Radboud University Nijmegen

Master Thesis

OLAP Cubes

Enabling Business Intelligence in Case Management Tools

-Version 1.0-

Author:
V.A. (Vincent) Kool

Supervisor:
Prof. Dr. M.C.J.D. (Marko) van Eekelen
Dr. A. (Arjan) van Rooij
Drs. M.A.J.M. (Marc) Verspeek MBA

Thesis number: 665
Faculty of Science - Computing Science
August 20, 2012

Abstract

Adding business intelligence to case management tools enables the management of management of cases, activities and users by monitoring, analysing, predicting and optimizing the business with the help of predefined reports, dashboard of self-service business intelligence. Online analytical processing (OLAP) databases are optimized for the required analytical operations.

Because a generic best design is not possible, design steps were formulated for creating business intelligence in case management tools. These steps require the case management tool, the business, the information need and the data as input.

The design steps were evaluated with a case study with Serenga and showed that most questions could be answered. The case study also showed some problems with the data, in order to fully support business intelligence some changes have to be made to the case management tool.

Executive Summary

The original research goal of this thesis was to find the best design for a business intelligence module in case management tools with the use of online analytical processing (OLAP) cubes. Case management tools are more flexible than workflow management systems and offer more support than for example document management systems. Adding business intelligence to case management tools should support the management of cases, activities and users by monitoring, analyzing, predicting and optimizing the business. The resulting data can be presented in the form of predefined reports or users are supported in finding the information themselves (self-service business intelligence). OLAP cubes are optimized for analytical operations and are therefore particular suited as database for business intelligence.

During the research it became clear that a generic best design does not exist. A best design for a business intelligence module must be specific to support the case management tool used, the business it support, the data available and the (business intelligence) questions asked. Therefore generic design steps are formulated for creating the business intelligence module and OLAP cube that support the case management tool the best.

The input necessary for the design steps is: the case management tool, the business and the business intelligence questions that will be asked and the data that is available in the case management tool. The first step is determining what the case management tool is, how the data is stored and how the data is connected. The second step is determining what questions must be answered by the tool. After looking at the data that is available, some deficiencies with the tool or the data can be found. These deficiencies have to be solved. Then the model of the OLAP cube can be made and the data can be loaded. The final step is deciding whether the resulting information must be presented with predefined reports/dashboards or that the users can navigate through the information with self-service business intelligence.

These design steps where used in a case study with Serenga. A database with actual user data was used in combination with business intelligence question from both the user and a consultant. One of the first observations was that users have questions that are too 'open'. The question "What are problem cases? was the only question that could not be answered. For answering some other questions data was not available in database. Another observation was that the naming of activities (sometimes too specific while at other times too generic) made that the required grouping and aggregation could not take place. For the case study assumptions were made (such as that finishing an activity was a milestone), but for using the tool in the future this has to be solved. The OLAP cube designed in the case study has been used as basis for predefined reports as well as for answering questions with self-service business intelligence.

The case study showed that the design steps result in a working OLAP database that add business intelligence to the case management tool. In order to do so, the case management tool should provide the necessary data and the business intelligence questions should be clear.

“ In theory, theory and practice are the same.
In practice, they are not ”

Albert Einstein

Contents

Chapter 1

Introduction

Computer systems try to support the work that has to be done. Emailing and digital text processing are maybe the most used and most known examples. Automation in industry is another example where the computer aids the human worker. Case management tools help knowledge workers in managing, planning and supporting cases they have to handle. Cases are a coherent quantity of work with a clear beginning and a well defined result, of which quality and lead time must be guarded. Cases can exist of tasks or activities and often involve multiple users.

Case management tools store a lot of data about all the transactions that take place and this data can be used for improvement of the process. Because normal databases are optimized for creating and updating transactions they are less suitable for querying over a lot of data and aggregating over that data. Online analytical processing (OLAP) databases are specially designed for retrieving data from large data warehouses and aggregate over this information. In this thesis we will look at the opportunities OLAP cubes can give for adding business intelligence to case management tools.

1.1 Research questions

The main research question was first defined as:

Original Research Goal *Find the best design for a business intelligence module in case management tools with the use of OLAP cubes and build a prototype.*

During the research it became clear that a best design in general doe not exist. The best design is specific for the case management tool, the business, the data and the information need. Therefore the research goal became:

New Research Goal *Find the design steps for creating the best design for a business intelligence module in case management tools with the use of OLAP cubes and build a prototype.*

For reaching the research goal several subquestions were made and divided in three categories.

1.1.1 Case management

First we look at what case management is (§ 2.1), what the relation and differences are with other systems like *document management systems* and *workflow management systems* (§ 2.1.1)

The following subquestions will be answered.

- What is case management?
- What are the characteristics of case management?
- What is a case management tool?
- What is the difference between case management (tool), document management (systems) and workflow management (systems)?

1.1.2 Business Intelligence

After looking at case management we look at business intelligence. We will first look at some definitions (§ 3.1), how business intelligence data can be visualized (§ 3.2) and what the importance is of business intelligence for case management (chapter 4).

The following subquestions will be answered.

- What is business intelligence?
- What is the difference between business intelligence and self-service business intelligence?
- How can you present (BI) information to the users? (tables/graphs/..)
- How can the information-need be converted to (OLAP cube)?
- How do users process (BI) information?

1.1.3 Databases

For both case management and business intelligence data is necessary. The data storage for case management must be optimized for the so-called CRUD-operations (create, read, update and delete) while business intelligence is more focused on aggregation. Online transaction processing (OLTP) databases are optimized for operational use (the CRUD-operations), Online analytical processing (OLAP) databases are optimized for aggregating measures along different dimensions.

In chapters 4.2 and 4.3 we look at the different types of databases. Further, we look at the necessary operations for both types of databases (§ 4.2.2 and § 4.3.3)

The following subquestions will be answered.

- What are Online transaction processing (OLTP) databases?
- How can OLTP databases formally be described? (Especially the database used for case management tools)?
  - Which Operations are available?
- What are Online analytical processing (OLAP) databases?
- How can OLAP databases formally be described?
  - Which operations are available?
- Which (formal) operations are necessary for transforming a OLTP database to an OLAP database?
- What are the benefits and disadvantages of OLAP databases (in comparison to OLTP databases)?
- What does it mean for the possibility of, and performance of asking question in dimension not defined in the OLAP cubes?

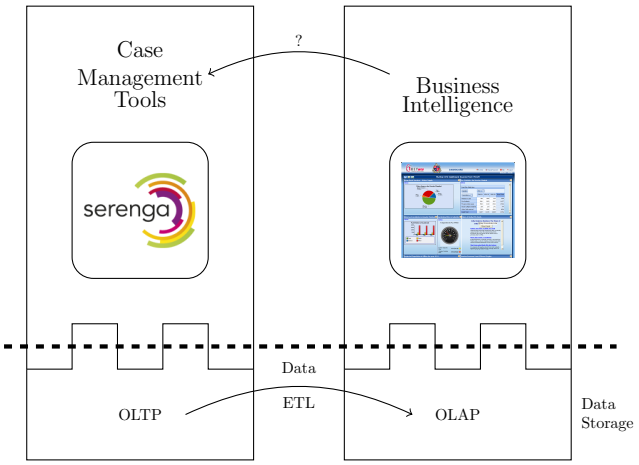


Figure 1.1: Relation between the different concepts

1.1.4 Case Study

Serenga will be used for creating a prototype for adding business intelligence to a case management tool. Therefore a description will be given (§5.1), the information need of the users as well

as the data that is currently available (§5.2 and §5.3). The creation of the cube and the loading of the data will be discussed in §5.4 and §5.5.

The following subquestions will be answered.

- How do the business processes supported by case management tools (and Serenga in particular) look like?
- What sorts of users use Serenga?
- What is the information need of the users of Serenga?
- How can the information needs, for the customers, be grouped?
- In what way will the information support their work?

1.1.5 Context

In figure 1.1 the relation between the concepts and research questions from the previous sections and the following chapters is illustrated. This illustration will be the guide in this thesis.

1.1.6 Conclusion

Finally the results will be evaluated, suggestions for further work/research and a conclusion will be given(Ch. 6).

Chapter 2

Case Management

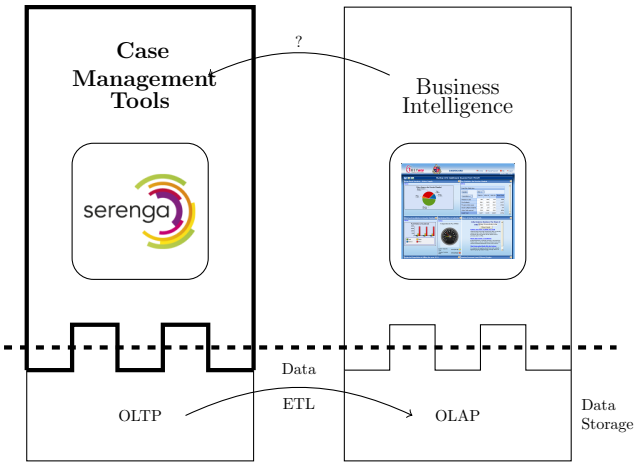


Figure 2.1: Case Management in relation with the different concepts

This thesis is about enabling business intelligence in case management tools. In order to go further we first have to define what a *case* is, what *case management* does and how *case management tools* support handling cases.

2.1 Overview

According to the Oxford Dictionaries[1], "(A case is) an instance of a particular situation; an example of something occurring". Merriam-Webster[2] defines a (medical) case manager as "a

[1] http://oxforddictionaries.com/
[2] http://www.merriam-webster.com/

person (as a social worker or nurse) who assists in the planning, coordination, monitoring, and evaluation of medical services for a patient with emphasis on quality of care, continuity of services, and cost-effectiveness". (Medical) Case management is defined by CSMA Australia[3] as "(...) a collaborative process of assessment, planning, facilitation and advocacy for options and services to meet an individuals holistic needs through communication and available resources to promote quality costeffective outcomes." Idealware made a comparison of multiple tools used the following definition of a case management tool: "A good case management tool will track the information you need to work with a client, such as their age, address, job history, medical history, and child care situation. It will also track all the contacts between your staff and the client, the individualized plan for your client, and the progress towards the plan. And it will allow you to report on all the information you've collected."[QL11]

This last definition reminds us of *customer relation management* tools, although those are really something different. "Customer relationship management (CRM) is a widely implemented strategy for managing a companys interactions with customers, clients and sales prospects. It involves using technology to organize, automate, and synchronize business processes, principally sales activities, but also those for marketing, customer service, and technical support. The overall goals are to find, attract, and win new clients, nurture and retain those the company already has, entice former clients back into the fold, and reduce the costs of marketing and client service. Customer relationship management describes a company-wide business strategy including customer-interface departments as well as other departments. Measuring and valuing customer relationships is critical to implementing this strategy."[4]

Characteristics A "case" can be defined by the following characteristics [PL11; Oce11]:

- It is a project, transaction service or response,
- it has a (well defined) start and end,
- it has as purpose to achieve resolution to a problem, claim, request, proposal, development or other complex activities,
- it is likely to involve multiple persons,
- it is likely to have multiple documents and messages,
- its business process will drive towards an outcome, conclusion or result and
- it can have multiple routes, options and alternatives which can be created ad-hoc.

Figure 2.2 shows a simple time line of a case.

[3] `http://www.cmsa.org.au`
[4] `http://en.wikipedia.org/wiki/Customer_relationship_management`

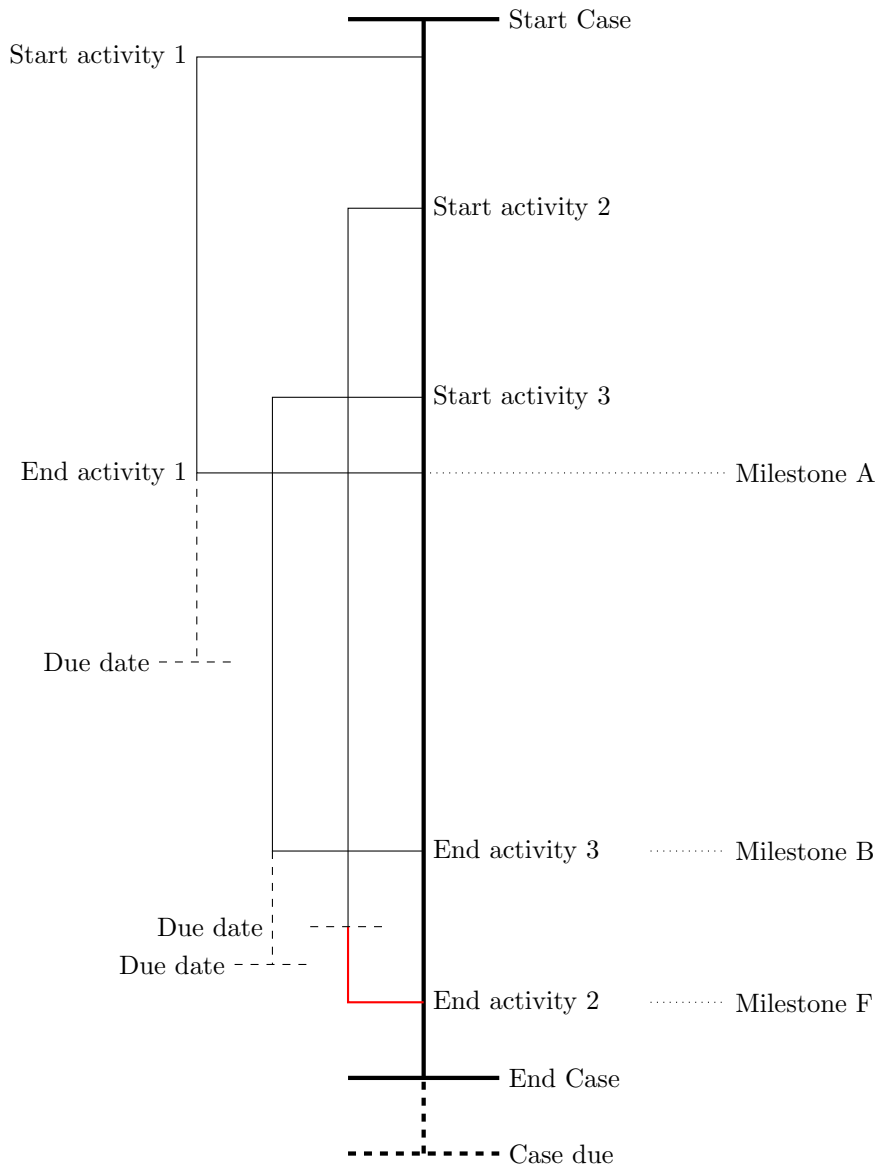


Figure 2.2: A (simple) visualization of the time line of a case, the elements should be supported by a case management system

Definitions A Dutch task force with members from municipalities and (software) suppliers came with the following definition of a case (in dutch) "Een zaak is een samenhangende hoeveelheid werk met een welgedefinieerde aanleiding en een welgedefinieerd eindresultaat, waarvan kwaliteit en doorlooptijd bewaakt moet worden." [Wer04] This definition will be used for the rest of the paper because it summarizes the most important characteristics.

Definition 1 *A case is a coherent quantity of work with a clear beginning and a well-defined result, of which quality and lead time must be monitored.*

Looking at the various definitions of case, case management and case management tools, the following definition of a case management tool will be used in this thesis.

Definition 2 *A case management tool is a (software) tool supporting knowledge workers to monitor the quality and the lead time of cases, quantities of work with a clear beginning and a well-defined result. A case management tool should also support the planning of cases and activities or processes attached to the case.*

The definition uses by Cordys implies that the traditional workflow systems are not capable in supporting cases because it emphasizes that the processes involved from beginning to result are not easily constrained to a process diagram.[Cor10]

2.1.1 Related systems

A workflow is, according to the Oxford Dictionaries, "the sequence of industrial, administrative, or other processes through which a piece of work passes from initiation to completion." Traditional workflow systems define the processes in advance in order to be able to delegate tasks and to monitor progress. Case management tools are more flexible because the processes do not have to be predefined, and if they are, they are allowed to change. In figure 2.3 the difference between workflows and cases is visualized. Several papers suggest that for workflow systems to reach their promised potential they have to be able to adapt to changing processes. [ADM00; KDB00; Kam+] The disadvantage of the traditional workflow systems is that either the system is very detailed to support the actual work (but risking that a small change in the process will make the tool unsuitable) or that the system is kept simple in order to support a simple, more generic processes (risking that the tool may not be able to support all steps). Both cases risk that the user will not make proper use of the system. In that case the system is not adding value to the process. [AB01]

Another related systems is a system that focuses on the storage, retrieval, logging and version control of (electronic) documents. The systems are called *document management systems*. A document management system (DMS) is a computer system (or set of computer programs) used to track and store electronic documents and/or digitized versions of paper documents. It is usually also capable of keeping track of the different versions of different users (history

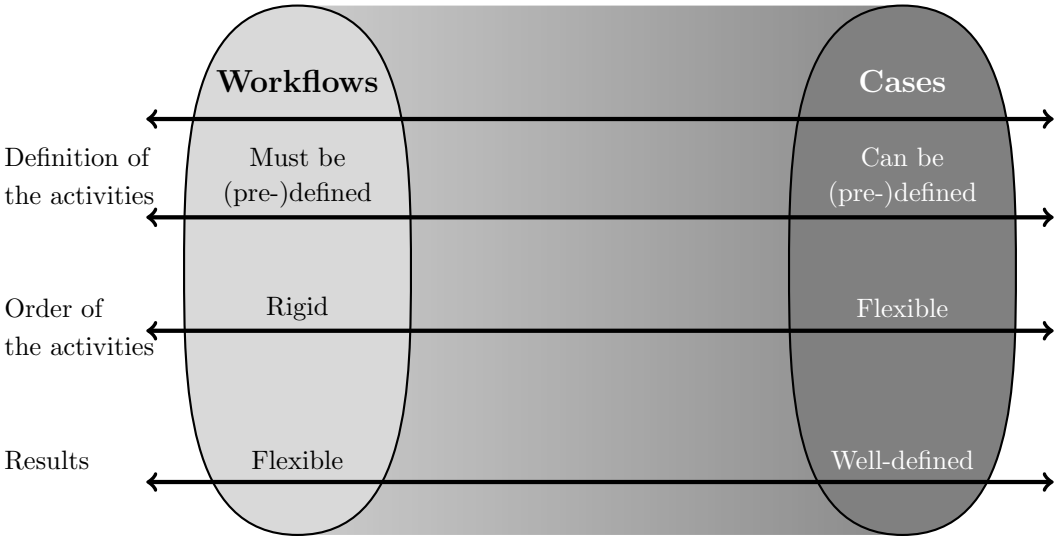


Figure 2.3: Landscape of workflow and cases

tracking). The definition has some overlap with the concepts of content management systems and is often viewed as a component of enterprise content management (ECM) systems and related to digital asset management, document imaging, workflow systems and records management systems.

In figure 2.4 the relation between cases, processes and documents is visualized.

2.2 Case Management Tools

Now that we have defined what a case management tool is, we can look to some products that are in the market. There are already multiple researchers and consultants which made lists with software for case management, in 2009 a (Dutch) consultant agency (M&I/Argitek as cited in [Dec]) looked at the Dutch market. In 2006, a nonprofit organization Idealware made a comparison of case management tools that were at the market at that time, but updated the list in 2011. Finally Strijbosch [Str11] looked during his thesis to multiple tools in 2011.

A merged list with case management tools [Dec; Str11; QL11]:

- Appian
- Brein InProces Zaken
- Centric Conductor / Key2Zaken
- Circle Verseon ® Zaaksysteem
- CiviCRM

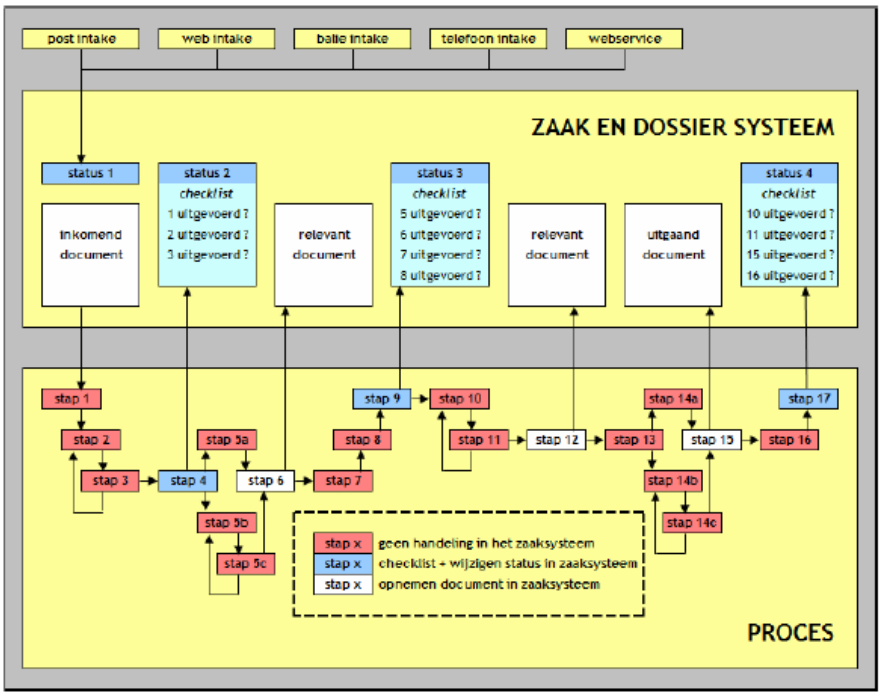


Figure 2.4: A visualization of processes, cases and documents. The white area's will be added to a DMS, the white and blue states will be added to a CMS, and all the step will be logged in a workflow management system. [Dek]

- Corsa case
- Decos D5 Zaaksysteem
- Differs Zaaksysteem
- DSI's ClientTrack
- (Excel, Access or FileMaker Pro)
- Groupion
- Kodision KIM® Zaaksysteem
- Logica GovUnited
- Pallas Athena Bomana
- PerfectView KCC-Zaaksysteem
- Pink Roccade
- Salesforce
- SarePoint
- Serenga (Océ)
- Social Solutions' Efforts to Outcomes
- Unicentric' Service Xpert Suite
- Zaaksysteem.nl (Mintlab)

2.2.1 Example tool

In appendix B a simplified case management tool is written in the functional language *Clean*[5] with the help of the *iTask*[6] toolkit. This is a toolkit for programming workflow management systems. The example program consists of 56 lines of code and supports case creation, delegating a task (or creating an activity), closing the activity and closing the case. This are the basic components of a case management tool, see also figure 2.2. This example is included to show the power of the functional language; as an experiment for making a program having the basics of a case management tool and for trying to get an impression whether it is possible to create a case management tool in *Clean*. The example shows that it is possible to create a case management tool in a functional language, but for a complete tool, supporting all the required possibilities more time is necessary for research and programming.

Figures 2.5 and 2.6 show two screenshots of this example.

[5] `http://wiki.clean.cs.ru.nl/Clean`
[6] `http://wiki.clean.cs.ru.nl/ITasks`

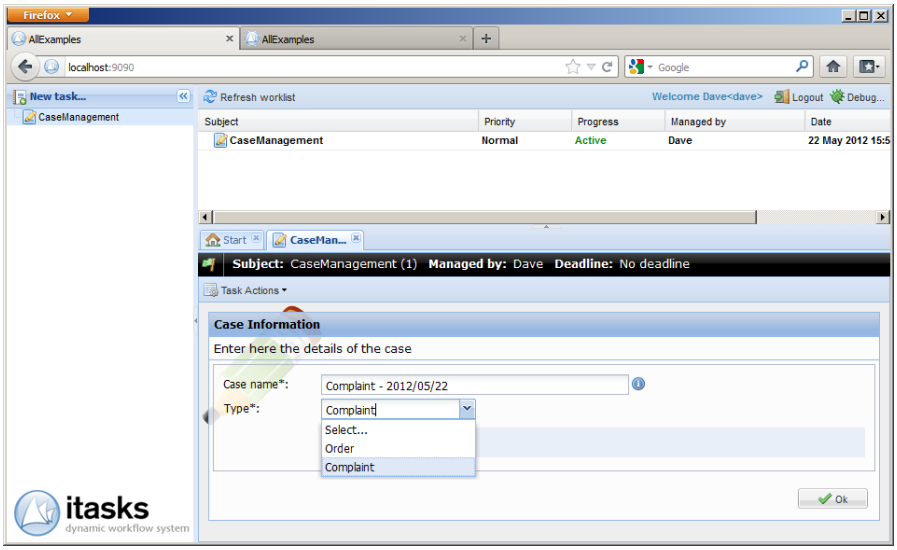


Figure 2.5: Enter the case information

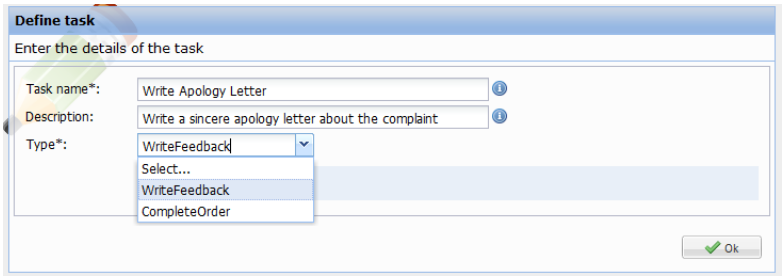


Figure 2.6: Define the Task

Chapter 3

Business Intelligence

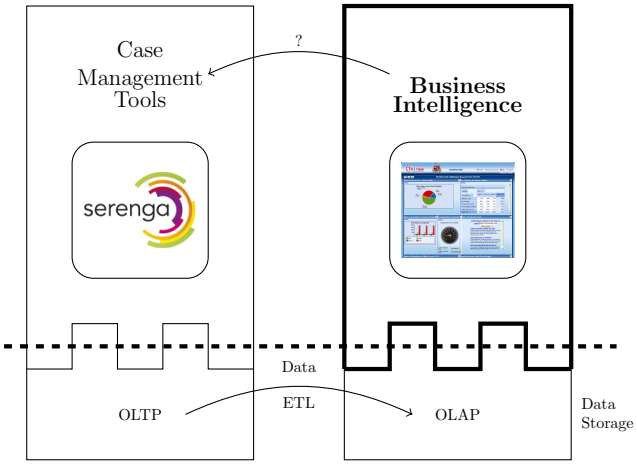


Figure 3.1: Business Intelligence in relation with the different concepts

In 1971 Gorry and Morton presented a framework for *management information systems* (or *management decision systems*). This framework was focused on the managerial activities in stead of on information systems.[GM71] This framework came after the first experimentations and research into systems supporting decision making and integrated different categories of management activity with different decision types. Some of these researchers from the late sixties were: Raymond (1966), Turban (1967), Urban (1967), Morton (1967) and Holt and Huber (1969). [Pow07] In the past 40 years there has been a lot of improvement and research to these kind of systems. They even changed the name a couple times. To understand what these systems do, who they help and how these systems can help case management, we describe the different concepts, give a small historic overview and discuss related, but different system (Competitive Intelligence). Because the systems can provide us with a lot of information (maybe too much)

we give a overview of possibilities to visualize this amount of information. Finally we conclude this chapter with the benefits of having a BI system for case management and how to create business intelligence.

3.1 Definitions

As mentioned in the introduction of this chapter, business intelligence exists about 40 years. The name changed from management decision systems to *decision support systems* to the term business intelligence.Of course the name was not the only thing that changed. The increasing capabilities of the computer systems may be one of the enablers of some of the changes. Data mining became possible as well as storage of 'all' historical data. Also visualization options made it possible to visualize data for example by making graphs or presenting multiple indicators on one dashboard.

Dan Powers divides the BI (or DSS) tools into five categories:[Wat09]

- Communication-driven
- Data-driven
- Document-driven
- Knowledge-driven
- Model-driven

For this thesis the data-driven and the document-driven tools are the most relevant. This will be explained in chapter 4

A definition of business intelligence that will be used for the rest of this thesis (based on[Wat09])

Definition 3 *Business Intelligence comprises applications, technologies, tools and processes that help knowledge workers to make better decisions by gathering, storing, accessing and analyzing data.*

Knowledge Facts are the basis of knowledge. Facts, or in other words *data*, are the foundation of making *factual-based decisions*. Data can lead to information, which in its turn can lead to knowledge. Data can be anything, an order, a phone call, an email, a recorded action etc. Companies sometimes believe that storing everything, thus creating a lot data, is the answer to every problem and that having a large amount of data would mean correct decisions will be suggested automatically.[DP00]

The next step is *information*, data becomes information because it contains also relevance and purpose. This extra value can be added by *contextualizing* (adding information regarding the

purpose of the data), *categorizing* (adding information about the key components of the data), *calculating* (adding information by means of mathematical or statistical functions), *correcting* (removing errors) or *condensing* (information by summarizing). [DP00]

The final step is *knowledge*. Knowledge is a concept which is much more difficult to grasp, almost everybody 'knows' what knowledge is, or has an idea about it. Therefore there is no clear definition. When describing knowledge, we talk about the combination of experiences, values, insight, context and information which help us to process new information or evaluating choices or comparing information. One thing that is easily seen from this description is that knowledge is inside people, and can not (easily) exist outside people. [DP00]

Interesting is a quote from an knowledge manager from Andersen Consulting: "We've got so much knowledge (...) in our Knowledge Xchange repository that our consultants can no longer make sense of it. For many of them it has become data".[DP00] Organizations have to make sure that they collect and disseminate only useful knowledge, something which was already known by Aeschylus (525 - 456 BC)[DP00]

> " Who knows useful things, not many things, is wise. "
>
> Aeschylus

Drucker writes over these concepts: "Information is data endowed with relevance and purpose. Converting data into information thus requires knowledge. And knowledge, by definition, is specialized" [Dru03, pp.101]

3.1.1 Benefits

Having Business Intelligences enables organizations to have an overview over the whole company and the environment. Integrating data from multiple sources, and presenting them in an understandable way, enables workers to make better (factual-based) decisions on time. The data is thus transformed into information. [Bur11; Sch]

3.1.2 Key Performance Indicators

To be successful some elements or processes must be running well in an organization. This can be monitored by looking at those indicators that give a reliable image of the performance of these processes. Being able to monitor those indicators gives a larger probability on success and high performance. The indicators are called *Key Performance Indicators* (KPI) and are almost the same as the *Critical Success Factors* (CSFs).[Wat09] The KPIs are often found on the dashboard of BI-tools. When defining the KPIs, the choice must not be influenced by the availability of data.[Bur11] In case there is no data available for presenting KPIs, changes should be made in the underlying systems so that the necessary data becomes available.

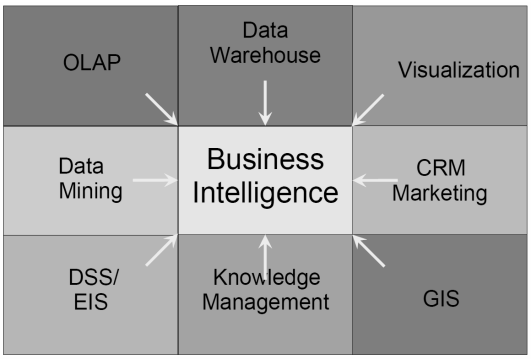


Figure 3.2: Relation between BI and other systems [Neg04]

3.1.3 Components

Business intelligence systems are usually considered to consists of multiple components. The basic and most used ones are the components that store the data (data-warehousing or data-marts) and that display the data. More advanced systems also consists of components like data mining, data visualization (see §3.2) or alerting services. In figure 3.2 the dependence on multiple components for a BI system is shown.

3.1.4 Usage

The usage of BI systems vary within different organizations or departments within organizations. It can even vary between people inside a department. A far from exhaustive list with use of BI-systems. [Neg04; Sch]

- Manage corporate performance
- Monitor business activity
- Keep track of customer relations
- Access management reports
- Create forecasts
- Evaluate different scenarios
- Ask Non-routine questions
- Identify exceptions/problems
- Compare (departments)

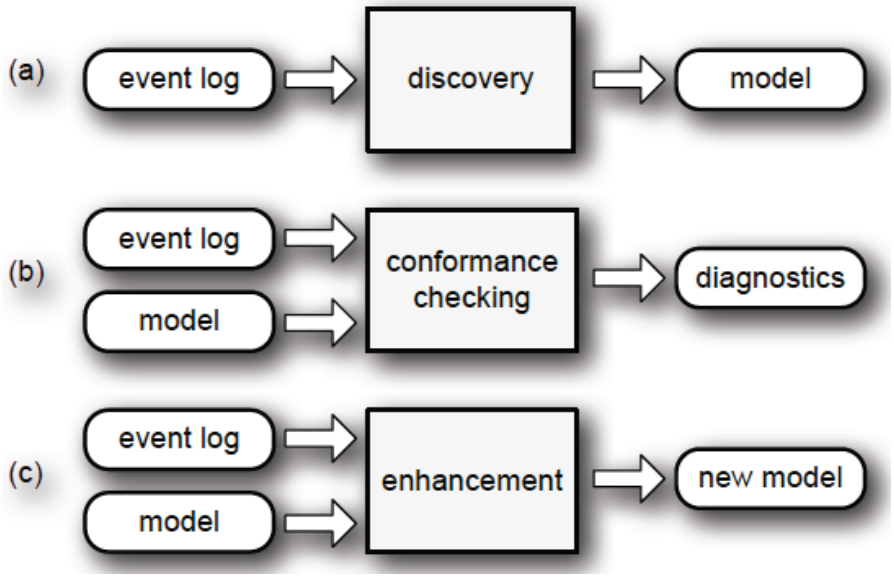


Figure 3.3: The different types of process mining (a) discovery, (b) conformance checking, and (c) enhancement [Aal+12]

Business Process Mining

As mentioned in the definition of business intelligence (ch. 3), the goal of BI is that knowledge workers can make better decisions by gathering, storing, accessing and analyzing data. *(Business) process mining* is about (business) processes and models of (business) processes. Creating or checking models of processes helps to create insight into the way the business works and can show processes that need optimization. According to Aalst et al. the gap between data and business process modeling and analysis can be bridged by process mining. [Aal+12] (Business) process mining on event logs can globally be done in three ways. By *discovery* a model is produced solely based on the event log. The second option is *conformance checking*, the real world (based on the event log) will be compared with the model. The result of conformance checking is an analysis or a diagnosis. The third type of process mining is *enhancement*. The difference between conformance checking and enhancement is that the result of conformance checking is an analysis about the differences (and similarities), enhancement strives to improve the (a-priori) model.[Aal+12] Figure 3.3 shows the different types of process mining.

3.1.5 Self-service BI

The number of people having to make decisions based on business intelligence has grown in the last years. One of the problems with enlarging the number of users for the BI system is that more business intelligence reports have to be created for satisfying the information need for all the

users. At first it was sufficient for an IT department to design a number of reports and views to support the information need for the limited group of users. A study conducted by consultancy agency Accenture found that sixty percent of the decisions were made while relying on gut feeling.[KK10] These decisions are probably made by users with experience. The experience of the user probably was the greatest contributor to this 'gut-feeling', but unfortunately this was not part of the research. These findings underline the need for more support for the workers at all the levels in the organization.

Acknowledging that work at lower (managerial) levels in an organization can also be information intensive, made the information need much larger. Besides the growth of the need for information, the information need became more volatile. The BI-tools needed to become more flexible and easier to use. The tools that supported workers in their day-to-day job mainly in reporting and analysis and they became know under the name *BI for the masses* or *Self-Service BI*[Neg04]

The ever faster changing environment demands systems that are able to incorporate new information quickly and workers that can act quickly and capable to these changes in the environment and available information. Delegating the generation of a new report to other departments (like IT) is then not suitable any more.[For11] The workers have to be able to generate those reports / find the information quickly in order to be able to use it for making sound decisions.

Self-service BI will enable knowledge workers to access information and fulfill their information needs without the support of others.

Because these knowledge workers are not specially educated or trained intensively to use BI-tools or read the results of reporting tools, the user interface as well as the display of the results should be intuitive and easy to understand. A portal or *dashboard* should enable users to get an quick overview of important values (KPIs) and allow customization and/or collaboration (e.g. share import views with colleagues). The user should not be confronted with (complicated) database designs, and easy search capabilities should be present.[For11] A search interface is especially important if there are a lot of *dimensions* (see §4.3.1). These dimensions can block users in finding the information they need.

3.1.6 Competitive Intelligence

Competitive intelligence (CI) is often mentioned together with business intelligence. Some say CI is always a part of BI while others claim that CI should be called BI, because 'business' suggest the total external environment and all aspects of a company's operation [Ett95]

Competitive intelligence is a process of knowing what the competitor does by tracking direct and indirect competitors in a range of fields. It enables the organization to monitor its own development, find opportunities, prevent (or limit) disasters. [Ett95; RS01]

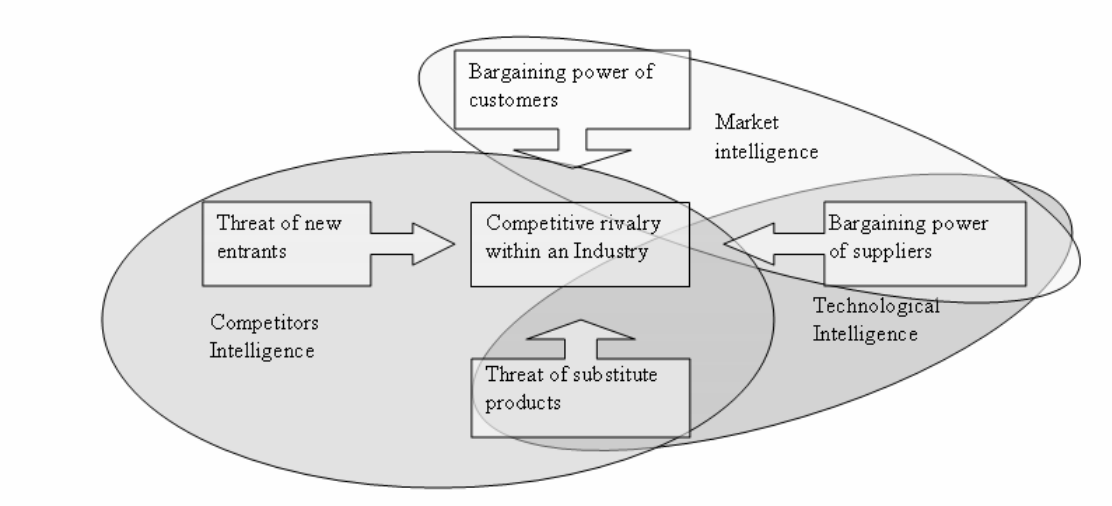


Figure 3.4: Porter's five forces model [Por79] combined with the Competitive Intelligence scope of DesChamps and Nayak [DN95]

John D. Rockefeller once said: "Next to knowing all about your own business, the best thing to know about is the other fellows business." This indicates the difference between BI and CI that will be used to address those concepts in this thesis.

Competitive intelligence is focused externally (everything outside the organization) while (the traditional use of) business Intelligence is focused on what happens inside the own organization (internally).

Competitive Intelligence can be categorized in various ways. One of this ways is the categorization by DesChamps and Nayak. Here, the categories *Market Intelligence*, *Competitor's Intelligence* and *Technological Intelligence* are distinguished.[DN95] This model can be combined with the five forces model developed by Michael Porter [Por79] about the governing competition forces. This is shown in figure 3.4

In 1986 Ghoshal and Kim published an article about intelligence system for competitive advantage. There they show the importance of having CI incorporated in the organization.([GK86]) In figure 3.5 is shown who should perform which role in CI.

3.2 Data visualization

Humans are limited in their ability to *receive, process* and *remember information* [Mil56] This holds especially for information that is presented sequentially. Techniques for increasing the ability to store (in the short-term memory) more information is using multi-dimensional dimensions (According to Miller an average person can process seven, plus or minus two, different values).

	Immediate Business Environment	Broad Business Climate
Information Acquisition	Line Managers	Intelligence Unit
Internal Circulation	Line Managers Intelligence Unit	Intelligence Unit
interpretation	Line Managers	Line Managers / Intelligence Unit

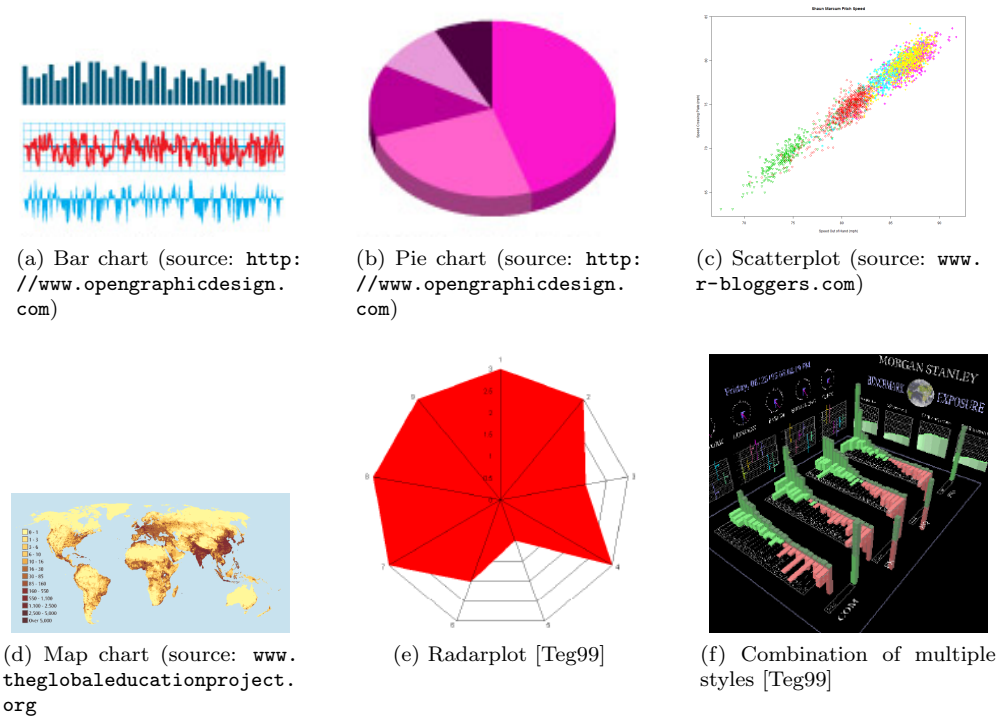
Figure 3.5: Role of Line Managers and Intelligence Units in Competitive Intelligence [GK86]

Another option is *information chunking*, the easiest example is seen in for example numbers (it is easier to remember a series of binary numbers if you chunk them as decimal numbers) and letters (it is easier to remember words than as a sequence of letters).[Mil56]

Because collecting information gives an enormous amount of data, in order for knowledge-workers to be able to do something with this data, the data should be presented in a way that the information needed can quickly be satisfied and answers can be easily extracted. Studies show that the quality of decisions improve when the presentation of the data matches the decision-making-task. [Ves91] There are roughly two type of questions. The first type of question involve discrete data extraction from the dataset (*Symbolic information*). Tables can give quickly answer to these kind of questions, an example is "How long did it take employer Y to perform task X". The second type of questions are those that require association or perceiving relationships (*Spatial information*). These questions benefit from a graphical visualization of the data, this are questions like "Which employer accomplishes task X the quickest?" [Ves91] This second type of questions solve problems regarding *explanatory* (search for structure or the creation and or testing of hypotheses), *confirmatory* (confirm or refute hypotheses) and *production* (creating a visualization-based report) [Teg99; GW02]

Some information (e.g. geographical information) can be visualized in a way that the visualization is analogue to the real world phenomena. Other information is more abstract and a visual analog must be created because there exists none.[Teg99]

Important to note is that visualization does not replace quantitative analysis. Instead it enables



(a) Bar chart (source: `http://www.opengraphicdesign.com`)

(b) Pie chart (source: `http://www.opengraphicdesign.com`)

(c) Scatterplot (source: `www.r-bloggers.com`)

(d) Map chart (source: `www.theglobaleducationproject.org`

(e) Radarplot [Teg99]

(f) Combination of multiple styles [Teg99]

Figure 3.6: Different visualizations

the user to focus the quantitative analysis. According to Grinstein and Ward visualization is "The graphical (as opposed to textual or verbal) communication of information (e.g, data, documents, structure)".[GW02]

Having a visualization of data enables the user to use the human visual system to *extract* information and to *identify* interesting parts of the data (e.g. by seeing structures, trends, anomalies, etc.) by presenting an overview over the dataset.[Teg99]

Typical forms of visualizing data are tables, line graphs, scatter plots (fig. 3.6c), bars and pie charts (fig. 3.6a and 3.6b. Some other visualization forms are radar plots (Kiviat diagrams, fig. 3.6e), volume renderings, surfaces, maps (fig. 3.6d), or a combination of these (e.g. a 3D visualization of a room with different visualization methods presented on the 'walls' and the 'floor', see also fig. 3.6f).

Edward Tufte suggests ten rules for creating clear and easy understandable data graphs.[1]

1. Show the data.
2. Use graphics. (only those that add information)

[1] `http://www.sealthreinhold.com/tuftes-rules/`

3. Avoid Chart junk (the visualization should not distract from the data that is shown) and
4. Utilize data-ink (there should not be unneccesary ink in the graph).
5. Use labels.
6. Utilize micro/macro (the clarity of the overview is determined by the details).
7. Separate layers
8. Use multiples (make the data easy comparable, e.g. by presenting two graphs side-by-side with the same scale on the axis)
9. Utilize color (but avoid colors that serve no purpose)
10. Understand narrative (show time and space according to expectations, e.g. jan-feb-march-april in stead of march-jan-april-feb)

Another important aspect is that visualization should, if possible, enable the possibility to view the data at different levels of detail (Drill-down/roll-up, see §4.3.1 ff.)

For each different measure/data that has to be displayed to the user, the visualization should fit the data.

3.2.1 Dashboard

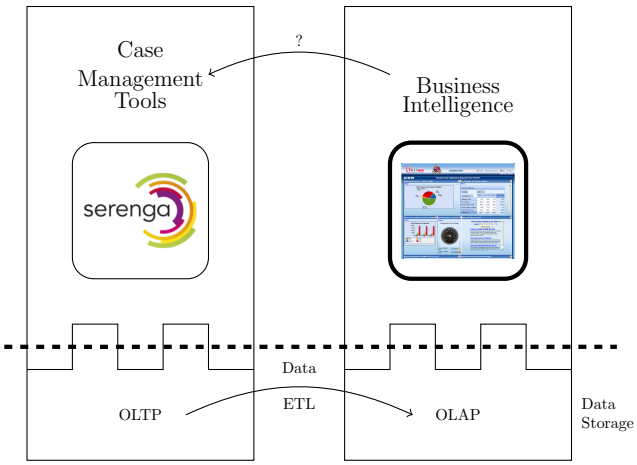
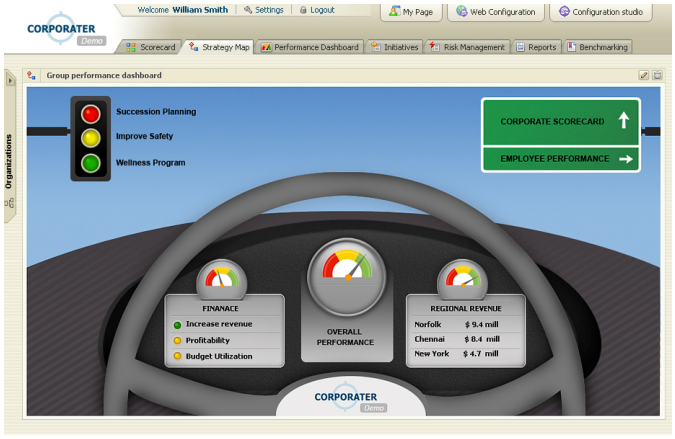


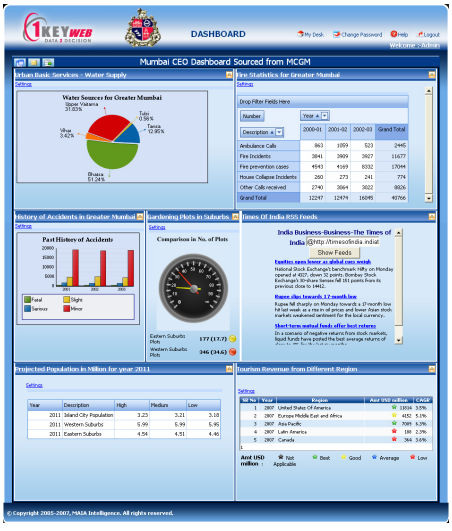
Figure 3.7: Business Intelligence dashboard in relation with the different concepts

A system that is often seen in business intelligence tools, but nowadays also in other tools, is a dashboard. A dashboard enables the user to get an overview of the most important data in one look. The name dashboard comes from the automotive industry, the dashboard in a car shows the most important values for the driver (speed, fuel level, revolutions, oil temperature, etc.).

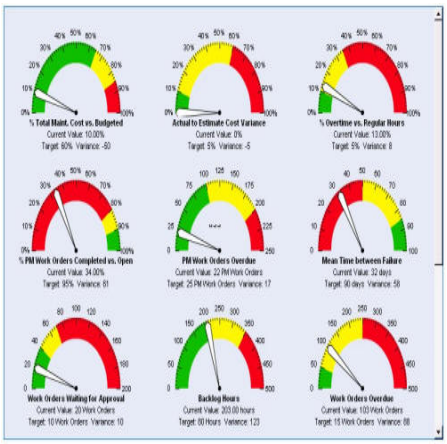
Figure 3.8a shows a dashboard interface which has much resemblance with the car origin and figure 3.8b shows an other dashboard interface. On this last dahsboard a lot of information can be displayed, both the spatial information (comparison, perceiving relationships) and symbolic information (the tables with data).



(a) Dashboard interface with close resemblance to a car (source: `http://www.corporater.com`)



(b) Dashboard interface with both tables and graphs (source: `http://blog.maia-intelligence.com`)



(c) This dashboard emphasizes the KPIs (source: `http://planner.zoho.com`

Figure 3.8: Three different dashboard designs

3.3 Tools

Business intelligence tools can consist of a range of components. The parts that are used every day by the knowledge worker (such as the dashboard), the systems underlying the dashboard and the tools to make the two type of systems mentioned earlier. BI-tools are tools to make applications and processes that help knowledge workers to make better decisions by gathering, storing, accessing and analyzing data. Most BI software has also tools to provide information to the system (e.g. the extract-transfer-load component for gathering the data).

Business intelligence tools can support the user globally on five levels, where each level is more sophisticated then the previous.[2]

1. Static reporting
2. Managed reporting with simple interactivity
3. Highly interactive reports and dashboards
4. Self-service reports and data analysis
5. Advanced analytics

3.4 Creating Business Intelligence

As already mentioned in the introduction, the original research goal was not possible. Instead the goal was changed to find the design steps necessary for creating a business intelligence module in case management tools withe the use of OLAP cubes. The steps (see figure 3.9) proposed in this section and tested in chapter 5 are based on the experiences when creating OLAP cubes which could be used to answer business intelligence questions and on literature [Bur11; GG85; Sch].

3.4.1 Design steps

First, the input has to be gathered. A best generic design for a business intelligence module could not be made because the best design is dependent on the the case management tool, the business and the business intelligence questions that will be asked and the data that is available in the case management tool. This is thus the required input for the next steps.

Characteristics of the case management tool The first step is determining what the case management tool is, how the data is stored and how the data is connected. The relation between the different objects determine what data could be extracted later and what the data means.

[2] `http://www.jaspersoft.com`

For example is the user that is connected to an activity the user that created the activity or the user who currently is tasked with the activity. Can cases have parent-child relations, or is there a sequential relation between cases such as customer → order → complaint → refund.

Information need The second step is determining what questions must be answered by the tool. This determines how data must be grouped later on. Also dimensions and hierarchies are influenced by these questions.

Available data The third step is to look at the available data. If answering some of the questions from the previous step is not possible, then the case management tool should be changed that the required data is stored in order to be able to analyze it. Other problems that can occur with the data is that some information is not stored (persistently) or that the expected relationship between data objects not is what it seemed to be.

Design of the OLAP Cube The fourth step is designing the OLAP cube. The OLAP cube consist of dimensions, hierarchies, measures, aggregations and relations between the dimensions and the measures. (In §4.3.1 these terms will be explained in detail).

Loading of the data The fifth step is determining how the data should be loaded. Possibilities are a normal batch (e.g. every night), high frequency batch (e.g. every hour or shorter) or real-time. The benefits of real-time are obvious but the costs are a lot higher than bath updates.

Presenting the data The final step is deciding whether the resulting information must be presented with predefined reports/dashboards or that the users can navigate through the information with self-service business intelligence. Usually users higher in the management hierarchy are best served with predefined reports that give a global overview of the business. A KPI scorecard can give a very brief summary of the current business. Figure 3.10 shows the intelligence pyramid. The higher in the pyramid the more there is information/knowledge. In the bottom layers there is only data, and time must be spend in order to create information from this data. Users lower in the management hierarchy often benefit from the freedom of self-service business intelligence. For using self-service business intelligence domain knowledge is more necessary than for using predefined reports.

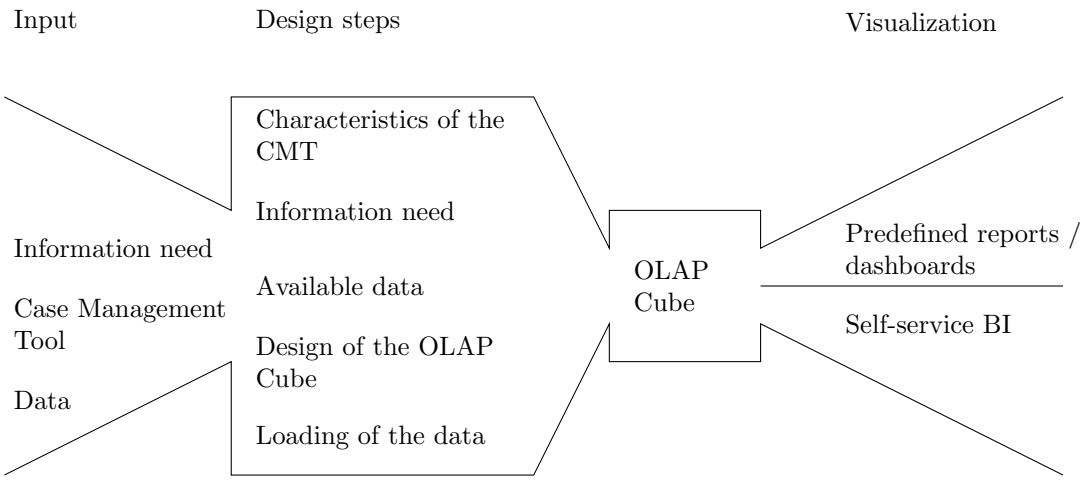


Figure 3.9: The design steps for adding business intelligence to a case management tool with OLAP cubes

Figure 3.10: The intelligence pyramid [Bur11, pp.8]

Chapter 4

BI in case management

After having discussed what business intelligence is (applications and processes that help knowledge workers to make better decisions by gathering, storing, accessing and analyzing data) in the previous chapters now we can look at how BI can support case management.

Grigori et al. identify in their article "Business Process Intelligence" three levels of automation for the management of process quality[Gri+04]. These levels of automation can be used to organize the information need for business intelligence in case management. First is the *analyzing*. In case management time and performance can be analyzed. For example the time certain activities take, or time between milestones. Also the performance of the user can be analyzed, both in time and in result (if different results are defined). The second level *prediction* can be seen for the allocation of users. If certain cases are open, and for later steps a particular user or group of users is necessary, than the number of users needed can be predicted. The following level, *monitoring*, has to do with the current state of the processes. In case management there are multiple measures that have to be monitored. Deadlines have to be monitored, but other elements like success rate of activities or time that an activity takes can also be valuable to monitor. The fourth level, *control* can take place for example to delegate tasks from one user to another user (e.g. because the first one has no time to handle the task). The final step, *optimization* is for example change processes within the workflow of a case. If the analysis show that a case or an activity is handled by a lot of users, then it might be the case that the information needed for finishing the case is scattered, and consolidating the information might reduce the time needed for finishing the case (or activity).

4.1 Data Storage

As mentioned on multiple occasions in chapter 3, business intelligence requires data which can be monitored, analyzed and used for prediction, control and optimization. So besides the question how do we present the data, the main questions are "*Where is the data ?*" and "*Can we use the original data directly or does it have to be extracted and transformed ?*". To answer these questions we look into the two different types of data storage: OLTP (optimized for operational databases) and OLAP (optimized for aggregating measures along different dimensions). A case management tool generally stores the data in an OLTP database, while a business intelligence tool benefits from storing the data in an OLAP database.

4.2 Data storage in Case Management: OLTP

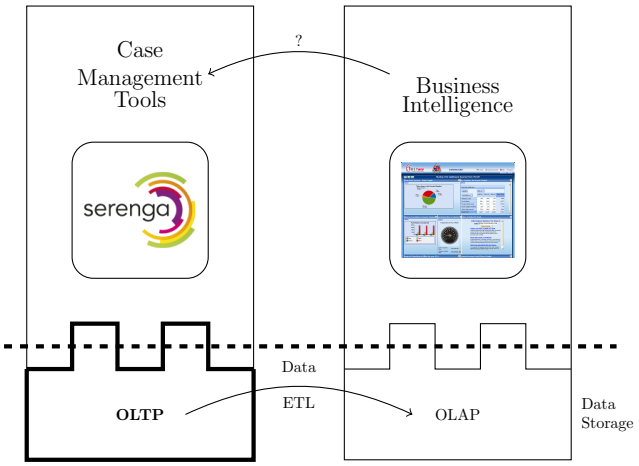


Figure 4.1: OLTP in relation with the different concepts

4.2.1 Informal description

Gray and Reuter have a definition of a transaction processing system, this definition encompasses more than is necessary, but is therefore more extensive.

Definition 4 *A* transaction processing system *(TP system) provides tools to ease or automate application programming, execution and administration. Transaction processing applications typically support a network of devices that submit queries and updates to the application. Based on these inputs, the application maintains a database representing some real-world state. Application responses and outputs typically drive real-world actuators and transducers that alter or control the state. The applications, databases and network tend to evolve over several decades. Increasingly, the system are geographically distributed heterogeneous (they involve equipment and software from many different vendors), continuously available (there is no scheduled down-time), and have stringent response time requirements. [GR93]*

The important part of this definition is *"submit queries and updates"*. A database that supports this operations often has the requirement that the transactions are *ACID* (atomic, consistent, isolated and durable). This means that they have to be:
Atomic: The transaction should be a complete success or a complete failure. In other words, the system must be able to roll-back to the state before the transaction when (a part of) the transaction fails.
Consistent: Every state a database is in, must be a valid state. The state before and after a transaction must be valid to all the invariant properties (such as integrity constraints).
Isolated: Transactions that affect the same data should not be allowed to run concurrently,

but should run sequentially instead (often implemented by locking the affected rows during a transaction).
Durable: Changes should hold. The effects must be persistent.

Normalization An OLTP-database is (nowadays) often a relational database. In a relational database data is stored in multiple tables, and - when the database is normalized - data is stored in only one place. If data is not stored normalized, then data is often stored redundant in the database. The benefit of storing everything (non-normalized) together is that retrieving the information is simple, but the disadvantage is that it costs more storage and updating information is more difficult. Consider for example a database with personnel and their salary. If the salary changes due to new inflation correction or a new collective bargaining agreement all records in the table have to be updated. In case of normalization the personnel table does not contain direct the salary amount but only a reference to a table with salaries. Now only the distinct salary values have to be updated.

Relational databases often are normalized (see figure 4.2), the benefit of normalization is that updating (and to a lesser extent inserting) values can be quick because only a small number of alterations have to be made as we showed in the example. An extra benefit is saving disk space. In a non-normalized form a database has a lot of data is that is duplicated (or redundant). Converting the design of a not-normalized database to a normalized form can save a lot of disk space. Instead of storing for example a description field of 255 bytes in comparison to an small integer (2 bytes) as id field can save almost 90% if there are 10000 entries of which there are 1000 unique entries. $\left(1 - \frac{1.000 \cdot 255 + 10.000 \cdot 2}{10.000 \cdot 255} = 89,22\%\right)$

The *first normal form* (NF1) has no repeating (groups of) elements (atomicity) and every row of data should have an unique identifier. [Cou09] This enables querying (e.g. select all products from department X).

In a database is in the *second normal form* (NF2) there can't be any partial dependencies on a (concatenated) key. Instead, all columns that have a partial dependency should be stored in a different table.

For a database to reach the *third normal form* (NF3), all other dependencies should also be removed from the tables (except the dependencies on the primary key).

4.2.2 Operations

Different Database operations are:

- Union

- Intersection
- Difference
- Cartesian product
- Restriction / selection
- Projection
- Join
- (relational) division
- Aggregate

The first four operators are basically the same as the (mathematical) set operations.

Union The union operator combines the two input sets into one output set. The tuples that exist in both the input sets (intersection) only occur once in the output set.(see figure 4.3a)

Intersection The intersection operator returns the tuples that exist in both input sets. (see figure 4.3b)

Difference The difference operator returns tuples of the first (input) set that do not exist in the second (input) set. (see figure 4.3c)

Cartesian product The cartesian products creates tuples of every element in the first input set with every element in the second input set. The number of tuples returned is the multiplication of the number of input tuples in the input sets. (see figure 4.4a)

Restriction / Selection The restriction (or selection) operator limits the number of tuples so that only those (input) tuples that meet the requirements are included in the output. (see figure 4.3d)

Projection The projection operator removes certain elements from the tuples. In terms of relational databases the number of retrieved columns can be reduced.

Join The join operator combines the input tuples based on a relation. The output tuples contain more elements (basically the sum of the number of items in the input tuples minus the element(s) the relation is based on). If the input is {Name, Adress, Office} and {Office, Department, Phone, Company} and the relation is based on the value of 'Office' than the result will be: {Name, Adress, Office, Department, Phone, Company}

(relational) Division The (relational) division operator limits the number of tuples from the input set (by means of a special restriction) and reduces also the elements of those tuples. The result is a set with unique tuples for which hold that in the input set there exists a relation with all the items from the divisor. This operation is the inverse of the Cartesian product, but can omit some values if they are not complete. (see figure 4.4b)

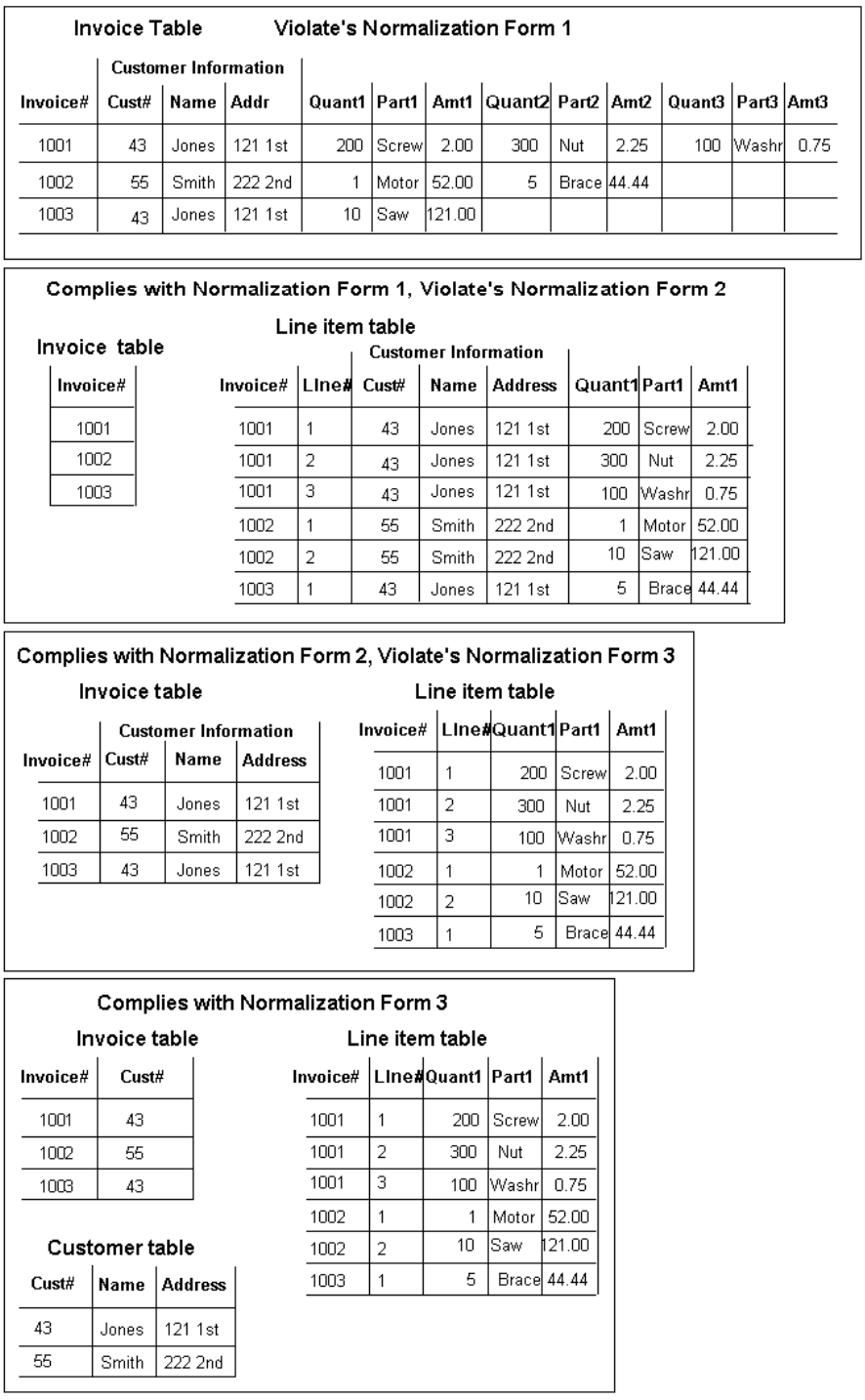
Aggregate The aggregation is a special operator, it is a combination of the *projection* operator and a aggregate function. The aggregate function is executed on the elements of the tuples that are left after projection. Aggregation is for example used to calculate the SUM of the values in the 'costs' column or to find the most favorable bid with the MIN function.

4.2.3 Conceptual model

For modeling and designing a database a number of conceptual models can be used, some of these are *Entity-Relationship*, *Object Role Modeling* and *Unified Modeling Language*. There are differences between the models, but the basic idea is that the relation or role between different (preferably real-world) objects or entities is visualized. One of the biggest differences between ER or UML and ORM is that the ORM does not have attributes; attributes are also objects in ORM. E.g. an entity `case` can have the attributes `case name, case type and taskfield` in ORM this would be different objects `case, case type and taskfield` with roles connecting the objects. To give an explanation of any of the methods would goes beyond the scope of this thesis. A small introduction to ORM will be given in order to compare OLTP and OLAP databases and describe a transformation (§ 4.4).

In ORM the main concepts are *objects* and *roles*. Objects represent both entities as well as their attributes. The relation between entities and (other) entities as well as the relation between entities and the attributes are the roles. The relationship can be of arbitrary arity (e.g. unary, binary, ternary, etc.). On these roles constrains can be added. Some possible constraints are `uniqueness`, `mandatory-role`, `disjunctive`, `equality` and `exclusion`.

In figure 4.5 a part of a case management system is visualized. For this visualization the rules that describe the model are listed in table 4.1.



Invoice Table — Violate's Normalization Form 1

	Customer Information											
Invoice#	Cust#	Name	Addr	Quant1	Part1	Amt1	Quant2	Part2	Amt2	Quant3	Part3	Amt3
1001	43	Jones	121 1st	200	Screw	2.00	300	Nut	2.25	100	Washr	0.75
1002	55	Smith	222 2nd	1	Motor	52.00	5	Brace	44.44			
1003	43	Jones	121 1st	10	Saw	121.00						

Complies with Normalization Form 1, Violate's Normalization Form 2

Invoice table

Invoice#
1001
1002
1003

Line item table

		Customer Information					
Invoice#	LIne#	Cust#	Name	Address	Quant1	Part1	Amt1
1001	1	43	Jones	121 1st	200	Screw	2.00
1001	2	43	Jones	121 1st	300	Nut	2.25
1001	3	43	Jones	121 1st	100	Washr	0.75
1002	1	55	Smith	222 2nd	1	Motor	52.00
1002	2	55	Smith	222 2nd	10	Saw	121.00
1003	1	43	Jones	121 1st	5	Brace	44.44

Complies with Normalization Form 2, Violate's Normalization Form 3

Invoice table

	Customer Information		
Invoice#	Cust#	Name	Address
1001	43	Jones	121 1st
1002	55	Smith	222 2nd
1003	43	Jones	121 1st

Line item table

Invoice#	LIne#	Quant1	Part1	Amt1
1001	1	200	Screw	2.00
1001	2	300	Nut	2.25
1001	3	100	Washr	0.75
1002	1	1	Motor	52.00
1002	2	10	Saw	121.00
1003	1	5	Brace	44.44

Complies with Normalization Form 3

Invoice table

Invoice#	Cust#
1001	43
1002	55
1003	43

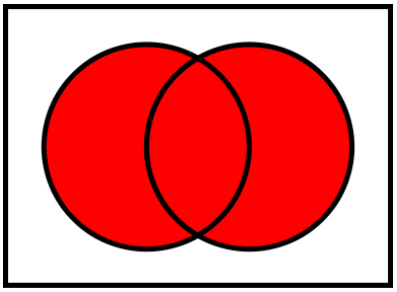
Line item table

Invoice#	LIne#	Quant1	Part1	Amt1
1001	1	200	Screw	2.00
1001	2	300	Nut	2.25
1001	3	100	Washr	0.75
1002	1	1	Motor	52.00
1002	2	10	Saw	121.00
1003	1	5	Brace	44.44

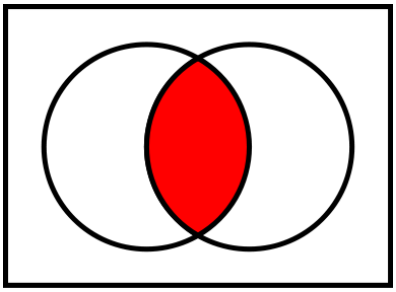
Customer table

Cust#	Name	Address
43	Jones	121 1st
55	Smith	222 2nd

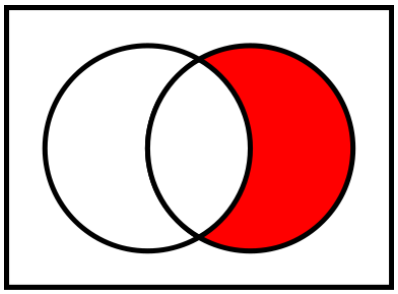
Figure 4.2: Database normalization



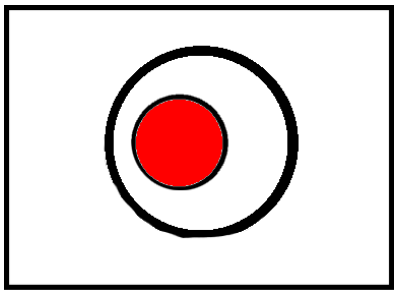
(a) Venn diagram of the union operator (source: `http://wikipedia.org`)



(b) Venn diagram of the intersection operator (source: `http://wikipedia.org`)

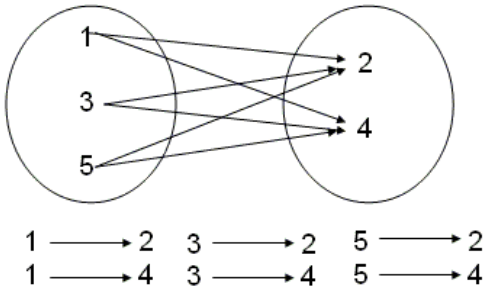


(c) Venn diagram of the difference operator (source: `http://wikipedia.org`)



(d) Diagram of the restriction operator

Figure 4.3: Visualization of the union, intersect, difference and restrict operations for OLTP databases



(a) Visualization of the Cartesian Product (source: `http://kwiznet.com`)

Person + Interests

Name	Interests
John	Music
John	Books
John	Cars
Lisa	Music
Lisa	Animals
Tim	Music
Tim	Books

Interesting Interests

Interests
Music
Books

Persons with interesting interests =
Person + Interests /
Interesting Interests

Name
John
Tim

(b) Visualization of the (relational) division operator.

Figure 4.4: Visualization of the Cartesian Product and (relational) division operations for OLTP databases

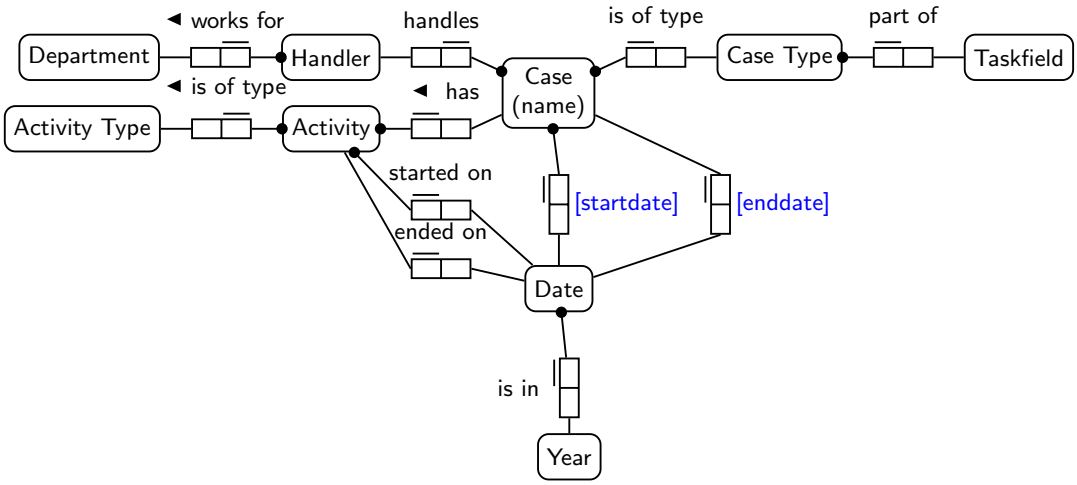


Figure 4.5: Example of an ORM model, the important objects are Case, Activity, Handler and Date.

A `case` (identified by a case name) is of type `case type`
A `case` belongs to one `case type`
A `case type` can be associated with multiple `cases`
A `case type` belongs to one `taskfield`
A `taskfield` can be associated with multiple `case types`
A `case` is started on `date` startdate
A `case` can be closed on `date` enddate
A `case` can be associated to `activities`
A `case` is handles by one `handler`
A `handler` can handle (multiple) `cases`
A `handler` works for `department`
A `department` employ `handlers`
An `activity` belongs to one `case`
A `case` can have one or more `activity`
An `activity` is started on a `date`
An `activity` can be closed on a `date`
An `activity` is of type `activity type`
An `activity type` can be associated with multiple `activities`
A `date` is in `year`

Table 4.1: The rules specifying the ORM model above

4.3 Data storage for Business Intelligence: OLAP

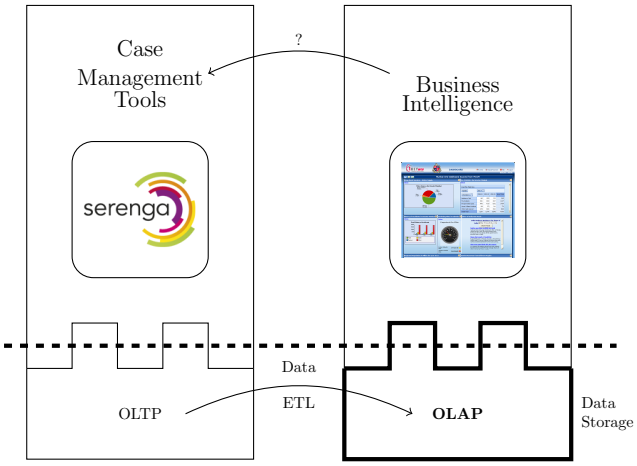


Figure 4.6: OLAP in relation with the different concepts

Besides the relational databases, which are very capable for OLTP tasks, knowledge workers (executives, managers and analysts) have a need for systems that supports them in making decisions. These so-called *decision support systems* base their support on aggregated data from the past (see § 3.1). This data can easily grow to sizes of gigabytes or even terabytes. Relational databases are optimized for online transaction processing [KS95, pp.92]. Using these databases for answering typical decision support questions can take a long time. Questions like "Am I profitable in store X" is a typical question, but can require enormous amount of resources because it usually has to join multiple tables, and perform multiple calculations on the results. (e.g. summing all costs and summing all profits). This is especially the case when the database is fully normalized (e.g. NF3). The OLAP databases are optimized for querying the database and presenting the data in contrast to OLTP databases which are optimized for a large number of concurrent CRUD (Create, Read, Update and Delete) transactions. [JJM08]

Historical data - stored for making business intelligence (BI) reports and supporting decisions - is often stored in Data Warehouses. This data is transferred from the operational databases to the *data warehouse* in order to keep a (relative) small and fast operational database. A common used definition of a data warehouse comes from Immon ([Inm05, pp. 29]):

Definition 5 *A data warehouse is a subject-oriented, integrated, nonvolatile, and time-variant collection of data in support of management's decisions.*

Such a system should have at least the ability "to consolidate, view, and analyze data according to multiple dimensions, in ways that make sense to one or more specific enterprise analysts at any given point in time." [CCS93, pp.4] A generic name for systems with these (kind of) requirements is online analytical processing (OLAP).

These requirements have an impact on the design of the database as well as the operations that are possible. This section will continue with a informal description and the requirements for OLAP. This section then continues with by describing how the data is stored in order to be easily used for BI and the important operations will be described.

One of the definitions found in [Cou95] of the term OLAP is:

Definition 6 *On-Line Analytical Processing (OLAP) is a category of software technology that enables analysts, managers and executives to gain insight into data through fast, consistent, interactive access to a wide variety of possible views of information that has been transformed from raw data to reflect the real dimensionality of the enterprise as understood by the user.*

The interactive access aspect from the definition suggest that you can navigate through the data and calculations in and across dimensions hierarchies and members. The wide variety of views and transformed data also suggest consolidated and aggregated data as well as grouping of information on different levels from detailed to an overview.

4.3.1 Informal description

As mentioned before, the OLAP databases should store a lot of information in a way that enables easy retrieval of the information - but more importantly - also lets you browse the information along multiple dimensions (§ 4.3.1) of the data. The information stored in OLAP databases will be split in facts which have measures and dimensions which give context to the facts.

The data used in typical OLAP databases have multiple sources (e.g. sales record, customer relationship management databases, ERP-systems). At the time of loading the data, the data has to be extracted from the sources, transformed into homonymous data and finally loaded into the destination database. This process (ETL) is described in a later section.

Measures

Measures are the (often numeric) part of facts that contain the data that can deliver information to the user. A common measure is ‘sales’ or ‘(order) quantity’. Measures are often numeric values that can be aggregated. When measures are not numeric or non-aggregative, like phone numbers, they often are better placed as an attributed to a dimension.

The granularity of the data is determined by the dimensions; it can occur that the source data contains more detailed information (e.g. sales data per transaction) while the detailed level for the appropriate dimension is at a higher level for example days instead of transaction. The measures should then be aggregated up to the appropriate level.

The three most important facts come from: [KR02; PJ01]

- **Events**: Modeling real-world events. (e.g. sales transaction)
- **Snapshots**: Modeling an entity's state at a given point in time. (e.g. an inventory count)
- **Cumulative snapshots**: Handling information about activities up to a certain moment (moment of the snapshot). (e.g. Total sales of this year up and including the current period)

According to Malinowski and Zimnyi the focus of the analysis is represented by the facts. The specific elements of the analysis are often represented by measures of the facts.[MZ04]

Dimensions

Dimensions give context to the fact and measures, making information out of data. Dimensions have attributes that are often textual and should (ideally) be descriptive. The dimension and attributes are both used for performing the characteristic operations on OLAP databases such as *slicing*, *dicing*, *drill-down* and *roll-up*. A dimension can have many attributes.

The attributes of the dimensions that can be used for exploring the data are called *dimensional levels*, *dimension attributes* or *category attributes* [MZ04]. Attributes that give extra information about the data, but which are not suitable for browsing OLAP cubes - e.g. the values are unique - are often called *property attributes* or *non-dimensional attributes*. Adding more attributes to a dimension increases the possibilities to navigate through the information.

As mentioned earlier the granularity of the dimension should match the granularity of the facts. If the dimensions are more detailed, then drilling-down to the most detailed level leads to empty cells in the cube. In the other case, if the facts are more detailed, then the information cannot be stored in the OLAP cube, the data first has to be aggregated to the most detailed level of the dimensions.

According to Malinowski and Zimnyi, the dimensions should include those attributes which allows users to navigate to different perspectives of analysis exploratory.[MZ04]

Hierarchy

Dimensions and hierarchies are very important for the OLAP databases, in order to drill-down and roll-up hierarchies must be defined. Hierarchies define the order in which drill-down and roll-up should work. For example the dimension 'date', which is often used in OLAP databases, has (among others) the hierarchies: Year - Quarter - Month - Day and Year - Week - Day. Both hierarchies are related to the dimension 'date' but cannot be combined (because a week can be part of multiple months). If a dimension has multiple hierarchies one or more levels of these hierarchies can be shared (in this case Year and Day).

Malinowski and Zimnyi makes a distinction between different categories of hierarchies [MZ04].

- Simple hierarchies (tree representations)
  - Symmetric
  - Asymmetric
- Non-strict hierarchy
- Multiple hierarchies
  - Inclusive
  - Alternative
- Parallel dimensions
  - Dependent
- Symmetric
- Non-covering
  - Independent

In symmetric simple hierarchies there is only one available path in which all levels are mandatory. In the asymmetric simple hierarchy the requirement that all levels are mandatory does not exist.

In non-strict hierarchies, there is no requirement stating that the link between parent and child levels should have a one-to-many cardinality. An example is the relation between departments and personnel. A person can work for multiple departments and departments have more than one employee.

Multiple hierarchies arise when some levels of two (or more) hierarchies are shared like in the case of Year - Quarter - Month - Day and Year - Week - Day. The difference between the multiple inclusive and the multiple alternative hierarchies is that in case of the multiple inclusive hierarchies the nodes at and above the splitting node have the same parents. In the case of multiple alternative hierarchies this is not the case.

We call hierarchies parallel if a dimension has multiple hierarchies for different analysis criteria. These parallel hierarchies are one of the hierarchies mentioned before. If some levels are shared then the parallel hierarchy is dependent on the other, otherwise they are independent. In figure 4.7 the meta model is shown.

Cubes

Perhaps the most characterizing of OLAP databases are the cubes that can be formed. The way n-dimensional data can be stored can be visualized as an n-dimensional cube. The real

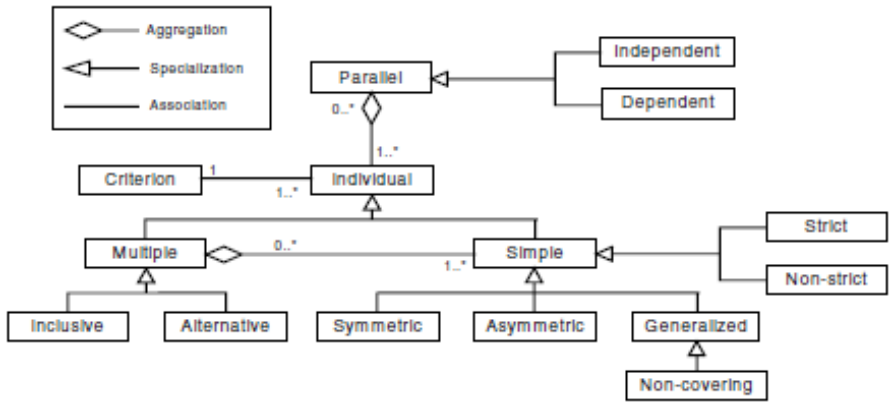


Figure 4.7: Metamodel of hierarchy classification. [MZ04, pp.486]

implementation of OLAP databases can be divided into three categories of storage. In figure 4.8 a 6 dimensional cube is illustrated.

- In a relational database (ROLAP),
- In a multidimensional database (MOLAP),
- In a hybrid storage solution (HOLAP).

Cube Storage

Zhao, Tufte, and Naughton have made a comparison between the performance of ROLAP and MOLAP databases. This comparison was made in 1996 and showed that, if the density of the data is larger than 1% MOLAP databases outperform ROLAP databases both in speed as well as size.

Since this report much research has taken place into optimization of ROLAP databases (among others because they are relatively easy to implement [Vas98]).

MOLAP performs poor when data is sparse [MI06] (according to Zhao, Tufte, and Naughton only up to 1% [ZTN96]) but most ROLAP techniques works only good with flat data (data whithout hierarchies).[MI06] Some techniques have been researched in order to be able to store both sparse and non-sparse data cubes in an efficient way as well as calculating aggregations in an efficient way. [MI06; BR99]

ROLAP databases are better / more efficient at storing sparse data because there is no need to reserve space for data that is not available. That the ROLAP only performs better when the density drops below 1% is due to the extra compression MOLAP uses, which can decrease the size quite significantly. (Without compression the density boundary for when MOLAP becomes better than ROLAP lies around 25%)

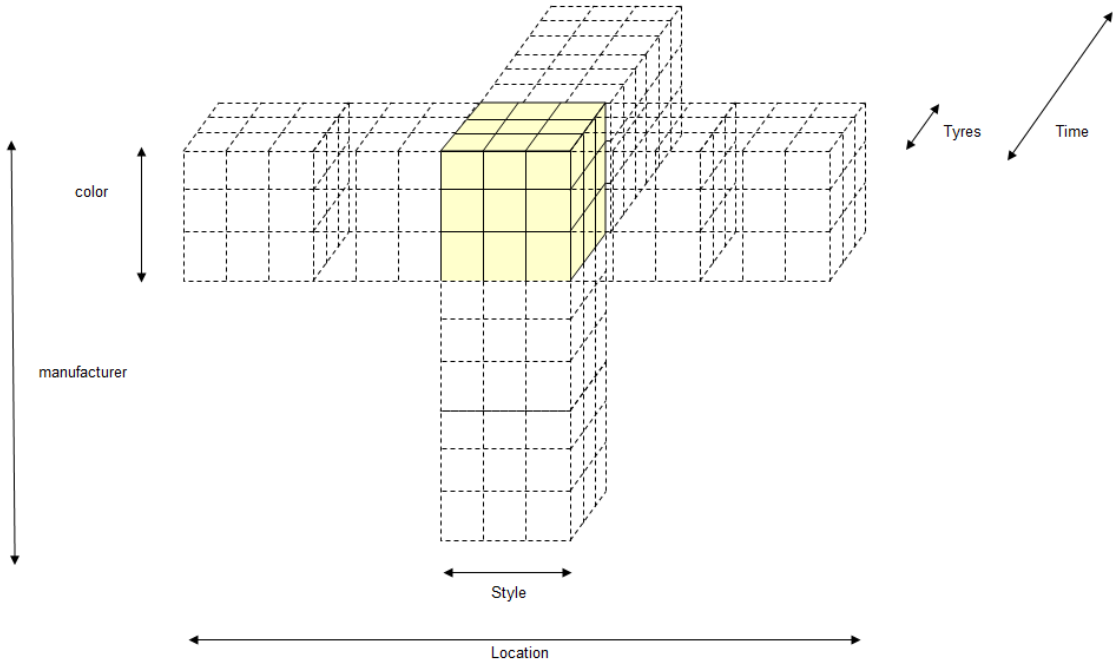


Figure 4.8: Possible visualization of a cube with 6 dimensions (e.g. car sales)

4.3.2 Requirements

Codd, Codd, and Salley have defined twelve requirements for OLAP products for which they can be evaluated. [CCS93, pp12]

1. **Multidimensional conceptual view** Multidimensional conceptual views are closer to the analysts view of the business universe and it makes manipulations like *slice-and-dice*, *pivoting* and *rotating* more easily.

2. **Transparency** Transparency preserves the productivity and proficiency. It should be clear whether OLAP is, or is not, part of the front-end. Also it should be clear whether the data comes from a homogenous or a heterogenous database.

3. **Accessibility** The OLAP tool should map its logical schema to other (heterogenous) data stores and present the data in a single coherent and consistent user view (sometimes by loading and transforming data from the other data sources).

4. **Consistent reporting performance** Even when increasing the number of dimensions or increasing the database, the performance should stay consistent. If the performance decreases, users are inclined to not use the OLAP tools to their full extend.

5. **Client-Server architecture** The data is often not stored on the same systems as the system from which the data is accessed.

6. **Generic dimensionality** There should not be any bias towards a specific data dimension.
7. **Dynamic sparse matrix handling** "Rather than basing a physical schema upon cells, records, two dimensional sheets, or some other similar structure, OLAP tools must dynamically adapt the models physical schema to the indicated dimensionality and especially to the data distribution of each specific model." [CCS93]
8. **Multi-User support** The system should continue to work when multiple users work at the same time with the system.
9. **Unrestricted cross-dimensional operations** Relationships in an OLAP model (such as 1:1 or 1:M) should be inferred by the tool without interference by an analyst. Of course the calculations between the different dimensions or levels in hierarchies should be complete in order to explore the cube freely
10. **Intuitive data manipulation**Navigational operations like drilling down across columns or rows or zooming in/out should be directly accessible. These operations should not be hidden in for example a menu
11. **Flexible reporting** In order to group data, to make visual comparison easier, the presentation of the data must be flexible (e.g. 0 to n dimensions on a row, column or page heading)
12. **Unlimited Dimensions and Aggregation Levels** The system should be able to accommodate at least 15 to 20 data dimensions in the system.

Kimball and Strehlo state that making a distinction between a central fact table (which can be very large) and surrounding smaller (dimensional) tables which is understood by the end users as representing the natural dimensions of the business is key to dimensional modeling. [KS95, pp.93]

Requirements for indexing

Sarawagi has stated the following requirements for indexing OLAP databases [Sar97]:

- **Symmetric partial match queries** Queries in OLAP can vary much. It can for example select a specific time period, or compare different time periods, or the comparison of items in one or more categories, or selections on measures (e.g. select the top-5 items from each category sold in January this year compared to last year per hour)
- **Indexing at multiple levels of aggregation**
- **Multiple traversal orders** Just as binary trees - often used for retrieving sorted data in OLTP databases - there must be a ordering mechanism that allows many group-bys
- **Efficient batch update** OLAP data is often inserted in batches

- **Handling of sparse data** Data in OLAP databases can be sparse, this should be considered.

4.3.3 Operations

In this section the operations mentioned before will be explained. The examples are partly from [JJM08]

Selection The *selection* (also known as *restriction*) operator restricts the results. Restriction can take place on the data or on levels of dimensions, further restricting the data can take place on one or more dimensions. [LT09; DT97; DT01]

Slice The *slice* operator limits the result by creating a subset of the values for a dimension. For one dimension only the selected level remains. The main difference with the selection operator is that it is not allowed to restrict only on dimensions and not on the measures. [JJM08; LT09]
E.g. from "For each individual store, show separately the number of product units sold for each product category during workdays and during holiday/weekend days" to "For each individual store, show separately the number of product units sold for the camping each product category during workdays and during holiday/weekend days"

Dice The *dice* operator is often combined with the slice operator under the term slice-and-dice. The main difference with the slice operation is that it is a restriction on multiple dimensions. A dice operation combines multiple slice operations. It can occur that there is only one dimension that is reduced, if it is reduced to one level, then it is a slice operation. When it is reduced to more than one value it is a dice operation. [JJM08; LT09]
E.g. When we only select January from the cube in figure 4.9 it is a slice operation, if we select also February then it is a dice operation.

Pivot The *pivot* operation aggregates the data with two or more grouping dimensions. The values of the dimension grouping attributes will become the row or column headers.

Drill-down The *drill-down* operation (also known as the *roll-down* operation) changes the granularity of the data. The result of the drill-down operation provides information with a higher level of detail than the original query. The drill-down operation navigates downwards in a hierarchy.
From "For each individual store, show separately the number of product units sold for each product category during workdays and during holiday/weekend days" to "For each individual

	Type	[LW96]	[DT97]	[JJM08]	[LT09]	[LT09]
Selection	Basic	-	X (Restriction)	-	X	X (restriction)
Slice	Basic	X	-	X	X	-
Dice	Basic	-	X (Partition)	X	X	-
Pivot	Basic	-	-	X	X	-
Drill-down	Basic	-	-	X	X	X
Roll-up	Basic	-	-	X	X	X
Aggregate	Basic	X	X	-	-	X (Merge)
Roll	-	X	-	-	-	-
Rename	-	X	-	-	X	-
Add Dimension	-	X	-	-	-	-
Transfer	-	X	-	-	-	-
Union	Binary operator	X	X	-	X	-
RC-Join	-	X	-	-	-	-
Difference	-	-	X	-	-	-
Pull / Push	Transformation	-	X	-	-	X
Partition	-	-	X	-	-	-
"Traditional Relational	-	X	-	-	-	-
Operations"						
Join	Binary operator	-	X	-	X	X
Cartesian product	Binary operator	-	X	-	-	-

Table 4.2: Overview of OLAP operation as described by multiple authors

store, show separately the number of product units sold for each product category and within each product category for each individual product name, during workdays and during holiday/weekend days"

Roll-up The *roll-up* operation (also known as the *drill-up* operation) is the opposite of the drill down operation. The roll-up operation navigates upwards in a hierarchy decreasing the granularity of the data. The result of the roll-up operations gives a more global view of the data. In order to get these results data from a more granular level within the dimension hierarchy has to be aggregated. [JJM08; LT09]
From "For each individual store, show separately the number of product units sold for each product category during workdays and during holiday/weekend days" to "For each individual store, show separately the number of product units sold for each product category during workdays and during holiday/weekend days"

Aggregate The *aggregate* operations performs - as the name suggests - aggregation on the selected dimensions. One or more dimensions can be specified which will then also be considered as grouping attributes.[DT97; DT01] There can be made a distinction between different types of aggregate functions [Tam98; Aga+96; Gra+97]:

- Distributive (e.g. Count, Min, Max and Sum)
- Algebraic (e.g. Avergate, standard deviation, MaxN and MinN)
- Holistic (e.g. median, Mode and Rank)

Rename The *rename* operation is introduced by Li and Wang in order to promote or demote an attribute from a regular to dimensional or vice versa. Besides this, the rename operation can also prevent that later cube operation result in attributes with the same name.[DT01]

Add Dimension The *add dimension* operator adds an empty dimension to the cube. This operation is needed as preparation for a union.

Transfer The *transfer* operation moves a (dimension) attribute to another dimension.

Cartesian product The *Cartesian Product* is similar to the OLTP equivalent. It combines the input cubes. The input cubes are not required to have a dimension in common.

Join The *join* is - just as with the OLTP operator - a special form of the Cartesian product. The input cubes should have at least one dimension in common. From this common dimension(s) the mapping to the attributes should be identical.

Union The *union* operation combines two input cubes into one output cube. The dimensions, attributes and measures must be equal in both input cubes.

RC-Join The *RC-Join* operation is a join operation that makes a dimensional relation out of a (regular) relation. [LW96]

Difference The *difference* operation outputs the differences between two cubes [DT97; DT01]. In other words, the difference operation removes the part of the first input cube that exists also in the second input cube. This operation is thus equivalent to the OLTP difference operator. (figure 4.3c)

Pull / Push The *pull/push* operation transforms a measure into dimension (pull) or vice versa (push). Also known as *Extract/Force*[DT97; DT01]

Traditional Relational Operations The traditional relational operations like *equal to*, *not equal to*, *greater than*, *less than*, *greater than or equal to* and *less than or equal to* ($=$, $\neq$, $>$, $<$, $\geq$ and $\leq$) are only mentioned by Datta and Thomas, but are required for performing (almost) all other operations.

Roll Li and Wang have defined the *roll* operation, this operation has nothing to do with the roll-up/down operation, and is often used for *rolling averages*. The roll operation sorts the attribute values on a predefined order. Then a grouping is made for the required interval. Some possible roll operations are: overlapped, non-overlapped, forward cumulating, and all intervals. [LW96] This operation is a grouping operation. It does not operate on the complete cube, but creates groups on which other operations can be performed such as *aggregation*.

4.3.4 Formal definitions

During (recent) years a number of researchers presented us with a large number of models for multi-dimensional databases or cubes. Some of these models can be found in [AGS97; CT98; DT97; DT01; LT06; LT09; LW96; Man+03; Man+05; MS97; NTW00; PJ99; Tes01; Vas98]. These models are sometimes defined in order to define a model for a data cube, sometimes for defining operations and sometimes for optimization research. The (conceptual) model proposed by Datta

and Thomas in [DT97] and further refined in [DT01] is a complete and very readable model. For this reason this model will be shown and used in the following sections. In order to increase the readability of this thesis only the definition of the cube is included in the main body. For the readers that are interested in the definition of the previously mentioned operations, these definitions have been added as an appendix (A)

Cube

The cube is a well-accepted model for describing multidimensional databases. Sometimes only a simple 3-dimensional model is shown, but the cube can consist of an arbitrary number of dimensions. A cube can be described as a six-tuple $\langle C, A, f, d, O, L\rangle$ [DT01]. In this model C is the set of characteristics $\{c_1, c_2, ..., c_m\}$. A is the set of attributes $\{a_1, a_2, ..., a_t\}$ where an attribute has a name and a domain. An arbitrary order on A is assumed ($\leq_A$) and every attribute is known by the cube $C.f$ is a (one-to-one) mapping from a set of attributes to each characteristic ($f : C \rightarrow 2^A$). All attributes are mapped to one and only one characteristic ($\forall_{i,j,i\neq j} f(c_i) \cap f(c_j) = \emptyset$ and $\forall_{x,x\in A}, \exists_{c,c\in C} x \in f(c)$). For convenience we define a inverse mapping g which gives the characteristic belonging to an attribute: $g : A \rightarrow C$ ($g(a) = c$ iff $a \in f(c)$). d partitions the characteristics (C) into dimensions (D) and measures (M). d is a boolean and defined as:

$$\forall x \in C, d(x) = \begin{cases} 1 & \text{if } x \in D, \\ 0 & \text{otherwise} \end{cases}$$

O is the set of partial orders. Each $o_i \in O$ is a partial order defined on $f(c_i)$ and $|O| = |C|$. As last element of the tuple, L is the set of (cube) cells. Each cell is represented by an $\langle$ address, content $\rangle$ pair. The address is an n-tuple. $n = |A_d|$ with $A_d = \cup_{d_i\in D} d(d_i)$, in other words, for every dimensional attribute there is a value in the address. The address can be written as $\langle\alpha_1, \alpha_2, ..., \alpha_n\rangle$. The elements of the address tuple are ordered in the same way as the dimensional attributes. For easier notation the address element of a cube cell will be denoted as l_α. The content of the cube cell is defined as a k-tuple (just as the address element). $k = |A_m|$ with $A_m = \cup_{m_i\in M} f(m_i)$. The content is written as $\langle\chi_1, \chi_2, ..., \chi_n\rangle$. Therefore the content element of the cube cell will be denoted as l_χ. From the previous it is obvious that $n + k = |A|$.

Example In figure 4.9 an example cube is shown. This **activity** cube can be written as:
C = {`Time, Case, Handler, Activity`}
The Case, Time and Handler characteristics are dimensions while the Activity characteristic is a measure.
The different attributes are: A ={`day, week, month, year, case_name, case_type, task_field, handler_name, department, nr_of_activities, avg_activity_handling_time`}

The characteristics are mapped to the attributes by:

$$
\begin{aligned}
f(time) &= \{\texttt{day, week, month, year}\} \\
f(case) &= \{\texttt{case\_name, case\_type, task\_field}\} \\
f(handler) &= \{\texttt{handler\_name, department}\} \\
f(activity) &= \{\texttt{nr\_of\_activities, avg\_activity\_handling\_time}\}
\end{aligned}
$$

As mentioned before $d(time) = d(case) = d(handler) = 1$ and $d(activity) = 0$. For the partial order O holds that:

$$
\begin{aligned}
O_{time} &= \{\langle \texttt{day, week}\rangle, \langle \texttt{day, month}\rangle, \langle \texttt{month, year}\rangle\} \\
O_{case} &= \{\langle \texttt{case\_name, case\_type}\rangle, \langle \texttt{case\_type, task\_field}\rangle\} \\
O_{handler} &= \{\langle \texttt{handler\_name, department}\rangle\} \\
O_{activity} &= \{\}
\end{aligned}
$$

For L, we simplify the model. Only a few examples are given for those attributes which are visible at figure 4.9, thus `month, case_type, handler_name, nr_of_activities and avg_activity_handling_time`. When assuming that this is also the ordering, then the visual cells are:

$$
\begin{aligned}
l &= \langle\langle \text{March, Complaint, Pete}\rangle, \langle 10, 3\rangle\rangle \\
l &= \langle\langle \text{March, Complaint, Ann}\rangle, \langle 23, 3\rangle\rangle \\
l &= \langle\langle \text{March, Complaint, John}\rangle, \langle 7, 5\rangle\rangle \\
l &= \langle\langle \text{March, Order, Pete}\rangle, \langle 30, 4\rangle\rangle \\
l &= \langle\langle \text{March, Order, Ann}\rangle, \langle 4, 18\rangle\rangle \\
l &= \langle\langle \text{March, Order, John}\rangle, \langle 2, 2\rangle\rangle \\
l &= \langle\langle \text{March, Tender, Pete}\rangle, \langle 2, 1\rangle\rangle \\
l &= \langle\langle \text{March, Tender, Ann}\rangle, \langle 25, 6\rangle\rangle \\
l &= \langle\langle \text{March, Tender, John}\rangle, \langle 13, 1\rangle\rangle
\end{aligned}
$$

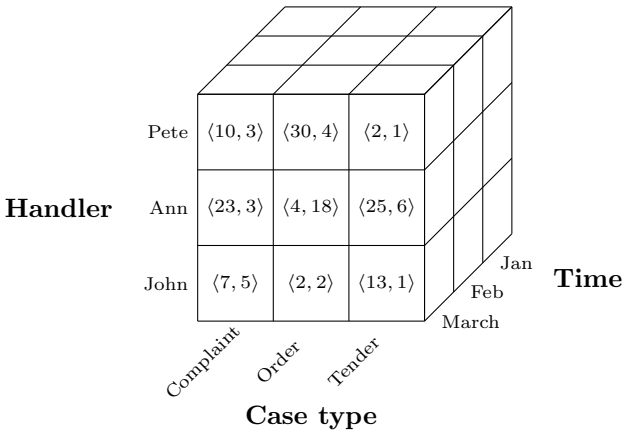


Figure 4.9: Cube example

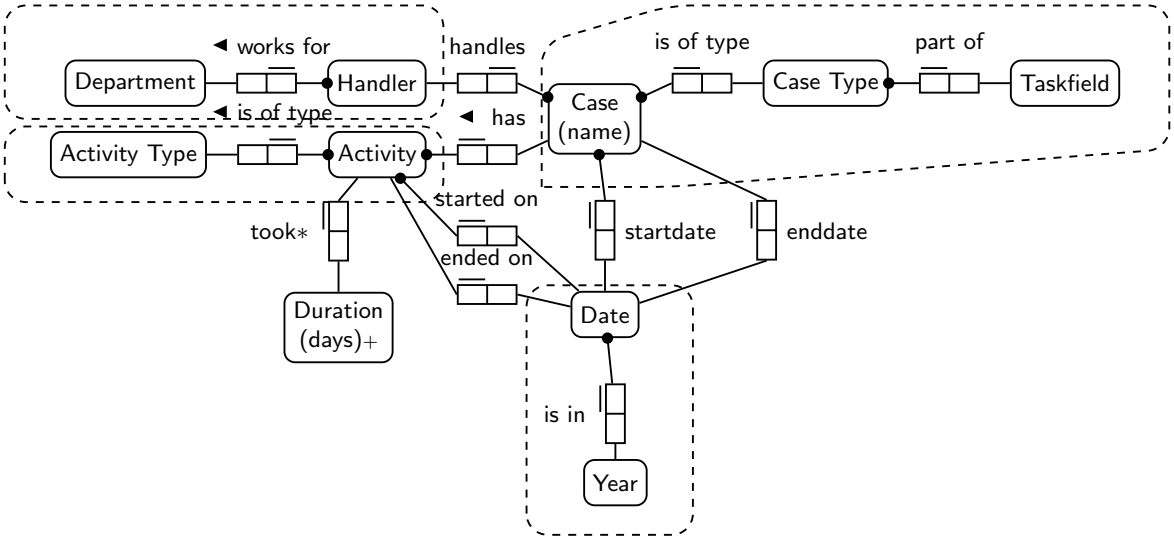
4.4 Data Transformation

We have looked so far at the conceptual models of both OLTP databases (§ 4.2) and OLAP databases (§ 4.3). The next step we have to take is to see whether it is possible to convert the databases and whether it is enough to just convert the database.

Lets begin by answering the second question. Remember the ORM-model used in one of the previous sections (figure 4.5), lets see what happens if we try to convert this model. Remember also the ingredients of an OLAP database (§ 4.3.1). An OLAP database has at least dimensions and measures, from the example multiple candidates for dimensions can easily be found (e.g. *handler, case (type), date* or *activity (type)*). A measure cannot be discovered directly, for a measure numeric values, which can be aggregated, are preferred. These values are not present in our current example. But the question that has to be asked here is *what measures do we want in the resulting cube?*. In section 5.2, this question will be discussed in more detail when we look at the information need of users of the case management tool Serenga. Suppose we would like to know how many activities are related to a case, and what the average activity duration was. The answer to both questions can be calculated from the available information. For this we have to add a number of derived facts to the model. For example `activity duration` , where `activity duration` is defined as `activity end date - activity start date` **iff** `activity` started on `activity start date` and ended on `activity end date`. This extra fact is shown in figure 4.10 Now we have an object which contains numeric information that can be aggregated.

OLTP databases can often not be converted directly because information first has to be added. For the parts of the database that can be converted the entities can be considered the attributes in the OLAP cubes. Entities that 'play' multiple roles - such as `Date` - have to be included multiple times for every role they play. For defining the characteristics, and grouping the attributes into the characteristics, domain knowledge is needed. In this example, we can user the characteristics (Handler, Case, Date and Activity) previously mentioned and one extra for the measure we just created can be used. (C = {`Handler, Case, Date, Activity and Measure`}). With domain knowledge the attributes can be manually mapped to the characteristics. In this example we have the following mapping:

$$
\begin{aligned}
f(Handler) &= \{\texttt{handler\_name, department}\} \\
f(Case) &= \{\texttt{case\_name, case\_type, task\_field}\} \\
f(Date) &= \{\texttt{date, year}\} \\
f(Activity) &= \{\texttt{activity\_name, activity\_type}\} \\
f(Measure) &= \{\texttt{activity\_duration}\}
\end{aligned}
$$



*** define** Activity took Duration(days)
as Activity started on activity_start_date **and**
Activity ended on activity_end_date **and**
Duration = activity_end_date - activity_start_date

Figure 4.10: ORM model from figure 4.5 with derived fact `Duration` and the dimensions shown by a dashed line

where $d(Handler) = d(Case) = d(Date) = D(Activity) = 1$ and $d(Measure) = 0$. The partial orders can be seen from the relations in the scheme, for the partial order O holds that:

$$
\begin{aligned}
O_{Handler} &= \{\langle \texttt{handler\_name, department} \rangle\} \\
O_{Case} &= \{\langle \texttt{case\_name, case\_type} \rangle, \langle \texttt{case\_type, task\_field} \rangle\} \\
O_{Date} &= \{\langle \texttt{date, year} \rangle\} \\
O_{Activity} &= \{\langle \texttt{activity\_name, activity\_type} \rangle\} \\
O_{Measure} &= \{\}
\end{aligned}
$$

The final steps to get the cube of (§ 4.3.4) is calculating the number of activities per case and the average time to finish an activity per case.

$$G = \{\texttt{date, year, case\_name, case\_type, task\_field, handler\_name, department}\}$$

$$\Gamma_{\text{AVG},G,\texttt{activity\_duration}}(Cube) \cup \Gamma_{\text{COUNT},G,\texttt{activity\_name}}(\Psi_{\texttt{activity\_name,Measure},\{\}}(Cube))$$

4.5 Real-Time

Naturally the data in the operational databases changes often. This is one of the reasons why OLTP databases are optimized for the insert and updating operations. New data can be inserted in an OLAP databases, especially if only new facts are being added. Although the database is not optimized for continues inserts it is possible. When data inside the OLAP databases is changed (updated information in the OLTP databases) the performance decreases in most OLAP databases. In case of small databases or a relatively low number of updates the performance decrease can be acceptable.[Lan04]

Often used alternatives to reach a near real-time database is increasing the update frequency or combining a part of the operational database with the OLAP cube. For both options a staging table (including all differences with the previous update) or a good log is essential

Chapter 5

Case Study

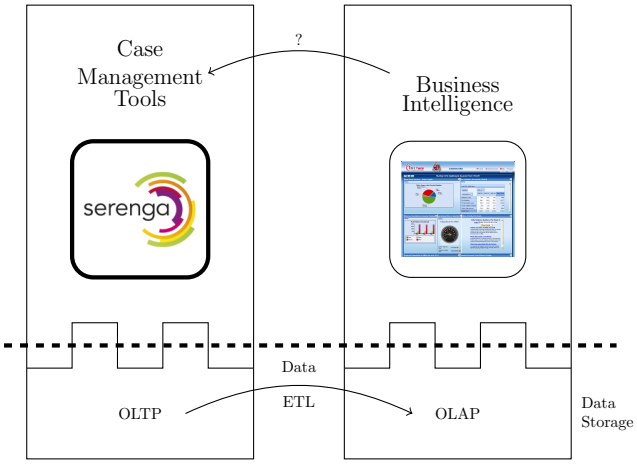


Figure 5.1: Serenga in relation with the different concepts

The case management tool used in this thesis for the case study is Serenga. The data (of the case management tool) is stored in a Microsoft SQL server and the data (for business intelligence) is stored in OLAP cubes in MS SQL Analysis Server. After an introduction to Serenga the information need and the available data will be discussed. This knowledge will be used for designing and creating an OLAP cube which will be used for business intelligence reports and self-service BI.

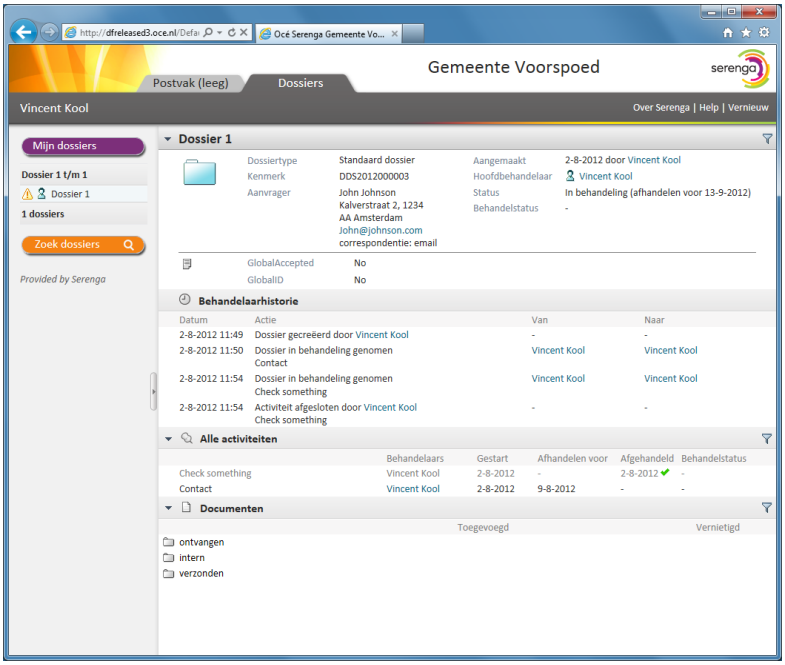


Figure 5.2: Serenga user interface

5.1 Serenga

Serenga is s case management tool from Océ-Technologies. The tool supports multiple organizations in their case management, from mail handling to supporting sales process. The tool is designed to be flexible and adaptive, this means that there is a possibility to create new activities on-the-fly (if enabled). The tool started as *DossierFlow*. This was a specific document management system centered around the concept of a 'dossier'. A typical dossier consist of multiple types of documents, and everyone who works on a dossier should access to the available information and can delegate the case or just a task.The user also is able to take over the case or task for example when a colleague becomes ill. The tasks can be pre-defined or be created ad-hoc. The tool evolved into a case management tool by supporting typical case management functions (such as activities, and states). In 2012 the product changed its name into Serenga, in figure 5.2 the user interface of Serenga is shown.

5.1.1 Processes

Case involves one or more handlers which can perform or delegate tasks (which can be parallel), in order to support the case handling documents can be added to the cases, These documents will be grouped in document types which are dependent on the type of case. After finishing a task, the task can be labeled 'accepted' or 'declined', finishing a task can lead to a 'process state'

(or *milestone*). Cases can be assigned meta-data at a global level (e.g. each case should have an 'customer ID' associated). On the level of case type, specific meta-data can be assigned (e.g. for digitalized dossiers a specific meta-data field can be the 'analog ID' or 'file cabinet'). Also the document types allowed, process states or predefined activities are assigned on the level of case type.

5.1.2 Users of Serenga

Serenga is used by multiple types of customers, from municipalities to companies to universities. The users having to work with the system all have a different background and in order to support all the users, the interface of the software is kept as easy as possible. The design goal was to have a user interface which could be used without having to consult a manual. Some of the users have to work every day with the system while others only have to use the system on rare occasions.

Besides the users that use the system to finish task (activities) or cases, there are also the managers who have to monitor the progress and optimize or adjust the processes and workloads. These managers can look at the cases individual or look at a logging report. For some customers special reports are built retrieving various information from the system in order for easier access to all the relevant management information.

5.1.3 Current BI

At the moment Serenga does not offer much business intelligence. The first steps for adding business intelligence to Serenga are taken. Welie explored the possibilities for adding business process mining to Serenga, this resulted in the possibility to view business process models such as in figure 5.3. [Wel12]

Other 'business intelligence' parts of Serenga are audit logs. These logs contain more data than information, so the term 'business data' is perhaps more suitable. Management reports, a traditional form of business intelligence are not part of a standard Serenga installation. For one of the customers an extended selection of management reports was created (in figure 5.12 one of these reports is shown with a OLAP database as source instead of an OLTP database for which the reports were created)

5.2 Information need

Some of the customers of Serenga expressed the need for business intelligence in order to monitor and optimize the case handling. The following list with questions are generalized questions

Figure 5.3: Model for the 'Investeringssubsidies' (Investment subsidy) process with dependency threshold of 0.1 [Wel12, pp. 61]

gathered from a consultant of Serenga, who knows the questions from multiple customers and another consultant of one of the customer who was assigned to optimize the process.

1. How do the users perform? (e.g. Number of finished dossiers/Number of approvals/Time to finish an activity)
2. How many cases are assigned to a user?
3. How many cases are (being) processed?
4. How many sub cases are there?
5. How many cases of type X result in case of type Y?
6. How many documents are there in a case?
7. What is the time of adding (certain) documents to a case?
8. What is the time between an activity and a milestone?
9. What is the turnaround time of activities?

10. What are problem cases?
11. How often are cases being delegated/taken over?
12. How many users are involved in a cases?
13. How many cases of type X are finished and approved per BU (Business Unit)? (And what is the percentage)
14. How many cases of type X are finished and declined per BU (Business Unit)? (And what is the percentage)
15. What is the average time from milestone X to milestone Y? (per milestone Y, per user)

Except for the question "*What are problem cases*", the questions ask for numerical information. But we have to keep in mind that these questions are formulated by those who expect to have, or at least have worked regularly with, tables. The user has to analyze the data and convert the data into information on which decision can be made. The idea behind a question like "How many cases are assigned to a user" might be the base for reducing workload or spread the workload between the different users. This indicates a spatial information need. Table 5.1 shows the identified measures. Questions 5, 13, 14 and 15 are specific for one user of Serenga (e.g. the use of BU). Using instead of Business Unit (in questions 13,14 and 15) an arbitrary meta data value, then we can generalize these questions. For question 5 we have to define how a case can result in another case. Cases have (besides possible metadata) only a notion of a parent case. There are of course possibilities to define when a case is a resulting case (e.g. if they share a parent case and the 'resulting' case is started within a specified period after the original case has finished) but defining these kind of conditions is much harder than adding a metadata field to a case which indicates whether it is a resulting case (and which case was the predecessor).

Question	Aggregate	Object	Role
1	Count	Activity	
1, 9	-	Activity	Handling time
8	-	Activity	Time to Milestone
1, 2, 3, 4, 13, 14	Count	Case	
11	Count	Case	Delegation
12	Count	Case	Take Over
6	Count	Document	
7	-	Document	Time to add
8	-	Milestone	Time to activity
15	-	Milestone	Time to milestone

Table 5.1: Measures wanted

The most difficult question is probably question 10 "*What are problem cases ?*" This question cannot be answered directly, because there is no attribute *problem* attached to a case, marking it as a problem case. In order to answer this question, first a definition has to be made of which cases qualify for being a *problem case*. A problem with what makes a case a problem case is that

it can depend on for example the case type. E.g. a case about permits can become a problem case when deadlines are not met (or the chance is high that the deadline will not be met), while a case of the type 'order' can for example be qualified as problem case when signatures are not on the required documents (thus the activity check signatures is closed with status 'closed, declined'). Because of this ambiguous definition of a problem case, it is better that the user defines what qualifies as a problem case and finds the desired results himself (self-service BI).

In the list with the questions, a number of attributes is mentioned for which grouping should be possible. In table 5.2 these are listed.

Object	Attribute
Activity	-
Activity	status
Case	-
Case	BU (metadata)
Case	parentCase
Case	status
Case	type
Milestone	-
Users	-

Table 5.2: Grouping Attributes

5.3 Available data

After looking at the information need, we will now look at what data is available in order to see whether we can convert the data directly to an OLAP cube or that we have to calculate some of the measures or that we have to include data from other sources. A complete and clear data model of Serenga was not available. The design of the database changed during the years of development of Dossierflow and later Serenga, this makes it hard to get all the required information (some of the information became only available in a later version). This model is far from complete, for instance the restriction are not included. Also the part of the database were the logging information is stored and the information about the applicant are not included in the model. This model gives a better understanding in the relations between the different objects, the model can be compared with the relation between the facts and the dimensions in the cube later in this chapter. (figure 5.6)

5.3.1 Missing information

First we look at the measures defined in the previous sections. The measures `count activity`, `count case` and `count document` can be calculated in the OLAP cube with the aggregate oper-

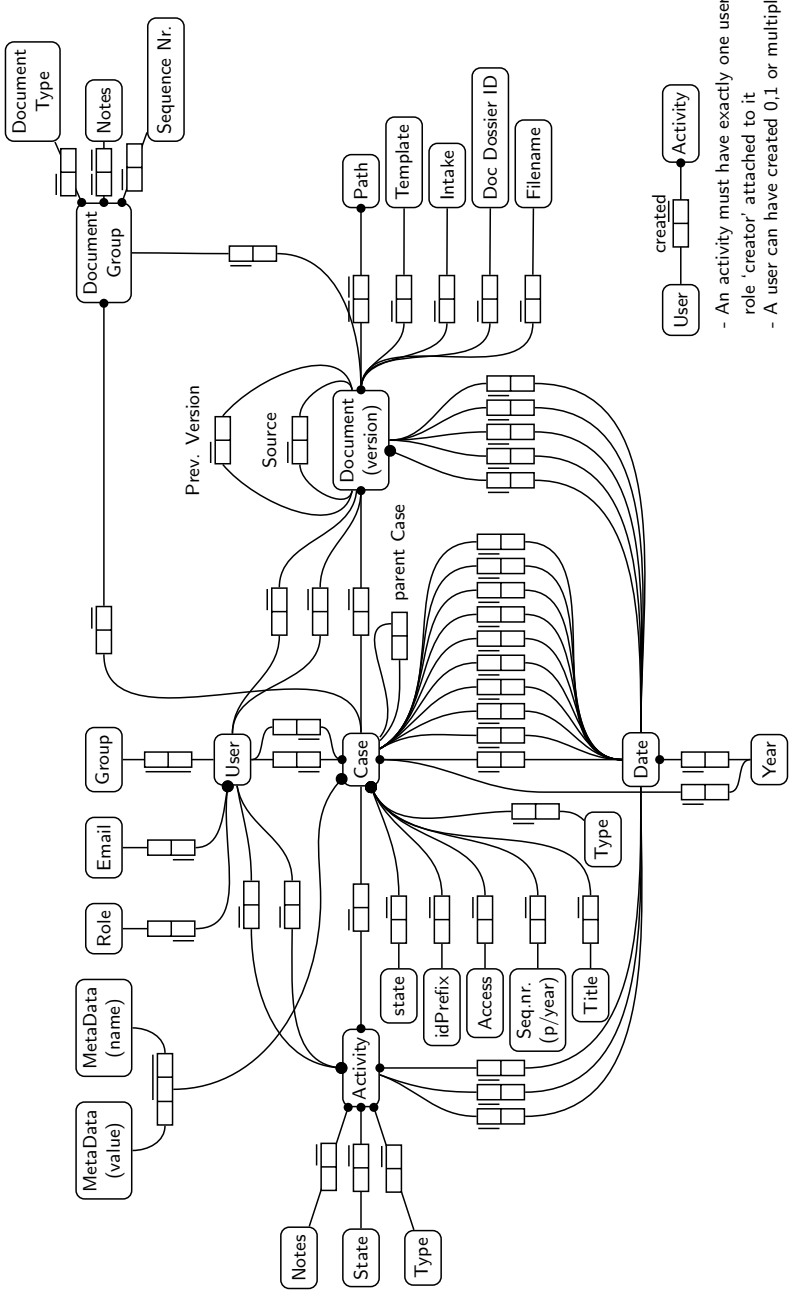


Figure 5.4: Simplified model of Serenga

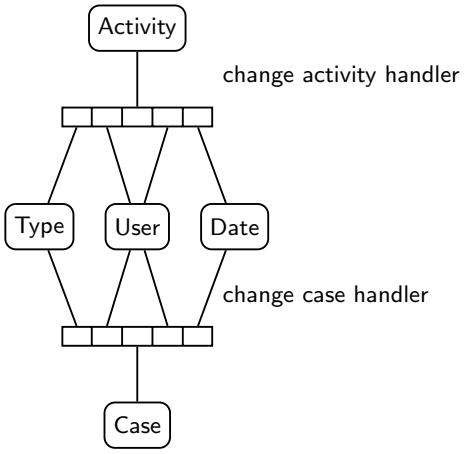


Figure 5.5: Relation between the objects when changing the handler

ation if we make sure the data has sufficient details (on document, activity and case level).

Time

The measure `activity handling time` and `document time to add` have to be calculated before loading the data in the cube. This can be done in the same way as we did in § 4.4.

define *Activity* took *Handling time*(days)
as *Activity* started on activity_start_date **and**
Activity ended on activity_end_date **and**
Handling time = activity_end_date - activity_start_date

and

define *Document* was added after *Time to add*(days)
as *Document.Case* started on case_start_date **and**
Document was created on document_creation_date **and**
Time to add = document_creation_date - case_start_date

Delegation

Case delegations and case take overs are when a case gets another handler. The difference is that by a delegation the user which initiates the delegation is different from the user who becomes the new handler. By a take over the user initiating the action becomes the new handler. Both are not stored directly in the database, but has to be extracted from the logging table (in which all actions are stored). The relation between the objects in the changing of a handler is shown in figure 5.5

Milestones

Unfortunately the version of Serenga on which this case study is performed does not include the previously mentioned process states (milestones). Because there is a need for including this information (it defines clear points in a case) we use the end of activities aas milestones when answering the questions form §5.2 in §5.6.1 and appendix C.

5.3.2 Naming of activities

Another problem was that the activities were not carefully defined, table 5.3 shows that most activities are of type 'handle'. This is a general type of activity which can describe a large number of actual activities this has as a result that no conclusions can be drawn from aggregations of these activities. This was not directly a deficient of the system. The problem was that the users used non-descriptive activity types for the (delegation of) activities. Notes were added to the activity in order to supply the delegate with information.

Occurrence		Activity	
7114	82, 10%	Handle	(behandelen)
853	9, 84%	for (your) information	(ter kennisname)
141	1, 63%	for (your) information	(ter informatie)
40	0, 46%	Co-handle	(Medebehandelen)
27	0, 31%	Check Letter	(controleren brief)

Table 5.3: Top 5 of activities. Total activities is 8665, total distinct activities is 152 (sample database 1)

Another database had the following top-5 of activities. (see table 5.4)

Occurrence		Activity	
8117	18, 28%	Order can be entered	Order kan ingevoerd worden
7274	16, 38%	Order is checked, archive case	order is gecuact, dossier archiveren
4986	11, 23%	Handle	Behandelen
3121	7, 03%	Access to the case is granted (do not close the case)	U heeft toegang tot het dossier (niet afhandelen)
2855	6, 43%	Order can be entered	Order kan worden ingevoerd

Table 5.4: Top 5 of (distinct) activities. Total activities is 44398, total distinct activities is 243 (sample database 2)

5.3.3 Date of Documents

Documents can have multiple versions and every version has an attribution attached to it with the date of creation. For looking at the time to add the document (from the start of a case)

and the time to the end of a case can be looked at the time of the first version (which makes sense for looking at the time to adding the document) or the last date of addition (which makes more sense when looking at the time to finish a case). A solution can be to split the field *date of addition* to *date of first addition* and *date of last edit*. Better grouping of documents in the dimension can solve this problem. The date of addition and last edit then become calculated members, while the information about the authors and the addition of the separate versions is preserved.

5.3.4 Removed Users

One remarkable observation was that users are not stored persistently in the system. When the access rights of a user are revoked (the user can not access the system anymore) the user is deleted from the database. Cases, activities and document created or delegated to such a user still have the reference to the user. In order to deal with this problem the missing users are added to the system from the logging table (for all the user references without an user, the last known name is retrieved). The case management tool could change the behavior when removing a user (for example by creating an attribute "access right" and setting it to *false* at deletion).

5.4 Cube model

In the previous sections we defined what information is needed and what data is available. In this section we will continue by finding a cube model that can be created with the available data and satisfies the information need. We start with the facts, because they will provide the information, then we will continue with the dimensions. The dimensions are related to the facts and allow navigation through the data.

5.4.1 Facts

Because the data in the operational database changes at several events (of which the most important are listed) using accumulated snapshots is the most common way to store the facts.[GMR98; KR02]

Case	
–	handling time
⊛	count activities
⊛	count documents
⊛	count handler changes

Activity	
⊖	handling time
–	time from start case → start activity
∗	time from start case → end activity
–	time from start activity → end case
∗	time from end activity → end case
∗	count handler changes

Document	
⊖	time from start case → add document
–	time from add document → end case
–	count versions

∗	Can be aggregated
⊛	Can be aggregated and was specified
–	Must be calculated
⊖	Must be calculated and was specified

Table 5.5: Facts

Moments the data in the database changes are:

- Start Case
- Close Case
- Start Activity
- Close Activity
- Reach Milestone
- Add Document
- Change Main handler
- Change Activity handler

For analyzing the facts it is useful to look at those activities or cases that are closed. From this data averages can be calculated and used as prediction for future activities (or cases). Of course this only holds for those cases / activities that have a resemblance. Using (almost) only one type of activity (e.g. 'Handle') does not give useful results.

In table 5.5 the facts with measures are shown. In table 5.6 the facts without measures are shown but these facts are only counted.

Activity delegation
– count handler changes

Case delegation
– count handler changes

Document checkout
– count document checkouts

Table 5.6: Facts with only a count of itself

Calculated Measures

Some of the measures can be calculated, (see also table 5.5) this happens after loading the data. Calculating averages is one of the most common calculations in the created cube. For almost all measures regarding time the aggregate function is averaging. Because this is not supported by the tool (Microsoft SQL Server Analysis Services), the averaging can be done as follows:

The average time for handling an activity per user can be calculated with the following formula in MDX:

```
CREATE MEMBER CURRENTCUBE.[Measures].[Avg Time Handling]
 AS [Measures].[Sum Time Handling]/[Measures].[Finished Activity],
FORMAT_STRING = "#,##0.00;-#,##0.00",
NON_EMPTY_BEHAVIOR = { [Sum Time Handling] },
VISIBLE = 1 ,  DISPLAY_FOLDER = 'Averages' ,  ASSOCIATED_MEASURE_GROUP = 'Fact Activity';
```

This measure is necessary for fulfilling the need to compare users in time for handling activities. The standard aggregations like `sum` or `count` are not useful. Note that comparing the handing time of an activity called 'handling' still does not produce information.

5.4.2 Dimensions

There are a number of dimensions in the data. `Case, Activity` and `User` are already mentioned in table 5.2. In figure 5.4 we can identify these dimension and it becomes clear what the attributes are (essentially all connected objects). However not all of these objects are suitable for grouping or navigating through the data. For example notes attached to an activity give information about the activity, but cannot be used for grouping or rolling up to. For example the attributes that are useful for `Activity` are `activity type, state, start date, due date, finish date, case, current user` and `creator`. This last attribute has to be retrieved from the logging because this is not stored with the activity.

For the `User` dimension, the attributes are `user name, role, email` and `group`.

The `Document` dimension has the following attributes `case, template, intake, document dossier id, creation date, deletion date, checked out date, retention date` and `destroy date`.

The `Case` dimension has `creator, handler, title, DDSNR`[1], `seq.nr., access, id prefix, state, type, start date, due date, finish date, deletion date, archive date, max due date, fixed due date, export date, destroy date` and `retention date` as attributes.

In figure 5.6 the relation between the facts and the dimensions is shown. The 'blue tables' store the information about the dimensions and the 'yellow tables' store information about the facts.

5.5 Loading of data

The data from the operational database has to be loaded into the OLAP cube. For referential integrity it is recommend that there is some order in adding the activities to the database. This will be (in principle) a chronological order. First the start of a case. Then either the closure of a case, delegating (starting an activity) or adding a document. Changing the handler of an activity can only take place after creating a case and before closing it. After closing a case, activities can't be closed any more.

Figure 5.7 shows the loading process (of a new cube, so no incremental updates).

5.5.1 Real-Time

In the case of Serenga where all the information can change, both the dimension data as the fact data can change, real-time updating decreases performance of the OLAP database and probably also the performance of the operational database. There are possibilities to decrease the time between updates, for example reduce the time between the data loading processes. One of the key issues with decreasing the time between updates is that the changed data since the last update should be easy accessible. E.g. a logging, temporary or staging table should make finding the changed data quicker and therefore reducing the workload of the operational database.

[1] A case identifier that is human readable.

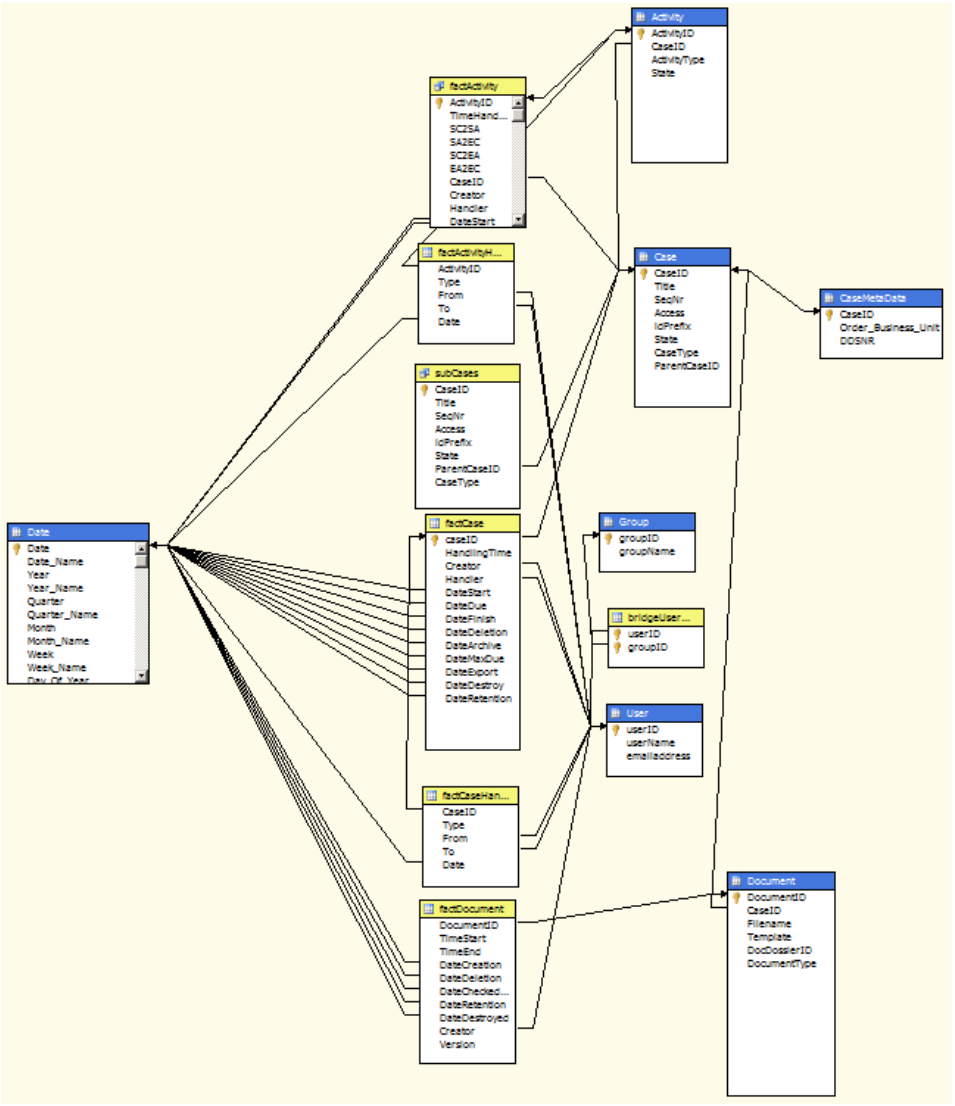


Figure 5.6: Relation between the dimensions and the facts

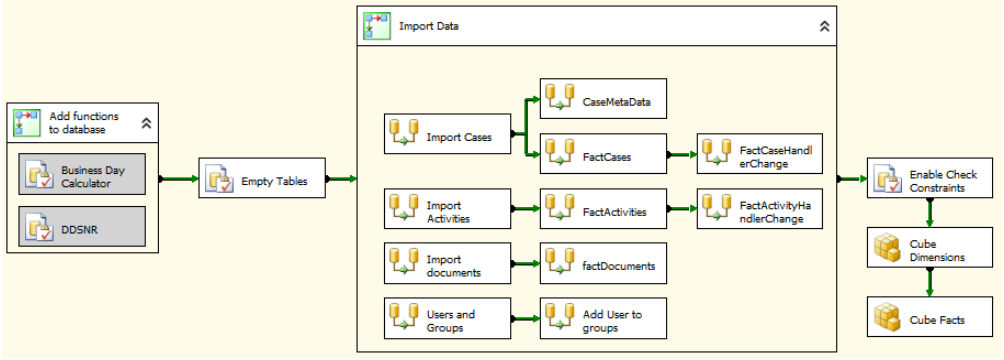


Figure 5.7: Loading of the data

5.6 Self-Service BI

One of the goals of the case study was looking at the possibilities for using the information from the OLAP cubes in Microsoft Excel. This can be used as tool for self-service BI, the user is free to do with the data as he/she pleases (e.g. creating tables, graphs). When a cube, created in §5.4 is deployed on the server, then tools sucha as Microsoft Excel can connect to this cube. With the (free) PowerPivot[2] add-in pivot tables can be created.

5.6.1 User Performance

The first question from §5.2 is "How do users perform?" This is a fairly general question and can be answered in several ways. With this question using a graph is more appropriate because then it is easier to compare different users. In every comparison it is necessary to keep in mind what values are displayed, are the values really comparable? (e.g. the activity 'handle' is not good for comparison). Three possible graphs will be shown that partially answer the question "How do users perform?". The queries, the extra calculations (when necessary) and the graphs will be shown. hat the users wants to see can be one of the graphs or the combination of graphs. The tables, graphs and queries for the other questions are included in appendix C

Number of finished cases In the first two graphs we compare the users by looking at the number of cases they have created and the percentage of those cases which are finished. For these graphs we need from the cube the case creator, the case count and the number of finished cases. In Excel we can calculate the percentage of finished cases (number of finished cases / case count) and the number of unfinished cases (case count - number of finished cases). The query is created with the help of the graphical "Table Import Wizard" (figure 5.8), the resulting query is:

[2] www.powerpivot.com

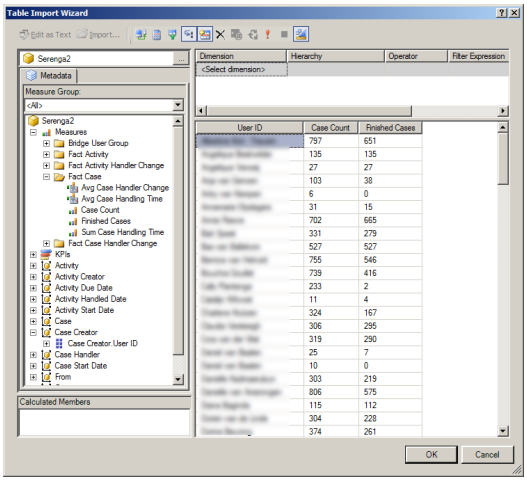


Figure 5.8: The 'Table Import Wizard' from the PowerPivot add-in can be used to create mdx-queries by drag-and-drop

```
SELECT
  NON EMPTY { [Measures].[Finished Cases],
              [Measures].[Case Count] } ON COLUMNS,
  NON EMPTY { ([Case Creator].[User ID].[User ID].ALLMEMBERS ) }
            DIMENSION PROPERTIES MEMBER_CAPTION,
                                 MEMBER_UNIQUE_NAME ON ROWS
FROM [Serenga2]
```

After creating the calculated members, two graphs are created. The first graph (figure 5.9-top) shows the number of finished and unfinished cases, the second graph (figure 5.9-bottom) shows the number of finished cases in percentages.

Activity handling time In the next two graphs we look at the time it takes to finish an activity grouped by the current activity handler. These graphs require the activity handlers, the average handling time and the case type (we filter on case type in the second graph). The query is:

```
SELECT
  NON EMPTY { [Measures].[Avg Time Handling] } ON COLUMNS,
  NON EMPTY { ( [Activity Handler].[User ID].[User ID].ALLMEMBERS *
                [Case].[CaseType].[CaseType].ALLMEMBERS ) }
DIMENSION PROPERTIES MEMBER_CAPTION,
                     MEMBER_UNIQUE_NAME ON ROWS
FROM [Serenga2]
```

The first graph we create (figure 5.10-top) shows the average handling time for an activity per activity handler (the last known handler of the activity). The second graph (figure 5.10-bottom) shows also the average handling time for an activity per activity handler but only for activities that belonged to a case of type 'order'. For both graphs a logarithmic scale was chosen due to the large differences. Another option is for example to filter the number of persons (e.g. by group) or on activity type.

Specific activity Instead of comparing averages of different activity types it often make more sense to compare the performance of users based on the handling time of activities of the same type of cases with the same type. In order to give some extra context to the resulting graph, the number of activities is included. This indicates whether the average is based on a small or on a large number of activities. From the cube we select the average handling time, the activity count, the activity handler and the activity type. The query is then:

```
SELECT
  NON EMPTY { [Measures].[Avg Time Handling],
              [Measures].[Fact Activity Count] } ON COLUMNS,
  NON EMPTY { ([Activity Handler].[User ID].[User ID].ALLMEMBERS *
               [Activity].[CaseType-ActivityType].[Activity Type].ALLMEMBERS ) }
            DIMENSION PROPERTIES MEMBER_CAPTION,
                                 MEMBER_UNIQUE_NAME ON ROWS
FROM [Serenga2]
```

If we make a graph for the activity 'complete order' as shown in figure 5.11 we can see the number of activities (the surface graph on the background) and the average time for the activity (the bars on the foreground).

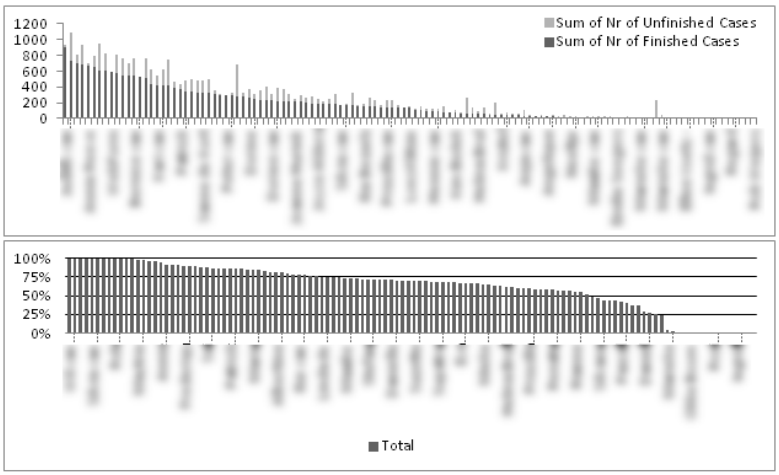


Figure 5.9: The number (percentage) of cases (both open and closed) per case creator

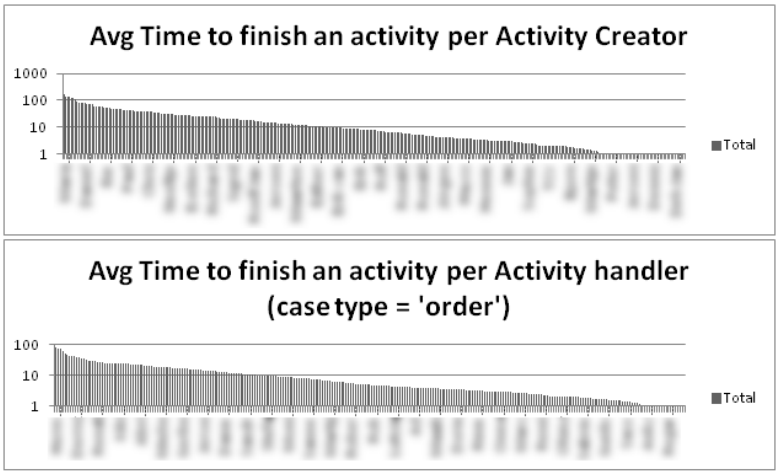


Figure 5.10: The average time to finish an activity per (last known) handler, because the differences are large a **logarithmic** scale was chosen. The upper graph shows activities of every case type, the bottom graph only activities belonging to the case-type 'order'

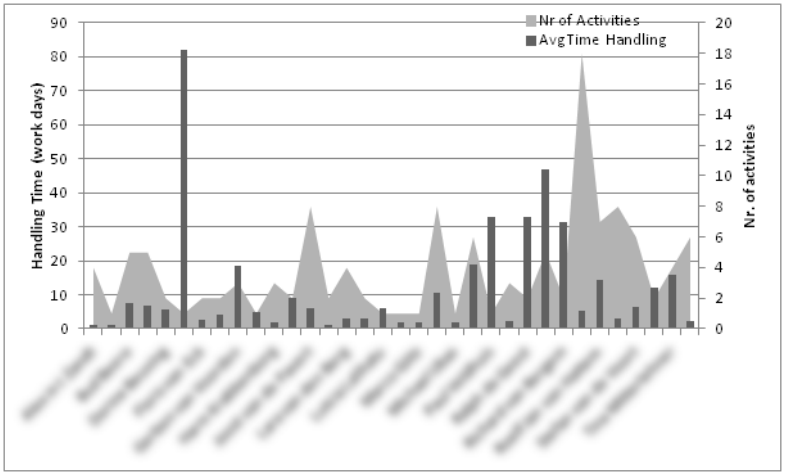


Figure 5.11: The average time to finish an activity per user (bars). Also shown is the number of activities the average is based on (background surface)

5.7 'Traditional' BI

Besides the self-service possibilities of the OLAP cube, we also looked at the advantages and disadvantages of having the OLAP cube as data source for traditional business intelligence. At the time of writing this thesis (BI) reports for Serenga are made on a consultancy base. If a customer wants these reports, these reports will be custom made, besides an audit log there are no standard reports.

One of the customers with custom reports is Océ itself. The reports included a KPI overview, specialized overviews and detail reports. These reports are based on the operational OLTP database. Generating the reports directly from the operational database has some advantages, the data shown is in real-time, the data does not need to be stored in an OLAP database and the queries can be written in (standard) SQL.

The disadvantage of having the reports generated from the OLTP database is that the data needed is stored in multiple tables, such that multiple joins are necessary. Some of the data is stored in xml in the tables, retrieving this data is costly. The heavy queries decreases the performance of Serenga in general, sometimes this causes a sluggish feel.

After creating the OLAP cube, two of the reports (the custom reports from Océ) were adjusted, so that the information was retrieved from the OLAP cube. The first observations showed a (great) performance increase in generating the reports ($2 \sim 50$ times faster). Part of the performance gain was because data that was stored in xml fields was extracted during the loading of the data in the OLAP cube.

One of the reports can be seen in figure 5.12. In this report the performance of users (account managers) is compared with other users. The number of order cases for which the order is approved in one time is compared with the total number of order cases. The results can be filtered by business unit.

A problem occurred when adding the filter dates, including the possibility to have a random start and end date decreased the performance drastically. The reason for this decrease is that the cube has to aggregate the specified dates instead of using pre-aggregated data for the whole cube. Changing the report so that instead of having a random start and end date the user has the ability to choose arbitrary months for the report results in a better performance while having some of the freedom.

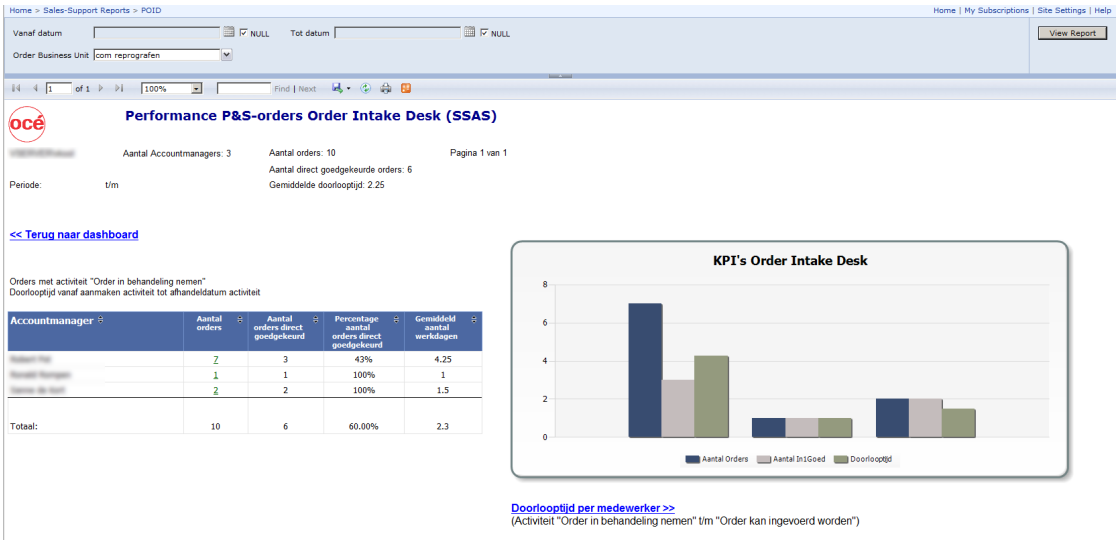


Figure 5.12: Performance P&S-order Order Intake Desk report based on the OLAP cube

Chapter 6

Conclusion

This thesis showed what case management tools are and what how the processes these tools support generally look like. The processes are flexible in contrast to those supported by workflow management systems. But the tool offers more support than document management systems. In order to monitor and improve the processes case management tools can be enhanced with business intelligence capabilities. Information from this business intelligence part can be presented to the users as graphs or tables dependent on the information need. Depending on the users using the system self-service can be the better choice if the user is familiar with some of these tools and knows what the data means. Standard predefined reports are more appropriate for users wanting only to consume data (because of time constrains, lack of experience with the tools or little knowledge about the meaning of the data).

OLAP databases or cubes are a type of databases that are optimized for analytical processes. Retrieving data, navigating through the data and calculating aggregations is faster on OLAP databases. The disadvantage is that such databases have to be designed specifically to match both the information need and the availability of the data.

The information need can be converted to an OLAP cube design by dividing the information between the measures (such as time of handling or number of cases) and the objects used for grouping or querying the database, the dimensions.

The original research goal was to find the best design for a business intelligence module for case management systems with the help of OLAP cubes. This goal was too broad, the best design for a module for all case management systems is not made. There are elements that are the same for case management tools in general, but a lot of data will differ but the best design will have a connection with the specific tool. The best BI module should be custom made for the case management tool and the data (e.g. adding information about the business unit in the case study). Therefore design steps were defined for creating a business intelligence module in case

management tools. With the steps made in this thesis, gathering the information need followed by looking at the available information the OLAP cube(s) can be created. With the OLAP cube the BI tool can be supported. But choosing the best BI tool (self service or 'traditional reports') is dependent on the users.

For Serenga a cube was developed that matches the information need and the available data in the example databases. The questions gathered, except for the vague question 'what are problem cases', could be answered with the data available in the OLAP cubes. That the answers could be found was demonstrated by using the cube as data source for a self service business intelligence tool that most of the expected users have experience with. The OLAP cube was also used as basis for standard reports which had as benefits a shorter query and a higher performance.

Generalizing

Although the OLAP cube made in the case study was specifically for Serenga, the experiences gained while making this cube can be extended to case management tools in general. Because case management tools are similar on the main points and users having the same (sort of) questions, the cube model can be very similar to the one described in this paper. The goal for Business Intelligence is improving the business. This is done by analyzing and monitoring the business to identify the current state of the business, prediction in order to spot possible problems and making optimizations.

For monitoring, making predictions and optimizing the business having performance information (such as throughput time, number of cases/activities made/handled or percentage of successful cases) is necessary in order to make decisions based on facts. Depending on the users Self-service BI or normal BI reports are appropriate. Giving users access to self-service BI, it benefits the organization if these self-made reports can be shared with coworkers (if necessary though a professional/manager who approves them).

During the case study we encountered some problems. Some of the information we could have used was not available or it was hard to find. This is a problem that is likely to be found in other case management tools. Most systems grow during the lifespan and new features and/or customer request are added to the product without the thought that storing some extra information might come in handy in a later when business intelligence is added to the tool.

Real-time information can stay hard to accomplish. But if the business value of having real-time information is high enough information can be added real time by combining information from the OLAP cube with the OLTP database, or adding the information directly into the OLAP cube (which is rather difficult).

Having quantitative reports to compare performance or spot problems, it is important to realize

that you cannot compare these reports with for example manufacturing reports where the process is standardized or robotized. Cases can, although the goal and the steps can be the same, still be very different. Finding abnormalities, for example that one users takes significantly more time to perform a task, does not necessarily mean that he/she is more slow or does not work fast enough; it could for example mean that he/she is given the hardest tasks.

Recommendations for Serenga

- Serenga could use a good model or definition of the complete case management system. Having a good model enables the creation of a good OLAP cube with correct dimensions. In figure 5.4 a simple model was made, but this is not a complete formal model.
- For adding business intelligence to Serenga OLAP cubes are useful and performance increasing, especially in larger databases (larger customers). For successful implementation of the OLAP cubes it is necessary to change the logging table or creating a new 'last changes' table with the information that has to be changed or added to the database. This allows for a higher update frequency with caller loading time, making the information more up-to-date (removing one of the downsides).
- A solution to increase performance in reports without using OLAP cubes is aggregating the information for historical data (e.g. once every night, week or month). In this case the changed data should also be made better accessible so that data can be combined (the aggregated data with the recent changes). This option is not a real recommendation but it gives the opportunity to increase the performance of the current management reports.
- The data given in the current reports might not give the information desired. For determining who the account manager is, the last handler for the activity 'Process order' is used. The user who made the activity is more likely to be the account manager. This confusion shows that Serenga was not designed to give information for Business Intelligence. Including good Business Intelligence requires clear defined requirements, and to include these requirements in the software.
- As mentioned earlier naming of the activities makes it difficult to make statements about the activity. The activity 'handle' for example is not useful for use in business intelligence. A similar problem occurs with spelling of the names of the activities. The activities 'Order in behandeling nemen', 'Orde in behandling nemen", 'Order in behandeling nemen.' and 'Order in behandeling nemen?' probably are the same activity 'Start processing order'.
- Store user data persistent. Not removing the user from the system makes analysis better. Group information can be retained also for including objects that belong to a group, but for which the relation is removed because of the removal of the user. Serenga should change

the behavior when removing a user (for example by creating an attribute "access right" and setting it to *false* when revoking access to the system).

- For business intelligence it is recommended that, when possible, for documents the relation to the activity is stored. For example when adding the document a choice can be made form a list of current assigned tasks. This enables better prediction of for example the time between adding a document and finishing an activity (and the prediction whether a deadline will be met).

Recommendations for further research

- There has been research on generating OLAP cubes from conceptual models (e.g. [HSB00]) and research regarding generating conceptual models from real databases (e.g. [CBS94; Joh94]) but no research in how OLAP cubes can be generated from real databases. Research into automatic generation of OLAP cubes from existing database or into systems supporting the process can be done in the future
- Most research on OLAP cubes is focused on sales databases or other sales related (e.g. tracking customers). Future research can be done into OLAP cubes for other purposes.

Index

Bibliography

Books

[DN95] J.P. DesChamps and P.R. Nayak. *Product Juggernauts: How Companies Mobilize to Generate a Stream of Market Winners*. Harvard Business School Press, 1995. ISBN: 9780875843414.

[Dru03] P.F. Drucker. *Peter Drucker On The Profession Of Management*. Harvard business review book series. Harvard Business School Press, 2003. ISBN: 9781591393221.

[GR93] Jim Gray and Andreas Reuter. *Transaction Processing: Concepts and Techniques*. Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann Publishers, 1993. ISBN: 9781558601901.

[Inm05] William H. Inmon. *Building the data warehouse*. Wiley, 2005. ISBN: 9780764599446.

[KR02] Ralph Kimball and Margy Ross. *The data warehouse toolkit: the complete guide to dimensional modeling*. Safari Books Online. Wiley, 2002. ISBN: 9780471200246.

(Peer Reviewed) articles

[Aal+12] Wil van der Aalst et al. "Process Mining Manifesto". In: *Lecture Notes in Business Information Processing*. Vol. 99.2. Springer, 2012, pp. 169–194. DOI: 10.1007/978-3-642-28108-2_19.

[AB01] W. M. P. van der Aalst and Paul J. S. Berens. "Beyond workflow management". In: *Proceedings of the 2001 International ACM SIGGROUP Conference on Supporting Group Work*. GROUP '01. ACM Press, 2001, p. 42. ISBN: 1581132948. DOI: 10.1145/500286.500296.

[ADM00] Alessandra Agostini and Giorgio De Michelis. "Improving Flexibility of workflow Management Systems". In: *Business Process Management*. Ed. by Wil van der Aalst, Jörg Desel, and Andreas Oberweis. Vol. 1806. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2000, pp. 289–342. ISBN: 978-3-540-67454-2. DOI: 10.1007/3-540-45594-9_14.

[Aga+96] Sameet Agarwal et al. “On the Computation of Multidimensional Aggregates”. In: *VLDB ’96 Proceedings of the 22th International Conference on Very Large Data Bases*. Sept. 1996, pp. 506–521. ISBN: 1-55860-382-4.

[AGS97] Rakesh Agrawal, Ashish Gupta, and Sunita Sarawagi. “Modeling multidimensional databases”. In: *Data Engineering, 1997. Proceedings. 13th International Conference on*. IEEE. 1997, pp. 232–243.

[BR99] Kevin Beyer and Raghu Ramakrishnan. “Bottom-up computation of sparse and Iceberg CUBE”. In: *Proceedings of the 1999 ACM SIGMOD international conference on Management of data*. SIGMOD ’99. Philadelphia, Pennsylvania, United States: ACM, 1999, pp. 359–370. ISBN: 1-58113-084-8. DOI: 10.1145/304182.304214.

[CBS94] Roger H.L. Chiang, Terence M. Barron, and Veda C. Storey. “Reverse engineering of relational databases: Extraction of an EER model from a relational database”. In: *Data & Knowledge Engineering* 12.2 (1994), pp. 107 –142. ISSN: 0169-023X. DOI: 10.1016/0169-023X(94)90011-6.

[CT98] L. Cabibbo and R. Torlone. “A logical approach to multidimensional databases”. In: *Advances in Database TechnologyEDBT’98* (1998), pp. 183–197.

[DP00] Thomas H. Davenport and Lawrence Prusak. “Working knowledge: how organizations manage what they know”. In: *Ubiquity* 2000.August (Aug. 2000). ISSN: 1530-2180. DOI: 10.1145/348772.348775.

[DT01] Anindya Datta and Helen Thomas. “A Conceptual Model and Algebra for On-Line Analytical Processing in Decision Support Databases”. In: *Info. Sys. Research* 12.1 (2001), pp. 83–102. ISSN: 1526-5536. DOI: 10.1287/isre.12.1.83.9715.

[DT97] A. Datta and H. Thomas. “A Conceptual Model and an algebra for On-Line Analytical Processing in Data Warehouses”. In: *Proc. of the WITS*. Citeseer. 1997.

[Ett95] Barbara Ettorre. “Managing competitive intelligence”. In: *Management Review* 84.10 (1995), pp. 15–20.

[GG85] Benny Gilad and Tamar Gilad. “A systems approach to business intelligence”. In: *Business Horizons* 28.5 (Sept. 1985), pp. 65–70. ISSN: 00076813. DOI: 10.1016/0007-6813(85)90070-9.

[GK86] Sumantra Ghoshal and Seok Ki Kim. “Building Effective Intelligence Systems for Competitive Advantage”. In: *Sloan Management Review* 28.1 (1986), pp. 49–58. ISSN: 0019848X.

[GM71] G.A. Gorry and M.S.S. Morton. “A framework for management information systems”. In: *Sloan management review* 13.1 (1971), pp. 55–70.

[GMR98] Matteo Golfarelli, Dario Maio, and Stefano Rizzi. “Conceptual design of data warehouses from E/R schemes”. In: *System Sciences International Conference*. Thirty-First Hawaii International Conference on System Sciences. Vol. 7. c. IEEE Comput. Soc, 1998, pp. 334–343. ISBN: 0-8186-8255-8. DOI: 10.1109/HICSS.1998.649228.

[Gra+97] Jim Gray et al. "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". In: *Data Mining and Knowledge Discovery* 1 (1 Mar. 1997), pp. 29–53. ISSN: 1384-5810. DOI: 10.1023/A:1009726021843.

[Gri+04] Daniela Grigori et al. "Business Process Intelligence". In: *Computers in Industry* 53.3 (Apr. 2004), pp. 321–343. ISSN: 01663615. DOI: 10.1016/j.compind.2003.10.007.

[GW02] Georges G. Grinstein and Matthew O. Ward. "Introduction to Data Visualization". In: *Information Visualization in Data Mining and Knowledge Discovery*. Ed. by Usama Fayyad, Andreas Wierse, and Georges G. Grinstein. The Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann, 2002, pp. 21–45. ISBN: 9781558606890.

[HSB00] Karl Hahn, Carsten Sapia, and Markus Blaschka. "Automatically generating OLAP schemata from conceptual graphical models". In: *Proceedings of the 3rd ACM international workshop on Data warehousing and OLAP*. DOLAP '00. McLean, Virginia, United States: ACM, 2000, pp. 9–16. ISBN: 1-58113-323-5. DOI: 10.1145/355068.355310.

[Jan+10] Jan Martin Jansen et al. "Embedding a web-based workflow management system in a functional language". In: *Proceedings of the Tenth Workshop on Language Descriptions, Tools and Applications*. LDTA '10. Paphos, Cyprus: ACM, 2010, 7:1–7:8. ISBN: 978-1-4503-0063-6. DOI: 10.1145/1868281.1868288.

[JJM08] N. Jukic, B. Jukic, and M. Malliaris. "Online Analytical Processing (OLAP) for Decision Support". In: *Handbook on Decision Support Systems 1* (2008), pp. 259–276.

[Joh94] P. Johannesson. "A method for transforming relational schemas into conceptual schemas". In: *Data Engineering, 1994. Proceedings.10th International Conference*. Feb. 1994, pp. 190 –201. DOI: 10.1109/ICDE.1994.283030.

[Kam+] Peter J Kammer et al. "Techniques for Supporting Dynamic and Adaptive Workflow". In: *Computer Supported Cooperative Work (CSCW)* 9.3 (), pp. 269–292. ISSN: 0925-9724. DOI: 10.1023/A:1008747109146.

[KDB00] Mark Klein, Chrysanthos Dellarocas, and Abraham Bernstein. "Introduction to the Special Issue on Adaptive Workflow Systems". In: *Computer Supported Cooperative Work (CSCW)* 9.3 (2000), pp. 265–267. ISSN: 0925-9724. DOI: 10.1023/A:1008764508237.

[KS95] Ralph Kimball and Kevin Strehlo. "Why Decision Support Fails and How To Fix It." In: *SIGMOD Record* 24.3 (Sept. 1995), pp. 92–97. ISSN: 0163-5808. DOI: 10.1145/211990.212023.

[LT06] Hans-Joachim Lenz and Bernhard Thalheim. "OLAP Schemata for Correct Applications". In: *Trends in Enterprise Application Architecture*. Ed. by Dirk Draheim and Gerald Weber. Vol. 3888. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2006, pp. 99–113. DOI: 10.1007/11681885_9.

[LT09] Hans-Joachim Lenz and Bernhard Thalheim. "A Formal Framework of Aggregation for the OLAP-OLTP Model". In: *Journal of Universal Computer Science* 15.1 (Jan. 1, 2009), pp. 273–303. DOI: 10.3217/jucs-015-01-0273.

[LW96] Chang Li and X Sean Wang. "A data model for supporting on-line analytical processing". In: *Proceedings of the fifth international conference on Information and knowledge management - CIKM '96*. New York, New York, USA: ACM Press, 1996, pp. 81–88. ISBN: 0897918738. DOI: 10.1145/238355.238444.

[Man+03] Andreas Maniatis et al. "CPM: A Cube Presentation Model for OLAP". In: *Data Warehousing and Knowledge Discovery*. Ed. by Yahiko Kambayashi, Mukesh Mohania, and Wolfram Woss. Vol. 2737. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2003, pp. 4–13. ISBN: 978-3-540-40807-9. DOI: 10.1007/978-3-540-45228-7_2.

[Man+05] A. Maniatis et al. "A presentation model & non-traditional visualization for OLAP". In: *International Journal of Data Warehousing and Mining (IJDWM)* 1.1 (2005), pp. 1–36.

[MEP02] Maarten de Mol, Marko C. J. D. van Eekelen, and Marinus J. Plasmeijer. "Theorem Proving for Functional Programmers". In: *Selected Papers from the 13th International Workshop on Implementation of Functional Languages*. IFL '02. London, UK, UK: Springer-Verlag, 2002, pp. 55–71. ISBN: 3-540-43537-9.

[MI06] Konstantinos Morfonios and Yannis Ioannidis. "CURE for cubes: cubing using a ROLAP engine". In: *Proceedings of the 32nd international conference on Very large data bases*. VLDB '06. Seoul, Korea: VLDB Endowment, 2006, pp. 379–390.

[Mil56] George A. Miller. "The magical number seven, plus or minus two: some limits on our capacity for processing information". In: *Psychological Review* 63.2 (1956), pp. 81–97.

[MS97] Arunprasad P. Marathe and Kenneth Salem. "A Language for Manipulating Arrays". In: *Proceedings of the 23rd International Conference on Very Large Data Bases*. Morgan Kaufmann Publishers Inc. 1997, pp. 46–55.

[MZ04] Elzbieta Malinowski and Esteban Zimnyi. "OLAP Hierarchies: A Conceptual Perspective". In: *Advanced Information Systems Engineering*. Ed. by Anne Persson and Janis Stirna. Vol. 3084. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2004, pp. 19–35. ISBN: 978-3-540-22151-7. DOI: 10.1007/978-3-540-25975-6_34.

[Neg04] Solomon Negash. "Business Intelligence". In: *Communications of the Association for Information Systems* 13 (2004), pp. 177–195.

[NTW00] Thanh Binh Nguyen, A Min Tjoa, and Roland Wagner. "An object oriented multidimensional data model for OLAP". In: *Web-Age Information Management* (2000), pp. 69–82.

[PA07] Rinus Plasmeijer and Peter Achten. "A Conference Management System based on the iData Toolkit". In: *Implementation and Application of Functional Languages*. Ed. by Zoltán Horváth, Viktória Zsók, and Andrew Butterfield. Vol. 4449. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2007, pp. 108–125. ISBN: 978-3-540-74129-9. DOI: 10.1007/978-3-540-74130-5_7.

[PJ01] T.B. Pedersen and C.S. Jensen. "Multidimensional database technology". In: *Computer* 34.12 (2001), pp. 40–46. ISSN: 00189162. DOI: 10.1109/2.970558.

[PJ99] T.B. Pedersen and C.S. Jensen. "Multidimensional data modeling for complex data". In: *Data Engineering, 1999. Proceedings., 15th International Conference on*. IEEE. 1999, pp. 336–345.

[Por79] Michael E. Porter. "How competitive forces shape strategy". In: *Harvard Business Review* 57.2 (1979), pp. 137–145. ISSN: 00178012.

[RS01] Daniel Rouach and Patrice Santi. "Competitive Intelligence Adds Value. Five Intelligence Attitudes". In: *European Management Journal* 19.5 (Oct. 2001), pp. 552–559. ISSN: 02632373. DOI: 10.1016/S0263-2373(01)00069-X.

[Sar97] Sunita Sarawagi. "Indexing OLAP Data". In: *Data Engineering Bulletin* 20.1 (1997), pp. 36–43.

[Teg99] David P. Tegarden. "Business information visualization". In: *Communications of the AIS* 1.1 (Jan. 1999), p. 4.

[Tes01] Olivier Teste. "Towards Conceptual Multidimensional Design in Decision Support Systems". In: *5th East-European Conference on Advances in Databases and Information Systems*. 2001, p. 10.

[Vas98] Panos Vassiliadis. "Modeling Multidimensional Databases, Cubes and Cube Operations". In: *Proceedings of the 10th International Conference on Scientific and Statistical Database Management*. Vol. 0. SSDBM '98. IEEE Computer Society, 1998, pp. 53–62. ISBN: 0-8186-8575-1. DOI: 10.1109/SSDM.1998.688111.

[Ves91] Iris Vessey. "Cognitive Fit: A Theory-Based Analysis of the Graphs Versus Tables Literature". In: *Decision Sciences* 22.2 (1991), pp. 219–240. ISSN: 1540-5915. DOI: 10.1111/j.1540-5915.1991.tb00344.x.

[Wat09] Hugh J. Watson. "Tutorial: Business Intelligence - Past, Present, and Future". In: *Communications of the Association for Information Systems* 25.1 (2009), p. 39.

Other

[Bur11] Dan Burrows. *The intelligent business and business intelligence*. White paper. Aug. 2011.

[CCS93] E F Codd, S B Codd, and C T Salley. *Providing OLAP to User-Analysts: An IT Mandate*. Article. E.F. Codd Associates, 1993, pp. 1–20.

[Cou09] Fred Coulson. *The 3 normal form: A Tutorial*. Tutorial. 2009, pp. 1 –19. URL: http://phlonx.com/resources/nf3/.

[Cou95] The OLAP Council. *OLAP AND OLAP Server Definitions*. Jan. 1995. URL: http://www.olapcouncil.org/research/glossaryly.htm (visited on 07/23/2012).

[Dec] Decos. *Onderzoek naar zaaksystemen*. URL: http://www.decos.nl/kennis/onderzoek-naar-zaaksystemen.htm (visited on 03/31/2011).

[Dek] Corné Dekker. *Zaakgerichtwerken*. Tech. rep. IenPM, pp. 1–81.

[KK10] Johan van Kampen and Emile Kramers. "Self Service Business Intelligence levert broodnodige productiviteitswinst". *Tiem*, 36:29–31. 2010.

[Lan04] Justin Langseth. *Real-Time Data Warehousing: Challenges and Solutions*. Claraview. Aug. 2, 2004. URL: http://dssresources.com/papers/features/langseth/langseth02082004.html (visited on 04/03/2012).

[Mic10] Steffen Michels. "Building iTask Applications. A GUI Paradigm Based on Workflows". Master Thesis. Radboud University Nijmegen, 2010.

[PL11] Joao Penha-Lopes. *The West African Examinations Council (WAEC) Result Scandal*. 2011. URL: http://www.rimaw.org/rimguide/september2011/article.html (visited on 04/03/2012).

[Pow07] Daniel J. Power. *A Brief History of Decision Support Systems*. Version 4. DSSResources.COM. Apr. 10, 2007. URL: http://DSSResources.COM/history/dsshistory.html (visited on 03/12/2012).

[QL11] Laura S. Quinn and Jay Leslie. *A Few Good Case Management Tools*. idealware. 2011. URL: http://www.idealware.org/articles/few-good-case-management-tools-0 (visited on 03/30/2012).

[Sch] Michael A. Schiff. *Business intelligence : A guide for Midsize Companies*. White paper. SAP Business Objects.

[Str11] Koen Strijbosch. "Adaptive Case Management; A new way of supporting knowledge work". Master Thesis. Radboud University Nijmegen, 2011.

[Tam98] Yin Jenny Tam. "Datacube: Its Implementation and Application in OLAP Mining". Master Thesis. Simon Frasier University, Sept. 1998, pp. 1–108.

[Wel12] Michel van Welie. "Mining Document-centric Process Models; The Product Data Model case". Master Thesis. Eindhoven University of Technology, 2012.

[ZTN96] Yihong Zhao, Kristin Tufte, and Jeffrey F. Naughton. *On the Performance of an Array-based ADT for OLAP workloads*. Tech. rep. CS-TR-96-1313. University of Wisconsin-Madison, CS Department, May 1996.

[Cor10] Cordys Software. *Cordys Case Management Leverage the Knowledge in the Enterprise*. Brochure. Cordys Software, 2010.

[For11] Forrester Consulting. *Self-Service : An Essential Capability Of BI*. Tech. rep. Forrester Consulting, Apr. 2011, pp. 1–14.

[Oce11] Oce-Technologies BV. *The dossierflow principle*. Presentation. Version 6. 2011.

[Wer04] Werkgroep GFO-Zaken. *Zaken in zicht*. Tech. rep. Vereniging van Nederlandse Gemeenten, Sept. 2004.

Appendix A

OLAP Operations

Because the meaning of the different operations are already explained in § 4.3.3, we will concentrate on the algebra. The input, output and notation will be given.

Selection
Input
- cube $C_I = \langle C, A, f, d, O, L \rangle$
- (compound) predicate P
Output
cube $C_O = \langle C, A, f, d, O, L_O \rangle$
where:
- $L_O \subseteq L$ - $L_O = \{l | (l \in L) \wedge (l \text{ satisfies } P)\}$
Notation
$\Sigma_P(C_I) = C_O$

Metric Projection
Input
- cube $C_I = \langle C, A, f, d, O, L \rangle$
- set of projection attributes $S (S \subseteq A_m)$
Output
cube $C_O = \langle C, A_O, f_O, d, O, L_O \rangle$
where:
- $A_O = S \cup A_d$
- $f_O(c) = f(c) \cap A_O$
- $L_O = \{l_O | \exists l \in L, l_{\alpha_O} = l_\alpha, l_{\chi_O} = \langle l_\chi[s_1], l_\chi[s_2], ..., l_\chi[s_n] \rangle\}$
- $\{s_1, s_2, ..., s_n\} = S$

Notation
$\Pi_S^M(C_I) = C_O$

Slice
The slice operation is a special *selection* operator for which hold that input predicate P is $a = v$ where:
- $a \in A$ and $|a| = 1$
- $v \in \mathbf{dom}(a)$ and $|v| = 1$

Dice
The dice operation is a special *selection* operator.
For every atomic predicate p_i in P holds that:
$a = \{v\}$ where:
- $d(g(a)) = 1$
- $\forall_{v_i \in v} v_i \in \mathbf{dom}(a)$ and $|\{v\}| \geq 1$

Aggregate
Input
- cube $C_I = \langle C, A, f, d, O, L \rangle$
- an aggregate function F
- a set of grouping attributes G
- a metric attribute to aggregate met

where
- $G \subseteq A_D$
- $met \in A_m$

Output
cube $C_O = \langle C_O, A_O, f_O, d_O, O_O, L_O \rangle$
where:
- $\{AGG\}$ is the name for aggregated metrics
- $\{agg\}$ is the set of aggregated measures
- $C_O = \{c | c \in C, \exists x, x \in G, c \in f(c)\} \cup \{AGG\}$
- $A_O = G \cup \{agg\}$
- $f_O = \begin{cases} \{\langle x, y \rangle | x \in C_O, \quad (\exists \langle x, z \rangle \in f, \\ \qquad\qquad\qquad\qquad x \neq \{AGG\}, y = \{a | a \in (z \cap A_O)\}) \\ \qquad\qquad\qquad\qquad \vee (\exists \langle x, z \rangle \in f, \\ \qquad\qquad\qquad\qquad x = \{AGG\}, y = \{a | a \in (z \cap A_O) \cup \{agg\}\})\} & \text{if } \exists \langle \{AGG\}, z \rangle \in f, \\ \{\langle x, y \rangle | x \in C_O, (\exists \langle x, z \rangle \in f, y = \{a | a \in (z \cap A_O)\})\} \\ \cup \{\langle \{AGG\}, \{agg\} \rangle\} & \text{otherwise.} \end{cases}$

- $\forall x \in C, d_O(x) = \begin{cases} d(x) & \text{if } \{AGG\} \in C, \\ d(x) \cup \langle AGG, 0 \rangle & \text{otherwise.} \end{cases}$

Notation

$\Gamma_{F,G,met}(C_I) = C_O$

Drill-down

For the *Drill-down* operation more granular data is necessary, the metrics have to be reloaded from the source to display (or calculate) the correct values.
A new attribute is added to the cube, as well as additional partial order for the new attribute.

Input

- cube $C_I = \langle C, A, f, d, O, L \rangle$
- the attribute which will be added. (a_n)
- set of partial order for the added attribute (o_{new})

Output

cube $C_O = \langle C, A_O, f_O, d_O, O_O, L_O \rangle$

where

- $A_O = A \cup a_n$
- $\exists c, c \in C, d(c) = 1, f_O(c) = a_n$
- $d_O(g(a_n)) = 1$
- $O_O = O - O(g(a_n)) + [g(a_n) -> O(g(a_n)) \cup o_{new}]$
- the cube cell have the new data.

Notation

$\Omega_{a_n, o_{new}}(C_I) = C_O$

Roll-up

The Roll-up operation is a combined version of a union of aggregations on the same input cube. A standard roll-up operation involves only one attribute to roll-up on, this could be defined by the smallest member of a partial order set if only one hierarchy can exists per characteristic (dimension). Because this is not the case we specify the attribute.

Input

- cube $C_I = \langle C, A, f, d, O, L \rangle$
- attribute to roll-up a
- set of tuples for each metric the aggregation function must be specified $MA = \{\langle met_1, F_1 \rangle, \langle met_2, F_2 \rangle, ..., \langle met_n, F_n \rangle\}$

where:

- $|MA| = \Sigma_{c \in C} |f(c)| \text{where} d(c) = 1$

Output

cube $C_O = \Gamma_{F_1, A_D - a, met_1}(C_I) \cup \Gamma_{F_2, A_D - a, met_2}(C_I) \cup ... \cup \Gamma_{F_n, A_D - a, met_n}(C_I)$

Notation

$\mho_{a, MA}(C_I) = C_O$

Rename (set operator)
Input
- Set S_I ($S_I = \{s_{I1}, s_{I2}, ..., s_{In}\}$)
- New name: N
Output
$S_O = \{N.s_{I1}, N.s_{I2}, ..., N.s_{In}\}$
Notation
$S_O = \Delta_N(S_I)$

Add Dimension
Input
- cube $C_I = \langle C, A, f, d, O, L\rangle$
- new dimension c_{new}
Output
- cube $C_I = \langle C_O, A, f, d_O, O_O, L\rangle$
where:
- $C_O = C \cup \{c_t\}$
- $d_O = d + [c_t- > 1]$
- $O_O = O + \langle c_t, \{\}\rangle$
Notation
$\rho_{c_t}(C_I) = C_O$

Transfer
Input
- cube $C_I = \langle C, A, f, d, O, L\rangle$
- attribute to transfer a_t
- dimension to transfer attribute to: c_t
- set of partial order for the added attribute (o_{new})
where:
- $d(g(a)) = 1$
- $c \in C$ and $d(c) = 1$
Output
- cube $C_I = \langle C, A, f_O, d, O_O, L\rangle$
where:
- $f_O = f - f(g(a_t)) + [g(a_t) \rightarrow f(g(a_t) - a_t)] + [c_t- > a_t]$

- $O_O = \begin{cases} o_i \setminus \{\langle x, y\rangle | x = a_t \vee y = a_t\} & \text{if } i = g(a_t) \\ o_i \cup o_{new} & \text{if } i = c_t \\ o_i & \text{otherwise} \end{cases}$

Notation

$\tau^{a_t}_{c_t, o_{new}}(C_I) = C_O$

Cartesian Product

Input

- Cube $C_1 = \langle C_1, A_1, f_1, d_1, O_1, L_1\rangle$
- Cube $C_2 = \langle C_2, A_2, f_2, d_2, O_2, L_2\rangle$

Output

Cube $C_O = \langle C_O, A_O, f_O, d_O, O_O, L_O\rangle$

where:

- $C_O = \Delta_{C_1}(C_1) \cup \Delta_{C_2}(C_2)$
- $A_O = \Delta_{A_1}(A_1) \cup \Delta_{A_2}(A_2)$
- $L_O = \{l_O | \exists l_1, \exists l_2, l_1 \in L_1, l_2 \in L_2, L_{\alpha_O} = L_{\alpha_1} \cdot L_{\alpha_2}, L_{\chi_O} = L_{\chi_1} \cdot L_{\chi_2}\}$
- $\forall c_i \in (C_1 \cap C_2)$,

$f_O = \begin{cases} f_1 & \text{when applied to } c_i \in C_1.c_i, \\ f_2 & \text{when applied to } c_j \in C_2.c_j \end{cases}$

$d_O = \begin{cases} d_1 & \text{when applied to } c_i \in C_1.c_i, \\ d_2 & \text{when applied to } c_j \in C_2.c_j \end{cases}$

- $\forall a_i \in (f(C_1) \cap f(C_2))$,

$O_O = \begin{cases} O_1 & \text{when applied to } a_i \in f(C_1), \\ O_2 & \text{when applied to } a_j \in f(C_2) \end{cases}$

Notation

$C_1 \otimes C_2 = C_O$

Join

Input

- Cube $C_1 = \langle C_1, A_1, f_1, d_1, O_1, L_1\rangle$
- Cube $C_2 = \langle C_2, A_2, f_2, d_2, O_2, L_2\rangle$

where:

- $cd = D_1 \cap D_2 \neq \emptyset$
- $A_{cd} = \{a_{cd1}, a_{cd2}, ..., a_{cdm}\}$
- $A_{cd} = \cup_{cd_i \in cd} f(cd_i)$
- $A_{cd} \subseteq A_d$

Output

Cube $C_O = \Sigma_P(C_1 \otimes C_2)$

where:

- $P = [(C_1 \cdot a_{cd1} = C_2 \cdot a_{cd1}) \wedge (C_1 \cdot a_{cd2} = C_2 \cdot a_{cd2}) \wedge, ..., \wedge (C_1 \cdot a_{cdm} = C_2 \cdot a_{cdm})]$

Notation

$C_1 \Theta C_2 = C_O$

Union

Input

- Cube $C_1 = \langle C_1, A_1, f_1, d_1, O_1, L_1 \rangle$
- Cube $C_2 = \langle C_2, A_2, f_2, d_2, O_2, L_2 \rangle$

where C_1 and C_2 are union-compatible.

Output

Cube $C_O = \langle C_O, A_O, f_O, d_O, O_O, L_O \rangle$

where:

- $C_O = C_1 = C_2$
- $A_O = A_1 = A_2$
- $f_O = f_1 = f_2$
- $d_O = d_1 = d_2$
- $O_O = O_1 = O_2$
- $L_O = L_1 \cup L_2$

Notation

$c_1 \cup C_2 = C_O$

Difference

Input

- Cube $C_1 = \langle C_1, A_1, f_1, d_1, O_1, L_1 \rangle$
- Cube $C_2 = \langle C_2, A_2, f_2, d_2, O_2, L_2 \rangle$

where C_1 and C_2 are union-compatible.

Output

- $C_O = C_1 = C_2$
- $A_O = A_1 = A_2$
- $f_O = f_1 = f_2$
- $d_O = d_1 = d_2$
- $O_O = O_1 = O_2$
- $L_O = L_1 - L_2$

Notation

$C_1 \theta C_2 = C_O$

Pull

Input

- cube $C_I = \langle C, A, f, d, O, L \rangle$
- the attribute to transform a_t
- the characteristic name c_t where the attribute should move to.
- a user specified set of new ordering relations between a_t and the elements of $f(c_t)$: o_n

where:

- $c_t \notin M \wedge (c_t \in D \vee c_t \notin C)$

Output

cube $C_O = \langle C_O, A_O, f_O, d_O, O_O, L_O \rangle$

where:

- $C_O = C \cup \{c_t\}$
- $f_O = f - f(g(a_t)) + [g(a_t) \rightarrow f(g(a_t) - a_t)] + [c_t - > a_t]$
- $O_O = \begin{cases} o_i \setminus \{\langle x, y \rangle | x = a_t \vee y = a_t\} & \text{if } i = g(a_t) \\ o_i \cup o_n & \text{if } i = c_t \\ o_i & \text{otherwise} \end{cases}$
- $l_O = \{l_O | \exists l \in L, l_{\alpha_O} = l_\alpha \circ \langle l_\chi[a_t] \rangle, l_{\chi O} = l_\chi - \langle l_\chi[a_t] \rangle\}$
- $d(c_t) = 1$

Notation

$\Phi a_t, c_t(C_I) = C_O$

Push

Input

- cube $C_I = \langle C, A, f, d, O, L \rangle$
- the attribute to transform a_t
- the characteristic name c_t where the attribute should move to.
- a user specified set of new ordering relations between a_t and the elements of $f(c_t)$: o_n

where:

- $c_t \notin D \wedge (c_t \in M \vee c_t \notin C)$

Output

cube $C_O = \langle C_O, A_O, f_O, d_O, O_O, L_O \rangle$

where:

- $C_O = C \cup \{c_t\}$
- $f_O = f - f(g(a_t)) + [g(a_t) \rightarrow f(g(a_t) - a_t)] + [c_t - > a_t]$
- $O_O = \begin{cases} o_i \setminus \{\langle x, y \rangle | x = a_t \vee y = a_t\} & \text{if } i = g(a_t) \\ o_i \cup o_n & \text{if } i = c_t \\ o_i & \text{otherwise} \end{cases}$
- $l_O = \{l_O | \exists l \in L, l_{\alpha_O} = l_\alpha - \langle l_\alpha[a_t] \rangle, l_{\chi O} = l_\chi \circ \langle l_\alpha[a_t] \rangle\}$
- $d(c_t) = 0$

Notation

$\Psi_{a_t, c_t, o_n}(C_I) = C_O$

Pivot

The pivot operation is a aggregate operation on two or more grouping attributes.
$\Gamma_{F,G,met}(C_I) = C_O$ with $|G| \geq 2$

Appendix B

iTask

iTask[1] is toolkit for programming workflow management systems with the functional programming language *Clean*[2] and is (largely) developed by the Computing Science department of the Radboud University. This (iTask) toolkit makes it possible to create interactive web applications. The toolkit was originally designed for building workflows, but the language and the toolkit are flexible enough to make it possible to make other web applications than a workflow management system.

In [Jan+10] Jansen et al. describe the basics of the iTask toolkit. From the examples and the text, the elements necessary for making a case management tool can be found. For example tasks are described, delegating a task is described. The tasks can be executed both sequentially as well as parallel. In [Mic10, pp. 69] interaction of the iTask system with files is shown.

A very simplified version of a case management tool is implemented in iTask. This version is able to create cases, delegate a task (one at the time), and close the case. We begin with type definition: (figure B.1)

```
1  ::Case = { caseName  :: String
2           , type      :: CaseType
3           }
4  ::CaseType = Order | Complaint
5
6  ::CaseTask = { taskName      :: String
7               , description  :: Maybe String
8               , type          :: TaskType
9               }
10 ::TaskType = WriteFeedback | CompleteOrder
```

[1] http://wiki.clean.cs.ru.nl/ITasks
[2] http://wiki.clean.cs.ru.nl/Clean

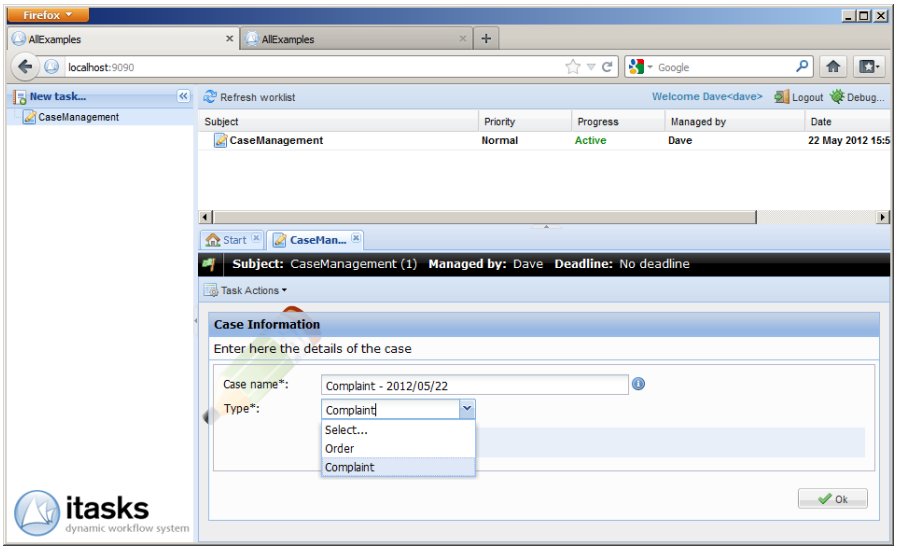


Figure B.1: Enter the case information

Then we define the basic of the workflow (creating a case and handling the case)

```
11 caseHandling :: Task Void
12 caseHandling
13   =   defineCase
14   >>= \zk ->
15       handleCase zk
16
17 defineCase :: Task Case
18 defineCase
19   = enterInformation "Case Information"
20     "Enter here the details of the case"
```

We continue with defining the handling of the case. First we let the user choose between ending the case or delegating it. (figure B.2)

```
21 handleCase :: Case -> Task Void
22 handleCase zk
23   =   defineNextStep
24   >>= \nextstep ->  case nextstep of
25     "Close Case"    = showMessageAbout "Case Closed"
26                       "This case is closed:" zk >>| stop
27     "Delegate Case"  = delegateCase zk >>= \newzk ->
28                       handleCase zk
29
30 defineNextStep :: Task String
```

```
defineNextStep
  = enterChoice "Next Step" "What would you like to do next?"
                ["Close Case", "Delegate Case"]
```

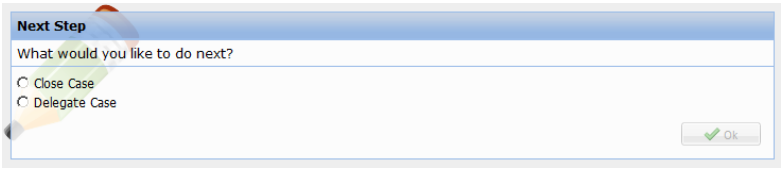


Figure B.2: Choose the next step

For delegating the case, we first show the case, then define the task and choose the delegate (figures B.3, B.4 and B.5)

```
delegateCase :: Case -> Task Case
delegateCase zk
  =  showMessageAbout "Delegate Case" "This case will be delegated:" zk
  >>| defineTask
  >>=  \ct ->
    defineDelegate
  >>= \delegate ->
    waitForDelegation zk ct delegate
  >>|  showMessageAbout "Task Completed" "This task is completed:" zk

defineTask :: Task CaseTask
defineTask
  = enterInformation "Define task" "Enter the details of the task"

defineDelegate :: Task User
defineDelegate
  = enterInformation "Choose delegate"
                     "Choose the person you want to delegate to"
```

Figure B.3: Show the case information before it is delegated

and finally we wait for the delegate to finish his task. (figure B.6)

```
waitForDelegation :: Case CaseTask User -> Task Void
waitForDelegation zk ct us
```

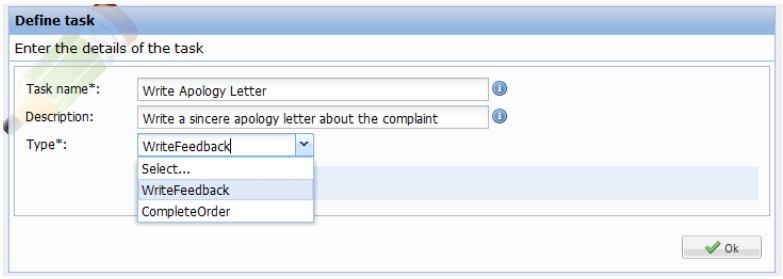


Figure B.4: Define the Task

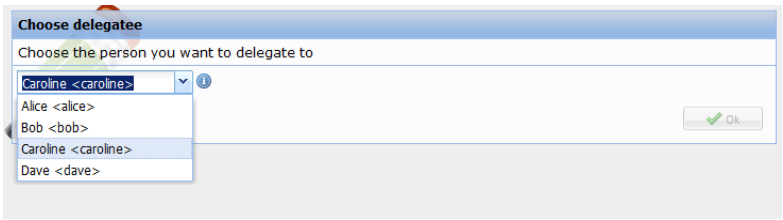


Figure B.5: Choose the delegatee

```
54    =   us @: (showMessageAbout ("Case Delegation:" +++ zk.caseName)
55                                "Handle this case" ct)
56    >>| stop
```

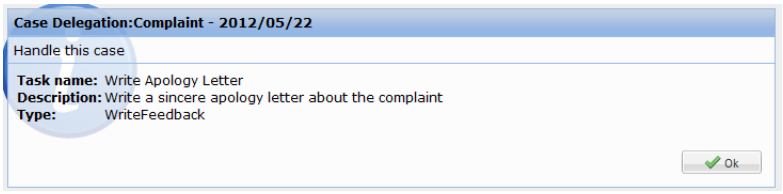


Figure B.6: Show the delegatee the case info

The original user can see when the task is complete (figure B.7) and can then choose to complete the case (figure B.8)

Evaluation of iTask

The previous example showed that with a limited number of lines of code (approx. 50 lines) a (very simple) case management tool can be written. One of the advantages of writing a case management tool in a functional language is that proving the correctness of the code is possible (for the Clean language there is a functional theorem prover Sparkle [MEP02]). In 2006 Plasmeijer and Achten showed a conference management system with the iData toolkit (which can be seen as a very early beginning of the iTask toolkit)[PA07]. This and the example

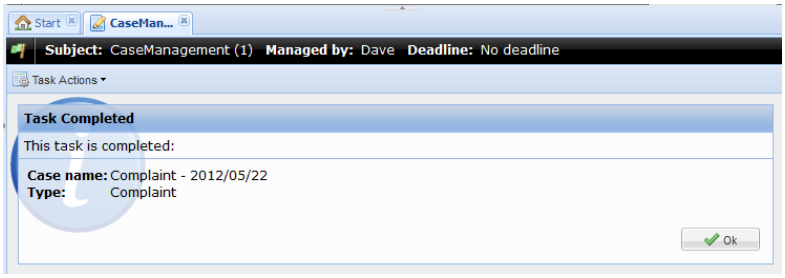


Figure B.7: The task is complete

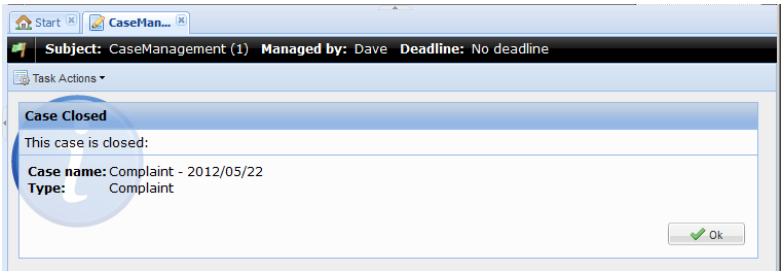


Figure B.8: Case is complete

showed indicates the possibilities or writing a case management tool purely in a functional language. There are (most) probably some issues for implementing a program like Serenga with iTask in Clean, but making such a program can reveal possible shortcomings in the iTask toolkit. For commercial use an implementation in Clean is not (yet) feasible because some features that are important for users are not possible (as far as the author knows of). One of this features is integration in office suites and / or existing collaboration tools.

Appendix C

Self-Service BI

An answer to the first questions defined in §5.2 is given in §5.6.1. In this appendix the answers on the other questions (except on the question 'what are problem cases') are illustrated.

Question	Figure
How do the users perform?	Figures 5.9,5.10 and 5.11
How many cases are assigned to a user?	Figure C.2
How many cases are (being) processed?	Figure C.1
How many sub cases are there?	Figure C.3
How many cases of type X result in case of type Y?	Figure C.3
How many documents are there in a case?	Figure C.9
What is the time of adding (certain) documents to a case?	Figure C.10
What is the time between an activity and a milestone?	Figure C.8
What is the turnaround time of activities?	Figure C.7
What are problem cases?	- (Not defined)
How often are cases being delegated/taken over?	Figure C.4
How many users are involved in a cases?	Figure C.6
How many cases of type X are finished and approved per BU	Figure C.5
How many cases of type X are finished and declined per BU	Figure C.5
What is the average time from milestone X to milestone Y?	Figure C.8 (by lack of milestones)

Table C.1: Fulfilling the information need with Excel

Cases

First we look at the number of cases that are being processed. This query shows how the measures *Open Cases* and *Open Case Percentage* are being calculated. These measures were not stored in the cube. A table is shown because a graph shows a comparison, but the values are not comparable. The table is shown in figure C.1.

```
WITH MEMBER [Measures].[Open Cases]
  AS [Measures].[Case Count]-[Measures].[Finished Cases]
MEMBER [Measures].[Open Case Percentage]
  AS [Measures].[Open Cases]/[Measures].[Case Count]
SELECT
  NON EMPTY { [Measures].[Open Case Percentage],
              [Measures].[Open Cases]
            } ON COLUMNS,
  NON EMPTY { ([Case].[CaseType].[CaseType].ALLMEMBERS )
            } DIMENSION PROPERTIES MEMBER_CAPTION, MEMBER_UNIQUE_NAME ON ROWS
FROM [Serenga2]
```

	Open Cases	Open Case Percentage
Account Manager	7	29,17%
Bestelbon	0	0,00%
Boekjaar	1	25,00%
Contract	3	100,00%
Hierarchy	7894	100,00%
Klacht	46	4,37%
Klacht CPM	1	100,00%
Klacht P&S	165	5,11%
Klacht SP	1	0,33%
Klant	96457	99,99%
Koopovereenkomst	0	0,00%
Managementrapportage	40	100,00%
Mantelovereenkomst	216	84,05%
Offerte	161	34,70%
Offerte/Creditcheck	729	12,88%
Order	6830	38,45%
Project	159	36,89%
Reports	1	100,00%
Servicewerkbon	0	0,00%
Servicewerkbonnen	0	0,00%
Stand-by offerte	0	0,00%
Suppliesorder	22	0,08%
Tender	10	37,04%
Tender Knowledgebase	8	100,00%
Verhuizing	0	0,00%
Grand Total	**112751**	**10,08048245**

Figure C.1: Number and percentage of open cases per case type

Case assignment

For the number of case assignments we also have to calculate the open cases. The resulting graph (figure C.2) shows the case assignment for one user.

```
WITH MEMBER [Measures].[Open Cases]
  AS [Measures].[Case Count]-[Measures].[Finished Cases]
SELECT
  NON EMPTY { [Measures].[Open Cases]
            } ON COLUMNS,
  NON EMPTY { ([Case].[CaseType].[CaseType].ALLMEMBERS )
            } DIMENSION PROPERTIES MEMBER_CAPTION, MEMBER_UNIQUE_NAME ON ROWS
FROM [Serenga2]
```

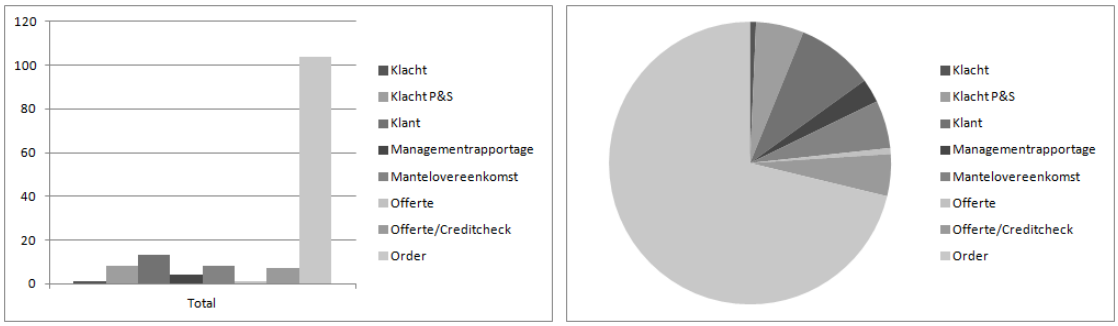


Figure C.2: Number of open cases assigned to a specific user

Sub cases

In order to find out how many cases led to another case, we define that as how many cases have a sub case attached to it. If another definition is wanted (e.g. In stead of having both tender and order as sub case of customer, you might want to have an order case as result of the tender case) than extra information needs to be stored. The table with the relation between parent and sub cases is shown in figure C.3

```
SELECT
  NON EMPTY { [Measures].[Sub Cases Count]
            } ON COLUMNS,
  NON EMPTY { ([Case].[CaseType].[CaseType].ALLMEMBERS *
               [subCase].[CaseType].[CaseType].ALLMEMBERS )
            } DIMENSION PROPERTIES MEMBER_CAPTION, MEMBER_UNIQUE_NAME ON ROWS
FROM [Serenga2]
```

number of Cases	Parent Cases							
Child Cases	Boekjaar	Hierarchy	Klacht SP	Klant	Mantelovereenkomst	Order	Suppliesorder	Grand Total
Account Manager	24							24
Contract		2						2
Klacht				1002				1002
Klacht P&S		1		3158				3159
Klant				3				3
Managementrapportage				38				38
Mantelovereenkomst				239				239
Offerte				343				343
Offerte/Creditcheck				5555				5555
Order				17260	1	21		17282
Project				418	1			419
Suppliesorder							7	7
Tender				25				25
Verhuizing			1					1
Grand Total	**24**	**3**	**1**	**28041**	**2**	**21**	**7**	**28099**

Figure C.3: Number of cases that led to another case

Case delegation

The following query finds the number of handler changes per case type per case (identified by the *DDSNR* attribute). The averages are shown in figure C.4.

```
SELECT
  NON EMPTY { [Measures].[Fact Case Handler Change Count]
            } ON COLUMNS,
  NON EMPTY { ([Case].[CaseType].[CaseType].ALLMEMBERS *
               [Case].[DDSNR].[DDSNR].ALLMEMBERS )
            } DIMENSION PROPERTIES MEMBER_CAPTION, MEMBER_UNIQUE_NAME ON ROWS
FROM [Serenga2]
```

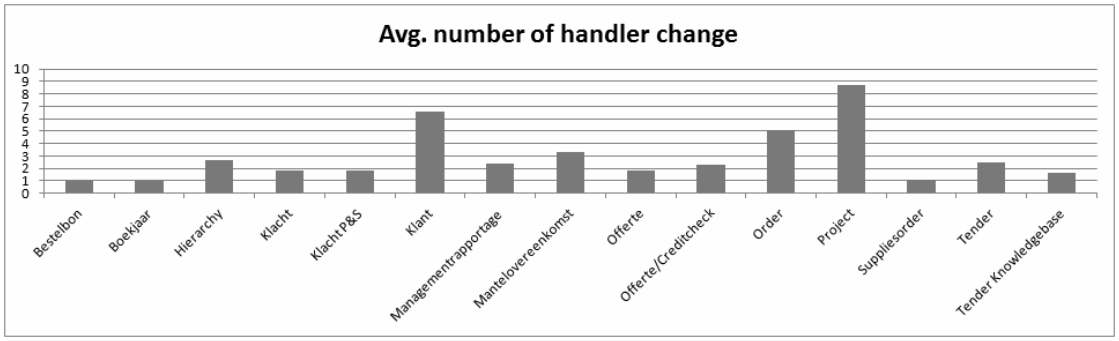


Figure C.4: Average times the handler of a case changed per case type

Case approval

In order to find the approval rate of the cases we retrieve from the database the business unit the cases belong to, the state and as measure the number of cases. The table is shown in figure C.5. The 'Grand Total' include also the cases with other states (open cases and closed cases without approval or decline). The 'Grand Total' row includes also the cases without business unit.

```
SELECT
  NON EMPTY { [Measures].[Case Count]
            } ON COLUMNS,
  NON EMPTY { ([Case].[Order Business Unit].[Order Business Unit].ALLMEMBERS *
               [Case].[State].[State].ALLMEMBERS )
            } DIMENSION PROPERTIES MEMBER_CAPTION, MEMBER_UNIQUE_NAME ON ROWS
FROM [Serenga2]
```

nr. Of Cases	State			
Business Unit	Afgehandeld, afgewezen	Afgehandeld, toegekend	Grand Total *
clf oost		1	52
clf west		3	32
com graphic arts oost		1	23
com graphical key accounts		1	12
com poing		1	7
com reprografen		3	20
ka finance	1	3	28
ka non profit	2	34	112
ka profit	1	1	44
ma oost		1	45
ma west		6	61
ma zuid	1	11	71
ma zuid-west		4	59
obs	1	5	27
ra oost	1	3	63
ra west		2	40
Grand Total *	**484**	**11182**	**162779**

Figure C.5: Cases that are approved versus cases that are not approved

Case Handlers

If we want to know who worked on a case we can look at the case handlers or the activity handlers. The table in figure C.6 shows the current case handler and all the users involved in the activities belonging to the case.

```
SELECT
  NON EMPTY { [Measures].[Fact Activity Count]
            } ON COLUMNS,
  NON EMPTY { ([Case].[CaseType].[CaseType].ALLMEMBERS *
               [Case].[DDSNR].[DDSNR].ALLMEMBERS *
               [Case Handler].[User ID].[User ID].ALLMEMBERS *
               [Activity Handler].[User ID].[User ID].ALLMEMBERS )
            } DIMENSION PROPERTIES MEMBER_CAPTION, MEMBER_UNIQUE_NAME ON ROWS
FROM [Serenga2]
```

Row Labels	Activity Count
Project	16
DDS2010174074	16
[redacted]	16
[redacted]	1
[redacted]	1
[redacted]	8
[redacted]	1
[redacted]	2
[redacted]	1
[redacted]	2
Grand Total	16

Figure C.6: Users that were involved as (activity) handler for a case

Activities

In figure C.7 the average handling time of the different activities from cases with the same case type are compared.

```
SELECT
  NON EMPTY { [Measures].[Avg Time Handling]
            } ON COLUMNS,
  NON EMPTY { ([Activity].[Activity Type].[Activity Type].ALLMEMBERS *
               [Activity].[Case Type].[Case Type].ALLMEMBERS )
            } DIMENSION PROPERTIES MEMBER_CAPTION, MEMBER_UNIQUE_NAME ON ROWS
```

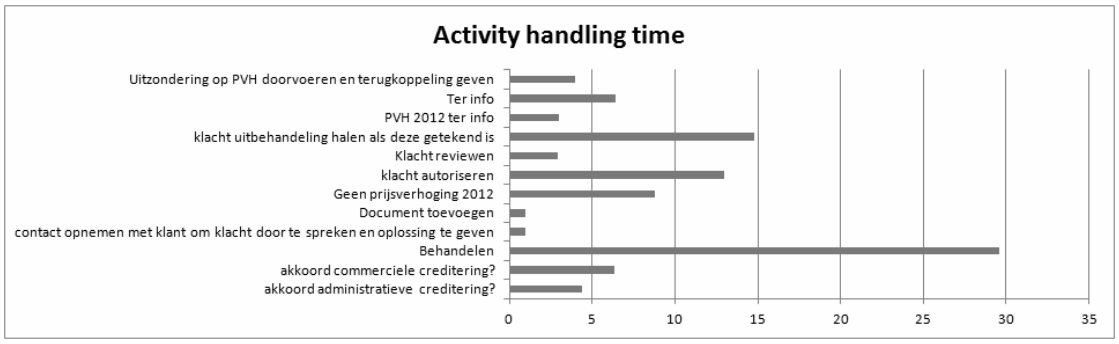


Figure C.7: Average time for handling an activity

Milestones

Because there are no milestones available, we look in figure C.8 at the time between the start of an activity and the end of a case and at the time between the end of an activity and the end of a case.

```
SELECT
  NON EMPTY { [Measures].[Avg Time EndActivity2EndCase],
              [Measures].[Avg Time StartActivity2EndCase]
            } ON COLUMNS,
  NON EMPTY { ([Activity].[Activity Type].[Activity Type].ALLMEMBERS *
               [Activity].[Case Type].[Case Type].ALLMEMBERS )
            } DIMENSION PROPERTIES MEMBER_CAPTION, MEMBER_UNIQUE_NAME ON ROWS
FROM [Serenga2]
```

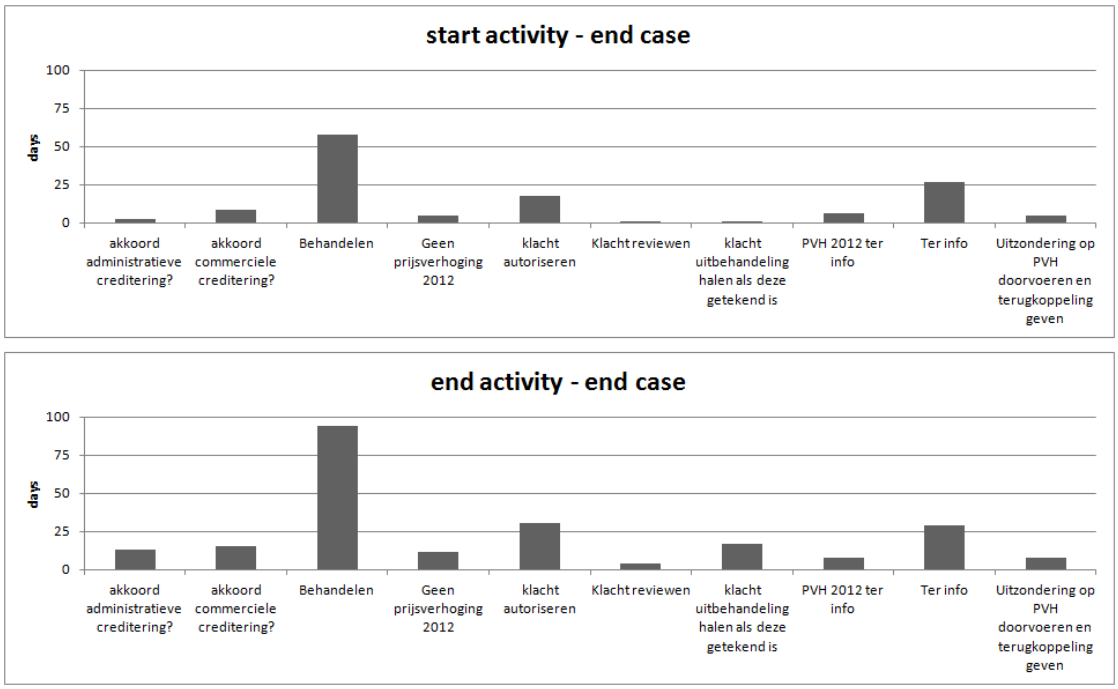


Figure C.8: Average time between start activity and the end of a case and the average time between the end of the activity and the end of the case

Documents

The number of documents can be easily retrieved from the cubes. Instead of having the average calculated in the cube, the average is calculated in the query. The average number of documents

per case as weel as the actual number of documents for six documents of the same case type are shown in figure C.9

```
WITH MEMBER [Measures].[AvgDocsPerCase]
  AS [Measures].[Document Count]/[Measures].[Case Count]
SELECT
  NON EMPTY { [Measures].[AvgDocsPerCase],
              [Measures].[Document Count]
            } ON COLUMNS,
  NON EMPTY { ([Case].[CaseType].[CaseType].ALLMEMBERS )
            } DIMENSION PROPERTIES MEMBER_CAPTION, MEMBER_UNIQUE_NAME ON ROWS
FROM [Serenga2]
```

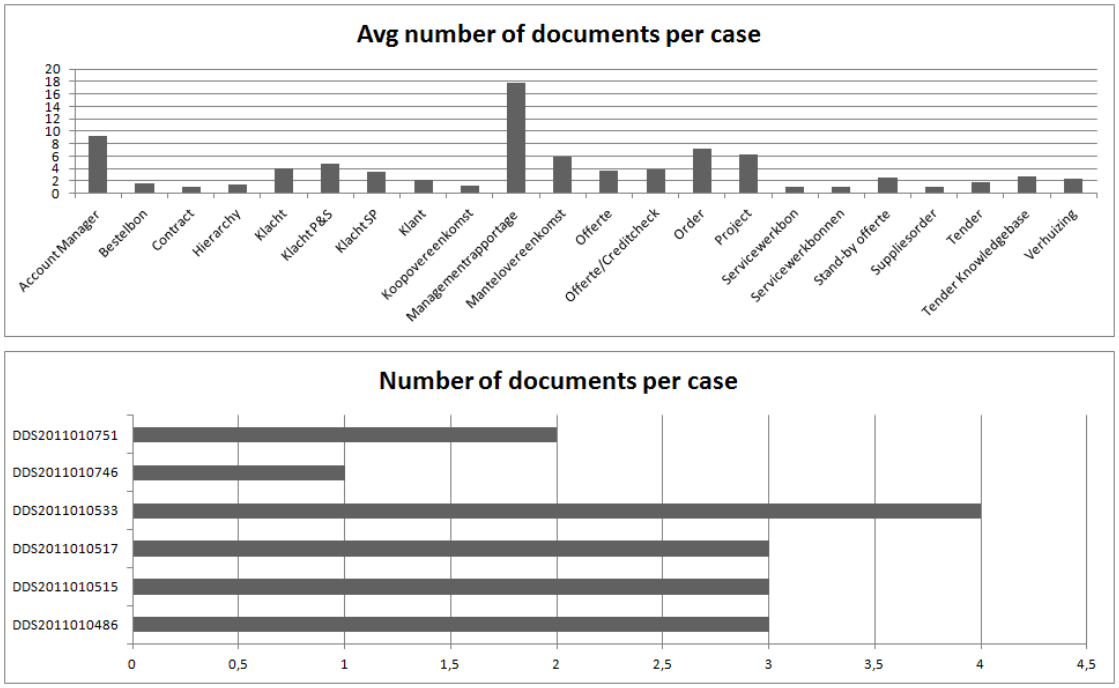


Figure C.9: Number of documents per case(type)

Time of adding documents

The average time to add documents to a case and the time from adding the document to the end of the case are shown in figure C.10.

```
SELECT
  NON EMPTY { [Measures].[Avg Time from Start],
              [Measures].[Avg Time to End]
            } ON COLUMNS,
  NON EMPTY { ([Case].[CaseType].[CaseType].ALLMEMBERS *
               [Document].[Document Type].[Document Type].ALLMEMBERS )
            } DIMENSION PROPERTIES MEMBER_CAPTION, MEMBER_UNIQUE_NAME ON ROWS
FROM [Serenga2]
```

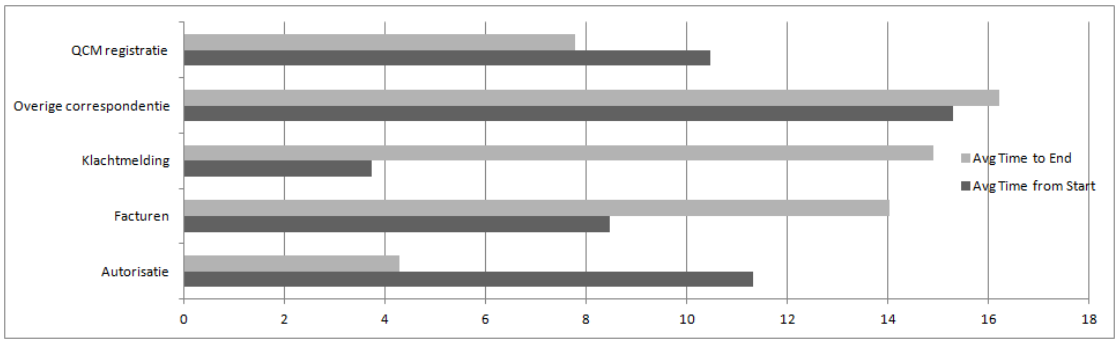


Figure C.10: Time to add documents to a case

Appendix D

Reports

Two reports that were created on the OLTP database are rewritten in order to connect to the OLAP database. The report has a business unit as variable as well as a start end end date for the activity creation date. In the report based on the OLAP cube, this data parameter is removed because of performance problems when querying on specific dates instead of querying on for example months or years. The reports shows the number of cases, the average handling time and the number of cases where the activity 'Order in behandeling nemen' was accepted in one time. In this example it is easily shown that the required query is shorter, better readable and easier to understand or create. The resulting report is shown in figure D.1

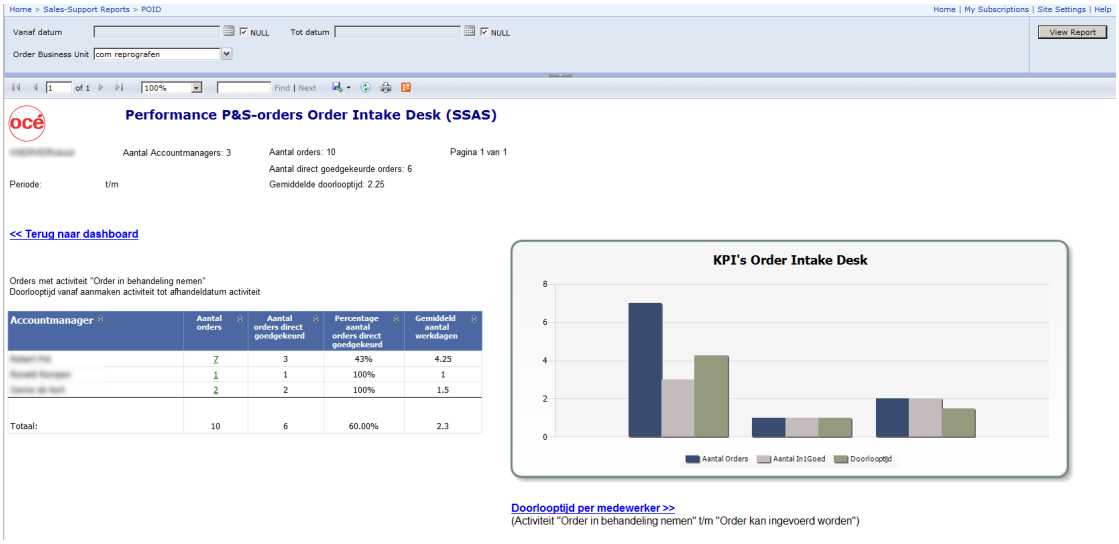


Figure D.1: Performance P&S-order Order Intake Desk report based on the OLAP cube

OLTP-query (SQL)

```
SELECT temp1.Name, temp1.AantalOrders, temp2.Doorlooptijd, temp3.AantalIn1Goed, temp4.BusinessUnit
FROM
        (SELECT
                DISTINCT A.prevusername AS Name,
                COUNT(DISTINCT A.dossierID) AS AantalOrders
        FROM dossierassistanthandlers AS A
        INNER JOIN dossiers AS B ON
                A.dossierID=B.ID AND
                B.DSPid='B00016'
        WHERE A.activity LIKE 'Order in behandeling nemen'
                AND (A.creationdate >= @rpStartDate OR @rpStartDate IS NULL)
                AND (A.creationdate < @rpEndDate OR @rpEndDate IS NULL)
                AND RTRIM(CAST(metadata.query('data(/keyvalues/Order_Business_Unit/value)')
                AS VARCHAR(512)))  IN (@rpBusinessUnit)
        GROUP BY A.prevusername) AS temp1

JOIN

        (SELECT
                A.prevusername AS Name2,
                AVG((DATEDIFF(dd, A.creationdate,(CASE WHEN A.handleddate IS NULL THEN GETDATE()
                        ELSE A.handleddate END)) + 1)
        -(DATEDIFF(wk, A.creationdate,(CASE WHEN A.handleddate IS NULL THEN GETDATE()
                        ELSE A.handleddate END)) * 2)
        -(CASE WHEN DATENAME(dw, A.creationdate) = 'Sunday' THEN 1 ELSE 0 END)
        -(CASE WHEN DATENAME(dw, (CASE WHEN A.handleddate IS NULL THEN GETDATE()
                        ELSE A.handleddate END)) = 'Saturday' THEN 1 ELSE 0 END)-1)
        AS Doorlooptijd
        FROM dossierassistanthandlers AS A
        INNER JOIN dossiers AS B ON
                A.dossierID=B.ID AND
                B.DSPid='B00016'
        WHERE A.activity LIKE
                'Order in behandeling nemen' AND
                (A.creationdate >= @rpStartDate OR @rpStartDate IS NULL) AND
                (A.creationdate < @rpEndDate OR @rpEndDate IS NULL) AND
                RTRIM(CAST(metadata.query('data(/keyvalues/Order_Business_Unit/value)')
                        AS VARCHAR(512))) IN (@rpBusinessUnit)
        GROUP BY A.prevusername) AS temp2

ON temp1.Name=temp2.Name2

JOIN

        (SELECT
                A.prevusername AS Name3,
                COUNT(DISTINCT A.dossierID) AS AantalIn1Goed
        FROM dossierassistanthandlers AS A
        INNER JOIN dossiers AS B ON
                A.dossierID=B.ID AND
                B.DSPid='B00016'
        WHERE A.activity LIKE
                'Order in behandeling nemen'         AND
                A.state LIKE 'Afgehandeld, toegekend'         AND
                (A.creationdate >= @rpStartDate OR @rpStartDate IS NULL) AND
                (A.creationdate < @rpEndDate OR @rpEndDate IS NULL) AND
                RTRIM(CAST(metadata.query('data(/keyvalues/Order_Business_Unit/value)')
                        AS VARCHAR(512))) IN (@rpBusinessUnit)         AND
```

```
                A.dossierID NOT IN
                        (SELECT
                                A.dossierID
                                FROM dossierassistanthandlers AS A INNER JOIN
                                        dossiers AS B ON A.dossierID=B.ID AND
                                        B.DSPid='B00016'
                                WHERE
                                        A.activity LIKE 'Order in behandeling nemen' AND
                                        A.state LIKE 'Afgehandeld, afgewezen')
                GROUP BY A.prevusername) AS temp3
ON temp1.Name=temp3.Name3

JOIN

        (SELECT
                DISTINCT A.prevusername AS Name4,
                (RTRIM(CAST(B.metadata.query('data(/keyvalues/Order_Business_Unit/value)')
                        AS VARCHAR(512)))) AS BusinessUnit
                FROM dossierassistanthandlers AS A
                INNER JOIN dossiers AS B ON
                        A.dossierID=B.ID AND
                        B.DSPid='B00016'
                WHERE
                  (RTRIM(CAST(B.metadata.query('data(/keyvalues/Order_Business_Unit/value)')
                AS VARCHAR(512)))) =
                (SELECT Temp1.BU FROM
                        (SELECT TOP 1
    (RTRIM(CAST(dossiers.metadata.query('data(/keyvalues/Order_Business_Unit/value)')
                        AS VARCHAR(512)))) AS BU,
                dossierassistanthandlers.creationdate
                FROM dossierassistanthandlers
                        INNER JOIN dossiers ON
                                dossierassistanthandlers.dossierID = dossiers.ID
                        WHERE dossierassistanthandlers.prevusername = A.prevusername
                        ORDER BY dossierassistanthandlers.creationdate DESC)
                        AS Temp1)
                ) AS temp4
        ON temp1.Name=temp4.Name4
ORDER BY temp1.Name
```

OLAP-query (MDX)

```
WITH
        MEMBER [Measures].[Order in 1 goed]
                AS IIF(        [Activity].[Case-Activity].LEVEL.Name = 'Case ID'
                                AND ([Activity].[Case-Activity].CurrentMember,
                                        [Activity].[State].&[Afgehandeld, afgewezen],
                                        [Measures].[Case Count]) =0
                                AND ([Activity].[Case-Activity].CurrentMember,
                                        [Activity].[State].&[Afgehandeld, toegekend],
                                        [Measures].[Case Count]),
                                1,null)
        MEMBER [Measures].[Sum in 1 goed]
                AS SUM([Activity].[Case-Activity].Children *
                       [Measures].[Order in 1 goed])
SELECT
        NON EMPTY { [Measures].[Case Count],
                    [Measures].[Avg Time Handling],
                    [Measures].[Sum in 1 goed]
                  } ON COLUMNS,
```

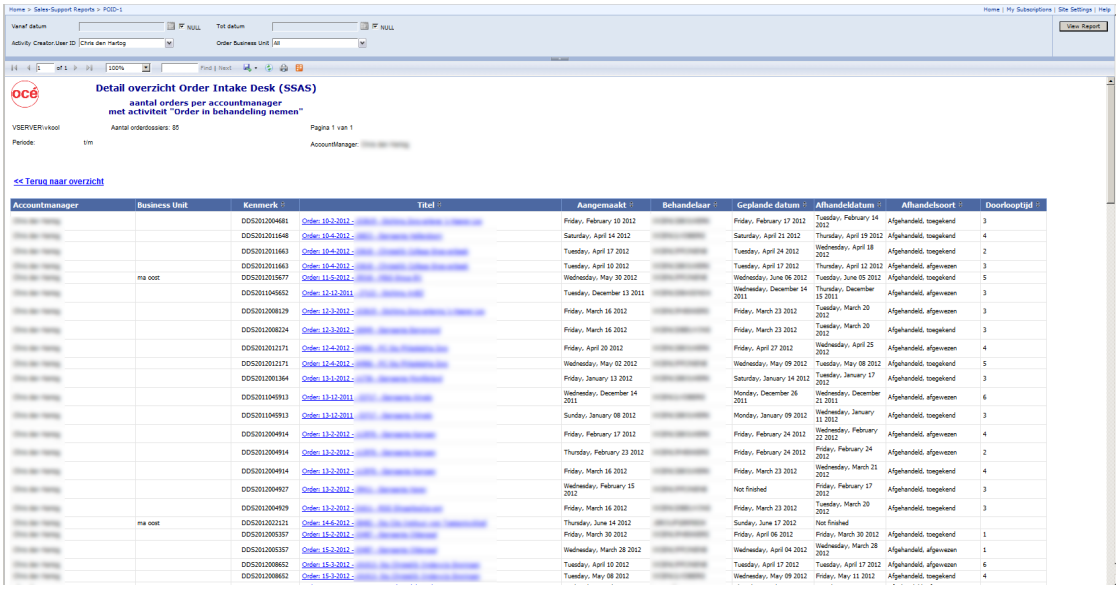
```
        NON EMPTY { ([Activity Creator].[User ID].[User ID].ALLMEMBERS )
                  } DIMENSION PROPERTIES MEMBER_CAPTION, MEMBER_UNIQUE_NAME ON ROWS
FROM [Serenga2]
  WHERE ( IIF(  STRTOSET('@CaseOrderBusinessUnit', CONSTRAINED).Count = 1,
                STRTOSET('@CaseOrderBusinessUnit', CONSTRAINED),
                [Case].[Order Business Unit].currentmember )
        , [Activity].[Case Type].&[Order]
        , [Activity].[Activity Type].&[Order in behandeling nemen] )
```

Second report

The second report shows more details for the selected account manager. It shows the cases, the current handler, and the dates associated with the activity 'Order in behandeling nemen'. The reports is shown in figure D.2

OLAP-query for the detailed report (MDX)

```
SELECT
  NON EMPTY {        [Measures].[Fact Activity Count],
              [Measures].[Avg Time Handling]
            } ON COLUMNS,
  NON EMPTY { ( [Case].[Title].[Title].ALLMEMBERS *
                [Case].[DDSNR].[DDSNR].ALLMEMBERS *
                [Case].[Order Business Unit].[Order Business Unit].ALLMEMBERS *
                [Activity].[Activity ID].[Activity ID].ALLMEMBERS *
                [Activity].[Creator].[Creator].ALLMEMBERS *
                [Activity].[Handler].[Handler].ALLMEMBERS *
                [Activity].[State].[State].ALLMEMBERS *
                [Activity Start Date].[Date].[Date].ALLMEMBERS *
                [Activity Due Date].[Date].[Date].ALLMEMBERS *
                [Activity Handled Date].[Date].[Date].ALLMEMBERS *
                [Activity Creator].[User ID].[User ID].ALLMEMBERS )
            } DIMENSION PROPERTIES MEMBER_CAPTION, MEMBER_UNIQUE_NAME ON ROWS
FROM (
  SELECT ( STRTOSET(@CaseOrderBusinessUnit, CONSTRAINED),
           STRTOSET(@ActivityCreatorUserID, CONSTRAINED)
         ) ON COLUMNS
        FROM [Serenga2])
WHERE ( [Case].[CaseType].&[Order],
        [Activity].[Activity Type].&[Order in behandeling nemen]
      )
```



Detail overzicht Order Intake Desk (SSAS)

aantal orders per accountmanager
met activiteit "Order in behandeling nemen"

VSERVER\vkool Aantal orderdossiers: 85 Pagina 1 van 1

Periode: t/m AccountManager:

<< Terug naar overzicht

Accountmanager	Business Unit	Kenmerk	Titel	Aangemaakt	Behandelaar	Geplande datum	Afhandeldatum	Afhandelsoort	Doorlooptijd
[redacted]		DDS2012004681	Order: 10-2-2012 - [redacted]	Friday, February 10 2012	[redacted]	Friday, February 17 2012	Tuesday, February 14 2012	Afgehandeld. toegekend	3
[redacted]		DDS2012011648	Order: 10-4-2012 - [redacted]	Saturday, April 14 2012	[redacted]	Saturday, April 21 2012	Thursday, April 19 2012	Afgehandeld. toegekend	4
[redacted]		DDS2012011663	Order: 10-4-2012 - [redacted]	Tuesday, April 17 2012	[redacted]	Tuesday, April 24 2012	Wednesday, April 18 2012	Afgehandeld. toegekend	2
[redacted]		DDS2012011663	Order: 10-4-2012 - [redacted]	Tuesday, April 10 2012	[redacted]	Tuesday, April 17 2012	Thursday, April 12 2012	Afgehandeld. afgewezen	3
[redacted]	ma oost	DDS2012015677	Order: 11-5-2012 - [redacted]	Wednesday, May 30 2012	[redacted]	Wednesday, June 06 2012	Tuesday, June 05 2012	Afgehandeld. toegekend	5
[redacted]		DDS2011045652	Order: 12-12-2011 - [redacted]	Tuesday, December 13 2011	[redacted]	Wednesday, December 14 2011	Thursday, December 15 2011	Afgehandeld. afgewezen	3
[redacted]		DDS2012008129	Order: 12-3-2012 - [redacted]	Friday, March 16 2012	[redacted]	Friday, March 23 2012	Tuesday, March 20 2012	Afgehandeld. toegekend	3
[redacted]		DDS2012008324	Order: 12-3-2012 - [redacted]	Friday, March 16 2012	[redacted]	Friday, March 23 2012	Tuesday, March 20 2012	Afgehandeld. toegekend	3
[redacted]		DDS2012012171	Order: 12-4-2012 - [redacted]	Friday, April 20 2012	[redacted]	Friday, April 27 2012	Wednesday, April 25 2012	Afgehandeld. afgewezen	4
[redacted]		DDS2012012171	Order: 12-4-2012 - [redacted]	Wednesday, May 02 2012	[redacted]	Wednesday, May 09 2012	Tuesday, May 08 2012	Afgehandeld. toegekend	5
[redacted]		DDS2012001364	Order: 13-1-2012 - [redacted]	Friday, January 13 2012	[redacted]	Saturday, January 14 2012	Tuesday, January 17 2012	Afgehandeld. toegekend	3
[redacted]		DDS2011045913	Order: 13-12-2011 - [redacted]	Wednesday, December 14 2011	[redacted]	Monday, December 26 2011	Wednesday, December 21 2011	Afgehandeld. afgewezen	6
[redacted]		DDS2011045913	Order: 13-12-2011 - [redacted]	Sunday, January 08 2012	[redacted]	Monday, January 09 2012	Wednesday, January 11 2012	Afgehandeld. toegekend	3
[redacted]		DDS2012004914	Order: 13-2-2012 - [redacted]	Friday, February 17 2012	[redacted]	Friday, February 24 2012	Wednesday, February 22 2012	Afgehandeld. afgewezen	4
[redacted]		DDS2012004914	Order: 13-2-2012 - [redacted]	Thursday, February 23 2012	[redacted]	Friday, February 24 2012	Friday, February 24 2012	Afgehandeld. afgewezen	2
[redacted]		DDS2012004914	Order: 13-2-2012 - [redacted]	Friday, March 16 2012	[redacted]	Friday, March 23 2012	Wednesday, March 21 2012	Afgehandeld. toegekend	4
[redacted]		DDS2012004927	Order: 13-2-2012 - [redacted]	Wednesday, February 15 2012	[redacted]	Not finished	Friday, February 17 2012	Afgehandeld. toegekend	3
[redacted]		DDS2012004929	Order: 13-2-2012 - [redacted]	Friday, March 16 2012	[redacted]	Friday, March 23 2012	Tuesday, March 20 2012	Afgehandeld. toegekend	3
[redacted]	ma oost	DDS2012021121	Order: 14-6-2012 - [redacted]	Thursday, June 14 2012	[redacted]	Sunday, June 17 2012	Not finished		
[redacted]		DDS2012005357	Order: 15-2-2012 - [redacted]	Friday, March 30 2012	[redacted]	Friday, April 06 2012	Friday, March 30 2012	Afgehandeld. toegekend	1
[redacted]		DDS2012005357	Order: 15-2-2012 - [redacted]	Wednesday, March 28 2012	[redacted]	Wednesday, April 04 2012	Wednesday, March 28 2012	Afgehandeld. afgewezen	1
[redacted]		DDS2012008652	Order: 15-3-2012 - [redacted]	Tuesday, April 10 2012	[redacted]	Tuesday, April 17 2012	Tuesday, April 17 2012	Afgehandeld. afgewezen	6
[redacted]		DDS2012008652	Order: 15-3-2012 - [redacted]	Tuesday, May 08 2012	[redacted]	Wednesday, May 09 2012	Friday, May 11 2012	Afgehandeld. toegekend	4

Figure D.2: Detail overview Order Intake Desk report based on the OLAP cube