UNIVERSITY OF TWENTE.

Automatic Distribution Analysis of Business Processes for Cloud-Based BPM

Master Thesis

Tesfahun Aregawy Tesfay
s1167081
Faculty of EEMCS - Electrical Engineering, Mathematics and Computer Science
University of Twente, Enschede, The Netherlands

Graduation committee:

dr. Luís Ferreira Pires	[UT 1st Supervisor]
dr.ir Marten Van Sinderen	[UT 2nd Supervisor]
dr. Luiz O. B. da Silva Santos	[BiZZdesign 1st Supervisor]
dr. Dick A.C. Quartel	[BiZZdesign 2nd Supervisor]

August 08, 2013

BiZZdesign www.bizzdesign.com

Abstract

Business Process Management (BPM) has been used to organize, visualize, analyze, optimize and continuously improve the business processes of organizations. BPM supports the composition of Service-Oriented Architecture (SOA) services and human tasks into complete organizational business processes at the level of business process modeling. Many of these business process models involve several on-premise computation-intensive activities and data, requiring expensive massive computing power and data storage.

In recent years, cloud computing emerged to offer opportunities, such as reducing the upfront investments in infrastructure and taking advantage of a vast amount of cheaper computational capacity and data storage resources. As a result, organizations considered migrating total or parts of on-premise business processes to cloud-based BPM as a favorable alternative for business process improvement. However, organizations face challenges regarding the identification and selection of distribution options of business processes into collaborating in-cloud and on-premise engines considering multiple cloud migration decision factors, such as cost benefits and privacy risks.

Our main research objective in this thesis is to identify the most relevant cloud migration drivers and barriers, investigate how they should relate to business processes, and define algorithms that can be used to automatically identify and rank distribution options for Cloud-based BPM. The results of our research facilitate the automatic identification and selection of distribution options of business processes into collaborating in-cloud and on-premise engines, and their consequences.

As a proof-of-concept implementation, we have developed an annotation language and automated system for identifying and ranking distribution options of business processes and their consequences by representing cost benefits and privacy risks for Cloud-based BPM in the BiZZdesign Architect modeling tool.

Preface

This report describes the results of my final project performed at BiZZdesign in order to obtain the Master degree in Computer Science, Software Engineering specialization from the University of Twente. What I presented in this report is a way less than the effort I put into this work. I read more than 300 papers in a much unstructured manner. Out of these papers only about 49 were relevant to this work. Many of the papers that seemed to be relevant based on the systematic literature search techniques ended up being irrelevant to my goals. Though, I do not regret reading them, this could have been more efficient, for example, by first somehow systematically selecting the most relevant papers from all of these 300 papers before reading them. This boils down to a very broad research problem. Narrowing down the research problem really improved this situation, which I have done later in the process with the help of several discussions with my supervisors. This work has given me the chance to improve my scientific writing and the way I think about research. I conclude the most challenging experience I ever had with this thesis and I am happy with my results. I have learned so much in all dimensions of my life over the last two years.

Tesfahun A. Tesfay

Enschede, August 08, 2013

Acknowledgments

I would like to express my deepest gratitude to my first supervisor, Luís, for his caring advice and the confidence he has shown in me during my master thesis and research topics work. His knowledge and experience in cloud computing, business processes, service-oriented architecture and model-driven engineering make him the ideal supervisor for this assignment. He has given me a genuine continuous guidance in every door I have opened during this work.

I would like to thank Marten for accepting to be my second supervisor and for the invaluable feedback and support he has given me.

I would like to thank my supervisors at BiZZdesign. Luiz, for initiating the assignment and the useful lectures and continuous feedback he has given me. Dick, for getting me started with the tools and languages of BiZZdesign, and his invaluable suggestions especially at the beginning of the assignment. I would also like to thank Henry Franken for his feedback during the SCRUM meetings.

I would like to thank BiZZdesign for giving me the chance to experience the adventure of combining academic theories with industry practices, and all the employees and fellow students of BiZZdesign for the great time we have had during this work.

I would like to thank the University of Twente for admitting me to study the subject of my choice, and the Huygens Scholarship Programme for providing the necessary financial support for my studies. I would like to thank Jan Schut for helping with all these procedures. I would like to thank Mehmet Aksit, Ivan Kurtev, Christoph Bockisch and all of my professors for giving me the wonderful lessons and experiences during the last two years.

I would like to thank my mother, Mebrhit, my grandmother, Tsegab and all my family for their support and prayers throughout my studies. I want to thank Goran, TK, Fasil, Shirin, Malkom, Ferry, Abel, Bini and all of my friends for helping me adapt to life in Enschede and the great times we spent together. I want to thank Ralf Laemmel for funding all the conferences and workshops I have attended, and for showing me that professors can also be great friends.

I would like to thank my fiancée, Maereg, for her great patience and strength to overcome all the difficulties of long-distance relationship to stay with me. I look forward to our wedding next month.

I thank Jesus Christ for giving me wisdom and strength to successfully complete my studies. I am looking forward to my next challenge at the Eindhoven University of Technology.

Table of Contents

Chapter 1: Introduction 10
1.1 Motivation 10
1.2 Problem Statement 11
1.3 Objectives 12
1.4 Approach 13
1.5 Structure 14
Chapter 2: Cloud and BPM 16
2.1 Cloud Computing 16
2.2 BPM 18
2.3 Cloud-Based BPM 20
2.3.1 Delivery as SaaS 20
2.3.2 Delivery as PaaS 21
2.3.3 Delivery as IaaS 21
2.4 Benefits 22
2.5 Challenges 22
Chapter 3: Cloud Migration 26
3.1 Cloud Migration Decision Support 26
3.2 General cloud migration decisions 27
3.2.1 $(MC^2)^2$ framework 27
3.2.2 CloudGenius framework 30
3.2.3 Value estimation framework 31
3.2.4 IaaS migration tools 31
3.2.5 Cloudstep 32
3.2.6 Questionnaires 35
3.2.7 Holistic analytical model 35
3.3 Business process cloud deployment and outsourcing decisions 36

3.3.1 Optimal activity and data distribution model 36

3.3.2 Developing a decision model for business process outsourcing........................36

3.3.3 Business activity outsourcing 37

3.4 General IT outsourcing decisions...37

3.5 Summary..38

Chapter 4: Cloud Migration Drivers and Barriers..40

4.1 Identification Approach...40

4.2 Cloud Migration Drivers..41

4.3 Cloud Migration Barriers...42

4.4 Factors for Further Consideration 43

Chapter 5: Automated System Design...45

5.1 Approach..45

5.2 Cloud Billing Models 49

5.3 Privacy Risk 53

5.4 Annotation Language Metamodel..53

5.5 Model-based Algorithms 56

Chapter 6: Implementation of the Annotation Language Metamodel...................................60

6.1 Enumeration Types...60

6.2 Annotation Language Elements for Cost and Privacy 62

6.3 Auxiliary Annotation Language Elements 64

6.4 Annotation Markers..65

Chapter 7: Model-based Algorithms 67

7.1 Conventions 67

7.2 Helpers...68

7.2.1 Elements Identifier 68

7.2.2 Limit Counter...69

7.2.3 Privacy Risk Counter...70

7.2.4 Elements Counter ... 71
7.2.5 Controller ... 72
7.3 Distribution Identification Algorithm ... 73
7.4 Cost Algorithms ... 75
7.4.1 Computing Capacity Cost ... 75
7.4.2 Data Storage Cost ... 76
7.4.3 Data Communication Cost ... 78
7.4.4 On-premise Costs ... 79
7.4.5 Cost Benefit ... 81
7.5 Privacy Risk Algorithm ... 81
7.6 Analysis and Ranking Algorithms ... 83
7.6.1 Analysis ... 83
7.6.2 Ranking ... 84
7.7 Display Distribution Options and Consequences ... 85
Chapter 8: Validation ... 87
8.1 Validation Purpose ... 87
8.2 ArchiSurance Broker ... 87
8.3 Annotation Step ... 88
8.4 Analysis Step ... 89
8.5 Validation Report ... 92
Chapter 9: Conclusions ... 94
9.1 Answers to Research Questions ... 94
9.2 Future work ... 95
9.3 Recommendations for BiZZdesign ... 95
Bibliography ... 97

List of Figures

Figure 1 The model of our research approach...………..……………………….…………………….14
Figure 2 Structure of this thesis showing all the chapters, where research questions are addressed and where steps of our research approach are applied.……..…………………………………......…… 15
Figure 3 The NIST definition of Cloud Computing adopted from [9] ……….................................. 16
Figure 4 BPM Lifecycle……………………………………………………………………….19
Figure 5 Cloud-Based BPM delivered in the SaaS model [5]………………..……………………20
Figure 6 Cloud-Based BPM delivered in the PaaS model [5]………………..……………………21
Figure 7 Cloud-Based BPM delivered in the IaaS model [5] …………………………….………..21
Figure 8 Patterns of BPM architectures …………………………………………………….………23
Figure 9 Cloud-based BPM with on-premise distribution ……………………………...………..24
Figure 10 An approach to systematically split one monolithic business process model into on-premise and cloud-side data and activities adopted from [17] ……………………….……..……….. 25
Figure 11 Process Steps of the generic (MC2)2 cloud adoption framework ………………………...27
Figure 12 The Process of Executing Evaluation Method …………………………………………….30
Figure 13 the complete workflow of Cloudstep cloud migration decision process ………………….34
Figure 14 Decision hierarchy for selecting activities to outsource ……………………..….….…….. 37
Figure 15 Decision Structure of IT Outsourcing …………………………………...…………...38
Figure 16 A Business Process Model Executed On-premise Engine ……………….……………….49
Figure 17 Manually identified one possible distribution option into collaborating in-cloud and on-premise engines ………………………………………………………….……………………... 47
Figure 18 How our automated system complements the work of [17] by automatically identifying and ranking distribution options and their consequences before the actual decomposition………….…48
Figure 19 Metamodel of our annotation language……………………………….…………...54
Figure 20 ArchiSurance Broker Model in BiZZdesign Architect Modeling tool by using BPMN 2.0 as a modeling language…………………….……………………………………………………..88
Figure 21 ArchiSurance Broker model after annotation by using our annotation language in the BiZZdesign Architect Modeling tool…………...………………………………………………..89
Figure 22 The annotation language elements available for annotating ArchiSurance Broker activities in the BiZZdesign Architect modeling tool………………………….………………………….89
Figure 23 Implementations of our algorithms available as viewpoints on the BiZZdesign Architect modeling tool……………………………………………………………………………………..90

List of Tables

Table 1 Questionnaire based cloud migration decision guide ………...……………………………..35

Table 2 Cloud Migration Drivers and Barrier …………………..…………..……………………… 41

Table 3 Mappings of BPMN 2.0 process constructs to our annotation language elements ..……... 55

Table 4 Sample detailed distribution options and their consequences in a table output type………...91

Table 5 Sample top 10 ranked distribution options and their consequences…………………………91

Chapter 1: Introduction

The purpose of this chapter is to provide the motivation for conducting this work, our problem statement, our research objectives and research questions, and research approach. This chapter also gives the structure of this thesis.

This chapter is structured as follows: Section 1.1 discusses our motivation; Section 1.2 presents our problem statement; Section 1.3 explains our main research objectives and the research questions addressed in this work; Section 1.4 presents our research approach and Section 1.5 gives the structure of the rest of this thesis.

1.1 Motivation

Business Process Management (BPM) has been used to organize, visualize, analyze, optimize and continuously improve business processes of organizations [1]. BPM supports the composition of Service-Oriented Architecture (SOA) services and human tasks into complete organizational business processes at the level of business process modeling [2]. Business process models facilitate common understanding among the management staff about the operation of an organization. Many of these business process models involve several on-premise computation-intensive activities and data, requiring expensive massive computing power and data storage [3].

In recent years, cloud computing emerged to offer opportunities, such as reducing the upfront investments in infrastructure and taking advantage of a vast amount of cheaper computational capacity and data storage resources [4]. As a result, organizations considered migrating on-premise data and computation-intensive business processes to *Cloud-based BPM*, as a favorable alternative for business process improvement regarding computational capacity and data storage resources. Since the past few years, a growing number of organizations are outsourcing their business processes and process engines to Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) or Software-as-a-Service (SaaS) cloud service models in order to exploit cheaper computing capacity, data storage and expertise of the cloud service providers [5].

Although cloud migration usually promises cost benefits, migrating all on-premise business processes to cloud service providers does not always guarantee the lowest costs and it may expose the organization’s strategic information to unauthorized access. Therefore, the analysis of distribution options of business processes into collaborating in-cloud and on-premise engines, and their consequences, by considering multiple cloud migration drivers and barriers, such as cost benefits and privacy risks is necessary to decide which parts are distributed where.

There are many cloud migration guides and models in literature that can be used to support these decisions [3] [6] [7] [8] [9] [10]. Nevertheless, these guides and models are not well integrated with business processes, and hence cannot be automatically applied by business organizations. Performing the analysis of distributions of business processes into collaborating in-cloud and on-promise engines at the level of activities and data items by hand is unrealistic for organizations. Organizations have neither the time nor the resources to compare each of the distribution options manually at the level of activities and data items.

An automated system can be used to identify and rank possible distribution options of business processes into collaborating in-cloud and on-premise engines, and their consequences, by considering the most relevant cloud migration drivers and barriers, such as cost benefits and privacy risks. This would be beneficial for organizations when taking decisions regarding the adoption of Cloud-based BPM. Building such an automated system requires thorough understanding of how the most relevant cloud migration drivers and barriers should relate to business processes. This knowledge can be used to design an annotation language that can be used for annotating business processes with cloud-related information. Algorithms are also required that can be used to automatically identify and rank distribution options, and their consequences based on annotated business processes.

1.2 Problem Statement

Cloud-based BPM allows organizations to benefit from the cheaper computing capacity and data storage resources offered by cloud technologies by carefully distributing business processes into collaborating in-cloud and on-premise engines without exposing their strategic information to unauthorized access [3]. Therefore, analysis is necessary to allocate non-sensitive computation-intensive data and activities to an in-cloud engine, and sensitive and non-computation-intensive data and activities to an on-premise engine.

Performing the analysis of distributions of business processes into collaborating in-cloud and on-promise engines at the level of activities and data items by hand is unrealistic for organizations. Organizations have neither the time nor the resources to compare each of the distribution options manually at the level of activities and data items. For example, a business process model with only six activities and data items in total would result in up to $2^6 = 64$ in-cloud and on-premise possible distributions of business processes, including the options in which everything is on-premise or in-cloud. These distributions have to be compared with each other, considering multiple cloud migration drivers and barriers, such as cost benefit and privacy risk, in order to select a distribution option that gives the organization the highest possible profit with a minimal acceptable risk.

An automated system that can be used to identify and rank possible distribution options of business processes into collaborating in-cloud and on-premise engines, and their consequences considering the most relevant cloud migration factors would be beneficial for organizations when taking decisions regarding the adoption of Cloud-based BPM. Developing such a system requires a proper understanding of how the cloud-related information regarding these factors should relate to business processes. This can be used to design an annotation language that can be used for annotating business processes with cloud-related information. Algorithms that can be used to automatically identify and rank distribution options based on annotated business processes are also needed. The knowledge of what exactly the cloud-related information should be and how it should relate to business processes is not readily available at the moment. This work fills this research gap by identifying the most relevant cloud migration drivers and barriers and how they should relate to business processes, and by defining algorithms that can be applied to such analysis.

1.3 Objectives

The main research objectives of this work are to identify the most relevant cloud migration drivers and barriers, investigate how these factors should relate to business processes, and define algorithms that can be used for analysis. The results of our research facilitate the automation and analysis of the distributions of business processes into collaborating in-cloud and on-premise engines, and the identification of benefits and risks based on process models.

We aim of achieving our objective by answering the following research questions:

RQ1: What are the most relevant cloud migration drivers and barriers that affect the identification and selection of distributions of business processes for cloud-based BPM?
The most relevant cloud migration drivers and barriers in literature have to be identified.

RQ2: How are the most relevant cloud migration drivers and barriers mapped to process modeling constructs?
A business process modeling language should be selected, and the mapping of the most relevant factors to process modeling constructs in that language should be investigated.

RQ3: How to identify and rank distribution options of business processes into collaborating in-cloud and on-premise engines?
Algorithms that can be used to identify and rank distribution options and their consequences, such as cost savings and privacy risks should be defined.

1.4 Approach

In order to answer our research questions and achieve our objective we have taken the following concrete steps:

1. Perform a literature study on cloud computing, cloud-based BPM, and cloud migration decision support systems.

   The purpose of this step is to:

   - Understand the benefits and challenges of cloud migration in general and Cloud-based BPM in particular.
   - Investigate if any available decision support systems can partially solve our problem.
   - Identify relevant cloud migration drivers and barriers that affect the identification and selection of distributions of business processes for Cloud-based BPM.

This step addresses research question RQ1.

2. Literature study on cloud computing billing models, cloud privacy issues, and the characteristics of business processes and process modeling constructs of the languages supported by the available tools at BiZZdesign.

The purpose of this step is to select a business process modeling language, identify cloud-related information regarding cost benefits and privacy risks, and investigate how this information should be mapped to the process modeling constructs in the chosen language. This step addresses research question RQ2.

3. Design an annotation language for annotating business processes with cloud-related information.

This contributes to answering research questions RQ2 and RQ3.

4. Define model-based algorithms for identifying and ranking distribution options and their consequences based on annotated business processes.

This step addresses research question RQ3.

5. Implement a proof-of-concept of our automated system in the BiZZdesign Architect modeling tool.

   The purpose of this step is to incorporate the knowledge in the tooling by:

   - Implementing the metamodel of our annotation language by using BiZZdesign Architect profiles.

- Implementing the model-based algorithms for identifying and ranking distribution options and their consequences by considering cost and privacy in the BiZZdesign Architect script language.

6. Validate our work with a realistic case study.

Step 1 addresses research question RQ1, step 2 addresses research question RQ2, step 3 contributes to answering research questions RQ2 and RQ3, steps 4 addresses research question RQ3, step 5 implements a proof-of-concept of our automated system and annotation language on the BiZZdesign Architect modeling tool and step 6 validates our work by using a realistic case study. These activities are taken based on the model shown in Figure 1.

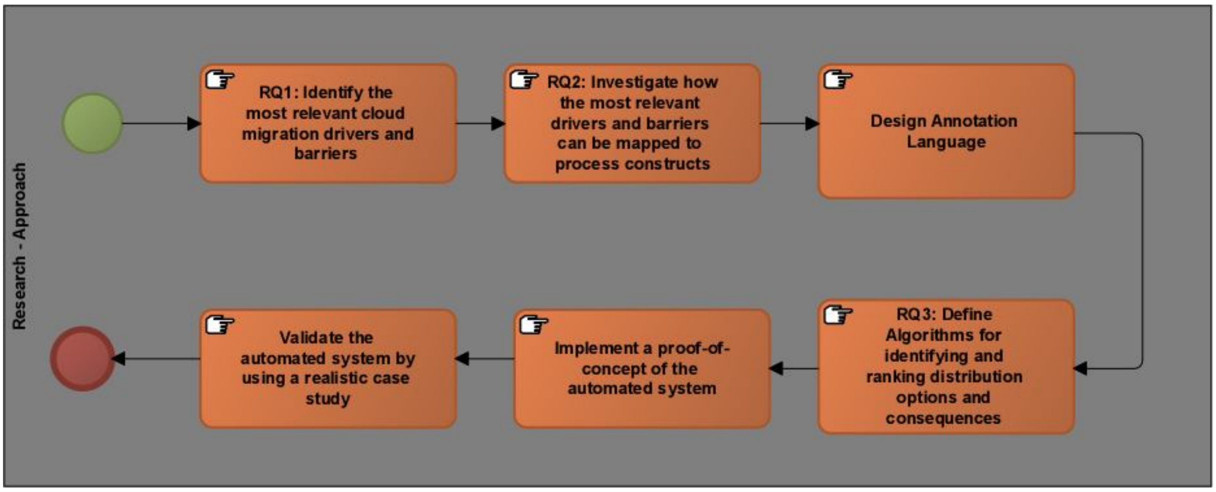

Figure 1 The model of our research approach

1.5 Structure

This thesis is further structured as follows:

Chapter 2 gives background information on cloud computing and BPM. The purpose of this chapter is to allow readers to understand the rest of this work, by discussing the basics of the cloud, BPM and their combination. This chapter also contributes to answering research question RQ1.

Chapter 3 gives background information on cloud migration. The purpose of this chapter is to allow readers to understand the rest of this work, by discussing the available techniques that can be used to guide the decision to migrate to the cloud. This chapter also contributes to answering research question RQ1.

Chapter 4 identifies the most relevant cloud migration drivers and barriers that affect the identification and selection of distributions of business processes for Cloud-based BPM. This chapter addresses research question RQ1.

Chapter 5 introduces the development approach we have taken in this work, our automated system and annotation language. This chapter answers research question RQ2.

Chapter 6 explains the implementation of our annotation language metamodel in the BiZZdesign Architect Profile Definition Language as required by the BiZZdesign Architect modeling tool.

Chapter 7 defines algorithms for identifying and ranking distribution options of business processes for cloud-based BPM. This chapter answers research question RQ3.

Chapter 8 validates our work by applying our automated system to a realistic case study.

Chapter 9 concludes this work.

Figure 2 illustrates the structure of this thesis showing the sequence of all the chapters, where the research questions are addressed, and where the steps of our research approach are applied.

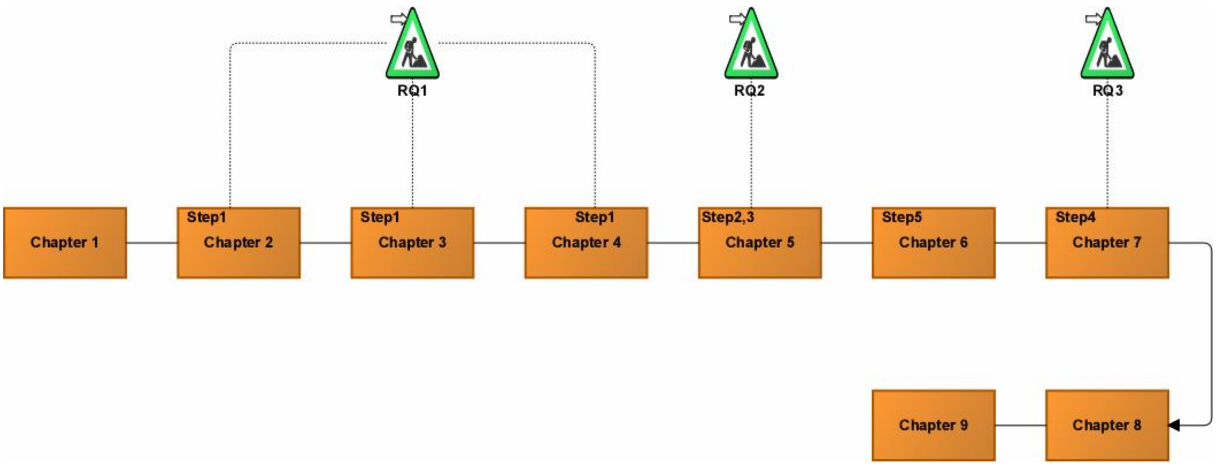

Figure 2 Structure of this thesis showing all the chapters, where research questions are addressed and where steps of our research approach are applied.

Chapter 2: Cloud and BPM

The purpose of this chapter is to allow the reader to understand the rest of this work by giving background information on cloud computing, BPM and the benefits and challenges of their combination (cloud-based BPM).

This chapter is further structured as follows: Sections 2.1 introduces cloud computing; Section 2.2 introduces BPM and Sections 2.3 discusses the benefits and challenges of their combination (cloud-based BPM).

2.1 Cloud Computing

According to the National Institute of Standards and Technology (NIST), cloud computing is the business of using or providing on-demand remote computing resources, such as networks, servers, storages and services, over the Internet [11]. In its cloud model guidelines, NIST defines five characteristics, three service models and four models for deployment on the cloud, as shown in Figure 3. These guidelines can be implemented by different organizations in multiple conformant ways.

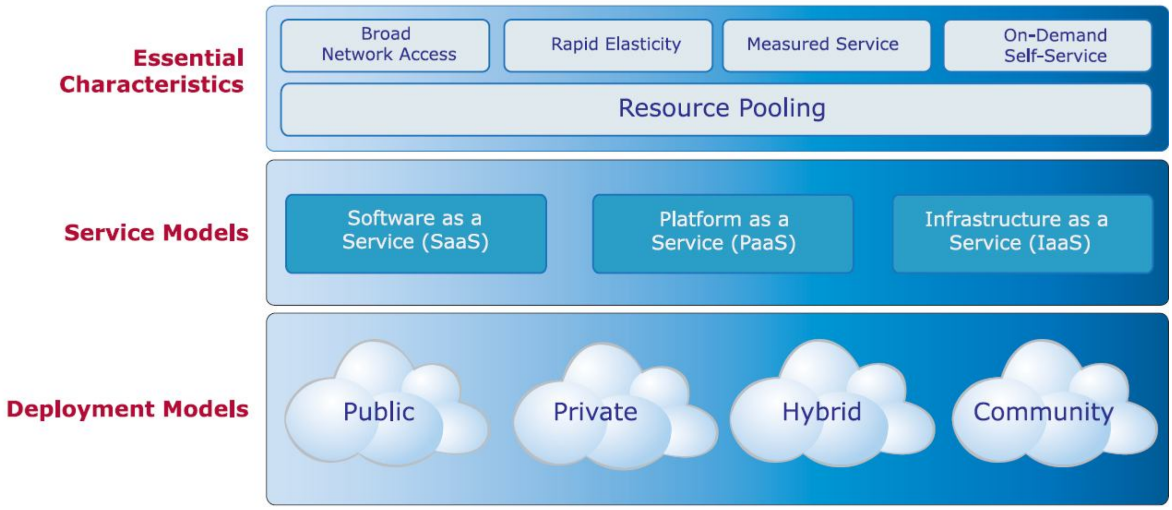

Figure 3 The NIST definition of Cloud Computing adopted from [9]

The five important characteristics identified by NIST as shown in the top layer of Figure 3 are:

1. *Broad Network Access*

Cloud services should be available though network access via heterogeneous standard mechanisms, including all kinds of platforms such as mobile phones and laptops.

2. *Rapid Elasticity*

Cloud services should appear like unlimited resources that can be quickly and elastically provisioned to scale out and be quickly released to scale in.

3. Measured Service

Cloud service usages should be automatically measured, monitored and controlled in a transparent way for both cloud service consumers and providers.

4. On-Demand Self-Service

Consumers should be able to automatically scale out computing resources such as service time and storage requirements, by themselves without having to directly contact the cloud service providers.

5. Resource Pooling

Cloud services should be able to serve multiple consumers according to their demand. Services should also be location-independent, in the sense that consumers should not be aware of any physical location of cloud services, except at higher level of abstraction, such as country, state or datacenter, under certain conditions.

The three Service Models identified by NIST as shown in the middle layer of Figure 3 are:

1. Software as a Service (SaaS)

Cloud service consumers use software applications of cloud service providers running on a cloud infrastructure through different thin client interfaces, which are often web browsers. Consumers do not manage cloud services (including cloud platforms and infrastructures), except for some user-specific application configuration possibilities under certain conditions. Microsoft's Office 365 [12] provides all the familiar office tools as a service through a network. Salesforce.com [13] provides customer relation management tools and capabilities as a service through the Internet. Both Office 365 and Salesforce.com can be considered as examples of SaaS.

2. Platform as a Service (PaaS)

Cloud service consumers deploy their software applications developed in a compatible way with the cloud provider's platform in the cloud infrastructure. Consumers have control over their deployed software applications and possibly some deployment configuration settings. Consumers have no control over the cloud infrastructures. Google App Engine [14] is a typical example of PaaS. It allows cloud consumers to build their web applications by providing runtime environments to maintain and automatically scale out or scale in their cloud usage according to their traffic needs, without having to worry about the management of platforms and servers.

3. Infrastructure as a Service (IaaS)

Cloud service consumers are able to deploy any software application and underlying platforms including operating systems in the cloud infrastructure. Organizations using the IaaS service model have no control over the cloud infrastructures. The Amazon Simple Storage Service (Amazon S3) [15] provides elastic data storage service for cloud consumers. It provides a web service interface that can be used to offload and download data to and from the Amazon cloud infrastructure. It allows developers to automatically scale out and scale in their data storage requirements. Amazon S3 also provides data security, management and usage restriction facilities. Amazon Elastic Compute Cloud (Amazon EC2) [16] provides a virtual elastic computing environment. The Amazon EC2 provides developers with a virtual computing environment and tools for developing their applications. Developers can select from the existing variety of Linux distributions or Windows operating systems, or they can also upload an operating system of their choice using the provided supporting tools. Amazon S3 in combination with Amazon EC2 can be considered as an example of IaaS.

The four deployment models identified by NIST as shown in the bottom layer of Figure 3 are:

1. Public cloud

Cloud services are provided publicly to anyone. These services are managed by the cloud service provider organization.

2. Private cloud

Cloud services are provided exclusively to a single organization. This type of cloud service may be placed in the cloud or on the premise of the organization. Private cloud services may be managed by the organization or by a trusted third-party.

3. Community cloud

Cloud services are provided exclusively to a certain group of organizations with common interests, such as domain, cooperation and goals. These services may be managed by the organizations themselves or by a trusted third-party.

4. Hybrid cloud

Cloud services are provided to a group of organizations. At the same time, some of these services are exclusively owned by a single organization or community. Hybrid deployment model combines all the other cloud deployment options.

2.2 BPM

Business Process Management (BPM) is a management approach to align business processes of an organization in an effective way according to organizational business needs. Organizations use BPM to systematically manage, continuously evaluate and improve their business processes in four iterative phases called BPM Lifecycle [1]. These phases are: design, configuration, enactment and evaluation as shown in Figure 4.

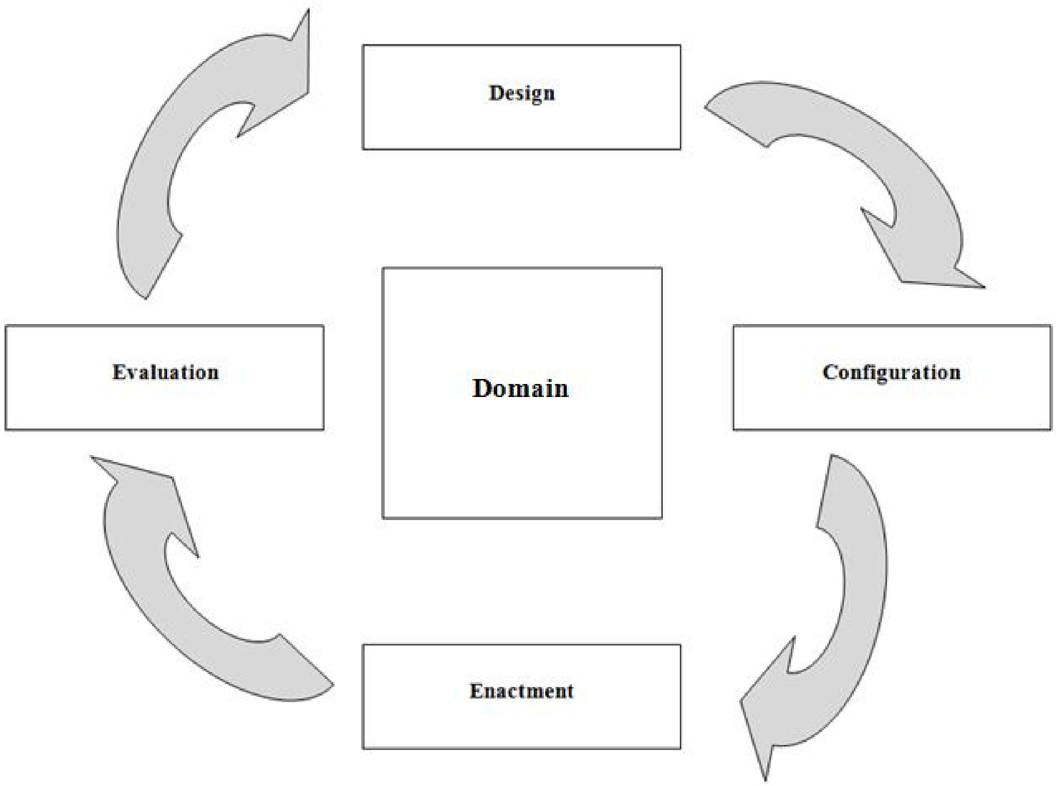

Figure 4 BPM Lifecycle

- Design

The main goal of this BPM phase is to identify business processes involved and represent them in a business process model. The operations that an organization performs and their relationships are identified, and organized in the form of business process models for a better understanding and efficient iterative improvement. Business process models are created in this phase.

The design phase is the entry point and most important part of the BPM Lifecycle. In this phase an extensive study is made in the business process domain in order to capture and model all relevant processes of the organization. These business process models are validated at this phase. The business process models identified and validated at this phase are processed in all the consecutive BPM Lifecycle phases.

- Configuration

In this phase business process models captured in the design phase are configured and implemented. Automated activities are linked to the software systems that realize them. The implementation platform, implementation language and other required implementation tools and techniques are chosen in this phase.

- Enactment

In this phase business process models are instantiated based on the configuration from the configuration phase. A process engine is used to control the coordination during enactment.

- Evaluation

The main goal of this phase is to evaluate the quality of business process models, their implementation and execution in order to improve them accordingly. In this phase available stored information such as execution logs are used to evaluate business process models and their implementations.

2.3 Cloud-Based BPM

The combination of cloud computing and BPM (Cloud-Based BPM) is an emerging remote delivery of integrated BPM technologies in a pay-per-use pricing scheme. Cloud-Based BPM can be delivered as Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS) or Infrastructure-as-a-Service (IaaS) models in the same way as the general software applications [5]. Major cloud service providers, such as IBM, Oracle and Microsoft are delivering Cloud-Based BPM.

2.3.1 Delivery as SaaS

In the Software-as-a-Service (SaaS) delivery model, the Cloud-Based BPM service providers provide everything that organizations need in order to create and execute their business processes in a pay-per-use pricing scheme as shown in Figure 5. Service providers provide organizations with the underlying hardware, operating systems, process engines, database management systems, middleware, applications and business processes.

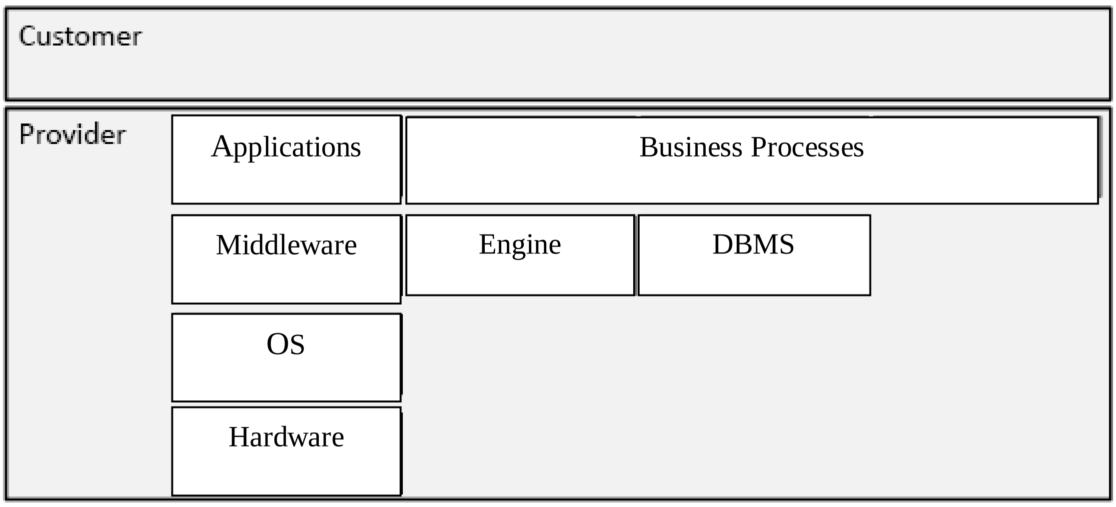

Figure 5 Cloud-Based BPM delivered in the SaaS model [5]

2.3.2 Delivery as PaaS

In the Platform-as-a-Service (PaaS) delivery model, the Cloud-Based BPM service providers provide hardware, operating systems, process engines, database management systems, middleware and other required platforms in a pay-per-use pricing scheme as shown in Figure 6. The organizations only manage their business processes and applications.

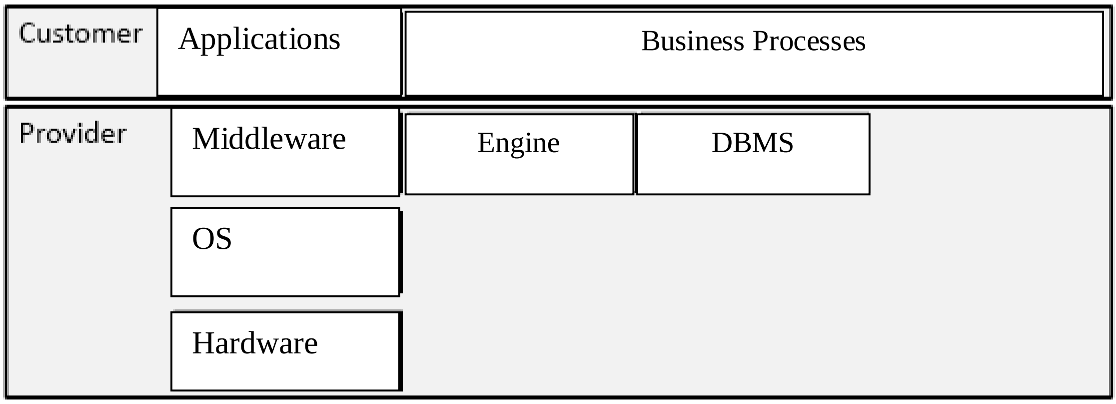

Figure 6 Cloud-Based BPM delivered in the PaaS model [5]

2.3.3 Delivery as IaaS

In the Infrastructure-as-a-Service (IaaS) delivery model, the Cloud-Based BPM service providers provide only the underlying hardware in a pay-per-use pricing scheme as shown in Figure 7. The operating systems, process engines, database management systems, middleware and other required platforms and applications are managed by the organization.

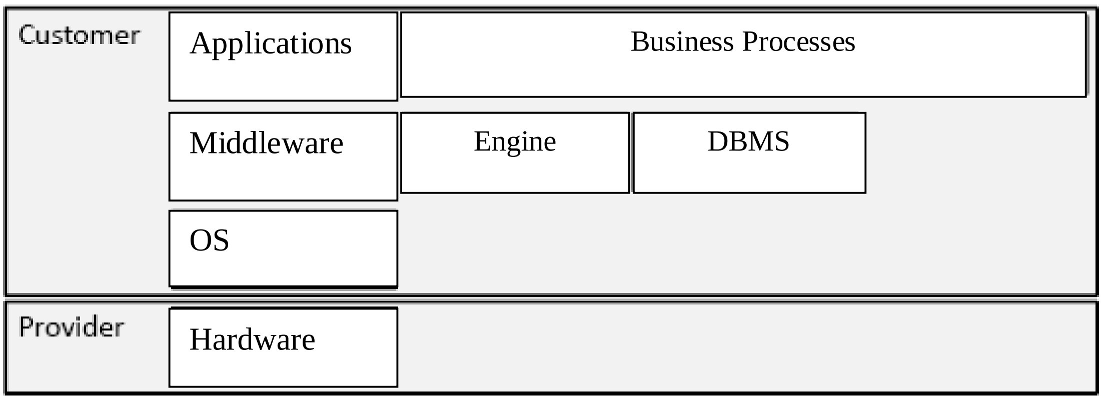

Figure 7 Cloud-Based BPM delivered in the IaaS model [5]

2.4 Benefits

Cloud-Based BPM can provide the following benefits:

1. *User-End activity and data distribution*

Recently, Cloud-Based BPM architectures with user-end data and activity distribution has been proposed to help organizations place their sensitive data and activities on-premise and data and computation-intensive activities in the cloud [3] [17].

2. *Business Case Transformation*

Cloud-Based BPM gives SMEs (small and medium enterprise) the opportunity to access business process solutions developed by experienced process experts available in the cloud in an affordable way.

3. *Business Process Outsourcing(BPO)*

Cloud-Based BPM gives opportunity for organizations with a special process expertise to develop cloud-based business process solutions and sell their expertise over the Internet.

4. *Rapid Prototyping and Try Before You Buy*

Cloud-Based BPM gives organizations the opportunity to conduct rapid prototyping and testing of business process solutions before buying and installing on-premise.

5. *Extending Business Process to Mobile Devices*

Cloud-Based BPM gives the opportunity for vendors to provide collaborative business process solutions on any device over the Internet.

2.5 Challenges

Cloud migration challenges which are common across all data and computation-intensive systems to cloud migration decision-making, such as go/no-go and service provider selection and best cloud service combination, still exist in Cloud-Based BPM. Furthermore, a number of data and computation-intensive business processes are organizational strategic information. As a result, organizations are unwilling to migrate their overall BPM to the Cloud-Based BPM because of fear of unintentionally disclosing this information.

A new architecture has been recently proposed for cloud-based BPM that allows organizations to deploy their sensitive data and activities on-premise and computation-intensive data and activities in the cloud [3]. However, there are still limitations of the techniques that guide the decision to safely migrate business process activities and data to the Cloud-Based BPM. Activity and data distribution recommendation techniques that can guide decision makers regarding the placement of total or parts of a business process in the cloud considering multiple factors, such as cost, privacy and performance are required.

BPM architectures are classified into four patterns based on a PAD model in [3], where P stands for Process Enactment, A for Activity Execution and D for Data Storage, as shown in Figure 8. Pattern 1 represents the traditional standalone BPM, where process enactment, activity execution and data storage are all on-premise. Pattern 2 represents a User-End BPM with Cloud-Side distribution where organizations have full-fledged on-premise process engine with an option to distribute some computation-intensive activities execution and data storage in the cloud. In this pattern process enactment is totally on-premise. Pattern 3 is a Cloud-Based BPM with User-End distribution where the process engine is in the cloud with an option to distribute sensitive data storage and activities execution on-premise in order to exploit the cheap cloud resources and overcome privacy related challenges. Patterns 2 and 3 consider two process engines, one on-premise and one in-cloud, in order to allow organizations to host non-computation-intensive sensitive data and execute business process activities on-premise and computation-intensive data and activities in the cloud as shown in Figure 9. Process enactment is in-cloud. Pattern 4 represents a Cloud-Based BPM, where process enactment, activity execution and data storage are totally in-cloud.

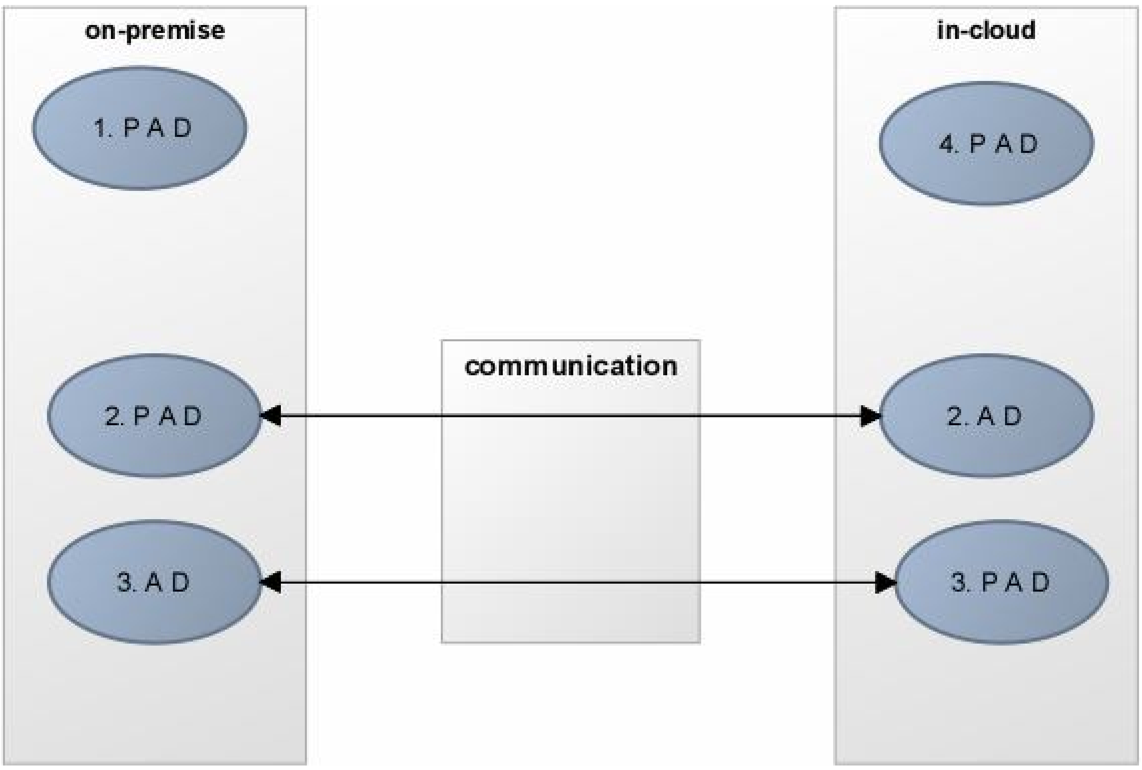

Figure 8 Patterns of BPM architectures

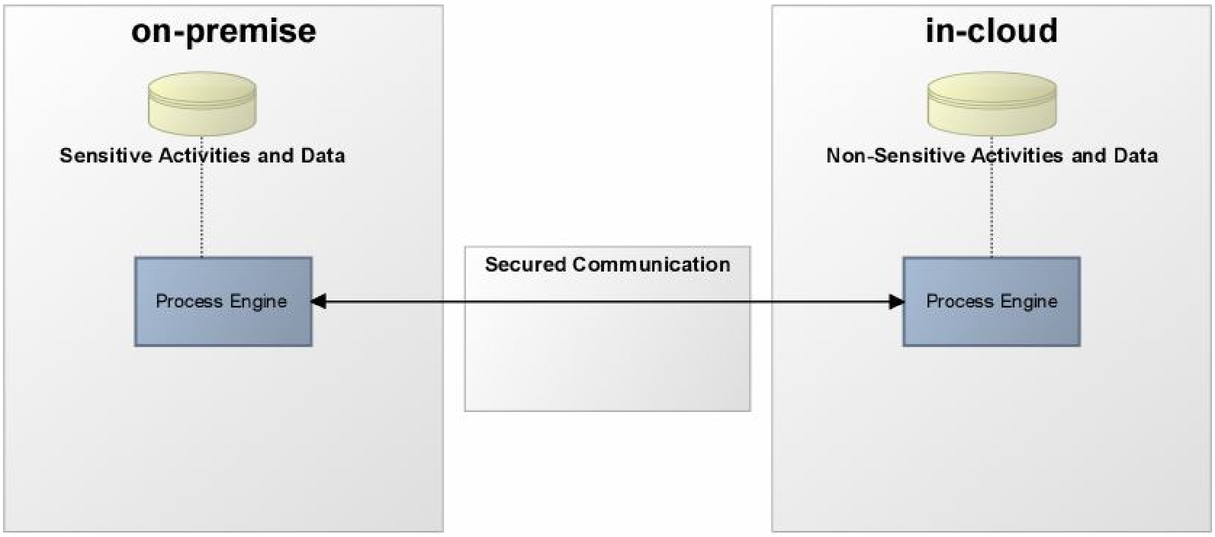

Figure 9 Cloud-based BPM with on-premise distribution

The Cloud-Based BPM with user-end distribution of non-computation-intensive data and activities shown in Figure 9 handles security and privacy related issues. This architecture contains two process engines, one on-premise and one in-cloud, and two repositories, one on-premise and one in-cloud. Non-computation-intensive activities can be executed on the on-premise engine and non-computation-intensive data can be stored in the on-premise repositories. Computation-intensive activities can be executed on the in-cloud engine and computation-intensive data can be stored in-cloud repositories.

Nevertheless, this architecture poses activity and data distribution decision challenges in which multiple cloud factors have to be considered. Different distributions can determine the benefits that an organization can gain from this new Cloud-Based BPM with user-end sensitive data and activity distribution architecture.

The designers of the Cloud-Based BPM architecture in Figure 9 proposed an optimal distribution mathematical model based on three cost factors: time cost, monetary cost and privacy cost. However, this approach is not suitable in most realistic situations. Because:

1. The authors did not formally relate the cloud-related information regarding these three factors to business processes. Therefore, automating their approach requires extra research.
2. The recommendation mathematical model can be used to propose a single optimal distribution list. However, organizations are more interested in automatically evaluating multiple distribution options, compare them based on different factors and select one or more feasible distributions according to their company goals. The choice of all possible distributions in order to select one or two distributions can lead to a very large search space when the business process contains a large number of activities and a large number of data items. However, this large search space can be limited by using different techniques and constraints, for example, to consider only the

distributions with a certain cost or with a certain privacy requirement. Algorithms can also be used to reduce this large search space into pair-wise comparisons of distributions.

3. The approach is only based on cost calculations. However, in real-life situations privacy risk may not be effectively evaluated using cost calculation techniques.

In order to support Cloud-Based BPM with a decentralized architecture, an approach to systematically split on-premise monolithic business process model into on-premise and in-cloud data and activities was proposed in [17] as shown in Figure 10. However, the approach considers that the actual on-premise and in-cloud distribution list has been defined somehow beforehand and concentrates on the decomposition of the original business process.

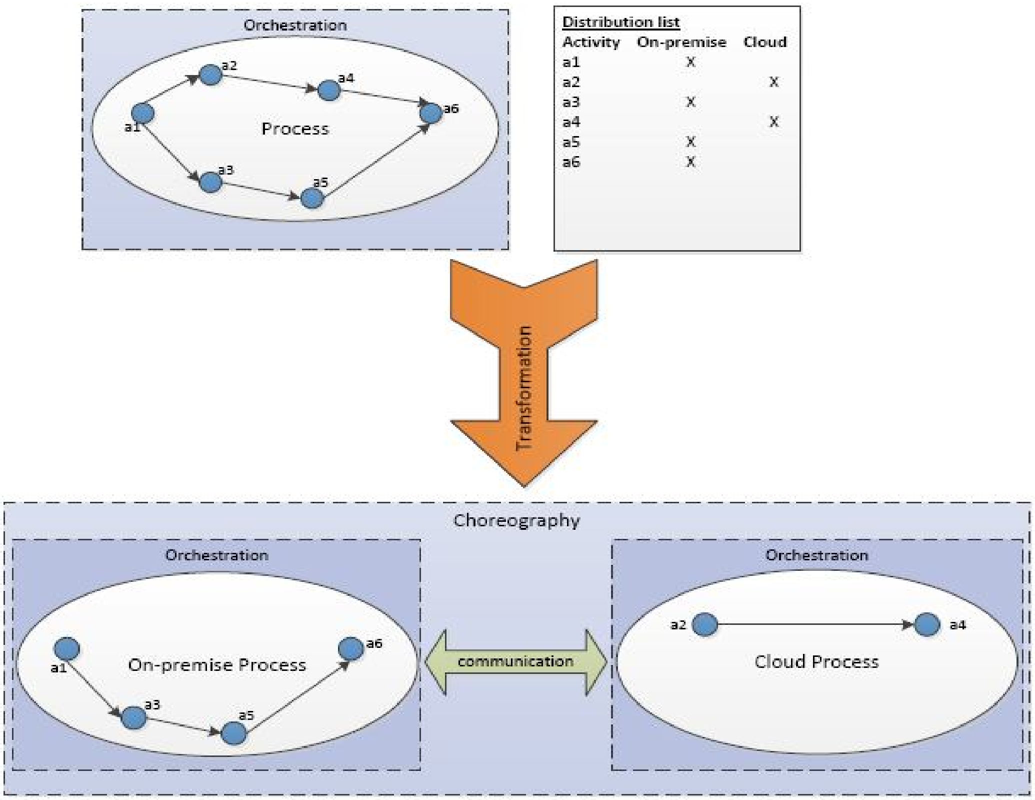

Figure 10 An approach to systematically split one monolithic business process model into on-premise and cloud-side data and activities adopted from [17]

Chapter 3: Cloud Migration

The purpose of this chapter is to give background information on the techniques that guide the decision to migrate to the cloud.

This chapter is further structured as follows: Section 3.1 introduces cloud migration decision support systems and their classification; Section 3.2 discusses the techniques that can be used to guide general cloud migration decisions; Section 3.3 discusses the techniques that can be used to guide business process cloud deployment and outsourcing; Section 3.4 explains techniques that can be used to support general IT outsourcing decisions and Section 3.5 summarizes this chapter.

3.1 Cloud Migration Decision Support

Cloud migration decision analysis involves many trade-off decision-making difficulties for organizations [6] [18]. The most relevant decision difficulty to our work, however, is the identification of the distributions of activities and data items into collaborating on-premise and in-cloud engines for deployment and execution, considering multiple factors, such as cost benefit, privacy, performance, scalability, availability and security issues. While generic optimization and decision-making approaches that can be used to construct generic decision support frameworks exist, the application of these techniques in the context of cloud migration decision analysis is yet an important open research question [19].

We classify the existing techniques in the literature that can be used to guide the decision to migrate to the cloud in three groups and discuss them with examples:

1. Techniques that can be used to guide the decision to migrate software systems to the cloud in general. These techniques are discussed in [6], [9], [18], [20], [21], [8], [22] and [23]. Since these techniques are not used in the context of business processes, additional research is required in order to apply them to cloud-based BPM.
2. Techniques that can be used to guide the decision to deploy business process activities and data in-cloud or on-premise, and those that can be used to outsource business process activities and data in general. These techniques are discussed in [3], [10] and [7]. Since these techniques are used in the context of business processes, they can be applied to cloud-based BPM with less effort than the techniques in category 1. However, additional research is required in order to automate these techniques since the authors did not formally relate their factors to business processes.
3. Techniques that can be used for IT outsourcing in general. An example of such techniques is discussed in [24]. In the traditional IT outsourcing, organizations take complex decisions to outsource activities that require IT skills to other organizations. Nowadays, cloud sourcing is replacing this traditional IT outsourcing. Therefore, we consider the traditional IT outsourcing, relevant to this work. Since these techniques are applied in the context of business processes, we

expect they can be applied to cloud-based BPM with less effort than category 1. However, since these techniques are not used in the context of cloud computing, we expect that more research is required than with the techniques in category 2 in order to apply them to cloud-based BPM.

3.2 General cloud migration decisions

These techniques can be used to guide decisions to migrate all kinds of on-premise systems to the cloud. These frameworks, tools, models, checklists and questionnaires are not devoted to support business process activities and data distribution analysis, which is of interest for this work. However, we expect that with additional research and improvements, these techniques can be customized to be used in the context of business process activity and data distribution decision analysis for cloud deployment.

3.2.1 $(MC^2)^2$ framework

A generic multi-factor-based framework called $(MC^2)^2$ that can be used in the context of cloud adoption decision-making is discussed in [6]. The process steps of the generic $(MC^2)^2$ cloud adoption framework are shown in Figure 11.

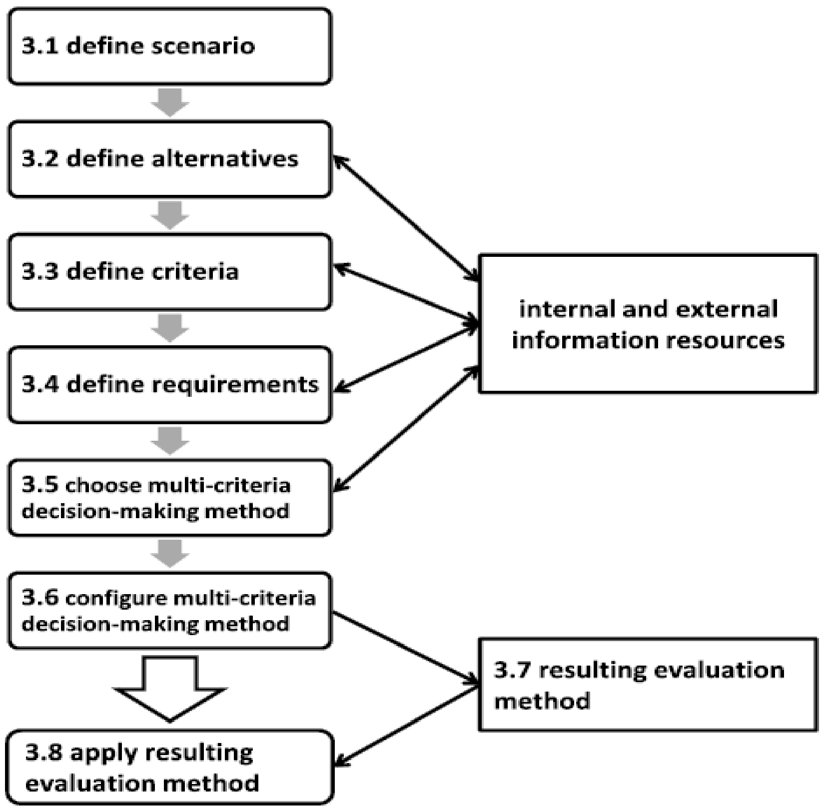

Figure 11 Process Steps of the generic $(MC^2)^2$ cloud adoption framework

A typical decision-making process based on $(MC^2)^2$ involves the following 8 concrete steps. We provide a short explanation and a practical example for each of these steps. In all of these steps our example is a cloud-email [25]. We discuss what an organization that would like to benefit from cloud-email solutions can do in each step of the $(MC^2)^2$framework regarding this decision.

1. Define scenario

In the $(MC^2)^2$ framework defining scenario is the initial step. In this step the particular cloud adoption decision situation and organizational goals are described. For example, in the cloud-email example the particular scenario is offloading on-premise email systems to the cloud. In this step, the organization should define the main goal of the cloud-email scenario, such as cost benefit, better user experience or accessibility.

2. Define alternatives

The second step of this generic framework is to define alternatives. For example, in the cloud-email scenario, the first alternative might be migrating their total on-premise email system to the cloud, and the second alternative can be migrating parts of their email system, such as the email archiving or emails of specific departments of the organization.

3. Define criteria

In the $(MC^2)^2$ framework, the third step is to define criteria. These criteria might be quantitative or qualitative in nature and they may have positive or negative influence on the achievement of the overall cloud adoption organizational goal. For example, in the cloud-email scenario, these factors can be cost benefit, privacy, performance, flexibility, scalability, availability, accessibility, and security and integration issues.

4. Define requirements

Define requirements is the fourth step in this framework. These requirements are used to filter out alternatives that are not realizable under the given criteria in the scenario under consideration. For example, in the cloud-email scenario, if email messages of a certain department must be handled on-premise, the messages of this particular department are filtered out of the candidate alternatives identified in step 2.

5. Choose an appropriate Multi-Criteria Decision-Making technique

The fifth step is to choose an appropriate Multi-Criteria Decision-Making technique. This technique has to be chosen according to the defined scenario, organizational and technical preferences from the set of eligible Multi-Criteria Decision-Making methods, such as Analytic Hierarchy Process and Analytic Network Process [26]. For example, in the cloud-email scenario, the organization can select analytic hierarchy process (AHP) or analytic network process (ANP) [26]. AHP is chosen when criteria can be pair-wise compared with respect to the goal of the decision making, independent of the alternatives. ANP is usually chosen when decisions and comparison involve dependencies and feedback. In the cloud-email scenario, the alternatives migrating total email system and migrating parts of the email system can be compared pairwise in terms of cost benefit. This pairwise comparison can be repeated independently in terms of each of the identified criteria in step 3.

6. Configure the Multi-Criteria Decision-Making method

The sixth step in the provided framework is to configure the chosen Multi-Criteria Decision-Making method. In this step the relevant factors, such as criteria, alternatives, requirements, weights and relations are set. For example, if the chosen multi-criteria decision-making technique is AHP in the cloud-email scenario, then the relative importance of the criteria, such as cost benefit, privacy and user experience, from the organizations perspective, the relative priorities of the alternatives with respect to each criterion should be configured at this step.

7. Execute Evaluation Method

The seventh step in the $(MC^2)^2$ framework is to execute the evaluation method as shown in Figure 12. Appropriate alternatives are chosen from the available total alternatives according to the requirements defined in step 4. Further evaluation is conducted on the remaining alternatives according to the criteria using the chosen multi-criteria decision-making technique. An optimal decision according to the overall decision situation is the alternative ranked first. For example, in the cloud-email scenario the purpose of this step would be to rank the alternative email systems and sub-systems that can go to the cloud using AHP. The email-system or sub-system ranked first is the optimal candidate according to this technique.

8. Select Result

The final step in this framework is to select appropriate alternatives, and the alternatives that cannot be realized are filtered out from the ranked list of alternatives in the previous step. For example, in the cloud-email scenario, an alternative that cannot go to the cloud for technical reasons identified after conducting step 7 is removed from the ranked candidate list.

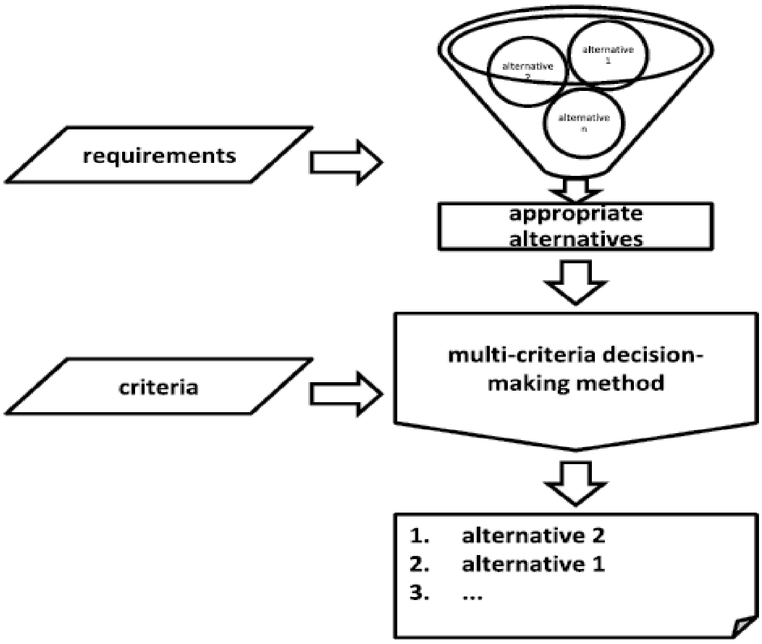

Figure 12 The Process of Executing Evaluation Method

3.2.2 CloudGenius framework

A multi-criteria decision support framework called CloudGenius that can be used to find the best combination of cloud infrastructure for a web server is presented in [18] . CloudGenius is developed based on the $(MC^2)^2$ generic framework discussed above and shown in Figure 11. The decision-making process steps of CloudGenius are the same as that of $(MC^2)^2$. However, CloudGenius is tailored to cloud infrastructure selection for a web server.

This framework was applied to an e-business company scenario that has been using an on-premise web server for years. The company decided to use cloud infrastructure to reduce maintenance costs. The e-business company has a scalable web application developed in PHP that requires the data to be stored on-premise. CloudGenius is applied to this scenario as follows:

1. Data migration will not be considered since the web application of the company needs on premise data storage.
2. Requirements, such as PHP support and Windows operating system are selected. Selection criteria are also selected, such as cost and latency.
3. Weights are assigned to each of these factors.
4. CloudGenius is initialized and suggests the available windows-based cloud VM images that support PHP from Amazon. However, Windows operating system is found to be incompatible.

5. The decision-makers of the company can go back and select different operating system other than Windows and restart CloudGenius, and redo this process until they find a compatible cost-effective combination.
6. Finally the actual migration of the local web servers is conducted.

3.2.3 Value estimation framework

A framework that can support cloud migration decision-makers by estimating the value of cloud computing is discussed in [21]. The authors identified key components of economic and technical aspects that should be considered during cloud migration, and structured them in a decision-making framework. Decision-makers can evaluate a particular business scenario and calculate the estimated cloud computing costs with this framework. The main goal of the framework is to conceptually classify general business scenarios that are suitable for cloud solutions.

The authors applied this value estimation framework to a real-life project called TimesMachine. The aim of TimesMachine is to provide access to 4 Terabytes of data in a PDF format which requires a massive computing power and data storage. With the help of the value estimation conceptual framework, the developer of TimesMachine decided to use Amazon's EC2 and S3 cloud services. According to the framework the main drivers in this scenario to go to the cloud include: simplicity of cloud solutions, no upfront cost, speed to convert the 4 Terabytes data PDF in only 36 hours, which might take too long on-premise and the one-time nature of the TimesMachine project.

3.2.4 IaaS migration tools

Two decision support tools that can be used to guide organizations during the migration of IT systems to public Infrastructure-as-a-Service (IaaS) clouds are described in [20]. These tools can be used to inform decision makers about the costs, risks and benefits of using public IaaS clouds. The first tool allows business organizations to model their software, data and resources to produce cost estimates of different IaaS cloud providers. The second tool is a spreadsheet that contains a table of risks and benefits that can support organizations to conduct IaaS cloud risk/benefit assessments. Organizations can rate the risks and benefits in the spreadsheet with respect to their organizational goals.

The authors applied these tools to the case study digital library and search engine called CiteSteer^{x}. The system contains the following on-premise service components: web application interface, document management service, maintenance service and data backup services. The system contains over 1.5 million documents requiring about 2 TB data storage, and receives about 2 million hits per day from visitors.

1. Cost modeling

The cost modeling tool was used to model the resource requirements of the system. Different cloud providers, including Amazon, FlexiScale and Rackspace were compared in terms of cost for the period of 3 years, and Amazon was found to be the cheapest.

2. Benefits and Risk Assessment

The risks/benefits spreadsheet was used to identify risks and benefits of migrating the digital library to the chosen cloud service provider, in this case, Amazon.

3.2.5 Cloudstep

A step-by-step cloud migration guide called Cloudstep that can help organizations assess risks and benefits when migrating on-premise generic software systems to the cloud is presented in [8]. The approach describes the organization, the on-premise software systems and the target cloud providers by using template-based profiles for analysis purposes. Cloudstep involves nine activities that should be performed according to a workflow shown in Figure 13.

We provide a short explanation and a practical example for each of these activities. In all of these activities, we use the cloud-email scenario [25]. We discuss what the activities would mean in the cloud-email scenario regarding cloud migration decisions by using the Cloudstep decision process.

1. Defining organizational profile

In the Cloudstep decision process, the first activity is defining organizational profile. In this activity, the organizational drivers that motivated the cloud migration process are identified. For example, in the cloud-email scenario, the main motivation for adopting the cloud-email, such as cost benefit and better user experience are identified in this activity.

2. Evaluating organizational constraints

The second activity is evaluating organizational constraints. In this activity, organizational aspects that can prevent cloud migration are identified. For example, in the cloud-email scenario, the fact that emails of some departments cannot go to the cloud due to privacy is identified in this activity.

3. Defining application profiles

The third activity is defining application profiles. In this activity, application level aspects that can affect the cloud migration process are identified. For example, in the cloud-email scenario, the application level characteristics of the email system, such as the number of users, number and nature of the email components, configuration preferences, software licenses and traffic are identified in this activity.

4. Defining the target cloud provider's profiles

The fourth activity is defining the target cloud provider's profiles. In this activity, aspects of the target cloud provider that can affect the cloud migration decision are identified. For example, in the cloud-email scenario, a specific target cloud service provider is selected based on the criteria identified in the first activity, such as Amazon or Google Apps. The type of service model, deployment models and prices are identified. The operating systems, file formats, supported protocols, availability and available support services are also identified in this activity.

5. Evaluating technical and financial constraints

The fifth activity is evaluating technical and financial constraints. In this activity, the organizational profiles, application profiles and service provider profiles are cross-checked for conformance. For example, in the cloud-email scenario, the cloud-email organizational profile, cloud-email profile and the profile of the chosen cloud service provider are cross-checked to identify inconsistencies.

6. Addressing the application constraints

The sixth activity is addressing the application constraints. In this activity, application constraints identified from the fifth activity are resolved. For example, in the cloud-email scenario, the application profile constraints identified in activity 5 might be improved by changing some of the constraints from the application profile.

7. Changing a cloud provider

The seventh activity is changing a cloud provider. In this activity, available cloud service providers are checked in the effort of choosing a cloud profile that addresses the identified constraints in activity 6. For example, in the cloud-email scenario, the provider profile constrained identified in activity 5 might be improved by changing the cloud service provider.

8. Defining migration strategy

The eighth activity is defining migration strategy. In this activity, a cost-effective migration strategy is defined. For example, in the cloud-email scenario, a single-tenant strategy, in which only one organization would use the email system or a multi-tenant strategy, in which multiple organizations can use the system are defined in this activity.

9. Performing the actual migration

The ninth and final activity is performing the actual migration. In this activity, the actual migration of the on-premise system to the chosen cloud provider is performed according to the chosen migration strategy. For example, in the cloud-email scenario, the actual migration can be taken according to the chosen strategy in activity 8, if the organization agrees with the overall results.

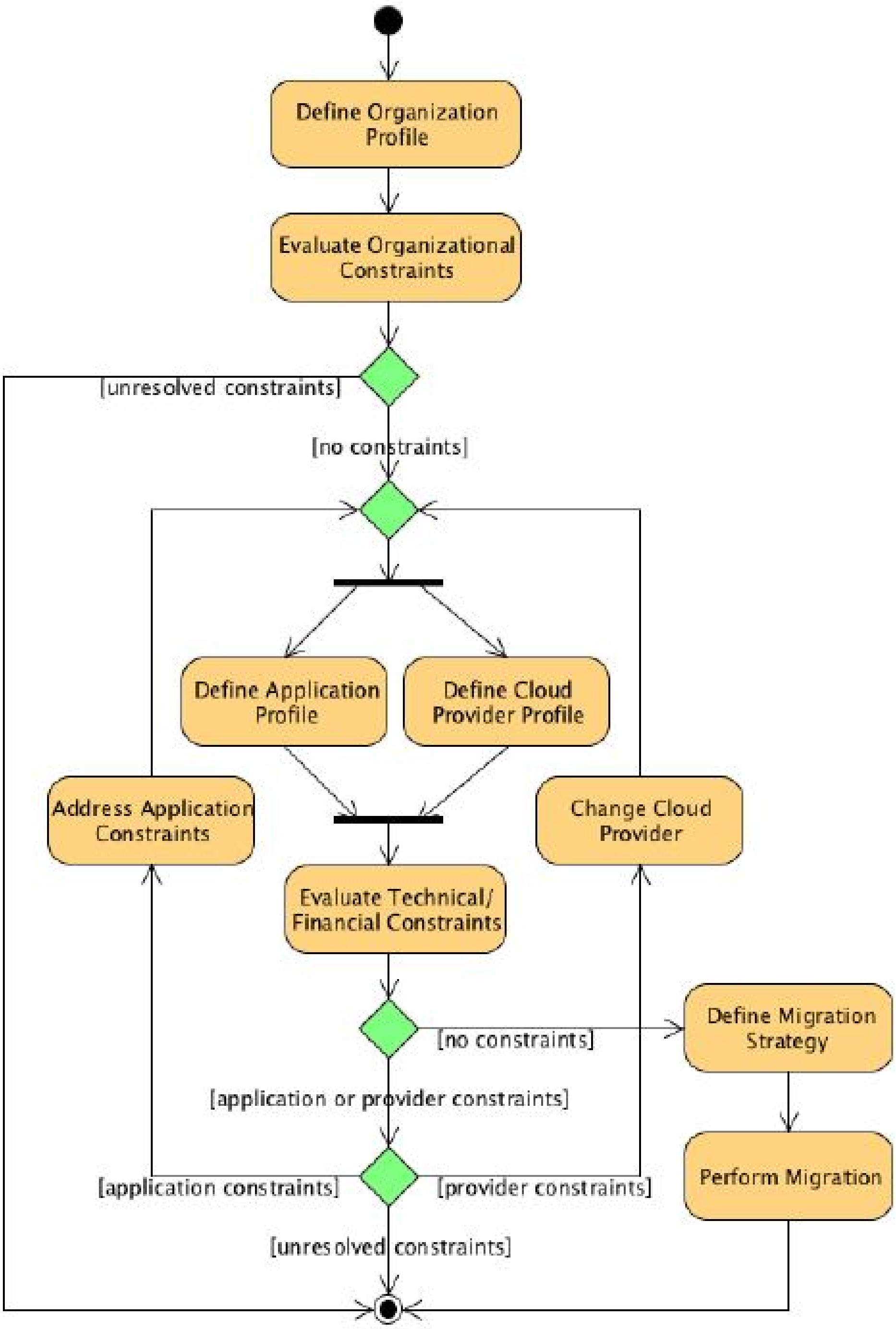

Figure 13 the complete workflow of Cloudstep cloud migration decision process

3.2.6 Questionnaires

Cloud migration guides based on long list of risk/benefit questioners that can be used to assess a specific cloud factor, such as cost, privacy, security and other broad range of factors are discussed in [9] and [22]. For example, in the cloud-email scenario used above, the organization needs to complete a matrix of list of risks/benefits and rate them as H/M/L, where H stands for high, M for medium and L for low risk or benefit as shown in Table 1. This matrix has to be filled in for all of the factors that can affect the cloud-email migration process. The cloud-email go/no-go decision can then be made based on the comparison of benefits and risks level.

Questionnaires	**Answers**	**Benefit (H/M/L)**	**Risk (H/M/L)**
1. How is the cost of the organization affected in the cloud-email?			
2. What security requirement is satisfied by the cloud-email service provider?			
3. Which parts of the email can be migrated to the cloud-email service provider?			
4. How are the cloud-email service provider's service level agreements (SLAs) when compared with the on-premise SLAs?			

Table 1 Questionnaire based cloud migration decision guide for the cloud-email scenario adopted from **[22]** and **[9]**

3.2.7 Holistic analytical model

A holistic analytical model for making cloud migration decisions with a special attention to security, availability, business economics and broad cloud migration concerns is discussed in [23]. For example, in the cloud-email scenario, the business considerations of the cloud-email, such as cost related issues, are studied carefully. Security and availability considerations of the cloud-email should also be studied. A trade-off decision can then be made based on the study results from these three dimensions.

The authors also applied this analytical model to the case study of a company that was considering to outsource storage resources to the cloud. Two cloud service providers were evaluated for cost, security and privacy factors by using their models. The results have shown that evaluating security and availability in addition to cost gives an extra level of confidence to cloud migration decisions.

3.3 *Business process cloud deployment and outsourcing decisions*

The techniques in this category are specifically dedicated to business process activities and data, and are the most relevant to this work. While there are many research papers on the techniques that guide general cloud migration, we could not find much research dedicated to business process activity and data distribution analysis for deployment in the cloud or on-premise.

3.3.1 Optimal activity and data distribution model

A mathematical model that can be used to recommend an optimal activity and data distribution list based on monetary cost, time cost and privacy risk cost was proposed in [3], as discussed in Section 2.5. Due to large search space problems, the authors reduced the number of activity and data distribution options by observation and using heuristic methods. The model is used to select one optimal distribution option based on cost calculations. The model does not allow organizations to consider multiple distribution options and compare distributions in terms of multiple factors that cannot be effectively evaluated using cost calculations. Furthermore, the authors did not discuss how the model can be integrated with business process models to automatically generate multiple distribution options. For example, consider an e-commerce company selling products online and runs using on-premise BPM to manage its business. This company would like to benefit from cloud-based BPM by outsourcing some of its activities or data. This decision model can be used to recommend one optimal distribution of activities and data for the e-commerce company based on time cost, monetary cost and privacy risk cost calculations. However, this work does not help this e-commerce company compare different business process distribution options and take trade-off decisions.

3.3.2 Developing a decision model for business process outsourcing

A decision model that can be used for business process outsourcing has been developed based on AHP prior to the existence of cloud computing in [10]. However, this decision model can be applied to cloud computing with some modifications. The authors classified the factors affecting business process outsourcing into core competency, risk factors and environmental perspective. During the development of the decision model, two consecutive surveys were conducted [10]. The first survey was to identify the three categories of main decision determinants, and the second survey was to identify the relative weight of these determinants. The decision situation had three alternatives: (i) outsource process (ii) maintain process and (iii) maintain and modify process. For example, consider an e-commerce company selling products online and runs using on-premise BPM to manage its business. This company would like to benefit from cloud-based BPM by outsourcing some of its activities or data. This decision model can be used to decide if the company should outsource the whole business process, maintain the current BPM, or maintain and modify some of the processes. The factors that can be considered during this decision include cost benefit, privacy and availability.

3.3.3 Business activity outsourcing

A Decision-Making Framework for Business Activity Outsourcing was proposed in [7]. The decision contains two purposes: select business activities to outsource and select the best combination of vendors and contracts to adopt. The authors suggested criticality, stability and simplicity as the three most determinant factors for deciding on IT outsourcing, and proposed the decision hierarchy structure shown in Figure 14. In the decision hierarchy structure, the overall goal is to outsource the business activity. The factors are criticality, stability and simplicity. Expert Choice [27] is used for calculating the activities priorities and overall ranking. For example, consider an e-commerce company selling products online and runs using on-premise BPM to manage its business. This company would like to benefit from cloud-based BPM by outsourcing some of its activities or data. This decision support framework can be used to select some of the activities or data to outsource to the cloud. The goal of this decision is to select activities or data for cloud. The alternatives are the set of activities and data in the product selling business process model of the e-commerce company. The factors to be considered can be cost benefit, privacy and availability. Weights are assigned for each of the alternatives with respect to the factors, and to each the factors in with respect to the overall goal by experts. Expert Choice can be used to calculate the overall hierarchy of these activities based on the weights assigned. The top-ranked activities are recommended for outsourcing.

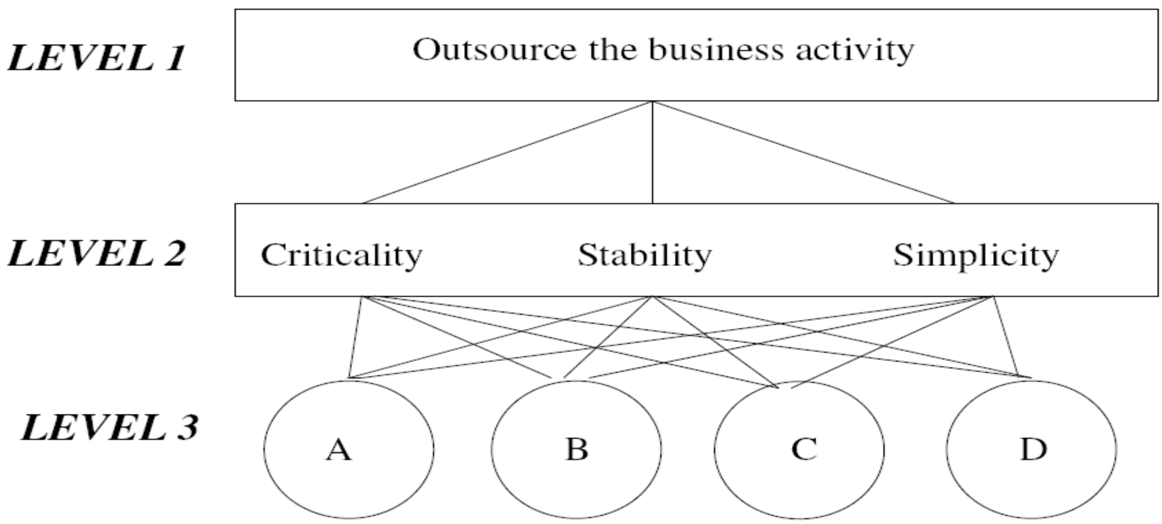

Figure 14 Decision hierarchy for selecting activities to outsource

3.4 General IT outsourcing decisions

Some decision-making frameworks and approaches have been developed before the existence of cloud computing to be used to support outsourcing decisions of IT systems. Although these techniques are not geared towards cloud computing or business process activities and data distribution analysis, they can be applied in the context of cloud migration.

An approach for developing a decision model for outsourcing general IT systems considering multiple factors can be found in [24]. The authors concluded that management, strategy, technology, economics and quality are the most critical factors that should be considered during an IT outsourcing decision, and developed a decision model as shown in Figure 15. For example, in the cloud-email scenario, the overall goal is to decide whether to adopt cloud-email solutions or not. The factors to be considered include cost benefit, privacy and availability. The alternatives are to adopt cloud-email and to keep on-premise. Weights are assigned to each of the alternatives with respect to the factors, and to each of the factors with respect to the overall goal by experts. Tools, such as Expert Choice, can be used to calculate the ranking of the two alternatives.

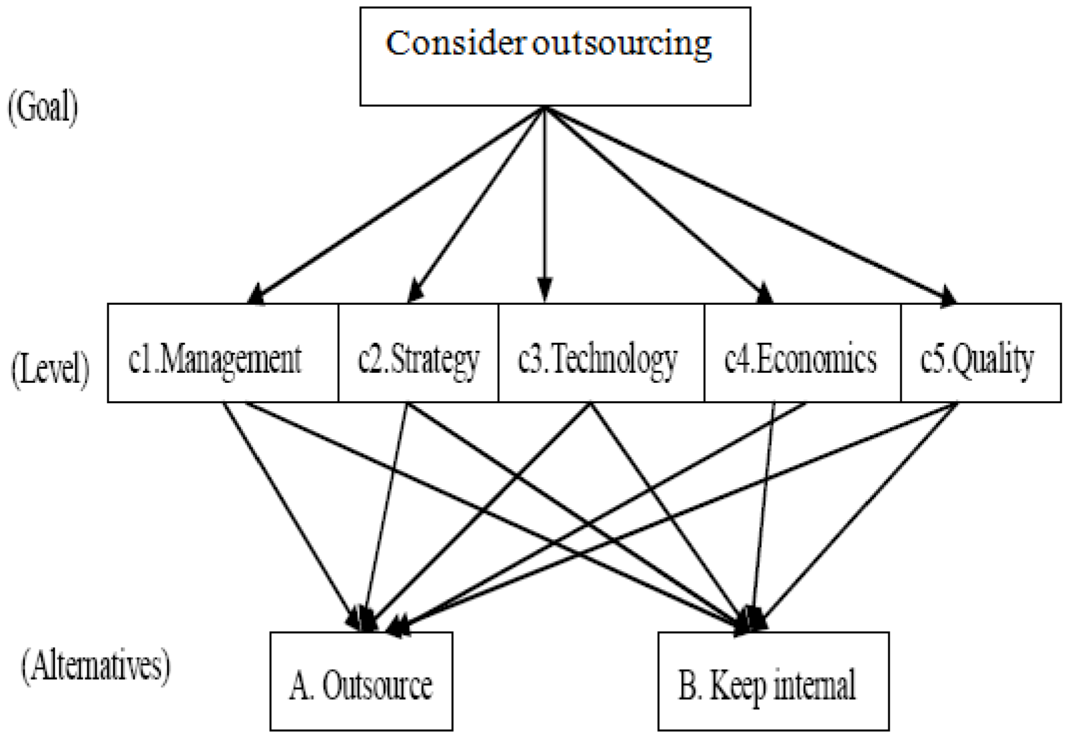

Figure 15 Decision Structure of IT Outsourcing

3.5 Summary

Our main conclusions concerning the literature on the decisions to outsource business process activities and data to cloud-based BPM are:

1. There has been only scarce research dedicated to the identification and ranking of distribution options of business processes for cloud-based BPM. Most of the research addresses decision techniques that can be used to migrate general IT systems to the cloud.

2. The existing techniques discussed in Section 3.3 do not provide business people with a realistic set of distribution options and their consequences.
3. The decision techniques are not formally related with business process models, and cannot be used by business organizations in model-based automated decisions.

Chapter 4: Cloud Migration Drivers and Barriers

The purpose of this chapter is to identify the most relevant cloud migration drivers and barriers that affect the identification and ranking of distribution options of business processes into collaborating in-cloud and on-premise engines and their consequences, briefly explain why each factor is relevant, and justify the choice of cost and privacy for further consideration.

This chapter is structured as follows: Section 4.1 discusses the approach we have taken to identify these cloud migration drivers and barriers; Section 4.2 identifies the most relevant cloud migration drivers; Section 4.3 identifies the most relevant cloud migration barriers and Section 4.4 justifies the choice of cost and privacy for further consideration.

4.1 Identification Approach

Cloud migration drivers and barriers are the factors that should be considered during the identification and ranking of business process distribution options and their consequences for Cloud-based BPM. Different organizations adopting Cloud-based BPM usually have different goals and factors to assess cloud risks and benefits. The cloud migration guides and risk/benefit checklists in literature are either too limited, for example, by considering only security [22], or too detailed in the form of questionnaires [9].

Although we do not intend to cover all cloud migration drivers and barriers in this rapidly evolving area in this work, we have identified the most relevant cloud migration drivers and barriers shown in Table 2 that are applicable to the majority of cloud adoption decisions. These factors are identified by examining over 30 scientific papers, cloud migration guides, company experience reports, public websites and blog posts in the area of cloud computing.

These resources are identified by applying keyword search to scientific research databases, such ScienceDirect, Springer, Citesteer, IEEE Xplore and the ACM Digital Library. Google's search results, such as popular blog posts and open source projects on the area of cloud computing are also considered. Some of the relevant keywords used for the search include "cloud migration factors", "cloud migration decisions", "cloud migration" and "business process outsourcing". We reviewed these scientific papers and related resources in order to identify the most relevant factors that are applicable to the identification and ranking of distribution options of business processes into collaborating on-premise and in-cloud engines and databases, and their consequences.

	Cloud Migration Factor		Main References
Migration Drivers	1	Cost benefit	[3] [20] [21] [8] [30] [31] [32] [33] [34] [35] [36] [37]
	2	Performance	[3] [8] [35] [38] [37] [39]
	3	Scalability	[21] [8] [38] [37]
	4	Availability	[21] [23] [38] [40]
	5	Maintainability	[5] [31] [35]
	6	Collaboration	[41]
Migration Barriers	1	Privacy	[3] [9] [42] [34] [19]
	2	Security	[20] [8] [23] [31] [43] [22] [34] [36]
	3	Transparency	[3]
	4	Migration complexity	[18] [31]
	5	Vendor Lock-in	[38] [40]
	6	Human factor	[9] [31]

Table 2 Cloud Migration Drivers and Barriers

4.2 Cloud Migration Drivers

Organizations adopting Cloud-based BPM can benefit from the upfront and operational cost reduction, performance, scalability, availability, maintainability and collaboration opportunities that the cloud computing provides. These benefits are briefly discussed below.

- Cost Benefit

Organizations adopt cloud computing in order to gain the cost advantages of the cheaper computational and data storage resources of cloud service providers [3] [20] [21] [8] [30] [31] [32] [34] [35].

- Performance

Organizations adopt cloud computing in order to benefit from the cost-effective high performance computing regarding the sharing of hardware, main memory and CPU by running multiple virtual machines [3] [8] [38]. However, if organizations are seeking high performance computing (HPC) comparable to an on-premise supercomputing datacenters, cloud computing may not be an option [39]. The authors have conducted comprehensive comparison between HPC platforms and Amazon EC2, and concluded that Amazon EC2 is twenty times slower than a modern HPC system. Furthermore, cloud computing performance, such as in the case of Amazon EC2, for example, regarding network and disk I/O sharing is found to be problematic.

- Scalability

The rapid elasticity and on-demand self-service characteristics of cloud computing promises to offer unlimited resources that can be quickly provisioned to scale out and be quickly released to scale in [21] [8]. This gives organizations the opportunity to allocate as much or as few resources as they need.

- Availability

The availability of cloud services is the degree to which services are operational when organizations try to access their functionality. Cloud computing organizations can access their resources at anytime from anywhere through the Internet via heterogeneous devices. While some cloud services, such as the Google Search Engine, have a better reputation regarding availability, other cloud services, such as the Amazon Simple Storage (S3), Google App Engine and Gmail, suffer noticeable service outages for technical and non-technical reasons [21] [23] [38]. Organizations can avoid a single point of failure by using cloud services from different cloud service providers.

- Maintainability

Maintainability is the ease with which an infrastructure, platform or software can be maintained. Organizations adopting cloud computing have the opportunity to outsource the maintenance of infrastructures, platforms and software to cloud service providers [5] [31].

- Collaboration

Cloud computing gives organizations the opportunity to work together on integrated common projects simultaneously [41]. For example, organizations can use Dropbox for data sharing and Google Docs to simultaneously edit and update documents. Similarly, small and large organizations can work together in order to achieve cost reduction and less administration by adopting Cloud-based BPM [44].

4.3 Cloud Migration Barriers

Organizations adopting cloud computing in order to gain all the promised benefits have to consider possible drawbacks concerning data privacy, security, transparency, migration complexity, vendor lock-in and human factors.

- Privacy

Cloud adoption can expose an organization's sensitive data to unauthorized access [3]. Organizations adopting cloud computing need to ensure that cloud service providers comply with their privacy preferences [9].

- Security

Security in the cloud is the extent to which cloud service providers guarantee security standards available for the cloud environment in order to avoid security threats. Organizations adopting cloud computing face security threats of their sensitive data depending on the deployment model being considered [20] [8] [23] [31] [43] [22]. Some of the most dangerous cloud security threats include: data breaches, data loss, service hijacking, denial of service and malicious insiders.

- Transparency

Transparency in the cloud adoption process refers to the openness and clarity of billing models, the privacy and security guarantees that cloud service providers can ensure [3].

- Migration Complexity

The actual migration of on-premise large systems to the cloud involves many technical challenges to business organizations, such as integration and dependency [18] [31]. Organizations may have to integrate cloud services with on-premise infrastructures which often lead to an expensive innovation of tools and techniques. Dependencies between the components of the on-premise system and between the services provided by the cloud service providers could lead to huge expenses during maintenance and management after cloud adoption.

- Vendor Lock-in

Organizations adopting cloud computing can face vendor lock-in. The data storage formats and interfaces in the cloud are usually proprietary, preventing organizations from extracting their data in one cloud service provider for execution or storage in another cloud service provider or on-premise [38].

- Human Factor

Organizations adopting cloud computing need to assess their expertise and how the cloud adoption affects the efficiency of their employees [9] [31]. Although human factors solely may not hinder the decision to migrate to the cloud, this assessment allows organizations to determine which existing expertise can be retired and which new expertise needs to be acquired in order to accommodate the cloud adoption. This allows organizations to anticipate the level of training their employees require in order to be ready for the cloud.

4.4 Factors for Further Consideration

While we believe that all of the cloud migration drivers and barriers identified in this chapter are applicable to cloud-based BPM, we have decided to consider one most relevant cloud migration driver, namely cost benefit, and one most relevant cloud migration barrier, namely privacy risk in order to keep this work within scope and reduce complexity. The decision to select cost benefit and privacy risk for further consideration is mainly because the main objective of cloud-based BPM architecture with an option to distribute sensitive activities and data on-premise is to benefit from the cheaper resources of cloud service providers and protect strategic information of an organization from unauthorized access [3] [17]. Cost, performance, privacy and security are also considered in previous research works, such as in [3] and [17] in the context of cloud-based BPM. However, none of these efforts have tried to formally relate these factors to business processes.

Chapter 5: Automated System Design

The purpose of this chapter is to introduce the approach we have taken in the development of our automated system for identifying and ranking distribution options and their consequences for cloud-based BPM, and discuss the design of the solution elements we have used in the development of this system.

This chapter is structured as follows: Section 5.1 introduces the approach we have taken in the development of our automated system for identifying and ranking distribution options and their consequences for cloud-based BPM; Section 5.2 investigates cloud billing models in order to identify the necessary cloud-related information regarding cost factor; Section 5.3 defines techniques that can be used to protect strategic information from unauthorized access regarding privacy risk factor; Section 5.4 discusses a metamodel of our annotation language that can be used to annotate business processes with cloud-related information; Section 5.5 introduces our model-based algorithm for identifying and ranking business process distribution options and their consequences.

5.1 Approach

Cloud-based BPM architecture allows organizations to systematically utilize in-cloud and on-premise resources, by distributing activities and data in both sides [3] [17]. This architecture assumes that collaborating on-premise and in-cloud process engines and databases are available. Assuming that each activity and data item can be distributed independently, an on-premise business process with m activities and n data items can be distributed in 2^{mn} ways for deployment in collaborating in-cloud and on-premise engines and databases [3]. These distribution options have different consequences regarding cloud migration drivers and barriers, such as cost benefits and privacy risks. For example, some distribution options may expose strategic information of the organization to unauthorized access, while others do not guarantee the lowest costs. This makes the identification of distribution options and their consequences before the actual decomposition absolutely necessary.

In a cloud-based BPM architecture, the input data for an activity deployed in-cloud engine might be the output data of an activity executed on-premise engine, or vice versa. This causes data communication between the in-cloud and on-premise process engines. For example, consider the on-premise business process model of a fictitious organization shown in Figure 16. The model contains activities A1 - A5 and a data item D1. The activities A1 - A5 represent the coordination and invocation of services from a process engine. Data item D1 is an output of activity A1 and an input for activity A2. Assume that activity A1 and some of its output data items D1 involve strategic information of the organization, and activities A2 - A5 are non-sensitive computation-intensive activities. Data item D1 is a physical data stored on-premise database management system. The organization wants to benefit from the cloud-based BPM architecture by distributing some of the activities A1 - A5 or data item D1 into collaborating in-cloud and on-premise engines, and database management systems.

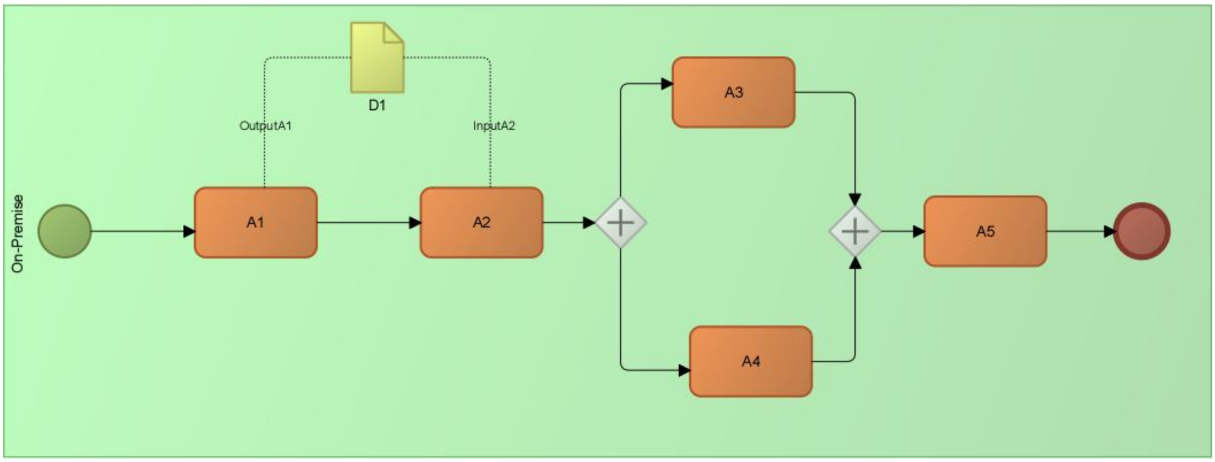

Figure 16 A Business Process Model Executed On-premise Engine

From the given scenario, one way of intuitively distributing the on-premise business process model into collaborating in-cloud and on-premise process engines, and databases is shown in Figure 17. In this work we assume that semantic correctness of the decomposition is attained once the identification and ranking of distribution options and their consequences is performed. This automated system complements the work of [17] by automatically identifying and ranking distribution options, and their consequences before the actual decomposition as shown in Figure 18. Therefore, for simplicity, call activities that might be involved in order to transfer non-sensitive output data of the on-premise activity A1 to the in-cloud activity A2 in the example in Figure 17 are not included. Figure 17 only illustrates that some communication between the in-cloud and on-premise engines is necessary after decomposition. Due to their sensitivity, activity A1 is deployed in the on-premise process engine, and data item D1 is stored in the on-premise database management system. Due to their computation-intensive nature, activities A2-A5 are deployed in the in-cloud engine. By executing activities A2-A5 in-cloud engine, the organization can benefit from the cheaper computing resources of cloud services. However, since the non-sensitive parts of the on-premise data item D1 is an input to the computation-intensive activity A2 deployed in-cloud engine, this distribution option involves extra communication steps between the in-cloud and on-premise engines in order to communicate the non-sensitive parts of data item D1 through the network. Sometimes, the cost of this communication can be higher than the benefit that the organization can gain from the cloud-based BPM architecture. Nevertheless, this is not always the case, and the benefits and risks remain to be assessed after the identification and ranking of distribution options and their consequences by using our automated system.

Annotation and analysis are two necessary consecutive steps for the users of our automated system. In order to identify and rank distribution options and their consequences of the model in Figure 16, the first step is to annotate the original model with cloud related-information and on-premise costs. The next step is to identify and rank the distribution options and their consequences, by using the annotated business process model as an input to our algorithms before the actual decomposition as shown in Figure 18.

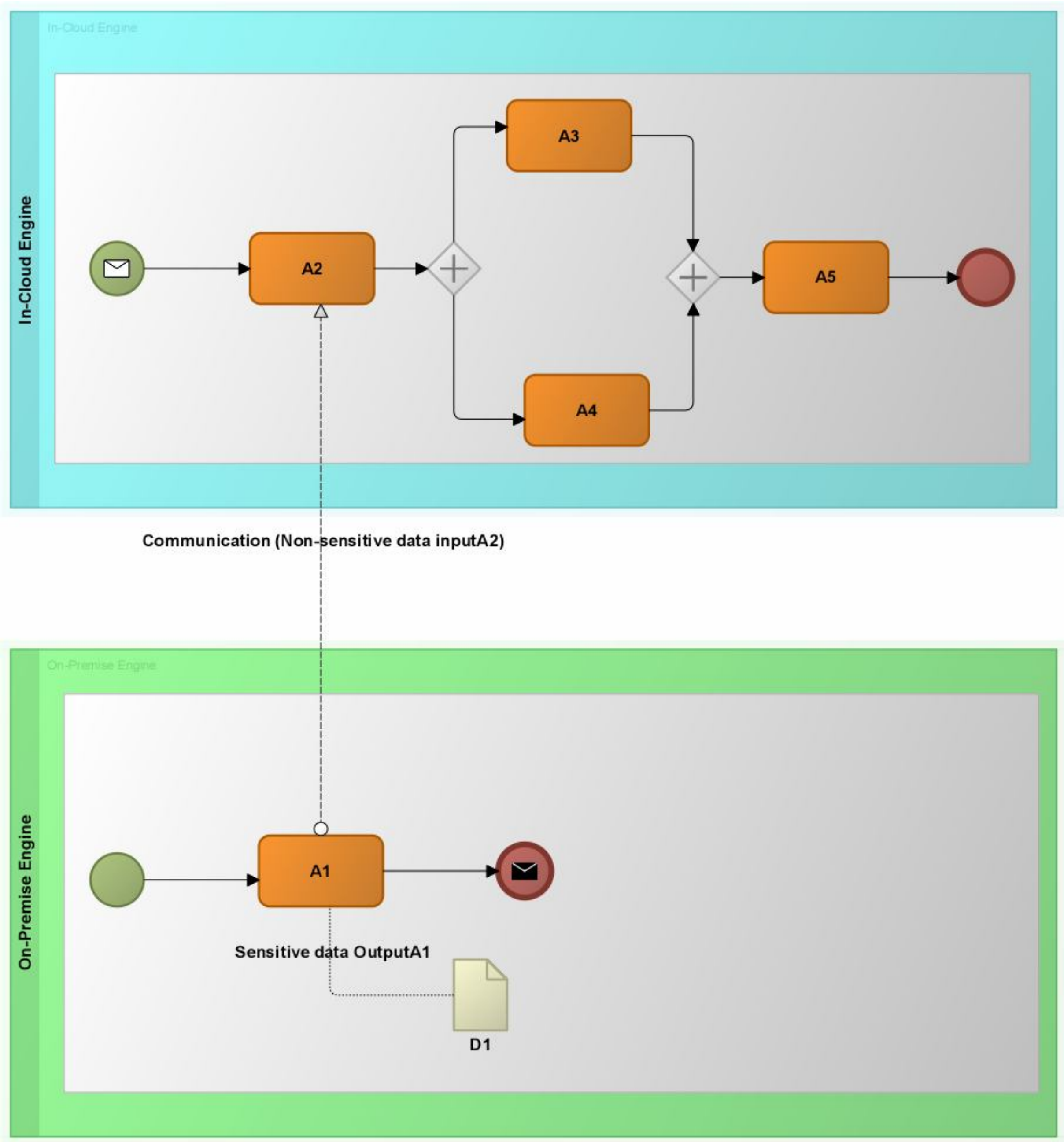

Figure 17 Manually identified one possible distribution option into collaborating in-cloud and on-premise engines

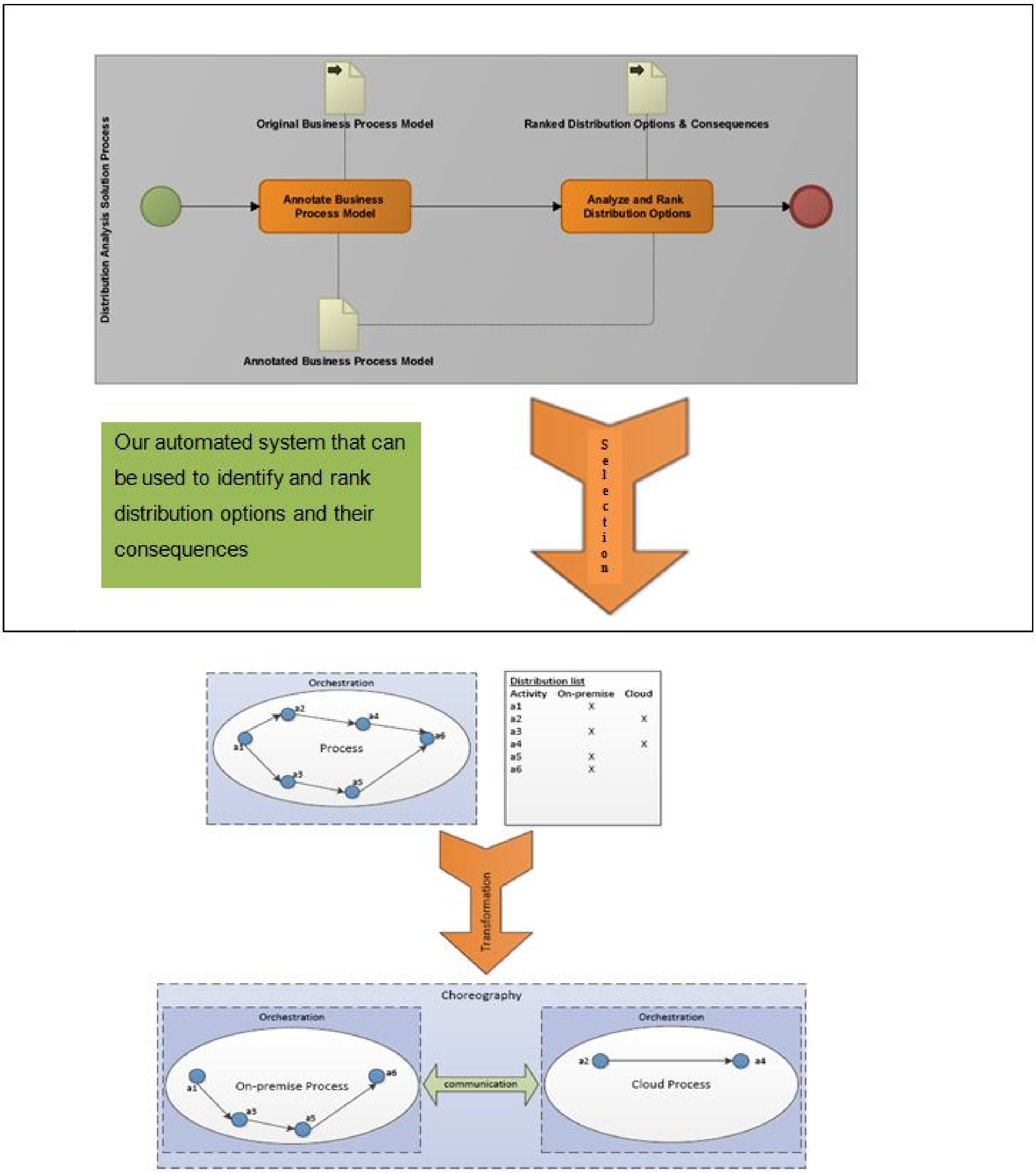

Figure 18 How our automated system complements the work of [17] by automatically identifying and ranking distribution options and their consequences before the actual decomposition

In order to automatically identify and rank business process distribution options and their consequences by using our automated system, cloud-related information and on-premise costs are necessary. Business processes should be annotated with this information by process designers at process design time. Therefore, in order to allow process designers annotate their business processes with this cloud-related information an annotation language is required. Algorithms for identifying and ranking distribution options and their consequences by using the annotated business processes as an input are also needed.

5.2 *Cloud Billing Models*

We have investigated cloud billing models to identify the necessary cloud-related information regarding cost, and how this information should relate to business processes. This information is used to design a visual annotation language for annotating business processes with cloud-related information.

We analyzed the cloud billing models of the most popular cloud service providers, such as Amazon Web Services Cloud [45], Google Cloud Platform [46], Microsoft Cloud Services [47] and Salesforce [13] in order to identify cost types that are applicable to cloud-based BPM. Based on this analysis we identified cloud resource attributes that can be used to estimate each of these cost types. Additionally, on-premise cost is considered in order to compare cost benefits.

The following cost types and the resource attributes that can be used to calculate each of these cost types are identified. Most of these resource attributes come from Amazon Web Services Cloud.

1. Computing capacity cost

A portion of the costs associated with cloud-based BPM comes from computing capacity. The cloud computing resources that are used for estimating computing capacity cost and their possible values are the following:

- Computing Capacity Cost per Hour

This cost refers to the amount of money that the organization has to pay per hour for executing business process activities in the in-cloud engine. This value is publicly available from cloud service providers.

- Execution Time

This refers to the time it takes to complete the execution of a business process activity in the in-cloud engine. This value can only be determined dynamically at the time of execution of the actual business process. However, process designers can assign this value once the business process is executed.

- Execution Cycle

The execution cycle refers to the expected execution frequency of a single business process activity in the in-cloud engine per time unit.

- Machine Type

This refers to the required operating system in the cloud environment that the organization wishes to execute their business processes. The most common values are:

  - ✓ Windows
  - ✓ Linux

- Machine Instance Type

This refers to the processing power of the cloud machine instance in terms of speed, memory and number of processor cores. The most common values are:

- ✓ Small

  In Amazon, for example, small stands for 1 virtual core, 1.7 GB RAM size, 160 GB storage capacity and moderate I/O.
- ✓ Large

  In Amazon, for example, large stands for 2 virtual cores, 7.5 GB RAM size, 850 GB storage capacity and high I/O.
- ✓ Medium

  In Amazon, for example, medium stands for 1 virtual core, 3.7 GB RAM size, 410 GB storage capacity and moderate I/O.

- Purchase Type

Purchase type refers to the way process designers would like to pay for cloud machine instances. Two most common values are:

- ✓ On-Demand
- ✓ Reserved

- Number of Machine Instances

This refers to the number of cloud machine instances that process designers would like to purchase in order to execute their business process activities in the in-cloud engines.

- Upfront Cost

Some cloud service providers give organizations an option to pay a one-time upfront fee in order to reserve cloud machine instances. This reduces the on-demand rate that the organization has to pay.

2. Data storage cost

The cloud computing resources that are used for estimating data storage cost, and their possible values are the following:

- Storage Cost per Hour

This cost refers to the amount of money that the organization has to pay per hour for storing 1 GB of data items in the in-cloud database. This value is publicly available from cloud service providers.

- Storage Size

This refers to the total size of all data items that the organization wishes to store in the in-cloud database.

- Storage Time

This refers to the time the organization wishes to store its business process data in the in-cloud database.

- Database Type

The database type refers to the available database management systems in the cloud environment. The three most common values are:

  - ✓ Oracle
  - ✓ Microsoft SQL Server
  - ✓ MySQL

- Database Instance Type

This refers to the processing power of the cloud machine instance on which the database instance is running in terms of speed, memory and number of processor cores. Possible values are:

  - ✓ Small

    In Amazon, for example, small stands for 1 virtual core, 1.7 GB RAM size, 160 GB storage capacity and moderate I/O.
  - ✓ Large

    In Amazon, for example, large stands for 2 virtual cores, 7.5 GB RAM size, 850 GB storage capacity and high I/O.
  - ✓ Medium

    In Amazon, for example, medium stands for 1 virtual core, 3.7 GB RAM size, 410 GB storage capacity and moderate I/O.

- Storage Class

This gives organizations an option to store their data with and without redundancy in the in-cloud database. The Standard storage class allows organizations to store critical data with redundancy for recovery. The Reduced Redundancy storage class allows organizations to store non-critical data at lower levels of redundancy in order to reduce storage costs.

  - ✓ Standard
  - ✓ Reduced Redundancy

- Purchase Type

This refers to how organizations would like to buy cloud storage services. To possible options are:

  - ✓ On-Demand
  - ✓ Reserved

- Region

This refers to geographic locations where the organization would like to store their data. Common values are:

  - ✓ EU
  - ✓ USA

3. Data communication cost

The cloud computing resources that are used for estimating data communication cost, and their possible values are the following:

- Communication Cost per GB

This refers to the cost of transferring 1 GB of data between the collaborating in-cloud and on-premise engines.

- Traffic Distribution

This refers to the geographical location of the in-cloud and on-premise engines. Some cloud service providers charge differently based on the location differences of the cloud machine instances.

  - ✓ The same location
  - ✓ Different locations

- Request Type

Cloud service providers charge organizations differently based on request types. Possible values are:

  - ✓ GET
  - ✓ PUT
  - ✓ COPY
  - ✓ DELETE

- Data Transfer Type

Cloud service providers charge organizations differently based on the required transport protocol. Possible values are:

  - ✓ HTTP
  - ✓ HTTPS

- Data Transfer Size

This refers to the size of the business process data to be communicated between the collaborating in-cloud and on-premise engines.

- Number of Requests

This refers to the expected number of requests per unit time.

4. On-premise cost

We assume that organizations know the on-premise costs associated with their business processes in detail, and that this cost is available locally to our automated system. The total on-premise costs involved are calculated by adding the following cost components:

- Software Cost

  These software costs include operating systems, BPM engines, database management systems and software licenses of running an on-premise BPM.

- Hardware Cost

  This refers to the total cost of hardware equipments required for running an on-premise BPM.

- Maintenance Cost

  This refers to the hardware and software maintenance costs of running an on-premise BPM.

- Operating Cost

These costs include: hardware, software and network operating costs i.e. employee salaries of running an on-premise BPM. The human factors including employee salaries of adopting cloud-based BPM is not considered.

5.3 Privacy Risk

When adopting Cloud-based BPM, some strategic information of the organization may be exposed to unauthorized access. In order to allow process designers and business analysts configure the privacy level of their activities and data items, we define the following four mappings of privacy levels to preferred allocations:

- If process designers do not care about the privacy level associated with an element, its intended allocation should be either on-premise or in-cloud and should be annotated as default.
- If an element is private its intended allocation should be annotated as Necessarily-on-premise since privacy can only be guaranteed on-premise.
- If an element is partially private its intended allocation should be annotated as On-premise-preferred.
- If the element has no privacy restriction, and that it can be made public, its intended allocation should be annotated as In-cloud-preferred since resources are usually cheaper in-cloud.

We have selected BPMN 2.0 as a process modeling language. The decision to select BPMN 2.0 as a modeling language is threefold. First, BPMN 2.0 is one of the languages supported by the available tools at BiZZdesign. Second, BPMN 2.0 is a standard business process modeling language developed by the Object Management Group (OMG). Third, the process modeling language that we want to consider should support annotations in order to allow process designers annotate business processes with cloud-related information by using our annotation language, and annotation is a BPMN 2.0 process modeling element.

5.4 Annotation Language Metamodel

We define a visual annotation language that can be used to annotate BPMN 2.0 process elements with cloud-related information and on-premise costs. Process designers can use this annotation language to annotate their business processes with the information that can be used to identify and rank distribution options and their consequences by using our automated system. Activities, data items and data associations are the BPMN 2.0 process constructs that need to be annotated with the cloud-related information by using this annotation language. These process constructs are selected from the descriptive conformance sub-class of the BPMN 2.0 specification described in [28]. Activities require resources for executing them in the in-cloud engine. Data items require storage capabilities for storing them in the in-cloud database management systems. Data associations are considered since data is potentially moved from one location to another. These process constructs are mapped to our annotation language elements in order to guide process designers during the annotation step. The annotation language determines which process constructs should be annotated with what cloud-related information based on these mappings. Our annotation language and automated system are built based on these mappings. The abstract syntax metamodel of our annotation language is shown in Figure 19. The enumeration types represent the cloud-related information that characterizes the elements of our annotation language metamodel. For simplicity, the metamodel does not show how the activities, data items and data associations should be mapped to our annotation language elements. This mapping is separately shown in Table 3.

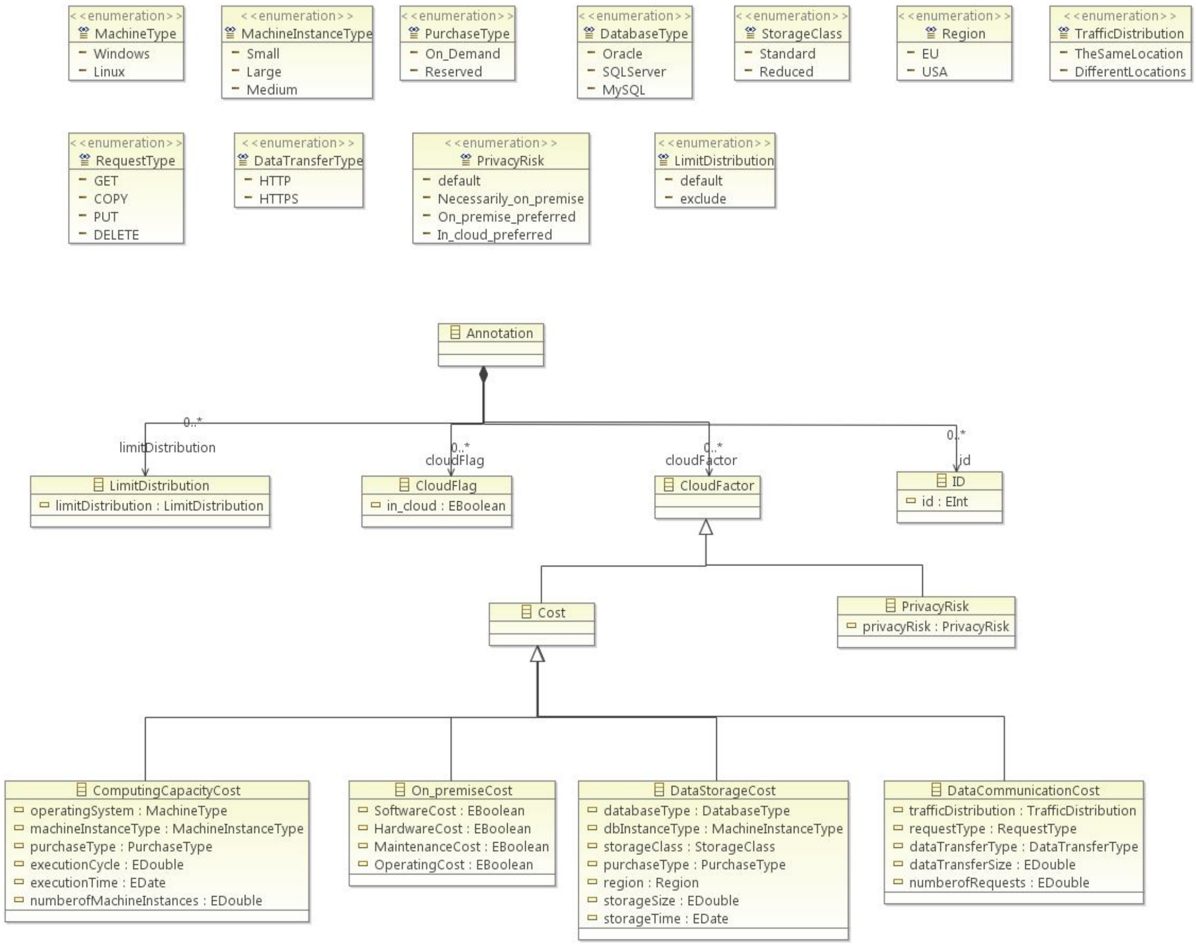

Figure 19 Metamodel of our annotation language

	Annotation Language Elements	BPMN 2.0 Process Constructs		
		Activities	Data Items	Data Associations
Main	ComputingCapacityCost	√	×	×
	DataStorageCost	×	√	×
	DataCommunicationCost	×	×	√
	On_premiseCost	√	√	×
	PrivacyRisk	√	√	×
Helpers	LimitDistribution	√	√	×
	CloudFlag	√	√	×
	ID	√	√	×

Table 3 Mappings of BPMN 2.0 process constructs to our annotation language elements

The main elements in our annotation language metamodel are explained below:

- ComputingCapacityCost

This language element is used to annotate activities with resource attributes that can be used to identify computing capacity cost of the annotated activity when executed in the in-cloud engine. These attributes include: operating system types, image machine instance types and cloud service purchasing types. These resources are described by using enumeration types in our annotation language metamodel.

- DataStorageCost

This language element is used to annotate data items with resource attributes that can be used to identify the data storage cost of storing a business process data in the in-cloud database management system. These resources include: database types, storage classes, and preferred region for storing data, database instance types and cloud service purchasing types. These resources are described by using enumeration types in our annotation language metamodel.

- DataCommunicationCost

This language element is used to annotate data associations with resource attributes that can be used to identify data communication costs between the collaborating on-premise and in-cloud engines after decomposition. These resources include: traffic distributions, request types and data transfer types. These resources are described by using an enumeration type in our annotation language metamodel.

- On_premiseCost

This language element is used to annotate business process activities and data items with the relevant on-premise costs for comparison. These costs include: software cost, hardware cost, maintenance cost and operating costs. These cost types are described by using an enumeration type in our annotation language metamodel.

- PrivacyRisk

This language element is used to annotate business process activities and data items with attributes that impose privacy restrictions. These attributes are: default, Necessarily_on_premise, On_premise_preferred and In_cloud_preferred. These attributes are described by using an enumeration type in our annotation language metamodel.

- ID

This is a hidden helper annotation language element with an integer attribute always assigned to activities and data items. The purpose of this element is to identify the business process elements that can be considered for cloud deployment and those that must be handled on-premise by using our algorithms.

- CloudFlag

This is a hidden helper annotation language element with a Boolean attribute always assigned to activities and data items. The purpose of this element is to identify business process elements that are assigned to the in-cloud engine or database, and those that are assigned to the on-premise engine or database in a particular distribution option by using our algorithms.

- LimitDistribution

This annotation language element is used to exclude business process elements from the annotated business process model in order to limit the number of elements in each of the distribution options that should be displayed. The possible values are: default and exclude. If an element is annotated as default which will be assigned automatically whenever an element is created, then the element should be shown in the distributions. Process elements that are explicitly annotated as exclude by process designers are not displayed in the distribution options. During the calculations of cost benefits and the analysis of privacy risk, an excluded process element has the same effect as if the element must be handled on-premise.

5.5 Model-based Algorithms

The main algorithms developed in this work are:

- Identify Elements

Purpose: this algorithm is used to identify activities and data items that can be considered for cloud deployment, and those that must be handled on-premise. This algorithm uses the integer attribute of the hidden ID annotation language element for identification.

Input: annotated business process model with any of our annotation language elements.

Output: the output of this algorithm is an annotated business process model with the activities and data items that can be considered for cloud deployment, and those that must be handled on-premise identified. The difference between the input and output models is that in the input model which activities and data items can be considered for cloud deployment and which must be handled on-premise are not checked. This is checked by the algorithm and in the output model, the ID attribute of the activities and data items that can be considered for cloud deployment is assigned integer values {1, 2, 3…}, and the ID attribute of those that must be handled on-premise is set to {0}.

- Identify Distribution Option

Purpose: this algorithm is used to identify possible distribution options. Each possible distribution option is sent as a model to the other algorithms for the identification of its consequences regarding cost benefits and privacy risk.

Input: an annotated business process model with the activities and data items that can be considered for cloud deployment and those that must be handled on-premise.

Output: set of distribution options.

- Control the number of distribution options

Purpose: this algorithm gives process designers an option to control the number and type of distribution options and consequences they would like to identify and rank. For example, process designers can identify only the distribution options with 1 element distributed in-cloud and the rest on-premise, or only the distribution options with 2 elements distributed in-cloud and the rest on-premise.

Input: annotated business process model.

Output: the number and type of distribution options that should be identified.

- Calculate Computing Capacity Cost

Purpose: this algorithm is used to calculate the computing capacity cost of a particular distribution option based on an annotated business process model.

Input: an annotated business process model representing a particular distribution option.

Output: total computing capacity cost of the distribution option given as input.

- Calculate Data Storage Cost

Purpose: this algorithm is used to calculate the data storage cost of a particular distribution option based on an annotated business process model.

Input: an annotated business process model representing a particular distribution option.

Output: total data storage cost of the distribution option given as an input.

- Calculate Data Communication Cost

Purpose: this algorithm is used to calculate the data communication cost of a particular possible distribution option.

Input: an annotated business process model representing a particular distribution option.

Output: the total data communication cost of the distribution option given as an input.

- Check Privacy Risk

Purpose: this algorithm is used to analyze the privacy risk consequence of a particular distribution option.

Input: an annotated business process model representing a particular distribution option.

Output: the privacy risk consequence of the distribution option given as an input.

- Rank Distribution Options

Purpose: this algorithm is used to rank distribution options based on cost benefits in order to facilitate choices.

Input: all possible distribution options and their consequences regarding cost and privacy risk

Output: ranking of the distribution options given as an input according to cost benefits. For simplicity, the ranking is made only based on cost benefits. However, the privacy risk consequences of each of the distribution options in the ranked list are also shown in order to protect the strategic information of the organization.

Chapter 6: Implementation of the Annotation Language Metamodel

The purpose of this chapter is to define the implementation of the annotation language metamodel discussed in Section 5.4 with BiZZdesign Architect Profiles as required by the BiZZdesign Architect modeling tool.

This chapter is structured as follows: Section 6.1 defines the implementation of the enumeration types in our annotation language metamodel; Section 6.2 defines the implementation of the annotation language elements for cost and privacy; Section 6.3 defines the implementation of the auxiliary annotation language elements that are used by our algorithms and Section 6.4 describes the implementation of annotation markers.

6.1 Enumeration Types

The Object Management Group (OMG) defines annotations as a standard extension mechanism to attach extra information to BPMN 2.0 modeling elements. However, there is a lack of methodological support during the implementation of these extensions [29]. These issues are left to be handled by the tool and language vendors. Accordingly, the BiZZdesign Architect modeling tool has a metamodel implementation mechanism based on profiling with well-defined syntax and semantics. Therefore, we define the implementation of our annotation language metamodel discussed in Section 5.4 with the BiZZdesign Architect Profile Definition Language as an extension to the BiZZdesign Architect modeling tool. This allows the abstract syntax metamodel to be automatically parsed into a visual concrete syntax by the BiZZdesign Architect modeling tool. This visual concrete syntax can also be interpreted by the compilers of the languages supported by this tool.

The following enumeration types are defined:

- MachineType

```
type MachineType = enum {
        Window, Linux
};
```

- MachineInstanceType

```
type MachineInstanceType = enum {
        Small, Large, Medium
};
```

- PurchaseType

```
type PurchaseType = enum {
```

```
        On_Demand, Reserved
};
```

- DatabaseType

```
type DatabaseType = enum {
        Oracle, SQLServer, MySQL
};
```

- StorageClass

```
type StorageClass = enum {
        Standard, Reduced
};
```

- Region

```
type Region = enum {
        EU, USA
};
```

- TrafficDistribution

```
type TrafficDistribution = enum {
        TheSameLocation, DifferentLocations
};
```

- RequestType

```
type RequestType = enum {
        GET, COPY, PUT, DELETE
};
```

- DataTransferType

```
type DataTransferType = enum {
        HTTP, HTTPS
};
```

- PrivacyRisk

```
type PrivacyRisk = enum {
        default, Necessarily_on_premise, On_premise_preferred, In_cloud_preferred
};
```

- LimitDistribution

```
type LimitDistribution = enum {
        default, exclude
};
```

6.2 Annotation Language Elements for Cost and Privacy

BiZZdesign Architect profiles are defined with the keyword **PROFILE** or **HIDDEN PROFILE** as a separate metaclass consisting of the set of properties that can be assigned to a certain process modeling element. These profiles are associated to the process modeling elements with the keywords **assignable to** and **always assigned to** whenever necessary without major changes in their definition. The keyword **assignable to** allows process designers to conditionally assign profiles to process modeling elements and the keyword **always assigned to** assigns profiles to process modeling elements by default. Hidden profiles are not graphically shown in the BiZZdesign Architect profiles section. BiZZdesign profiles allow single inheritance by using the keyword **extends**.

We define our mappings of the BPMN 2.0 activities, data items and data associations to the elements of our annotation language metamodel with the keywords **assignable to** and **always assigned to.**

- BPMN_CloudFactor

This profile implements the CloudFactor metaclass in our annotation language metamodel. This profile contains the generic cloud-related information regarding all cloud factors applicable to all other profiles. The other profiles defined afterwards extend this profile. We have not identified any generic information at the moment, however we considered this in our design to allow future extensions.

```
PROFILE BPMN_CLoudFactor {
        // generic cloud-relate information  regarding all factors
};
```

- BPMN_Cost

This profile implements the Cost metaclass in our annotation language metamodel. This profile contains the generic cloud-related information applicable to all profiles regarding cost. This profile extends the generic profile BPMN_CloudFactor.

```
PROFILE BPMN_Cost extends BPMN_CLoudFactor {
        // generic cloud-relate information  regarding cost factor
};
```

- BPMN_ComputingCapacityCost

This profile implements the ComputingCapacityCost metaclass in our annotation language metamodel. In the BiZZdesign Architect modeling tool, BPMN_Task stands for activities. The keyword **assignable to** defines that the BPMN_ComputingCapacityCost profile is mapped to BPMN 2.0 activities.

```
PROFILE BPMN_ComputingCapacityCost extends BPMN_Cost {
        assignable to BPMN_Task;

        MachineType operatingSystem;
        MachineInstanceType computingPower;
        PurchaseType purchaseType;
        real executionCycle;
        real executionTime;
        real numberofMachineInstances;
};
```

- BPMN_DataStorageCost

This profile corresponds to the DataStorageCost metaclass in our annotation language metamodel. In the BiZZdesign Architect modeling tool, BPMN_ItemAwareElement stands for data items. The keyword **assignable to** defines that the BPMN_DataStorageCost profile is mapped to BPMN 2.0 data items.

```
PROFILE BPMN_DataStorageCost extends BPMN_Cost {
        assignable to BPMN_ItemAwareElement;

        DatabaseType databaseType;
        DBInstanceType dbInstanceType;
        StorageClass storageClass;
        PurchaseType purchaseType;
        Region region;
        real storageSize;
        real storageTime;
};
```

- BPMN_DataCommunicationCost

This profile corresponds to the DataCommunicationCost metaclass in our annotation language metamodel. In the BiZZdesign Architect modeling tool, BPMN_DataAssociation stands for data associations. The keyword **assignable to** defines that the BPMN_DataCommunicationCost profile is mapped to BPMN 2.0 data associations.

```
PROFILE BPMN_DataCommunicationCost extends BPMN_Cost {
        assignable to BPMN_DataAssociation;

        TrafficDistribution trafficDistribution;
```

```
        RequestType requestType;
        DataTransferType dataTransferType;
        real dataTransferSize;
        real numberofRequests;
};
```

- BPMN_On_premiseCost

This profile corresponds to the On_premiseCost metaclass in our annotation language metamodel. The keyword **assignable to** defines that the BPMN_On_premiseCost profile is mapped to BPMN 2.0 data items and activities.

```
PROFILE BPMN_On_premiseCost extends BPMN_Cost {
        assignable to BPMN_ItemAwareElement, BPMN_Task;

boolean SoftwareCost; // including licensing costs
boolean HardwareCost;
boolean StorageCost;
boolean CommunicationCost;
boolean MaintenanceCost;
boolean OperatingCost;// including staff costs
};
```

- BPMN_PrivacyRisk

This profile corresponds to the PrivacyRisk metaclass in our annotation language metamodel. The keyword **assignable to** defines that the BPMN_PrivacyRisk profile is mapped to BPMN 2.0 data items and activities.

```
PROFILE BPMN_PrivacyRisk extends BPMN_CLoudFactor{
        assignable to BPMN_ItemAwareElement, BPMN_Task;

        PrivacyRisk PrivacyRisk;
};
```

6.3 Auxiliary Annotation Language Elements

The purpose of these auxiliary annotation language elements is discussed in Section 5.4.

- BPMN_ID

This profile corresponds to the ID metaclass in our annotation language metamodel. The keyword **always assigned to** defines that the BPMN_ID profile is mapped to BPMN 2.0 data items and activities.

```
HIDDEN PROFILE BPMN_ID{
        always assigned to BPMN_ItemAwareElement, BPMN_Task;
```

```
        integer ID;
};
```

- BPMN_CloudFlag

This profile corresponds to the CloudFlag metaclass in our annotation language metamodel. The keyword **always assigned to** defines that the BPMN_CloudFlag profile is mapped to BPMN 2.0 data items and activities.

```
HIDDEN PROFILE BPMN_CloudFlag{
        always assigned to BPMN_ItemAwareElement, BPMN_Task;
        boolean in_cloud;
};
```

- BPMN_LimitDistribution

This profile corresponds to the LimitDistribution metaclass in our annotation language metamodel. The keyword **always assigned to** defines that the BPMN_LimitDistribution profile is mapped to BPMN 2.0 data items and activities.

```
PROFILE BPMN_LimitDistribution {
        always assigned to BPMN_ItemAwareElement, BPMN_Task;
        LimitDistribution limitDistributions;
}
```

6.4 Annotation Markers

In order to remind process designers that a BPMN 2.0 activity, data item or data association has been annotated with the cloud-related information by using our annotation language, we define graphical icon and associate them to the profiles so that they are shown by the BiZZdesign Architect modeling tool. Whenever an element is annotated with BPMN_ComputingCapacityCost, BPMN_DataStorageCost, BPMN_DataCommunicationCost or BPMN_On_premiseCost, a "moneybag" graphical icon is used. Whenever an element is annotated with BPMN_PrivacyRisk, a "risk-icon" graphical icon is used. Whenever an element is annotated with BPMN_LimitDistribution, a "cross-icon" graphical icon is shown.

- The moneybag graphical icon

This code snippet displays a moneybag graphical icon whenever an activity, data item or data association is annotated with BPMN_ComputingCapacityCost, BPMN_DataCommunicationCost or BPMN_On_premiseCost.

```
if ( object.hasProfile("BPMN_ComputingCapacityCost") ) {
	output "overlay", svg:addStyle(symbolPath("moneybag", List(), List()), "fill:#00ff00");
}
if ( object.hasProfile("BPMN_DataCommunicationCost") ) {
	output "overlay", svg:addStyle(symbolPath("moneybag", List(), List()), "fill:#00ff00");
}
if ( object.hasProfile("BPMN_On_premiseCost") ) {
	output "overlay", svg:addStyle(symbolPath("moneybag", List(), List()), "fill:#00ff00");
}
```

- The risk-icon graphical icon

This code snippet displays a risk-icon graphical icon whenever an activity or data item is annotated with BPMN_PrivacyRisk.

```
if ( object.hasProfile("BPMN_PrivacyRisk") ) {
	output "overlay", svg:addStyle(symbolPath("risk-icon", List(), List()), "fill:#ff0000");
}
```

- The cross-icon graphical icon

This code snippet displays a cross-icon graphical icon whenever an activity or data item is annotated with BPMN_LimitDistribution and if the annotation value is exclude.

```
if ( object.hasProfile("BPMN_LimitDistribution")){
	if(object.attrValue("limitDistributions").toString() == "exclude"){
	output "overlay", svg:addStyle(symbolPath("cross-icon", List(), List()), "fill:#ff0000");
	}
}
```

Chapter 7: Model-based Algorithms

The purpose of this chapter is to discuss the model-based algorithms that we have defined to identify and rank distribution options and their consequences regarding cost benefit and privacy risks based on annotated business process models.

This chapter is structured as follows: Section 7.1 gives an overview of the conventions we have used; Section 7.2 discusses the helper algorithms we have defined; Section 7.3 discusses the algorithm we have defined to identify distribution options; Section 7.4 discusses the cost algorithms we have defined; Section 7.5 discusses the privacy risk algorithm we have defined; Section 7.6 describes the analysis and ranking algorithms we have defined and Section 7.7 discusses the algorithm we have defined to display the result of our analysis in a tabular form.

7.1 Conventions

The algorithms we have developed for identifying and ranking distribution options into collaborating in-cloud and on-premise engines, and their consequences are described by using pseudocode. The following pseudocode conventions apply throughout the description of these algorithms:

1. Some basic built-in functions of the BiZZdesign Architect Advanced Scripting Language which we have used for identification and ranking of distribution options are embedded in the pseudocode. The purpose of these built-in functions will be discussed when they appear for the first time.
2. The terms activity, data and communication stand for the activities, data items and data associations of the annotated input model.
3. Variables, structures, sets and lists have the same interpretation as in any imperative programming language, such as Java or C.
4. A block structure is indicated by using indentation and end statements. For example, forall loops are terminated with the keyword 'end forall', while loops are terminated with 'end while', and if-then-else statements are terminated with 'end if'.
5. The symbol "▹" indicates that the rest of the line is a comment.
6. The symbol "←" represents a value assignment to a variable.
7. All parameters are passed by reference to other algorithms.
8. The Boolean operators and and or are represented by the symbols & and || respectively, and they have the same interpretation as their Java, C++ and C counterparts.
9. Comparison operator is represented by the symbol =.
10. String values are represented by using double quotes. For example, "Tesfahun".

11. The basic mathematical operators, +, -, * and / have the same symbol and interpretation as their Java, C++ or C counterparts.
12. The original business process model is assumed to be on-premise. The cost of on-premise deployment is used for comparison.
13. We consider separate upfront costs for computing capacity, data storage and communication services.
14. We assume cloud service prices remain constant for the duration of time considered.

7.2 Helpers

Helpers are used to support the functioning of the main algorithms we have defined. These helper algorithms are discussed as follows:

7.2.1 Elements Identifier

This algorithm is used to identify the activities and data items that can go to the in-cloud engine and database. The algorithm accepts an annotated model as an input. This is the first algorithm that should be executed before any other algorithm in our automated system. Some activities and data items in the annotated input model must stay on-premise for two reasons: those that represent strategic information of the organization, and those that are excluded for the purpose of limiting the number of distribution options. The rest of the activities and data items in the input model can be considered for deployment in the in-cloud engine, and database. This algorithm uses a hidden profile with an attribute called ID in order to identify activities and data items that can go to the cloud, and those that should be handled on-premise. The ID attribute of the activities and data items that can go to the cloud is assigned integer values {1, 2, 3, …}, whereas the ID attribute of the activities and data items that must stay on-premise are assigned the value of {0}. The function setAttrValue("ID") is used to assign values to the attribute ID of activities and data items. The function hasProfile("BPMN_PrivacyRisk") is used to check if an activity or data item is annotated with the BPMN_PrivacyRisk profile. If the element is annotated, the function attrValue("PrivacyRisk") is used to extract the actual value of the enumeration type attribute PrivacyRisk. If the privacy risk annotation value of the element is "Necessarily_on_premise", the element must stay on-premise and its ID is set to {0}. Similarly, if the annotation value of the enumeration type attribute limitDistributions of the element is "exclude", its ID is set to {0} since excluded elements are assumed to be on-premise according to our approach. The algorithm returns the annotated model with all the activities and data items that can go to the cloud, and those that must stay on-premise identified. This algorithm is described by using a pseudocode as shown below:

Algorithm 7.2.1 **Elements Identifier**

Input: Annotated Business Process Model inputModel

Output: Business Process Model with all activities and data items that can go to the cloud, and those that must stay on-premise identified

```
ELEMENTS-IDENTIFIER (inputModel)
ID ← 1
forall a in inputModel do
    assign ← true ▹ initially assign is set to true which means the element can go to cloud
▹ checking if an activity or data item is annotated with privacy risk profile, and if it must stay on-premise
    if a is activity || a is data & a.hasProfile("BPMN_PrivacyRisk") then
        if a.attrValue("PrivacyRisk") = "Necessarily_on_premise" then
            assign ← false ▹ this element must be handled on-premise
        end if
    end if
▹ checking if an activity or data item is excluded in order to limit distribution options
    if a is activity || a is data then
        if a.attrValue("limitDistributions") = "exclude" then
            assign ← false ▹ this element is assumed to be handled on-premise
        end if
    end if
▹ Assigning ID of {1,2,3,..} to the activities and data items that can go to cloud
    if a is activity || a is data & assign = true then
        a.setAttrValue("ID") ← ID
        ID ← ID + 1
▹ Assigning ID of {0} to the activities and data items that must be handled on-premise
    else
        if a is activity || a is data then
            a.setAttrValue("ID") ← 0
        end if
    end if
end forall
end algorithm
```

7.2.2 Limit Counter

This algorithm is used to count the number of activities and data items that are excluded from the input model in order to limit the number of distribution options. The algorithm accepts an annotated business process model. Activities and data items that are annotated with limitDistributions profile where the value is “exclude” are counted. This algorithm returns the total number of such activities and data items in the input model. The pseudocode of this algorithm is shown below:

Algorithm 7.2.2 **Limit Counter**

Input: Annotated Business Process Model inputModel

Output: The total number of activities and data items excluded in order to limit the number of distribution options

```
LIMIT-COUNTER (inputModel)
    limit ← 0
    forall a in inputModel do
        if a is activity || a is data then
            if a.attrValue(“limitDistributions”) = “exclude” then
                limit ← limit + 1
            end if
        end if
    end forall
    return limit
end algorithm
```

7.2.3 Privacy Risk Counter

This algorithm is used to count the number of private activities and data items in the annotated input model. The algorithm walks through the model in order to identify private and excluded activities and data items. The activities and data items that are excluded in order to limit the number of distributions are not counted. This avoids double counting, since they are already counted in algorithm 7.2.2. The algorithm checks if an activity or data item is annotated with the BPMN_PrivaryRisk profile. If the activity or data item is annotated, the algorithm extracts the annotation value. If the value of the annotation is “Necessarily_on_premise”, the element is counted as private. The algorithm returns the total number of private activities and data items. The complete pseudocode of this algorithm is shown below:

Algorithm 7.2.3 **Privacy Risk Counter**

Input: Annotated Business Process Model inputModel

Output: The total number of private activities and data items

```
PRIVACY-RISK-COUNTER (inputModel)
    private ← 0
    counted ← false
    forall a in inputModel do
 ▷  checking if the activity or data item is counted already in the limit counter algorithm
        if a is activity || a is data then
            if a.attrValue("limitDistributions") = "exclude" then
                counted ← true
            end if
        end if
 ▷  counting private activities and data items that has not been counted
        if a is activity || a is data  & a.hasProfile("BPMN_PrivacyRisk") & counted = false then
            if a.attrValue("PrivacyRisk") = "Necessarily_on_premise" then
                private ← private + 1
            end if
        end if
    end forall
    return private
end algorithm
```

7.2.4 Elements Counter

This algorithm counts the number of activities and data items that can go to the in-cloud engine and database. The built-in function inputModel.allChildren("BPMN_Task").size() returns the total number of activities in the input model. The total number of data items in the annotated input model are counted by using the function inputModel.allChildren("BPMN_Data").size(). This algorithm uses algorithms 7.2.2 and 7.2.3 in order to count the number of private and excluded activities and data items respectively. The algorithm calculates the total number of activities and data items that can go to the cloud by subtracting private and excluded elements from the total activities and data items in the annotated input model. This algorithm is described by using a pseudocode as follows:

Algorithm 7.2.4 **Elements Counter**

Input: Annotated Business Process Model inputModel

Output: The number of activities and data items that can go to the in-cloud engine and database

```
ELEMENTS-COUNTER (inputModel)
    private ← privacyCounter(inputModel)
    limit ← lmitCounter(inputModel)
```

```
        activities ← inputModel.allChildren("BPMN_Task").size()
        dataItems ← inputModel.allChildren("BPMN_Data").size()
        numberofElements ← (activities + dataItems) – (private + limit)
        return numberofElements
    end algorithm
```

7.2.5 Controller

This algorithm gives process designers and business analysts an option to control the number and type of distributions they would like to identify. This algorithm aims at improving the speed of execution and limits the number of distribution options that should be identified. The algorithm described below with a pseudocode shows the worst case scenario in which process designers would like to identify all possible distribution options, and their consequences. The variable in_cloudElements introduced in line 2 determines the type and number of distribution options that should be identified. For example, if its value is 1, only the distribution options with 1 element distributed in-cloud and all the other elements on-premise, and their consequences should be identified. If the value of the variable is 2, only the distribution options with 2 elements distributed in in-cloud and the rest of the elements on-premise, and their consequences should be identified. The value of in_cloudElements can be controlled by varying the value of the variable totalNumberofElements. This way analysts can control the number and type of distribution options that should be identified, and hence the number of times that the algorithm should be executed.

Algorithm 7.3 **control distributions**

Input: Annotated Business Process Model inputModel, integer byref numberofDistributions, structure byref distributionCost, structure byref distributionPrivacyRisk, structure byref costList

Output: The number of distribution options that should be identified

```
CONTROL-DISTRIBUTIONS (inputModel, numberofDistributions, distributionCost,
                                   distributionPrivacyRisk, costList)
1  totalNumberofElements ← elementsCounter(inputModel)
2  in_cloudElements ← 1
3  while in_cloudElements <= totalNumberofElements do
4      identifyDistributionOption(inputModel, in_cloudElements, numberofDistributions, distributionCost,
                                   distributionPrivacyRisk, costList)
5      in_cloudElements ← in_cloudElements +1
6  end while
7  return
8  end algorithm
```

7.3 *Distribution Identification Algorithm*

We have defined two closely related main algorithms during the identification of distribution options. The first algorithm i.e. IDENTIFY-DISTRIBUTION-OPTION is visible to the other algorithms in our automated system, and it initializes the second recursive combinatorial algorithm. This algorithm returns the total number of distribution options. The algorithm accepts annotated business process model inputModel , the number of elements that should go to the cloud in each distribution option in_cloudElements, the total number of distribution options identified numberofDistributions, cost saving consequences of each identified distribution option distributionCost, the privacy risk consequence of each identified distribution option distributionPrivacyRisk and the cost list of each distribution option for ranking distributions costList.

Algorithm 7.2.5.1	**Identify Distribution Options**

Input: Annotated Business Process Model inputModel, integer byref in_cloudElements, integer byref numberofDistributions, structure byref distributionCost, structure byref distributionPrivacyRisk, structure byref costList

Output: all possible distribution options

IDENTIFY-DISTRIBUTION-OPTION (inputModel, byref in_cloudElements, byref numberofDistributions, byref distributionCost, byref distributionPrivacyRisk, byref costList)

totalNumberofElements ← elementsCounter(inputModel)
totalOptions ← List()
option ← List()
recursiveWalkthrough (inputModel, totalNumberofElements, in_cloudElements, option, totalOptions, numberofDistributions, distributionCost, distributionPrivacyRisk, costList)
return totalOptions
end algorithm

The second recursive combinatorial main algorithm described below recursively picks all possible distribution options, and sends them to algorithm 7.6.1 for cost saving and privacy risk analysis.

Algorithm 7.2.5.2	**Recursive Walkthrough**

Input: Annotated Business Process Model inputModel, integer byref totalNumberofElements, integer byref in_cloudElements, integer byref numberofDistributions, structure byref distributionCost, structure byref distributionPrivacyRisk, structure byref costList

Output: A particular distribution option

```
RECURSIVE-WALKTHROUGH (InputModel, byref totalNumberofElements, byref in_cloudElements, byref
option, byref totalOptions, byref numberofDistributions, byref distributionCost, byref distributionPrivacyRisk,
byref costList)
if option.size() = in_cloudElements  & !totalOptions.contains(option) then
totalOptions.add(option)
    forall a in inputModel do
            if a is activity || a is data then
                    if option.contains(a.attrValue("ID")) then
                            a.setAttrValue("in_cloud", true)
                    else
                            a.setAttrValue("in_cloud", false)
                    end if
            end if
    analyzeDistribution(inputModel, numberofDistributions, distributionCost, distributionPrivacyRisk,
            costList)
    end forall
else if option.size() > in_cloudElements then
     return
else
    i ← totalNumberofElements
    While i >= 1 do
            distribution_option ← List()
            if !option.empty() then
            distribution_option ← distribution_option + option
            end if
            distribution_option.add(i)
            total ← i - 1
            recursiveWalkthrough (inputModel, total, in_cloudElements, distribution_option, totalOptions,
                    numberofDistributions, distributionCost, distributionPrivacyRisk, costList)
            i ← i - 1
            end while
    end if
    return true
end algorithm
```

7.4 Cost Algorithms

Cost algorithms are used to calculate the cost consequences associated with each of the identified distribution options. These algorithms are discussed below.

7.4.1 Computing Capacity Cost

This algorithm is used to estimate the total computing capacity cost associated with each identified distribution option. Since activities are mapped to computing capacity cost, this algorithm considers only activities by recursively checking for activities in the annotated input model. If an activity is found, the algorithm checks if the activity is annotated with BPMN_ComputingCapacityCost profile, and if it is in the in-cloud engine. If this situation is true the algorithm extracts the annotation values, and retrieves corresponding prices from our local in-cloud computing capacity price list. If the activity is not in the in-cloud engine, it automatically means the activity is in the on-premise engine. In this situation, the algorithm checks if the activity is annotated with BPMN_On_premiseCost profile. If this is true, the annotation values are extracted, and the corresponding on-premise cost is retrieved from our local on-premise deployment price list. The total computing capacity cost is calculated based on both values, and finally the total value is returned. This algorithm is described below by using a pseudocode.

Algorithm 7.4.1	**Calculate Computing Capacity Cost**

Input: Annotated *Business Process Model* inputModel

Output: *Total computing capacity cost of a particular distribution option*

```
CALCULATE-COMPUTING-CAPACITY-COST (inputModel)
computingCapacityCost ← 0
upfrontCost ← 0
on_premiseCost ← 0
forall a in inputModel do
    if a is activity then
        if a.hasProfile(“BPMN_ComputingCapacityCost”) & a.attrValue(“in_cloud”) = true then
            queryString ← a.attrValue("operatingSystem").toString() +
            a.attrValue("computingPower").toString() +
            a.attrValue("purchaseType").toString()
            executionCycle ← a.attrValue("executionCycle").toString().toNumber()
            executionTime ← a.attrValue("executionTime").toString().toNumber()
            numberofMachineInstances ← a.attrValue("numberofMachineInstances")
    ▹ using a function called query to retrieve real cost from a database (structure in this case)
            realCloudCost ← query("ComputingCapacityCostStore", queryString)
```

```
        ▷ realCloudCost[1][1] returns hourlyCost and realCloudCost[2][1] returns upfrontCost
                upfrontCost ← realCloudCost[2][1].toNumber()
                computingCapacityCost ← computingCapacityCost +
                    (realCloudCost[1][1].toNumber() * executionCycle * executionTime *
                    numberofMachineInstances)
            end if
            if a.hasProfile("BPMN_On_premiseCost") & a.attrValue("in_cloud" = false) then
                if a.attrValue("SoftwareCost") = true then
                    queryString ← a.toString() + "SoftwareCost"
                    result ← query("On_premiseCostStore", queryString)
                    on_premiseCost ← on_premiseCost + result[1][1].toString().toNumber()
                end if
                if a.attrValue("HardwareCost") = true then
                    queryString ← a.toString() + "HardwareCost"
                    result ← query("On_premiseCostStore", queryString)
                    on_premiseCost ← on_premiseCost + result[2][1].toString().toNumber()
                end if
                if a.attrValue("MaintenanceCost") = true then
                    queryString ← a.toString() + "MaintenanceCost"
                    result ← query("On_premiseCostStore", queryString)
                    on_premiseCost ← on_premiseCost + result[5][1].toString().toNumber()
                end if
                if a.attrValue("OperatingCost") = true then
                    queryString ← a.toString() + "OperatingCost"
                    result ← query("On_premiseCostStore", queryString)
                    on_premiseCost ← on_premiseCost + result[6][1].toString().toNumber()
                end if
            end if
        end if
end forall
▷ adding upfront Cost
computingCapacityCost ← computingCapacityCost + upfrontCost + on_premiseCost
▷ returning result
return computingCapacityCost
end algorithm
```

7.4.2 Data Storage Cost

This algorithm estimates the total data storage cost associated with each identified distribution option. Since data items are mapped to data storage cost, the algorithm considers only data items by recursively checking for data items in the annotated input model. If a data item is found, the algorithm checks if the data item is annotated with BPMN_DataStorageCost profile, and if it is in the in-cloud database. If this situation is true, the algorithm extracts annotation values and retrieves corresponding prices from our local in-cloud data storage price list. The algorithm calculates and returns the total data storage cost. If the data item is not in the in-cloud database, it automatically means the data item is on-premise. In this situation, the algorithm checks if the data item is annotated with BPMN_On_premiseCost profile. If this is true, the annotation values are extracted, and the corresponding on-premise cost is retrieved from our local on-premise price list. The total data storage cost is calculated based on both values, and finally the total value is returned. This algorithm is described below by using a pseudocode.

Algorithm 7.5.1 **Calculate Data Storage Cost**

Input: Annotated *Business Process Model* inputModel

Output: *Total data storage cost of a particular distribution*

```
CALCULATE-STORAGE-COST (inputModel)
dataStorageCost ← 0
upfrontCost ← 0
on_premiseCost ← 0
forall a in inputModel do
    if a is data then
        if a.hasProfile("BPMN_DataStorageCost") & a.attrValue("in_cloud") = true then
            queryString ← a.attrValue("databaseType") + a.attrValue("dbInstanceType")
                + a.attrValue("storageClass") + a.attrValue("purchaseType") +
                a.attrValue("region")
            storageSize ← a.attrValue("storageSize")
            storageTime ← a.attrValue("storageTime")
            result ← query("StorageCostStore", queryString)
            hourlyCost ← result[1][1]
            upfrontCost ← result[2][1]
            dataStorageCost ← hourlyCost * storageSize * storageTime + dataStorageCost
        end if
        if a.hasProfile("BPMN_On_premiseCost") & a.attrValue("in_cloud") == false then
            if a.attrValue("SoftwareCost") == true then
                queryString ← a.toString() + "SoftwareCost"
                result ← query("On_premiseCostStore", queryString)
                on_premiseCost ← on_premiseCost +  result[1][1].toString().toNumber()
            end if
            if a.attrValue("HardwareCost") == true then
```

```
                queryString ← a.toString() + "HardwareCost"
                result ← query("On_premiseCostStore", queryString)
                on_premiseCost ← on_premiseCost +  result[2][1].toString().toNumber()
            end if
            if a.attrValue("MaintenanceCost") = true then
                queryString ← a.toString() + "MaintenanceCost"
                result ← query("On_premiseCostStore", queryString)
                on_premiseCost ← on_premiseCost +  result[5][1].toString().toNumber()
            end if
            if a.attrValue("OperatingCost") = true then
                queryString ← a.toString() + "OperatingCost"
                result ← query("On_premiseCostStore", queryString)
                on_premiseCost ← on_premiseCost +  result[6][1].toString().toNumber()
            end if
        end if
    end if
end forall
dataStorageCost ←  dataStorageCost + upfrontCost + on_premiseCost
return dataStorageCost
end algorithm
```

7.4.3 Data Communication Cost

This algorithm calculates the total data communication cost associated with each identified distribution option. Since data associations are mapped to data communication cost, this algorithm considers only data associations by recursively checking for data associations in the annotated input model. The algorithm identifies data associations by first checking activities or data items in the in-cloud engine. If in-cloud activity or data item is found, the algorithm checks if there are incoming or outgoing data associations with an on-premise element by using the built in function relations(). If a data association is found, then the activity or data item in the other end is checked. If the element is on-premise, the algorithm checks if the data association is annotated with BPMN_DataCommunicationCost profile. If this situation is true, the annotation values are extracted and the corresponding data communication prices are collected from our local data communication price list. Then the total communication cost is returned.

Algorithm 7.6.1	**Calculate Data Communication Cost**

Input: *Business Process Model* inputModel

Output: *Total communication cost for a particular distribution*

```
CALCULATE-COMMUNICATION-COST (inputModel)
dataCommunicationCost ← 0
realCLoudCost ← 0
upfrontCost ← 0
dataTransferSize ← 0
forall a in inputModel do
    if a is data || a is activity & a.attrValue("in_cloud") == true then
        communicatingElements ← Set()
        forall r in a.relations() do
            if r is communication then
                communicatingElements.add() ← a.relatedTo(r)
                forall items in communicatingElements do
                    if items.attrValue("in_cloud") == false then
                        if r.hasProfile("BPMN_DataCommunicationCost") then
                            queryString ← r.attrValue("trafficDistribution").toString()+
                                r.attrValue("requestType").toString() +
                                r.attrValue("dataTransferType").toString()
                            realCLoudCost ← query("CommunicationCostStore",
                                queryString)
                            upfrontCost ← realCLoudCost[2][1].toNumber()
                            dataTransferSize ← r.attrValue("dataTransferSize")
                            numberofRequests ← r.attrValue("numberofRequests")
dataCommunicationCost ← dataCommunicationCost + (realCLoudCost[1][1].toNumber() *
                dataTransferSize * numberofRequests)
                            ▹ clearing container for the next data or activity
                            communicatingElements.remove() ← items
                        end if
                    end if
                end forall
            end if
        end forall
    end if
end forall
dataCommunicationCost ← dataCommunicationCost + upfrontCost
return dataCommunicationCost
end algorithm
```

7.4.4 On-premise Costs

This algorithm is used to calculate the original total cost of running a traditional on-premise BPM. This cost is used for comparison. Since activities and data items are mapped to on-premise cost, this algorithm considers only activities and data items by recursively walking through the annotated input model. When an activity or a data item is found, the algorithm checks if the element is annotated with the BPMN_On_premiseCost profile. If this situation is true, the annotation values are extracted and the corresponding on-premise prices are collected from our local on-premise price list. Finally, the total on-premise cost is returned.

Algorithm 7.7.1 **Calculate On-premise Cost**

```
CALCULATE-ON-PREMISE-COST (inputModel)
on_premiseCost ← 0
forall a in inputModel do
    if a is activity || a is data then
        if a.hasProfile("BPMN_On_premiseCost") then
            if a.attrValue("SoftwareCost") = true then
                queryString ← a.toString() + "SoftwareCost"
                result ← query("On_premiseCostStore", queryString)
                on_premiseCost ← on_premiseCost + result[1][1].toString().toNumber()
            end if
            if a.attrValue("HardwareCost") = true then
                queryString ← a.toString() + "HardwareCost"
                result ← query("On_premiseCostStore", queryString)
                on_premiseCost ← on_premiseCost + result[2][1].toString().toNumber()
            end if
            if a.attrValue("MaintenanceCost") = true then
                queryString ← a.toString() + "MaintenanceCost"
                result ← query("On_premiseCostStore", queryString)
                on_premiseCost ← on_premiseCost + result[5][1].toString().toNumber()
            end if
            if a.attrValue("OperatingCost") = true then
                queryString ← a.toString() + "OperatingCost"
                result ← query("On_premiseCostStore", queryString)
                on_premiseCost ← on_premiseCost + result[6][1].toString().toNumber()
            end if
        end if
    end if
end forall
return on_premiseCost
end algorithm
```

7.4.5 Cost Benefit

This algorithm calculates the final cost benefit that can be saved from each of the identified distribution options by calling the algorithms we have defined earlier. The algorithm returns the total cost saving for each distribution option.

Algorithm 7.8.1	**Calculate Cost Benefit**

Input: *Business Process Model* inputModel

Output: *Total cost saving*

```
CALCULATE-COST-BENEFIT (inputModel)
        computingCapacityCost ← calculateComputingCapacityCost(inputModel)
        dataStorageCost ← calculateDataStorageCost(inputModel)
        dataCommunicationCost ← calculateDataCommunicationCost(inputModel)
        on_premiseCost ← calculateon_premiseCost(inputModel)

        costBenefit ← on_premiseCost - (computingCapacityCost + dataStorageCost +
                        dataCommunicationCost)

        return costBenefit
        end algorithm
```

7.5 Privacy Risk Algorithm

This algorithm checks the privacy risk associated with each of the identified distribution options. Since activities and data items are mapped to privacy risk, this algorithm considers only activities and data items by recursively walking through each of the distribution options. When an activity or data item is found, the algorithm checks if the element is annotated with BPMN_PrivacyRisk profile. In this situation, the algorithm checks if the element is in-cloud, and the annotation value is extracted. The annotation value of this in-cloud element can never be Necessarily_on_premise, because private activities and data items must be handled on-premise. This situation is handled by algorithm 7.2.1. If the annotation value of this in-cloud element is default, there is no any privacy risk. If the annotation value is On_premise_preferred, this has a privacy risk to a certain degree, since on-premise preferred element is moved to the cloud. If the annotation value of this in-cloud element is In_cloud_preferred, this is interesting since the element is in-cloud already. If the activity or data item is not in-cloud, this automatically means the element is on-premise, and the annotation value of this element is extracted. If the annotation value of this on-premise element is Necessarily_on_premise, this is interesting since the element is on-premise already. If the annotation value is default, there is no privacy risk. If the annotation value is On_premise_preferred, this is interesting since the element is on-premise anyway. If the annotation value is in_cloud_preferred, this has a problem to a certain degree, since an in-cloud preferred element is distributed on-premise. Finally, the algorithm stores the result of the privacy risk analysis into a structure variable distributionPrivacyRisk, with a reference to the particular distribution option. The complete algorithm is shown below:

Algorithm 7.9.1	**Check Privacy Risk**

Input: *Business Process Model* inputModel, integer byref numberofDistributions, structure byref distributionCost, structure byref distributionPrivacyRisk
Output: *total privacy analysis for a particular distribution*

```
CHECK-PRIVACY-RISK (inputModel)
on_premise_preferred ← null
in_cloud_preferred ← null
forall a in inputModel do
    if a is activity || a is data then
        if a.hasProfile("BPMN_PrivacyRisk") then
            if a.attrValue("in_cloud") = true then
                if a.attrValue("PrivacyRisk") = "default" then
                    return
                else if a.attrValue("PrivacyRisk")="On_premise_preferred" then
                    ▹ on premise preferred activity or data is moved to cloud
                    if on_premise_preferred = null then
                        on_premise_preferred ← on_premise_preferred + "" + a.name()
                    else
                        on_premise_preferred ← on_premise_preferred + "," + a.name()
                    end if
                    else if a.attrValue("PrivacyRisk")="In_cloud_preferred" then
                        ▹ do nothing, it is in-cloud anyway which is good
                    end if
                if a.attrValue("in_cloud") = false
                    if a.attrValue("PrivacyRisk") = "Necessarily_on_premise" then
                        ▹ do nothing, it is on-premise anyway which is good
                    else if a.attrValue("PrivacyRisk") = "default" then
                        ▹ do nothing because there is not problem in this case
                else if a.attrValue("PrivacyRisk")="On_premise_preferred" then
                    ▹ no problem, it is already on-premise
                else if a.attrValue("PrivacyRisk")="In_cloud_preferred" then
```

```
                    ▹ interesting, because in-cloud preferred item is on-premise
                    if In_cloud_preferred = null then
                        in_cloud_preferred ← in_cloud_preferred + “” + a.name()
                    else
                        in_cloud_preferred ← in_cloud_preferred + “,” + a.name()
                    end if
                end if
            end if
        end if
    end forall
    if on_premise_preferred != null then
        on_premise_preferred ← on_premise_preferred + “-” + “preferred on-premise, but now in-
            cloud”
    end if
    if in_cloud_preferred!= null then
        in_cloud_preferred ← in_cloud_preferred + “-” + “preferred on-premise, but now in-cloud”
    end if
    if in_cloud_preferred!= null & on_premise_preferred != null then
        on_premise_preferred ← “No Privacy Risk problem”
    end if
    distributionPrivacyRisk.add(“DistributionPrivacyRisk” + numberofDistributions) ← in_cloud_preferred
                                                                  + on_premise_preferred
end algorithm
```

7.6 Analysis and Ranking Algorithms

These algorithms are used for analyzing distribution options and their consequences by calling the other algorithms we have defined, and for ranking these distributions based on cost benefits. These algorithms are discussed below.

7.6.1 Analysis

This algorithm is responsible for keeping track of the number of distribution options, calculating cost benefits, checking privacy, storing the cost list of each distribution option with a reference to the distribution option. The algorithm finally displays the distribution options and their consequences by calling to the corresponding algorithms. The complete description of this algorithm is shown below:

Algorithm 7.10.1 **Analyze Distribution Option**

Input: *Business Process Model* inputModel, integer byref numberofDistributions, structure byref distributionCost, structure byref distributionPrivacyRisk, byref costList
Output: *Cost benefit and privacy risk analysis of a particular distribution option*

```
ANALYZE-DISTRIBUTION (inputModel, byref numberofDistributions, byref distributionCost, byref
        distributionPrivacyRisk, byref costList)
▹ Keeping track of the number of distribution options
numberofDistributions ← numberofDistributions + 1
▹ Calling cost benefit function
costBenefit ← calculateCostBenefit(inputModel)
▹ checking privacy
checkPrivacyRisk(inputModel, numberofDistributions, distributionPrivacyRisk)
▹ Storing cost benefit in a structure
distributionCost.add( "DistributionCost" + numberofDistributions.toString(), costBenefit)
▹ Storing copy of the cost benefit for ranking
costList.add(distributionCost.valueFor("DistributionCost" + numberofDistributions.toString()))
▹ Showing result in a viewpoint
display(inputModel, numberofDistributions, distributionCost, distributionPrivacyRisk)
return true
end algorithm
```

7.6.2 Ranking

This algorithm is used to rank the identified distribution options based on cost benefits. The algorithm uses a built-in functions max() in order to find the maximum cost benefit from the cost benefit list. When the maximum cost benefit is found, the distribution option with that cost benefit is displayed and the cost benefit is removed from the cost benefit list. This process is iterative until all distribution options and their consequences are ranked based on cost benefits.

Algorithm 7.11.1 **Rank Distribution Options**

Input: *Business Process Model* inputModel, integer byref numberofDistributions, structure byref distributionCost, structure byref distributionPrivacyRisk
Output: *distribution options ranked based on cost benefits*

```
RANK-DISTRIBUTIONS (inputModel)
rank ← 1
maxRank ← numberofDistributions
while costList.size() != 0 do
```

```
        result ← max(costList)
        i ← 1
        while i <= numberofDistributions do
                if distributionCost.valueFor(“DistributionCost” + i.toString()) = result then
                        if  rank <= maxRank
                                output ← rank, “distribution” + i.toString(), result,
                                distributionPrivacyRisk.valueFor(“DistributionPrivacyRisk” +
                                                        i.toString())
                                rank ← rank + 1
                        end if
                end if
        i ← i + 1
        end while
        costList.remove(result)
end while
return true
end algorithm
```

7.7 Display Distribution Options and Consequences

This algorithm displays the identified distribution options in a table viewpoint for the final selection. Activities, data items, their detailed distribution location, and their corresponding cost benefits and privacy risks are displayed. Private and excluded activities or data items are not displayed.

Algorithm 7.12.1 Display Distribution Options and Consequences

Input: *Business Process Model* inputModel, integer byref numberofDistributions, structure byref distributionCost, structure byref distributionPrivacyRisk

Output: *displays each of the distributions and analysis results in a table view*

```
DISPLAY-DISTRIBUTION-AND-CONSEQUENCES (inputModel, byref numberofDistributions, byref
                                        distributionCost, byref  distributionPrivacyRisk)
▹   displaying the distribution cost and privacy analysis results for a particular distribution
output ← numberofDistributions, distributionCost.valueFor(“DistributionCost” + numberDistributions),
        distributionPrivacyRisk.valueFor(“DistributionPrivacyRisk” +
        numberDistributions)
forall a in inputModel do
         if a is activity || a is data then
                if a.attrValue(“in_cloud”) = true then
                        output ← a, “in_cloud”
```

```
        else
            display = true
            if a.hasAttr("PrivacyRisk") then
                if a.attrValue("PrivacyRisk") = "Necessarily_on_premise" then
                    display = false
                end if
            end if
            if a.attrValue("limitDistributions") != "exclude" & display = true then
                output ← a, "on_premise"
            end if
        end if
    end if
end forall
end algorithm
```

Chapter 8: Validation

The purpose of this chapter is to validate our work by applying our results to a fictitious but realistic case study, the ArchiSurance Broker.

This chapter is structured as follows: Section 8.1 explains the purpose of this validation; Section 8.2 describes the ArchiSurance Broker case study and its process model in the BiZZdesign Architect modeling tool; Section 8.3 explains the annotation step by using our annotation language; Section 8.4 discusses the analysis step by using our algorithms and Section 8.5 reports on the results of the validation.

8.1 Validation Purpose

The purpose of this validation is to ensure that the intended objectives of our automated system are met by using a realistic example. We validate the system against the following issues:

- Does the system always distribute elements annotated as Necessarily_on_premise on-premise?
- Does the system always give a warning whenever an element annotated with On_premise_preferred is actually distributed in-cloud, or whenever an element annotated with In_cloud_preferred is distributed on-premise?
- Does the system generate correct cost benefit calculations?
- Does the system rank distribution options correctly according to cost benefits?
- How is the identification and ranking of distribution options and their consequences improved compared with if this would have been done by hand?

8.2 ArchiSurance Broker

For validating our automated system, we present a fictitious but realistic insurance broker case study that has been published in [48]. This case study is motivated by the ArchiSurance, which is a large fictitious insurance company that has been formed by combining other three independent companies. ArchiSurance is widely used as a running example for illustrating the realistic use of the ArchiMate modeling language [49]. In order to avoid an incomplete and faulty insurance package, ArchiSurance offers car insurance services to customers via ArchiSurance Broker. The ArchiSurance Broker is an intermediary between customers and ArchiSurance. The ArchiSurance Broker works as follows:

- The ArchiSurance Broker receives car insurance request data from customers.
- The ArchiSurance Broker receives complete insurance information from customers.
- The ArchiSurance Broker creates an insurance package for the customer.
- The ArchiSurance Broker finds an appropriate insurer from its database.
- The ArchiSurance Broker informs the customer about the appropriate matches and offerings.

- If the customer agrees, the ArchiSurance Broker creates the complete insurance package and hands it in to ArchiSurance.
- Finally, the ArchiSurance Broker receives payment from ArchiSurance, and the process terminates.

The business process of the ArchiSurance Broker is modeled in the BiZZdesign Architect modeling tool by using BPMN 2.0 as a modeling language as shown in Figure 20.

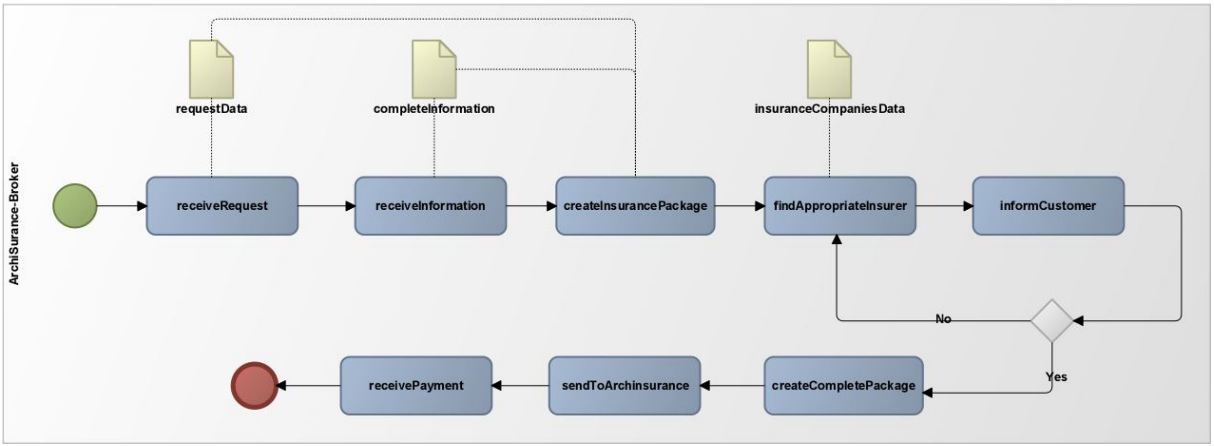

Figure 20 ArchiSurance Broker Model in BiZZdesign Architect Modeling tool by using BPMN 2.0 as a modeling language

Consider that the ArchiSurance Broker runs traditional on-premise BPM tools in order to organize, visualize, analyze, optimize and continuously improve their processes. In order to benefit from the cheaper resources of cloud computing, the ArchiSurance Broker has decided to make use of Cloud-based BPM by distributing total or parts of their business processes into collaborating in-cloud and on-premise engines and databases.

The ArchiSurance Broker can use our automated system to identify and rank distribution options, and their consequences in order to facilitate choices in two steps. In the annotation step, the ArchiSurance Broker process model is annotated by using our annotation language. In the analysis step, the annotated ArchiSurance Broker model is given as an input to the algorithms in our automated system.

8.3 Annotation Step

In this step the ArchiSurance Broker process model is annotated with cloud-related information and on-premise costs by using our annotation language as shown in Figure 21. When an activity, data item or a data association is annotated, an annotation marker is displayed on the element. The annotation language elements available for annotating the ArchiSurance Broker activities are:

BPMN_LimitDistribution, BPMN_ComputingCapacityCost, BPMN_PrivacyRisk, BPMN_On_premiseCost and the cloud-related information that characterize them as shown in Figure 22. The annotation language elements (profiles) available for annotating the ArchiSurance Broker data items are: BPMN_LimitDistribution, BPMN_DataStorageCost, BPMN_PrivacyRisk, BPMN_On_premiseCost and their corresponding cloud-related information. The annotation language element available for annotating data associations is BPMN_DataCommunicationCost and its cloud-related information.

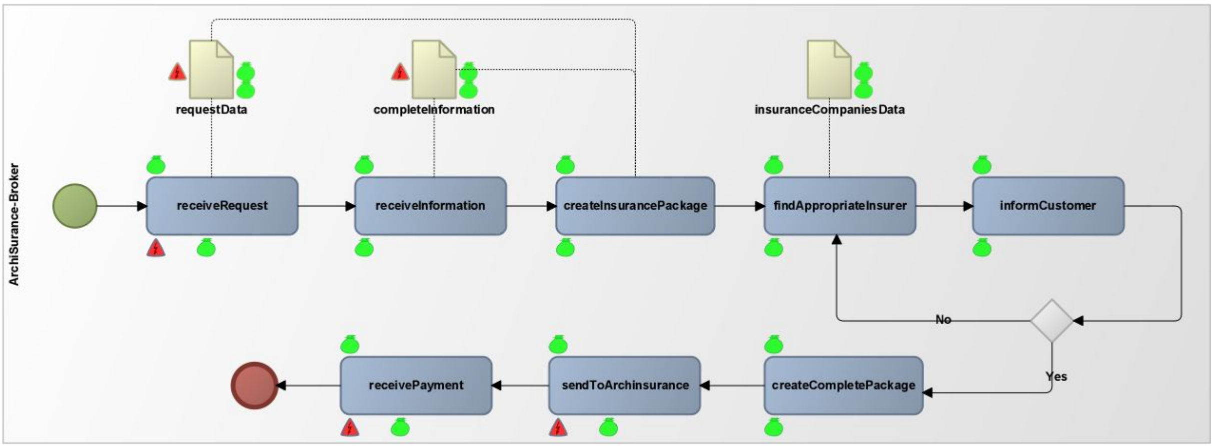

Figure 21 ArchiSurance Broker model after annotation by using our annotation language in the BiZZdesign Architect Modeling tool

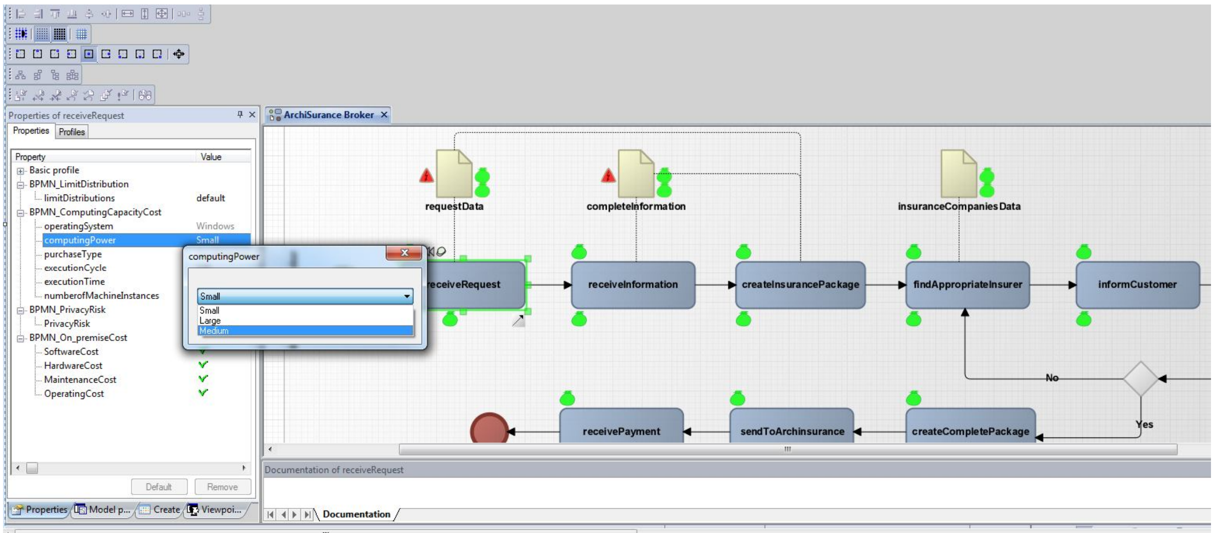

Figure 22 The annotation language elements available for annotating ArchiSurance Broker activities in the BiZZdesign Architect modeling tool

8.4 Analysis Step

In this step, the annotated ArchiSurance Broker model is given as an input to our algorithms implemented in the BiZZdesign advanced scripting language, and available as viewpoints. This is done with a simple mouse click on our main viewpoint i.e. AutomaticDistributionAnalysis that implements the main algorithms as shown in Figure 23.

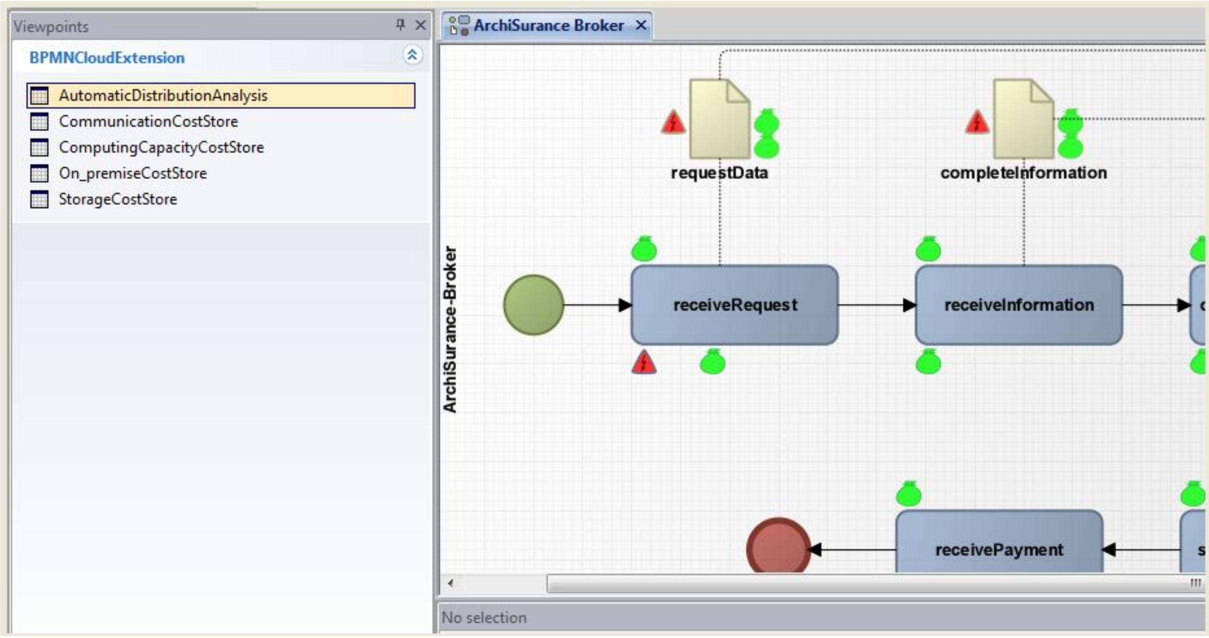

Figure 23 Implementations of our algorithms available as viewpoints on the BiZZdesign Architect modeling tool

Upon the execution of this viewpoint, the distribution options and their consequences regarding cost benefits and privacy risks are automatically identified and ranked. The result of the analysis is displayed in detail in a tabular form as shown in Table 4. Towards the end of the report the identified distribution options are ranked based on cost benefits as shown in Table 5.

Distributin Options	Distribution Location	Cost Benefit Analysis	Privacy Risk Analysis
Distribution1		EUR -3585.00	No Privacy Risk!
createInsurancePackage	on_premise		
findAppropriateInsurer	on_premise		
informCustomer	on_premise		
createCompletePackage	on_premise		
sendToArchinsurance	on_premise		
receivePayment	on_premise		
receiveRequest	on_premise		
receiveInformation	on_premise		
requestData	on_premise		
completeInformation	on_premise		
insuranceCompaniesData	in_cloud		

Table 4 Sample detailed distribution options and their consequences in a table output type

The identified and ranked distribution options and their consequences are also displayed in a tabular form The ArchiSurance Broker model contains eight activities and three data items in total. Since all elements are annotated and they can be assigned to the cloud, the number of identified and ranked distribution options and their consequences is $2047 = 2^{11} - 1$, excluding the original on-premise distribution option. The analysis automatically generated 552 pages report. This report contains the list of distribution options, the detailed distribution location of each element in each of these distributions, the cost saving that can be expected from each distribution, and the associated privacy risk. Towards the end of the report, the identified distribution options are ranked in terms of cost benefits in order to facilitate choices. The top ranked distribution options with acceptable privacy risk can be chosen for the cloud-based BPM.

Rank out of 2047 Options	Distribution Options	Cost Benefit Analysis	Privacy Risk Analysis
1	distribution1812	EUR 9882.99	receivePayment, receiveRequest - preferred on-premise, but now in-cloud!
2	distribution1980	EUR 9310.64	receivePayment, receiveRequest - preferred on-premise, but now in-cloud!
3	distribution1968	EUR 9267.99	receivePayment, receiveRequest - preferred on-premise, but now in-cloud!
4	distribution1466	EUR 9174.48	receivePayment, receiveRequest - preferred on-premise, but now in-cloud!
5	distribution1477	EUR 8790.99	receivePayment - preferred on-premise, but now in-cloud!
6	distribution1483	EUR 8711.29	receivePayment, receiveRequest - preferred on-premise, but now in-cloud!
7	distribution2034	EUR 8695.64	receivePayment, receiveRequest - preferred on-premise, but now in-cloud!
8	distribution1810	EUR 8602.13	receivePayment, receiveRequest - preferred on-premise, but now in-cloud!
9	distribution1760	EUR 8559.48	receivePayment, receiveRequest - preferred on-premise, but now in-cloud!
10	distribution1464	EUR 8545.39	receivePayment, receiveRequest - preferred on-premise, but now in-cloud!

Table 5 Sample top 10 ranked distribution options and their consequences

8.5 Validation Report

Based on the automatically identified and ranked distribution options and their consequences, we validated the following issues:

- Does the system always distribute an elements annotated as Necessarily_on_premise on-premise?

After running the system multiple times and manually checking the results, we conclude that the system ignores distribution options in which an element annotated as Necessarily_on_premise is distributed in-cloud. Therefore, all elements annotated Necessarily_on_premise are automatically distributed on-premise by the system. This guarantees that private elements are on-premise in every identified distribution option.

- Does the system always give a warning whenever an element annotated with On_premise_preferred is actually distributed in-cloud, or whenever an element annotated with In_cloud_preferred is distributed on-premise?

We found out that in all of the 2047 automatically identified distribution options, the system warns whenever an on-premise preferred element is distributed in-cloud or vice versa.

- Does the system generate correct cost benefit calculations?

Since we do not have a tested automated system for comparison, we manually calculated the cost benefit of 10 randomly handpicked distribution options and concluded that they are the same as the cost benefits calculated by the automated system.

- Does the system rank distribution options correctly according to cost benefits?

We manually checked the 2047 automatically identified and ranked distribution options. We concluded that the system has correctly ranked them based on cost benefits.

- How is the identification and ranking of distribution options and their consequences improved compared with if it would have been done by hand?

This validation is conducted based on expert interview. The interviewee is an experienced BPM consultant with 10 years work experience, and also with experience in the field of cloud computing. We gave the interviewee the same case study that we have used for validating our system automatically. We asked him to perform the identification and ranking of distribution options by hand. The interviewee stated that this is a very hard question, and that most organizations do not perform such detail decisions regarding cloud adoption. The interviewee said most organizations perform such identifications based on their intuitions and educated guesses. Subsequently, we discussed our solution with the interviewee. While he was quite impressed with our systematic approach, he required that the solution should also consider other factors as the ones we identified in Chapter 4 in order to be more realistic. Our solution does not consider these factors as this has been left for future work. The interviewee also advised that more ways to visualize the result are necessary. At the moment, our implementation only displays the result in tabular form. Therefore, this should also be considered in future implementations.

Chapter 9: Conclusions

This chapter gives the conclusions of this work by summarizing the answers for the research questions, identifying possible future areas of research and recommending some possible improvements to the BiZZdesign tools and languages.

This chapter is structured as follows: Section 9.1 summarizes the answers to the research questions addressed in this work; Section 9.2 identifies possible areas of future work and Section 9.3 recommends some possible improvements to the BiZZdesign tools and languages.

9.1 Answers to Research Questions

The answers for the research questions addressed in this work are summarized below:

- **RQ1: What are the most relevant cloud migration drivers and barriers that affect the identification and selection of distributions of business processes for cloud-based BPM?**

The most relevant cloud migration drivers and barriers mentioned in the literature have been identified in Chapter 4. Cost from the cloud migration drivers and privacy from the cloud migration barriers have been selected for further consideration.

- **RQ2: How are the most relevant cloud migration drivers and barriers mapped to process modeling constructs?**

BPMN 2.0 is selected as a business process modeling language for this work. Activities, data items and data associations are selected as the most relevant business process constructs. The cloud-related information regarding cost and privacy has been identified. This cloud-related information is mapped to activities, data items and data associations. An annotation language for annotating business processes with this cloud-related information has been developed. These issues are explained in Chapter 5.

- **RQ3: How to identify and rank distribution options of business processes into collaborating in-cloud and on-premise engines?**

We developed algorithms to identify and rank distribution options and their consequences. These algorithms are explained in Chapter 7. We also built an automated system for identifying and ranking distribution options based on annotated business processes. This system is explained in Chapter 5.

Our research has shown that the automatic identification and ranking of distribution options and their consequences for cloud-based BPM is feasible. This claim is validated by using a realistic case study and expert interviews as explained in Chapter 8.

9.2 Future work

The following additional future research directions would make our automated system more realistic.

- Identify cloud-related information regarding the cloud migration drivers and barriers we identified in Chapter 4. In this work we only considered one cloud migration driver, namely cost and one cloud migration barrier, namely privacy, so that our solution only supports cost and privacy. In the future, the solution can be extended in order to support more factors.

- The automated system needs to be tested in a real organization that would like to migrate to a cloud-based BPM environment. Although we have tested the automated system with a fictitious but realistic case study, the associated on-premise costs and cost benefit values are only rough estimates. It would be interesting to test if real organizations can actually profit from the cost benefits identified by our automated system. Furthermore, the case study consists of only 11 activities and data items. However, the business process model of organizations may have up to 25 activities and data items on average [3]. In such extreme situations, the performance of the distribution identification algorithm may need to be improved by applying other more efficient techniques such as Gray Code [50].

9.3 Recommendations for BiZZdesign

From our experience with the tools and languages of BiZZdesign, we recommend the following improvements:

- An editor with code completion and syntax highlighting for the Profile Definition Language.
- The Query tool editor for the scripting language could be improved by including code completion and syntax highlighting.
- The scripting language should support polymorphism for model elements. This would allow the developer to define multiple functions with the same name, but different modeling elements as parameters, model elements would then be dispatched to the function with the matching parameter. At the moment, the scripting language ignores the types of parameters in functions, and it does not allow the developer to define functions with the same name, but with different parameters. For example, the following code snippet is not allowed. The consequence is that programmers have to manually check the correct modeling element for each of these functions.

```
function analysis(Activity element){
    // code;
return;
```

```
}

function analysis(DataItem element){
    //code;
return;
}

forall modelElement in inputModel{
    analysis(modelElement);
}
```

- BiZZdesign Architect should support better printing facilities for the tabular form.

Bibliography

[1] M. Weske, "Business process management: concepts, languages, architectures. 2007," *Leipzig, Alemania: Springer-Verlag Berlin Heidelberg*, vol. 368.

[2] S. Brahe, "BPM on Top of SOA: Experiences from the Financial Industry," in *Business process management*, Springer, 2007, pp. 96–111.

[3] Y. B. Han, J. Y. Sun, G. L. Wang, and H. F. Li, "A Cloud-Based BPM Architecture with User-End Distribution of Non-Compute-Intensive Activities and Sensitive Data," *Journal of Computer Science and Technology*, vol. 25, no. 6, pp. 1157–1167, 2010.

[4] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: state-of-the-art and research challenges," *Journal of Internet Services and Applications*, vol. 1, no. 1, pp. 7–18, 2010.

[5] T. Anstett, F. Leymann, R. Mietzner, and S. Strauch, "Towards BPEL in the Cloud: Exploiting Different Delivery Models for the Execution of Business Processes," presented at the Services - I, 2009 World Conference on, 6, pp. 670–677.

[6] M. Menzel, M. Schönherr, J. Nimis, and S. Tai, "(MC2) 2: A Generic Decision-Making Framework and its Application to Cloud Computing," *arXiv preprint arXiv:1112.1851*, 2011.

[7] V. Pandey and V. Bansal, "A decision-making framework for IT outsourcing using the analytic hierarchy process."

[8] P. V. Beserra, A. Camara, R. Ximenes, A. B. Albuquerque, and N. C. Mendonca, "Cloudstep: A step-by-step decision process to support legacy application migration to the cloud," presented at the Maintenance and Evolution of Service-Oriented and Cloud-Based Systems (MESOCA), 2012 IEEE 6th International Workshop on the, 24, pp. 7–16.

[9] Maurice Buckley, CEO, NSAI, Joan Mulvihill, CEO, Irish Internet Association (IIA) with Richard Bruton, T.D., Minister for Jobs, Enterprise and Innovation., "Adopting the Cloud - decision support for cloud computing." Maurice Buckley, CEO, NSAI, Joan Mulvihill, CEO, Irish Internet Association (IIA) with Richard Bruton, T.D., Minister for Jobs, Enterprise and Innovation., 2012.

[10] D. H. Yang, S. Kim, C. Nam, and J. W. Min, "Developing a decision model for business process outsourcing," *Computers & Operations Research*, vol. 34, no. 12, pp. 3769–3778, 2007.

[11] P. Mell and T. Grance, "The NIST definition of cloud computing (draft)," *NIST special publication*, vol. 800, p. 145, 2011.

[12] "Office 365." [Online]. Available: http://www.microsoft.com/en-us/office365/what-is-office365.aspx. [Accessed: 02-Jan-2013].

[13] "Saleforce.com." [Online]. Available: http://www.salesforce.com/uk/. [Accessed: 02-Jan-2013].

[14] Google, "Google App Engine." [Online]. Available: https://developers.google.com/appengine/. [Accessed: 18-Dec-2012].

[15] "Amazon Simple Storage Service (Amazon S3)." [Online]. Available: http://aws.amazon.com/s3/. [Accessed: 19-Dec-2012].

[16] "Amazon Elastic Compute Cloud (Amazon EC2)." [Online]. Available: http://aws.amazon.com/ec2/. [Accessed: 19-Dec-2012].

[17] E. F. Duipmans, L. F. Pires, and L. O. B. da Silva Santos, "Towards a BPM Cloud Architecture with Data and Activity Distribution," presented at the Enterprise Distributed Object Computing Conference Workshops (EDOCW), 2012 IEEE 16th International, 10, pp. 165–171.

[18] M. Menzel and R. Ranjan, "CloudGenius: decision support for web server cloud migration," in *Proceedings of the 21st international conference on World Wide Web*, 2012, pp. 979–988.

[19] M. Hauck, M. Huber, M. Klems, S. Kounev, J. Müller-Quade, A. Pretschner, R. Reussner, and S. Tai, "Challenges and opportunities of cloud computing."

[20] A. Khajeh-Hosseini, I. Sommerville, J. Bogaerts, and P. Teregowda, "Decision Support Tools for Cloud Migration in the Enterprise," presented at the Cloud Computing (CLOUD), 2011 IEEE International Conference on, 4, pp. 541–548.

[21] M. Klems, J. Nimis, and S. Tai, "Do clouds compute? a framework for estimating the value of cloud computing," *Designing E-Business Systems. Markets, Services, and Networks*, pp. 110–123, 2009.

[22] D. G. Rosado, R. Gómez, D. Mellado, and E. Fernández-Medina, "Security analysis in the migration to cloud environments," *Future Internet*, vol. 4, no. 2, pp. 469–487, 2012.

[23] B. Johnson and Yanzhen Qu, “A Holistic Model for Making Cloud Migration Decision: A Consideration of Security, Architecture and Business Economics,” presented at the Parallel and Distributed Processing with Applications (ISPA), 2012 IEEE 10th International Symposium on, 10, pp. 435–441.
[24] Bu Xinyi and Xu Jingjing, “Developing a Decision Model for It Outsourcing Using Analytic Hierarchy Process,” presented at the Management and Service Science, 2009. MASS ’09. International Conference on, 20, pp. 1–5.
[25] “cloud-email.” [Online]. Available: http://www.cloudemail101.org/home.
[26] T. L. Saaty, “Decision making—the analytic hierarchy and network processes (AHP/ANP),” *Journal of systems science and systems engineering*, vol. 13, no. 1, pp. 1–35, 2004.
[27] “expertchoice.” [Online]. Available: http://expertchoice.com/.
[28] B. Martens, M. Walterbusch, and F. Teuteberg, “Costing of Cloud Computing Services: A Total Cost of Ownership Approach,” in *System Science (HICSS), 2012 45th Hawaii International Conference on*, 2012, pp. 1563–1572.
[29] IBM, “Assess enterprise applications for cloud migration.” [Online]. Available: http://www.ibm.com/developerworks/cloud/library/cl-assessport/. [Accessed: 31-Jan-2013].
[30] Andrew Reichman, “File Storage Costs Less In The Cloud Than In-House,” 25-Aug-2011. [Online]. Available: http://media.amazonwebservices.com/Forrester_File_Storage_Costs_Less_In_The_Cloud.pdf.
[31] S. Marston, Z. Li, S. Bandyopadhyay, J. Zhang, and A. Ghalsasi, “Cloud computing—The business perspective,” *Decision Support Systems*, vol. 51, no. 1, pp. 176–189, 2011.
[32] European Network and Information Security Agency (ENISA), “Benefits, risks and recommendations for information security,” 2009.
[33] A. Khajeh-Hosseini, D. Greenwood, and I. Sommerville, “Cloud migration: A case study of migrating an enterprise it system to iaas,” in *Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on*, 2010, pp. 450–457.
[34] M. Hajjat, X. Sun, Y.-W. E. Sung, D. Maltz, S. Rao, K. Sripanidkulchai, and M. Tawarmalani, “Cloudward bound: planning for beneficial migration of enterprise applications to the cloud,” *ACM SIGCOMM Computer Communication Review*, vol. 40, no. 4, pp. 243–254, 2010.
[35] P. Brebner and A. Liu, “Performance and cost assessment of cloud services,” in *Service-Oriented Computing*, Springer, 2011, pp. 39–50.
[36] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, and I. Stoica, “A view of cloud computing,” *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2010.
[37] K. R. Jackson, L. Ramakrishnan, K. Muriki, S. Canon, S. Cholia, J. Shalf, H. J. Wasserman, and N. J. Wright, “Performance analysis of high performance computing applications on the amazon web services cloud,” in *Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on*, 2010, pp. 159–168.
[38] G. Lewis, “Basics about cloud computing,” *Techreport. Software Engineering Institute Carnegie Mellon University*, 2010.
[39] “Collaboration in the cloud Untethered technologies for scholarly pursuits.” [Online]. Available: http://crln.acrl.org/content/72/8/454.full.
[40] S. Subashini and V. Kavitha, “A survey on security issues in service delivery models of cloud computing,” *Journal of Network and Computer Applications*, vol. 34, no. 1, pp. 1–11, 2011.
[41] ENISA, “Cloud Computing Risk Assessment,” 2009. [Online]. Available: http://www.enisa.europa.eu/activities/risk-management/files/deliverables/cloud-computing-risk-assessment.
[42] F. Doko, I. Mishkovski, and S. Filiposka, “Integrated business collaboration in the cloud,” presented at the Telecommunications Forum (TELFOR), 2012 20th, 20, pp. 1428–1431.
[43] “How AWS Pricing Works,” 10-Dec-2011. [Online]. Available: http://media.amazonwebservices.com/AWS_Pricing_Overview.pdf.
[44] “Google App Engine Pricing.” [Online]. Available: https://cloud.google.com/pricing/.
[45] “Pricing Details.” [Online]. Available: http://www.windowsazure.com/en-us/pricing/details/?currency-locale=en-us.

[46] Object Management Group, Inc. (OMG), “Business Process Model and Notation (BPMN),” Jan-2011. [Online]. Available: http://www.omg.org/spec/BPMN/2.0/PDF/. [Accessed: 28-Dec-2012].
[47] L. J. R. Stroppi, O. Chiotti, and P. D. Villarreal, “Extending BPMN 2.0: Method and Tool Support,” in *Business Process Model and Notation*, Springer, 2011, pp. 59–73.
[48] S. de Kinderen, K. Gaaloul, and H. E. Proper, “Integrating value modelling into archimate,” in *Exploring Services Science*, Springer, 2012, pp. 125–139.
[49] H. Jonkers, I. Band, and D. Quartel, “The ArchiSurance Case Study,” *White paper, The Open Group (Spring 2012)*, 2012.
[50] J. R. Bitner, G. Ehrlich, and E. M. Reingold, “Efficient generation of the binary reflected Gray code and its applications,” *Communications of the ACM*, vol. 19, no. 9, pp. 517–521, 1976.

Appendix: Compact Disc (CD)

This appendix explains the contents of a CD accompanying this thesis. The CD contains the implementation of our annotation language metamodel the BiZZdesign Architect profile definition language, the implementation of our model-based algorithm in the BiZZdesign Architect advanced scripting language and the automatically generated 552 pages analysis report containing the identified and ranked business process distribution options and their consequences for cloud-based BPM.

The CD is structured as follows:

- /BPMN4Cloud

This folder contains the abstract syntax of the annotation language metamodel.

- /Algorithms

This folder contains the implementations of the algorithms used for identifying and ranking distribution options and their consequences in the BiZZdesign advanced scripting language.

- /Analysis

This folder contains the automatically generated report from the case study.

- /Thesis

This folder contains this thesis report.

- /Presentation

This folder contains the presentation slides used during the graduation colloquium.