure AIII.10: Report View-Generated from the GUI

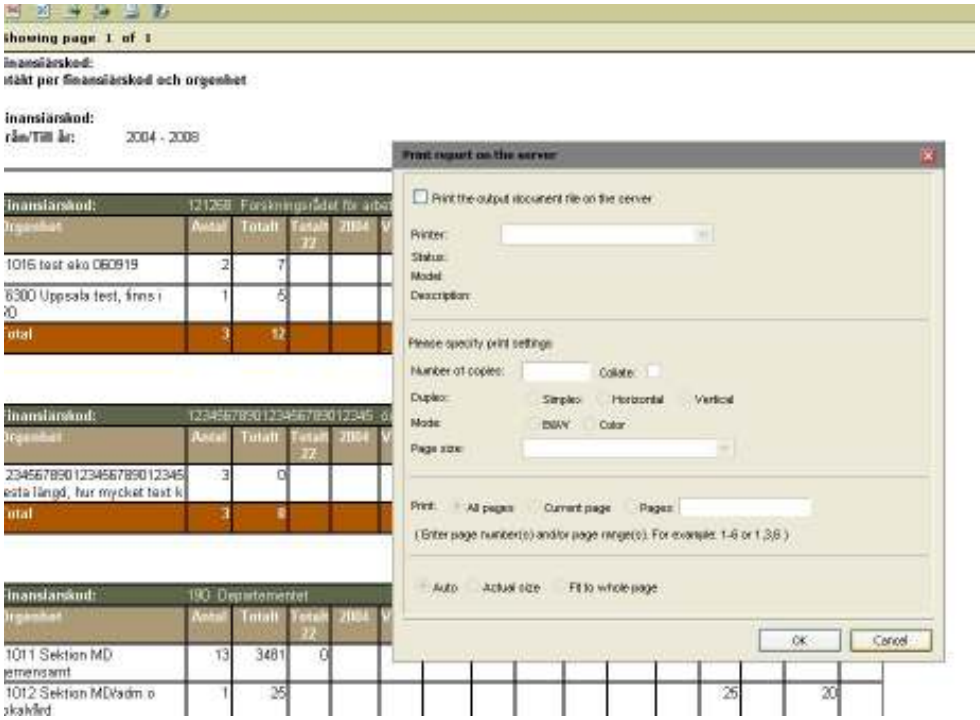


Figure AIII.11: Report View-Server Printer setup

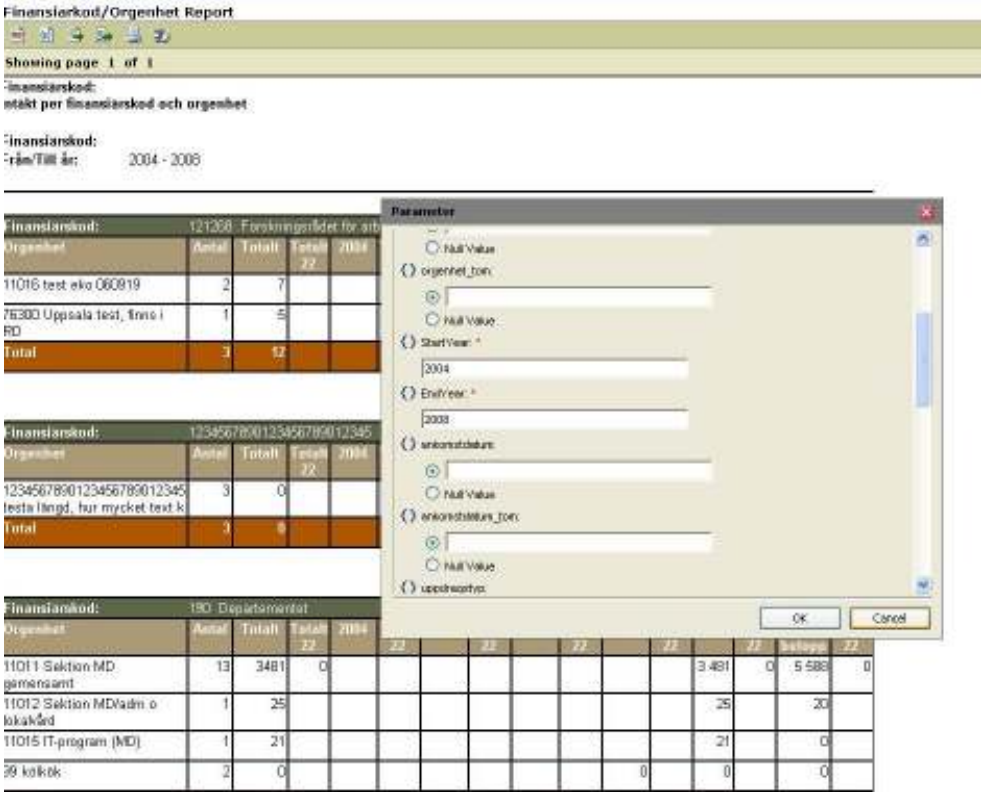


Figure AIII.12: Report View-Parameter Dialog box

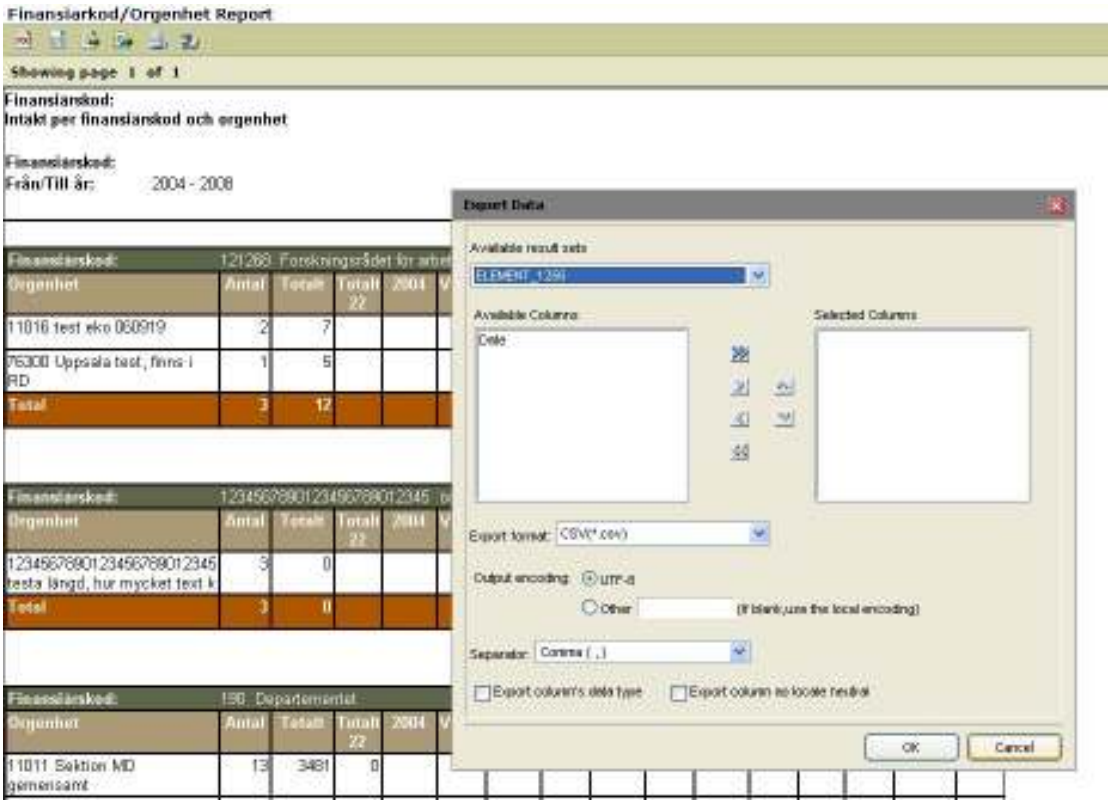


Figure AIII.13: Report View-CSV format output

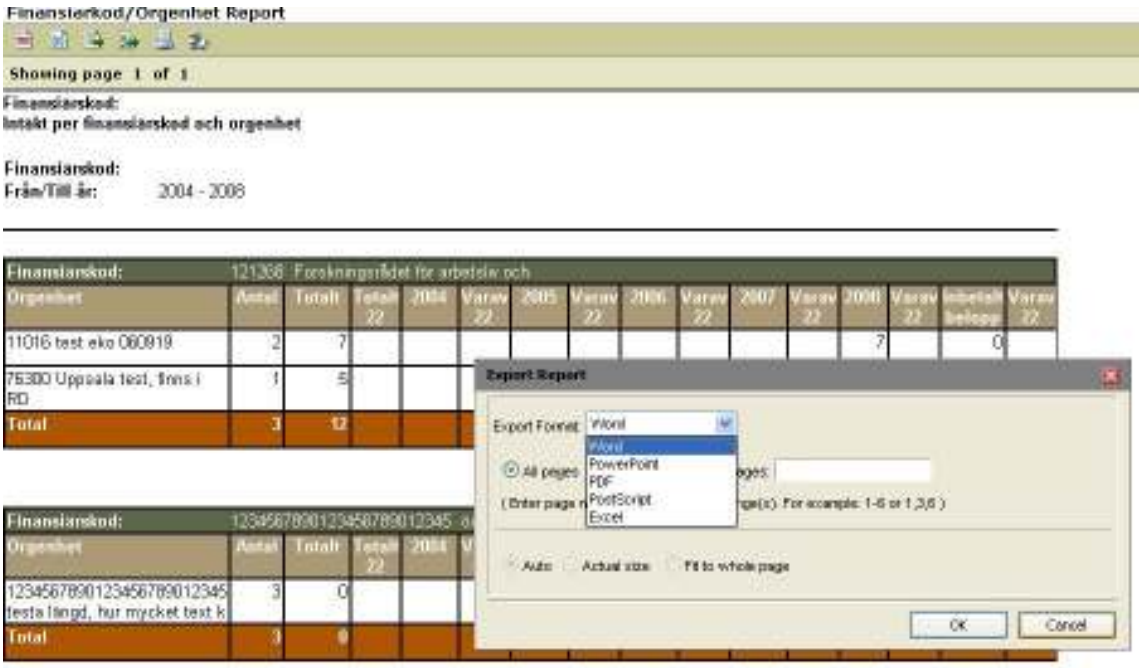


Figure AIII.14: Report View-Export Format