A Decision Support Tool for Allocating SAP Application Data

Master’s Thesis

Thijs Zandvliet
A Decision Support Tool for Allocating SAP Application Data

THESIS

submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

in

COMPUTER SCIENCE
TRACK INFORMATION ARCHITECTURE

by

Thijs Zandvliet
born in Leiderdorp

Web Information Systems Group
Department of Software Technology
Faculty EEMCS, Delft University of Technology
Delft, the Netherlands
http://eemcs.tudelft.nl

Capgemini
SAP Solutions
Papendorpseweg 100
Utrecht, the Netherlands
http://www.nl.capgemini.com
A Decision Support Tool for Allocating SAP Application Data

Author: Thijs Zandvliet
Student id: 1245198
Email: m.r.zandvliet@student.tudelft.nl

Abstract

Large software corporations like SAP AG started offering their services in the cloud. Most companies using a SAP system run these systems on-premise. The step of SAP AG to bring services to the cloud caused companies to think about their systems. At the moment cloud computing is very popular because of the low costs and the flexibility. For companies running their SAP systems on-premise it is interesting to find out what this same system will cost for them in the cloud. We developed a decision support tool which can help companies performing this step. The tool enables the user to provide the costs involved on-premise and helps them completing the costs for in the cloud. By using a visual presentation of the SAP applications the user is able to drag its applications between the on-premise location and the cloud. The tool provides the costs involved for both storage locations to the user and shows a clear impact of moving an application. We show that we have made a user-friendly interface and a generic structure for such a decision support tool which can be extended and adjusted to the user’s needs.

Thesis Committee:

Chair: Prof. dr. ir. G.J.P.M. Houben, Faculty EEMCS, TUDelft
University supervisor: Dr. ir. A.J.H. Hidders, Faculty EEMCS, TUDelft
Company supervisor: L.H. Steenbergen, SAP Solutions, Capgemini
Committee Member: Dr. ir. M.F.W.H.A. Janssen, Faculty TPM, TUDelft
Preface

This master thesis is my final piece of work for my study at the Technical University of Delft in order to retrieve my master’s degree in Computer Science. I performed my master thesis at Capgemini in the SAP Solutions department.

From Capgemini I want to thank Femke Hoekstra for introducing me at Capgemini and for her support, Manja Kerstholt for the daily supervision and Lando Steenbergen for his knowledge and his referral to the experts. Of course I am also very grateful to the experts from whom I gained very valuable knowledge.

I want to thank the TU Delft for the great studies they supply. Thanks to the TU Delft I achieved lots of very valuable knowledge during the past years, knowledge which I certainly needed during the performance of my master thesis. Therefore I am thankful to all the teachers who provided me this knowledge.

Further, I want to thank the Web Information Systems group headed by Prof. Geert-Jan Houben for giving me the opportunity to be part of the group during my master thesis. I am grateful to Jan Hidders for his, guidance, supervision and input. I also want to thank Prof. Marijn Janssen for his valuable input in the last stage of my master thesis and for being part of my thesis committee.

Finally I want to thank my parents for their faith in me and my friends and girlfriend for listening to me, even though they had no idea what I was jabbering about.

I hope you enjoy reading my work.

Thijs Zandvliet
Leiden, the Netherlands
April 26, 2013
## Contents

Preface ........................................................................ iii

Contents ..................................................................... v

List of Figures ............................................................. ix

1 Introduction ................................................................ 1
  1.1 Background .......................................................... 1
    1.1.1 About Capgemini ........................................... 1
    1.1.2 About SAP .................................................... 2
    1.1.3 About SAP Business ByDesign ....................... 2
  1.2 Problem description ............................................... 2
  1.3 Project goal .......................................................... 3
  1.4 Approach ............................................................. 3
  1.5 Outline ............................................................... 4

2 Background .............................................................. 5
  2.1 SAP landscape ...................................................... 5
    2.1.1 System architecture ....................................... 5
    2.1.2 SAP landscape architecture ......................... 6
    2.1.3 SAP applications .......................................... 7
    2.1.4 Types of enterprise data ............................... 7
  2.2 Data Storage ......................................................... 8
    2.2.1 Traditional storage ....................................... 8
    2.2.2 Cloud computing .......................................... 9
    2.2.3 Data migration ............................................. 13
    2.2.4 Total Costs of Ownership ............................ 13
  2.3 Cloud providers ................................................... 16
    2.3.1 Amazon Web Services (AWS) ....................... 16
    2.3.2 Google Cloud ............................................. 17
    2.3.3 Windows Azure ........................................... 17

3 Architecture and Design ............................................ 19
  3.1 Requirements ....................................................... 19
    3.1.1 Functional Requirements ............................. 19
    3.1.2 Non-functional Requirements ....................... 20
    3.1.3 Design decisions ......................................... 20
  3.2 Configuration design ............................................. 21
    3.2.1 Architecture design .................................... 21
# Functional Design and Implementation

4.1 The first mockup
   4.1.1 XML serializer

4.2 The configuration tab
   4.2.1 The PropertyGrid
   4.2.2 Local instances
   4.2.3 Cloud instances
   4.2.4 SAP applications

4.3 The distribution tab
   4.3.1 The distribution area
   4.3.2 Calculating the costs
   4.3.3 Showing the costs
   4.3.4 Interacting with graphical objects

4.4 Menu options
   4.4.1 Export function

4.5 Summary

# Detailed Description

5.1 Configuration options
   5.1.1 Global options
   5.1.2 Cloud options
   5.1.3 Data properties

5.2 Costs functions
   5.2.1 Traditional co-location costs
   5.2.2 Private cloud costs
   5.2.3 Public cloud costs

5.3 Possible configurations
   5.3.1 Local configurations
   5.3.2 Cloud configurations
   5.3.3 Converting configurations
   5.3.4 Defining the SAP application location

5.4 Summary

# Evaluation

6.1 The first iteration

6.2 User tests
   6.2.1 User test 1
   6.2.2 User test 2
   6.2.3 User test 3

6.3 Results

6.4 Experts
## Contents

7 Conclusions and Future Work  
7.1 Conclusions .......................... 69  
7.2 Future work .......................... 71  

Bibliography .......................... 73  

A Variables ............................ 77  

B Costs functions ...................... 83
List of Figures

2.1 Three-tier architecture example, source: SAP (2008) ........................................ 6
2.2 A typical landscape and promote to production process, source: SAP (2008) ........ 6
2.3 What are your plans to use cloud storage? source: Gartner (March 2012) .......... 12

3.1 Part of the high level object diagram for the configuration ............................... 21
3.2 Left, the object diagram for retrieving prices from the cloud. Right, the parser to use
the prices for the cloud providers ........................................................................ 23
3.3 A sequence diagram for scraping and using data from the websites of the cloud providers
3.4 Part of the high level object diagram for the distribution .................................. 24
3.5 A sequence diagram for adding objects to the distribution ............................... 25
3.6 The configuration tab ......................................................................................... 27
3.7 Instance dialogs ................................................................................................. 28
3.8 The formula editor ............................................................................................ 29
3.9 The distribution tab ......................................................................................... 31

4.1 A first mockup for the tool .............................................................................. 33
Abbreviations

APPO  Advanced Planning and Optimization
AWS  Amazon Web Services
BW  Business Warehouse
CRM  Customer Relationship Management
DEV  Development System
EBS  Elastic Block Store
EC2  Elastic Compute Cloud
ECU  Elastic Compute Unit
ERP  Enterprise Resource Planning
GCE  Google Compute Engine
GRS  Geo Redundant Storage
GUI  Graphical User Interface
HSM  Hierarchical Storage Management
HTML  HyperText Markup Language
IaaS  Infrastructure as a Service
JSON  Javascript Object Notation
MDM  Master Data Management
PaaS  Platform as a Service
PRD  Production System
QAS  Quality Assurance System
S3  Simple Storage Service
SaaS  Software as a Service
SAP  Systems, Applications and Products in Data Processing
SAPS  SAP Application Performance Standard
SBX  Sandbox
SLA  Service-Level Agreement
SME  Small and Medium Enterprises
SRM  Supplier Relationship Management
URL  Uniform Resource Locator
VPC  Virtual Private Cloud
WAVN  Windows Azure Virtual Network
WPF  Windows Presentation Foundation
XAML  Extensible Application Markup Language
XML  Extensible Markup Language
Chapter 1

Introduction

Companies are constantly trying to find ways to reduce costs. A part of these costs consists of computer usage and data storage costs. In the earlier days larger companies had their own datacenters. Later on many of these companies replaced their own datacenters with traditional co-locations managed by outsourcers. Both solutions experienced the problem that there was a lot of unused capacity. A solution for this problem is found in cloud computing, where users only pay for the resources they use.

Cloud computing has increased in popularity over the past few years. Many companies recently made the choice to move parts of their data into the cloud. Companies can save a significant amount of money making use of the cloud instead of own facilities or traditional co-locations. Other companies are interacting with the market by delivering software services in the cloud.

One of the companies providing widely used software for multinationals is SAP AG. Their software is used on a daily basis and can involve large amounts of data. To keep up with the market SAP is providing cloud solutions which can replace (or be used in combination with) their current software. A software solution running only in the cloud is SAP Business ByDesign. Due to the introduction of this software solution and other similar solutions it is important for companies to know where to store their data. When storing data in a "local" (private cloud and traditional co-location) and public cloud environment it is important that no redundant data has to be managed twice.

1.1 Background

First we will provide some information about Capgemini and SAP. Capgemini is the company where the master thesis was performed and SAP the company which produces the Enterprise Resource Planning system, which is the basis of the decision support tool.

1.1.1 About Capgemini

Capgemini provides IT services and is one of the world’s largest consulting, outsourcing and professional services companies. These services focus primarily on system architecture, -integration and -infrastructure. Capgemini is working with strategic partners to solve the issues and challenges of customers with all the necessary expertise.

The master thesis is performed inside the SAP Solutions division. Capgemini invests proactive in the development of new services and together with SAP Capgemini is a developer of new products and methods. Capgemini is a SAP-partner for fifteen years now and has executed more than 3000 SAP-projects for 1500 customers worldwide.
1.1.2 About SAP

SAP AG is the market leader in enterprise application software. Founded in 1972, SAP has a rich history of innovation and growth as a true industry leader. SAP applications and services enable more than 183,000 customers in over 120 countries worldwide to operate profitably, adapt continuously and grow sustainable. SAP has more than 54,000 employees and sales and development locations in more than 50 countries worldwide. The abbreviation SAP stands for "Systems, Applications and Products in Data Processing".

1.1.3 About SAP Business ByDesign

The current objective of SAP is to provide solutions for companies of all sizes. The newest solution is SAP Business ByDesign which was introduced on September 19, 2007. Business ByDesign is a one-size fits all, subscription based ERP system aimed at mid-market companies and was released as the first SAP ERP SaaS product. After several troubled installations and flawed go to market strategy, SAP pulled ByDesign from the market for a system revamp and significant code refactoring.

Three years later Business ByDesign reemerged as a multi-tenant Software as a Service (SaaS) solution, complete with a new architecture. The presentation layer is built with Silverlight and PaaS tools are used for extensibility. In December 2011 the solution has acquired its first 1000 customers and is now available in Australia, Austria, Canada, China, Denmark, France, Germany, India, Italy, the Netherlands, Spain, Switzerland, the United Kingdom, and the United States.

SAP Business ByDesign is targeting SMEs with more than 25 users but can also be used by organizations with as few as 10 users. The best range is 25-500 users.

1.2 Problem description

The expectation is that organizations are going to use SAP Business ByDesign in combination with their current SAP ERP system in order to add cloud functionality to their business. Business ByDesign however has its own storage in the cloud. Once the two systems are used together the redundancy will be very high which could have a serious effect on the data quality. The same data needs to be inserted and managed twice which increases the possibility on faults and results in higher costs for labor. When storing the data twice, in the cloud and local, the costs for storage are much higher.

A company always wants to reduce the amount of costs involved with the storage of the data. These costs can be measured in the total costs of ownership of data per month. It is important to come up with a solution to reduce these costs and have an efficient allocation of data.

There are some constraints involved in finding solutions for the problems mentioned. These problems involve for instance the Patriot Act of America which provides America the "right" to have insight in information stored in America. The cloud stores information all over the world so also in America. Therefore a company will not be eager to store sensitive information in the cloud. This and other political and business related problems have to be taken into account.

---

1.3 Project goal

The goal is to come up with a decision support tool which presents a user-friendly environment which makes it possible to view the impact on the total monthly costs when allocating data between cloud storage and local storage. Therefore the following questions have to be answered:

- What configurations are possible?
- What are the possible cost functions and configuration options?
- What are interesting rules of thumb to decide where to store the data?
- How to make the tool in such a way that it is generic, so that it can be used by several companies with minor changes?
- How to design a user friendly interface which can easily be used by the target group?

The focus for the tool will be based on SAP Application data, although it will be possible to use the tool with other systems after some minor modifications.

The tool will contain many configuration options which are necessary to calculate the total costs involved. Once all the necessary variables are provided a data distribution is shown with a pre-configuration based on the user’s constraints and the minimal costs. The distribution screen exists of three areas. On the left side a local storage area, in the middle a neutral area where applications can be "parked" and on the right side the cloud storage area. The application data can be dragged between areas and the impact of these actions are visible in the costs presented on the bottom of each area.

It is important that configuration managers can work easily with the tool and do not have to provide the same kinds of data over and over again. The tool must be clear and help the user by providing the necessary information where possible. The user must have the possibility to play around with the data in order to see the impact of the actions.

1.4 Approach

To start the research it is important to get to know something about SAP ERP and SAP Business ByDesign. Capgemini provides some information and they organize a training which will take two weeks. To create the tool a lot of research will be done in cloud-based systems and local systems. Beyond this it is possible to get feedback from experts at Capgemini.

The approach is to start designing the tool and come up with a first mockup. In order to do this several cost elements have to be taken into account. The next step would be to create a basic design of the tool and make it possible to insert values, navigate around the tool and generate a visual presentation without linking this to actual data. Once this is done the tool will be shown to experts at Capgemini in order to get some feedback. The expectation is that this will result in some valuable feedback about the properties and the presentation of the data in order to meet the wishes and expectations of the experts as much as possible. These experts will be people working with data integration at the SAP Solutions department and other experts involved with management of data. Once there is some attention from the experts the hope is to get some more feedback from them during the development of the tool. This can also result in using them for the testing phase.

Once the different elements of the tool are finished it is possible to start with processing the input data. There are several formulas used for the costs of traditional co-location data and several
for cloud data, dependent of the cloud provider. With these formulas and the input variables the first calculations will be made. As this is done the impact of moving data in the distribution tab will be possible to see.

Once the tool is finished several experts from within Capgemini will be asked to use and test it so they become satisfied with the results.

1.5 Outline

The thesis starts with a literature review in chapter 2 with information about the SAP landscape in section 2.1, this involves the system architecture and information about SAP applications. To get some insight in data storage we presented some basic information about traditional storage and cloud computing in section 2.2 of the literature review, this also includes data migration and the storage costs. The last section of the literature review is about the different cloud providers used during the master thesis, in section 2.3.

In chapter 3 we introduce the structure of the decision support tool and we provide information about the architecture and the design decisions. Section 3.4 of this chapter is about the Graphical User Interface. Chapter 4 is about the functional design and the implementation of the decision support tool. What toolkits we used and how we managed to solve the technical challenges behind the tool. A more detailed description of the decision support tool is provided in chapter 5. This chapter is about the several configuration options and the used cost functions. In chapter 6 the usability of the tool is tested by using user- and cognitive walkthrough tests.

The final chapter, chapter 7, contains several concluding remarks and future work.
Chapter 2

Background

In order to understand something about the purpose of the tool it is important to learn something about how the SAP ERP software is structured and what parts are covered by the tool. Therefore this chapter will start with a small section about the SAP system architecture followed by information about the SAP landscape architecture and some information about the SAP applications.

2.1 SAP landscape

The SAP system landscape is the arrangement of a companies SAP servers, which is sometimes even called an architecture of servers. SAP is divided into several different landscapes. Examples are development, quality assurance and production. Information about this system architecture and the different landscapes is available in the next sections.

2.1.1 System architecture

The SAP ERP software is an integrated suite of financial, manufacturing, distribution, logistics, quality control and human resources application systems (Bancroft, 1996). The architecture of SAP ERP is based on a three-tier client/server architecture, shown in figure 2.1:

- Presentation layer
- Business logic / Application layer
- Database layer

The presentation layer consists of the Graphical User Interface which is the direct link between the user and the SAP ERP system. This can be a client application installed on a computer or a web interface as is used for Business ByDesign. The presentation layer does not have to be installed on the same server as the SAP applications. Its function is to operate as a front-end to the applications running on the application layer.

The application layer provides the application logic, a runtime, system management and operation tools, development and change management environments and serves as an abstraction layer for the database and operating system.

The database layer is where all the data is stored. Because of performance and security reasons the database is kept on a separate server.

When using a SAP system commands are executed by using the presentation layer. The application layer does the processing and communicates with the database to retrieve or manipulate
2.1 SAP landscape Background

Figure 2.1: Three-tier architecture example, source: SAP (2008)

Once the processing in the application layer is done the results are sent back to the presentation layer.

In this master thesis the focus is on the application layer and the database layer. The clients in the presentation layer can be installed on almost any computer, the application and the database layer require more specific systems.

2.1.2 SAP landscape architecture

A very common landscape architecture used with SAP systems consists of a Development System (DEV), a Quality Assurance System (QAS) and a Production System (PRD). This is typically called a Three System Landscape (SAP, 2008).

Next to the three system landscape many customers add a fourth environment, a standalone sandbox (SBX) environment used for destructive testing, learning, and testing. The sandbox is not part of the promote to production landscape, therefore it is still called a three system landscape.

Figure 2.2: A typical landscape and promote to production process, source: SAP (2008)
In the *Development System* all the customizing, system maintenance and development work is performed. After all the changes have been unit tested the changes can be transferred to the *Quality Assurance System* for further system testing.

In the *Quality Assurance System* the configuration, development or changes undergo further tests and checks to ensure that they do not adversely affect other modules.

The *Production System* is used by a company for its live, productive work. On this system the real business processes are executed. The quality of the DEV and QAS system and the implemented change management processes impact the quality of the production system directly (SAP, 2008).

### 2.1.3 SAP applications

The SAP ERP software is an integrated software solution. The possibility exists to add extra functionalities to the ERP system by adding applications. Examples of these applications are Supplier Relationship Management (SRM), Customer Relationship Management (CRM), Advanced Planning and Optimization (APO), Business Warehouse (BW). All these applications are located in the application layer as described in section 2.1.1.

By using *Server virtualization* it is possible to reduce the number of individual servers utilized within a landscape and having multiple systems consolidated and installed on a single large server. Each of the systems is viewed as an independant system, each with its own database. In most cases the operating system is common to all systems on the server. This reduces the amount of effort needed to maintain the individual server hosts (SAP, 2008).

The technical foundation for many SAP applications is SAP NetWeaver. This is SAP’s integrated technology computing platform and is a service-oriented application and integration platform. SAP NetWeaver provides the development and runtime environment for SAP Applications and can be used for custom development and integration with other applications and systems\(^1\). One of the applications built on SAP NetWeaver is Master Data Management (MDM) which provides the possibility to consolidate, cleanse and synchronize a single version for master data within a heterogeneous application landscape\(^2\). Due to the use of several applications with their own database lots of data is stored multiple times, the MDM application can be used to manage this data\(^3\). Many of the information stored in a SAP system is master data.

### 2.1.4 Types of enterprise data

Inside the SAP software five varieties of physical data are stored. These varieties of data are characterized by their data types and their purpose within the company (Wolter and Haselden, 2006).

**Unstructured**

Unstructured data is data found in e-mail, white papers, magazine articles, corporate intranet portals, product specifications, marketing collateral and PDF files.

---

\(^1\) [http://en.wikipedia.org/wiki/SAP_NetWeaver](http://en.wikipedia.org/wiki/SAP_NetWeaver)

\(^2\) [http://en.wikipedia.org/wiki/Master_data_management](http://en.wikipedia.org/wiki/Master_data_management)

2.2 Data Storage

Transactional data

Transactional data supports the on-going operation of an organization. This can include areas such as sales, service, order management, manufacturing, purchasing, billing, accounts receivable and accounts payable. Transactional data commonly refers to the data that is created and updated within the operational system. Examples of transactional data are orders, invoices and payments in finance (Otto and Reichert, 2010).

Meta data

Meta data is data about other data and may reside in a formal repository or in various other forms such as XML documents, report definitions, column descriptions in a database, log files, connections and configuration files.

Hierarchical data

The relationship between other data is stored in hierarchical data. This data may be stored as part of an accounting system or separately as descriptions of real-world relationships, such as company organizational structures or product lines. Hierarchical data is sometimes considered a super MDM domain, because it is critical to understanding and sometimes discovering the relationships between master data.

Master data

Master data is the consistent and uniform set of identifiers and extended attributes that describe the core entities of the enterprise, and are used across multiple business processes. These core entities are for instance parties (customers, prospects, people, citizens, employees, vendors, suppliers or trading partners), places (locations, offices, regional alignments or geographies) and things (accounts, assets policies, products or services). Groupings of master data include organizational hierarchies, sales territories, product roll-ups, pricing lists, customer segmentations, preferred suppliers etc. Master data is not all the data, only the subset or finite list of elements required for sharing and standardization. Master data is not changed very often and is often referenced by a business process or event. (White et al., 2006).

According to a research done by Otto and Reichert (2010) the main focus of organizations is on customer master data (84%), followed by material/product master data (68%), and supplier vendor data (63%). Less than 27% of the organizations have their main focus on the management of master data related to human resources.

2.2 Data Storage

In section 2.1 the several layers of the SAP system architecture were explained. This section focuses on information about the storage methods for the application layer and the database layer. First the advantages and disadvantages of traditional storage and cloud computing are explained. Then there is a part about data migration and this section ends with some information about the total costs of ownership of data.

2.2.1 Traditional storage

Traditional approaches for storage are storage on location (on-site), where the company owns its own datacenter and manages its own systems. The costs for preparing such a datacenter are very high, taking into account that the company needs to take care of issues like costs for hardware, power, cooling, network, floor space, fail over facilities etc. Furthermore there are costs like, data
protection requirements, annual growth requirements, costs of disaster recovery, percentage of usable capacity etc. Once such a datacenter is operational, data can be reliably stored (Kozhipurath, 2012). In the case of a SAP ERP system this means that data from several applications is stored on multiple servers. The problem rises that data cannot be stored anymore when there is no disk space left on a specific server, in that case a larger hard disk must be installed and all the data has to be transferred to the new disk. This causes downtime and multiple hours of labor.

Many outsourcers make use of Hierarchical Storage Management (HSM) which is a data storage technique that moves data between high-cost and low-cost storage media. This HSM technique is necessary to reduce the costs involved with data storage. Data stored on hard disk drive arrays is more expensive than storage on slower devices like optical discs and magnetic tape drives. These different kinds of storage can be defined in tiers (Goda and Kitsuregawa, 2012). In a SAP production system data is mostly stored on tier 1 storage, in this case this could be a 10k rpm hard disk which is a very fast hard disk. In a non-production system data is often stored on tier 2 storage, this could be a 7200 rpm hard disk. If data is almost not used or when backups are needed, the data is stored on lower tiers. The higher the number of the tier the cheaper the media that could be used.

To make optimal use of the space available on the several servers it is possible to use storage virtualization techniques. This technique makes it easier to increase and decrease storage space without having to move data and having expensive downtimes (Singh et al., 2008). This technique is one of the advantages of cloud computing as we will explain in the next section.

2.2.2 Cloud computing

The traditional system configurations are only existing of on-premise systems where the company owns and manages its own data in their own environment. This can be in a setting where the datacenter is located at the company itself but more often it happens that the data storage and maintenance is outsourced. This means that another independent company is contracted to take care of an existing part of the business, in this case the datacenter. When working with third parties which take care of the datacenter it is very important to address legal, security and compliance issues through a contract between the client and the suppliers, this is called a Service-Level Agreement (SLA). There are several reasons to outsource the data storage, the most important one is to reduce the costs, because it is not necessary to build an in-house datacenter. Another reason is that the business is more able to focus on their core business. Other reasons are access to more knowledge, talent and experience, and increased profits (Girma and Gorg, 2004).

Basic idea

The definition of Mell and Grance (2009): “Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction”. There are different types of clouds. A Public Cloud which is made available in a pay-as-you-go manner to the general public; The service being sold is Utility Computing. There is also the possibility to have internal data-centers of a business or other organization which is not available to the general public, this is called a Private Cloud. When the cloud is provisioned for exclusive use by a specific community of consumers from organizations that have shared concerns this is called a Community cloud. The last type of cloud is the Hybrid cloud which is a composition of two or more distinct cloud infrastructures that remain unique entities (Joha and Janssen, 2012). Cloud Computing is

\(^4\)http://en.wikipedia.org/wiki/Hierarchical_storage_management

\(^5\)http://en.wikipedia.org/wiki/Storage_virtualization

\(^6\)http://en.wikipedia.org/wiki/Outsourcing
2.2 Data Storage

the combination of SaaS and Utility Computing. Private Clouds not included (Armbrust et al., 2009). There are three new aspects in Cloud Computing:

1. For the cloud-user the cloud exists out of unlimited resources, although this is not entirely true there is no need to worry about resources for the cloud-user.

2. Cloud Companies do not have to have lots of resources in the beginning. They can start small and increase hardware resources when there is an increase in needs.

3. Payments for computing resources can be done on a short-term basis. Think about processors by the hour and storage by the day. It is possible to release these resources when they are no longer needed.

There are different service models. Software as a Service (SaaS) that allows the consumer to use software running on a cloud infrastructure and which is maintained and delivered by the provider. Platform as a Service (PaaS) that allows the consumer to deploy software onto the cloud infrastructure. The consumer has control over the deployed applications and possible configuration settings for the application-hosting environment. Infrastructure as a Service (IaaS) provides the consumer the capability to provision processing, storage, networks and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications.

Private cloud

More and more companies use the cloud to store their data. In the case of outsourcers the data is moved to a private cloud. Private clouds can also be located on-premise and are operated solely within a single organization, and managed by the the organization or a third party, the outsourcer. Private clouds have their own firewall. The reason for companies to move to a private cloud is to maximize and optimize the utilization of existing in-house resources. Other reasons are security concerns including data privacy and trust, data transfer costs which are lower than in a public cloud and the possibility of full control over the data behind their firewalls (Dillon et al., 2010).

Public cloud

The public cloud exists of a pool of computing resources offered by some vendor that supplies software. The public cloud is used by the general public cloud consumers and the cloud service provider has the full ownership of the public cloud with its own policy, value, and profit, costing, and charging model. Examples of public clouds are Amazon Web Services (AWS)\textsuperscript{7}, Google Compute Engine (GCE)\textsuperscript{8}, Windows Azure\textsuperscript{9} and Force.com\textsuperscript{10} (Dillon et al., 2010).

The use of the cloud has several advantages and disadvantages. In order to decide if the cloud is suitable for a companies purpose it is important to know what these advantages and disadvantages are. Therefore we came up with a list of the most common characteristics of the cloud.

Advantages of the cloud

(Grossman, 2009)(Leavitt, 2009)

- Maximize and optimize the utilization of existing resources: By using cloud software it is possible to use all the resources available on existing resources, without having to worry on the amount of space available per web server.

\textsuperscript{7}http://aws.amazon.com/
\textsuperscript{8}https://cloud.google.com/products/compute-engine
\textsuperscript{9}http://www.windowsazure.com/en-us/
\textsuperscript{10}http://www.force.com/
• Costs are lower: The cloud providers are responsible for the cloud and they manage the resources. They only focus on this task and are offering a load of available storage space with many web servers in several data centers, this way they can reduce the costs involved. Another reason the costs for the providers are lower is because they can use lower cost and energy-saving PC’s (Qian et al., 2009). Therefore it is much cheaper for companies to pay for storage at a cloud provider instead of building and managing their own data center.

• Scalability: The cloud provider adds web servers to the resources when necessary so for the cloud user it seems that there are unlimited resources. The cloud user does not have to worry about the available storage. When there is a moment the company needs less storage they do not get stuck with unused storage.

• Pay per use: Cloud users only pay for the storage they use and the amount of data they transfer over the network, in most of the cases the cloud users only have to pay for the data going out of the cloud servers, and not in.

• Data restore: Data stored in the cloud is stored in several pieces over several disks. Once one of the pieces is lost the complete data can be restored with the rest of the data. Most cloud providers store their data redundant.

• Effortless upgrades: The company does not have to worry about upgrades done to servers, this is all taken care off by the cloud provider.

• Minimized end-user training: SaaS applications are highly standardized so easier to work with once end-users are familiar with the applications (Janssen and Joha, 2011).

• Less administration tasks: Lots of the administration related to the cloud is done by the cloud provider.

• Lower licensing costs: Customers do not need their own licenses for all the in-cloud software they use.

Disadvantages of the cloud

(Armbrust et al., 2009)(Subani, 2009)

• Dependency: As mentioned earlier, the provider has the full ownership of the public cloud and decides what the rules are, the SLA. This is not a safe feeling when your company stores all kinds of sensitive data into the cloud. If there is a technical error at the Cloud Provider Company the company is dependent on their services. A solution for these issues is to use a private cloud, drawback of this solution is that you lose part of the elasticity of the public cloud.

• Internet speed: Data which is used very often can better be as near as possible where it is needed. It can take a while before large amounts of important data are downloaded over the Internet. This can slow down the process and cause extra time and therefore money. Employees are expensive and each minute they spend on waiting for data to be processed is a minute less they can spent on more valuable tasks.

• Data accessible after computer shutdown: Once the workday is over and all the employees are at home, the data is still publicly accessible in the public cloud. This means that hackers can try to access the private data even after working hours (Janssen and Joha, 2011).

• Provider can access info: Major ISPs have come under fire because of spying on their customers on behalf of the Recording Industry. Another reason for providers to infringe your
data privacy is the American Patriot Act\textsuperscript{11}. Most major Cloud Computing Servers are operated by companies based in the United States. And even if that is not the case it is possible that the data goes through American ISPs, that provide the cloud with uptime. Data could be intercepted before it reaches the cloud, all due to the American law.

- **No Internet, no data:** Once the Internet is not accessible for several possible reasons, it is also not available to access the company’s data. A company can survive several minutes but there are many companies who can not survive a downtime of several days. However, when companies outsourced their data, they have to deal with the same problem.

According to Gray (2008) the computation has to be put near the data because the transfer costs are to high to bring all the data to the computation. This means that transactional applications such as ERP/CRM may not be suitable for cloud computing if the cost-savings do not offset the extra data transfer costs (Dillon et al., 2010). However, the transfer costs in the cloud become lower each time which means that even ERP/CRM systems can be stored in the cloud, from an economic perspective.

There are lots of disadvantages related to the cloud but most of these also apply to outsourced data where a third party has to be reached over an Internet connection. According to a survey held by IDC in 2011 the number one initiative to take in 2012 is invest in cloud services.

**Hybrid cloud**

Most companies prefer to keep their core information in-house because they just do not trust cloud providers enough to store their core data into the public cloud according to research done by Gartner (Ruth, 2012), see figure 2.3.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure2_3.png}
\caption{What are your plans to use cloud storage? source: Gartner (March 2012)}
\end{figure}

For this reason a better solution is to keep sensitive data on-premise and store less sensitive data in the public cloud. This is possible when using a hybrid cloud solution. The local data is stored in a private cloud, so the advantages of the private cloud can still be used, and the rest of the data can be stored in a public cloud. Amazon Elastic Compute Cloud (EC2)\textsuperscript{12}, Google App

\textsuperscript{11}http://en.wikipedia.org/wiki/Patriot_Act
\textsuperscript{12}http://aws.amazon.com/ec2/
Engine\textsuperscript{13} and Windows Azure\textsuperscript{14} all support hybrid cloud solutions by providing the users a Virtual Private Cloud (VPC). By using such a VPC it is possible to use the resources and advantages of the public cloud and the data is secure. The connection between the corporate datacenter and the VPC can be established over a Virtual Private Network (VPN). Another solution would be to create an own private cloud, or use a private cloud of an outsourcer and link that private cloud to a public cloud. Even if the trust towards cloud providers increases, there would still be data which a company would never publish in the public cloud because of regulations or because the data is to sensitive like financial data, therefore a hybrid cloud is a good solution.

2.2.3 Data migration

One of the biggest challenges related to the data distribution is to make decisions about what data has to be stored where. What data has to be migrated? Organizations are conservative in employing IaaS compared to SaaS. This is partly because marginal functions are often brought to the cloud, and core activities are kept in-house. According to a survey conducted by IDC in 2008, 31.5\% of the organizations will move their storage capacity to the public cloud (Dillon et al., 2010). In 2011 however Gartner stated that companies rather implement it as a private environment with only selective data placed in public cloud facilities (Ruth, 2012).

The biggest reason for not moving to the public cloud is because of security issues. 88.5\% of IT companies think this is the biggest challenge/issue according to a survey held by IDC in August 2008. The importance of security is substantiated by other sources, as already mentioned in 2.2.2. An example to tackle the security issue is to split confidential data into pieces and distribute them onto different clouds so that security compromise in one cloud will not lead to disaster as a whole. However, this distribution technique adds extra financial costs and can can cause an impact on the system performance (Dillon et al., 2010).

For using SAP in the cloud, SAP already teamed up with Amazon Web Services (AWS)\textsuperscript{15}. Between SAP and Microsoft, and SAP and Google there are at the moment no real collaborations concerning storing SAP software and related data in their clouds. Although there is no real collaboration it is still possible to use Microsoft Azure, Google Compute Engine or Google App Engine for storing a SAP system\textsuperscript{16} (Seitz, 2010). Given the collaboration between SAP and these two other parties on other areas it would be likely that direct support for SAP systems in their clouds will follow in the future (Williams, 2011)\textsuperscript{17}.

2.2.4 Total Costs of Ownership

To be able to define the minimum costs of distributed data on-premise and in the public cloud different factors have to be taken into account. In case of cloud storage the factors are roughly similar between the different cloud providers. These factors exist of the hours of use of the cloud instances (virtual servers), upfront costs when using reserved instances, the amount of storage, the number of I/O requests, transfer costs and backup storage costs. The cloud providers offer several options to use the cloud. There is the possibility to reserve cloud instances for a specified amount of time and it is possible to use the cloud instances on-demand. The on-demand solution is without obligations for longer periods, so the client only pays what is used and it is possible to stop using the services at any time. The on-demand solutions however are more expensive for over a longer

\textsuperscript{13}https://developers.google.com/secure-data-connector/
\textsuperscript{14}http://www.windowsazure.com/en-us/pricing/details/#header-6
\textsuperscript{15}http://aws.amazon.com.sap/
\textsuperscript{16}http://www.vogella.com/blog/2010/11/22/sap-google-app-engine/
\textsuperscript{17}http://www.liventerprise.com/news/3855/
period, especially when the instance is used for many hours. In that case it is cheaper to use reserved instances in the cloud. Because there is a fixed amount of upfront costs, with lower hourly costs. When using a reserved instance for many hours the total costs involved will be lower. In this thesis we will only focus on the direct costs involved. The indirect costs are out of the scope of this thesis and will require extra research.

Cloud comparison

<table>
<thead>
<tr>
<th></th>
<th>Amazon Web Services</th>
<th>Windows Azure</th>
<th>Google Compute Engine</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hourly instance costs</td>
<td>€278,82</td>
<td>€529,26</td>
<td>€201,07</td>
</tr>
<tr>
<td>Storage costs</td>
<td>€83,00</td>
<td>€70,00</td>
<td>€80,00</td>
</tr>
<tr>
<td>I/O requests</td>
<td>€4,15</td>
<td>€4,00</td>
<td>€4,00</td>
</tr>
<tr>
<td>Data transfer</td>
<td>€45,00</td>
<td>€45,00</td>
<td>€45,00</td>
</tr>
<tr>
<td>Backups</td>
<td>€23,18</td>
<td>€22,54</td>
<td>€32,19</td>
</tr>
<tr>
<td><strong>Per month</strong></td>
<td><strong>€434,16</strong></td>
<td><strong>€670,79</strong></td>
<td><strong>€357,76</strong></td>
</tr>
</tbody>
</table>

Table 2.1: A comparison between the different public cloud providers

<table>
<thead>
<tr>
<th></th>
<th>Amazon Web Services</th>
<th>Windows Azure</th>
<th>Google Compute Engine</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>High Memory Extra Large</td>
<td>Extra Large</td>
<td>High Memory, 2 cores</td>
</tr>
<tr>
<td>Cores</td>
<td>2</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>Memory</td>
<td>17.1 GB</td>
<td>14 GB</td>
<td>13 GB</td>
</tr>
<tr>
<td>Instance storage</td>
<td>420 GB</td>
<td>2040 GB</td>
<td>870 GB</td>
</tr>
</tbody>
</table>

Table 2.2: Specifications of the instance types used for the comparison

In table 2.1 a comparison is made between the different clouds. Assumptions made for the calculations in the example are that the application runs on one instance with an usage of 728 hours a month, a constant storage of 1 TB during that month, 50 million I/O requests, 500 GB data transfer (out), backups consisting of an average amount of 355 GB per month. The instances used per cloud provider are shown in table 2.2. Windows Azure does not have high memory instances, this means that the largest available instance must be used in order to match the memory of the other cloud providers. SAP systems require an instance with high memory. The Windows Azure instance now has 8 cores and 2040 GB of instance storage, which is not necessary. Amazon and Google both provide high memory instances and their instance options are similar although in this example the Google instance has 4.1 GB of memory less. The next step for a Google high memory instance would be 26 GB of memory. The amount of instance storage is not very important. Only the operating system and the application has to be stored on the instance storage, the database is stored elsewhere.

Although the costs are calculated for one instance, the amount of instances can differ per company, just like the other factors which are assumed. Most multi-nationals running SAP do have multiple servers used for different parts of the SAP system. A SAP CRM system for instance often runs on its own server, just like many other SAP applications. This means that the actual total costs are much higher than suggested in the table.
2.2 Data Storage

<table>
<thead>
<tr>
<th></th>
<th>Traditional Co-Location</th>
<th>On-Site</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server hardware</td>
<td>€90,94</td>
<td>€90,94</td>
</tr>
<tr>
<td>Network hardware</td>
<td>€18,22</td>
<td>€18,22</td>
</tr>
<tr>
<td>Hardware maintenance</td>
<td>€37,59</td>
<td>€37,59</td>
</tr>
<tr>
<td>Power and cooling</td>
<td>€48,23</td>
<td></td>
</tr>
<tr>
<td>Data center construction</td>
<td>€75,81</td>
<td></td>
</tr>
<tr>
<td>IT administration and support</td>
<td>€962,00</td>
<td></td>
</tr>
<tr>
<td>Co-Location expense</td>
<td>€143,01</td>
<td></td>
</tr>
<tr>
<td>Remote hands support</td>
<td>€0,96</td>
<td></td>
</tr>
<tr>
<td>Database administration</td>
<td>€155,00</td>
<td>€155,00</td>
</tr>
<tr>
<td>Data transfer</td>
<td>€191,38</td>
<td>€59,84</td>
</tr>
<tr>
<td><strong>Total per month</strong></td>
<td><strong>€637,10</strong></td>
<td><strong>€1149,81</strong></td>
</tr>
</tbody>
</table>

Table 2.3: Similar configuration for a non-cloud solution

Non cloud comparison

At the moment many SAP systems run on traditional co-located systems and there are even companies still running their own systems on-site. Therefore another comparison is made with the same configuration details as the cloud examples. The costs are extracted from the Amazon EC2 cost comparison calculator\(^1\) and explained by Varia and Papo (2012). The server hardware is amortized over a period of 3 years. All the costs are divided into months so they give an impression of the monthly costs.

The results in table 2.1 and 2.3 shows that the total costs for most cloud providers are lower than the costs for traditional co-location, and the most expensive solution is on-site storage. Although it has to be taken into account that the costs are based on an own server including data center facilities for just 1 TB. Especially in case of very large amounts of data transfer, the on-site solution shall be less expensive than traditional co-location. Still the cloud solution is cheaper and therefore purely based on direct costs the better solution. The cloud can be much cheaper when using reserved instances in this example. Google however has no reserved instances available, therefore we used only on-demand instances.

SAPS

In private clouds, running SAP systems, the costs for the instances is often measured in SAPS. SAPS stands for SAP Application Performance Standard and is a hardware-independent unit of measurement that described the performance of a SAP system configuration. SAPS are derived from the Sales and Distribution benchmark where 100 SAPS is defined as 2000 fully business processed order line items per hour. This throughput also corresponds to 6000 processed dialog steps (screen changes), or 2400 SAP transactions\(^2\).

Backups

For local and cloud storage it is important to keep recent backups. Backups protect file systems from user errors, disk or other hardware failures, software errors that may corrupt the file system and natural disasters. Because most systems, also cloud systems, are redundant it is possible that the faults are also redundant. When a backup is made of the entire contents of a file system this is called a full backup. Disadvantages of full backups are that reading and writing of the entire

---

\(^1\) http://awsmedia.s3.amazonaws.com/Amazon_EC2_Cost_Comparison_Calculator.xls
\(^2\) http://www.sap.com/campaigns/benchmark/measuring.epx
\(^2\) http://en.wikipedia.org/wiki/SAP_Application_Performance_Standard
file system is slow and it takes a lot of storage space. Faster and smaller backups can be achieved using an incremental backup scheme. Such a backup only copies those files that have been created or modified since a previous backup. Restoring a deleted file or an entire file system is slower in an incremental backup system, because it can require a chain of backup files. A possibility for on-line backups where files are being accessed during a backup is to create a snapshot. This is a frozen, read-only copy of the current state of the file system (Chervenak et al., 1998).

The costs for backups are dependent on the number of full and incremental backups and the filling of the hard disks. The backup costs in table 2.1 and 2.3 are calculated according to the number of full backups, incremental backups, the number of month backups and the number of year backups. Also important is the retention time in weeks for storing a backup. The total amount is divided by the number of years and months to get the monthly backup costs.

2.3 Cloud providers

We used three cloud providers for the tool. The three cloud providers are chosen on size and publicity. The cloud providers are growing each year and are competing to each other by offering lower prices for their services and by providing more features. In this section we tell something more about the cloud providers and the services they offer, which we use for the tool.

2.3.1 Amazon Web Services (AWS)

Amazon is the oldest among the cloud providers and was launched in July 2002. The idea behind offering cloud services was to make profit on the infrastructure required to run the Amazon.com store. The list of AWS products is quite long, we only focus on the services used for the tool.

Amazon Elastic Compute Cloud (EC2)

The Amazon EC2 service is a web service that provides resizable compute capacity in the cloud. The customer can choose between a wide variety of configurations. Amazon provides on-demand instances where the user only pays for the compute capacity by the hour with no long term commitments. For reserved instances the user makes a one-time payment for each instance and in turn receives a discount on the hourly charge for that instance. Amazon EC2 is an IaaS cloud computing platform\textsuperscript{12}.

Amazon Elastic Block Store (EBS)

These are block level storage volumes which can be used with Amazon EC2 instances. The Amazon EBS volumes are network-attached. The storage volumes can have a storage space between 1 GB and 1 TB and it is possible to mount multiple volumes to the same instance. They are placed in the same Availability Zone as the EC2 instance for fast response times. Each storage volume is automatically replicated within the same Availability Zone for prevention of data loss due to failure of hardware components. Amazon EBS is particularly suited for applications that require a database, filesystem, or access to raw block level storage\textsuperscript{21}.

Amazon Simple Storage Service (S3)

The Amazon S3 service is storage for the Internet. The service has a simple web services interface that can be used to store and retrieve any amount of data, at any time, anywhere on the web. The object size can be between 1 byte and 5 terabyte of data. There is the possibility to store an unlimited amount of objects. The user can choose its own region to store data. The data objects

\textsuperscript{21}http://aws.amazon.com/ebs
will never leave the region where the data object is stored unless the user requests to. Amazon S3 storage is suitable for storing backup data, like snapshots\(^22\).

### 2.3.2 Google Cloud

Google offers two suitable services for cloud applications. Google App Engine and Google Compute Engine. The App Engine service was released as a preview in April 2008 and came out of that preview period in September 2011. Since June 2012 Google also provides the GCE service.

**Google App Engine (GAE)**

Google App Engine is a PaaS cloud computing platform. It is meant for developing and hosting web applications. App Engine offers automatic scaling for web applications. The supported languages are Python, Java, JRuby, Scala, Clojure, Jython, PHP and Go. App Engine is designed in such a way that it can sustain multiple datacenter outages without any downtime\(^23\).

**Google Compute Engine (GCE)**

This IaaS product from Google had some major changes in November 2012. Now it supports much more instances than during its launch. The new instance overview is now comparable to the Amazon EC2 instances. The GCE instances are running on Linux virtual machines\(^8\).

### 2.3.3 Windows Azure

Microsoft offers several services on their Windows Azure platform. The ones suitable for the tool are virtual machines and cloud services. The virtual machines however, are still in a preview stage. Windows Azure can be seen as a "cloud layer" on top of a number of Windows Server systems, which use Windows Server 2008 and a customized version of Hyper-V\(^24\).

**Virtual machines**

The service was first announced in June 2012. It is an IaaS cloud computing platform. The preview version now supports Windows Server 2008 and 2012 RC and a few distributions of Linux. In comparison to AWS and GCE there are only standard instance types available at the moment\(^25\).

**Cloud services**

The Cloud services comprise one aspect of the PaaS offerings from the Windows Azure Platform. The services are containers of hosted applications. These can be internet-facing public web applications or private processing engines. There is a variety of different programming languages supported: Python, Java, node.js and .NET. Other languages may also be available through Open Source projects\(^26\).

---

\(^22\) [http://aws.amazon.com/s3](http://aws.amazon.com/s3)

\(^23\) [https://developers.google.com/appengine/](https://developers.google.com/appengine/)


\(^26\) [https://www.windowsazure.com/en-us/home/scenarios/cloud-services/](https://www.windowsazure.com/en-us/home/scenarios/cloud-services/)
Chapter 3

Architecture and Design

The purpose of the tool is to give the user the possibility to get know how of the impact on the costs of moving SAP applications from a local environment to a cloud environment and vice versa. This user is the one creating the SAP system configurations before they are sent to the infrastructure department of a company. The results of the tool can be useful for Capgemini but also for the clients of Capgemini. In this chapter we will first describe the design decisions made, than we will write more detailed architectural information about the configuration and the distribution part of the tool. At the end of this chapter we describe the Graphical User Interface.

3.1 Requirements

With the purpose in mind it is very important to design the tool in such a way that it meets the requirements. These requirements are mentioned in the next section (Lethbridge and Laganiere, 2002). These requirements were derived from the purpose of the tool in combination with input from experts. During the design phase we got several proposals for additions and modifications.

3.1.1 Functional Requirements

The functional requirements describe what the system should do. These requirements are as follows:

Create a new configuration: Possibility to create a new clean configuration. This configuration should exist of:

- global cost related properties
- local instance properties
- cloud instance properties
- SAP application properties

Save a configuration: Possibility to save all the completed information in the tool without any loss of data.

Load a configuration: Possibility to load a complete configuration which restores all the data in the right places, exactly the same as just before the configuration was saved.

Export a configuration: It should be possible to export a configuration as a XLS file which can be opened in a spreadsheet viewer.

Calculate costs: The tool must be able to calculate the costs per SAP application for storage.

Calculate location: Given several constraints the tool must be able to make a first distribution between the local and cloud storage environment based on the costs involved.
Interactivity: The user must be able to influence the result provided by the tool in an easy manner.

Generic: The tool must be usable by several companies with several wishes. Therefore the tool must provide multiple options to manipulate the cost model in such a way that it fits the companies needs.

3.1.2 Non-functional Requirements

The non-functional requirements are requirements that describe what the system should be like, rather than what the system should do. The important non-functional requirements for the tool are as follows (Lethbridge and Laganiere, 2002):

Usability

The users targeted for the tool are people with a high level of knowledge of their domain. These people are familiar with the technical terms involved with SAP systems and data storage. During the development of the tool these users will be involved and we request feedback from them. This feedback is taken into account and used for further development of the tool. Because of the large amount of variables involved in the tool needed for the calculations, it is important to come up with a user-friendly GUI design. The tool will only contain features which are necessary to achieve the goal and features which make it easier to achieve this goal. There will be no features that are little used, as is the case with "shelfware". The tool is completely custom and build according to the needs of the user.

Efficiency

A configuration can consist of many elements. Long waiting times during the use of the tool is not desirable. Therefore it is important to keep efficiency into mind during the development of the tool. Not calculating things twice when it can be done once and keep everything well structured. The tool is not a complete system so it should not be hard to keep everything smooth and fast.

Reliability

By using defensive programming techniques the tool must become reliable. Users must be warned when improper input variables are provided instead of a software crash. Several configurations will be tested thoroughly and in each step of the development phase the tool is tested for failures.

Maintainability

Maintainability is very important. Because we are dealing with cloud solutions, the speed of development in that area is very high. Even during performing this thesis we noticed several changes within the cloud providers. Price changes, configuration changes, new configuration options and new elements involved in the pricing scheme. To keep up with the developments it is very important to structure the code in such a way that these kind of changes are easy to implement without great impacts on the code.

3.1.3 Design decisions

There are several possibilities in order to achieve the goal of creating a decision support tool for allocating SAP application data. The risk with many variables is that the system becomes very complex. To deal with this problem the system is cut up into several pieces. There are several objects that are responsible for the SAP applications, objects for the cloud instances and objects for the local instances. When changes to one of these parts have to be made, only the objects directly related to that part have to be changed. Functions belonging to a specific part are kept as much as
possible inside the part itself. Objects are kept as short as possible, just like the functions, in order
to keep the code clean and clear. Where possible, external toolkits are used to fulfill functionalities
which would otherwise take a large amount of time to develop ourselves.

The tool is written on the .NET framework with C# because the support for this framework
is high and the language is well known by many programmers. This makes editing the code by
other programmers easier. Because of the similarities with Java people with Java knowledge can
also easily use the C# programming language. A disadvantage is that C# is only well supported
under windows but because most companies use windows and C# is fast and powerful the tool is
developed in that language.

The tool consists of two main parts. The configuration and the distribution. The configuration
is the part where all the information must be gathered in order to let the distribution work. Changes
in the data collected in the configuration part can have a direct impact on the distribution.

3.2 Configuration design

The tool has to do many calculations. Therefore a great amount of information must be gathered.
To keep a good overview of the application structure we made use of object-oriented program-
ming. The real world artifacts are as much as possible mapped to objects. The result has to be a
distribution of the SAP applications between the local storage location and the public cloud stor-
age. The local instances can be physical servers in a co-location, or private cloud instances. The
public cloud instances are further defined by the properties related to the used cloud providers.
The most important objects are the SAP application objects which will eventually contain all the
information necessary for the distribution.

3.2.1 Architecture design

A partial object diagram is shown in figure 3.1. This diagram only consists of the main objects
used for the configuration screen (Rozanski and Woods, 2011).

![Object Diagram](image)

**Figure 3.1:** Part of the high level object diagram for the configuration
3.2 Configuration design

**MainWindow:** This is the main object where the GUI elements are directly related to. All the necessary information is accessible from within this class and can be passed on to other objects which need specific information in order to do their job.

**Costs:** The Costs object contains more global parameters. Many variables are different per instance or application but the variables which are the same for all the same types of instances or applications are collected in this object. Most of the variables in the Costs object are intended for costs calculations for complete configurations of a specific type. Such a configuration can be a traditional co-location with several servers. There can be only one Costs object.

**LocalSystem:** This is an abstract class with several common variables for the subclasses. LocalSystem represents a local instance of which there are two types. A local instance can be a traditional co-location server represented by the Colocation object and a private cloud instance represented by the PrivateCloud object. Both subclasses contain variables related to the instance type, a function to calculate the instance costs and a function to calculate the data related costs. The configuration can contain multiple local instance classes with no limit.

**Cloud:** This is also an abstract class with common variables for the subclasses. Different is that the abstract Cloud object is for the public cloud instances which has different common variables. The subclasses of the abstract Cloud object are instances of cloud providers. The tool supports cloud instances of AWS, Windows Azure, GCE and there is the possibility to define a custom cloud. The subclasses belonging to the cloud providers are respectively AmazonCloud, AzureCloud, GoogleCloud and CustomCloud. These subclasses also (just like the local instances) contain variables related to the instances of the cloud providers, a function to calculate the instance costs and a function to calculate the data related costs. The configuration can contain multiple cloud instances with no limit.

**SAPApp:** This is one of the most important objects. The SAPApp object represents a SAP application. This means that all the properties related to a SAP application and in the scope of the tool are included in the object and that the object is responsible for creating a visual object representation of its own object. This visual representation is a data block visible in the data distribution section of the tool. The location where the data block has to be stored according to the tool is also defined in the SAPApp object. Each SAPApp object can contain a reference to the local instance it can be stored on, and one reference to the cloud instance it can be stored on.

### 3.2.2 Exporting configurations

Configurations can become really big. Therefore there is the possibility to save and load configurations. In order to do this there is a Config object. This Config object can contain all the local instances, cloud instances, SAP applications and the Costs object. Once a user requests to save a configuration the Config object is created and all the required data is sent to the object. Once this is done the Config object is serialized to an XML (Extensible Markup Language)\(^1\) file. If a user wants to load the configuration again the file is deserialized from the XML file and put back in the configuration environment. The Config file is also used to export a configuration to an Excel file, the Config object contains all the information needed for this action.

### 3.2.3 Price retrieve model

An important function in the tool is the option to retrieve live pricing schemes from the Internet by using data from the cloud providers’ websites. The pricing data is used so that a part of the data

\(^1\)http://en.wikipedia.org/wiki/XML
needed for defining the variables for a cloud provider is automatically completed. This saves the user a lot of time and the risk of errors is much lower.

In figure 3.2 the structure for the pricing retrieve function is shown in an object diagram. The scraper retrieves the required data from the target website. Amazon Web Services has all their pricing data stored in JSON (Javascript Object Notation)\(^2\) files on their website. This is, just like XML, a structured data file which is human readable and easy to parse by the computer which makes it possible to be used for someone’s own purpose. In our case the JSON files as Amazon presents them are perfect for caching the pricing schemes. The needed JSON files are extracted from the Amazon website and saved. Windows Azure and Google Compute Engine do not provide such data files on their website. For these websites the data has to be scraped, also called web scraping\(^3\). This is extracting information from websites. For the genericity the data extracted from the Windows Azure and GCE website are converted to JSON files and saved as such.

**Figure 3.2:** Left, the object diagram for retrieving prices from the cloud. Right, the parser to use the prices for the cloud providers

**Scraper:** This object retrieves the data from the website and, in the case of Amazon, saves the data as a cache file. The data retrieved from the Windows Azure and Google Compute Engine website have to be processed. Therefore this data is forwarded respectively to the scrapeAzure and scrapeGoogle objects.

**scrapeAzure:** This object is responsible for processing the HTML code to usable code structures. The scrapeAzure object contains functions specified for processing the Azure website content. When the content is structured it is sent to the JBuilder object.

**scrapeGoogle:** What the scrapeAzure object does for the Windows Azure website does this object for the Google Compute Engine website. The structured content is forwarded to the JBuilder object for further processing.

**JBuilder:** The JBuilder object retrieves the content from the scrapeAzure or the scrapeGoogle object and processes this data to a JSON file. This is not just a regular JSON file but a JSON file which has the same structure as the Amazon JSON files. The object saves these files in a cache location.

Until now the pricing data is only stored in JSON format but it is not used yet. These JSON files are parsed by the Parser object as shown on the right side in figure 3.2. The data extracted from the JSON files is stored in a Cloud object which is used by the tool. The whole sequence is shown in figure 3.3.

\(^2\)http://en.wikipedia.org/wiki/JSON  
\(^3\)http://en.wikipedia.org/wiki/Web_scraping
3.3 Distribution design

The distribution section is responsible for presenting the data distribution between local and cloud storage instances. In the tool the distribution section is shown in a separate tab.

3.3.1 Architecture design

The draggable objects contain the belonging SAPApp objects. The SAPApp objects and the Costs object are forwarded to the Calculator class for calculations as shown in figure 3.4.

**MainWindow**: The distribution section is, just like the configuration, part of the MainWindow object. By making use of the Calculator object the MainWindow gathers visual objects with the belonging costs.

**Costs**: The Costs are sent to the Calculator object to have access to the global cost elements.
SAPApp: Each defined SAP application is forwarded to the Calculator object and a visual object is sent back which is presented in the distribution grid.

Calculator: A new object, not previously introduced in the configuration diagram in figure 3.1, is the Calculator object. All the SAPApp objects are passed through this object which allows the Calculator object to define the local and cloud costs for that object and store these costs in the SAPApp object. During the processing of the SAPApp object the storage location is defined based on the given variables.

3.3.2 Creating the distribution grid

Figure 3.5: A sequence diagram for adding objects to the distribution

To understand how the process of calculating the costs and creating the visible objects is executed we made it visible with a sequence diagram in figure 3.5. When the user wants to see the distribution of the SAP applications, the createDragableObjects function is called. A new Calculator object is created and a loop is set into action. For each SAPApp object the processItem function in the Calculator object is called, where the costs for the local instance and for the cloud instance are calculated. These costs are stored in the SAPApp object and after that the createRectangle function in the SAPApp object is called. This visible object is returned to the GUI where it is shown to the user.
3.4 Graphical User Interface design

The usefulness of the tool can be considered on two quality dimensions. The first one is that the tool has the raw functionality to allow the users to achieve their goal. Storing the right data, allowing the right operations and do the right calculations. This first dimension is called the utility of the tool. The second dimension is that the tool allows the user to use the raw capabilities easily, the usability (Lethbridge and Laganiere, 2002). This section is about the usability of the tool by designing a user-friendly GUI (Graphical User Interface).

3.4.1 Global design

The complete user interface originated during the development phase. One of the first ideas was to have two screens, one for the configuration details and one for the data distribution. The best way to solve this was to use tabs, see figure 3.6. There is not enough space on the screen to put the configuration details and the graphical distribution on one screen. Another alternative would be to only show the data distribution and put the configuration details in another screen or vice versa. There are several setbacks for this last solution. The user should navigate as little as possible and for a good clear overview sequences of modal dialogs should be avoided, since they slow users down and give them the feeling that the computer is in control of the interaction. Using tabs allows the user to separate the configuration and distribution and still have a complete overview of all the options. All the configuration elements are visible in the screen, a drawback of this solution is that the user is confronted with much information at once. The reason to do this in such a way is that all the variables can be changed with just a few mouse-clicks.

At the top of the screen is a menu which is always visible. The file menu contains standard options for creating a new configuration, open a configuration, save a configuration, export a configuration and close the application. The edit menu contains the cut, copy and paste functions for text. At the bottom of the screen is a status bar which can contain short status messages. This is used for instance to notify the user that a configuration is loaded or saved. A status bar on the bottom of the page is much more subtle than showing a message dialog each time an action must be notified to the user. There are several actions possible in the GUI where it can take a few seconds before the tool is finished. One of these actions is loading a configuration. During this event the user is shown a waiting bar and all the controls are disabled to prevent the user from taking further actions during the loading process.

3.4.2 The configuration tab

The configuration screen is separated in four columns. The first column contains variables which applies to configurations. Up till now we make a difference between three different costs:

**Instance costs:** Costs related to instances, cloud and local.

**Data costs:** Costs involved with the amount of data stored, transferred, backup-ed and requested.

**Configuration costs:** Costs applied to configurations, a configuration exists of a group of instances of the same type. All member of a private cloud, a traditional co-location or a public cloud.

**Configuration variables**

Next to the configuration variables there are several variables related to system backups. The variables are presented in a properties widget. The widget is separated in sections, each of them can be collapsed. The properties are shown with a label and a value next to it. These values can only be numbers, not text. When a user inserts text in a field in the properties widget the text is replaced...
by a zero value when the field has lost focus. Each value can be increased and decreased by using the up and down arrows next to the value field. There is one combobox in the properties widget, by selecting an option out of that combobox the website with pricing information belonging to the selected cloud provider is shown. Changing a value in the properties widget is enough, it is not necessary to press a save button. The value is saved after it has been changed. At the top of the widget there are two buttons and a search field. The buttons are meant to change the view, the user can switch between a categorized view and an alphabetical view. The categorized view is shown by default. The search field makes it possible to search for properties in the widget. When selecting a property, the user is informed with a small description at the bottom of the widget. The description provides some extra information about what kind of value is expected for that specific property.

![Figure 3.6: The configuration tab](image)

The other three columns are structured similar to each other. Each column contains from top to bottom a name field for a new item, a list of items, a label and at the bottom the item details in a properties widget. The columns are similar in order to improve the learnability. It is not possible to decrease the number of elements because of the large amount of variables, it is however possible to create a more intuitive process. Once the user knows how to threat one column, the others are less hard to learn. The columns contain from left to right information about the local instances, cloud instances and the SAP applications. This order can also be seen as a procedure. First the user fills out the configuration variables. When that's done the user continues adding a local instance, a cloud instance and the related SAP applications.

**Local instances**

The local instances column contains at the top a name field and a list. When a name is inserted and the enter key is pressed, a dialog pops up, see figure 3.7(a). The user can choose between a private cloud instance and a traditional co-location instance. Only the options belonging to a choice are active for inserting information. The first group of variables is to define the instance.
The second group is for the storage type and amount, these options are for both instance types. The last groups are dependent on the selected instance type. At the bottom of the dialog is some space left to inform the user. Just as is the case in the properties widget, a description is shown when the user selects a property control. The user should know what kind of information is requested, the constant support with information messages should help making this clear. When the user chooses for a private cloud instance the cloud instance dialog pops up when finished adding a local instance, see figure 3.7(b). This cloud instance dialog is already completed according to the amount of SAPS defined in the private cloud instance. This conversion step saves the user a lot of time because only the private cloud instance has to be defined.

![Dialog for adding a local instance](a)

![Dialog for adding a cloud instance](b)

**Figure 3.7: Instance dialogs**

After adding a local instance the instance is shown in the list. Each item in the list is accompanied with two small icons. The first icon indicates the system type as explained in section 2.1.2. These icons can have the letters PRD, DEV and QAS which stands respectively for production, development and quality assurance system. The second icon indicates the instance type, a cloud with a lock for the private cloud and a globe with a server for the traditional co-location. When the user selects an item from the list the label underneath the list shows which instance type is selected and the properties widget on the bottom of the column shows the properties belonging to that instance. When the user wants to edit or remove an item from the list this can be done by using the small pop-up menu which appears when clicking the right mouse-button on an item. Values belonging to a specific instance can be changed in the properties widget, but in order to change an instance type and a system type the pop-up menu has to be used.

**Cloud instances**

The cloud instances column, as mentioned before, has the same design as the local instances column. When providing a name and pressing the enter-key the cloud instance dialog pops up, see figure 3.7(b). Other than the local instance dialog, this dialog only contains comboboxes. The reason for this is that the cloud instance dialog is more the GUI for the price retrieve function, see section 3.2.3. The user has to select values for all the active comboboxes, otherwise a message is shown to notify the user that not all information is provided. The user has to choose a cloud provider and the virtual machine type. Only the possible solutions are shown each time. Just like with the local instance dialog, there is some space left at the bottom of the dialog with information for the user. When a cloud provider is selected the user can use the *Info* button to get more information about the optional virtual machines for that cloud provider, the button directs the user to the section on their website where this is explained.
When the options are selected and the \textit{OK} button is pressed the price information is retrieved live from the cloud provider. During this event a waiting dialog is shown, so the user knows that this can take a few seconds. When the cloud instance is added a new item appears in the list. Just like with the local instance each item in the list is accompanied with two small icons. The first is the same as with the local instance, the second icon indicates the cloud provider chosen for that instance. When the user selects an item, the label underneath the list shows the selected instance name. The properties belonging to the selected instance are shown in the properties widget at the bottom of the column. All the prices are already completed because of the price retrieve function. This feature saves the user a large amount of time because these values have not to be searched for on the website of the cloud provider, converted to the euro currency, and inserted by hand. The only values the user has to provide are quantities. The variables for the cloud providers are separated in categories to make it easier and more clear for the user. It is possible to update the prices of an instance to the most recent prices. By pressing the right mouse-button on an item in the list a pop-up menu appears with the options to remove or edit the item. By choosing \textit{edit} the user can easily press \textit{OK} in order to update the prices, the values are automatically completed according to the earlier defined values. Another possibility is to change the virtual machine, the prices change while the quantities remain the same as defined by the user.

\textbf{Custom cloud}

When choosing a custom cloud in the dialog only the system type can be defined. After adding a custom cloud the instance can be further defined using the properties widget. The formulas and the extra variables can be defined in the \textit{Variables} category.

The clouds owned by Amazon, Microsoft and Google have their own predefined formulas for calculating the prices. The advantage of the custom cloud option is that these formulas can be defined by the user. The variables can be chosen from three different comboboxes. The first combobox contains variables for the instance costs, the second combobox for the data costs and the third contains custom variables. The data related variables are only active when the user has chosen to add a data related formula. There are two kinds of formulas, instance formulas and data formulas. The instance formulas is only used for instances and the data formula is used for all the SAP applications in an instance. The variables in a combobox have a number followed with a title. The easiest way to use the formula editor is to select a variable from a combobox, which adds the belonging number directly to the formula field. A formula can be added by only using the mouse, all the available operators are presented as buttons underneath the comboboxes. By pressing one of these buttons the operator is added to the formula field. Another option is to add the formula into the field by hand. Before the formula is added to the list on the right side of the dialog box the formula is validated. If the formula is valid the field turns green and the formula is added to the list as is visible in figure 3.8(a), otherwise it turns red and the formula is not added. The custom
variables can be added in another dialog also accessible from within the variables category in the properties widget. Each new variable can have a name and a decimal value, these values are costs or amounts.

SAP applications

The last column in the configuration tab is the SAP applications column. Adding a name in the name field and pressing the enter-key adds a new SAP application to the list. Each SAP application is predefined with standard values. These preselected values are the most common values. Changing the name or removing an item from the list can be done with a menu which appears after clicking the right mouse-button on an item. Just like the instance columns, the SAP applications column also contains a label with the current selected item. The property fields are different however. The instance columns both have a properties widget which can contain all kinds of properties. This is necessary for the local and cloud instances because there are several types of instances possible with different variables. The SAP applications all have the same properties so a more static solution is possible in this case. Changing a value in the application specifications directly changes the value in the tool. Just as with all the properties widgets it is not necessary to confirm a change with a save or OK button. The last two options for the SAP application are the selection of the belonging instances. When the user named the local instance ECC, and the cloud instance ECC the instances selected by a SAP application named ECC are automatically selected. The options available in the instances only exist out of instances which have the same system type as defined in the SAP application specifications.

3.4.3 The distribution tab

Once a configuration is added in the configuration tab it is possible to get a graphical overview in the distribution tab. On the left side of the tab there are several options available. All the objects can be removed and a new calculation can be made. When a SAP application in the screen is selected a label on the left side notifies the user which application is selected. There is the possibility to change the local and cloud instance belonging to that application. The last two options exists of moving all the SAP applications to the cloud or all the applications to the local storage.

The graphical part of the screen is split in three sections. On the left side is the local storage area, in the middle a neutral area where SAP applications can be "parked", and on the right side is the cloud storage area. At the bottom of the local and cloud area is a cost distribution based on the costs of that area. Shown are the instance costs, the data costs and the total costs of both costs together. Behind the total costs is a number between brackets showing the difference with the former distribution. When the user moves the mouse over the costs the costs screen is enlarged and a complete costs distribution is shown, see figure 3.9. The neutral area has the total costs on the bottom. The large number at the top are the total costs, the smaller number underneath contains the difference with the former distribution. A graph at the bottom right of the neutral area shows the ratio between the local and cloud costs.

The first time the distribution tab is opened after a configuration is added in the configuration tab, a distribution is proposed by the tool. This distribution is based on the lowest costs and the several properties given to a SAP application. SAP application items can be dragged to other areas. The impact on the costs is visible at the bottom of the screen. When an area becomes more expensive, the costs are shown in the color green. When the area becomes less expensive the costs are shown in the color red. The most important number is the one in the middle, showing the total costs. If this number becomes green the total costs are lower. During the change of the costs the costs values are animated by an increasing and decreasing font size, which highlight the change. The neutral area has a small index at the top right showing the colors belonging to a privacy level.
3.5 Summary

The tool has to deal with configurations and many variables. The tool is able to save and load these configurations. Two important aspects of the tool are usability and maintainability. The tool should be easy and intuitive to use and easy to be maintained. The maintainability is important because of the rapid development in the cloud services industry.

The tool is build in an object-oriented manner in order to keep it well structured. The functions and objects are as small as possible to keep it clean and clear and therefore easy to maintain. The tool is separated in two main parts, a configuration and a distribution part.

The configuration section exists of configuration costs stored in the Costs object, local instances presented by an abstract object and cloud instances presented by an abstract object. All the options related to a SAP application are stored in a SAPApp object.

To make it easier for the user there is a possibility to retrieve live pricing information from the websites of the cloud providers. These cost values are automatically inserted into a Cloud object which is used by the tool.

Before the objects are shown in the distribution grid the costs are calculated using the Calculator object.

The GUI is build in such a way that it helps the user where possible. The GUI has the same structure as the architecture and the user is constantly supported with feedback and information. Local instances, cloud instances and SAP applications can easily be added and edited. Once a configuration is inserted the distribution tab shows a proposed distribution. By dragging applications from local to cloud the impact on the costs is clearly visible.

Each application block has a color matching one of the colors in the index. At the top of each application block is the title and underneath, the size in gigabytes and the costs for that block. The width of an application block is based on the data size of the SAP application. On the right side, from top to bottom, is a label which shows the system type of the SAP application. PRO, DEV and QUA which respectively mean production, development and quality assurance.
Chapter 4

Functional Design and Implementation

In this chapter we explain more about how the tool was developed in a more technical way. As mentioned before, the tool was programmed in C# and object-oriented programming was used. Before we started programming lines of code the structure of the tool was created. This structure changed during the development due to changes in the design of the tool. In the first section we tell something about the first steps in the tool and in the following sections we explain more about the toolkits used and about implementation decisions.

4.1 The first mockup

The first mockup of the tool existed of a few controls, see figure 4.1. The focus was on the data and the costs for the local and cloud storage were generalized. After doing more research in local storage solutions and the public cloud providers it seemed not as simple, see section 2.2.

Figure 4.1: A first mockup for the tool

The cloud providers offer many different solutions in the form of different instance types. There is not one cost model for storing data in the cloud. The costs are dependent on the cloud provider and the instance type chosen. The same goes for the local storage solutions. The traditional co-location costs are dependent of the number of servers and the amount of data. The private cloud has their own variables, more similar to the cloud solutions.
4.1 The configuration tab

4.1.1 XML serializer

The first goal was to make it possible to enter some variables in the tool and to add data blocks to a list of objects. Once a small configuration could be added to the tool the data could be used for further actions. To save time during the development of the tool the possibility to save and load a configuration with the tool had a very high priority. When the tool is able to load configurations we did not have to enter a fresh configuration each time we wanted to test something. There are several possibilities to save data to an external file with the purpose to save it. For very large configurations this can be done using a database. The tool has not that much data to save. And a disadvantage of using a database is that there should be a database available. The usage of the tool should be very approachable and the requirement to have a database management system installed does not support this. When not using a database the solution would be to store the data in a file. This can be done by requesting all the variables that have to be stored and store them in a structured file. An easier approach is to serialize the data to an external file. By using serialization, complete objects can be saved. Serialization is the process of converting a data structure or object state into a format that can be stored\(^1\).

To keep the serialized file well structured and human readable we chose for XML serialization (Bhatti and Hassan, 2007). All the data which has to be saved is collected in the Config object as described in section 3.2.2. This object contains the Costs object and three list types with local instances, cloud instances and SAP applications. Two functions in the object are taking care of serializing and deserializing the object. The serialized output is in XML format and is nested according to the structure of the Config object. The interpretation of objects by the XML serializer can be modified. The subclasses of the abstract objects where not recognized by default. By using the [XmlInclude(typeof(Objectname))\] command directly above the class definition it was possible to define the possible subclasses of the abstract class. We wanted to use some variables in the objects to be represented by an attribute in the XML output file. This could be done using the [XmlAttribute("attributename")\] command directly above the targeted variable declaration. The last command used for the specification of the serialization method is the [XmlIgnore\] command. Using this commands above the variable declaration has the effect that that variable will be ignored by the serializer.

4.2 The configuration tab

The configuration tab consists of several different controls which help the user to insert a configuration into the tool. One of these controls is the PropertyGrid which is explained in the first section.

4.2.1 The PropertyGrid

As mentioned in the GUI section 3.4, the configuration tab is separated in four columns. Several columns contain a properties widget, also called a PropertyGrid. This PropertyGrid is a control used from the Extended WPF Toolkit\(^2\). The PropertyGrid control allows the user to inspect and edit properties of an object. The Costs object, the LocalSystem object (presented by a Private-Cloud or Colocation object) and the Cloud object are linked by such a PropertyGrid. The Cloud object is presented by a AmazonCloud, AzureCloud, GoogleCloud or CustomCloud object. Some items in the linked objects should not be presented in a PropertyGrid, these items can have the [Browsable(false)]\] command directly above their declaration. When that command is used than that variable is not visible in the PropertyGrid. For the combobox in the configuration costs widget an ItemsSource is used. This source is defined in the Costs object and contains the selectable

---

\(^1\)http://en.wikipedia.org/wiki/Serialization
\(^2\)http://wpftoolkit.codeplex.com/
options. When the user changes a variable in the PropertyGrid, the value is directly updated in the object.

### 4.2.2 Local instances

If the user enters a name in the name field and presses the enter-key an event is triggered to open the dialog for adding a local instance. This dialog is a new WPF window with the properties defined in such a way that the window acts like a dialog window. This means that the window is not resizable, not shown in the taskbar and the window only has a close button in the top right corner.

**Dialog structure**

As long as there is no instance method selected it is not possible to add the instance to the list. Once a method is selected the OK-button becomes active and the co-location or the private cloud section, dependent on the chosen method. Together with enabling the controls the LocalSystem object in the dialog class is set to a Colocation or PrivateCloud object respectively. When the user focuses a control the information field on the bottom of the dialog shows content explaining what kind of value is expected. To keep the code clean there is only one element focus function. When a control is activated and the function is triggered the control is identified in the function. The label containing the information is then set to the defined content belonging to the control. All the available combobox values in the dialog are defined in the dialog class and referred to in the XAML file.

**Adding an instance**

After the user has entered information in the fields and clicks on the OK-button the variables of the LocalSystem are set according to the provided values, and the values of the derived class are set. When this is done the dialog returns true. The MainWindow object from where the dialog was called retrieves the true value and requests the LocalSystem object from the dialog. This object is added to a listbox. The added item to the listbox is selected by default so that the user can check and/or change the values from that object. The recently added object is also scrolled into view, for if the list is that long that the item is not visible in the first place. The LocalSystem object contains an overridden Equals function which matches the name of the object and the system type. This combination should be unique and is therefore enough to match an object. This Equals function is used when the selected item is set in the listbox. The ToString function is also overridden which ensures that the name of the object is shown in the listbox. The icons shown in the listbox are defined in the LocalSystem object and linked to in the XAML file belonging to the dialog.

**Converting private cloud to public cloud**

If the user added a private cloud instance to the list a function is called to convert the private cloud values to an AWS cloud configuration. The configuration is converted to an AWS cloud because of the partnership between SAP and Amazon. The other supported cloud providers do not have such a partnership with SAP. The convert function creates a AmazonCloud object based on the amount of SAPS in combination with the recommendations by Amazon. This recommended cloud configuration is send to a dialog, the same dialog which is used to add a new cloud instance. The predefined options can be changed by the user, so it is not forced upon the user. The rest of the procedure is the same as with adding a new cloud, see section 4.2.3.
Modifying instances

When the user wants to remove an instance this can be done using the pop-up menu appearing after clicking the right mouse-button on a listitem. The instance is than removed from the list. The same pop-up menu can be used to edit an item. When the user wants to edit an item, the former defined configuration is sent to the dialog. A function in the dialog class sets all the dialog controls to the values matching the given configuration. Changing the values in the dialog will change the configuration which is edited at the moment. When a SAP application is selected at the moment, the selectable local instances belonging to a SAP application are updated to the most recent situation after adding or editing an instance.

Editing the instance related properties can also be done by using the PropertyGrid at the bottom of the column. Changing values will directly be changed in the instance class.

4.2.3 Cloud instances

Adding a cloud instance to the configuration is different than adding a local instance. Especially because the cloud instance dialog is based on retrieving live pricing information. The user can get the cloud dialog after adding a name in the name field and pressing enter, just like with the local instance.

Dialog structure

The options in the dialog are based on the chosen value in the combobox for the provider. When the user selects a provider the dialog class’ Cloud object is set to the cloud providers object. Only the options possible with a specific cloud provider are made active, all the other controls are reset and deactivated. The sizes for the cloud instances are different per cloud provider. There is a function that adds the belonging instance sizes to the combobox for the instance size. For Google and Azure this combobox is set after selecting a provider because there are no other possible options. When choosing for Google Compute Engine the type, class and OS are automatically set and can not be changed. There are no other possible combinations, therefore this choice is made by the tool. When the user chooses for a custom cloud there are no further options available in the virtual machine group because the definitions for the custom cloud has to be made in another stage. The information method is the same as in the case of the local instances.

Selecting options

The cloud location can only be chosen when the user chooses for AWS or GCE. Windows Azure does have locations in different continents but it is not possible for the user to choose a preferred location. The other cloud providers do have that possibility. Each time when an option in a combobox is selected, only the options possible in combination with the selected option are activated. When the selected provider is AWS, the instance sizes are available when an instance class is selected. Each instance class has different sizes so the function that adds the available instance sizes checks which instance class is chosen.

Information

The dialog contains an Info button. This button can be used when a provider is selected. The button directs the user to the website of the selected cloud provider where there is information available about the possible instances. This can help the user making the right choice.
Adding an instance

When the user presses the OK-button the selected configuration is checked for inconsistencies. If there are controls that are active but not provided with a value, an error message is shown to the user. The user can not add an instance until all the active options are set. When the selected provider is a custom cloud the instance is added to the list and the user has to define the custom cloud further by using the PropertyGrid. When another provider is selected the BackgroundWorker class is used to create a separate thread. In this thread the price retrieve function is executed.

Retrieving pricing information

A new Parser object is created and the retrieve function is called, see section 3.2.3. In the retrieve function the cloud provider is detected and a function related to the cloud provider is called. Each cloud related function starts by creating their class, derived from the Cloud object. When this is done the cache function is called from the Scraper object. The cache function belonging to the AWS provider calls the JSON files from their website. Only the JSON files needed for the chosen configuration are cached.

There is a common cacheData function. In this function it is checked whether a cache file already exists for the same date. Each cache file is saved with an identifiable name and a date. When a cache file already exists for that date, the function is finished and no data has to be retrieved from the Internet. If there is no Internet connection available, the website is down or there is another reason why it is not possible to retrieve the data an older cache file is used when existent. The user is informed with a message box if an older cache file is used or if no pricing information could be retrieved at all. If there is no recent cache file and the tool is able to connect to the target URL (Uniform Resource Locator)\(^3\), a connection is made. The content from the target URL is received and saved in a new cache file. The older cache files from the same URL are removed from the cache folder.

The Windows Azure and the Google cloud do not have JSON files available. For these cloud providers there is an extra step involved in the process. The HTML data is retrieved from their website and cached with the cacheData function. The retrieved data is send to the ScrapeAzure or ScrapeGoogle object as shown in section 3.2.3. The scrape objects consist of several functions. One for obtaining instance price information, one for storage price information etc. The JSON files for the Amazon cloud are also separated this way, the JSON files structure has to become the same for the other clouds in order to retain consistency. The HTML file is processed based on unique text elements, like ID’s. The cloud providers both use tables to store their pricing data, so only the required table is extracted from the HTML code. This extracting process is done by a scrapeData function in the Scraper object. This is a static function so that it can be used by several objects. The complete Scraper object is static for this reason. The scrapeData function requires data from a start identifier till an end identifier. For instance \(<table\> and \</table\>\), in order to get the correct table an unique identifier can be provided to the function so it knows where to start looking.

There are several functions in the scraper objects which help retrieving specific information. When one of the two cloud providers changes their website these functions can easily be changed using different identifiers. All the required pricing elements are stored in Dictionary types. A Dictionary provides the possibility to store key value pairs. That is exactly what we need in order to build the JSON file in a later stage. A new JBuilder object is created for each new to create JSON file. This object is provided with the Dictionary file and some values which help to define what kind of JSON structure is needed for the specified data. In the JBuilder object the Dictionary data is used to create a JSON object. To create this JSON object, the Json.NET JSON framework is used\(^4\). When the JSON object is ready it is written to an external JSON cache file in the cache.

\(^3\)http://en.wikipedia.org/wiki/Uniform_resource_locator
\(^4\)http://nuget.org/packages/newtonsoft.json
folder. Each cache function finishes by returning true to notify the Parser object that the caching process is done.

A getData function in the Scraper object checks if the required cached JSON file is available and returns a string with the JSON data. The Parser object contains functions to require the requested data from the JSON files after it has been parsed to a JSON object. The required data from the JSON files is stored in the object derived from the Cloud object and returned to the dialog class from the cloud, where it is saved. Retrieving the pricing information can take a few seconds, during this time the dialog is deactivated and a layer on top of the dialog shows a dialog box with a message and an animated bar to inform the user that the tool is busy. The dialog box used for this is also part of the Extended WPF Toolkit, called BusyIndicator.

Providing further data

In the MainWindow object from where the dialog was called the Cloud object is retrieved and added to the listbox for the cloud instances. The further procedure is the same as with adding a local instance to the designated listbox. In contrast to adding a local instance, the cloud instance is not finished yet. The only values completed in the cloud instance are the prices. The quantities still have to be completed by the user. These are completely different based on a configuration, which means these can not be automatically completed by the tool.

Custom cloud

The CustomCloud object contains a List type for variables and two List types for the instance formulas and dynamic formulas. All these List types are used at a later stage by the Calculator object to calculate the costs for the CustomCloud. There is a function in the CustomCloud which can be triggered by the user to open a formula dialog. This function is triggered when the user chooses for add instance formulas or add data formulas from the combobox next to add formulas. In both cases the same formula dialog is opened but when the user chose for the option to add instance formulas, it is not possible to use the variables which has to do with dynamic data.

When the formula editor dialog is initialized the predefined variables are added to the list boxes. If there are custom variables set by the user the custom variables combobox is enabled, otherwise it stays disabled. Then the custom variables are added to the designated listbox.

The user must provide a name for the new formula and then has to construct a formula in the field underneath. Adding a variable can be done by selecting a variable from one of the comboboxes or the user can add the referrer to the variable themselves. In each combobox the referrer is shown with the variable name next to it. Underneath the comboboxes there are several operators which can be used by the user. When one of them is clicked it is added at the place of the cursor inside the formula field.

Once a formula is finished the user can add it with the Add-button. The formula is then validated for suitability. This is done by checking if there are no alphabetic characters in the formula field and if the formula can be parsed by the Mathematical Expressions Evaluator. This evaluator is used for executing the formulas in a later stage. When the formula is not valid the formula field turns red and the formula is not added to the list of formulas, otherwise the field turns green and the formula is added.

Editing a formula can be done by selecting one from the formula list and changing the formula. The Add-button then turns into an Edit-button, also visible to the user by the label. When the user wants to add a new formula after a formula is selected, the New-button can be used. The info-field at the bottom of the dialog shows an example of a formula to the user.

The OK-button tells the CustomCloud object that the dialog is done. The formulas are then retrieved from the dialog class and saved into the designated List types.

---

Custom variables can be added to a CustomCloud object using a dialog provided by the PropertyGrid for List types. In the dialog VarItem objects can be added to the List type. Each VarItem can contain a name and a decimal value. A decimal value is chosen because all the custom variables have to be numbers.

4.2.4 SAP applications

In the SAP applications column the user starts again by entering a name in the name field. No dialog pops-up as with the former two columns but the item is directly added to the listbox with some predefined values. The predefined values are based on some logical values, most often used. This saves the user from selecting the same values every time a new SAP application is added to the list. Each time a value is changed in a SAP application object and the control loses focus, a function is called. This function checks which control lost focus and saves the new changed value in the SAPApp object. The open fields for the size of data and the amount of data transfer are DecimalUpDown boxes from the Extended WPF Toolkit. These boxes only allow numeric values and no alphabetic ones. When the user wants to remove a SAP application this can be done by using the pop-up menu after clicking the right mouse-button on a listitem.

4.3 The distribution tab

The first mockup contained a simple interface with the possibility to insert a small configuration and it was possible to save and load this configuration. The next step was to come up with a graphical interface where data blocks would be shown and able to be dragged around. The first mockup was built using Windows Forms. While thinking about the graphical part and especially the intended features a better solution was to build the interface using WPF (Windows Presentation Foundation). WPF is a computer-software graphical subsystem for rendering user interfaces in Windows-based applications. WPF employs XAML (Extensible Application Markup Language). XAML is a XML-based language used for initializing structured values and objects. WPF is the future of .NET user interface development which provides better design tools, some nice support for flashy display transitions and animations (Freeman, 2010). Especially the better animation support is useful for the graphical part of the tool. Cost changes can be animated to make changes very clear to the user. A tabcontrol was added to make a difference between the configuration part and the graphical distribution part. The current configuration items where placed in the configuration tab.

4.3.1 The distribution area

The areas in the distribution tab are Canvas elements contained in a common Grid. The Grid is separated in two columns, the left column is for the controls and the right column for the graphical screen. Each Canvas has a ScrollViewer which enables the element to be scrollable. When there are too many SAP applications in an area to show at once, the user can scroll down to see the elements which are out of view. The local and cloud area both have a gradient to provide a smooth transition between the areas. These areas have a fixed width and the area in the middle, the neutral area, takes the remaining space of the screenspace. The reason for this decision is that the neutral area is not that important as the local and cloud area. SAP applications “parked” in the neutral area are not part of the cost calculations. In most configurations the neutral area should be empty when the distribution is completed.

\[\text{http://wpftoolkit.codeplex.com/wikipage?title=DecimalUpDown}\]
\[\text{http://en.wikipedia.org/wiki/Windows_Presentation_Foundation}\]
When clicking on the distribution tab a function is called to create the draggable objects. This function clears all the current objects in the area, creates a Calculator object and sends all the SAPApp objects from the list to this object. Each SAPApp object can have a reference to a local and cloud instance. This reference is the index number of the instance in the list. When a SAPApp object is forwarded to the Calculator object the referenced local and cloud instance are also forwarded.

4.3.2 Calculating the costs

The SAPApp object is received by a function in the Calculator object. The data and instance costs are directly requested from the local and cloud instances. This is done by functions which detect what type of local and cloud instance is used and request the costs from the correct object. These costs are added to the SAPApp object and added to the storage type for data costs and instance costs in the Calculator object. The costs for an instance are only added to this storage type if it is not added before, this is tracked by using a list with added instances. When the costs are calculated the location where the SAP application should be stored is defined. This location is based on the costs for the local instance, the cloud instance and the other defined values for a SAP application.

Cloud object structure

The AmazonCloud, AzureCloud and the GoogleCloud objects are filled with public variables. All these variables have a category, display name and description which are used by the PropertyGrid. There are two functions in each of the objects. A fixedCosts function and a dynamicCosts function. All the cost elements are calculated in these functions. These elements are added to a Dictionary type. The reason to do this is to keep track of the separate cost elements. These can later be used to provide the user with a cost distribution. In the fixedCosts function the costs related to an instance are calculated. These costs are only calculated per instance. In the dynamicCosts function the data costs are calculated, this is done for each SAP application. The Dictionary type is returned to the Calculator object when that object requests costs from the Cloud derived object. In the Calculator object the Dictionary type is added to another Dictionary type. This results in a nested Dictionary type with cost elements. Both costs functions contain a cost element for the backup costs. To calculate these backup costs the size for the backup has to be obtained. The size for a backup is calculated in a function in the SAPApp object. The variables used to obtain the backup size are stored in the Costs object.

Custom cloud

The global structure of the CustomCloud object is the same as the other cloud objects. Great difference are some extra functions for processing the formulas and custom variables. When one of the costs functions is called from the CustomCloud object these functions are triggered. All the predefined and custom variables are added to a string array. By default some standard formulas are added based on the other cloud providers. In the costs functions all defined formulas are processed. A formula contains operators and index numbers. The format of these index numbers are the same as expected for the String.Format function. These index numbers refer to the index numbers of the string array which is filled earlier with the predefined and custom variables. The String.Format function takes care that the index numbers in the formula are replaced by the variable values from the string array.

Each formula is parsed by NCale\(^9\). This is a Mathematical Expressions Evaluator for .NET. NCale can process any expression and evaluate the result. This result is saved in a Dictionary type just as with the other cloud provider objects.

\(^9\)http://ncalc.codeplex.com/
Creating graphical objects

When the costs are processed and added to the SAPApp objects the graphical object can be constructed. The Calculator object calls the createRectangle object in the SAPApp object which is responsible for creating the graphical object. This graphical object is a Grid with a Rectangle and several TextBlock controls. The position of the Grid is also defined in the function. This position is saved in the Calculator object so this object knows where the next graphical object has to be placed. The Grid contains its own SAPApp object so it can be referred to when graphical objects are dragged and clicked on. The width of the object is decided by taking a look at the defined data size for the SAP application. The color for the object is based on the privacy level, a color index explaining which color belongs to what privacy level is provided at the top right corner of the neutral area in the distribution grid. The Calculator object returns the Grid to the MainWindow. There a mousebutton eventhandler is added to the Grid so the object knows when it is clicked on. When this is done the graphical object is added to the area of the defined location.

4.3.3 Showing the costs

After processing all the SAP applications the configuration costs are calculated. These are costs which are calculated only once per configuration. In this tool these configuration costs are only needed for the local instances. These configuration costs are added to the instance costs Dictionary type in the Calculator so they are also visible in the costs distribution. The costs for the local storage, cloud storage and the overall total costs are added to the designated labels. Changes in the costs are emphasized by using animations. These animations change the font-size from large to the original size in a defined time span. The color is changed in the same time span. The font-color turns red when the costs are higher than before and green when they are lower. The chart in the neutral area is set based on the ratio between the local and cloud costs.

When a user moves the mouse over the total local or cloud costs a function is called which resizes the costs area with an animation to a larger area. This provides more room for the costs distribution which is then shown in the area. The cost elements are retrieved from the Dictionary type where they are all stored. The costs shown in the area are indented for further distribution. The area has a scrollbar when necessary so that the user can scroll through the cost if the height is higher than the area space.

4.3.4 Interacting with graphical objects

Each area in the distribution Grid has drag drop events linked. These events are DragOver, DragEnter, DragLeave and Drop. Each graphical object has a mousedown event linked. When the user clicks on a graphical object this function is called. The opacity of the clicked object is than decreased and the object is cloned, these actions are done so that the user knows it is currently interacting with an object. This cloned object is made transparent and saved as a global variable. The DragOver event takes care that the cloned object is moved with the mouse cursor as long as the mouse button is not released. This event is referred to as dragging. When the mouse-button is already released in the same area this is seen as just a mouse-click. In that case the possible instances for the graphical object are shown in the left column of the distribution tab and the linked instances to the SAPApp stored in the object are automatically selected. These comboboxes allow the user to change the linked instances to the object.

Releasing the mouse-button after clicking an object is handled as a Drop event. When leaving an area while dragging a graphical object, the cloned object is removed from that Canvas area and added to the entered Canvas. For example, when the object leaves the local area it is added to the neutral area. If the user releases the object in another area than where it was located the location of the object is changed to the new location and all the costs are recalculated.

In the left column of the distribution tab there are two buttons. One for moving all the graphical objects to the local storage and one to move all of the object to the cloud storage. This means
that all the locations in the SAPApp objects are changed to the targeted location and that the costs are recalculated.

4.4 Menu options

When the user wants to create a new configuration the New option can be used. In that case all the fields are cleared and the tool is brought back to its starting state. Before this happens the user is warned with a dialog message that the current content is removed. The user has to approve in order to continue the process. When the user chooses to open an existing project the XML file, as described in section 4.1.1 is deserialized to the Config object. All the values in the Config object are completed in the designated controls. A configuration can be "saved" or "saved as". When no existing configuration is opened the "save" function is working exactly the same as the "save as" function. The "save" function saves the changes to the opened file. The "save as" function asks the user to provide a new name for the file and serializes the prepared Config object to that file in the provided file location.

4.4.1 Export function

The last functional option in the menu is the export function. By using this function the user can export a configuration to a XLS-file for spreadsheet programs. For this function there is a special Excel object. This object uses the Config object for its information. In the Excel object all the SAP applications are divided based on system types. This results in four separate List types with SAP applications. A list for production, development, quality assurance and test systems. All these lists are printed to a XLS file with subheaders between the different system types.

4.5 Summary

The tool has grown from a simple interface with a few variables to a complete interface with lots of options. Configurations can be stored, loaded and even exported to a excel sheet. The tool provides the user with lots of information where necessary. Local instances can be added, edited and removed.

To save the user time, local private cloud instances are converted to cloud instances using a conversion table between SAPS and systems. The cloud instances are partly completed by a price retrieve function which retrieves live pricing information from the Internet. To prevent requesting pricing information many times on a day from the Internet, the data is cached. The Amazon pricing information is already in the structured JSON format, the other cloud prices are also converted to JSON files. The pricing data is parsed and used in the cloud objects. When the user wants to use another cloud than the ones provided by the tool, the custom cloud option can be used. This cloud allows the user to add an unlimited amount of self defined variables and formulas adapted to the user’s needs.

All the data stored in and linked to a SAP application is used to calculate the total costs involved in a configuration. When the user is not satisfied with the data distribution between local and cloud storage it is easy to drag objects to another location. The costs are recalculated and the impact is clearly visible. A more detailed costs distribution is visible by moving the mouse over the total costs at the bottom of a canvas area. The instances linked to a SAP application can also be changed under the distribution tab so even this change is instantly visible.
Chapter 5

Detailed Description

The decision support tool is able to distribute predefined SAP applications between an on-premise location and an on-demand location. On-premise in the tool means co-location and the private cloud, on-demand means the public cloud. In order to do this it is possible to define several input variables which are necessary to perform the calculations. These variables are presented in tables A.1 to A.7. There are several fixed- and dynamic costs. The fixed costs are related to the basic instance costs, costs which are not dependent on the amount of SAP applications running on an instance. The dynamic costs differ related to the application specifications of the SAP applications.

This chapter provides more detailed information about the variables used by the decision support tool and some possible configurations including their costs functions. A detailed description of all the variables is available in Appendix A and a detailed description of the costs functions is available in Appendix B.

5.1 Configuration options

The tool has to be very generic in order to be used by all kinds of companies and not by one specific company. Therefore many variables can be defined. Once all the necessary variables are defined the tool is able to make decisions and come up with a data distribution. This distribution is visible in a different tab. In this tab the data blocks are divided between the local and the cloud area and there is a possibility to drag data blocks to other areas. The impact of the actions are visible in the total amount of costs. The several possible options are described in the next sections.

5.1.1 Global options

The global options are all presented on the left side of the application. These properties consist of variables for backups, traditional co-location calculations and calculations for the private cloud. All the costs are calculated per month. Most cloud providers provide their costs per month so for more consistency the costs related to the traditional co-location are also defined per month. A total list of the factors related to the traditional co-location costs, including a description, is shown in table A.1. These factors are based on the factors as described by Amazon in their EC2 Cost Comparison Calculator and confirmed by Kozhipurath (2012).

The private cloud has some similar costs as traditional co-location. Several outsourcers are modernizing their data centers with private clouds (Chawla and Sogani, 2011). The costs for the private cloud used for SAP systems is highly based on SAPS, which defines the system performance. See table A.2.
5.1 Configuration options

The last global parameter options exist of parameters to calculate the total amount of backup storage in GB. There are global backup options and options related to the type of system, production or non-production. See table A.4. These variables are obtained from Capgemini and confirmed by Chervenak et al. (1998).

The global options can be extended easily by adding public variables to the Costs object. These variables can then be referred to in the cost formulas used in the several cloud objects. If a programmer adds a public variable to the Costs object, the variable becomes automatically visible and editable in the designated PropertyGrid. It is not mandatory to complete all the variables in the global options. All the variables have as default value zero so they do not have a negative or positive impact on the costs when not used. A company often only uses a traditional co-location or a private cloud, not both. This means that only one of these two categories have to be completed and not both.

5.1.2 Cloud options

At the moment Amazon is the most popular cloud service. Just like Windows Azure, they provide the possibility for the use of a Virtual Private Cloud which can be attractive for companies to use (Staten et al., 2009). AWS has a very wide choice of IaaS instance types, especially in comparison to Windows Azure which has a lack of high memory instances and are mostly providing PaaS solutions. The several cost elements per cloud provider are provided in the tables A.6 to A.7. The IaaS product from Google is Google Compute Engine, they provide several instance types from which the high-memory types are the most interesting. These are the ones available in the price retrieve function in the tool. The high-memory instances are the most interesting because they are comparable with the AWS cloud instances which are recommended by SAP.

The dynamic costs for the cloud providers are similar to each other and exist of the variable parameters storage, data traffic and backup storage shown in table A.3.

Adding variables to a cloud is even easier as adding variables to the global options. The programmer has to add a public variable to the cloud object and refer to this variable in the cost functions which are located in the same object. By adding a new item to the Dictionary type in the cost functions the costs are shown as a separate expense in the distribution tab. It is important that these objects are easy to extend because of the many developments in the cloud industry. Adding a new cloud can simply be done by copying one of the existing clouds and adjusting the variables and formulas as preferred. A disadvantage is that the cloud dialog has to be changed for the available options and two lines of code have to be added in the Calculator object for the calculations. Even easier is to add a cloud on user level. This can be done by using a custom cloud and defining new variables and formulas.

5.1.3 Data properties

In order to make decisions based on the different SAP applications, several input variables are required. A SAP application is a module in the SAP ECC system. Examples are CRM, BW, ERP and SRM.

Each SAP application can obtain the following properties in the tool, the available options are shown between the []:

Name: The label given to the SAP application, most often the name of the SAP application.

Size of data: The total size of data in GB for database storage related to the SAP application. This number is required for the cost calculations.
**Data transfer:** The total amount of data in GB which is transferred from the server to other locations each month. This number is required for the cost calculations.

**Privacy sensitivity:** The privacy sensitivity of the SAP application data, companies still have trust issues with the public cloud, see section 2.2.2. [very low, low, medium, high, very high]

**Storage location:** Restrictions to the storage location, some countries have privacy laws which restrict the storage location of data (Doelitzscher et al., 2010). [own storage location, inside own country, inside the EU, doesn’t matter]

**System type:** The purpose of the SAP application, is it for development, production etc., see section 2.1.2. [testing, production, quality assurance, development]

**Local instance:** The local instance used for this SAP application. The options exist of local instances with the same system type. [...]

**Cloud instance:** The cloud instance used for this SAP application. The options exist of cloud instance with the same system type. [...]

Adding options to a SAP application is less easy than adding options to cloud instances. Reason is that the cloud is changing very fast, where the SAP applications are more stable. Adding options to a SAP application implies adding controls to the window, and adding two lines of code to the MainWindow object for saving and retrieving values in the control. The last change is adding the variable in the SAPApp object.

## 5.2 Costs functions

All the variables in section 5.1 are used for calculations. The formulas used for the calculations are presented in Appendix B. In this section an explanation is given for the formulas. The licensing costs are not included in the costs comparison. The local and cloud instances both need licenses. The licensing costs are therefore excluded.

### 5.2.1 Traditional co-location costs

The costs for traditional co-location as used by most larger companies do exist of several factors as listed in table A.1. The tool provides the user the possibility to define all these factors as is best suitable for their company. The costs exist of instance costs and data costs. The costs are calculated as shown in table B.2. The table is categorized in the categories instance, data, configuration and total. The same categorization as with the variables tables. The formulas used in the table under the category "configuration" are based on the Amazon EC2 cost comparison calculator and explained by Varia and Papo (2012). In the traditional co-location example the configuration costs are suggestions. The user does not have to use these calculations. These suggested formulas are only used when the user entered a value of -1 in an indicated field. The description of a value field informs the user if a suggested formula is available.

### 5.2.2 Private cloud costs

The costs for the private cloud are similar to the public cloud but on a smaller scale (Khajeh-Hosseini et al., 2010). In case of a co-located private cloud there is also a pricing scheme, just like with the public cloud providers. The public cloud providers pricing schemes are all billed on a pay-per-use basis. Another possibility is a combination with per-SAP Application Performance Standard (SAPS) basis (Smith, 2011). The costs are calculated per SAPS instead of instance costs. SAPS are often used to measure the performance of a system configuration running SAP
software, see section 2.2.4. Each instance needs storage for the operating system and the SAP software which is measured in GB per month. The price is dependent on the tier used, see section 2.2.1. The data costs are also dependent on the tier. The configuration costs are similar to the configuration costs for the traditional co-location. Each system type has its own VLAN and each configuration has its own firewall, these costs together represent the networking costs.

5.2.3 Public cloud costs

The public cloud costs are defined by the cloud providers. Each provider has its own pricing scheme dependent on the selected configuration. The cost elements are roughly similar, these similarities are used for defining the standard formulas and variables for the custom cloud in the tool. Each cloud has the variables processing hour, storage, I/O requests and traffic costs. Some of the cloud providers have cheaper storage for backups, like AWS and GCE. The cost calculations for the clouds are shown in tables B.4 to B.6. AWS distinguishes different types of I/O requests, GCE and Azure do not have such a distinction.

5.3 Possible configurations

There are some problems related to a SAP configuration which should eventually be distributed between an on-premise location and an on-demand location. The ideal situation would be to separate all private data from less private data in order to keep the more sensitive data in-house (Dillon et al., 2010). This can also mean that tables have to be split in two separate tables, one local and one in the cloud. In case of a SAP system the data has to be stored on the same location as the application, this can be all in the cloud or all local but not in separate places. The reason is that the response time has to be very low if the SAP application wants to retrieve data from the database, otherwise the application will not work. Storing the data and the application in the same datacenter or nearby would work depending on the response time. A longer response time can also cause delays which directly affect the user (Schneider, 2006).

When dealing with SAP systems the only possibility in distributing the data is to make decisions on application level. This means that only applications like CRM (Customer Relationship Management), SCM (Supply Chain Management) etc. can be separated from each other but not the application itself.

5.3.1 Local configurations

A local configuration can exist of a traditional co-location and a private cloud. The two configuration types will almost never be used together, but the tool provides the possibility to do so. The options for the local instances are completely defined by the costs. A larger server with more memory means more costs. The tool does not provide predefined configurations with linked pricing information for the local instances. The costs are dependent on many factors which are different for each company. The user is free to provide the costs applicable to the intended company. Cost elements which are not necessary can be left out by keeping their value zero. Costs which are not specifically defined by the tool can be added to similar costs. An example of such costs can be indirect costs which are not available as separate variables in the tool.

5.3.2 Cloud configurations

The specifications for cloud instances are different for each cloud provider. The tool provides the most options for AWS. The reason for this is that there is a collaboration between SAP and Amazon and the SAP systems are tested on AWS. The configurations shown in tables 5.1 to 5.2 are the configurations from which the pricing information can automatically be retrieved. The user is
completely free to edit the pricing information afterwards for more specific configurations. If the configuration is not between the available options this does not mean that it can not be used for the calculations. A user can provide the costs for a public cloud instance by hand, according to another instance type.

One of the strong points of the cloud is the elasticity and the flexibility, only use the resources when necessary. Therefore the tool contains on-demand cloud instances in the price retrieve function for all the cloud providers. Only the Amazon cloud contains reserved instances in the price retrieve function.

Amazon Web Services

When making use of the Amazon cloud, Amazon recommends to use the SAP certified Amazon EC2 instances. SAP and AWS have tested the performance of the underlying AWS resources, verified their performance, and certified them against the same standards that apply to servers and virtual platforms. For storage they recommend Elastic Block Store (EBS) and backups can best be transported to the Simple Storage Service (S3). To secure the whole system the Virtual Private Cloud (VPC) service can be used. The VPC enables the user to provision a private, isolated section of the AWS cloud (Wood et al., 2009). AWS resources can be launched in an own defined virtual network. When using SAP applications in a local environment and in the cloud it is possible to use Amazon Direct Connect which allows a dedicated/private network connection between the corporate network and the VPC (SAP, 2012).

In order to ensure full supportability of the SAP software by AWS and SAP they recommend the use of the High-memory Double Extra Large instance with 13.0 ECU (Elastic Compute Unit) and representing 3700 SAPS. And the High-memory Quadruple Extra Large instance with 26.0 ECU and representing 7400 SAPS. Both options are present in the tool, but for more flexibility all the other available instance types are optional. The tool provides configuration options for reserved instances and for on-demand instances.

The available AWS instance options in the tool are presented in table 5.1, ECU stands for EC2 Compute Unit.

Google Compute Engine

For Google App Engine there was already the possibility to run SAP but this only got better with GCE. App Engine is a PaaS and GCE is an IaaS which offers much more possibilities for a SAP configuration, see section 2.2.2. The price retrieve function only supports the high-memory on-demand instances from GCE because these are, referring to the suggested configurations by SAP in the former paragraph, the most suitable.

When using GCE a Virtual Machine is needed, provisioned storage for the SAP application and operating system, Google Cloud Storage for data storage and snapshot storage for backups. The Google Secure Data Connector can be used for a secure connection with the local environment.

The available GCE instance options in the tool are presented in table 5.2, GCEU stands for Google Compute Engine Units.

---

1http://aws.amazon.com/vpc
2http://aws.amazon.com/directconnect/
3http://www.stechno.net/sap-notes.html?view=sapnote&id=1656250
4http://aws.amazon.com/ec2/instance-types/
5https://developers.google.com/appengine/
### 5.3 Possible configurations

#### Detailed Description

<table>
<thead>
<tr>
<th>Instance size</th>
<th>Virtual cores</th>
<th>ECU*</th>
<th>Memory</th>
<th>Instance storage</th>
<th>I/O performance</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Standard</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Small</td>
<td>1</td>
<td>1</td>
<td>1.7 GB</td>
<td>160 GB</td>
<td>Moderate</td>
</tr>
<tr>
<td>Medium</td>
<td>1</td>
<td>2</td>
<td>3.75 GB</td>
<td>410 GB</td>
<td>Moderate</td>
</tr>
<tr>
<td>Large</td>
<td>2</td>
<td>2</td>
<td>7.5 GB</td>
<td>850 GB</td>
<td>High</td>
</tr>
<tr>
<td>Extra large</td>
<td>4</td>
<td>2</td>
<td>15 GB</td>
<td>1690 GB</td>
<td>High</td>
</tr>
<tr>
<td><strong>Micro</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Micro</td>
<td>?</td>
<td>up to 2</td>
<td>613 MB</td>
<td>None</td>
<td>Low</td>
</tr>
<tr>
<td><strong>High-memory</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Extra large</td>
<td>2</td>
<td>3.25</td>
<td>17.1 GB</td>
<td>420 GB</td>
<td>Moderate</td>
</tr>
<tr>
<td>XXL</td>
<td>4</td>
<td>3.25</td>
<td>34.2 GB</td>
<td>850 GB</td>
<td>High</td>
</tr>
<tr>
<td>XXXXXL</td>
<td>8</td>
<td>3.25</td>
<td>68.4 GB</td>
<td>1690 GB</td>
<td>High</td>
</tr>
<tr>
<td><strong>High-CPU</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Medium</td>
<td>2</td>
<td>2.5</td>
<td>1.7 GB</td>
<td>350 GB</td>
<td>Moderate</td>
</tr>
<tr>
<td>Extra large</td>
<td>8</td>
<td>2.5</td>
<td>7 GB</td>
<td>1690 GB</td>
<td>High</td>
</tr>
<tr>
<td><strong>Cluster Compute</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XXXXXL</td>
<td>2</td>
<td>16.75</td>
<td>23 GB</td>
<td>1690 GB</td>
<td>Very high</td>
</tr>
<tr>
<td>XXXXXXXXXL</td>
<td>2</td>
<td>44</td>
<td>60.5 GB</td>
<td>3370 GB</td>
<td>Very high</td>
</tr>
<tr>
<td><strong>Cluster GPU</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XXXXXL</td>
<td>2</td>
<td>16.75</td>
<td>22 GB</td>
<td>1690 GB</td>
<td>Very high</td>
</tr>
<tr>
<td><strong>High I/O</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XXXXXL</td>
<td>16</td>
<td>2.2</td>
<td>60.5 GB</td>
<td>2048 GB (SSD)</td>
<td>Very high</td>
</tr>
</tbody>
</table>

**Table 5.1: Available AWS instances**

<table>
<thead>
<tr>
<th>Instance size</th>
<th>Virtual cores</th>
<th>GCEU*</th>
<th>Memory</th>
<th>Instance storage</th>
<th>I/O performance</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>High-memory</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Extra large</td>
<td>2</td>
<td>2.75</td>
<td>13 GB</td>
<td>870 GB</td>
<td>?</td>
</tr>
<tr>
<td>XXL</td>
<td>4</td>
<td>2.75</td>
<td>26 GB</td>
<td>1770 GB</td>
<td>?</td>
</tr>
<tr>
<td>XXXXXL</td>
<td>8</td>
<td>2.75</td>
<td>52 GB</td>
<td>2x1770 GB</td>
<td>?</td>
</tr>
</tbody>
</table>

**Table 5.2: Available GCE instances**

### Windows Azure

At the moment of writing the Windows Azure cloud only offers virtual machines in a preview phase. This means that the prices are lower than normal and it is not clear when the service becomes widely available. Therefore the PaaS solution from Windows Azure is used in the price retrieve function, the cloud services. Including the preview version of the IaaS solution would give an unfair comparison with the other cloud providers. This can however easily be changed in the code when the virtual machines leave the preview phase.

A similar configuration with AWS when using Windows Azure would exist of Virtual Machines with Geo Redundant Storage (GRS) and Windows Azure Virtual Network (WAVN) which is the VPN. For a connection with the local environment the Windows Azure Connect and Traffic Manager can be used which is comparable with Amazon Direct Connect.

The available Windows Azure instance choices in the tool are shown in table 5.3.

* Values per virtual core

---


### Custom cloud

The custom cloud allows the user to define its own variables and formulas. This makes this cloud option very flexible. A disadvantage is that it takes more time to add an instance. There is no price retrieve function available. The standard variables are visible in table A.5 and the standard formulas in table B.7. The standard variables are based on the corresponding cost elements from the commercial public clouds used in this thesis, the same for the formulas.

The price calculation possibilities with the custom cloud are almost unlimited. An unrestricted amount of variables can be added to a custom cloud. The user is able to define its own formulas using the custom variables. Each formula defined and added by the user is shown as a separate expense in the cost distribution overview under the distribution tab.

#### 5.3.3 Converting configurations

After the user added a private cloud instance to the configuration a dialog box pops up with a public cloud suggestion. This suggestion is always provided with an AWS cloud instance. The reason is the same as why there are more configuration options for the price retrieve function for AWS. A great advantage of AWS is that they provided recommended configurations with the belonging SAPS. These SAPS are needed in order to be able to convert the private cloud instances to public cloud instances. The public cloud suggestion provided by the tool after adding a private cloud instance can of course be changed to another cloud provider. The classes and sizes of instances are comparable between the cloud providers.

The conversion table is based on a measurement used by Capgemini. Each 3 GB of memory is equal to 500 SAPS.

The conversion as used by the tool is as follows:

<table>
<thead>
<tr>
<th>Instance size</th>
<th>Memory</th>
<th>SAPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Micro</td>
<td>630 MB</td>
<td>0 - 100</td>
</tr>
<tr>
<td>Standard</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Small</td>
<td>1.7 GB</td>
<td>100 - 275</td>
</tr>
<tr>
<td>Medium</td>
<td>3.75 GB</td>
<td>275 - 600</td>
</tr>
<tr>
<td>Large</td>
<td>7.5 GB</td>
<td>600 - 1250</td>
</tr>
<tr>
<td>Extra large</td>
<td>15 GB</td>
<td>1250 - 2500</td>
</tr>
<tr>
<td>High-memory</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Extra large</td>
<td>17.1 GB</td>
<td>2500 - 2875</td>
</tr>
<tr>
<td>Double extra large</td>
<td>34.2 GB</td>
<td>2875 - 5750</td>
</tr>
<tr>
<td>Quadruple extra large</td>
<td>68.4 GB</td>
<td>5750 - 10000</td>
</tr>
</tbody>
</table>

| Table 5.4: SAPS conversion table |
The SAPS range values for each configuration are just an approximation. For a really good conversion SAP developed a complete sizing tool called "Quick Sizer" which can be used. The values provided by SAP were 3700 SAPS for the Double extra large high-memory instance and 7400 SAPS for the Quadruple extra large high-memory instance. These values fit perfectly in the table.

5.3.4 Defining the SAP application location

The SAP application location is decided based on several variables. The most important one is the privacy level. The user can select a privacy level for each application. Based on this level the tool decides if the application can be stored in the cloud or not. The boundary of this level can be defined by the user in the global options. If the user selects a boundary of "high" this means that all SAP applications with a privacy level of "high" and stricter must be stored in the local storage.

Another important variable is the restriction to the data location provided by the user. If the user defined that the SAP application has to be stored in Europe and the cloud instance related to the SAP application is placed in America this cloud instance will not be used. If the user defined a location nearer than Europe, the application shall not be stored into the cloud.

The last restriction is based on the storage costs. When all other restrictions are no problem for storage into the cloud, the storage costs decide where the application will be stored.

5.4 Summary

The tool covers a lot of variables and options. All these variables can be defined by the user. The variables for a storage type, like the private cloud, traditional co-location etc. can be provided in the global options properties widget on the left of the tool. Not all the variables have to be provided, only the ones needed. The costs are divided in instance costs, data costs and configuration costs. The instance costs are calculated per instance, the data costs per SAP application and the configuration costs per storage type, as already explained in the first paragraph.

The cloud options are different per cloud provider. The most options are provided for AWS because of the support by SAP. Than GCE has the most support because of their high-memory instances. The least support is there for Windows Azure because they do not meet the requirements for most SAP systems. Extending the variables for the local and cloud instances is easy. Every programmer can add a new variable to the local and cloud objects and use this in the formulas. Even easier is to add a custom cloud which allows the user to define all the variables and formulas. The SAP application has properties for data size and traffic. The other properties are to define the location, like privacy and the storage location. Each SAP application can be linked to a local and cloud instance. All the instances and the SAP application can be assigned a system type. This type is used to match instances with SAP applications.

The cost functions for the traditional co-location are based on all kinds of hardware related costs. The private cloud costs functions are based on SAPS, transition and infrastructure costs. The costs for the public cloud are defined by the cloud providers and mostly exist out of instance and data costs. SAP has recommended some configurations for the AWS cloud but there are many other configurations available in the tool for AWS. For the Google Cloud only the high-memory instances are used because these are, according to SAP the most suitable. For Windows Azure cloud services all the options are available, although the specifications are not very impressive. The configuration options for the custom cloud can completely be decided by the user.

Private cloud instances can automatically be converted to cloud instances based on the amounts of SAPS. Each 3 GB of memory can be defined as 500 SAPS. This conversion is just a suggestion, the user is completely free to change the cloud instance to its own flavor.

---

Chapter 6

Evaluation

One of our goals was to develop a user-friendly interface which can easily be used by the target group. We do not know if this goal is reached until we test the tool with the target group. The technique for evaluating the design of the user interface we used is the cognitive walkthrough. This test method pays special attention to how well the interface supports "explanatory learning". The user should be able to use the tool for the first time without formal training (Rieman et al., 1995). Another important issue is that the tool should have a small learning curve. We used the cognitive walkthrough in combination with user tests.

6.1 The first iteration

The tests can be done in several iterations. We already did the first iteration during the development of the tool. We used the tool to add a large configuration and during each action we asked ourself if it was clear for the user and if we could do something to make it easier for the user. This resulted in several improvements. We added information fields in several parts of the decision support tool to inform the user, so they would get appropriate feedback. We added a convert function from a private cloud instance to a public cloud instance, and the other way around. To reduce the amount of information for the user to complete we added a live price retrieve function for the public cloud instances, which already completes the costs fields for the public cloud instances. For the SAP applications we already predefined some standard values, the ones that are most often used. This prevents the user from providing the same information over and over again.

6.2 User tests

For the user tests based on the cognitive walkthrough technique we needed the following information:

- Identification of the users
- Tasks for evaluation
- Description or implementation of the interface
- Action sequences (scenarios) for completing the tasks

Each user was asked to provide his name and function. Then ten tasks had to be done which cover all the important functionalities of the tool. The happy path is added to all the questions, this path was not provided to the user because of the influence on their actions. The tasks were the following:

1. Complete the required options in the configuration options.
2. Add a private cloud instance and an Amazon cloud instance and link these with a new SAP application.
3. Save the configuration, re-open the application and load the saved configuration.
4. Add a new SAP application with a traditional co-location instance and a custom cloud.
5. Add a new variable and a new instance formula to the custom cloud.
6. Open the distribution tab and calculate the location.
7. Edit the Amazon cloud instance.
8. Drag a SAP application from local to cloud, or the other way around.
9. Take a look at the costs distribution of a storage area (local or cloud).
10. Export the configuration to an excel sheet.

The tasks include several subtasks. The happy path for each subpath is the following:

- Add a private cloud instance
  1. Enter a name in the textfield under local instances
  2. Press "Add" or the enter key
  3. Select as method "Private Cloud"
  4. Select a system type
  5. Select a storage tier
  6. Enter a value for Non-DB storage
  7. Enter a value for Non-DB backup storage
  8. Enter a value for the SAPS
  9. Click the "OK" button

  After action 8 above the user can check the checkbox for "Convert to cloud" in that case the user only has to click the "OK" button in the next action sequence.

- Add an Amazon Cloud instance
  1. Enter a name in the textfield under cloud instances
  2. Press "Add" or the enter key
  3. Select under provider "Amazon Web Services"
  4. Select a system type
  5. Select a location
  6. Select a type
  7. Select all the active comboboxes following a former selection
  8. Click the "OK" button
  9. Enter the values for the desirable quantities in the properties widget

  The prices are completed by using the dialog but the quantities still have to be entered by the user after the pricing information is retrieved.

- Add a SAP application
  1. Enter a name in the textfield under SAP applications
  2. Press "Add" or the enter key
  3. Enter a value in the size of data field
  4. Enter a value in the data transfer field
  5. Change the values for privacy, storage location and system type when desirable
  6. Select the local instance
  7. Select the cloud instance

  The cloud instances are automatically linked if the SAP application name matched the instance names, in that case step 6 and 7 can be left out.

- Save a configuration
1. Click "File" in the menu
2. Click in the menu on "Save" or "Save as"
3. Enter a name in the dialog
4. Press "Save"

• Load a saved configuration
  1. Click "File" in the menu
  2. Click in the menu on "Open"
  3. Browse the dialog for the saved file
  4. Press "Open"

• Add a traditional co-location instance
  1. Enter a name in the textfield under local instances
  2. Press "Add" or the enter key
  3. Select as method "Private Cloud"
  4. Select a system type
  5. Select a storage tier
  6. Enter a value for Non-DB storage
  7. Enter a value for Non-DB backup storage
  8. Enter a value for the hardware costs
  9. Enter a value for the rack space
  10. Enter a value for the years
  11. Click the "OK" button

• Add a custom cloud instance
  1. Enter a name in the textfield under cloud instances
  2. Press "Add" or the enter key
  3. Select under provider "Custom Cloud"
  4. Select a system type
  5. Click the "OK" button
  6. Enter the values for the desirable quantities in the properties widget

• Add a new variable to the custom cloud
  1. Click "Collection" next to "Custom variables" in the properties widget
  2. Click "Add"
  3. Enter a value next to the "Name" field
  4. Enter a value next to the "Value" field
  5. Click the "OK" button

• Add a new instance formula
  1. Select "Add instance formulas" next to "Add formulas" in the properties widget
  2. Enter a value for "Name"
  3. Enter a formula using the "Available variables" or by providing one manually in the formula field
  4. Press "Add"
  5. Click the "OK" button

• Calculate the distribution
  1. Click on the "Distribution" tab
  2. Press "Calculate location"
6.2 User tests

Evaluation

- Edit the Amazon cloud instance
  1. Right mouse-click on the amazon cloud instance under cloud instances
  2. Select "Edit" from the pop-up menu
  3. Change some values in the dialog
  4. Click the "OK" button

- Drag a SAP application from local to cloud
  1. Click on the "Distribution" tab
  2. Click with a mouse-button on the SAP application object
  3. Hold the mouse-button and move the mouse to the other area
  4. Release the mouse-button

- Take a look at the costs distribution of a storage area
  1. Move the mouse over the costs related to the area

During the execution of the tasks we observed the user. We made notes of the actions of the user and looked whether these were consistent with the intended actions. After the tasks were completed we asked the user to fill out a small questionnaire. This questionnaire is linked to the System Usability Scale (SUS) which is a simple ten-item scale giving a global view of subjective assessments of usability. The questions in the test itself are very generic, therefore it is just an addition to the user tests we have held. The SUS is a Likert scale (Brooke, 1996). In our test we had to do with typical five-level Likert items from strongly disagree to strongly agree. The SUS test can result in a score. Each item can have a score between 0 and 4. For the odd numbered items this is the scale position minus 1. For the even numbered items this is 5 minus the scale position. The sum of all the scores have to be multiplied by 2.5 to obtain the overall value of software usability. The scores have a range between 0 and 100.

After each test we gathered the remarks made by the user and when necessary we directly made changes to the tool. A difference was made between personal problems with the user interface and parts of the tool that where unclear because of a lack of information. Each issue was investigated for improvement. After each user test we discuss the changes made.

The users used were people from different departments of SAP but all in a function were they could use such a tool as we made. The input of some of these test users was used during the development of our tool. The tool is not meant for unexperienced users but for users who work in especially the SAP architecture area. The users we used for testing had enough knowledge in this area.

6.2.1 User test 1

Identification of the user

Function: solutions architect / sales support at Capgemini

Interface

We have a complete tool with working GUI.

Task 1: Add a configuration to the decision support tool

- What’s the user’s goal, and why?
  - The user wants to add a configuration to the decision support tool.
Evaluation 6.2 User tests

- After a configuration is added a user can see the distribution in the distribution screen.

- Is the action obviously available?
  - No, the user did not complete the configuration values by himself and thought that only a cloud instance or only a local instance where needed.
  - The "convert to cloud" function was not used, and therefore also not obviously available.

- Does the action or label match the goal?
  - The name of the properties widget is "Configuration values" and the properties widget is categorized by labels defining the relation of the requested variables.
  - The user knew without hesitation where to complete the information, he had however problems with a lack of unit indication.
  - The "Add" button after the name field for the local instance, cloud instance and the SAP application was clear enough for adding these three items.

- Is there good feedback?
  - The user had difficulties to figure out what values had to be entered in the designated fields. After notifying the user on the information fields available in all the steps each time a value field is clicked, it became more clear.
  - Information is available in almost all the steps in the tool. Although the information field is clearly visible, the user still had problems using it.

The user started taking a close look at the tool. After a short search he found the local instances column and used it to add a new private cloud. The user had the possibility to select the "convert to cloud" option which makes it unnecessary to add a new cloud instance by hand. The user did not even look at this option and clicked "OK". The system type was not selected before clicking "OK". Although this is not mandatory, it is still needed in a later stage. After adding a cloud the user had trouble with the units for the quantities necessary to complete the information for the cloud. This user was not familiar with all the needed variables. Normally he asks colleagues for these values. When entering the values he used the up and down arrows next to a value field. These values increased with 1, even when cents were requested. This caused some confusion about the required values.

**Task 2: Load a saved configuration**

- What’s the user’s goal, and why?
  - Load an earlier saved configuration.
  - The user has to add multiple configurations, therefore it is a good idea to know how to save and load them.

- Is the action obviously available?
  - Yes, the user intentionally clicked the "File" option in the menu and selects "Save", reopened the program and selected "Open" from the "File" menu.

- Does the action or label match the goal?
  - Yes, the labels say "Save", "Save as" and "Open" which is exactly what the user wants.
  - These labels were very clear for the user.

- Is there good feedback?
  - After selecting "Save" the user is required to provide a name in the filedialog. After saving this is mentioned in the status bar.
  - The user was very confident doing this procedure.

This task is the same for almost all programs. Therefore there were no problems at all.
6.2 User tests

Evaluation

Task 3: Add a second SAP application configuration

The user had to add a traditional co-location instance and a custom cloud. Another configuration with the same procedure as in task 1. Because of the similarity with task 1 no cognitive walkthrough was performed. This task was added to test the learnability of the user. Adding a new configuration was no problem at the end. The user had problems figuring out where to add a traditional co-location instance. After a while he found it under local instances. Adding a custom cloud was no problem, although the user was a bit confused that no options could be selected besides the system type.

Task 4: Add a new variable and a new instance formula to the custom cloud

- What’s the user’s goal, and why?
  - Add a new variable and a new instance to the custom cloud instance.
  - Use the flexibility of the custom cloud.

- Is the action obviously available?
  - The action is visible when the user has selected the custom cloud instance.
  - The user had no problems finding the options to open the dialog for a new variable and a new formula.
  - An info text helps the user with the expected content. The formula editor was hard for the user to understand.

- Does the action or label match the goal?
  - The labels are "Add formulas" and "Custom variables"
  - The labels in the variable dialog are "Add" which matches the action for adding a new variable and "Name" and "Value" indicate where to complete the values.
  - The user was able to add a new variable.
  - The labels in the formula editor are all matching the actions. It was hard however for the user to use the editor because of the formula structure.

- Is there good feedback?
  - The user is told during the whole process what kind of information to provide. Except when entering a formula, the formula field had no label informing the user that it was a formula field.

Adding a variable and a formula to the custom cloud are more advanced tasks. This was very obvious with this user. Adding a variable went fine, but adding a formula took some time. The user was overwhelmed by the editor and did not know what to do. After some time of trying and observing the dialog, the user was able to add a new formula.

Task 5: View the data distribution

- What’s the user’s goal, and why?
  - View the distribution of SAP applications.
  - See what costs are involved per SAP application in the local and cloud area.

- Is the action obviously available?
  - There is a distribution tab but it is not directly clear to a user that it is linked to the configuration.

- Does the action or label match the goal?
  - The tab is called "Distribution" and under that tab there is a button "Calculate location".

- Is there good feedback?
No separate feedback is needed for this task.

This was an easy task and the user had no problems with it.

**Task 6: Edit a cloud instance**

- What’s the user’s goal, and why?
  - Change the configuration of a cloud instance.
  - The user made a mistake or wants to change an instance for other reasons.

- Is the action obviously available?
  - Users are familiar with changing items by clicking with the right-mouse button on an item.
  - The user clicked next to the item and could not open the menu, this was a fault in the tool.

- Does the action or label match the goal?
  - The label in the pop-up menu is "Edit".

- Is there good feedback?
  - The edit menu is opened.

The user had to be helped because of a fault in the decision support tool. The action provided by the user was fine but had not the intended effect.

**Task 7: Move an application to another area, and check the costs distribution**

- What’s the user’s goal, and why?
  - Move a SAP application to the cloud or to the local area.
  - See the impact on the costs.

- Is the action obviously available?
  - There are two areas with graphical data blocks. The user had no trouble "grabbing" a block with the mouse and moving it.
  - The action of moving objects is presented in lots of applications and is no problem for the user.
  - The costs distribution is not obviously available. It seems more like an easter egg.

- Does the action or label match the goal?
  - Yes, the user can drag an application to another area.
  - No, it is not clear how to see the costs distribution without accidentally moving the mouse over the total costs.

- Is there good feedback?
  - Moving a block to another area is clearly visible because an object is attached to the mouse and the source block has a lower opacity.
  - When the costs distribution is found it is clearly visible due to a sliding window.

Moving the application was no problem for the user. He was impressed by the possibility to drag the objects to other areas and the changing costs. He played around a while with the blocks before going on with the next task. The costs distribution was already opened by accident so the user knew where to find it. It was however not visible otherwise.
Task 8: Export a configuration to excel

- What’s the user’s goal, and why?
  - Create an excel sheet.
  - To use for other purposes.

- Is the action obviously available?
  - It is available in the menu where most of such actions are available in applications.
  - An export function is not uncommon in tools like this.

- Does the action or label match the goal?
  - Yes, the label is "Export" under the "File" menu.

- Is there good feedback?
  - The user gets a filedialog where he has to insert a name. After clicking "save" a message is shown in the status bar.

The user had no problems exporting the configuration to an excel file. It seemed like he had done this before.

**Software Usability Scale**

<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. I think that I would like to use this system frequently</td>
<td>agree</td>
<td>3</td>
</tr>
<tr>
<td>2. I found the system unnecessarily complex</td>
<td>disagree</td>
<td>3</td>
</tr>
<tr>
<td>3. I thought the system was easy to use</td>
<td>agree</td>
<td>3</td>
</tr>
<tr>
<td>4. I think that I would need the support of a technical person to be able to use this system</td>
<td>strongly disagree</td>
<td>4</td>
</tr>
<tr>
<td>5. I found the various functions in this system well integrated</td>
<td>neutral</td>
<td>2</td>
</tr>
<tr>
<td>6. I thought there was too much inconsistency in this system</td>
<td>strongly disagree</td>
<td>4</td>
</tr>
<tr>
<td>7. I would imagine that most people would learn to use this system very quickly</td>
<td>strongly agree</td>
<td>4</td>
</tr>
<tr>
<td>8. I found the system very cumbersome to use</td>
<td>strongly disagree</td>
<td>4</td>
</tr>
<tr>
<td>9. I felt very confident using the system</td>
<td>neutral</td>
<td>2</td>
</tr>
<tr>
<td>10. I needed to learn a lot of things before I could get going with this system</td>
<td>disagree</td>
<td>3</td>
</tr>
</tbody>
</table>

Total of 32 points: \(32 \times 2.5 = 80\)

*Table 6.1: SUS, user test 1*

**Changes according to this test**

According to the user feedback the following changes were made to the decision support tool.

- The "convert to cloud" option when adding a private cloud instance is now automatically checked.
- The local cloud dialog is checked for forgotten values. When a value is not provided, a dialog box informs the user that not all values are provided.
- The instance hours under a cloud instance were not clear about the term. Text is added to notify the user that the hours are per month.
- An item can now be clicked on with the right mouse-button over the whole width of the listbox, this was only possible on the name.
Units are added to the info texts.

The order of the actions was not completely clear to the user, therefore the tool is extended with a 4-steps procedure. Above each column is an info field with the step and action.

Added a watermark text in the formula field, to make clear it is a formula field.

Added arrows to the top of the costs fields in the distribution area to make clear that it can slide up.

6.2.2 User test 2

Identification of the user

Function: SAP architect at Capgemini

Interface

We have a complete tool with working GUI.

Task 1: Add a configuration to the decision support tool

What’s the user’s goal, and why?

– The user wants to add a configuration to the decision support tool.
– After a configuration is added a user can see the distribution in the distribution screen.

Is the action obviously available?

– Due to the info boxes with the steps above each column in the GUI, the user knew which actions to take.
– The user knew that all data in the configuration had to be completed for the goal.

Does the action or label match the goal?

– The labels and actions were all clear to the user

Is there good feedback?

– The info boxes at the top of each column tell the user where to start and end.
– It was not clear for the user if cents or euros were needed in the costs fields.

The user started adding all the values in the configuration values column. He had some problems with completing the values for the traditional co-location because that is not available inside Capgemini. The privacy boundary value in the configuration values column was not very clear, he thought that the selected value was also included in the cloud. Just like the former user, all the values for the years of amortization were set on 3, which is very common. The user had some confusion with the OS Backup storage field in the local dialog because there is also a Non-DB storage field which seemed the same for the user. When adding the private cloud, the cloud instance dialog automatically popped up which surprised the user. He only had to press the "OK" button in order to complete the cloud instance step. Completing the information for a SAP application was no problem, the user was familiar with the values needed.

Task 2: Load a saved configuration

What’s the user’s goal, and why?

– Load an earlier saved configuration.
– The user has to add multiple configurations, therefore it is a good idea to know how to save and load them.

Is the action obviously available?
6.2 User tests

Evaluation

– Yes, the user intentionally clicked the "File" option in the menu and selects "Save", reopened the program and selected "Open" from the "File" menu.

• Does the action or label match the goal?
  – The labels were very clear for the user.

• Is there good feedback?
  – The user was very confident doing this procedure and got enough feedback.

During the test the user constantly saved the changes he made. The user already assumed there was a save function available.

Task 3: Add a second SAP application configuration

The user had to add a traditional co-location instance and a custom cloud. It was not directly clear for the user that the traditional co-location had to be added in the local instances column. After a while he figured it out and completed the required values. When adding a custom cloud the user wanted to complete the other values which were inactive. He tried clicking the "Info" button but that had not the desired effect. By a lack of other options he clicked the "OK" button. The user continued completing the values for the custom cloud in the properties widget because this was also required when an Amazon cloud instance was added.

Task 4: Add a new variable and a new instance formula to the custom cloud

• What’s the user’s goal, and why?
  – Add a new variable and a new instance to the custom cloud instance.
  – Use the flexibility of the custom cloud.

• Is the action obviously available?
  – The user had no problems finding the options to open the dialog for a new variable and a new formula.

• Does the action or label match the goal?
  – The user had no problems performing the actions and figured it out by reading the labels and looking for available actions.

• Is there good feedback?
  – The formula field was indicated this time and the the user knew how to use the info fields to inform him.

This was a more advanced user working on such configurations all the time. The formula editor did overwhelm the user a bit. But after some trying and clicking he knew how to use it.

Task 5: View the data distribution

• What’s the user’s goal, and why?
  – View the distribution of SAP applications.
  – See what costs are involved per SAP application in the local and cloud area.

• Is the action obviously available?
  – The user almost instantly clicked the distribution tab and the "calculate costs" button.
  – He knew the purpose of the tool and what kind of information was generated by the tool.

• Does the action or label match the goal?
– This was all clear for the user.

- Is there good feedback?
  – The actions do what they have to do, and it is clear to the user.

This was an easy task and the user had no problems with it.

**Task 6: Edit a cloud instance**

- What’s the user’s goal, and why?
  – Change the configuration of a cloud instance.
  – The user made a mistake or wants to change an instance for other reasons.

- Is the action obviously available?
  – The user searched for a short time for a visible edit function and then clicked the right mouse-button on an item.

- Does the action or label match the goal?
  – The user did the correct actions reading the options.

- Is there good feedback?
  – The actions had the intended effect.

The user already used the edit function during the other tasks. He assumed there was an edit function and searched for it.

**Task 7: Move an application to another area, and check the costs distribution**

- What’s the user’s goal, and why?
  – Move a SAP application to the cloud or to the local area.
  – See the impact on the costs.

- Is the action obviously available?
  – The graphical data blocks in an area gave the user already the idea that they could be moved.

- Does the action or label match the goal?
  – Moving an application was clear for the user and the same for finding the costs distribution.

- Is there good feedback?
  – The user had no trouble with the actions and got the required feedback.

The user was confused that the configuration costs were shown in the local cloud when there were no applications in that area. This was a fault in the application. Dragging the applications in the distribution screen had the same fun factor as with the first user. In the distribution tab there is also a graph with a ratio of costs between the local and the cloud area. This was not very clear to the user because the scale changed with the ratio.

**Task 8: Export a configuration to excel**

Using the export function was no hard task for the user. It was the same kind of action as saving a configuration. The user clicked on "Exit" the first time but that was more a mistake than a design flaw.
6.2 User tests

**Evaluation**

### Software Usability Scale

<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. I think that I would like to use this system frequently</td>
<td>agree</td>
<td>3</td>
</tr>
<tr>
<td>2. I found the system unnecessarily complex</td>
<td>strongly disagree</td>
<td>4</td>
</tr>
<tr>
<td>3. I thought the system was easy to use</td>
<td>agree</td>
<td>3</td>
</tr>
<tr>
<td>4. I think that I would need the support of a technical person to</td>
<td>disagree</td>
<td>3</td>
</tr>
<tr>
<td>be able to use this system</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5. I found the various functions in this system well integrated</td>
<td>agree</td>
<td>3</td>
</tr>
<tr>
<td>6. I thought there was too much inconsistency in this system</td>
<td>strongly disagree</td>
<td>4</td>
</tr>
<tr>
<td>7. I would imagine that most people would learn to use this system very</td>
<td>neutral</td>
<td>2</td>
</tr>
<tr>
<td>quickly</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8. I found the system very cumbersome to use</td>
<td>disagree</td>
<td>3</td>
</tr>
<tr>
<td>9. I felt very confident using the system</td>
<td>agree</td>
<td>3</td>
</tr>
<tr>
<td>10. I needed to learn a lot of things before I could get going with this</td>
<td>disagree</td>
<td>3</td>
</tr>
<tr>
<td>system</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Total of 31 points: $31 \times 2.5 = 77.5$

**Table 6.2: SUS, user test 2**

### Changes according to this test

According to the user feedback the following changes were made to the decision support tool.

- All the costs fields in the decision support tool have gotten an euro mark in the textfield with a clear 0,00 format to show the use of decimals.
- All the percentage fields in the configuration widget have gotten a percentage mark in the textfield.
- The increment for the costs values is decreased to 0,01 per step instead of 1,00.
- The privacy boundary value now includes the selected value instead of all the values higher than the selected value.
- A new info field is added to the SAP application column to provide the user more information.
- All the years of amortization in the tool are set to 3 by default.
- The OS Backup storage label is changed to Non-DB backup storage.
- The info button in the cloud dialog now shows a dialog with info when a custom cloud is selected.
- The fault in the application that the configurations costs were calculated for local instances without applications in that area is solved.
- The scale for the graph in the distribution area on the y-axis is fixed from 0 to 100.

### 6.2.3 User test 3

#### Identification of the user

Function: Functional product owner application hosting

#### Interface

We have a complete tool with working GUI.
Task 1: Add a configuration to the decision support tool

- What’s the user’s goal, and why?
  - The user wants to add a configuration to the decision support tool.
  - After a configuration is added a user can see the distribution in the distribution screen.

- Is the action obviously available?
  - The user wanted to add a new instance and clicked on the "Add" button without providing a name in the name field, this resulted in no action at all.
  - The user solved the issue by providing the name.

- Does the action or label match the goal?
  - The user knew where to click and what to do by reading the available information.

- Is there good feedback?
  - This user had no problems with the feedback.

The screen resolution for this user was too low to see the bottom of the values in the SAP application column. The resolution had to be changed in order to use these values. Pressing the "Add" button had not the intended result, nothing happened and the user clicked several times to know for sure. After providing a name the "Add" button did work. When converting the private cloud to an Amazon cloud instance the user notices that the non-db storage had to be completed again in the cloud, although he had already provided them for the local instance. The costs could be negative, which is not really a problem but it is not very common in configurations.

Task 2: Load a saved configuration

The result was the same as with the earlier two users. The save options was found instantly and loading the configuration was no problem at all.

Task 3: Add a second SAP application configuration

Adding a traditional co-location instance was no problem, the years of amortization were already completed which saved the user two values to complete. The custom cloud was found under the cloud instances and although the user tried all the disabled options, he ended clicking the "OK" button. The prices were directly completed by the user after the custom cloud was added. A new application was added and the user had no problem linking the newly created instances to the SAP application.

Task 4: Add a new variable and a new instance formula to the custom cloud

- What’s the user’s goal, and why?
  - Add a new variable and a new instance to the custom cloud instance.
  - Use the flexibility of the custom cloud.

- Is the action obviously available?
  - The custom cloud does not provide any options in the cloud dialog and the values in the properties widget are standard values. The custom cloud name indicates that there is a possibility to customize the values.
  - The user did not know how to add a new formula.

- Does the action or label match the goal?
  - The labels were not clear enough for the user.

- Is there good feedback?
6.2 User tests

Evaluation

- The user noticed the feedback.

The formula editor was not really user friendly according to this user. The user was able to add a variable, adding a formula however was too complex. The user expected a drag and drop function to create formulas. Selecting variables and operators was not user-friendly enough. After a small explanation the user was able to add a formula.

Task 5: View the data distribution
- What’s the user’s goal, and why?
  - View the distribution of SAP applications.
  - See what costs are involved per SAP application in the local and cloud area.
- Is the action obviously available?
  - The user expected the function.
- Does the action or label match the goal?
  - The user expected a button on the configuration tab to achieve this goal.
- Is there good feedback?
  - Once clicking the distribution tab and the calculate button the feedback was good.

The user wanted a better integration between the configuration tab and the distribution tab. Adding a configuration did not directly indicate that there was a distribution tab. When he entered all the information he expected a button to calculate the location in the calculation tab or some kind of confirm button.

Task 6: Edit a cloud instance
- What’s the user’s goal, and why?
  - Change the configuration of a cloud instance.
  - The user made a mistake or wants to change an instance for other reasons.
- Is the action obviously available?
  - Yes, the user wanted to edit a cloud instance so he directly opened the edit menu by clicking the right mouse-button on an item.
- Does the action or label match the goal?
  - The user did the correct actions reading the options.
- Is there good feedback?
  - The actions had the intended effect.

The edit function gave no further problems.

Task 7: Move an application to another area, and check the costs distribution
- What’s the user’s goal, and why?
  - Move a SAP application to the cloud or to the local area.
  - See the impact on the costs.
- Is the action obviously available?
  - The user got the idea that the SAP applications could be moved.
- Does the action or label match the goal?
— Dragging a SAP application to another area had the effect that the costs are re-calculated. 
The goal and the action was clear.

• Is there good feedback?
  — The user had no trouble with the actions and got the required feedback.

Again the user had fun playing with the graphical SAP applications. It was not directly clear to the user what the local and the cloud area were. He would like to see that better indicated. Another suggestion was to have the possibility to change the areas depending on the goal the user has. It could be that a user want to compare the Amazon cloud and the Azure cloud, changing the left side to an Amazon cloud and the right side to an Azure cloud would be welcome. Another suggestion was to divide an area in separate areas so that the difference between the different types of local and cloud instances were more clear. The user had the intention to drag SAP application outside of the distribution area. This resulted in a glitch which forced the user to first click another application, or to recalculate the costs. The application itself became inactive due to that action.

**Task 8: Export a configuration to excel**

Using the export function was no hard task for the user. It was the same kind of action as saving a configuration. The user wanted to see more information in the excel sheet, but the function itself was no problem.

**Software Usability Scale**

<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. I think that I would like to use this system frequently</td>
<td>agree</td>
<td>3</td>
</tr>
<tr>
<td>2. I found the system unnecessarily complex</td>
<td>disagree</td>
<td>3</td>
</tr>
<tr>
<td>3. I thought the system was easy to use</td>
<td>agree</td>
<td>3</td>
</tr>
<tr>
<td>4. I think that I would need the support of a technical person to be able to use this system</td>
<td>disagree</td>
<td>3</td>
</tr>
<tr>
<td>5. I found the various functions in this system well integrated</td>
<td>neutral</td>
<td>2</td>
</tr>
<tr>
<td>6. I thought there was too much inconsistency in this system</td>
<td>neutral</td>
<td>2</td>
</tr>
<tr>
<td>7. I would imagine that most people would learn to use this system very quickly</td>
<td>agree</td>
<td>3</td>
</tr>
<tr>
<td>8. I found the system very cumbersome to use</td>
<td>disagree</td>
<td>3</td>
</tr>
<tr>
<td>9. I felt very confident using the system</td>
<td>neutral</td>
<td>2</td>
</tr>
<tr>
<td>10. I needed to learn a lot of things before I could get going with this system</td>
<td>strongly disagree</td>
<td>4</td>
</tr>
</tbody>
</table>

Total of 28 points: $28 \times 2.5 = 70$

**Table 6.3: SUS, user test 3**

**Changes according to this test**

According to the user feedback the following changes were made to the decision support tool.

• A scroll option is added to the SAP application column which enables the user to use a lower screen resolution.
• It is now possible to use the "Add" button for adding a local and a cloud instance without providing a name first, the name can be entered in the dialog.
• The non-DB backup storage and non-DB storage values are now copied to the cloud instance when using the "convert to cloud" function.
6.3 Results

- The prices in the cloud are now limited to 0, and can not be lower.
- Dragging a SAP application outside the distribution area is no problem anymore, the application remains in the same location as before the drag event.
- The difference between the local and cloud area is improved by adding labels at the top of the cloud areas, and not only at the bottom.
- The gradient in the storage areas is removed so that the areas are better visible. The gradient had the effect that it was not clear where an area ended.
- A button is added to the configuration screen with a label "Calculate" which calculated the distribution and shows the content of the distribution tab to the user.

6.3 Results

<table>
<thead>
<tr>
<th>Task</th>
<th>User 1</th>
<th>User 2</th>
<th>User 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Add a configuration to the decision support tool</td>
<td>FAIL</td>
<td>FAIL</td>
<td>OK</td>
</tr>
<tr>
<td>2. Load a saved configuration</td>
<td>OK</td>
<td>OK</td>
<td>OK</td>
</tr>
<tr>
<td>3. Add a second SAP application configuration</td>
<td>OK</td>
<td>FAIL</td>
<td>OK</td>
</tr>
<tr>
<td>4. Add a new variable and a new instance formula to the custom cloud</td>
<td>OK</td>
<td>OK</td>
<td>FAIL</td>
</tr>
<tr>
<td>5. View the data distribution</td>
<td>OK</td>
<td>OK</td>
<td>OK</td>
</tr>
<tr>
<td>6. Edit a cloud instance</td>
<td>FAIL</td>
<td>OK</td>
<td>OK</td>
</tr>
<tr>
<td>7. Move an application to another area, and check the costs distribu-</td>
<td>FAIL</td>
<td>OK</td>
<td>OK</td>
</tr>
<tr>
<td>tion</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8. Export a configuration to excel</td>
<td>OK</td>
<td>OK</td>
<td>OK</td>
</tr>
</tbody>
</table>

Table 6.4: Results of the cognitive walkthrough

The tests went better with each user. Doing the test with the first user resulted in a total of 3 failures as shown in table 6.4. The user did not know exactly what information to complete and the order of the steps to take was not clear. Editing a cloud instance failed because of a design fault which forced the user to click the right mouse-button on the name instead of the complete list item. Checking the costs distribution was not clear, there was no marker to tell the user about the hidden costs distribution. The problems causing the failures were fixed and we went on to the second user.

The second user had also problems with adding a new configuration. The information was adjusted but still it was not clear what was expected for the costs. The user did not know if the values had to be cents or euro’s. The user chose cents where euro’s were expected. Adding a second SAP application was not really a problem but the user had trouble finding the traditional co-location instance in the beginning. He was confused by the use of a traditional co-location and a private cloud as local instances. The expected values for the costs were clarified and some useful suggestions by the user were processed.

The third user had no trouble with adding a configuration. The units were clear now. This user had a problem with customizing the cloud. The other users had to get used to it as well but this third user actually wanted to stop doing the task. It was not clear enough for him to continue. He expected a drag and drop interface and not a more complex combination of selectboxes and operators.

An overview of the System Usability Scale tests taken is shown in table 6.5. The higher the score in the table, the better. Stronger points are the complexity, consistency, the difficulty and the necessary knowledge before being able to use the tool. All these stronger points are very important for the usability of the tool, which is one of the main goals. Less strong points were the
integration of the functions and the confidence the users had using the tool. The tool contains lots of options. Some options were added in a later stage of the development to the tool. These options were suggested by the experts we spoke with. These options could be better integrated in a future version of the tool. The reason for the confidence people had using the tool can easily be explained by the fact that this was the first time the test users used the tool. The large amount of textfields the user can complete overwhelmed them a bit. Although these points were less stronger this does not mean that they were weak. Almost all the scores in the table were higher than neutral. The times a user chose neutral, most of the times this was because the user did not know what to answer.

All the users learned very quickly and they liked working with the tool. The average SUS score provided by the users was 75.8 on a scale of 0 to 100.

### 6.4 Experts

During the development of the tool we used ten experts with different specialties. People with knowledge of the SAP architecture, SAP applications, SAP development, SAP maintenance, data management and more. All the users used were people with several years of experience with SAP. The development of the tool was an incremental process in which the knowledge of these experts was very useful. The knowledge of the experts sometimes overlapped with each other but there were no contradictions. The suggestions of the experts were consistent with each other.

Not all the suggestions from the experts were possible to implement due to a lack of possibilities. An example is the separation of SAP data tables over different servers. This was according to another expert, very familiar in that area, not possible because of limitations in SAP. Another suggestions which was not used was the amount of use of several data sets. A dataset that is used less often or is not used for a longer period can be stored on cheaper storage devices. This feature is however to complex to use for several reasons. The most important one is the lack of the possibility of separating data tables in SAP.
In this chapter we look back at the complete thesis and evaluate if we reached our project goals. The answers to the questions asked in the project goals are given in the conclusions. In the last section of this chapter we provide some recommendations for future work.

### 7.1 Conclusions

Our task was to develop a tool which can help the user with distributing SAP application data between an on-premise location and an on-demand location. We defined the on-premise location as a traditional co-location and a private cloud, which can also be stored at a co-location. This is the reason that we used the term traditional co-location during the thesis and not just co-location. In the first chapter of this thesis we presented our project goal. In this section we look back at this goal and answer the questions asked in that project goal.

**What configurations are possible?**

The tool provides the possibility to define many different configurations. We described a detailed explanation of these configurations in section 5.3. The important elements are global cost elements mostly used in the configuration costs. Local instances which can be a traditional co-location and a private cloud. And cloud instances which can be one of the clouds from the three cloud providers we used in our tool. These three cloud providers are Amazon Web Services, Google Compute Engine and Windows Azure. As an extra option we added a custom cloud. This cloud gives the user the possibility to define an own set of variables and formulas. These variables and formulas are already predefined for the three cloud providers. The custom cloud can, with the right formulas and variables, replace all the predefined clouds. We added the predefined clouds because we combined these clouds with a price retrieve function. This price retrieve function automatically completes the prices for these clouds. The last important element is the definition of a SAP application. This is the actual SAP application where the tool is about. The application is linked to one defined local instance and one cloud instance. All the ingredients together form a possible configuration which makes the tool usable.

The possible configurations were composed with the input from experts from within Capgemini and by using literature. These experts were asked, after implementing the tool, about the completeness of the tool and they could confirm this. Within Capgemini only their own private cloud is used for local storage, the factors involved with this cloud were retrieved and confirmed by an expert from Infrastructure Services. The elements involving the traditional local storage were substantiated using literature. The available clouds were selected from the most common used clouds and these can be extended by custom clouds when necessary. The SAP application options were composed with a data expert from within Capgemini and confirmed by other experts.
Which costs functions belong to the possible configurations?

The costs involved in the tool are highly dependent on the defined configuration. The costs for a configuration include the necessary performance and the preferred flexibility. For both, local and cloud instances it means that adding a stronger server results in higher costs. When making use of cloud instances the user can choose between reserved and on-demand instances. Reserved instances can be cheaper when needing an high utility, but these instances are less flexible. On-demand instances allow the user to change the instances at any time.

The costs functions for the local instances are based on available literature as described in section 5.2. The costs functions used for the cloud instances were already defined by the cloud providers. We analyzed these costs and processed the formulas in the tool. To keep the costs dynamic we separated the costs in three different types. The first type are instance costs which are calculated per instance. When multiple SAP applications are stored in one instance the instance costs are only calculated once. The second type are data costs which are calculated per SAP application. And the last type are the configuration costs. These costs are calculated per storage location. The configuration costs are only used for the local instances. A storage location in that case can be a traditional co-location or a complete private cloud used for the SAP applications. A detailed description of the formulas used in the tool is presented in Appendix B.

What are interesting rules of thumb to decide where to store data?

The tool provides a suggested distribution after a configuration is provided by the user. This distribution is based on some rules of thumb. A description about how the location is decided is given in section 5.3.4. The most important elements are privacy and location restrictions. The boundary from what privacy level a user wants to allow storage of data in the cloud can be defined in the tool. Each user can have his or her own preference. The Patriot Act, for instance, can be the reason for a company to only prefer storage of data in an European cloud.

How to make the tool in such a way that it is generic?

The tool is designed to be generic. New cloud providers can easily be added, the same for local instance types. The custom cloud allows the user to completely define an own cloud with variables and formulas defined by the user. The generic structure of the instance objects is shown in section 3.2. The large amount of customizable variables allows the user to define only the variables suitable for a specific company. The variables used for the tool are not company dependent. The variables and formulas used are as generic as possible. Adding variables to the existing instances and a SAP application is possible as described in chapter 4.

The structure of the tool allows programmers to use the tool for similar purposes. The decision support tool calculates costs and allows the user to define constraints for a SAP application which is presented in a graphical drag and drop area. This SAP application data can be replaced by other data block definitions. This can be a very similar program like Oracle, but also something like a very large database with tables which should be distributed between several locations.

How to design a user-friendly interface which can easily be used by the target group?

The design of the tool is based on some User Interface guidelines as described in section 3.4. The tool has several features which makes the tool easier to use. Variables which could already be completed with suggested values are already completed. The price retrieve function helps the user completing the costs for a public cloud. A convert function provides the user an easy way to convert a private cloud to a similar Amazon cloud instance, this function can also be used the other way around.
We tested the usability of the tool with multiple test users. The results are described in chapter 6. According to the user tests we made changes to the tool to make it more user-friendly. The user tests showed that the users could work with the tool and that they liked the playful design of the draggable SAP applications. After performing the tasks the users became more confident using the tool. They learned using the tool very quickly.

The tool is not perfect. The usability is especially achieved by the available feature set but there are still some shortcomings in the intuitive design of the tool. Some of the features, like the custom cloud option, are not very clear to the user. Another weaker point of the tool is the large amount of options the user is confronted with. A solution would be to hide the options which are not used or necessary to use at the moment by the user. The choice to confront the user with all the options at once was to provide a clear overview of the available options. The result of this decision is that the user has to learn more in the beginning but once the options are known the use of the tool would be easier and quicker. Further research in application design and gaining knowledge from UI design experts could solve this problem.

7.2 Future work

In the future more features can be added to the decision support tool. And some modifications are welcome to make the tool more generic.

The locations in the distribution tab are now fixed to local storage on the left and cloud storage on the right. One of the reasons these areas are fixed this way is because the problem description was about data distribution between on-premise and on-demand storage locations. To make the tool more available for other purposes the areas can become dynamic. This would mean that the purpose of an area could be defined by a user. At the moment the tool is a decision support tool, this feature would upgrade the application to a more generic comparison tool. Clouds and local instances could be compared to each other.

The formula editor is still a bit complex. Once the user knows how to use it, adding a custom cloud is no problem. The usability could however be improved. A solution would be a drag and drop interface where the user can drag and drop variables and operators to the formula field.

The rules of thumb to decide when a SAP application is stored in the cloud and when it is stored in a local location are static and not adjustable by a normal user. A solution as is used for the custom cloud would be nice, to allow the user to define its own rules of thumb and restrictions.

A more complex addition to the tool would be a complete dynamic variables option. The possibility for the user to define own variables for all sections of the tool. A disadvantage of such an option would be that the tool becomes more complex to use. The formula editor in the decision support tool was already a bit to advanced for some users.
Bibliography


Appendix A

Variables
## Variables

### Instance

<table>
<thead>
<tr>
<th>Factor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server hardware ($shw$)</td>
<td>The total costs for one server</td>
</tr>
<tr>
<td>Server space (Rack U) ($clu$)</td>
<td>The amount of server space needed</td>
</tr>
<tr>
<td>Years of hardware usage ($clyi$)</td>
<td>The number of years before write-off</td>
</tr>
</tbody>
</table>

### Data

<table>
<thead>
<tr>
<th>Factor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Backup costs ($clbu$)</td>
<td>The costs for backup storage per GB</td>
</tr>
<tr>
<td>Tier 1 costs ($cltr_1$)</td>
<td>The costs for the storage per GB in tier 1 storage</td>
</tr>
<tr>
<td>Tier 2 costs ($cltr_2$)</td>
<td>The costs for the storage per GB in tier 2 storage</td>
</tr>
</tbody>
</table>

### Configuration

<table>
<thead>
<tr>
<th>Factor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network hardware ($clnhw$)</td>
<td>The total costs for network hardware, can be calculated as 20% of the hardware costs</td>
</tr>
<tr>
<td>Hardware maintenance ($clmtn$)</td>
<td>Costs for hardware maintenance, can be calculated as 15% of the hardware costs</td>
</tr>
<tr>
<td>Monthly recurring costs ($clexp_1$)</td>
<td>The monthly recurring costs for the co-location</td>
</tr>
<tr>
<td>Non recurring costs ($clexp_2$)</td>
<td>The non-recurring costs for the co-location</td>
</tr>
<tr>
<td>Remote hands support ($clrhs$)</td>
<td>Additional costs for solving network failures</td>
</tr>
<tr>
<td>Power supply space (Rack U) ($clpsu$)</td>
<td>The amount of rack space needed for power supplies</td>
</tr>
<tr>
<td>Years of hardware usage ($cly$)</td>
<td>The number of years before write-off</td>
</tr>
<tr>
<td>WAN costs ($clwan$)</td>
<td>Monthly costs for the data connection</td>
</tr>
</tbody>
</table>

*Table A.1: The parameters for the co-location storage*
### Variables

#### Instance

<table>
<thead>
<tr>
<th>Factor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAPS costs ((pcsaps))</td>
<td>The monthly costs per SAPS</td>
</tr>
</tbody>
</table>

#### Data

<table>
<thead>
<tr>
<th>Factor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Backup costs ((pcbu))</td>
<td>The costs for backup storage per GB</td>
</tr>
<tr>
<td>Tier 1 costs ((pcr_{1}))</td>
<td>The costs for the storage per GB in tier 1 storage</td>
</tr>
<tr>
<td>Tier 2 costs ((pcr_{2}))</td>
<td>The costs for the storage per GB in tier 2 storage</td>
</tr>
</tbody>
</table>

#### Configuration

<table>
<thead>
<tr>
<th>Factor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transition costs ((pctc))</td>
<td>The monthly transition parameters</td>
</tr>
<tr>
<td>VLAN costs ((pcvl))</td>
<td>The costs per VLAN per month</td>
</tr>
<tr>
<td>Firewall costs ((pcfw))</td>
<td>The costs for the firewall per month</td>
</tr>
<tr>
<td>WAN costs ((pcwan))</td>
<td>Monthly costs for the data connection</td>
</tr>
</tbody>
</table>

Table A.2: The parameters for the private cloud storage

#### Instance

<table>
<thead>
<tr>
<th>Factor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instance hours ((cih_{j}))</td>
<td>Number of processing hours for instance (j)</td>
</tr>
<tr>
<td>I/O requests ((cio_{j}))</td>
<td>Number of I/O requests for instance (j)</td>
</tr>
<tr>
<td>Storage ((cs_{j}))</td>
<td>Amount of storage for instance (j)</td>
</tr>
<tr>
<td>I/O requests ((ciop_{j}))</td>
<td>Number of I/O Requests PUT, COPY, POST or LIST for instance (j)</td>
</tr>
<tr>
<td>I/O requests ((ciog_{j}))</td>
<td>Number of I/O Requests GET and all other requests for instance (j)</td>
</tr>
<tr>
<td>Service bus messages ((csb))</td>
<td>Number of Service Bus Messages per instance (j)</td>
</tr>
</tbody>
</table>

Table A.3: The instance parameters for the public cloud storage
### Variables

<table>
<thead>
<tr>
<th>Factor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filesystem filling in %</td>
<td>The amount of data used of the available data</td>
</tr>
<tr>
<td>Incremental backup size in %</td>
<td>Size for incremental backups in percentage of the full backup size</td>
</tr>
</tbody>
</table>

#### PRD backups

<table>
<thead>
<tr>
<th>Factor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Archivelogs changes per week</td>
<td>Size of archivelogs changes in percentage of the full backup size</td>
</tr>
<tr>
<td>Retention time in weeks</td>
<td>The full retention time for all the PRD backups together</td>
</tr>
<tr>
<td>Number of full backups</td>
<td>The number of full backups per week</td>
</tr>
<tr>
<td>Number of incrementals per</td>
<td>The number of incremental backups per week</td>
</tr>
<tr>
<td>week</td>
<td></td>
</tr>
<tr>
<td>Number of month backups</td>
<td>The number of month backups for the system</td>
</tr>
<tr>
<td>Number of year backups</td>
<td>The number of year backups for the system</td>
</tr>
</tbody>
</table>

#### non PRD backups

<table>
<thead>
<tr>
<th>Factor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Archivelogs changes per week</td>
<td>Size of archivelogs changes in percentage of the full backup size</td>
</tr>
<tr>
<td>Retention time in weeks</td>
<td>The full retention time for all the PRD backups together</td>
</tr>
<tr>
<td>Number of full backups</td>
<td>The number of full backups per week</td>
</tr>
<tr>
<td>Number of incrementals per</td>
<td>The number of incremental backups per week</td>
</tr>
<tr>
<td>week</td>
<td></td>
</tr>
</tbody>
</table>

**Table A.4:** The parameters for the backup storage

<table>
<thead>
<tr>
<th>Factor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processing unit (ccph)</td>
<td>depending on instance type</td>
</tr>
<tr>
<td>Upfront (ccu)</td>
<td>Only when using a reserved instance</td>
</tr>
<tr>
<td>Storage (ccs)</td>
<td>per GB</td>
</tr>
<tr>
<td>I/O requests (ccio)</td>
<td>per unit for data storage</td>
</tr>
<tr>
<td>Traffic (cctr)</td>
<td>per GB</td>
</tr>
</tbody>
</table>

**Table A.5:** The pricing scheme for the custom cloud
<table>
<thead>
<tr>
<th>Factor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processing hour (amph)</td>
<td>depending on instance type</td>
</tr>
<tr>
<td>Upfront (amu)</td>
<td>Only when using a reserved instance</td>
</tr>
<tr>
<td>Storage (ams)</td>
<td>per GB for database storage</td>
</tr>
<tr>
<td>I/O request (amio)</td>
<td>per million for the database storage</td>
</tr>
<tr>
<td>Snapshot storage (amss)</td>
<td>costs per GB for backups</td>
</tr>
<tr>
<td>I/O request (amiop)</td>
<td>for PUT, COPY, POST or LIST per thousand</td>
</tr>
<tr>
<td>I/O request (amiog)</td>
<td>for GET and all other requests per 10 thousand</td>
</tr>
<tr>
<td>Traffic (amtr)</td>
<td>per GB</td>
</tr>
<tr>
<td>VPC (amvpc)</td>
<td>per VPN connection hour</td>
</tr>
</tbody>
</table>

Table A.6: The pricing scheme for Amazon Web Services

<table>
<thead>
<tr>
<th>Factor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processing hour (azph)</td>
<td>depending on instance type</td>
</tr>
<tr>
<td>Storage (azs)</td>
<td>per GB</td>
</tr>
<tr>
<td>I/O requests (azio)</td>
<td>per 100,000 requests for data storage</td>
</tr>
<tr>
<td>Traffic (aztr)</td>
<td>per GB</td>
</tr>
<tr>
<td>Connections to applications (azcn)</td>
<td>per month from different hosts</td>
</tr>
</tbody>
</table>

Table A.7: The pricing scheme for Windows Azure (Martens et al., 2012)

<table>
<thead>
<tr>
<th>Factor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processing hour (ggph)</td>
<td>depending on instance type</td>
</tr>
<tr>
<td>Storage (ggs)</td>
<td>per GB</td>
</tr>
<tr>
<td>I/O requests (ggio)</td>
<td>per million requests for data storage</td>
</tr>
<tr>
<td>Snapshot storage (ggss)</td>
<td>costs per GB for backup storage</td>
</tr>
<tr>
<td>Traffic (ggtr)</td>
<td>per GB</td>
</tr>
</tbody>
</table>

Table A.8: The pricing scheme for Google Compute Engine
Appendix B

Costs functions

\[ n \] Number of SAP applications in local storage
\[ m \] Number of SAP applications in the public cloud
\[ k \] Number of co-location instances
\[ l \] Number of private cloud instances
\[ p \] Number of AWS cloud instances
\[ q \] Number of Windows Azure cloud instances
\[ r \] Number of GCE cloud instances
\[ s \] Number of Custom cloud instances
\[ t \] Number of system types
\[ saps_j \] Amount of SAPS for instance \( j \)
\[ os_j \] OS storage for an instance \( j \) in GB
\[ t_i \] Traffic for SAP application \( i \) in GB per month
\[ s_i \] Database Storage for SAP application \( i \) in GB per month
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Factor</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Backup storage for PRD</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\text{prd}_{\text{sapfull}_i}$</td>
<td>Full backup SAP</td>
<td>$\frac{\text{bufill}}{100} \times s_i \times p\text{full}$</td>
</tr>
<tr>
<td>$\text{prd}_{\text{sapinc}_i}$</td>
<td>Incremental backup SAP</td>
<td>$\frac{\text{bainc}}{100} \times \text{prd}_{\text{sapfull}_i} \times p\text{inc}$</td>
</tr>
<tr>
<td>$\text{prd}_{\text{arch}_i}$</td>
<td>Archive backups</td>
<td>$\frac{\text{parch}}{100} \times \text{prd}_{\text{sapfull}_i}$</td>
</tr>
<tr>
<td>$\text{prd}_{\text{osfull}_j}$</td>
<td>Full backup OS</td>
<td>$\frac{\text{bufill}}{100} \times o\text{s}_j \times p\text{ret}$</td>
</tr>
<tr>
<td>$\text{prd}_{\text{osinc}_j}$</td>
<td>Incremental backup OS</td>
<td>$\frac{\text{bainc}}{100} \times \text{prd}_{\text{osfull}_j} \times p\text{ret}$</td>
</tr>
<tr>
<td>$\text{prd}_{\text{tot}_i}$</td>
<td>Total backup storage PRD</td>
<td>$(\text{prd}_{\text{sapfull}<em>i} + \text{prd}</em>{\text{sapinc}<em>i} + \text{prd}</em>{\text{arch}_i}) \times p\text{ret}$</td>
</tr>
<tr>
<td>$\text{prd}_{\text{month}_i}$</td>
<td>Month backups</td>
<td>$\text{prd}_{\text{sapfull}_i} \times p\text{month}$</td>
</tr>
<tr>
<td>$\text{prd}_{\text{year}_i}$</td>
<td>Year backups</td>
<td>$\text{prd}_{\text{sapfull}_i} \times p\text{year}$</td>
</tr>
<tr>
<td><strong>Backup storage for non PRD</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\text{nprd}_{\text{sapfull}_i}$</td>
<td>Full backup SAP</td>
<td>$\frac{\text{bufill}}{100} \times s_i \times n\text{pfull}$</td>
</tr>
<tr>
<td>$\text{nprd}_{\text{sapinc}_i}$</td>
<td>Incremental backup SAP</td>
<td>$\frac{\text{bainc}}{100} \times \text{nprd}_{\text{sapfull}_i} \times n\text{pinc}$</td>
</tr>
<tr>
<td>$\text{nprd}_{\text{arch}_i}$</td>
<td>Archive backups</td>
<td>$\frac{\text{n parch}}{100} \times \text{nprd}_{\text{sapfull}_i}$</td>
</tr>
<tr>
<td>$\text{nprd}_{\text{osfull}_j}$</td>
<td>Full backup OS</td>
<td>$\frac{\text{bufill}}{100} \times o\text{s}_j \times n\text{pret}$</td>
</tr>
<tr>
<td>$\text{nprd}_{\text{osinc}_j}$</td>
<td>Incremental backup OS</td>
<td>$\frac{\text{bainc}}{100} \times \text{nprd}_{\text{osfull}_j} \times n\text{pret}$</td>
</tr>
<tr>
<td>$\text{nprd}_{\text{tot}_i}$</td>
<td>Total backup storage non PRD</td>
<td>$(\text{nprd}_{\text{sapfull}<em>i} + \text{nprd}</em>{\text{sapinc}<em>i} + \text{nprd}</em>{\text{arch}_i}) \times n\text{pret}$</td>
</tr>
<tr>
<td><strong>Total backup storage</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\text{bu}_{\text{dbtot}_i}$</td>
<td>Total DB storage</td>
<td>$\text{prd}_{\text{tot}<em>i} + \text{nprd}</em>{\text{tot}<em>i} + \text{prd}</em>{\text{month}<em>i} + \text{prd}</em>{\text{year}_i}$</td>
</tr>
<tr>
<td>$\text{bu}_{\text{ostot}_j}$</td>
<td>Total OS storage</td>
<td>$\text{prd}_{\text{osfull}<em>j} + \text{prd}</em>{\text{osinc}<em>j} + \text{nprd}</em>{\text{osfull}<em>j} + \text{nprd}</em>{\text{osinc}_j}$</td>
</tr>
</tbody>
</table>

Table B.1: The storage calculations for the backups
### Costs functions

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Factor</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>$cl_{shw}$</td>
<td>Server hardware</td>
<td>$\sum_{i=1}^{k} \frac{shw_i}{cl_{cl}}$</td>
</tr>
<tr>
<td>$cl_{tr_x}$</td>
<td>Tier $x$ storage</td>
<td>$\sum_{i=1}^{n} cl_{tr_x} \times s_i$</td>
</tr>
<tr>
<td>$cl_{dbbu}$</td>
<td>Backup storage DB</td>
<td>$\sum_{i=1}^{n} bu_{dbtot_i} \times cl_{bu}$</td>
</tr>
<tr>
<td>$cl_{osbu}$</td>
<td>Backup storage OS</td>
<td>$\sum_{j=1}^{k} bu_{ostot_j}$</td>
</tr>
<tr>
<td>$cl_{nhw}$</td>
<td>Network hardware</td>
<td>$cl_{shw} \times 0.20$</td>
</tr>
<tr>
<td>$cl_{mtn}$</td>
<td>Maintenance</td>
<td>$(cl_{shw} + cl_{nhw}) \times 0.15$</td>
</tr>
<tr>
<td>$cl_{exp}$</td>
<td>Co-Location expense</td>
<td>$\frac{40(clu + cl_{psu})}{12cly} + 45(clu + cl_{psu})$</td>
</tr>
<tr>
<td>$cl_{rhs}$</td>
<td>Remote hands support</td>
<td>$k \times 0.10 \times 1 \times 150$</td>
</tr>
<tr>
<td>$cl_{tot}$</td>
<td></td>
<td>$cl_{shw} + cl_{tr_x} + cl_{dbbu} + cl_{osbu} + cl_{nhw} + cl_{mtn} + cl_{exp} + cl_{rhs} + cl_{wan}$</td>
</tr>
</tbody>
</table>

**Table B.2:** The cost calculations for co-location
## Costs functions

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Factor</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>$pc_{saps}$</td>
<td>SAPS</td>
<td>$\sum_{j=1}^{l} saps_j \times pc_{saps}$</td>
</tr>
<tr>
<td>$pc_{oss}$</td>
<td>Non DB storage tier $x$</td>
<td>$\sum_{j=1}^{n} lsto_j \times pc_{tr}$</td>
</tr>
<tr>
<td>$pc_{dbs}$</td>
<td>Database storage tier $x$</td>
<td>$\sum_{i=1}^{n} s_i \times pc_{tr}$</td>
</tr>
<tr>
<td>$pc_{dbbu}$</td>
<td>Backup storage DB</td>
<td>$\sum_{i=1}^{n} bu_{dbbtot_i} \times pc_{bu}$</td>
</tr>
<tr>
<td>$pc_{osbu}$</td>
<td>Backup storage OS</td>
<td>$\sum_{j=1}^{l} bu_{ostot_j} \times pc_{bu}$</td>
</tr>
<tr>
<td>$pc_{tr}$</td>
<td>Transition</td>
<td>$pc_{tr}$</td>
</tr>
<tr>
<td>$pc_{nw}$</td>
<td>Networking</td>
<td>$(pcvl \times t) + pc_{fw}$</td>
</tr>
<tr>
<td>$pc_{wan}$</td>
<td>WAN</td>
<td>$pc_{wan}$</td>
</tr>
<tr>
<td>$pc_{tot}$</td>
<td></td>
<td>$pc_{saps} + pc_{oss} + pc_{dbs} + pc_{dbbu} + pc_{osbu} + pc_{tr} + pc_{nw} + pc_{wan}$</td>
</tr>
</tbody>
</table>

Table B.3: The cost calculations for the private cloud
### Costs functions

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Factor</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>$am_{iu}$</td>
<td>EC2 instance usage</td>
<td>$\sum_{j=1}^{q} amu_j + (cih_j \times amph_j)$</td>
</tr>
<tr>
<td>$am_{io}$</td>
<td>EBS I/O requests</td>
<td>$\sum_{j=1}^{q} ami_io_j \times cio_j$</td>
</tr>
<tr>
<td>$am_{nds}$</td>
<td>S3 Non-DB storage</td>
<td>$\sum_{j=1}^{q} amss_j \times cs_j$</td>
</tr>
<tr>
<td>$am_{iop}$</td>
<td>S3 I/O requests PUT, COPY, POST</td>
<td>$ciop_j \times amiop_j$</td>
</tr>
<tr>
<td>$am_{iog}$</td>
<td>S3 I/O requests GET and all other</td>
<td>$ciog_j \times amiog_j$</td>
</tr>
<tr>
<td>$am_{osbu}$</td>
<td>Backup OS storage</td>
<td>$\sum_{j=1}^{q} bu_{ostot_j} \times amss_j$</td>
</tr>
<tr>
<td>$am_{dbs}$</td>
<td>Database storage</td>
<td>$\sum_{j=1}^{q} \sum_{i=1}^{m} si \times ams_j$</td>
</tr>
<tr>
<td>$am_{tr}$</td>
<td>Traffic</td>
<td>$\sum_{j=1}^{q} \sum_{i=1}^{m} ti \times amtr_j$</td>
</tr>
<tr>
<td>$am_{dbbu}$</td>
<td>Backup DB storage</td>
<td>$\sum_{j=1}^{q} \sum_{i=1}^{m} bu_{dbtot_i} \times amss_j$</td>
</tr>
</tbody>
</table>

**Total**

$am_{tot} = am_{iu} + am_{io} + am_{nds} + am_{iop} + am_{iog} + am_{dbs} + am_{tr} + am_{dbbu} + am_{osbu}$

Table B.4: The cost calculations for Amazon Web Services
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Factor</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Instance</strong></td>
<td></td>
</tr>
<tr>
<td>$az_{iu}$</td>
<td>Instance usage</td>
<td>$\sum_{j=1}^{q} cih_j \times azph_j$</td>
</tr>
<tr>
<td>$az_{nds}$</td>
<td>Non-DB storage</td>
<td>$\sum_{j=1}^{q} azs_j \times cs_j$</td>
</tr>
<tr>
<td>$az_{io}$</td>
<td>I/O requests</td>
<td>$cio_j \times azio_j$</td>
</tr>
<tr>
<td>$az_{ms}$</td>
<td>Messaging</td>
<td>$csb_j \times azcn$</td>
</tr>
<tr>
<td>$az_{osbu}$</td>
<td>Backup OS storage</td>
<td>$\sum_{j=1}^{q} buostot_j \times azs_j$</td>
</tr>
<tr>
<td></td>
<td><strong>Data</strong></td>
<td></td>
</tr>
<tr>
<td>$az_{dbs}$</td>
<td>Database storage</td>
<td>$\sum_{j=1}^{q} \sum_{i=1}^{m} s_i \times azs_j$</td>
</tr>
<tr>
<td>$az_{tr}$</td>
<td>Traffic</td>
<td>$\sum_{j=1}^{q} \sum_{i=1}^{m} ti \times aztr_j$</td>
</tr>
<tr>
<td>$az_{dbbu}$</td>
<td>Backup DB storage</td>
<td>$\sum_{j=1}^{q} \sum_{i=1}^{m} bu_{dbtot_i} \times azs_j$</td>
</tr>
<tr>
<td></td>
<td><strong>Total</strong></td>
<td></td>
</tr>
<tr>
<td>$az_{tot}$</td>
<td></td>
<td>$az_{iu} + az_{nds} + az_{io} + az_{ms} + az_{dbs} + az_{tr} + az_{dbbu} + az_{osbu}$</td>
</tr>
</tbody>
</table>

Table B.5: The cost calculations for Windows Azure
### Costs functions

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Factor</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>$g_{iu}$</td>
<td>Instance usage</td>
<td>$\sum_{j=1}^{q} ci_{j}h_{j} \times g_{ph_{j}}$</td>
</tr>
<tr>
<td>$g_{nds}$</td>
<td>Non-DB storage</td>
<td>$\sum_{j=1}^{q} g_{s_{j}} \times cs_{j}$</td>
</tr>
<tr>
<td>$g_{io}$</td>
<td>I/O requests</td>
<td>$cio_{j} \times g_{gio_{j}}$</td>
</tr>
<tr>
<td>$g_{osbu}$</td>
<td>Backup OS storage</td>
<td>$\sum_{j=1}^{q} bu_{osbot_{j}} \times g_{ss_{j}}$</td>
</tr>
<tr>
<td>$g_{dbs}$</td>
<td>Database storage</td>
<td>$\sum_{j=1}^{q} \sum_{i=1}^{m} s_{i} \times g_{gs_{j}}$</td>
</tr>
<tr>
<td>$g_{tr}$</td>
<td>Traffic</td>
<td>$\sum_{j=1}^{q} \sum_{i=1}^{m} t_{i} \times g_{gr_{j}}$</td>
</tr>
<tr>
<td>$g_{dbbu}$</td>
<td>Backup DB storage</td>
<td>$\sum_{j=1}^{q} \sum_{i=1}^{m} bu_{dbbot_{j}} \times g_{ss_{j}}$</td>
</tr>
<tr>
<td>$g_{tot}$</td>
<td></td>
<td>$g_{iu} + g_{nds} + g_{io} + g_{dbs} + g_{tr} + g_{dbbu} + g_{osbu}$</td>
</tr>
</tbody>
</table>

**Table B.6:** The cost calculations for Google Compute Engine
### Costs functions

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Factor</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>$cc_{iu}$</td>
<td>Instance usage</td>
<td>$\sum_{j=1}^{q} cc_{u_j} + (ci_{h_j} \times cc_{ph_j})$</td>
</tr>
<tr>
<td>$cc_{nds}$</td>
<td>Non-DB storage</td>
<td>$\sum_{j=1}^{q} cc_{s_j} \times cs_j$</td>
</tr>
<tr>
<td>$cc_{io}$</td>
<td>I/O requests</td>
<td>$cio_j \times ccio_j$</td>
</tr>
<tr>
<td>$cc_{osbu}$</td>
<td>Backup OS storage</td>
<td>$\sum_{j=1}^{q} bu_{ostot_j} \times cc_{s_j}$</td>
</tr>
<tr>
<td>$cc_{dbs}$</td>
<td>Database storage</td>
<td>$\sum_{j=1}^{q} \sum_{i=1}^{m} s_i \times cc_{s_j}$</td>
</tr>
<tr>
<td>$cc_{tr}$</td>
<td>Traffic</td>
<td>$\sum_{j=1}^{q} \sum_{i=1}^{m} t_i \times cc_{tr_j}$</td>
</tr>
<tr>
<td>$cc_{dbbu}$</td>
<td>Backup DB storage</td>
<td>$\sum_{j=1}^{q} \sum_{i=1}^{m} bu_{dbtot_i} \times cc_{s_j}$</td>
</tr>
</tbody>
</table>

#### Total

| $cc_{tot}$ | $cc_{iu} + cc_{nds} + cc_{io} + cc_{dbs} + cc_{tr} + cc_{dbbu} + cc_{osbu}$ |

*Table B.7: The cost calculations for the custom cloud*