



HAL
open science

Supporting cloud resource allocation in configurable business process models

Emna Hachicha

► **To cite this version:**

Emna Hachicha. Supporting cloud resource allocation in configurable business process models. Networking and Internet Architecture [cs.NI]. Université Paris Saclay (COMUE), 2017. English. NNT : 2017SACLL007 . tel-01617093

HAL Id: tel-01617093

<https://theses.hal.science/tel-01617093v1>

Submitted on 16 Oct 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Supporting Cloud Resource Allocation in Configurable Business Process Models

Thèse de doctorat de l'Université Paris-Saclay
préparée à TELECOM SudParis

ÉCOLE DOCTORALE N°580
Sciences et technologies de l'information et de la communication

Spécialité de doctorat : Informatique

Thèse présentée et soutenue à Evry, le 22/09/2017, par

Emna Hachicha Belghith

Composition du Jury :

| | |
|--|--------------------|
| Mme Salima Benbernou Professeur, Université Paris Descartes | Rapporteur |
| M. Claude Godart Professeur, Université de Lorraine | Rapporteur |
| M. Nejib Ben Hadj-Alouane Professeur, Université Tunis El Manar | Examineur |
| Mme Daniela Grigori Professeur, Université Paris Dauphine | Examineur |
| M. Kais Klai Maître de Conférences HdR, Université Paris 13 | Examineur |
| Mme Amel Maamar Maître de Conférences HdR, Télécom SudParis | Examineur |
| M. Walid Gaaloul Professeur, Télécom SudParis | Directeur de thèse |

Titre : Supporter l'allocation de ressources Cloud dans les modèles de processus configurables

Mots clés : Modèles de processus configurables, allocation de ressources configurables, Cloud Computing, web sémantique, social computing, algorithmes génétiques

Résumé : Motivés par l'adaptation aux exigences commerciales et par la réduction des coûts de maintenance, les organisations externalisent leurs processus dans le Cloud. Selon l'Institut NIST, Cloud Computing est un modèle qui permet aux fournisseurs de partager leurs ressources et aux utilisateurs d'y accéder de manière à la demande. Dans un tel environnement, l'utilisation de modèles de processus configurables permet aux fournisseurs de processus de fournir un processus personnalisable entre par différents tenants en fonction de leurs besoins.

La conception et la configuration des ressources dans les processus, sont des tâches fastidieuses. D'une part, la perspective ressource est peu définie, ce qui empêche une interopérabilité efficace. D'autre part, la façon dont les ressources Cloud peuvent être configurées et intégrées n'a pas été traitée.

Dans cette thèse, nous proposons une approche pour supporter la modélisation et la configuration de l'allocation des ressources Cloud dans les modèles de processus configurables. Nous visons à (i) définir une description unifiée et formelle pour la perspective ressource, (ii) assurer une allocation de ressource correcte, sans conflits et optimisée, (iii) aider les fournisseurs de processus à modéliser leur allocation de ressources configurable de manière fine afin d'éviter des résultats complexes, et (iv) optimiser la sélection des ressources Cloud par rapport aux exigences liées aux propriétés Cloud et QoS.

Pour ce faire, nous proposons (i) des définitions formelles pour la gestion des ressources Cloud à l'aide de la sémantique, et (ii) d'étendre les processus configurables afin de permettre aux fournisseurs de processus de personnaliser l'allocation des ressources selon leurs besoins.

Title : Supporting Cloud resource Allocation in configurable process models

Keywords : Configurable process models, configurable resource allocation, Cloud Computing, Semantic Web, Social Computing, Genetic Algorithms

Abstract : Motivated by adapting to the rapid changing business requirements and reducing maintenance costs, organizations are outsourcing their processes using Cloud resources. Cloud Computing enables users sharing and accessing computing resources in an on-demand way. In such environment, using configurable process models enables Cloud providers to deliver a customizable process according to tenants needs.

The design and configuration of resources in process models are labor-intensive task. On the one hand, the resource perspective is poorly operated which prevent an efficient interoperability. On the other hand, the way in how Cloud resources can be configured and integrated are hardly handled.

In this thesis, we propose an approach for supporting the design and configuration of cloud resource allocation in configurable process models. We target to (i) define unified formal descriptions for the resource perspective, (ii) ensure a correct, free-of-conflict and optimized use of cloud resource consumption, (iii) assist process providers to design their configurable resource allocation in a fine-grained way to avoid complex and large results, and (iv) optimize the selection of Cloud resources against the requirements of the Cloud and QoS properties. To do so, we propose (i) formal definitions for cloud resource management using semantics and social techniques, and (ii) to extend configurable process models to allow cloud process providers to customize resource allocation according to their requirements.



to my family

Acknowledgment



First of all I thank God for making all things possible for me and giving me strength to go. A thesis journey is not always easy, but with the supervision, support and encouragement of many people, I have never regretted that started it.

I would like to thank all members of the jury. I thank Professor Claude Godart and Salima Benbernou for accepting being my thesis reviewers and for their attention and thoughtful comments. I also thank Professor Daniela Grigori, Professor Amel Maamar, Professor Kais Klai and Professor Nejib Ben Hadj-Alouane for accepting being my thesis examiners.

I would like to express my appreciation and gratitude to my supervisor Walid Gaaloul. His valuable advice, enthusiasm and constant support during this thesis allowed me to acquire new understandings and extend my experiences. He was not only an advisor but also a good friend. Thank you for your guidance, it has been a true pleasure and I hope that we can continue our collaboration.

I owe my deepest gratitude and warmest affection to the members of the computer science department of Telecom SudParis. I would like to thank Brigitte Houassine for her kind help and assistance. A special thank you to my office mates Rami, Nour, Souha, Nabila, Slim, Abderrahim, Rania and Hayet.

Thanks to my friends especially Wadii, Sirine, her lovely son Aziz, and Manel, Emna, Taha, Hayet, Soumaya for their emotional support and for all the beautiful moments we shared in France and Tunisia.

I am forever thankful to my parents especially my dear mother Ahlem, my grandmother Mamie Biha, my father Mohamed, my father in law Habib, my loving brother Amine, my parents in law: Bouthaina and Ghazi, my brother in law Ahmed, my sister in law Hanen, and their cute children Aziz and Malek, my brother in law Yassine, my sister in law Doniez. Your encouragement made me go forward and made me want to succeed. Thank you so much for having faith in me! You supported me without even you know it. I love you so much.

I express my deepest gratitude to my loving husband, Amine. His love and encouragement fostered me to concentrate at work. His understanding and support helped me to get through many difficult times. To my little angel, my beautiful daughter Dina whose smile and joy of life were the best encouragement for me. I hope that my thesis will be a source of pride for you.

I dedicate this thesis to all of you, my wonderful family.

Abstract

Organizations are recently more and more adopting Process-Aware Information Systems for managing and executing their service-based processes using process models referred to as business process models. Motivated by adapting to the rapid changing business requirements and reducing maintenance costs, organizations are outsourcing their processes using Cloud Computing resources. Cloud Computing enables users sharing and accessing computing resources in an ubiquitous, and on-demand way. In such multi-tenant environment, the resource perspective need to be explicitly defined in process models.

The integration of the resource perspective especially the *cloud resource allocation* is a current interesting topic that increasingly involves many researches in both academics and industry. The *design* and *configuration* of resources whether in simple or configurable process models are undoubtedly sensitive and labor-intensive task, in such environment. On the one hand, the resource perspective in Business process models is poorly operated in comparison to other perspectives such as the control-flow, or the organizational perspective. Although several approaches have been proposed in the literature, they all targeted human resources rather than cloud resources. This is due to the lack of formal semantic definitions, which prevent an efficient interoperability. On the other hand, despite of the fact that the concept of configurable process models is highly complementary to Cloud Computing, the way in how resources can be configured and integrated are hardly handled. Few proposals have been suggested in the literature to support resource variability, and cover required Cloud properties such as elasticity or multi-tenancy.

In this thesis, we address the above shortcomings by proposing an approach for supporting the design and configuration of cloud resource allocation in configurable process models. We target to (i) define unified formal descriptions for the resource perspective, (ii) ensure a correct, free-of-conflict and optimized use of cloud resource consumption, and (iii) assist process providers to design their configurable resource allocation in a *fine-grained* way to avoid complex and large results. To do so, we propose (i) formal definitions for cloud resource management using semantics and social techniques, and (ii) to extend configurable process models in order to allow cloud process providers to customize resource allocation according to their requirements. To validate our approach, we (i) develop three proof of concepts as extensions of existing business process modeling tools, (ii) perform experiments on real process models from large datasets. Experimental results show that our approach is feasible, accurate and has good performance in real use-cases.

Table of contents

| | | |
|----------|---|-----------|
| 1 | Introduction | 17 |
| 1.1 | Research context | 17 |
| 1.2 | Research problem: <i>How to integrate Cloud Resource Allocation in business process models?</i> | 21 |
| 1.2.1 | On Formalizing the resource perspective definition | 22 |
| 1.2.2 | On Supporting and Optimizing the resource variability in configurable process models | 23 |
| 1.3 | Motivating example | 24 |
| 1.4 | Thesis principles, objectives and contributions | 27 |
| 1.4.1 | Thesis principles | 27 |
| 1.4.2 | Thesis objectives | 27 |
| 1.4.3 | Thesis contributions | 28 |
| 1.5 | Thesis outline | 29 |
| 2 | Preliminaries and Basic Concepts | 31 |
| 2.1 | Process Modeling | 31 |
| 2.1.1 | Business Process Model and Notation (BPMN) | 32 |
| 2.1.2 | Configurable BPMN (C-BPMN) | 33 |
| 2.1.3 | Process Graphs | 35 |
| 2.2 | Semantic Business Process Modeling | 37 |
| 2.2.1 | Web Ontology Language for Web Services (OWL-S) | 37 |
| 2.2.2 | General Process Ontology (GPO) | 38 |
| 2.2.3 | Business Process Modeling Ontology (BPMO) | 39 |
| 2.3 | Cloud Computing | 41 |
| 2.3.1 | Generalities | 41 |
| 2.3.2 | Open Cloud Computing Interface (OCCI) | 42 |
| 3 | State of the Art | 47 |
| 3.1 | Introduction | 47 |
| 3.2 | On Formalizing the resource perspective in Business processes | 48 |
| 3.2.1 | Resource perspective in business processes | 48 |
| 3.2.2 | Resource orchestration in Cloud-based business processes | 49 |
| 3.2.3 | Semantics in business process models | 50 |
| 3.2.4 | Social business processes | 51 |
| 3.2.5 | Synthesis | 52 |
| 3.3 | On supporting and Optimizing the resource variability in configurable process models | 53 |
| 3.3.1 | Configurable process modeling | 54 |
| 3.3.2 | Resource Variability | 55 |

| | | |
|----------|---|-----------|
| 3.3.3 | Resource Allocation Optimization in business processes | 57 |
| 3.3.4 | QoS-aware business processes | 58 |
| 3.3.4.1 | QoS properties | 58 |
| 3.3.4.2 | Ecological properties | 59 |
| 3.3.5 | Synthesis | 61 |
| 3.4 | Conclusion | 62 |
| 4 | Supporting Cloud Resource Allocation in Service-based Business Processes | 63 |
| 4.1 | Introduction | 64 |
| 4.2 | Semantic Formalization of the Resource perspective in Business Processes | 65 |
| 4.2.1 | Resource perspective in Business Processes | 65 |
| 4.2.2 | Approach Overview | 66 |
| 4.2.3 | Extending the resource perspective in BPMN 2.0 | 67 |
| 4.2.4 | Semantic Model for Resource management in Business processes | 69 |
| 4.2.4.1 | Cloud-based process Ontology | 69 |
| 4.2.4.2 | Resource Constraints verification Rules | 72 |
| 4.2.4.3 | Process resource Properties and events | 72 |
| 4.2.4.4 | Rules formalization | 74 |
| 4.3 | Supporting Process Socialization in Cloud-based Business Process Models | 75 |
| 4.3.1 | Approach Overview | 75 |
| 4.3.2 | Social-based Resource Management in Business Processes . . . | 76 |
| 4.3.2.1 | So-CloudPrO: Social Cloud business Process Ontology | 76 |
| 4.3.2.2 | Ontology Definition | 76 |
| 4.3.2.3 | OCCI Resource Structure | 80 |
| 4.3.2.4 | Resource social dependencies | 80 |
| 4.3.3 | Resolving resource-based Conflicts using SWRL Strategies . . . | 82 |
| 4.3.3.1 | Instantiation Constraints | 82 |
| 4.3.3.2 | Resource and Task Constraints | 82 |
| 4.3.3.3 | Resolution Strategies | 83 |
| 4.4 | Evaluation and validation | 86 |
| 4.4.1 | Ontology Validation | 86 |
| 4.4.2 | Supporting Cloud resource Descriptions | 87 |
| 4.4.3 | Cloud resource Knowledge Base | 88 |
| 4.5 | Conclusion | 88 |
| 5 | Configurable Resource Allocation for Multi-tenant Process Development in the Cloud | 91 |
| 5.1 | Introduction | 92 |
| 5.2 | Configurable Cloud Resource Allocation | 93 |
| 5.2.1 | Configurable Resource Assignment Operator | 93 |
| 5.2.2 | Configurable Resource Elasticity Operator | 95 |

| | | |
|----------|--|------------|
| 5.2.3 | Configurable Resource Sharing/Batching Operator | 97 |
| 5.3 | Optimization of Cloud resource allocation using Genetic Algorithms | 98 |
| 5.3.1 | Motivating Example | 99 |
| 5.3.2 | Cloud-specific Resource Properties (CRP) Problem | 102 |
| 5.3.3 | Problem Formalization using Genetic Algorithm | 106 |
| 5.3.3.1 | Genome Encoding | 106 |
| 5.3.3.2 | Fitness Function considering Green properties | 108 |
| 5.3.3.3 | Fitness Function considering QoS properties | 110 |
| 5.4 | Evaluation and validation | 111 |
| 5.4.1 | Supporting Configurable Resource Allocation | 112 |
| 5.4.2 | Experimentation | 113 |
| 5.4.2.1 | Structural Complexity experiments | 113 |
| 5.4.2.2 | Solving optimal cloud resource allocation | 115 |
| 5.5 | Threats to Validity | 118 |
| 5.6 | Conclusion | 118 |
| 6 | Conclusion and Future Works | 121 |
| 6.1 | Contributions | 121 |
| 6.2 | Future work | 123 |
| | Appendices | 125 |
| A | List of Publications | 127 |
| B | Proof of Concepts | 129 |

List of Tables

| | | |
|-----|--|-----|
| 2.1 | Configuration constraints of configurable gateways | 34 |
| 3.1 | Comparative table for the evaluation of previous approaches | 52 |
| 3.2 | Synthesis of the approaches for formalizing the resource perspective in Business processes | 53 |
| 3.3 | Evaluation of previous approaches | 61 |
| 3.4 | Synthesis of the approaches for supporting and Optimizing the resource variability in configurable process models | 61 |
| 4.1 | Resource Dependencies Descriptions | 71 |
| 4.2 | Cloud resource properties | 80 |
| 4.3 | Description of BPs relations | 81 |
| 4.4 | SWRL strategies for resource conflicts resolution | 85 |
| 4.5 | S-CloudPrO coverage to OCCI standard | 86 |
| 5.1 | Configurable assignment parameters and configuration constraints . . | 94 |
| 5.2 | Configurable elasticity parameters and configuration constraints | 96 |
| 5.3 | Configurable sharing/batching parameters and configuration constraints | 98 |
| 5.4 | CRP requirements | 104 |
| 5.5 | Green properties Description | 108 |
| 5.6 | Green metrics of cloud resources | 109 |
| 5.7 | Structural Complexity metrics for different approaches | 114 |

List of Figures

| | | |
|------|---|-----|
| 1.1 | Configuration and individualization of a configurable process model . . | 19 |
| 1.2 | Resource Orchestration phases in configurable BPM lifecycle | 20 |
| 1.3 | Research Problem | 22 |
| 1.4 | A Service Supervision Business Process | 24 |
| 1.5 | A configurable service supervision process | 25 |
| 1.6 | Variant 1: A process variant derived from the configurable process in Figure 1.5 and its allocated cloud resources | 25 |
| 1.7 | Variant 2: A process variant derived from the configurable process in Figure 1.5 and its allocated cloud resources | 25 |
| 2.1 | A configurable activity and its possible configuration choices | 34 |
| 2.2 | Configurable events and their possible configuration choices | 35 |
| 2.3 | An example of a process graph | 36 |
| 2.4 | The Process ontology in OWL-S [1] | 38 |
| 2.5 | General process ontology (GPO) [2] represented using RML [3] | 39 |
| 2.6 | The Business Process Modeling Ontology (BPMO) [4] | 40 |
| 2.7 | Conceptual Reference Model of Cloud Computing (defined by NIST) . | 41 |
| 2.8 | OCCI place in provider architecture (defined by OGF) | 43 |
| 2.9 | Class diagram of OCCI core model | 43 |
| 2.10 | Overview Diagram of OCCI Infrastructure types | 44 |
| 3.1 | The audio editing process variant modeled in C-iEPC notation [5] . . | 56 |
| 3.2 | The cost-informed process support framework [6] | 59 |
| 3.3 | Green Business Process Classification [7] | 60 |
| 4.1 | Approach Overview | 66 |
| 4.2 | Extension of the resource element in BPMN | 67 |
| 4.3 | The CloudPrO ontology | 70 |
| 4.4 | Overview | 75 |
| 4.5 | So-CloudPrO representation | 77 |
| 4.6 | Lifecycle of resource state | 79 |
| 4.7 | Lifecycle of task state | 79 |
| 4.8 | Application screenshot: Supporting resource allocation | 87 |
| 5.1 | Configurable resource allocation operators | 93 |
| 5.2 | A configurable service supervision process with configurable resource operators | 100 |
| 5.3 | Variant 1: A process derived from the configurable process in Figure 5.2 with its allocated cloud resources | 101 |
| 5.4 | Variant 2: A process derived from the configurable process in Figure 5.2 with its allocated cloud resources | 101 |

| | | |
|------|--|-----|
| 5.5 | Configurable business process example | 106 |
| 5.6 | Cloud resource configuration represented as bits | 107 |
| 5.7 | Application Screenshot: Configurable Resource Allocation | 112 |
| 5.8 | fragments of configurable processes of the three models | 113 |
| 5.9 | The impact of the of δ on the quality of the solution | 116 |
| 5.10 | Comparing Genetic-based approach using QoS properties with LIP . . | 117 |
| 5.11 | Comparing Genetic-based approach using green properties with LIP . | 117 |

Introduction

Contents

| | | |
|------------|--|-----------|
| 1.1 | Research context | 17 |
| 1.2 | Research problem: <i>How to integrate Cloud Resource Allocation in business process models?</i> | 21 |
| 1.2.1 | On Formalizing the resource perspective definition | 22 |
| 1.2.2 | On Supporting and Optimizing the resource variability in configurable process models | 23 |
| 1.3 | Motivating example | 24 |
| 1.4 | Thesis principles, objectives and contributions | 27 |
| 1.4.1 | Thesis principles | 27 |
| 1.4.2 | Thesis objectives | 27 |
| 1.4.3 | Thesis contributions | 28 |
| 1.5 | Thesis outline | 29 |

1.1 Research context

Due to competitive business environments, enterprises should support efficient Information Technology (IT) in order to achieve excellent performance processes management [8, 9]. Therefore, Process-Aware Information Systems (PAIS) have widely been adopted, which depict software systems that permit to manage and execute operational processes using process models [10]. Business Process Management (BPM) are examples of such systems [11–15].

Then, enterprises are motivated by the need of adopting agile, flexible and cost effective business processes (BPs). To this end, they are looking for available services outside of their organizations to quickly adapt to new business requirements and also reduce process development and maintenance costs. Cloud Computing is recently gaining momentum due to its capability of outsourcing service-based BPs based on a scalable pay-per-use model. According to the National Institute of Standards and Technology (NIST), Cloud Computing is a model that enables providers sharing their computing resources (e.g., networks, servers, storage, applications, and services) and

users accessing them in an ubiquitous, convenient and on-demand way with a minimal management effort [16]. In such environment, using configurable process models [17] is more desirable especially in Cloud infrastructure [18] than the classical method that is based on using ad-hoc customizations to fit the needs of different organizations. Such models enable a cloud business process provider to afford a customizable process that can be configured by different tenants depending on their specific requirements [18].

Resource orchestration stands for the key technology to correctly exploit the cloud potential [19]. According to R&D community, resource orchestration is a set of operations for selecting, deploying, monitoring, and dynamically controlling the hardware and software resources to be delivered to end users. We are interested in this thesis in Cloud Resource Allocation for service-based business processes that consists in selecting the most suitable resources required by the different business activities. This task, which is a sensitive and serious task, is highly important and should be well managed for ensuring a correct and optimal business process completion.

Even though BPM researches have widely improved the performance of their business processes, several challenges still remain for effective and optimal cloud resource allocation with respect to the fast changing business needs [20]. In such environment, while the control-flow perspective have extensively been studied the resource perspective is not well defined [21]. Besides, resource descriptions are of a high heterogeneity, which prevents an easy and efficient interoperability. Therefore, seeking for a formalized and unified understanding becomes a must to ensure the interoperability between cloud process providers. To this end, different approaches have been proposed for extending the definition of the resource perspective [22–25], for providing formal semantic specifications to this perspective [26,27], and for realizing a correct and optimal resource allocation [28]. Furthermore, the specifics of cloud computing, specifically in how resources can be configured and integrated, are hardly considered in configurable process modeling. Thus, the resource allocation in business processes should be flexible, customizable in process models. Nevertheless, few proposals have been suggested to address the variability and the adaptability of cloud resource allocation in BPM [5,29,30].

On the one hand, formal and unified definitions were introduced [31,32], under the SUPER research project ¹, to BPM by using Semantic Web and Semantic Web Services. They provide semantic annotations and constraints in order to apply reasoning techniques for discovery, composition, mediation and execution of business processes. They have developed the Business Process Modeling Ontology [32] (BPMO) which aims at representing business processes at an abstract level of detail in order to ensure the interoperability between process modeled with different languages. However, such approaches do not take into account the characteristics of Cloud environment to provide formal definitions for Cloud resources in BPM.

On the other hand, configurable process models were presented in order to enable flexibility into process models [17,33]. These models permit an explicit representation

¹<http://www.ip-super.org>

of commonalities and differences of similar processes into one customizable process model. A configurable process model is a generic model which includes several process variants of a same business process in a given domain through variation points. These variation points are referred to as *configurable elements* and allow for multiple design options in the process. Such model enables to process analysts to have a global view on the commonalities and differences between multiple variants of a business process. According to specific requirements, this model could be configured by selecting one design option for each configurable element. Afterwards, an individualized process variant is derived from the set of selected configurations with a minimal design effort.

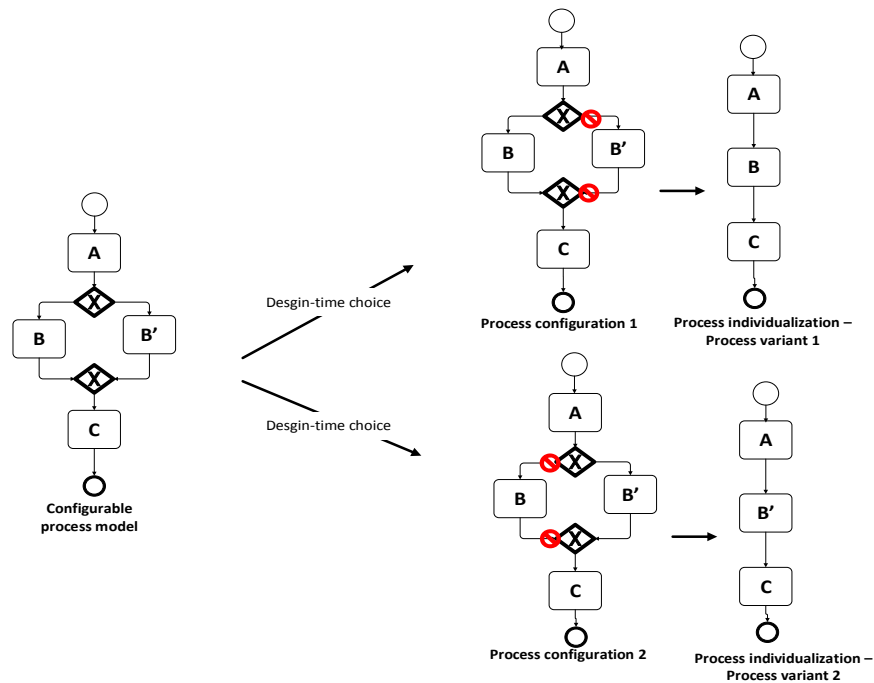


Figure 1.1: Configuration and individualization of a configurable process model

We present a simple example of a configurable process model in Figure. 1.1. On the left side of the figure, a configurable process modeled with the Configurable Business Process Model and Notation (C-BPMN) is depicted. BPMN basically consists of three main elements for modeling the control-flow in a business process: event, activity, and gateway. More details about these elements are discussed in Section 2.1.1. C-BPMN allows the control-flow elements to be configurable. Configurable elements are graphically modeled with thick lines. Returning to our example, the configurable process process contains 4 activities: A , B , B' , and C . B and B' are connected through a configurable XOR denoted as XOR^c . Unlike the ordinary control-flow gateway, the XOR^c does not represent a run-time decision. It represents a design choice that will need to be made by an analyst to adapt the configurable process

model to a particular setting. For instance, one analyst may choose to exclude the functionality implemented by the task B' . This exclusion corresponds to the fact to block the path of XOR^c leading to B' (see Process configuration 1 in Figure. 1.1). Then, the individualization phase consists of (i) deriving a process variant from the configured process that does not contain the elements excluded during configuration and (ii) mapping the configurable elements to normal ones. For example, in Figure. 1.1), the derived variant (see Process individualization 2 - Process variant 2) does not contain B and the configurable XOR gateways are mapped onto sequences. This variant does not contain any configurable element and therefore can be executed by the PAIS. Nevertheless, approaches working on such models lack supporting variability at the resource perspective level and optimizing the Cloud resource use.

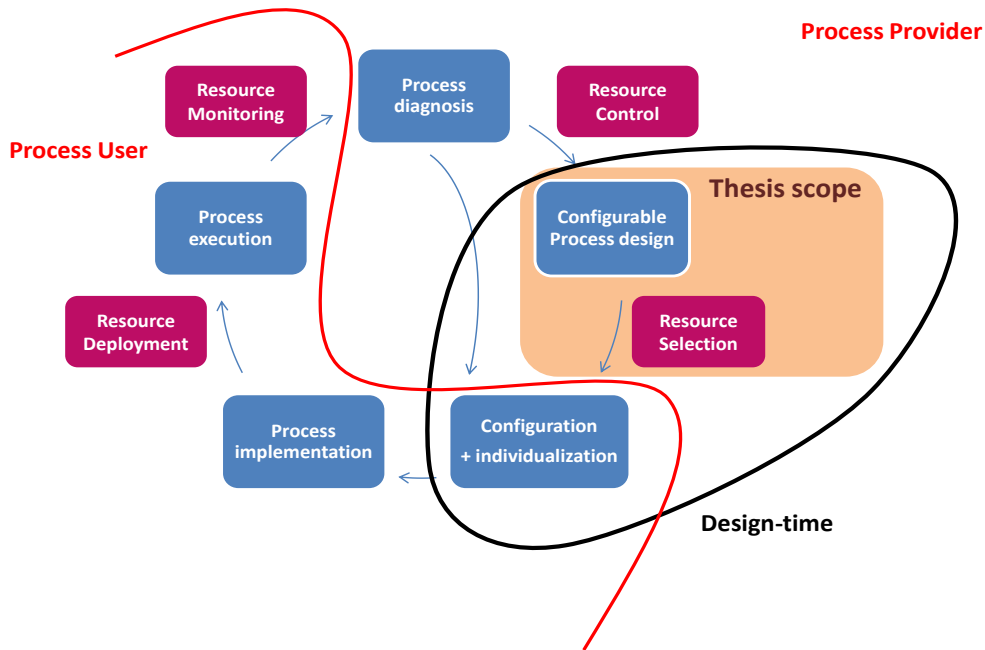


Figure 1.2: Resource Orchestration phases in configurable BPM lifecycle

More recent R&D activities on enhancing the resource perspective into BPM through semantic platforms and on supporting the variability using configurable process models, have been proposed. Resource orchestration and more specifically resource allocation issue, in multi-tenant business processes, have been investigated. We mean by multi-tenant business process the fact that, on the one hand multiple affiliates use this process model according to their specific needs and on the other hand the same process model could be outsourced by multiple Cloud process providers.

Concretely, as mentioned before many approaches have been proposed to provide formal and unified definitions using semantics since there is a high heterogeneity among the process providers in terms of resource description [34,35]. However, these proposals lack integrating the specificities of Cloud resources such as the elasticity, shareability, etc. Besides, several approaches have been proposed for configurable process modeling with a main focus on configuring the control-flow [36,37]. Since then, the issue of supporting the resource allocation, especially for Cloud resources, in BPM is highlighted.

Figure 1.2 denotes the resource orchestration phases in the configurable BPM lifecycle that is inspired from [38]. First, to show the configurable model specificity we add the *configuration+individualization* step to the traditional BPM lifecycle and replace the process design phase with *configurable process design*. Second, we add the different phases of the resource orchestration in accordance of the ordinary steps of BPM. The *resource selection* and the *configurable process design* are mainly the scope of this thesis work. The three steps *configurable process design*, *resource selection* and *configuration+individualization* are realized at design-time. Yet, we partially take into account the processing at the run-time level (*Process Implementation* and *Resource Deployment*). The lifecycle steps are performed by two roles (i) *the process provider* who is responsible of designing the configurable process, providing the required resources and analyzing the designed configurable process, and (ii) *the process user* who is responsible of configuring and individualizing the configurable process with the desired resources and then executing the diagnosis phases.

1.2 Research problem: *How to integrate Cloud Resource Allocation in business process models?*

As mentioned before, the resource orchestration in BPM becomes an active research area i.e., resource selection (including resource allocation), resource deployment, monitoring, and resource control. The aim of this vision is to ensure a correct use of cloud resources by the multiple activities in business processes. Our focus herein is the resource allocation in business that are deployed in Cloud environments. We illustrate this research problem in Figure 1.3.

First, users from several organizations need for their business processes, whether configurable or not, to consume cloud resources originating from different cloud providers. As these cloud providers have different policies of resource consumption, APIs, and networking models, a unified common understanding becomes a must (section 1.2.1). Second, process users require to configure their desirable resources as they can choose the configuration of the control-flow elements in configurable process models, with the aim of optimizing their use (section 1.2.2).

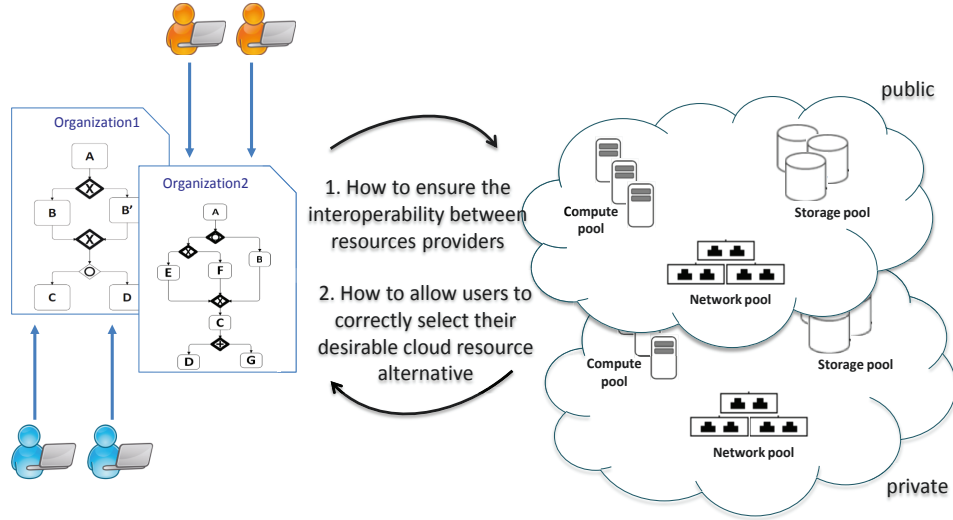


Figure 1.3: Research Problem

1.2.1 On Formalizing the resource perspective definition

BPM is among the fields that embraces the growth of Cloud Computing so that a better performance level of BPs over a lower operating cost is achieved. The combination of cloud and BPM has drawn the attention of the R&D community [39, 40].

Motivated by analyzing BP models from the resource perspective, several works have been proposed to integrate human resource description, interactions, roles, etc [41, 42]. Nevertheless, cloud resources are somehow overlooked. Moreover, each cloud provider possesses a different policy to define their resources as well as their management, which implies high heterogeneity. Thus, there is a need to uniformly define cloud resource descriptions in order to insure the interoperability between cloud process providers. Moreover, this integration should take into account the cloud features such as the elasticity, the shareability in order to ensure a correct allocation of resources.

To meet these needs, our first objective is to build a unified common understanding between providers and tenants. We aim at integrating the cloud resource definition into BPM while ensuring a correct use and consumption of them. We also target a free-of-conflict resource allocation during BP execution. To this end, we consider

semantic modeling in order to provide formal definitions and helped the heterogeneity of resource description.

Concretely, to address this research problem, we need to answer the following questions:

1. How to define cloud resources in business processes in a formal and unified way?
2. How to enhance the resource perspective in BPM?
3. Can semantic technologies be useful? and how?
4. How to take into account the specificities of cloud resources into BPM?
5. How to enrich business process models with social technologies?
6. How efficient our approach is?

1.2.2 On Supporting and Optimizing the resource variability in configurable process models

Motivated by the need of adopting agile, flexible and cost-effective business solutions, enterprises are looking for available business processes outside of their organizations to quickly adapt to new business requirements and also reduce process development and maintenance costs. Cloud Computing is more and more utilized to outsource service-based BPs based on a scalable pay-per-use model.

In such a multi-tenant environment, using configurable process models [17] allows a cloud business process provider to deliver a customizable process that can be configured by different tenants according to their specific needs [18]. Different approaches for configurable process modeling have been proposed so far, mainly with a focus on configuring the control flow [38]. Even though the concept of configurable process models is highly complementary to cloud computing, there has been hardly any uptake in that area. The problem is apparently that specifics of cloud computing, specifically in how resources can be configured and integrated, are hardly considered in configurable process modeling. Existing proposals on extending configuration to resources [29, 43, 44] do not cover required cloud concepts such as elasticity or multi-tenancy and focus on human resources and their dependencies [41, 45].

To address this research problem, we need to answer the following questions:

1. How to allow process users to configure their choices in terms of cloud resources?
2. How to integrate the cloud resource variability in configurable process models w.r.t the cloud properties i.e., elasticity or shareability?
3. How to select the optimal resource allocation?
4. How efficient our approach is?

1.3 Motivating example

We present in the following two business process models taken from the French Telecommunication operator Orange, which is an industrial partner, to illustrate and motivate our approach. They are also used to explain our approach in the next chapters.

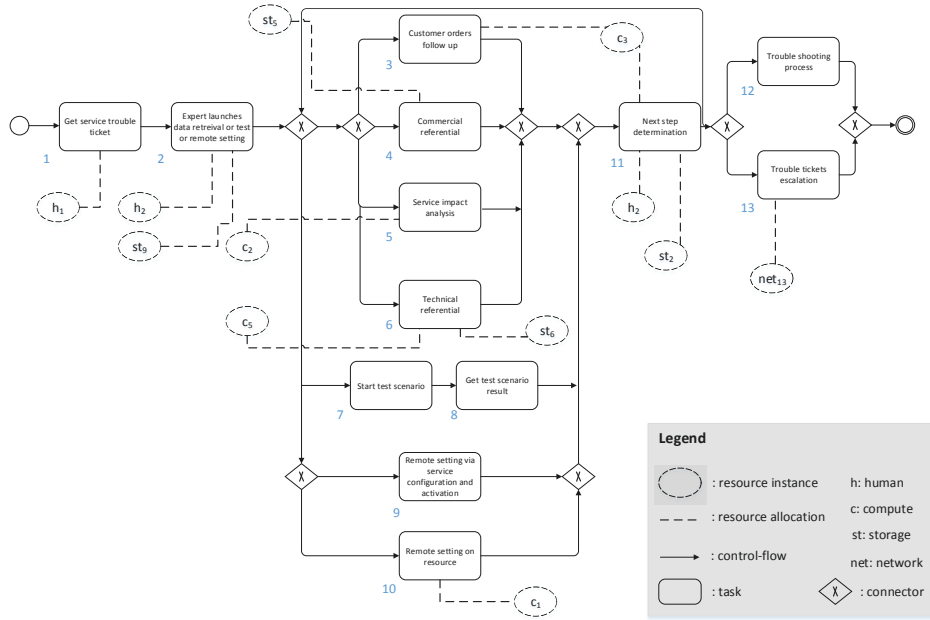


Figure 1.4: A Service Supervision Business Process

First, we introduce a service supervision business process which presents how a customer’s complaint is addressed due to service quality drop. Figure 1.4 models the supervision process in *BPMN* 2.0. Upon complaint lodging through a ticket, three sub-processes² are initiated: (i) manual data retrieval: an expert executes activities or tasks namely a_3, a_4, a_5, a_6 so that customer data are pulled from a database, (ii) manual test: the expert executes a_7, a_8 to detect anomalies related to the complaint, and finally (iii) remote setting: a_9, a_{10} are executed to set remotely the necessary parameters for complaint analysis. These sub-processes are re-executed until the problem is identified properly. Finally, either troubleshooting or ticket escalation is performed.

Please note that, although our processes are modeled with *BPMN*, our work can be easily extended to other graph-based business process modeling notations such as *EPC*.

²In compliance with *BPM* terminology, a sub-process is part of a process, consists of activities, and has a control-flow (www.appian.com/bpmbasics/bpm-glossary).

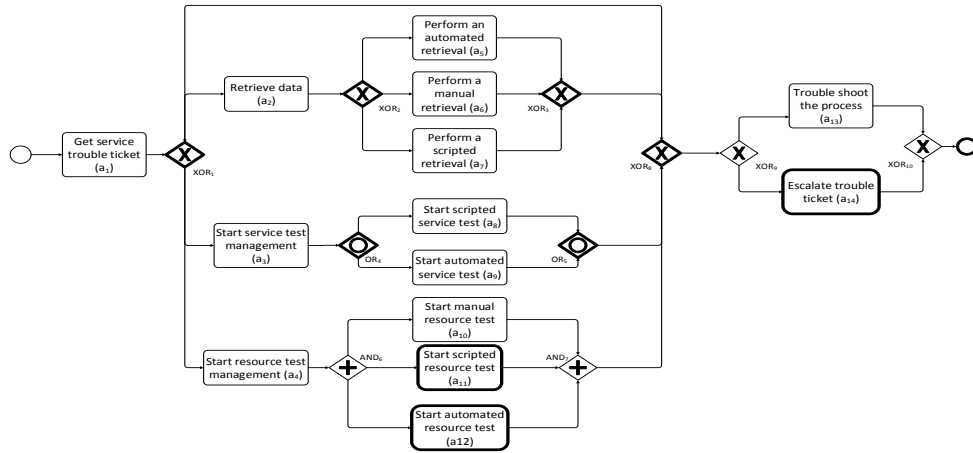


Figure 1.5: A configurable service supervision process

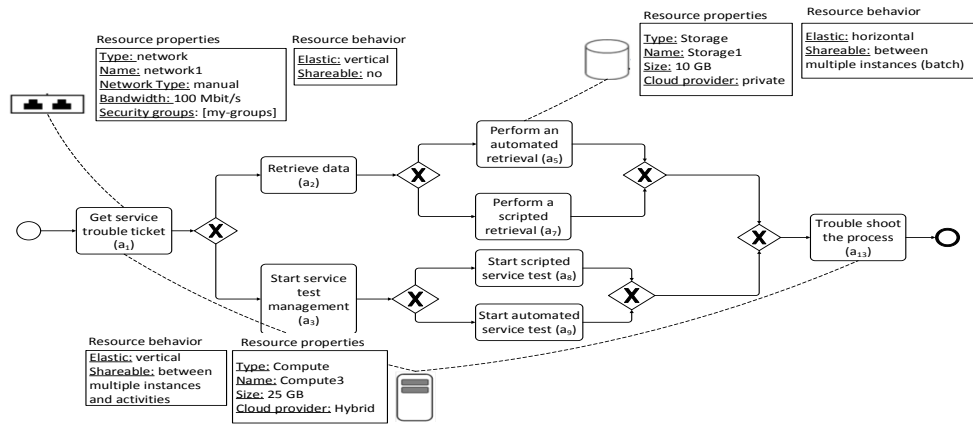


Figure 1.6: Variant 1: A process variant derived from the configurable process in Figure 1.5 and its allocated cloud resources

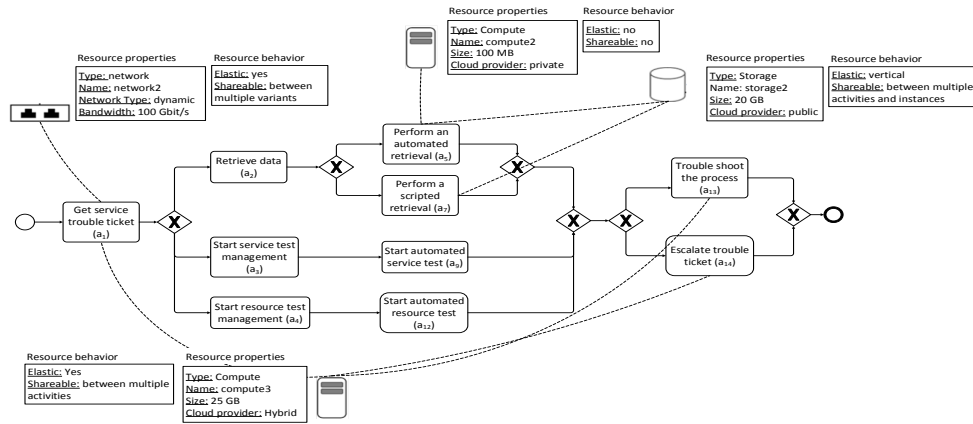


Figure 1.7: Variant 2: A process variant derived from the configurable process in Figure 1.5 and its allocated cloud resources

To correctly execute the process model, activities require different resources whether human or non-human. However process modeling does not support resource allocation and then there is a lack of formal description of cloud resources. Moreover, cloud properties such as the elasticity and shareability are not taken into account in multi-tenant BPs.

Second, we present in Figure 1.5 an extended version of the above-mentioned process in his configurable version. In order to consolidate its expertise in service supervision processes, Orange affiliates share the configurable process in Figure 1.5 in a common infrastructure. According to its specific needs, each affiliate configures the process by taking into account the countries legislation and internal regulations. For instance, suppose that an affiliate *A* does not have access to the resource test management functionalities (the subprocess starting with the activity a_4) and does not have the right to neither perform manual tasks (activity a_6) nor trouble ticket escalation (activity a_{14}). Therefore, it configures the process in Figure 1.5 to exclude these functionalities, resulting in a variant as illustrated in Figure 1.6.

Since configurable process modeling approaches do not support the resource allocation in multi-tenant cloud environments, the affiliate defines the required resources for its derived variant in an ad-hoc manner. For example, for the derived variant in Figure 1.6, the activity a_1 needs a network resource to communicate with a virtual machine via virtual networking. The network type is manual with a bandwidth of 100Mbit/s and which is accessible for a specific security group. These parameters are identified in the "Resource properties" label in Figure 1.6. Furthermore, the activity needs an elastic network resource (vertical elasticity), that for security issues should be not share with other activities or instances. These parameters are specified in the "Resource behavior" label. Suppose that another affiliates B configures the process as shown in Figure 1.7 including its required resources. Activity a_1 needs an elastic network resource (horizontally, vertically or both according to the run-time requirements). The network is dynamic with a bandwidth of 100 Gbit/s in order to support the workload from different variants' instances. We notice that the allocated resources for the remaining activities are similar to those allocated in the variant 1 in Figure 1.6 but with some variations. This example shows that multi-tenant business processes do not only share commonalities between their executed tasks, but also between their allocated resources. In fact, different tenants allocate similar resources that slightly differ according to the resource properties and behavior.

Up until now, these allocation parameters are hard-coded in an ad-hoc manner which is certainly undesirable in such a multi-tenant environment. Therefore, there is a need for a process configuration support at the cloud resource allocation level and which shifts the cloud resource allocation parameters from the tenant side (at the process variant level) to the cloud process provider side (at the configurable process level).

At this stage, the process provider needs some assistance in order to design the process with the possibility of customizing the selection of cloud resources that suit

best its needs. Therefore, we propose to (i) formally define the resource perspective, (ii) support cloud resource allocation and verify resource constraints, (iii) allow process providers to customize their configurable BPs with their needed resources, and (iiii) optimize the selection of allocated Cloud resources.

1.4 Thesis principles, objectives and contributions

1.4.1 Thesis principles

In our approach we take into account the following principles:

- **Heterogeneous data modeling:** The approach should provide a model that defines heterogeneous Cloud resources in a machine-interpretable way. Our goal is to ensure the interoperability among an organization.
- **Exploitation of knowledge:** As the approach is based on the blending of Cloud and BPM, it should consider the explicit structure and features from Cloud APIs and the dependencies that can exist between the cloud resources.
- **Balanced computation complexity:** In order to not make things complicated, the approach should make a compromise between the computational complexity and the quality of results.
- **Fine-grained results:** The approach should provide fine-grained and focused results that are not confusing process users.

Besides, our approach in this thesis needs to be (i) validated through proof of concepts and (ii) evaluated through experiments on real datasets. Then, the validation, and experiments results should be discussed.

1.4.2 Thesis objectives

In this thesis, we aim at **integrating the support of cloud resource allocation in configurable business process models**. Our objective is twofold: (i) Integrating the resource allocation in business process models by providing formal definitions of the resource perspective to ensure correct resource allocation and (ii) Allowing the variability and the optimization of cloud resource allocation in configurable process models.

To realize the first objective, we propose to benefit from semantic technologies to formalize resource descriptions in order to ensure the interoperability between process providers. This formalization aims at ensuring a correct use of resources and preventing resource-based conflicts. To realize the second objective, we propose to define a novel approach for modeling configurable BPs with configurable resource allocation operators that allow to explicitly model resource allocation alternatives

in multi-tenant process models. Then, we define algorithms in order to obtain an optimal selection of the resource configuration.

As process models are not explicitly integrating the modeling of cloud resource allocation, we believe that taking into account the attributes of cloud resources as well as the cloud properties (i.e. the elasticity and shareability), using semantic definitions, becomes essential. Therefore, we define the variability of resource allocation depending on process users requirements, using new configurable resource operators. Afterwards, we propose genetic-based approaches to optimize the selection of these resource configurations.

1.4.3 Thesis contributions

To support resource allocation in cloud-based BPs, we proceed in three steps.

Firstly, we introduce a semantic framework for a semantically-enriched resource description in BPs. With the aim of formalizing the consumed cloud resources using a shared knowledge base, this framework allows for (i) resource description in a formal and unique way, (ii) resource allocation management, and (iii) verification of resource constraints. For this end, we extend BPMN as a modeling language with the resource perspective. Besides, we adopt semantic resource modeling using ontologies by integrating resource description into BPMO.

The second step consists of building upon social BPs to provide strategies in order to ensure a controlled resource allocation without conflicts in terms of resources. We aim at responding to SLA constraints that are defined by tenants. For instance, a tenant may request to not share his resource with other tenants, or may want to use a particular elasticity policy. To do so, we define a set of social dependencies that exist between cloud resources, and different process elements involved in resource allocation.

Finally and most importantly, we propose a novel approach that extends configurable process models to permit a configurable cloud resource allocation. Our purpose is to shift the cloud resource allocation from the tenant side to the cloud process provider side for a centralized resource management. This approach allows different tenants to customize the selection of the needed resources w.r.t two important properties i.e. elasticity and shareability. Afterwards, we propose genetic-based approaches that aim at selecting optimal resource configuration in an energy efficient manner and to improve non-functional properties.

In summary, our contributions in this thesis are as follows:

1. A semantic approach towards resource-aware business process development in the Cloud:
 - A BPMN 2.0 extension to *integrate* and *enrich* the description of the resource perspective.

- A cloud resource ontology (CloudPrO) that extends BPMO to formally describe cloud resources and their dependencies in a business process.
2. A Social-based approach for cloud resource management in business processes:
 - An extension of CloudPrO to provide social-based formal description of cloud resources according to the architecture of Cloud API OCCI [46].
 - Semantic rules that enable the resolution of resource-based conflicts.
 - Building a knowledge base from heterogeneous business process models.
 3. A novel approach for configurable resource allocation for multi-tenant process development in the Cloud:
 - Extending configurable processes with new configurable cloud resource allocation operators that permits the customization of the different resource alternatives by the tenants.
 - A genetic algorithm which aims at selecting the optimal cloud resource configuration that best fits to the tenant requirements, and reducing the environmental impact.
 4. A validation approach:
 - Four proof-of-concepts implemented as extensions of Signavio process editor and ontology validation to validate each contribution.
 - Experiments on real datasets to demonstrate the feasibility and the efficiency of our proposals.

1.5 Thesis outline

This thesis is organized as follows: Chapter 2 presents some preliminaries and basic concepts used throughout the thesis. We show definitions of the different modeling languages that we use (i.e., BPMN and C-BPMN), and an abstract representation of process models using process graphs. We also present the characteristics of the Cloud Computing which represents the context of our thesis.

Chapter 3 presents the state of the art of our research context. It starts by presenting the existing approaches working on enhancing and formalizing the resource perspective in business process models. Then, we introduce the different proposed configurable process modeling approaches working on supporting and optimizing the resource variability. We present and criticize their models and their solutions. This evaluation allows us to justify the need for proposing an automated support for configurable Cloud resource Allocation in business process models.

Chapters 4, and 5 represent the core of our thesis which elaborate our approach to support the design and configuration of configurable resource allocation in process

models. In Chapter 4, we present our solution to semantically model heterogeneous Cloud resource use in business process models and support this modeling a common knowledge base. To this end, we reuse an existing ontology, namely BPMO, to integrate Cloud resource' structure.

In Chapter 5, we present our solution to assist the design of configurable Cloud resource allocation within configurable process models with a set of novel configurable resource operators. Then, we present a further approach that aims at optimally allocating Cloud resources to business process variants using genetic algorithm. We formalize our problem using a genome encoding then we apply a genetic algorithm that allows the user to identify the most optimal Cloud resource allocation.

Finally, Chapter 6 concludes this thesis by summarizing the work presented and discussing possible extensions.

Preliminaries and Basic Concepts

Contents

| | |
|--|-----------|
| 2.1 Process Modeling | 31 |
| 2.1.1 Business Process Model and Notation (BPMN) | 32 |
| 2.1.2 Configurable BPMN (C-BPMN) | 33 |
| 2.1.3 Process Graphs | 35 |
| 2.2 Semantic Business Process Modeling | 37 |
| 2.2.1 Web Ontology Language for Web Services (OWL-S) | 37 |
| 2.2.2 General Process Ontology (GPO) | 38 |
| 2.2.3 Business Process Modeling Ontology (BPOM) | 39 |
| 2.3 Cloud Computing | 41 |
| 2.3.1 Generalities | 41 |
| 2.3.2 Open Cloud Computing Interface (OCCI) | 42 |

This chapter presents the preliminaries used in the remainder of this thesis. In Section 2.1, we introduce process modeling standards and specifically the standards used to illustrate this work that are Business Process Model and Notation (BPMN) (Section 2.1.1), and Configurable BPMN (Section 2.1.2), and a brief description of process graphs (Section 2.1.3).

2.1 Process Modeling

A process model enables the representation of the behavior of a business process according to its three perspectives: (i) **control flow** which describes the temporal ordering of the process tasks (ii) **data flow** which describes the data exchanged between the tasks [47] and (iii) **resource flow** which describes the physical objects and human performers required to accomplish a task. In this thesis we mainly focus on the resource flow perspective of the processes especially on the cloud resources, and therefore the models are used to capture the resource allocation required by the process tasks.

To represent a business process, multiple graphical process modeling languages exist in the BPM community such as BPMN, EPC, YAWL, UML activity diagram, etc. Despite their differences in modeling notations, they all share the common concepts of tasks, events, gateways, artifacts and resources, as well as relations between them, such as transition flows [48]. We select and use BPMN in our approach as it is one of the most popular business process modeling language. Therefore, in Section 2.1.1, we present the main elements of BPMN. Configurable BPMN which extends BPMN with configurable elements is then discussed in Section 2.1.2. In Section 2.1.3, we present some definitions related to an abstract representation of a (configurable) process model, referred to as (configurable) process graph, to which most of the process modeling languages can be mapped. Then, we present the Business Process Modeling Ontology (BPMO) in Section 2.2.3. We discuss next, in Section 2.3, the Cloud Computing paradigm as its characteristics, and more specifically the popular OCCI API.

2.1.1 Business Process Model and Notation (BPMN)

The Business Process Model and Notation (BPMN) was first released in 2004 by the Business Process Management Initiative (BPMI) [49]. BPMN is a standard for business process modeling which permits the creation as well the documentation of process models. Known as *de facto* process modeling notation [50], it is widely used in the industry. Actually, BPMN has been enhanced with executable semantics enabling the execution of the modeled process.

In order to capture different perspectives of the business process at various detail levels, BPMN supplies a rich set of elements. These elements can be categorized into a *core set* which contains the basic elements and an *extended set* which contains specialized elements to specify more complex business scenarios [51]. In fact, BPMN defines 50 constructs decomposed into four categories: *Flow objects*, *Connecting objects*, *Swimlanes* and *Artifacts*. *Flow Objects* allow to model the control flow perspective of a business process in terms of activities, events and gateways. An activity, also called task, is the main element of a process model which describes the work that should be achieved. It is graphically represented as a rectangle (see Figure 1.4 for an example of a process model in BPMN notation). An event is something that happens during the business process' execution. Three types of events that exist: Start, Intermediate and End events which may be specialized to *Message*, *Error*, etc. An event is represented as a circle. A gateway allows to model the splits and joins in the process model. Three main types are used to represent the different behaviors in a business process: *AND* (parallel synchronization), *XOR* (exclusive choice and merging) and *OR* (inclusive choice and merging). Besides, other specialized gateways exist in BPMN (e.g., event-based gateway, complex gateways), yet they can all be mapped to one of the three main types *OR*, *AND* or *XOR*. The *flow objects* elements are connected via the *Sequence flow* element in *Connecting objects* category. They

allow to specify the order in which the activities will be executed in a process.

The *Artifacts*, *Swimlanes* and other elements in *Connecting objects* allow to model the resource and data perspectives in the process. For instance, the *Pools* and *Lanes* elements in *Swimlanes* allow to group a set of activities that are executed by a specific role. The *Data object* element in *Artifacts* provides information about the data required by an activity. An *Association* element in *Connecting objects* is used to associate *Data objects* with a flow or connect them to activities.

2.1.2 Configurable BPMN (C-BPMN)

A configurable process model is a process model which contains configurable elements. A configurable element allows process analysts to make a *design-time choice* in addition to traditional *run-time choices* [52, 53]. This configurable element is graphically modeled with a thick line. For example, in the configurable process model in Figure 1.5, the gateway XOR_1 is configurable while XOR_9 is not. The difference between these two elements is that XOR_9 depicts a simple *run-time choice*, i.e. the choice to execute either a_{13} or a_{14} is based on the run-time execution data. While XOR_1 has a *design-time choice* in addition to the run-time choice. The design-time choice, called also *configuration choice*, allows to choose one design option from multiple ones. Returning to our example, a choice can be taken to keep or remove one of the outgoing branches of XOR_1 (i.e. XOR_2 or OR_4 or AND_6) from the model. If only one of them is selected, XOR_1 is transformed to a simple sequence whose outgoing branch is executed at run-time, otherwise XOR_1 is transformed to a normal XOR whose decision is made at run-time.

We present the notation of a configurable BPMN (C-BPMN) in which the control flow elements (i.e. activities and gateways) can be configurable. Configurable activities and gateways have been discussed in [37, 52].

A configurable activity can be included (i.e. configured to *ON*), excluded (i.e. configured to *OFF*) or optionally excluded (i.e. configured to *OPT*) from the process model (Figure 2.1). The latter allows to make a run-time decision to execute or to skip the activity. However, we do not take into consideration the latter configuration in this thesis.

A configurable gateway is characterized by a generic behavior which is restricted by configuration. A gateway can be configured according two different ways (1) changing its type while preserving its behavior and/or (2) restricting its incoming (respectively outgoing) branches in case of a join (respectively split). Table 2.1 depicts the configuration constraints of the various types of gateways [52]. A configurable gateway is denoted by $[type]^c$. Each row in the table corresponds to a configurable gateway which can be configured to one or more of the gateways presented in columns. The last column (i.e. *Seq*) corresponds to a *Sequence flow*. For example, the configurable OR (OR^c) can be configured to any gateway's type while a configurable AND (AND^c) can be only configured to an (AND). These configuration constraints are formalized

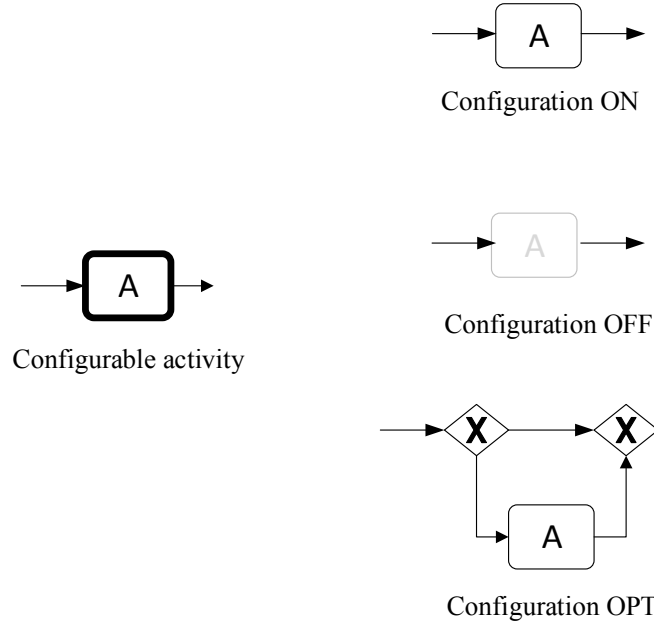


Figure 2.1: A configurable activity and its possible configuration choices

| | <i>OR</i> | <i>AND</i> | <i>XOR</i> | <i>Seq</i> |
|---------|-----------|------------|------------|------------|
| OR^c | ✓ | ✓ | ✓ | ✓ |
| AND^c | | ✓ | | |
| XOR^c | | | ✓ | ✓ |

Table 2.1: Configuration constraints of configurable gateways

through the partial order \preceq_g that specifies which concrete gateway may be used for a given configurable gateway [52].

Definition 2.1.1 (Partial order \preceq_g). *Let g^c be a configurable gateway and g be a normal gateway or a sequence flow (i.e. “Seq”). $g \preceq_g g^c$ iff $(g^c = OR^c) \vee (g^c = XOR^c \wedge g = Seq) \vee (g^c = g)$.*

In our motivating example, in the configurable process model in Figure 1.5, the configurable gateway XOR_1^c can be configured to an *XOR* with two or three outgoing branches or a *Seq* with the restriction of one of its outgoing branches.

A configurable event can be included (i.e. *enable*), excluded (i.e. *disable*) or change its type to one of the BPMN events’ types (e.g. *Message*, *Error*, etc.) (Figure 2.2). The latter configuration is allowed when the corresponding configurable event has an *abstract* type, i.e. a configurable event without a type. We use the *None* event (i.e event without label) from BPMN to denote a configurable abstract event. Nevertheless, configurable events are not taken into account in this thesis for

simplicity's sake.

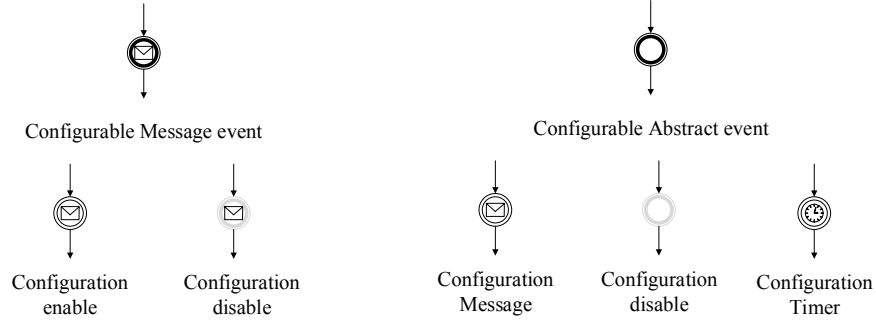


Figure 2.2: Configurable events and their possible configuration choices

2.1.3 Process Graphs

To represent business process models, graph theory is the most used technique. Hence, we choose to represent a process model as a directed graph denoted as *process graph*. This graph allows to capture the nodes and edges as attributes. This technique is derived from the generally known constructs of graphical process modeling notations. Thus, it can be generalized afterwards for most of them (e.g. BPMN, EPC) [54].

Definition 2.1.2 (Process graph). *A process graph $P = (id, N, E, T, L, I)$ is a directed graph where:*

- *id is its unique identifier;*
- *N is the set of nodes. In BPMN, N is the set of activities, events and gateways;*
- *$E \subseteq N \times N$ is the set of edges connecting two nodes. We denote by $source_e$ and $target_e$ the source and target nodes of an edge $e \in E$;*
- *$T : N \rightarrow \mathcal{T}$ where \mathcal{T} is the set of elements' types of the modeling languages metamodel and T is a function that assigns for each node $n \in N$ a type $t \in \mathcal{T}$. In BPMN, $\mathcal{T} = \{activity, Start\ event, End\ event, Intermediate\ event, gateway\}$;*
- *$L : N \rightarrow \mathcal{L}$ where \mathcal{L} is the universe of elements' labels and L is a function that assigns for each node $n \in N$ a label $l \in \mathcal{L}$. In BPMN, if $T(n) \in \{event, activity\}$, then $L(n)$ is its name, and if $T(n) = gateway$ then $L(n) \in \{OR, XOR, AND\}$.*
- *$I : N \rightarrow \mathbb{N}$ is a function that assigns for each node $n \in N$ a unique identifier $id \in \mathbb{N}$.*

Figure 2.3 represents an example of a process graph of a process model. The nodes are connected to a text annotation including informations (i.e., types, labels and identifiers).

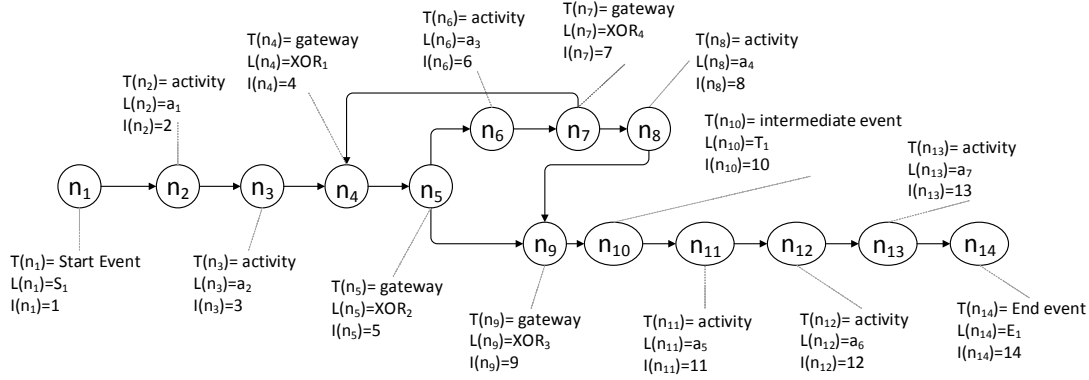


Figure 2.3: An example of a process graph

Let $P = (id, N, E, T, L, I)$ be a process graph.

The preset and postset of $n \in N$ is the set of elements in its incoming and outgoing branches respectively (Definition 2.1.3).

Definition 2.1.3 (preset \bullet, n , postset $n \bullet$). *The preset of an element $n \in N$ denoted as $\bullet n$ is defined as $\bullet n = \{n_x \in N : (n_x, n) \in E\}$. The postset of n denoted as $n \bullet$ is defined as $n \bullet = \{n_x \in N : (n, n_x) \in E\}$.*

A gateway g is a **split** if $|g \bullet| > 1$; it is a **join** if $|\bullet g| > 1$.

A configurable process graph is a process graph in which the nodes can be either configurable or not. A configurable node has a set of configuration choices. In case of BPMN, the configurable nodes can be activities, events and gateways and their configuration choices are as presented in Section 2.1.2. Formally, the configuration of a node is given in Definition 2.1.4.

Definition 2.1.4 (Configuration $Conf$). *A configuration of a configurable node n^c denoted as $Conf_{n^c}$ is defined as following:*

- if $T(n^c) = \text{activity}$ then $Conf_{n^c} \in \{ON, OFF\}$;
- if $T(n^c) = \text{event}$ then $Conf_{n^c} \in \{\text{enable}, \text{disable}, e\}$ such that $e \preceq_e n^c$.
- if $T(n^c) = \text{gateway}$ then $Conf_{n^c} \in \{(c', s) : (c', s) \in CT \times \mathcal{P}(S)\}$ where:
 - $CT = \{OR, AND, XOR, Seq\}$ and $c' \preceq_g n^c$,
 - $S = \bullet n^c$ (respectively $S = n^c \bullet$) in case n^c is a join (respectively split) gateway.

We denote by \mathbb{C}_{n^c} the set of all configurations of the configurable element n^c . For example, in Figure 1.5, the set of configurations of the configurable gateway XOR_1 is $\mathbb{C}_{XOR_1^c} = \{(XOR, \{XOR_2^c, XOR_3\}), (Seq, \{XOR_2\}), (Seq, \{XOR_3\})\}$.

Definition 2.1.5 (Configurable process graph). *A configurable process graph is denoted as $P^c = (id, N, E, T, L, I, B, \mathbb{C}^*)$ where:*

- id, N, E, T, L, I are as specified in Definition 2.1.2;
- $B : N \rightarrow \{true, false\}$ is a boolean function returning true for configurable nodes;
- $\mathbb{C}^* = \{\mathbb{C}_n : n \in N \wedge B(n) = true\}$ is the set of configurations of all configurable nodes.

2.2 Semantic Business Process Modeling

A main issue exists in business process modeling is the semantic interoperability, that is evident to solve especially in collaborative organizations' PAIS [55]. These organizations employ heterogeneous processes according to its different branches. We present in the following brief descriptions of the main ontologies that are developed in this area: Web Ontology Language for Web Services (Section 2.2.1), General Process Ontology (Section 2.2.2) and Business Process Modeling Ontology (Section 2.2.3).

2.2.1 Web Ontology Language for Web Services (OWL-S)

The Web Ontology Language for Web Services, denoted as OWL-S, represents an ontology that builds on the Web Ontology Language (OWL) following the layered approach to markup language development. It is produced by the Web-Ontology Working Group at the World Wide Web Consortium¹ to provide semantic definitions for Web Services [1]. We use this language to define our proper ontologies in Chapter 4 in sections 4.2.4.1 and 4.3.2.1. The purpose of OWL-S is to allow the users to automatically discover, perform, as well as monitor web services. This language offers the definition of a process ontology that is included in the OWL-S ontology aiming at defining the way the interaction with a service is and represent this interaction as a process. It is based on the basic elements of Resource Description Framework (RDF)² language which is developed by the World-Wide Web Consortium (W3C) in order to provide a structure for describing identified information in the Web.

Figure 2.4 depicts the process ontology modeled in OWL language (i.e., classes, properties and axioms). Three main process types are specified: (i) atomic processes which depict the actions that a service may invoke within a single interaction, (ii) composite processes which represent the actions that multistep protocols and/or multiple actions require, and (iii) simple processes which supply abstract various process' views.

¹<https://www.w3.org/Submission/OWL-S/>

²Latest version: <https://www.w3.org/TR/rdf-schema/>

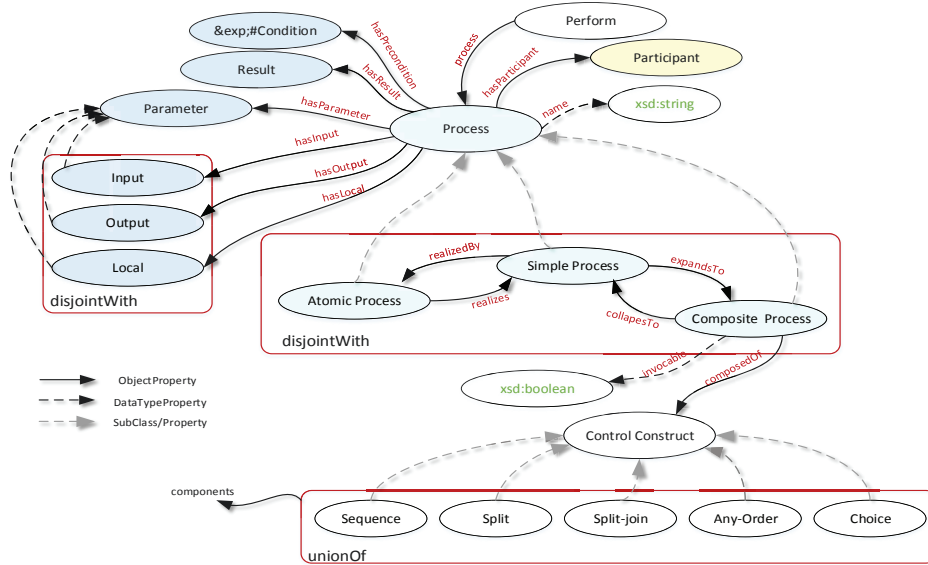


Figure 2.4: The Process ontology in OWL-S [1]

2.2.2 General Process Ontology (GPO)

The General Process Ontology, denoted as GPO, represents a formal language of concepts for modeling business processes [2]. It is known as the semantic mediator among various process modeling languages. Besides, it is defined in OWL to make use of semantic web technology in order to create a computer-interpretable semantic markup language for process modeling. Briefly, this ontology provides a unified schema for semantically annotating business process models. The aim of GPO is to assist human and machines to understand heterogeneous business process models using semantic definitions. Figure 2.5 depicts the main concepts of this ontology. Some of these concepts are described as follows:

- Activity for representing the decomposition of a process works.
- Artifact for representing something involved in an activity such as information, software, etc.
- WorkflowPattern for representing the temporal ordering of the business activities of a process. WorkflowPattern can be categorized into several patterns based on ordinary workflow patterns [56], such as Sequence, Choice (ExclusiveChoice,

MultipleChoice, ParallelSplit), and Merge (SimpleMerge, MultipleMerge, Synchronization), that depict workflow control patterns supported by most process modeling languages.

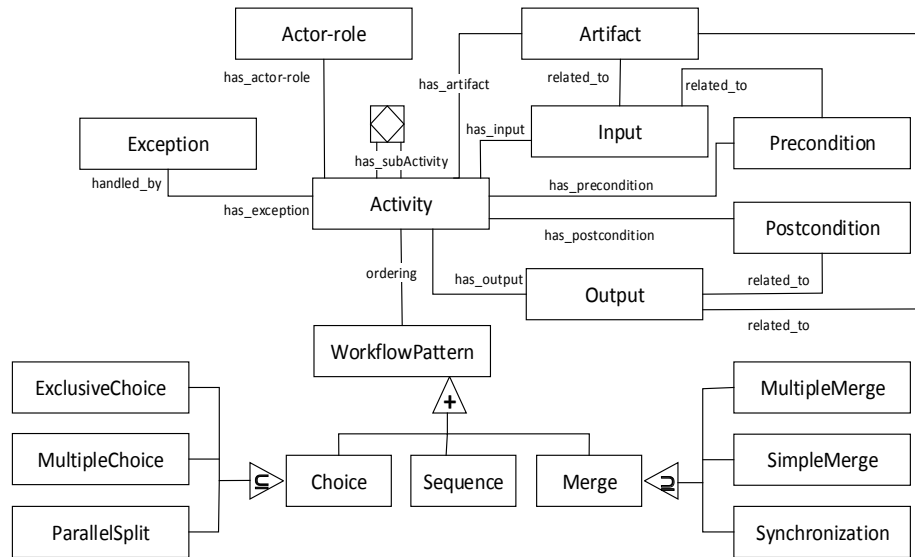


Figure 2.5: General process ontology (GPO) [2] represented using RML [3]

2.2.3 Business Process Modeling Ontology (BPMO)

The Business Process Modeling Ontology, referred to as BPMO [4] represents one of the major contributions of the European project SUPER [57]. It allows to integrate knowledge about the organizational context, workflow activities and Semantic Web Services. It provides support from different process modeling languages, such as BPMN and EPC [58], that present specific notations. Concretely, BPMO enables to represent workflow and organizational concerns in a uniform and abstract way, using ontological descriptions. Figure 2.6 represents BPMO concepts and relationships. We present herein the main BPMO concepts:

- bpmo:Process represents an abstraction of a business process model.
- bpmo:WorkflowElement represents a model element in a business process model. It involves three sub-concepts: bpmo:Task, bpmo:WorkflowEvent and bpmo:GraphPattern. It is linked to a process model via the property bpmo:hasHomeProcess.
- bpmo:Task represents an atomic unit of work in a business process model.

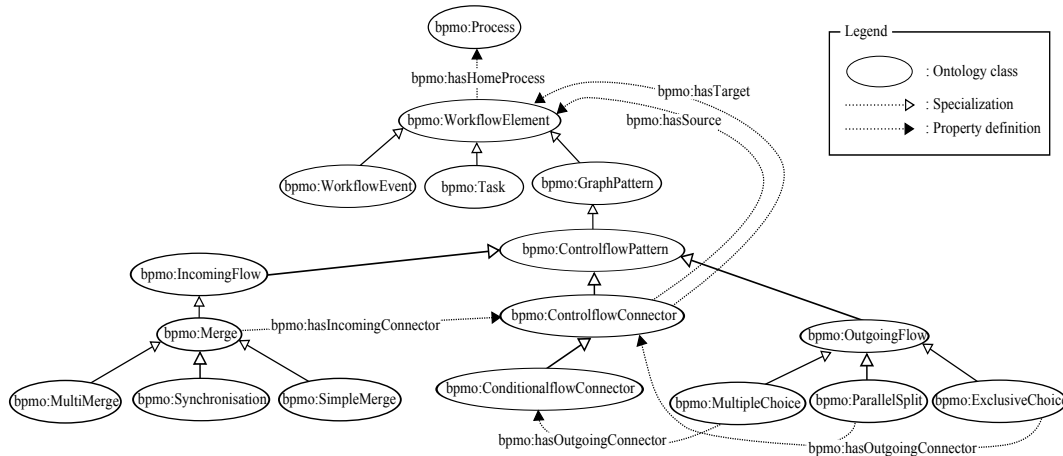


Figure 2.6: The Business Process Modeling Ontology (BPMO) [4]

- bpmo:WorkflowEvent represents an event that happens during the business process' execution. It has three sub-concepts bpmo:StartEvent, bpmo:IntermediateEvent and bpmo:EndEvent.
- bpmo:GraphPattern represents the connection between workflow elements in a business process model. It has three sub-concepts:
 - bpmo:ControlflowConnector is the control-flow element (i.e., sequence) which relates two model elements through the properties bpmo:hasSource and bpmo:hasTarget.
 - bpmo:IncomingFlow depicts the different logical connectors with a merging or join behavior. The bpmo:hasIncomingConnector property links a merging connector to the workflow elements in its incoming flow. A bpmo:Merge connector possesses three main sub-concepts: bpmo:MultipleMerge, bpmo:SimpleMerge and bpmo:Synchronization.
 - bpmo:OutgoingFlow depicts the different logical connectors with a split behavior. The bpmo:hasOutgoingConnector property links a split connector to the workflow elements in its outgoing flow. bpmo:OutgoingFlow connector has three sub-concepts: bpmo:ExclusiveChoice, bpmo:ParallelSplit and bpmo:Multiplechoice.

Two main ontologies sBPMN [126] and sEPC [53] have been also developed in order to formalize the process languages BPMN and EPC, respectively. Processes can be translated, through using these ontologies, from BPMN and EPC to BPMO and vice versa [127]. As a result, using such ontologies is appropriate for modeling heterogeneous business processes with their resources.

2.3 Cloud Computing

2.3.1 Generalities

Cloud Computing is an emerging paradigm in Information Technology which has witnessed a major growth in a short period of time. It is defined by the National Institute of Standards and Technology [16] as a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort. Figure 2.7 depicts an overview of Cloud Computing Conceptual reference model defined by NIST [59]. It includes the main actors (cloud consumer, cloud provider, cloud carrier, cloud auditor and cloud broker), their activities and functions in the Cloud [60].

The Cloud offers different types of computing resources to be consumed as needed. Furthermore, Cloud resources are accessible over the network through standard mechanisms that promote their use by different platforms. The resources are offered to consumers using a multi-tenant model and are provisioned in an elastic manner that allows to scale up or down in line with demand. Therefore the two major properties of cloud resources are: elasticity and shareability.

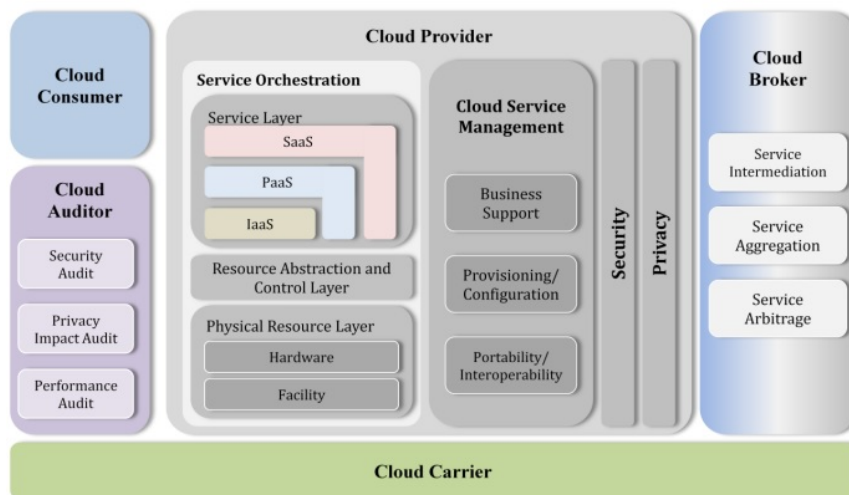


Figure 2.7: Conceptual Reference Model of Cloud Computing (defined by NIST)

Basically the Cloud delivers services under three layers namely the Infrastructure as a Service (IaaS), the Platform as a Service (PaaS) and the Software as a Service (SaaS):

- **IaaS:** Consumers are able to access Cloud resources on demand. These resources can be spread into virtual machines (VMs). They consist of compute, network,

and storage resources. The provider is responsible of installing, transparently managing and maintaining these resources.

- **PaaS:** Consumers are allowed to develop, deploy and manage their applications onto the Cloud using the libraries, editors and services offered by the provider. The provider is responsible of provisioning, managing and maintaining the Infrastructure resources.
- **SaaS:** Consumers are able to use running applications on an IaaS or a PaaS through an interface. They are not responsible of managing or maintaining the used Cloud resources.

Clouds can be provisioned following different models according to the users requirements. We denote by Private Cloud the case where the Cloud is used by a single organization. This organization owns the Cloud and is responsible of its management and maintenance. Otherwise, if the Cloud is owned by various organizations, we are talking about community or federation Cloud. Whenever the Cloud is exposed to public use, we are talking about Public Cloud. In this case, an organization owns the Cloud and manages it while it is used by other organizations. Another case where the cloud is composed of two or more clouds. We talk herein about a combination between Public and Private clouds.

2.3.2 Open Cloud Computing Interface (OCCI)

OCCI working group provides a high-level definition of a RESTful Protocol and API, within the Open Grid Forum (OGF) [61]. OCCI is defined as “an abstraction of real world resources, including the means to identify, classify, associate and extend these resources”. At the beginning, it was initiated to create a remote management API for IaaS model-based services. Besides, it allows the development of interoperable tools for common tasks including deployment, autonomic scaling and monitoring [?]. Then, it progresses towards a flexible API that strongly focus on interoperability while still offering a high degree of extensibility. The current release of the OCCI is appropriate to serve many other models in addition to IaaS, including PaaS and SaaS.

The current specification of OCCI consists of three documents:

- **OCCI core:** It describes the formal definition of the OCCI Core Model as seen by OGF [61].
- **OCCI HTTP Rendering:** It defines how to interact with the OCCI Core Model using the RESTful OCCI API [62].
- **OCCI Infrastructure:** It contains the definition of the OCCI Infrastructure extension for the IaaS domain [63].

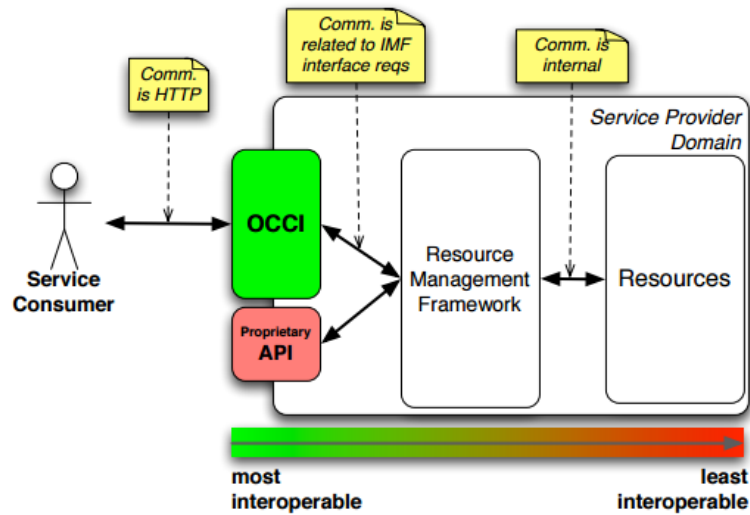


Figure 2.8: OCCI place in provider architecture (defined by OGF)

Figure 2.8 shows the position of OCCI in a provider’s architecture. A service consumer can be either an end-user or other system instance. OCCI can be used as a management API for all kinds of resources while maintaining a high level of interoperability.

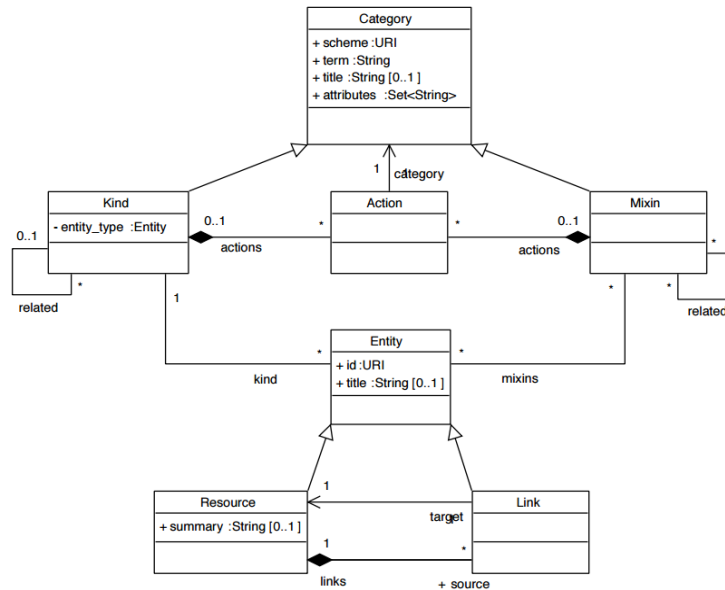


Figure 2.9: Class diagram of OCCI core model

OCCI Core formally describes the OCCI Core Model as shown in Figure 2.9. This Model defines a representation of instance types which can be manipulated through an OCCI rendering implementation. It is an abstraction of real-world resources, including the means to identify, classify, associate and extend those resources. This core model describes Cloud resources as instances of *Resource* or a sub-type thereof. Resources could be combined and linked to each other using instances of *Link* or a sub-type of this latter. *Resource* and *Link* are sub-types of *Entity*. As shown in Figure 2.9 [61], each *Entity* instance is typed using an instance of the class *Kind* and could be extended using one or more instances of the class *Mixin*. *Kind* and *Mixin* are subtypes of *Category* and each *Category* instance can have one or more instances of the class *Action*. A fundamental advantage of the OCCI Core Model is its extensibility. It is noteworthy that any extension will be discoverable and visible to an OCCI client at run-time. An OCCI client can connect to an OCCI implementation using an extended OCCI Core Model, without knowing anything in advance, and still be able to discover and understand, at run-time, all the instance types supported by that implementation. Extending OCCI core is possible using sub-types of the existing types or using *Mixins*.

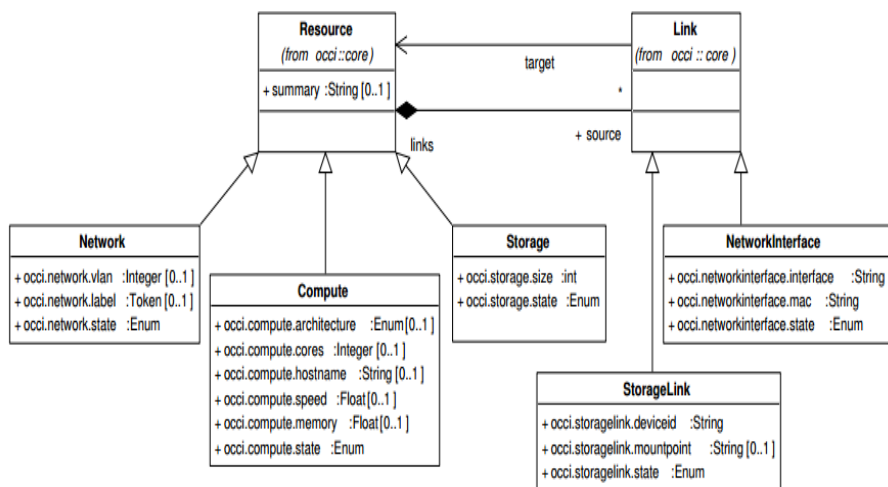


Figure 2.10: Overview Diagram of OCCI Infrastructure types

The second document is OCCI HTTP Rendering [62]. It specifies how the OCCI Core Model [61], including extensions thereof, is rendered over the HTTP protocol. The document describes the general behavior for all interaction with an OCCI implementation over HTTP together with three content types to represent the data being transferred. The OCCI model can be extended to cover different levels in the cloud (i.e., IaaS, PaaS, SaaS, etc.). And that is the purpose of the third document that is

OCCI Infrastructure [63]. It is an extension of the Core Model to represent the cloud infrastructure layer. This extension brings basically the definition of new resources that inherit the core basic types Resource and Link. As shown in Figure 2.10 [63], the Resource is specialized into: (i) *Compute*, representing a generic processing resource, (ii) *Network*, representing a networking entity (e.g. a virtual switch), and (iii) *Storage*, representing a resource that record information to a data storage device. *Link* is also specialized in OCCI infrastructure to: (i) *NetworkInterface*, representing an interaction between *Compute* and *Network* (e.g. network adapter), and (ii) *StorageLink*, representing an interaction between *Resource* to a *Storage*.

State of the Art

Contents

| | | |
|------------|---|-----------|
| 3.1 | Introduction | 47 |
| 3.2 | On Formalizing the resource perspective in Business processes | 48 |
| 3.2.1 | Resource perspective in business processes | 48 |
| 3.2.2 | Resource orchestration in Cloud-based business processes | 49 |
| 3.2.3 | Semantics in business process models | 50 |
| 3.2.4 | Social business processes | 51 |
| 3.2.5 | Synthesis | 52 |
| 3.3 | On supporting and Optimizing the resource variability in configurable process models | 53 |
| 3.3.1 | Configurable process modeling | 54 |
| 3.3.2 | Resource Variability | 55 |
| 3.3.3 | Resource Allocation Optimization in business processes | 57 |
| 3.3.4 | QoS-aware business processes | 58 |
| 3.3.4.1 | QoS properties | 58 |
| 3.3.4.2 | Ecological properties | 59 |
| 3.3.5 | Synthesis | 61 |
| 3.4 | Conclusion | 62 |

3.1 Introduction

In this chapter, we review the researches that exist in the literature that are relevant to the topic of supporting cloud resource allocation in configurable business process models. In section 3.2 we study existing solutions for formalizing the resource perspective in business process models. We classify them into four steps: (1) Resource perspective in business processes, (2) Resource orchestration in Cloud-based business processes, (3) Semantics in business process models, and (4) Social business processes. Next, in section 3.3 we discuss existing works on supporting and optimizing the resource variability in configurable process models. These proposals can be within five categories (1) Configurable process modeling, (2) Resource Variability, (3) Resource

Allocation Optimization in business processes, and (4) QoS-aware business processes. We present shortcomings of the related approaches, identify the difference and bring out the advantages of our approach.

3.2 On Formalizing the resource perspective in Business processes

Many perspectives exist for the definition of a business process such as the control-flow perspective (i.e., the temporal ordering of the process activities), the data perspective (Business documents and other objects), or the resource perspective (i.e., information about resources and device roles responsible for executing activities) [64]. Several researches in BPM R&D have been widely proposed at the level of the former perspectives, essentially the control-flow one [65–68]. However, the resource perspective is not well defined in particular when business processes are implemented in Cloud environments. These resources are often heterogeneous which prevents an easy and dynamic interoperability between different Cloud providers and users.

In this section, we review existing approaches for formalizing the resource perspective in business processes and categorize them into four categories: (i) resource perspective in business processes (Section 3.2.1), (ii) resource orchestration in Cloud-based business processes (Section 3.2.2), (iii) semantic web in business process models (Section 3.2.3) and (iv) Social business processes (Section 3.2.4). In Section 3.2.5, the presented approaches are evaluated against a set of assessment criteria that are important to our thesis context.

3.2.1 Resource perspective in business processes

In the literature, there exists previous works on the representation of the resource perspective in BPM. For example, in [24], Luis Jesús Ramón Stroppi et al. have addressed the modeling and visualization of resource perspective requirements by extending the BPMN 2.0 metamodel which was validated against workflow resource patterns [69]. The same authors propose thereafter [23, 70] an approach that enables the implementation of the requirements of resource perspective defined in extended BPMN process models and resource structure models into BPEL process definitions. Furthermore, they aim at providing a support to the lifecycle (i.e., definition, implementation, verification and validation) of resource perspective requirements in the development of PAISs based on WfMSs. [22]. These works support the three aspects of resource view: resource structure, work distribution, and authorization. The resource structure involves two aspects: the characterization and classification of resources. The characterization is the definition of the resource related information. Once resources' information are characterized, a classification of the resources can be established based on a set of common properties. The work distribution is concerned with the work distribution and its binding to specific resources for execution.

In other words, it defines the manner in which the work of a process is distributed and allocated to resources. The authorization deals with the definition of privileges owned by resources regarding the execution of operations in order to organize the work advertised to them.

In [71], Nick Russell and Wil M. P. van der Aalst use the workflow resource patterns [69] as an assessment framework to evaluate the capabilities of the BPEL extensions (*BPEL4People* and *WS-HumanTask*). Their objective is also to identify their strengths as well as the different areas where opportunities are offered for further improvement. Anastasiia Pika et al. propose an automated framework that allows to extract knowledge about the behavior of human resources, using event logs [72]. Moreover, this framework enables the analysis of the dynamics of human behavior over time.

A general framework for document-driven workflow systems was proposed in [73]. It allows to discover data dependencies between tasks in a process, assist workflow designers, and achieve efficient control-flow design. Several works have basically focused on human resources behavior and allocation [41, 42, 74–76]. For instance, Cristina Cabanillas et al. introduce in [42] a complete graphical notation for assigning human resources within business process models. Whereas in [74] Stefan Schönig et al. present an integrated process mining framework that aims at efficiently discover resource assignment patterns and teamwork patterns. Also, they adopt a declarative process mining technique to take into consideration the resource perspective.

We notice, from the presented proposals, that the main focus is on human and data resource allocation as well as their behavior. However, they do not neither enable cloud resources' representation nor consider verification and security properties. Besides, an explicit support of resource variability is missing. In contrast, our work integrates cloud aspects and seeks for checking resource allocation properties.

3.2.2 Resource orchestration in Cloud-based business processes

The general benefits of the blending of Cloud and BPM have been stressed by different authors [77–79]. Thus, organizations have lately been investigated in the deployment of service-based business processes in Cloud environments.

Different works on Cloud resource orchestration in such context have been proposed [19, 28, 80–88]. Rajiv Ranjan et al. [39] present an overview of the main Cloud resource types as well as operations of resource orchestration. Also, they seek for identify research issues that involve these resource operations. In [85], Stefan Schulte et al. establish a platform that enables Business Process Management Systems (BPMS) to manage resource elasticity. The platform is called Vienna Platform for Elastic Processes (ViePEP). Philipp Hoenisch et al. in [28, 84] propose respectively a resource-efficient scheduling algorithm for cloud-based computational resources in business processes, and an elastic scheduling approach and an optimization model to schedule service invocations among hybrid cloud resources in a cost-efficient way. In [80], the

same authors enhance their work by defining a system model for elastic process landscapes with taking into account complex process patterns, penalty cost, and Billing Time Unit (BTU) (i.e., the cost per leasing period). Amziani et al. discuss in [89] the modeling of single elastic processes using Petri nets and simulate elasticity strategies.

Luise Pufahl et al. deal with batch processing which is a recent technique to synchronize the execution of certain activities of different process instances, for enhancing process performance. They define in [86] flexible adjustments to enable improved batch activities execution at run-time level. Thereafter, they extend BPMS in [90] by specifying batch processing requirements in order to synchronize process instances of different process models. They allow the users to decide about batch assignment i.e., whether a process instance is executed as batch or not.

However, such approaches, addressing Cloud resource allocation, lack considering semantic descriptions which is not helping in addressing the heterogeneity issue. Furthermore, since we are in a Cloud setting, configurable process models are recommended to be taken into consideration to avoid the ad-hoc process customizations [18]. Hence their use enables significant cost reductions. In our work, we aim at developing a unified common understanding for Cloud resource Allocation based on semantic definitions. Then, we target to more extend the variability in configurable process models with the possibility of configuring the Cloud resources' selection.

3.2.3 Semantics in business process models

Since different benefits that exist in using semantics in business process modeling, multiple works have revealed these advantages [91–93]. Thus, Semantic Web technology and ontology engineering researches have been extensively investigated in business processes [27, 31, 66, 94–101].

Oliver Thomas and Michael Fellmann propose extensions of different process modeling languages: EPC, BPMN and OWL with formal semantic annotations [102, 103]. They propose a framework which is based on the assignments of formalized in ontologies to process model elements. In [104], Markovic presents a framework for querying and reasoning business process models. To this end, the framework takes into account different perspectives (i.e., functional, behavioral, organizational and informational perspectives) by representing the process elements as ontology instances. Marin Dimitrov et al. use the BPMO ontology (Section 2.2.3) to define semantic annotations of both BPMN and EPC process models, under the SUPER project [26]. The aim herein is to support automated composition, mediation and execution. The SUPER project allows, indeed, to model business processes on an abstract level.

In [99, 105, 106], Chiara Di Francescomarino et al. define a semantic model that combines static, procedural and data knowledge, which enables semantic reasoning. This model allows business analysts to use the proposed semantic-based techniques in order to infer knowledge and analyze system executions.

Carlos Pedrinaci et al. provide a survey describing the particularity of Semantic

BPM, developed within the SUPER project [31]. Besides, they illustrate how this approach contributes in improving existing BPM solutions aiming at realizing more flexible, dynamic and manageable business processes. Meanwhile, Liliana Cabral et al. present a survey of the state of the art concerning works on enabling technologies for Semantic Web Services [27, 107]. They highlight also the orthogonal dimensions of the infrastructure of Semantic Web Services: activities, architecture and service ontology.

Concerning the management of Cloud resources, many works have been developed [108–113]. Lamia Youseff et al. propose an ontology for Cloud Computing that shows a classification of the Cloud into five main layers: applications, software environments, software infrastructure, software kernel and hardware [108]. They also highlight the strengths and the limitations of these layers, as well as their reliance compared to prior computing concepts. Yong Beom Ma et al. [109] proposed an ontology-based resource management system. They aim at correctly allocating requested jobs to Cloud resources that fit the requirements of cloud users. Amir Vahid Dastjerdi et al. introduce an effective architecture that allows to provide QoS-aware deployment of appliances on Cloud service providers, using ontology-based discovery technique [110]. Contrarily to our focus, these works are more oriented Cloud than business processes.

Even so, we observe a clear missing of the support of Cloud resource definitions in semantic business process management systems. Thus, the interoperability among Cloud process providers remains not treated. In our work, we adopt the use of ontologies, in Chapter 4, to establish a semantic formalization for supporting the allocation of Cloud resources within the different business activities in process models.

3.2.4 Social business processes

The blend of BPM and social computing is still at its “infancy” stage so a lot of R&D opportunities are still untapped. Yet, several researches have basically studied the impact of social human interactions on business process organizations [114, 115]. Informal social interactions are specified on which network-based business processes are developed, and a user-driven framework for monitoring business processes is presented by respectively E. Kajan et al. in [45] and Z. Maamar et al. [116]. Moreover, [117] highlight human resources as social compute unit to resolve incidents in IT service organization. Elnaffar et al. in [118] discuss challenges associated with social computing with cloud computing. However, cloud resource behavior has not been considered.

Further, in [119] Vanilson Arruda Burégio et al. suggest a guiding framework to integrate social technologies into business operations. Contrarily, we aim, at formalizing social interactions in a common way using semantic definitions. V. der Aalst and M. Song focus on mining social networks to extract and analyze interpersonal relations in an organization [120]. In [121] Maamar et al. develop social relations between BP components (i.e., tasks, persons, and machines). Same authors propose,

in [122], a network-based coordination framework to resolve conflicts in business processes. However several other conflicts are not yet treated. In our work, we handle further conflicts according to SLA requirements.

The proposed approaches basically focus on relations or dependencies that exist between human resources. Nonetheless, they do not consider non-human resources. In particular, there is clear lack for handling Cloud resources as well as their characteristics and dependencies. We choose to use the social perspective of business processes, in Chapter 4 in Section 4.3, to build social dependencies among consumed Cloud resources in order to allow the tenants to be aware of such relations and resolve resource-based conflicts.

3.2.5 Synthesis

Table 3.1 depicts a comparative table for the evaluation of the most interesting researches according to our context. This comparison is realized depending on assessment criteria defined in the first row of this table. The choice of these criteria is argued by the fact that we are interested in a set of properties on which we check if the concerned approach take into account or not or partially consider. In our context, we are interested in: (i) Resource perspective, (ii) Human resources, (iii) Cloud resource Definitions, (iv) Social techniques, and (iv) Semantic stack. The sign ‘+’ denotes that the corresponding property is fulfilled by the corresponding approach, while the sign ‘-’ indicates that the corresponding property is not fulfilled and the sign ‘+/-’ shows that the corresponding property is partially fulfilled.

Table 3.1: Comparative table for the evaluation of previous approaches

| Approaches \ Criteria | Resource perspective | Human resources | Cloud resources | Social techniques | Semantic stack |
|--------------------------|----------------------|-----------------|-----------------|-------------------|----------------|
| [22, 23, 41, 42, 70, 71] | + | + | - | - | - |
| [45, 115, 117, 119, 122] | +/- | + | - | + | - |
| [28, 80–82, 84, 85] | + | - | + | - | - |
| [27, 31, 66, 94–99] | - | - | - | - | + |
| [109–113] | + | - | + | - | + |

We notice that the approaches working on enhancing the resource perspective in business processes can be divided into two classes: Focusing on human resources, or focusing on Cloud resources. The former class extend business process models with information about human behavior (Line 1), as well as dependencies that exist between them using social computing techniques (Line 2). Whereas, other works privilege the application of semantic web in BPM to add formal semantic annotations and the possibility of reasoning and querying business process models (Line 4). For researches working on Cloud resource management, two classes are defined: in BPM context, or in a Cloud context. The former class provide models that aim at verifying and optimizing the use of Cloud resources in business processes however they do not

tackle the interoperability gap (Line 3). However the latter class involve the use of semantics by defining semantic models in order to support Cloud resource annotations and discovery (Line 5). We denote herein that none of the proposed approaches fulfill the combination of our proposed criteria. In our work, we aim to define a semantic model that provides formal and unified definitions for Cloud resources business process models by the means of using of Social techniques and Semantic web.

Table 3.2 illustrates a synthesis of the presented approaches in terms of our principles that we have identified in Section 1.4.1: *(i)* Exploitation of knowledge, *(ii)* Balanced computation complexity, and *(iii)* Fine-grained results. None of them can fulfill all principles.

| Approaches | Principles | | |
|---------------------------------|---------------------------|---------------------------------|----------------------|
| | Exploitation of knowledge | Balanced computation complexity | Fine-grained results |
| [45], [120], [102], [117], [72] | +/- | +/- | - |
| [123], [116], [22], [124] | +/- | +/- | + |
| [104], [73], [42], | - | +/- | - |

Table 3.2: Synthesis of the approaches for formalizing the resource perspective in Business processes

As a first step in our approach, we focus on semantically formalizing the resource perspective in business process models. To achieve this goal, we build on semantic definitions to unify the structure of Cloud resources that can be consumed in process models. Besides, we define social relations that describe the behavior of these resources. We represent this knowledge in a knowledge base. Then, we refine our model by allowing process users to configure its proper requirements in terms of Cloud resources using configurable process models. Our approach does not face the complexity problem since we separately define and evaluate the resource perspective regarding the control-flow perspective.

3.3 On supporting and Optimizing the resource variability in configurable process models

Configurable business process models enable the representation of the commonalities and differences that exist among different variants of a business process. To design these configurable models, two major steps are needed: (1) recognize the various process variants that can exist for a business process, and (2) combine these identified variants into one customizable process model.

Several approaches working on configurable process modeling have been proposed aiming at facilitating the design of configurable business process models and the resource allocation' configuration. Nevertheless, the experience showed that the man-

ual construction of a configurable process, and more specifically with its allocated resources, is a error-prone and time-consuming task. Moreover, there is a lack of an explicit support of the cloud resource variability. Therefore, automated approaches have been proposed in the literature to assist the design of configurable process models in terms of control-flow and resource perspectives.

In this section, we review existing approaches for supporting the resource variability in configurable process models and classify them into four categories: (i) configurable process modeling (Section 3.3.1), (ii) resource variability (Section 3.3.2), (iii) Optimization methods in business processes (Section 3.3.3) and (iv) QoS-aware business processes (Section 3.3.4). In Section 3.3.5, the proposed approaches are evaluated against criterions that are relevant to our thesis context.

3.3.1 Configurable process modeling

The limitation and rigid business process models has led to the definition of configurable process models [125]. Business process modeling permits the representation of business processes through appropriate graphical notations [126]. We reduce the complexity of real worlds by designing process models and concentrate efforts in the most important system' parts [127]. Several process modeling languages have been proposed to describe business processes such as Business Process Model and Notation (BPMN), Unified Modeling Language (UML), Event-driven process chain (EPC), Yet Another workflow Language (YAWL), XML Process Definition Language (XPDL), Extended Business Modeling Language (xBML), etc. Nevertheless, these languages are not capable of modeling the variability in business processes in order to design configurable processes. Hence, many configurable process modeling languages have been recently suggested to facilitate the modeling of configurable process [30,52,128–130]. Most of these proposals are extending existing languages such as BPMN [49], EPC [131], and UML [132] with customizable control-flow constructs [21].

In [52,128], Rosemann and al. describe the requirements for a configurable process modeling technique and introduce the configurable EPC notation. The EPC notation mainly consists of three control-flow elements: event, function and gateway. An event is a condition that triggers a function. A function describes the work which should be realized. Three types of connectors are used to model the splits and joins: OR, exclusive OR (XOR) and AND. Two constructs are added to the original EPC language: configurable nodes and configuration requirements and guidelines. These guidelines are useful to assist the users selecting the right configuration choices. Further, several approaches propose variability techniques are developed in order to manage different process' variants of organizations. For instance, Hallerbach et al. [133,134] introduce a method referred as Provop (PROcess Variant by OPTions) for managing and modeling process variants. This method is based on deriving a process variant from a reference model referred to as base model by applying a set of operations (INSERT/DELETE/MOVE fragment, MODIFY attribute). Gottschalk et al. [135]

propose an approach that includes the most known method for process configuration. This method is based on hiding and blocking operators to enable configurable workflow modeling using Labeled Transition Systems (LTSs). In [130], authors have used hiding and blocking operators to define the Configurable YAWL (C-YAWL) language that extends YAWL.

In spite of the usefulness of various configurable process modeling approaches, these works have been essentially focused on the control-flow perspective and neglect the resource perspective. Indeed, a configurable process model may allocate a set of resources to its activities, in particular Cloud resources in multi-tenant environments. Moreover, to define his own process variant each tenant may configure the set of resources according to his requirements. In our work, we propose to inform the process designer about the allocated resources and assist him to design his configurable resources (see Chapter 5). To do so, we build upon the approaches that use configurable nodes [52, 128] since it is at the top of the theoretical study on process model configuration [53], and allow a complete representation of all the possible process variants.

3.3.2 Resource Variability

Recently, works in the area of configurable process modeling have been proposed towards such a configuration of the resource perspective [25, 29, 30, 43].

In [5], Marcello La Rosa et al. propose the configurable integrated EPC (C-iEPC) with features for capturing resource, data and physical objects. Configuration of these elements is achieved using configurable connectors borrowed from the control-flow perspective to model the variable allocation of resources. The authors represent the variability of resources by proposing to associate the process functions to a variable number of resources and data objects through configurable connectors having a range parameter. The range allows to specify the minimal and maximal elements to be selected in a configuration choice. The Figure 3.1 depicts a configurable process model for audio editing modeled in Configurable Integrated EPC (C-iEPC). The configurable nodes called also variation points are modeled with thick lines. The authors propose *configurable roles* for human resources, *configurable objects* for objects, and *configurable range connectors* for restricting the branches. For instance, the object “Picture cut” related to the function “Picture editing” can be specialized to “Tape” if the project does not support an editing on Film. The role “Producer” associated with the function “Progress update” can be specialized to “Line Producer”. This can be made following role-hierarchy and object-hierarchy models already predefined. In addition, the configurable range connector 2:k related to the function “Spotting session” means that the minimal number of roles should be 2 and the maximal is 3.

A. Kumar and W. Yao in [29] propose a novel approach for modeling configurable business processes that integrates resource and data needs using process templates and business rules. They suggest different rule categories: Control-flow related rules,

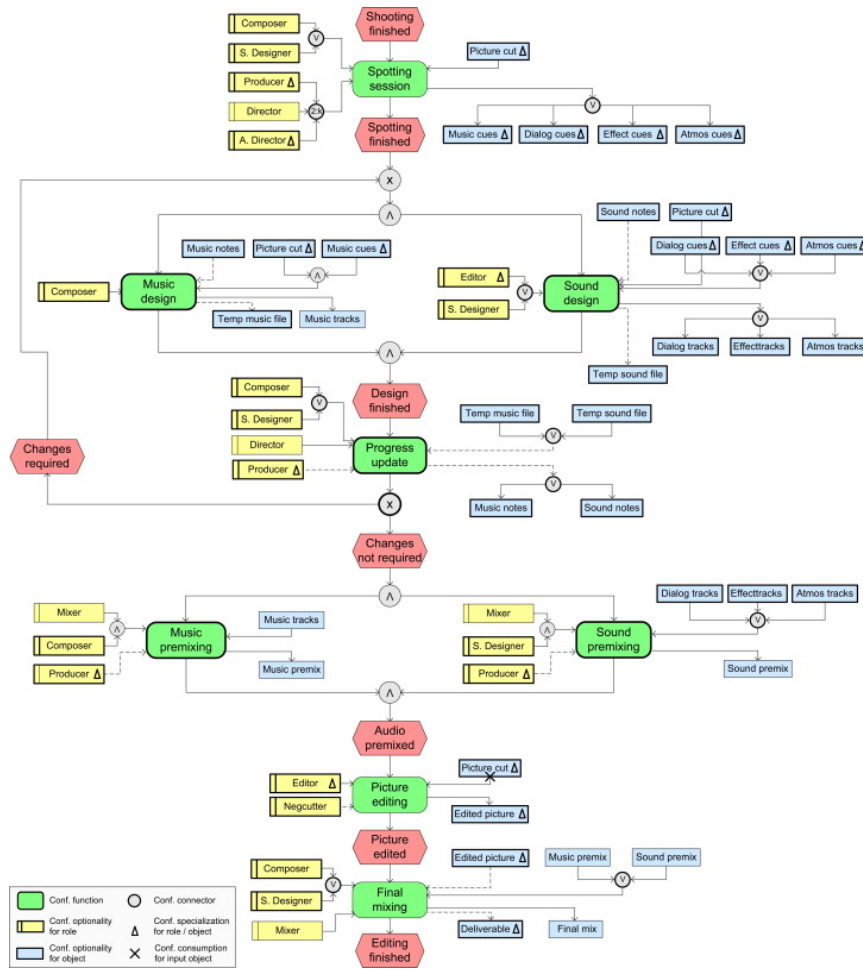


Figure 3.1: The audio editing process variant modeled in C-iEPC notation [5]

Resource related rules, Data related rules and hybrid rules. Indeed, the authors exploit additional information such as resource needs (e.g., equipment, and facilities) for task completion and data values of parameters associated with a task in order to realize a holistic process model.

To the best of our knowledge, few are the approaches that handle the resource variability in configurable process models and even less the proposals that allow the variability of Cloud resources. The focus of the aforementioned approaches is, yet, on human resources (the roles, hierarchies) and data (objects). Thus, there is no direct support for Cloud resources including *resource sharing* and *resource elasticity*. If configurable process models have the potential to be an efficient solution for modeling multi-tenant business processes [18], they need to integrate the resource perspective. In our proposal, we treat the variability of resources in Cloud environments by propos-

ing a set of configurable resource operators (see Chapter 5 Section 5.2). Besides, we aim at assisting the process designer to customize its process with his Cloud resources selection w.r.t a set of restrictions.

3.3.3 Resource Allocation Optimization in business processes

Optimizing resource allocation in BPM is a hot research field, however it remains at infancy level despite the recognized importance of the problem [136]. Yet, there exist approaches that handle such problem [75, 137–141].

Giray Havur et al. propose in [75] a formal technique that allows to acquire an optimal scheduling of workitems that have a set of dependencies and resource conflicts in Business Process Management Systems (BPMS). To achieve this goal, authors have employed the well-known formalism from logic programming Answer Set Programming (ASP). Also, in [76] same authors introduces an approach for automatically allocating resources to process activities in a time optimal way. They also are based on ASP as a formalism. Their objective is to find an optimal minimum amount of time where all process activities achieve their work given a set of resources. Whilst in [137], Van der Aalst propose a method for identifying a local optimal resource allocation at each process step using a greedy approach that might discover a feasible and optimal solution.

All of the above approaches do not tackle the resource allocation problem while considering the variability in business process models at both levels: control-flow and resource. In this thesis, we deal with such problem in configurable process models and integrate the Cloud resource variability.

Optimizing resource allocation in Cloud environments is also a challenging issue that has been considered in several approaches [142–144]. For instance, Wei Shu et al. propose in [143] an improved clonal algorithm based on time cost and energy consumption in Cloud Computing models. Whereas, in [144] Hwa-Min Lee et al. suggest performance analysis based resource allocation scheme aiming to an efficient allocation of virtual machines on the Cloud infrastructure. Fangzhe Chang et al. aim in [142, 145] at discovering the best solution of resource allocation in polynomial time. To this end, they propose algorithms to reduce the resource management cost. Cloud providers may thus use the solution to share their resources among a larger number of users. Molka Rekik et al [146] propose an approach for optimal configurable process deployment into a cloud federation. For this purpose, they use binary linear program to look for the best process variant with taking into account activities' capacity requirements, the enterprise QoS requirements and the variability requirements expressed in the configurable process.

On the contrary, some of these approaches do not take into account the configurability related to Cloud resources. Moreover, taking into consideration the requirements, that Cloud context impose essentially the elasticity and shareability, is relevant. In our thesis, we aim at discovering the best resource allocation alternative

that fulfill Cloud requirements while reducing cost and response time (see Chapter 5 Section 5.3).

3.3.4 QoS-aware business processes

In this section, we discuss related works that take two consideration two aspects: QoS properties, and Ecological properties.

3.3.4.1 QoS properties

The QoS properties depict the non-functional requirements which are also denoted as non-behavioral requirements or quality properties. They judge how well the functionality of the system in question is achieved. Several properties are involved e.g., cost, performance, cost, safety, etc. From QoS vision, organizations are constantly seeking to improve the non functional properties of their business processes in many aspects [6, 147–152]. There are domain-dependent aspects (e.g., resolution for image processing business processes) and domain-independent (e.g., response time, or cost). In fact, in [148] Wei Zhe Low et al. present a novel way to identify potential efficiency achievements in business operations through examining the way they were carried out in the past and thereafter exploring better ways of executing. For this purpose, they take into account trade-offs between three QoS properties: time, cost and resource utilization. The main objectives of this proposal are (1) to discover the different scenarios of process execution that are less expensive than the original scenario to gain insights for future redesign activities, (2) to propose a generic cost structure to answer for different cost trade-offs, and (3) to introduce optimization techniques to extract cost-optimal execution scenarios that consider multiple QoS properties.

Michael Adams et al. propose in [6] a framework for supporting process-related decisions, guided by cost-informed considerations, in workflow management systems (WfMS). To demonstrate the efficiency of their proposal, the authors present a realization of a cost-informed workflow environment within the well-known open-source WfMS system environment YAWL. To build such cost-informed WfMS framework, authors take into consideration a set of cost properties at all levels of a business process: For instance, we should associate cost information (e.g. cost rates, cost functions) to different elements in a process (e.g. activities, case attributes and durations). Also, process resources should be aware of suitable information (e.g. names, roles or experience levels). Moreover, the WfMS should (i) provide cost information about control-flow definitions and resource allocation rules (at design-time), and (ii) support system-based decisions and system-based user decisions (at runtime). Furthermore, the WfMS should be able to compute processing costs of a case at runtime and to provide a detailed cost analysis after execution. Figure 3.2 an overview of this framework that describes (i) Data input, i.e. the requirements to trigger actions that can be taken by a WfMS in order to support cost-informed decision making, (ii) the different actions that can be taken within a WfMS at various levels of process

(process, activity, and resource), and (iii) the output that is the system decisions and system supported user decisions.

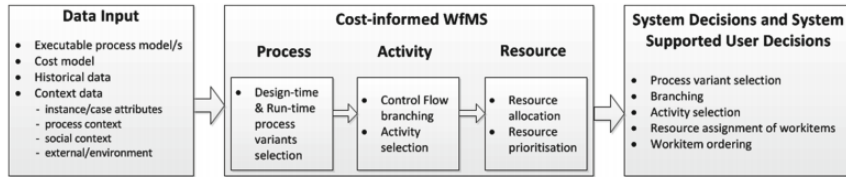


Figure 3.2: The cost-informed process support framework [6]

Ralph Mietzner et al. show in [153] that variability modeling and management techniques can correctly support the handling of variability in service-oriented Software as a Service (SaaS) applications. They propose to derive customization and deployment information for individual SaaS tenants, based on their requirements in particular availability-based needs. The objective within the proposed approach is to efficiently deploy SaaS applications for new tenants, using configurable models.

Despite the great support of QoS properties in business process models, the management of Cloud resources as well as their specificities (i.e., requirements in terms of elasticity and shareability) remain not taken into account. Besides, there is a lack of handling the heterogeneity issue that exist among process providers. In this work, we aim at filling these gaps by optimizing the cloud resource allocation in configurable process models w.r.t a set of QoS requirements that business process providers define for their variants (see Chapter 5 Section 5.3.3.3). We consider, in addition to the elasticity and shareability properties, three main QoS properties: cost, response time, and availability.

3.3.4.2 Ecological properties

In recent few years, organizations are increasingly improving the environmental impact of BPM in several domains e.g., security, compliance, Green IT [154–159]. In fact, user needs and legal and financial requirements are pushing organizations to perform their business processes in a green manner. We denote by “Green BPM” the class of technologies that leverage and extend existing BPM technology to enable process design, analysis, execution and monitoring in a manner informed by the ecological characteristics (e.g., carbon footprint) of process models and instances [156]. Business process models need to be aware about their associated emission impact for cost reductions, quality improvements, time savings, and increased flexibility [160].

The ecological characteristics are defined as multiple green metrics such as carbon footprint, CO2 emissions, and CPU utilization [161]. Based on these metrics specified for a business process, sustainable adaptation strategies can be chosen to enhance the total environmental impact of the business process. These strategies are denoted as “Green Business Process Patterns” [160].

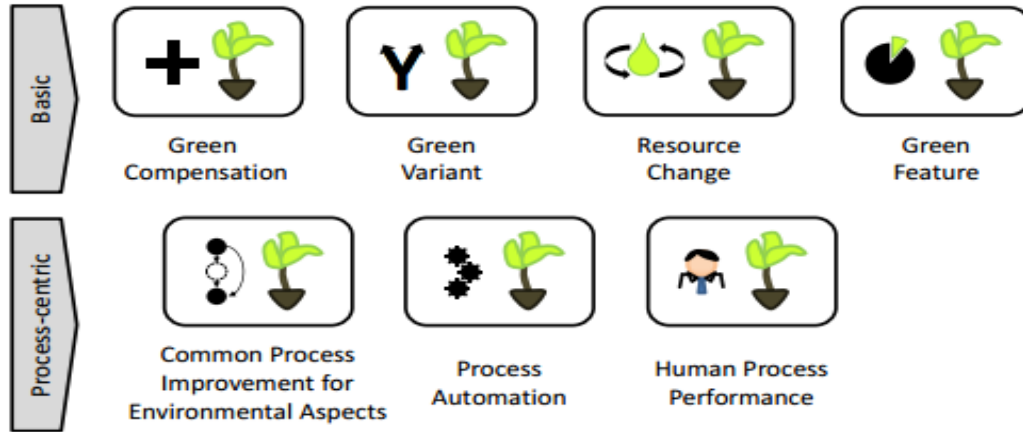


Figure 3.3: Green Business Process Classification [7]

A classification is proposed for the green patterns. As shown in Figure 3.3 two categories are introduced: (i) Basic patterns include optimization methods that are used on top of a business process, i.e. without the necessity of changing the structure of a process. The concerned patterns are: Green Compensation, Green Variant, Resource Change, and Green Feature. (ii) The process-centric patterns involve the changing of the structure of business processes and how to execute the activities. The concerned patterns are: Common Process Improvement for Environmental Aspects, Process Automation, and Human Process Performance.

Hoesch-Klohe Kostantin et al. in [157], highlight the role of Green BPM and discuss related challenges to reduce carbon emissions. Since then, several approaches have been proposed: Alexander Nowak et al. in [7] define green business process design patterns that cover the environmental impact, and develop a guiding method to help stakeholders to identify suitable patterns that fit to their domain of interest. While in [162], same authors introduce a cloud and green business process pattern correlation approach to support developers to adapt the green patterns' application. Moreover in [163], the latter authors discuss about BPM perspectives that can be extended, or refined to decrease the environmental impact of business processes. Their focus is to assist organizations to optimize their processes, using a set of identified key aspects with respect to the environmental impact while not omitting their business performance.

Yet, the above works do not neither present explicit methods to optimize the resource allocation in BPM while reducing the environmental impact, nor considering variability in process models. In our work, we aim at optimizing the Cloud resource allocation with taking into consideration a set of ecological properties (called also green properties) as metrics (see Chapter 5 Section 5.3.3.2). We take into account

three main metrics: Green Efficiency, Energy Productivity, and CO2 emissions. For more details, please refer to [164].

3.3.5 Synthesis

Table 3.3 summarizes the presented approaches and relates them to four properties that are important in our context: (i) Resource variability, (ii) Control-flow variability, (iii) Cloud Resources & features, and (iv) QoS properties.

Table 3.3: Evaluation of previous approaches

| Approaches | Criteria | | | | |
|----------------|---------------------|----------------------|--------------------------|----------------------------|----------------|
| | Resource Allocation | Resource variability | Control-flow variability | Cloud resources & features | QoS properties |
| [6, 147–150] | - | - | - | - | + |
| [30, 128–130] | - | - | + | - | - |
| [29, 30, 43] | +/- | + | + | - | +/- |
| [52, 153] | - | - | + | - | + |
| [142–144, 152] | + | - | - | +/- | + |

We observe that the properties Resource Allocation Cloud resources and features and Resource variability are poorly fulfilled for some works and not at all for others. We denote, also, that there is no approaches that combine the totality of the proposed properties. In our work, we aim to fulfill all of these properties especially Resource Allocation and variability with Cloud features.

Table 3.4 depicts a synthesis on the presented approaches in terms of our principles that we have already specified in Section 1.4.1: (i) Exploitation of knowledge, (ii) Balanced computation complexity, and (iii) Fine-grained results.

| Approaches | Principles | | |
|-------------------------------------|---------------------------|---------------------------------|----------------------|
| | Exploitation of knowledge | Balanced computation complexity | Fine-grained results |
| [5], [52], [30], [128], [29], [130] | +/- | +/- | - |
| [147], [149], [151], [6], [152] | +/- | +/- | + |

Table 3.4: Synthesis of the approaches for supporting and Optimizing the resource variability in configurable process models

The table shows that none of the approaches meet the totality of our identified principles. As a second step of our approach, we focus on providing a model that includes Cloud resource variability within configurable process models. We aim at assisting process users in the design and implementation of his process variant involving the appropriate configuration of Cloud resources. Thereafter, we intend to optimize, using genetic algorithms, the resource selection with respect to the Cloud features (i.e., elasticity and shareability) and QoS properties.

3.4 Conclusion

In this chapter we presented different approaches that support the design and configuration of Cloud resource allocation in business process models. We divided these approaches into two classes aiming at two main purposes: The formalization the resource perspective in business processes, and the support and optimization of the resource variability in configurable process models. The former class is classified into four categories: Resource perspective in business processes, Resource orchestration in Cloud-based business processes, Semantics in business process models, and Social business processes. The latter class is also categorized into four: Configurable process modeling, Resource Variability, Resource Allocation Optimization in business processes, and QoS-aware business processes. We briefly introduced these approaches and identified their principles. Thus, we demonstrate that most of the existing approaches do not allow the modeling of heterogeneous Cloud resources, especially semantic-based approaches. They also lack of automatic techniques that enable the sharing of heterogeneous process cloud resources in a common understanding. Moreover, such approaches do not present solutions to formalize the description of user requirements in terms of Cloud resources configuration, which is still in an ad-hoc manner. Furthermore, we showed the differences between current approaches and our approach.

In the following chapters we present in details our approach. In chapter 4, we present our approach for the semantic formalization of the resource perspective in business process models. We also present our common knowledge base to resolve the interoperability issue. In chapter 5, we show our approach that is based on how we assist the design and configuration of the use of Cloud resources in configurable process models. Moreover, we present an approach for optimally allocating these resources to business process variants using genetic algorithms.

Supporting Cloud Resource Allocation in Service-based Business Processes

Contents

| | | |
|------------|---|-----------|
| 4.1 | Introduction | 64 |
| 4.2 | Semantic Formalization of the Resource perspective in Business Processes | 65 |
| 4.2.1 | Resource perspective in Business Processes | 65 |
| 4.2.2 | Approach Overview | 66 |
| 4.2.3 | Extending the resource perspective in BPMN 2.0 | 67 |
| 4.2.4 | Semantic Model for Resource management in Business processes | 69 |
| 4.2.4.1 | Cloud-based process Ontology | 69 |
| 4.2.4.2 | Resource Constraints verification Rules | 72 |
| 4.2.4.3 | Process resource Properties and events | 72 |
| 4.2.4.4 | Rules formalization | 74 |
| 4.3 | Supporting Process Socialization in Cloud-based Business Process Models | 75 |
| 4.3.1 | Approach Overview | 75 |
| 4.3.2 | Social-based Resource Management in Business Processes | 76 |
| 4.3.2.1 | So-CloudPrO: Social Cloud business Process Ontology | 76 |
| 4.3.2.2 | Ontology Definition | 76 |
| 4.3.2.3 | OCCI Resource Structure | 80 |
| 4.3.2.4 | Resource social dependencies | 80 |
| 4.3.3 | Resolving resource-based Conflicts using SWRL Strategies | 82 |
| 4.3.3.1 | Instantiation Constraints | 82 |
| 4.3.3.2 | Resource and Task Constraints | 82 |
| 4.3.3.3 | Resolution Strategies | 83 |
| 4.4 | Evaluation and validation | 86 |
| 4.4.1 | Ontology Validation | 86 |

| | | |
|------------|--|-----------|
| 4.4.2 | Supporting Cloud resource Descriptions | 87 |
| 4.4.3 | Cloud resource Knowledge Base | 88 |
| 4.5 | Conclusion | 88 |

4.1 Introduction

This chapter presents our approach for supporting Cloud resource allocation in service-based business processes. As mentioned in Chapter 1, resource descriptions are of a high heterogeneity with their management strategies since each Cloud provider define its proper resource consumption policies, APIs, and networking models. For example, a provider uses OCCI (Open Cloud Computing Interface) API [46], while another uses a different API OpenTOSCA¹ (OASIS Topology and Orchestration Specification for Cloud Applications) to define cloud resources. As a result, a unified common understanding between providers and tenants becomes a must.

Besides the several benefits of cloud resources in terms of economy of scale and pay-per-use, a main issue regarding their correct use still exists which is to ensure a controlled resource allocation w.r.t. SLA constraints. For instance a tenant may request not to share a CPU resource with other tenants, or may want to use a particular *elasticity* policy. Another example is when a resource stops responding, it must be automatically and quickly substituted by resources so that SLAs are satisfied.

To address the above challenges, we propose in this chapter two approaches: (i) *a semantic formalization of the resource perspective in business processes* (Section 4.2), and (ii) *a social-based semantic framework for cloud resource management in business processes* (Section 4.3). The former approach provides (i) an extension of the process modeling language BPMN that allows to incorporate the definition of resources in cloud environments, and (ii) a Cloud-based process ontology based on an extension of BPMO. The latter approach, which depicts an extension of the previous proposal, provides (i) a Social-based Cloud business Process Ontology w.r.t OCCI architecture, and (ii) a resolution of conflicts over resources to insure their correct allocation.

To define our first approach, we start by introducing a general description of the resource perspective in business processes (Section 4.2.1). Next, we present an overall overview of our proposed approach where we specify the inputs, the outputs as well as the stages of our proposal (section 4.2.2). An extension of the process modeling language BPMN is then proposed. This extension allows to incorporate the definition of resources in cloud environments (section 4.2.3). Then, we detail our semantic framework for a semantically-enriched resource description in BPs (section 4.2.4).

To define our second approach, we make use of a recent R&D trend which advocates for analyzing BPs from a **social perspective** [165]. This vision aims at building *social* relations between the components of a BP, namely *task*, *person*, and *machine*,

¹<http://www.iaas.uni-stuttgart.de/OpenTOSCA/indexE.php>

and how these relations capture previous experiences associated with interactions (i.e., task-2-task, person-2-person, and machine-2-machine) that could help improve BPM [166–168]. Many organizations already acknowledge the role of Web 2.0 technologies and applications in improving their images, reaching out to more customers, and adjusting their BPs in response to social media [121, 165]. Thus, we adopt the feasibility of identifying *social* relations between human resources to cloud resources which are assigned to BPs so that networks of resources are developed in conjunction with networks of processes, tasks, and persons. Afterwards, we define a set of strategies for resolving conflicts over resources using Semantic Web Rule Language (SWRL). We start by introducing a general view of our approach (Section 4.3.1). Then, we discuss our approach which mainly focus on two goals (i) Building a semantic formal model to describe cloud resources according to OCCI architecture (Section 4.3.2), and (ii) resolving resource-based conflicts using semantic rules (Section 4.3.3). Finally we present an evaluation and validation part to our approaches (Section 4.4).

The work of this chapter was published in conference proceedings [169, 170].

4.2 Semantic Formalization of the Resource perspective in Business Processes

4.2.1 Resource perspective in Business Processes

As mentioned before, the resource perspective in business processes lacks a formal and unified description. It is poorly operated compared to other perspectives such as the control-flow [65, 66] or the organizational perspective [55, 171]. But with regard to recent efforts realized to bridge this gap, the resource perspective in business process models has been classified into three important classes: resource structure, work distribution, and authorization [70]. The resource structure involves two aspects: the characterization and classification of resources. The characterization is the definition of the resource related information. Once resources' information are characterized, a classification of the resources can be established based on a set of common properties. The work distribution is concerned with the work distribution and its binding to specific resources for execution. In other words, it defines the manner in which the work of a process is distributed and allocated to resources. The authorization deals with the definition of privileges owned by resources regarding the execution of operations in order to organize the work advertised to them. In our thesis, we follow this classification in order to extend the resource perspective in business processes using BPMN 2.0 (see section 4.2.3).

As we are interested in the Cloud setting, we have studied the specificities of the available resources that can be used. As mentioned before (section 2.3), the Cloud delivers three important types of resources on demand which are: computing, storage, and networks. The computing resources provide mechanisms in order to deploy and run softwares [117]. This type of resource depicts an information processing resource

(e.g. virtual machine). The Cloud storage simply represents an information recording resource in data storage devices. It offers great benefits as for example reliability, faster deployment, reduced costs and protection insurance in case of loss. Actually there are different ways to use this type of resources depending on users requirements: private data or shared data. Network resources allow to have mechanisms that are used for communication, and might also offer added-value services such as load balancers [172]. The network type plainly denotes an interconnection resource (e.g a virtual switch). These resource types are caught from the cloud resource description in a specific cloud computing API which is OCCI [173] We formally describe these resources in our semantic proposal (see section 4.2.4).

4.2.2 Approach Overview

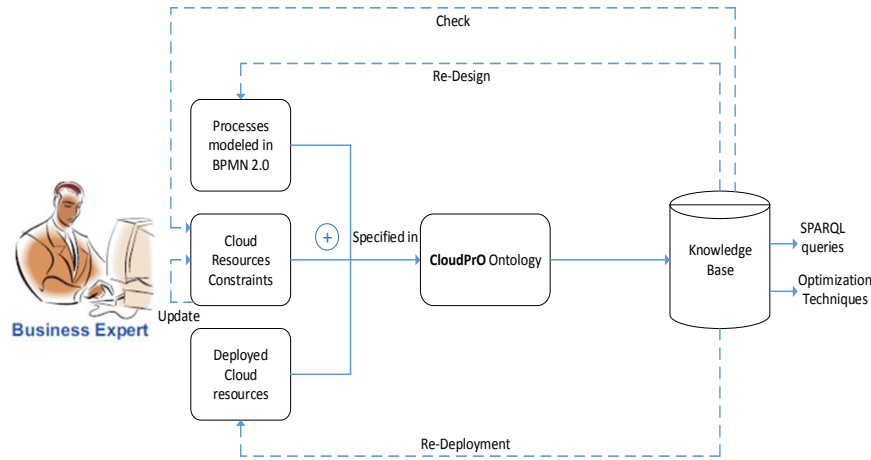


Figure 4.1: Approach Overview

We present, in the present section, an overall overview of our proposed approach toward the formalization of the resource perspective in cloud-based business processes through the use of semantic techniques. As shown in Figure 4.1, our approach uses three inputs: (i) Business processes described in the extended BPMN (see section 4.2.3), (ii) the deployed Cloud resources, and (iii) Cloud resource constraints that comprise rules and properties to be and/or checked (see section 4.2.4.2). This latter entry is of a high importance because, by property verification and constraint compliance, we aim at ensuring the proper management of resource utilization.

To this end, we first establish a **CloudPrO** ontology (see section 4.2.4.1) which formally specify these three entries by describing consumed resources and their associated dependencies in BPs, in a semantic and uniform way. To do so, we use the RDF/RDFS language in order to model the business processes and cloud resources components, and Semantic Web Rule Language (SWRL) rules to properly define the constraints of resources. This ontology is then stored in a knowledge base which can

be later interrogated with SPARQL queries in order to further apply optimization techniques. Furthermore, cloud resource constraints can be checked using rule verification engines (e.g. Protege, Bossam, Hozo). The knowledge base is continuously updated whenever the business process is re-designed, the constraints are updated or the cloud resources are re-deployed.

We advocate that our approach can be applied at design- and execution-time: on one hand, at design-time, the control-flow and resource perspectives of the modeled business process, the resource constraints and the deployed cloud resources are formalized in the **CloudPro** ontology, and the process resource distribution is verified against the defined constraints. On the other hand, at run-time, the resource consumption can be continuously checked during the process execution according to the pre-defined constraints.

4.2.3 Extending the resource element in BPMN 2.0

The BPMN notation is extensively used in the BPM community for business process modeling. The latest version of BPMN 2.0 incorporates resource perspective concepts which are *resource assignment* and *human interactions*.

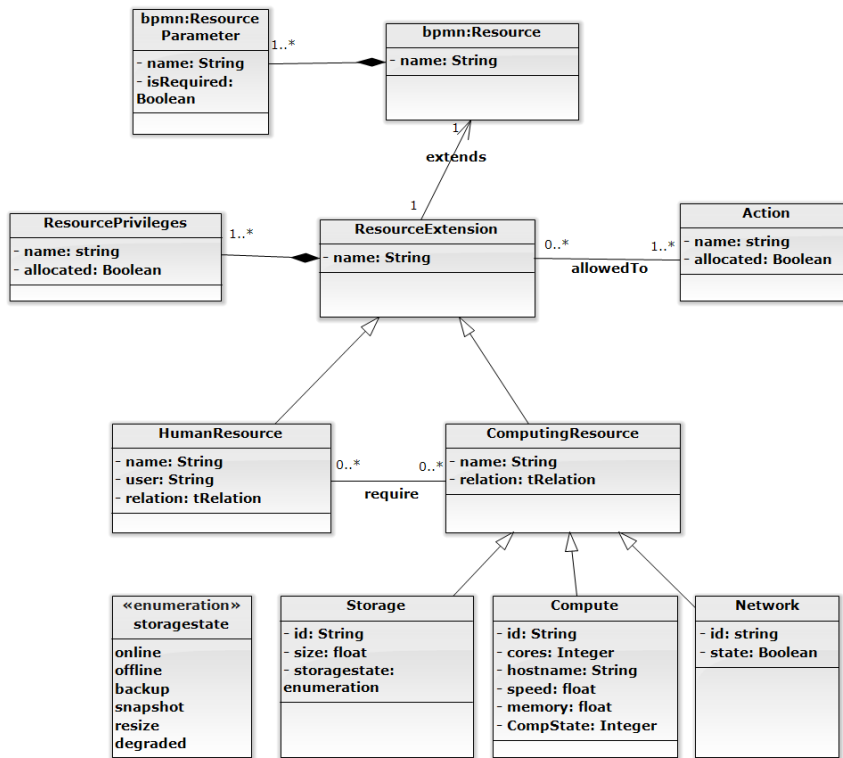


Figure 4.2: Extension of the resource element in BPMN

Nevertheless, it does not provide a formal definition of resource types in cloud environments. Therefore, we propose an extension, which is called “ResourceExtension”, to the existing BPMN element named “bpmn:Resource” (see Figure 4.2). A “ResourceExtension” could be a resource of type human referred to as “HumanResource” or a non human resource denoted as “ComputingResource”. The latter one represents the cloud resources which are Storage, Compute and Network. The “HumanResource” and “ComputingResource” may require each others.

Listing 4.1: xsd document to extend BPMN

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <xs:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
3 targetNamespace="http://www.omg.org/spec/BPMN/20100501/Semantic.xsd"
4 xmlns:bpmn20="http://www.omg.org/spec/BPMN/20100501/BPMN20.xsd" >
5 <xs:import namespace="http://www.omg.org/spec/BPMN/20100501/BPMN20"
6 schemaLocation="http://www.omg.org/spec/BPMN/20100501/BPMN20.xsd"/>
7
8 <xsd:group name="ResourceExtension">
9   <xsd:sequence>
10     <xsd:element name="HumanResource" type="tHumanResource" minOccurs="0"/>
11     <xsd:element name="ComputingResource" type="tComputingResource" minOccurs="0"/>
12     <xsd:element name="ResourcePrivileges" type="tResourcePrivileges" minOccurs="0"/>
13     <xsd:element name="Actions" type="tActions" minOccurs="1"/>
14   </xsd:sequence>
15 </xsd:group>
16
17 <xsd:complexType name="tHumanResource">
18   <xs:attribute name="name" type="xsd:string"/>
19   <xs:attribute name="user" type="xsd:string"/>
20   <xs:attribute name="relation" type="tRelation"/>
21 </xsd:complexType>
22
23 <xsd:group name="tComputingResource">
24   <xs:attribute name="name" type="xsd:string"/>
25   <xs:attribute name="relation" type="tRelation"/>
26   <xsd:sequence>
27     <xsd:element name="Storage" type="tStorage" minOccurs="0" />
28     <xsd:element name="Compute" type="tCompute" minOccurs="0" />
29     <xsd:element name="Network" type="tNetwork" minOccurs="0"/>
30   </xsd:sequence>
31 </xsd:group>
32
33 <xsd:complexType name="tResourcePrivileges">
34   <xs:attribute name="name" type="xsd:string"/>
35   <xs:attribute name="allocated" type="xsd:boolean"/>
36 </xsd:complexType>
37 <xsd:complexType name="tActions">
38   <xs:attribute name="name" type="string"/>
39   <xs:attribute name="accorded" type="boolean"/>
40 </xsd:complexType>
41 <xsd:complexType name="tRelation">
42   <xs:attribute name="source" type="ResourceExtension"/>
43   <xs:attribute name="destination" type="ResourceExtension"/>
44 </xsd:complexType>
45 <xsd:complexType name="tStorage">
46   <xs:attribute name="name" type="string"/>
47 </xsd:complexType>
48 <xsd:complexType name="tCompute">
49   <xs:attribute name="name" type="string"/>
50 </xsd:complexType>
51 <xsd:complexType name="tNetwork">
52   <xs:attribute name="name" type="string"/>
53 </xsd:complexType>
54 </xs:schema>

```

A “ResourceExtension” is composed of one or more privileges denoted as “ResourcePrivileges” according to whether the resource is private or shared respectively. The “ResourcePrivileges” depicts the resource state (allocated or not). Whereas, the “Actions” element represents a workitem allowed on a resource as for example “execute a resource”, “terminate a resource”, “duplicate a resource”, etc. We note that

we rely on the OCCI standard to define the structure of cloud resources, however the OCCI integration is out of scope at this stage of our thesis.

The class diagram shown in Figure 4.2 is translated to an xsd document as shown in Listing 4.1. The “ResourceExtension” is defined using “xsd group” (Line 8) which includes the elements “HumanResource”, “ComputingResource”, “ResourcePrivileges”, “Actions” (Lines 9-14). Each of these elements has a type which can be a “complexType” or “xsd group”. A “complexType” consists of a set of simple or complex attributes. For example, “HumanResource” is a “complexType” (Line 17-21) which has the simple type attributes “name” and “user” and the “complexType” attribute “relation”. This xsd extension is imported in the xsd of BPMN 2.0.

4.2.4 Semantic Model for Resource management in Business processes

In the previous section, we presented our approach for extending BPMN with a cloud-based resource perspective. In this section, we define a semantic model for resource management in business processes. To do so, we develop a cloud-based process ontology referred to as **CloudPrO** which semantically specify our extended BPMN (see section 4.2.3). Then we define a set of resource constraints over predefined properties which are formalized through verification rules (see section 4.2.4.2).

4.2.4.1 Cloud-based process Ontology

The **CloudPrO** ontology aims at providing a semantic-based resource-aware business process description and is written in RDF/RDFS format (see Figure 4.3).

Its conception is extending the BPMO ontology which has been developed in the European project SUPER (section 3.2.3). As explained before, the purpose of BPMO is to represent business processes at an abstract level of detail in order to ensure a high interoperability between business processes modeled with different languages. It includes concepts that define the semantics of the most used notations in BPM community such as BPMN, EPC, BPEL, etc. For example, BPMO includes the concepts “process”, “activity”, “resource”, etc which are common for all modeling languages. Formally, the CloudPrO ontology is a 4 tuple $\langle C; A; P; R \rangle$ where C represents the set of concepts, A is the set of concept attributes, P is the set of verification properties, and R is the set of constraints rules. The main concept in **CloudPrO** is “Resource” which has the sub-concept “ResourceExtension”. The concept “ResourceExtension” has two sub-concepts “HumanResource” and “ComputingResource”. The “ComputingResource” has three sub-concepts “Storage”, “Compute” and “Network”. Dependencies between human resources are defined through properties. We define three types of relationships: *Substitution*, *Delegation* and *Peering* (see Table 4.1).

The *Substitution* relation allows a person p_j to replace a person p_i if this latter is planned to be not available. The *Delegation* dependency differs from the *Substitution*

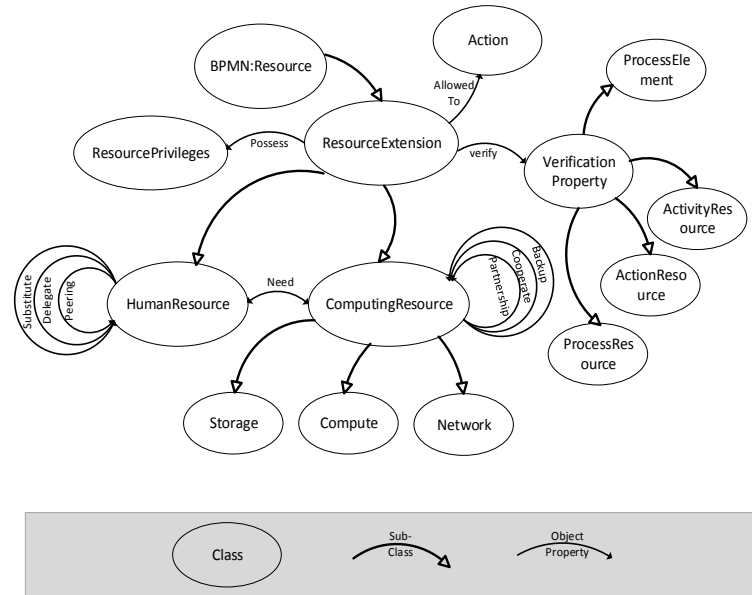


Figure 4.3: The CloudPrO ontology

where p_i is not available in unexpected way. Two persons are in a *Peering* relation if they are assigned to the same activity. Regarding the “ComputingResource” concept, we define three types of relationships: *Cooperation*, *Partnership* and *Backup* (see Table 4.1). Two resources are in a *Cooperation* if they have the same capacity and assigned to the same activity. The *Partnership* differs from the *Cooperation* relation where the resources have complementary capacities. The *Backup* relation enables a resource c_j to replace another resource c_i if this latter face and have the same capacity. The *need* dependency between the “ComputingResource” and the “HumanResource” concepts exemplifies the case when a human calls for a computing resource and vice versa. An action is allowed on a “resourceExtension”. The “Action” concept can be instantiated to ‘create’, ‘edit’, ‘duplicate’, etc. Finally, the “ResourcePrivileges” concept enables an instance of a “ResourceExtension” to be allocated or not to an activity. The set of concept attributes are as defined in the BPMN extension (see section 4.2.3).

Listing 4.2 depicts an excerpt of the **CloudPrO** ontology in *RDF* format. In this snippet, we define the dependencies between “ComputingResource” and the “HumanResource” concepts. Each relationship is defined as *owl:ObjectProperty*. For example, we define the relationship *backup* between two instances of the concept “ComputingResource” (Line 8,9).

Table 4.1: Resource Dependencies Descriptions

| Relation Type | Dependency Name | Description |
|---------------------------------|------------------------------------|---|
| Human resource dependencies | <i>Substitution</i> (p_i, p_j) | p_j replaces p_i if expected unavailability of p_i |
| | <i>Delegation</i> (p_i, p_j) | p_j replaces p_i if unexpected unavailability of p_i |
| | <i>Peering</i> (p_i, p_j) | p_i and p_j are assigned to the same activity which needs both capacities |
| Computing resource dependencies | <i>Cooperation</i> (c_i, c_j) | Execution of both resources c_i and c_j if concerned activity needs both similar capacities |
| | <i>Partnership</i> (c_i, c_j) | Execution of both resources c_i and c_j if concerned activity needs both complementary capacities |
| | <i>Backup</i> (c_i, c_j) | c_j replaces c_i if failure of c_i and capacities of c_i and c_j are similar |
| Hybrid resource dependencies | <i>need</i> (p_i, c_j) | Execution of both resources p_i and c_j if concerned activity needs both capacities |

Listing 4.2: Relationships CloudPro Ontology Snippet

```

1 <rdf:RDF xmlns="http://www.owl-ontologies.com/Ontology-essaiResBP.owl#"
2   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
3   xmlns:owl="http://www.w3.org/2002/07/owl#"
4   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
5   xmlns:Ontology-essaiResBP="http://www.owl-ontologies.com/Ontology-essaiResBP.owl#">
6
7 <owl:ObjectProperty rdf:about="#backup">
8   <rdfs:range rdf:resource="#ComputingResource"/>
9   <rdfs:domain rdf:resource="#ComputingResource"/>
10 </owl:ObjectProperty>
11 <owl:ObjectProperty rdf:about="#cooperate">
12   <rdfs:domain rdf:resource="#ComputingResource"/>
13   <rdfs:range rdf:resource="#ComputingResource"/>
14 </owl:ObjectProperty>
15 <owl:ObjectProperty rdf:about="#delegate">
16   <rdfs:range rdf:resource="#HumanResource"/>
17   <rdfs:domain rdf:resource="#HumanResource"/>
18 </owl:ObjectProperty>
19 <owl:ObjectProperty rdf:about="#need">
20   <rdfs:type rdf:resource="&owl;SymmetricProperty"/>
21   <rdfs:range rdf:resource="#ComputingResource"/>
22   <rdfs:domain rdf:resource="#HumanResource"/>
23   <owl:inverseOf rdf:resource="#need"/>
24 </owl:ObjectProperty>
25 <owl:ObjectProperty rdf:about="#partnership">
26   <rdfs:range rdf:resource="#ComputingResource"/>
27   <rdfs:domain rdf:resource="#ComputingResource"/>
28 </owl:ObjectProperty>
29 <owl:ObjectProperty rdf:about="#peering">
30   <rdfs:range rdf:resource="#HumanResource"/>
31   <rdfs:domain rdf:resource="#HumanResource"/>
32 </owl:ObjectProperty>
33 <owl:ObjectProperty rdf:about="#substitute">
34   <rdfs:domain rdf:resource="#HumanResource"/>
35   <rdfs:range rdf:resource="#HumanResource"/>
36 </owl:ObjectProperty>
37 <owl:ObjectProperty rdf:about="#allowTo">
38   <rdfs:range rdf:resource="#Actions"/>
39   <rdfs:domain rdf:resource="#ResourceExtension"/>
40 </owl:ObjectProperty>

```

4.2.4.2 Resource Constraints verification Rules

So far, the concepts, their attributes and dependencies of the **CloudPrO** ontology are defined. In the followings, we complete the ontology description with the definition of resource constraints over a set of verification properties.

4.2.4.3 Process resource Properties and events

We classify the properties with respect to the process elements and resources using the *RDF/RDFS* format. Concretely, four classes of properties are defined:

1. Process elements related properties
2. Activity/Resource related properties
3. Action/Resource related properties
4. Process resource related properties

According to our first approach in this chapter, a concept instance can be written following this format, we take as example a storage resource:

$$\begin{aligned} & \text{we denote } \text{storage}(X) \\ & \text{iff } X \text{ rdf:type cloudPrO:Storage.} \end{aligned}$$

Let $O_{bp} = \langle C_{bp}, A_{bp}, P_{bp}, R_{bp} \rangle$ be the BPMO ontology,
and $O_{CprO} = \langle C, A, P, R \rangle$ be the CloudPrO ontology,
and $C_r = \text{HumanResource, ComputingResource} \subset C$.

Definition 1: ProcessElement related Properties

A verification property X is a ProcessElement related Property, denoted as P_p^X ,
iff $P_p^X \text{ rdf:type cloudPrO:ProcessElement}$,
and $P_p^X \text{ rdfs:subClassOf cloudPrO:VerificationProperty}$,
and $\exists C_i, C_j, C_i \in C_{bp}, C_j \in C_{bp}$,
 $(C_i P_p^X C_j) \text{ rdf:type rdf:Statement}$,
 $P_p^X \text{ rdf:subject } C_i$,
 $P_p^X \text{ rdf:object } C_j$,
where we denote $(C_i P_p^X C_j)$ as $P_p(C_i, C_j)$

For instance, the process element related property $P_p^{\text{linkedWith}} = P_p(\text{Activity, ExclusiveGateway})$ describes the relation between the process element concepts “Activity” and “ExclusiveGateway”. According to our motivating example presented in section 1.3, this property relates the activity a_3 (customer orders follow up) with the gateway o_4 .

Definition 2: ActivityResource related Properties

A verification property X is a ActivityResource related Property, denoted as P_{ar}^X ,
iff P_{ar}^X *rdf:type* *cloudPrO:ActivityResource*,
and P_{ar}^X *rdfs:subClassOf* *cloudPrO : VerificationProperty*,
and $\exists C_i, C_j, C_i \in C_{bp}$,
 $C_j \in C$,
 $(C_i P_{ar}^X C_j)$ *rdf:type* *rdf:Statement*,
 P_{ar}^X *rdf:subject* C_i ,
and P_{ar}^X *rdf:object* C_j ,
where we denote $(C_i P_{ar}^X C_j)$ as $P_{ar}(C_i, C_j)$

For example, the ActivityResource related Property $P_{ar}^{consume} = P_{ar}(Activity, Resource)$ describes that an instance of “Activity” consumes an instance of “Resource”. If we refer to our example in section 1.3, this property links two times the activity a_6 (technical referential) with two resources c_5 and st_6 .

Definition 3: ActionResource related Properties

A verification property X is a ActionResource related Property, denoted as P_{acr}^X ,
iff P_{acr}^X *rdf:type* *cloudPrO:ActionResource*,
and P_{acr}^X *rdfs:subClassOf* *cloudPrO : VerificationProperty*,
and $\exists C_i, C_j, C_i \in C, C_j \in C$
 $(C_i P_{acr}^X C_j)$ *rdf:type* *rdf:Statement*,
 P_{acr}^X *rdf:subject* C_i ,
and P_{acr}^X *rdf:object* C_j ,
where we denote $(C_i P_{acr}^X C_j)$ as $P_{acr}(C_i, C_j)$

The ActionResource related Property $P_{acr}^{allowedTo} = P_{acr}(Action, Resource)$ describes that an instance of “Resource” is allowed execute an “Action”. For instance, the cloud provider defines that the resource st_6 is allowed be duplicated, so the according action (*duplicate*) is linked to this resource via the property *allowedTo*.

Definition 4: ProcessResource related Properties

A verification property X is a ProcessResource related Property, denoted as P_r^X ,
iff P_r^X *rdf:type* *cloudPrO:ProcessResource*,
and P_r^X *rdfs:subClassOf* *cloudPrO : VerificationProperty*,
and $\exists C_i, C_j, C_i \in C_r, C_j \in C_r$
 $(C_i P_r^X C_j)$ *rdf:type* *rdf:Statement*,
 P_r^X *rdf:subject* C_i ,
and P_r^X *rdf:object* C_j ,
where we denote $(C_i P_r^X C_j)$ as $P_r(C_i, C_j)$ or $P_r^X(C_i)$ if $C_i = C_j$

The resource related Property $P_r^{backup} = P_r(Resource, Resource)$ describes the dependency type *backup* between two instances of the Resource concept. $P_r^{Storage} = P_r(Resource)$ describes an instantiation of a storage resource type. For example, the cloud provider defines that a resource c_6 can replace the work of the resource c_3 if this latter is no more available.

4.2.4.4 Rules formalization

Using our classified properties, we define in this section a set of rules for constraints verification through the ECA [174] (Event/Conditions/Actions) structure formalism. An ECA rule has the general syntax:

$$\text{On Event If Conditions Do Actions} \quad (4.1)$$

The *Event* part specifies that the rule can be triggered. The *Conditions* part is then verified and if satisfied the action part is executed. In order to formally describe our constraint rules following the *ECA* structure formalism, we use the *SWRL* language. A constraint rule is defined as:

$$\bigwedge_i E_i \bigwedge_j C_j \Rightarrow \bigwedge_k A_k \quad (4.2)$$

where $\bigwedge_i E_i$, $\bigwedge_j C_j$ and $\bigwedge_k A_k$ are conjunctions of different verification properties and are called the *rule antecedent* and *rule consequent* respectively. $\bigwedge_i E_i$ is a set of events, $\bigwedge_j C_j$ is a set of conditions, and $\bigwedge_k A_k$ is a set of actions.

We define three types of constraint rules:

- **Simple rules:** are the rules that involve only one resource type, i.e. $E_i = (\cup P_{ar}^X) \cup (\cup P_{acr}^X)$ and $C_j = (\cup P_r^X)$ and $A_k \neq E_i, A_k \neq C_j$ where X is a unique resource type.

An example of a simple rule is $Storage(Y) \wedge allocated(A, Y) \rightarrow consume(A, Y)$. This rule states that if a storage resource Y is instantiated (event) and allocated by an activity A (condition) then Y is consumed to A (action). Returning to our example depicted in section 1.3, if the resource c_2 is allocated to the activity a_5 via the property *allocated* then it is concretely consumed.

- **Composite rules:** are the rules that involve two or many resource type, i.e. $\exists P_1 = E_1^X(Z) \text{ or } P_1 = C_1^X(Z) \wedge P_2 = A_2^X(Y) : P_1, P_2 \in (\cup_i E_i) \cup (\cup_j C_j) \cup (\cup_k A_k) \wedge Z \neq Y$.

An example of a composite rule is $consume(A, Z) \rightarrow Storage(Y) \wedge allocated(A, Y)$. This rule states that if the activity A consumes the resource Z (condition) then a storage resource Y is allocated to A (action). It is true for the resources st_6 and c_5 in our example: if the activity a_6 consumes c_5 then it is requiring a storage resource st_6 to be allocated.

- **Dependency-based rules:** are the rules that involve at least one of the process resource related properties, i.e. $\exists P \in (\cup_i E_i) \cup (\cup_j C_j) \cup (\cup_k A_k)$, $P \in (\cup_r P_r^X)$.

An example of a dependency-based rule is $consume(A, X) \wedge notstorage(X) \wedge substitute(Y, X) \rightarrow allocated(A, Y)$. This rule states that if the resource X disappears (action) and an activity Y consumes X and there is a substitute (conditions) then A consumes Y. Referring to our example, the cloud provider defines that a resource st_{10} can substitute st_9 , which is consumed by the activity a_2 , if this resource does not respond anymore (i.e., no more available).

4.3 Supporting Process Socialization in Cloud-based Business Process Models

4.3.1 Approach Overview

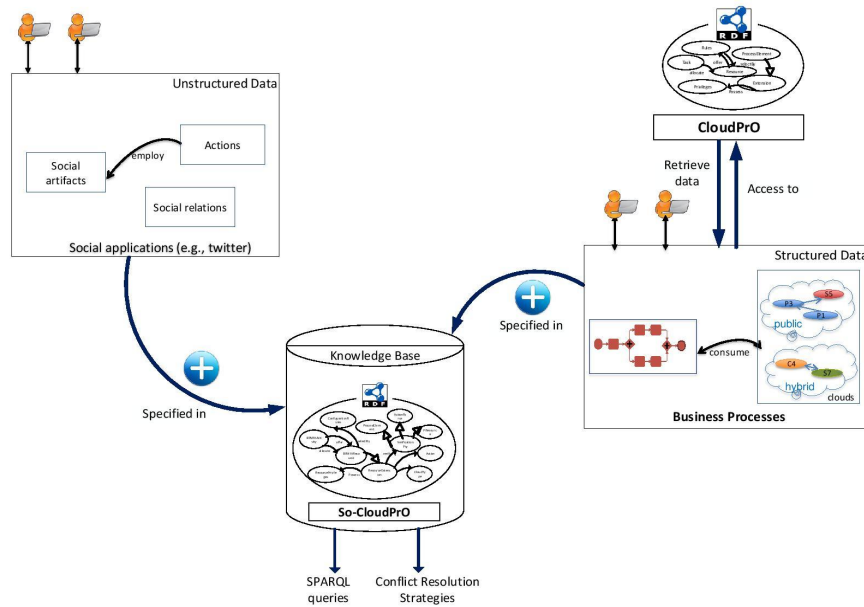


Figure 4.4: Overview

We present in this section a global view of our proposed approach. As shown in Figure 4.4, our approach considers two inputs: (i) the structured nature of the *business world* in terms of BPs and (ii) the unstructured nature of the *social world* in terms of relations established between business process components (e.g., tasks, users)

especially between resources and business process. The latter input have already access to the *CloudPrO* developed in the precedent chapter (Section 4.2.4.1). Both inputs are then specified in a shared knowledge base which is built upon a semantic model which extends BPMO and described in details in the following (Section 4.3.2.1). This knowledge base could be later interrogated with SPARQL queries in order to resolve conflicts in order to achieve free-of-conflict resource allocation in a business process development.

4.3.2 Social-based Resource Management in Business Processes

Our framework is built upon a semantic model that provides a formal description of resources used during resource management in BPs. To put into practice the semantic of BP modeling, we capitalize on the *CloudPrO* ontology which provides semantic definitions for resources in BPs by extending BPMO, that is previously detailed (Section 4.2.4.1). We aim herein at identifying most of the resource entities and relations among them.

4.3.2.1 So-CloudPrO: Social Cloud business Process Ontology

We develop a **Social CloudPrO** ontology (**So-CloudPrO**) that covers these two worlds, which representation is depicted in Figure 4.5. The prefix “SCP” refers to the **So-CloudPrO** namespace and the prefix “Bpmo” represents the BPMO namespace.

As mentioned before, the structure of cloud resources is caught from the cloud resource specification in the OCCI API since it represents the most popular standard. OCCI is a RESTful protocol and API that mainly serves IaaS layer. For sake of simplicity, we only focus on IaaS model for not increasing the complexity of the cloud resource management.

Formally, **So-CloudPrO** is a 4 tuple $\langle C_{So-CloudPrO}; A_{So-CloudPrO}; R_{So-CloudPrO}; S_{So-CloudPrO} \rangle$ where $C_{So-CloudPrO}$ is a concept set (Section 4.3.2.2), $A_{So-CloudPrO}$ is attribute set (Section 4.3.2.2), $R_{So-CloudPrO}$ is a set of rules based on existing relations (Section 4.3.2.4), and $S_{So-CloudPrO}$ is a set of strategies (Section 4.3.3).

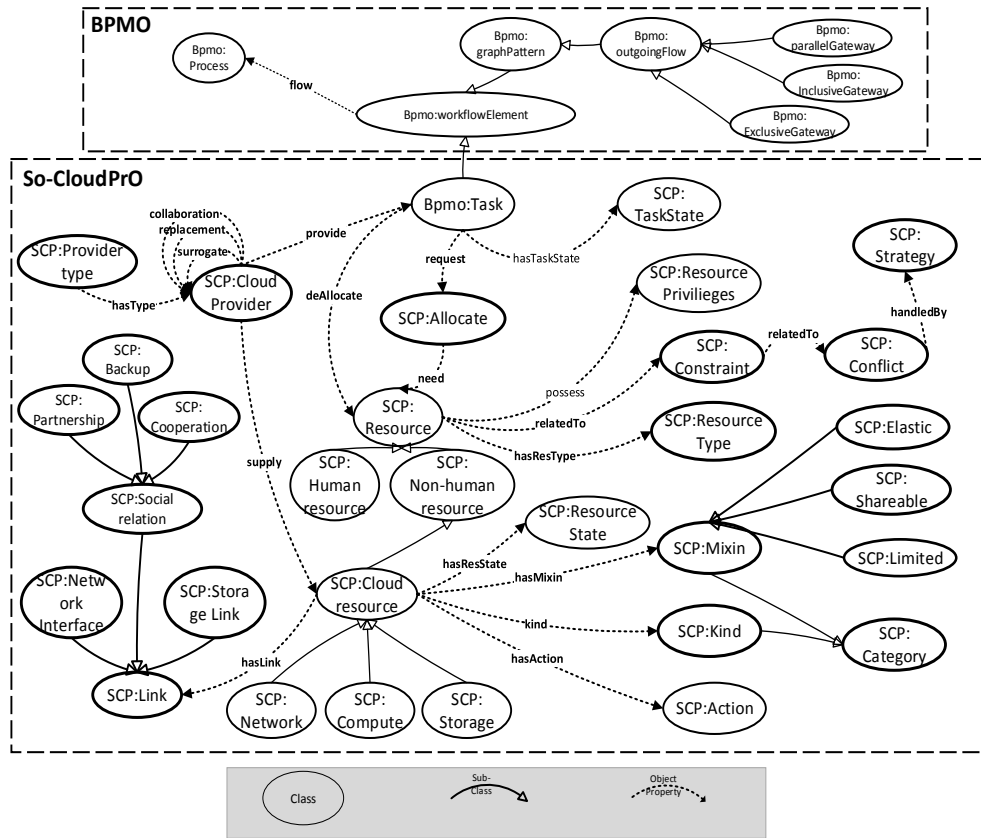
4.3.2.2 Ontology Definition

Ontology Concepts:

Since the emphasis is on resource, the main concept is *SCP:Resource* which is defined as an additional concept related to *BPMO:Task* through *SCP:Allocate*. The resource definition is reported in $C_{So-CloudPrO}$ in terms of *RDF/RDFS*².

We define two sub-concepts *SCP:Human* and *SCP:Non-human* in which *SCP:Cloud resource* is located. As mentioned before, human resources are out of the scope of this thesis, and we are particularly interested in cloud resources. Cloud resources

²OMG: Ontology Definition Metamodel, v1.1 (2014), www.omg.org/spec/ODM/1.1/PDF/

Figure 4.5: **So-CloudPrO** representation

have three sub-types: *SCP:Storage*, *SCP:Compute*, and *SCP:Network* describing the OCCI model (Figure 2.10). The **So-CloudPrO** ontology have focused on non-human resources and more specifically Cloud resources. This ontology have extended several concepts from the *CloudPrO* ontology that was previously detailed in Section 4.2.4.1. The concept referred as *SCP:resource* extends *ResourceExtension*, and *SCP:Cloudresource* extends *ComputingResource*. The relations that exist between *ComputingResources* in *CloudPrO* ontology are extended by concepts that inherits from *SCP:Link* (*SCP:NetworkInterface*, *SCP:StorageLink*, and *SCP:Socialrelation*).

As shown in Figure 4.5, *SCP:Resource* concept is linked to different other concepts. *SCP:Resource Type* means that the resource can be of type *logical* (i.e., their consumption leads into a decrease of their availability) or *physical* (i.e., their consumption does not lead to a decrease of their availability). *SCP:Constraint* depicts restrictions over resources attributes or behavior. According to OCCI core model showed in Figure 2.9, there are several concepts bonded to resource concept. Hence, we define *SCP:Mixin* concept that depicts an extension mechanism which enables to

add new resource capabilities to resource instances.

SCP:Kind concept is also specified to represent the type identification mechanism. Both latter concepts are a specialization of *Category* which refers to the basis of the type identification mechanism used by the OCCI. Besides, each cloud resource is able to invoke a set of operations defined through *SCP:Action*. *SCP:Action* has two *SCP:CloudResource* as range and domain. *SCP:Resource Privileges* represent resources state (allowed or not) regarding the execution of these actions in order to organize the work advertised to them. For instance, st_5 in our running example is allowed to be duplicated (Figure 1.4).

An important concept is also defined, referred to *SCP:Link*, which depicts relations among resources. More details about *SCP:Link* and cloud properties are presented later. Further, tasks and resources may be supplied by *SCP:Cloud Provider* that has a *SCP:Provider type* (e.g., ‘public’, or ‘private’).

To enrich BP models with resource details [175,176], we consider *resource structure* (i.e., characterization refers to resource attributes and classification refers to resource types), *work distribution* (i.e., the way resources are assigned to workitems or operations via *SCP:Action*), and *authorization* (i.e., privileges related to resources).

Ontology Attributes:

Following the OCCI infrastructure showed in Figure 2.10, cloud resources have specific attributes such as necessary memory (size) for *SCP:Storage* resources, speed for *SCP:Compute* type, and vlan identifier for *SCP:Network* resources. These attributes are stored in $A_{So-CloudPrO}$.

For instance, Listing 4.3 shows a description of a resource instance c_3 (Figure 1.4). This resource possesses a set of attributes. It is of type *compute* (Line 3), a CPU architecture of type $\times 86$ (Line 4), 2 CPU cores assigned (Line 5), a DNS hostname referred to “107.74.112.1” (Line 6), 2.4 GHz as speed (Line 7), and 1 Go as memory (Line 8), and 2.4 GHz as capacity (Line 9).

OCCI allows to link the different resource types through the *SCP:Link* class (Figure 2.10). This class has two sub-types *SCP:NetworkInterface* to link a compute instance to a *Network* instance, and *SCP:StorageLink* to link it to a *Storage* instance. These entities are depicted as concepts in our ontology (Figure 4.5). Moreover, we extend *Link* by adding other link types depicting the social relations (section 4.3.2.4). Besides, *hasResState* and *hasTaskState* are also outlined to link *SCP:Resource* and *BPMO:Task* with their current state defined in *SCP:TaskState* and *SCP:ResourceState*, respectively.

The resource state may be among the specific states specified in Figure 4.6 (i.e., ‘created’, ‘consumed’, ‘released’, etc). Likewise task state may vary among a set of predefined state which are defined in Figure 4.7 (i.e., ‘created’, ‘canceled’, ‘running’, etc). For instance, if c_3 is consumed by t_3 , then c_3 moves from ‘created’ state to ‘allocated’ state [177] (see Listing 4.3, Line 15), and a_3 from ‘initiated’ to ‘running’

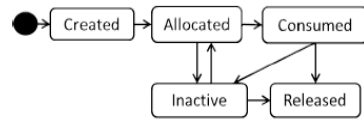


Figure 4.6: Lifecycle of resource state

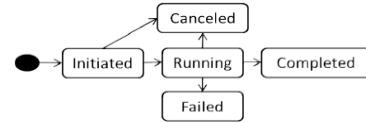


Figure 4.7: Lifecycle of task state

(see Listing 4.3, Line 27).

Listing 4.3: Resource instance description snippet

```

1 <owl:Thing rdf:about="#c3">
2   <rdf:type rdf:resource="#Compute"/>
3   <hasType rdf:datatype="xsd:string">compute</hasType>
4   <architecture rdf:datatype="xsd:integer">86</architecture>
5   <cores rdf:datatype="xsd:integer">2</cores>
6   <hostname rdf:datatype="xsd:string">107.74.112.1</hostname>
7   <speed rdf:datatype="xsd:float">2.4</speed>
8   <memory rdf:datatype="xsd:float">1</memory>
9   <hasCapacity rdf:datatype="xsd:float">2.4</hasCapacity>
10  <hasResState rdf:ResourceState="#s3"/>
11  <need rdf:Allocate="#a3"/>
12  <hasMixin rdf:Mixin="#m3"/>
13 </owl:Thing>
14 <owl:Thing rdf:about="#s3">
15   <resState rdf:datatype="xsd:string">allocated</hasResState>
16   <hasTime rdf:datatype="xsd:float">10</hasTime>
17 </owl:Thing>
18 <owl:Thing rdf:about="#a3">
19   <rdf:type rdf:Allocate="#a3"/>
20   <hasType rdf:datatype="xsd:string">compute</hasType>
21   <time rdf:datatype="xsd:float">10</time>
22   <capReq rdf:datatype="xsd:float">2.2</capReq>
23   <need rdf:Resource="#c3">
24   <request rdf:BpmoTask="#a3">
25 </owl:Thing>
26 <owl:Thing rdf:about="#t3">
27   <hasTaskState rdf:TaskState="#ts3"/>
28 </owl:Thing>
29 <owl:Thing rdf:about="#ts3">
30   <taskState rdf:datatype="xsd:string">running</hasTaskState>
31   <hasTime rdf:datatype="xsd:float">10</hasTime>
32 </owl:Thing>
33 <owl:Thing rdf:about="#m3">
34   <rdf:type rdf:Mixin="#m3"/>
35   <rdf:type rdf:Elastic="#e13"/>
36   <rdf:type rdf:Shareable="#sh3"/>
37   <rdf:type rdf:Limited="#l3"/>
38 </owl:Thing>
39 <owl:Thing rdf:about="#e13">
40   <rdf:type rdf:Elastic="#e13"/>
41   <hasElastic rdf:datatype="xsd:boolean">true</elastic>
42   <hasElasticType rdf:datatype="xsd:string">vertical</elastic>
43 </owl:Thing>
44 <owl:Thing rdf:about="#sh3">
45   <rdf:type rdf:Shareable="#sh3"/>
46   <shareable rdf:datatype="xsd:boolean">false</shareable>
47 </owl:Thing>
48 <owl:Thing rdf:about="#l3">
49   <rdf:type rdf:Limited="#l3"/>
50   <limited rdf:datatype="xsd:boolean">false</limited>
51 </owl:Thing>

```

4.3.2.3 OCCI Resource Structure

After detailing the resources, the next step is to classify them using properties. To this end, we propose to extend the *SCP:Mixin* with a set of descriptive properties as sub-concepts (i.e., *SCP:Shareable*, *SCP:Elastic*, and *SCP:Limited*). Each resource has his own set of properties.

Table 4.2: Cloud resource properties

| Types | Description |
|---------------|--|
| shareable | At least two tasks use the resource at the same time |
| non shareable | Only one task use the resource at the same time |
| limited | The resource has a maximum capacity and if it is reached the resource is no longer available |
| non limited | The resource has unlimited capacity |
| elastic | The possibility to add/remove resources to increase its capacity (vertical), or to add/remove instances of tasks with their consumed resources (horizontal) is enabled |
| non elastic | The possibility to neither add/remove resources nor instances of tasks is not enabled |

As per Table 4.2 a resource can be *shareable*, *non shareable*, *elastic*, *limited*, etc. These properties are not exclusive. For example, a resource can be *shareable* and *elastic* [122]. In our running example, st_5 has the combined properties of ‘non shareable, elastic, and non limited’. However, c_3 is neither *shareable* (see Listing 4.3, Line 37) nor *limited* (Listing 4.3, Line 45). However it may be *elastic* (Listing 4.3, Line 32) and its allowed elasticity type is ‘vertical’ (Listing 4.3, Line 33).

4.3.2.4 Resource social dependencies

In this section, we present different relations that exist between resources, and between resources and tasks. First we define relations that link tasks and resources within the same business process. Concerned entities (Figure 4.5) are defined as follows:

- *SCP:Allocate*: to bind a task to a resource with storing allocation information (i.e., type of requested resource, requested capacity, and time of allocation).
- *request*: a task requests to allocate a resource as per the properties of this resource (Table 4.2).
- *need*: an allocation’ demand is linked to a resource as per its type, i.e., logical *versus* physical.
- *deAllocate*: a task releases a resource when it stops to consume it.

Table 4.3: Description of BPs relations

| Relations | Relation Names | Description |
|----------------|-------------------------------------|---|
| Cloud resource | <i>cooperation</i> (c_i, c_j) | c_i and c_j are both allocated to the same task and they are of the same Category |
| | <i>partnership</i> (c_i, c_j) | c_i and c_j are both to the same task and they are linked through StorageLink or NetworkInterface |
| | <i>backup</i> (c_i, c_j) | c_j replaces c_i if failure of c_i and capacities of c_i and c_j are similar |
| Cloud Provider | <i>surrogate</i> (p_i, p_j) | p_j substitutes p_i if p_i is expectedly unavailable |
| | <i>replacement</i> (p_i, p_j) | p_j substitutes p_i if p_i is unexpectedly unavailable |
| | <i>collaboration</i> (p_i, p_j) | p_i and p_j are providing resources to the same task |

Returning to our example (Figure 1.4), c_3 is allocated to a_3 (see Listing 4.3, Line 11-20) at time 10 s (see Listing 4.3, Line 17) with requested capacity (*capReq*) of 2.2 GHz (see Listing 4.3, Line 18).

Second, we propose a set of social relations between cloud resources, *SCP:Social relation* as sub-concept of the *SCP:Link* (from OCCI core) along with *SCP:StorageLink* and *SCP:NetworkInterface*. For instance, a *SCP:Cooperation* relation between two resources means that both are allocated to the same task, and having the same *SCP:Category*. Besides other social relations are specified among other concepts, as *object properties*. For example, a provider can substitute another provider if this latter is unavailable in a planned way (*surrogate*) or in unexpected manner (*replacement*). Table 4.3 outlines some of these defined relations.

Listing 4.4: Resource relations snippet

```

1 <owl:Class rdf:ID="Backup">
2   <rdfs:subClassOf rdf:ID="Link"/>
3   <rdfs:range rdf:resource="#CloudResource"/>
4   <rdfs:domain rdf:resource="#CloudResource"/>
5   <hasState rdf:datatype="&xsd:boolean">true </hasState>
6 </owl:Class>
7 <owl:ObjectProperty rdf:about="#surrogate">
8   <rdfs:range rdf:resource="#CloudProvider"/>
9   <rdfs:domain rdf:resource="#CloudProvider"/>
10 </owl:ObjectProperty>

```

Listing 4.4 depicts some relations defined in *RDF/RDFS*. For instance, *SCP:Backup* relation denotes that a resource can substitute another one (Line 1-6), whereas *Surrogate* relation happens between two cloud providers (Line 7-10).

4.3.3 Resolving resource-based Conflicts using SWRL Strategies

To define the resolution strategies, we first work on instantiating entities and related constraints (Section 4.3.3.1). Then, we categorize constraints on the basis of resources and tasks (Section 4.3.3.2). These constraints mean restricting the behavior of BP entities, w.r.t SLA requirements in order to guarantee BP smooth execution. Finally, we specify conflict resolution strategies using semantic rules (Section 4.3.3.3).

4.3.3.1 Instantiation Constraints

In this section, we present the constraints linked to resource instantiations. These constraints should be implied to respect the OCCI architecture. To do so, we specify them using SWRL rules. Some examples are defined as follows.

- The instantiation of *storage* resource (*?s*) automatically implies the state ‘created’ of this resource at time *?tps*.

$$\begin{aligned} \text{Storage}(?s) \wedge \text{swrlb} : \text{date}(?tps) \rightarrow \text{hasResState}(?s, ?rs) \wedge \\ \text{resState}(?rs, \text{“created”}) \wedge \text{hasTime}(?rs, ?tps) \end{aligned} \quad (4.3)$$

- The instantiation of an allocation request which depicts a task (*?t*) that desires to consume a resource (*?r*) with a requested type (*?type*), the capacity (*?cap*) and at a time (*?tps*). This implies that if there is a resource with type and capacity that respond to the above needs, the resource is moved to allocated state.

$$\begin{aligned} \text{Allocate}(?t, ?a) \wedge \text{need}(?a, ?r) \wedge \text{hasType}(?a, ?type) \wedge \text{time} \\ (?a, ?tps) \wedge \text{capReq}(?a, ?cap) \wedge \text{hasType}(?r, ?type1) \wedge \text{swrlb} : \\ \text{equal}(?type1, ?type) \wedge \text{hasCapacity}(?r, ?cap1) \wedge \text{swrlb} : \text{greater} \\ \text{OrEqual}(?cap1, ?cap) \Rightarrow \text{hasResState}(?r, ?s) \wedge \text{resState}(?s, \\ \text{“allocated”}) \wedge \text{hasTime}(?s, ?tps) \wedge \text{hasTaskState}(?t, ?ts) \\ \wedge \text{taskState}(?ts, \text{“running”}) \wedge \text{hasTime}(?ts, ?tps) \end{aligned} \quad (4.4)$$

4.3.3.2 Resource and Task Constraints

To insure a correct use of resources, properties related to cloud environment should be considered. So, we define proper constraints that provide information about classes and properties, especially those related to resources. We categorize constraints into resource property constraints and resource relations constraints. These constraints are expressed in terms of SWRL rules in $R_{So-CloudPrO}$.

- Resource property constraints: refer to resources’ properties such as *elastic*, *limited* or *shareable*. Equation 4.5 depicts that a resource (*?r*) is *non shareable*

if there is a *Task* ($?t$) which allocates it at time $?tps$ and then, there should be no another *Task* ($?t_2$) that could allocate this resource at the same time.

$$\begin{aligned}
& hasMixin(?r, ?sh) \wedge Shareable(?sh, "false") \wedge Allocate(?t, ?a) \\
& \wedge need(?a, ?r) \wedge time(?a, ?tps) \wedge hasTaskState(?t, ?ts) \wedge taskState \\
& (?ts, "running") \wedge hasTime(?ts, ?tps) \Rightarrow (notAllocate)(?t_2, ?a_2) \\
& \wedge need(?a_2, ?r) \wedge time(?a_2, ?tps)
\end{aligned} \tag{4.5}$$

Equation 4.6 defines the vertical elasticity property. An *elastic* resource ($?r$) may apply its elasticity policy if the number of allocated resources to the concerning task ($?t$) can be increased by adding more resources such as r_2 . Finally, the cumulative capacity (cp_c) should not exceed the maximum requested capacity (cp_r).

$$\begin{aligned}
& hasMixin(?r_1, ?el) \wedge hasElastic(?el, "true") \wedge hasElasticType \\
& (?el, "vertical") \wedge Allocate(?t, ?a) \wedge need(?a, ?r_1) \wedge hasType \\
& (?a, ?type) \wedge time(?a, ?tps) \wedge capReq(?a, ?cap_r) \wedge hasCapacity \\
& (?r_1, ?cap_1) \wedge hasTaskState(?t, ?ts) \wedge taskState(?ts, "running") \\
& \wedge hasTime(?ts, ?tps) \Rightarrow Allocate(?t, ?a_2) \wedge need(?a_2, ?r_2) \\
& \wedge hasType(?a_2, ?type_2) \wedge time(?a_2, ?tps) \wedge capReq(?a_2, ?cap_r) \\
& \wedge hasCapacity(?r_2, ?cap_2) \wedge swrlb : add(?cap_c, ?cap_1, ?cap_2) \\
& \wedge swrlb : lessOrEqual(?cap_c, ?cap_r)
\end{aligned} \tag{4.6}$$

- Resource relation constraints: include resource relations such as *SCP:Cooperation* between two resources or *SCP:Backup*. The former relation is illustrated through the following example. Equation 4.7 represents how the *cooperation* of two resources is detected. r_1 can be in a *cooperation* if r_2 is consumed by the same task and are of the same type and *SCP:Category*.

$$\begin{aligned}
& hasLink(?r_1, ?l_1) \wedge hasLink(?r_2, ?l_2) \wedge Cooperation(?l_1, ?l_2) \wedge Allocate \\
& (?t, ?a) \wedge need(?a, ?r_1) \wedge hasType(?r_1, ?type) \wedge kind(?r_1, ?cat) \Rightarrow \\
& Allocate(?t, ?a_2) \wedge need(?a_2, ?r_2) \wedge hasType(?r_2, ?type) \wedge kind(?r_2, ?cat)
\end{aligned} \tag{4.7}$$

4.3.3.3 Resolution Strategies

In case of constraint violation, conflicts hampering business process completion can occur. To address these conflicts, our framework suggests strategies defined as semantic rules to guarantee resources substitution. To do so, a *conflict* is related to *resources* constraints and handled by a strategy (*Strategy*) that consists of actions. Thus, we specify semantic rules following (E)vent-(C)ondition-(A)ction structure (**On Event If Conditions Do Actions**) [178]. Events represent conflicts and conditions denote constraints. Actions suggests the *strategies* which are a set of solutions to take for resolving the conflict in question. These actions may include the elasticity feature of cloud by increasing resources capacities, or make additional resources or tasks. Many

solutions can exist for the same conflict. To formally describe our ECA-based conflict resolution strategies, we use *SWRL*. We present in Table 4.4 the SWRL descriptions referring to our rules in accordance to the running example. We explain below these strategies:

1. Strategy denoted as $S_{So-CloudPrO1}$ consists of:
 - Event: a_{11} needs c_3 which is non *shareable* and already consumed by a_3 .
 - Conditions: if there is a substitute for c_3 , in our case c_6 can replace it ($Backup(?c_6, ?c_3)$).
 - Actions: c_6 is automatically allocated to a_{11} , or c_6 can look for another substitute by comparing memory and speed of the existing resources.
2. $S_{So-CloudPrO2}$ consists of:
 - Event: the *storage* resource st_5 is not sufficient to respond to the requested capacity needed by a_4 .
 - Conditions: if a_4 is enabled to apply a vertical elasticity, or respectively a horizontal elasticity policy.
 - Actions: add two resources st_4 and st_9 and allocate them to a_4 or respectively add two instances of a_4 ($?t_{4-1}$ and $?t_{4-2}$) as well as allocate st_5 to both.
3. $S_{So-CloudPrO3}$ consists of:
 - Event: the *network* resource net_{13} , which is consumed by a_{13} , becomes out of commission.
 - Conditions: if *Backup* relation exist that means there is a substitute (net_{10}), or if a_{12} can replace a_{13} and resources st_2 and r_{12} are in *partnership* relation.
 - Actions: allocate net_{10} to a_{13} or respectively a_{12} takes over and allocate both r_{12} and st_2 .

We note that we may have several solutions to a conflict and this happens depending on the constraints that can exist in a particular case.

Table 4.4: SWRL strategies for resource conflicts resolution

| Strategy Name | Event/Conflict | Conditions/Constraints | Action/SWRL Strategy |
|---------------------|--|--|---|
| <i>SSoCloudPrO1</i> | $Allocate(?t_{11}, ?a) \wedge need(?a, ?c_3) \wedge$ $hasType(?c_3, "compute") \wedge time(?a, 13) \wedge$ $Allocate(?t_3, ?a_3) \wedge need(?a_3, ?c_3) \wedge time(?a_3, 13)$ $\wedge hasMixin(?c_3, ?st_3) \wedge$ $hasShareableType(?st_3, "false")$ $\wedge hasResState(?c_3, ?s) \wedge resState(?s, "consumed") \wedge$ $hasTime(?s, 13)$ | $hasLink(?c_6, ?l_6)$ $hasLink(?c_3, ?l_3)$ $Backup(?l_6, ?l_3) \wedge Compute(?c_6)$ $(notHasMixin(?c_6, ?l_6) \wedge swrlb :$ $equal(memory(?c_3, ?m_3),$ $memory(?c, ?m)) \wedge swrlb : equal$ $(speed(?c_3, ?s_3), speed(?c, ?c))$ | $Allocate(?t_{11}, ?a) \wedge need(?a, ?c_6) \wedge$ $hasType(?c_6, "compute") \wedge$ $time(?a, 13)$ $Allocate(?t_{11}, ?a) \wedge need(?a, ?c) \wedge$ $hasType(?c, "compute") \wedge$ $time(?a, 13)$ |
| <i>SSoCloudPrO2</i> | $Storage(?st_5) \wedge Allocate(?t_4, ?a) \wedge$ $need(?a, ?st_5) \wedge hasType(?a, "storage") \wedge$ $capReq(?a, ?cap_r) \wedge memory(?st_5, ?cap_5) \wedge$ $swrlb : lessThan(?cap_5, ?cap_r)$ | $hasMixin(?st_5, ?el_5)$ $hasElastic(?el_5, "true")$ $hasElasticType(?el_5, "vertical")$ | $Storage(?st_4) \wedge Storage(?st_9) \wedge$ $Allocate(?t_4, ?a_1) \wedge need(?a_1, ?st_4) \wedge$ $time(?a_1, 6)$ $\wedge Allocate(?t_4, ?a_2) \wedge need(?a_2, ?st_9) \wedge$ $\wedge time(?a_2, 6)$ |
| <i>SSoCloudPrO3</i> | $(notNetwork)(?net_{13}) \wedge Allocate(?t_{13}, ?a) \wedge$ $need(?a, ?net_{13})$ $\wedge time(?a, 22) \wedge hasResState(?net_{13}, ?rs) \wedge$ $resState(?rs, "consumed")$ $\wedge hasTime(?rs, 22)$ | $Network(?net_{10})$ $hasMixin(?net_{10}, ?n_{10})$ $hasMixin(?net_{13}, ?n_{13})$ $Backup(?n_{10}, ?n_{13})$ $BpmoTask(?t_{12})$ $substitution(?t_{12}, ?t_{13})$ $hasLink(?r_{12}, ?l_{12})$ $hasLink(?st_2, ?l_2)$ $Partnership(?l_{12}, ?l_2)$ | $Allocate(?t_{13}, ?a) \wedge need(?a, ?net_{10})$ $time(?a, 22)$ $Allocate(?t_{12}, ?a) \wedge need(?a, ?r_{12})$ $time(?a, 22) \wedge Allocate(?t_{12}, ?a_2) \wedge$ $need(?a_2, ?st_2)$ $time(?a_2, 22)$ |

4.4 Evaluation and validation

In this section, we present a set of proof of concepts to validate our proposals to assist the design of process models and populate a knowledge base of heterogeneous process models with cloud resources.

We implemented three proof of concepts. A validation of our proposed ontology *So-CloudPrO* is realized, then two extensions of a well known web-based process modeling tool namely *Signavio* are implemented. *Signavio* is an open source web application for developing process models in BPMN. Thus, it can support our context of cloud-based processes.

- A validation of the Cloud resource Ontology *So-CloudPrO*: We aim at validating our resource ontology developed in section 4.3.2.1 by evaluating it using the gold standard method (Section 4.4.1).
- A support of Cloud resource descriptions: This second extension aims at assisting the design of process models by allocating cloud resources to appropriate activities. Its implementation is based on the ontologies and semantic rules developed in sections 4.2.4, 4.3.2 and 4.3.3.
- An application that populates a knowledge base of cloud resources. Its implementation is based on the ontologies developed in sections 4.2.4 and 4.3.2.

4.4.1 Ontology Validation

To ensure the ontologies' quality of content, several evaluation methods have been developed (e.g., golden standard comparison [179] and ontology application [180]). The most used approach is to compare the ontology to a golden standard [181]. This method involves checking the coverage of classes defined in standards by defined concepts in ontologies. Since the fundamental goal of our work is to semantically represent and share cloud resources among process providers, we compare OCCI standard to our *S-CloudPrO* concepts.

Table 4.5: S-CloudPrO coverage to OCCI standard

| Specifications | Classes | Quantity | Coverage | Ratio |
|----------------|---|----------|----------|-------|
| Core Model | Action,Resource,Category,Kind, Mixin,Link,Entity | 7 | 7 | 100% |
| Infrastructure | Resource,Network, Compute,Storage,Link, StorageLink,NetworkLink | 7 | 7 | 100% |

To do so, we consider two specifications: core and infrastructure. The former describes the formal definition of the OCCI Core Model [61] (Figure 2.9) while the latter specifies the definition of the OCCI Infrastructure [63] (Figure 2.10) extension for the IaaS domain. Therefore, we evaluate our ontology by analyzing its coverage

to these specifications. We summarize the results in Table 4.5 to show that we cover all OCCI classes.

4.4.2 Supporting Cloud resource Descriptions

We develop an extension to *Signavio* process editor³ as a first proof of concept to integrate cloud resource description into process models. Our proof of concept can be found on line⁴.

We have added two main functionalities that are described as follows.

- *Integration of cloud resource definitions*: we have extended the latest version of BPMN (2.0) to take into account the three main types of IaaS cloud resources (i.e., compute, network and storage) as well as their attributes (e.g., speed, and hostname) following the OCCI standard (Area 1 in Figure 4.8). When designing the orange supervision process under the Signavio editor, the user can drag and drop the cloud resources needed for different activities in the process. He can also specify the different attributes and properties as defined in the OCCI standard.

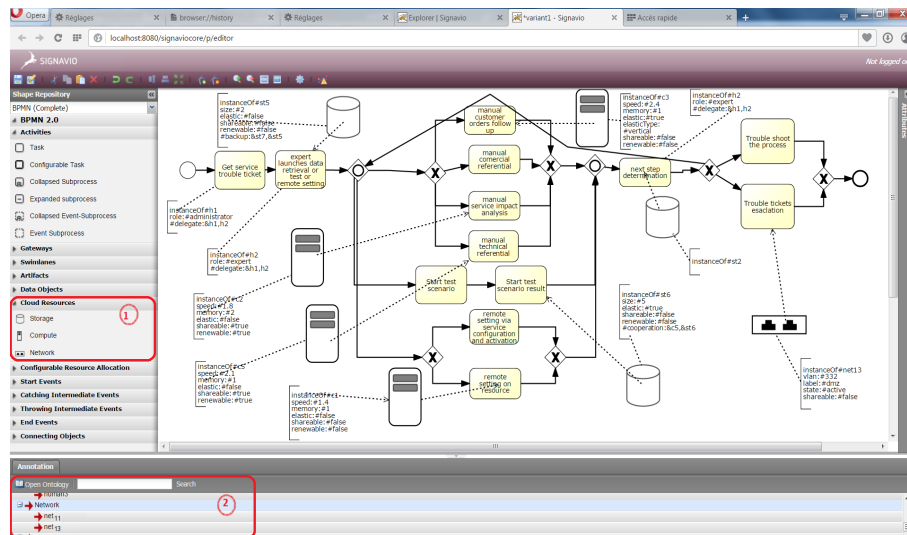


Figure 4.8: Application screenshot: Supporting resource allocation

- *Semantic description using So-CloudPrO*: This functionality allows to the user to map the process resources and their allocation to the concepts including the

³Source Code at: <https://code.google.com/p/signavio-core-components/>

⁴Description: <http://www-inf.it-sudparis.eu/SIMBAD/tools/So-CloudPrO/>

resource instances in *So-CloudPrO* (Area 2 in Figure 4.8). The **So-CloudPrO** is modeled through Protege⁵ which is an open source ontology editor and a knowledge acquisition system. On clicking on the resource, its main properties are shown. If this resource has relations with other resources, it is also indicated in the description. All these data are described through the *text annotation* bounded to resource types. Once the mapping is defined, automatic inference and defined resource constraints can be verified to ensure the correctness of the process model according to the ontology relations and their SWRL rules.

4.4.3 Cloud resource Knowledge Base

Once the process models are semantically annotated with concepts from **So-CloudPrO**, we annotate them with BPMO notations [182]. Thus, the process models become semantically equivalent at the meta-model level. For instance in our example showed in Figure 1.4, *st₅* which is assigned to the activity *a₂*, (*expert launches data retrieval or test or remote setting*) is neither *elastic*, nor *shareable* according to the appropriate text annotation. Once the annotation is finished, execution events logs are collected and processed to populate our knowledge base. Concretely, we build RDF triples to represent our base. Then, we store these triples into a triplestore database. We choose *OpenLink Virtuoso*⁶ as an open source database engine for triplestores. In fact, the addition of the access to *OpenLink Virtuoso* to Signavio is made through the API *Virtuoso Jena Provider* [183].

Once process models with cloud resource descriptions have been stored in the knowledge base, cloud users (e.g., organizations) can retrieve and manipulate information through SPARQL queries using SPARQL endpoint supplied by *OpenLink Virtuoso*. Also, these tenants may define links between cloud resource descriptions at this point in order to share existing resources and thus enable interoperability between organizations. Moreover, in case of conflicts, SWRL queries are applied to resolve them according to the specified strategies. Our proof of concept can be found on line⁷ as a step that complements the previous approach.

4.5 Conclusion

In this chapter, we answered the two questions mentioned in our thesis problematic (see section 1.2.1) which are: *How to take into account the specificities of cloud resources into BPM?*, and *How to enrich business process models with social technologies?*

To take into consideration the specificities of cloud resources into BPM, we have studied deeper the structure of cloud resources from the API OCCI and transfer

⁵<http://protege.stanford.edu/>

⁶<http://virtuoso.openlinksw.com/>

⁷<http://www-inf.it-sudparis.eu/SIMBAD/tools/So-CloudPrO/>

it through the extension of BPMO using our proposed CloudPrO ontology. The obtained *So-CloudPrO* provides formal semantic definition using RDF/RDFS format with covering the totality of OCCI concepts.

To enrich business process models with social technologies, we adopt the use of social dependencies to define a set of relations among cloud resources and cloud providers. These dependencies are useful to resolve resource-based conflicts using SWRL rules.

In this second work of our thesis, our principles presented in Section 1.4.1 are respected:

- Heterogeneous data modeling: We propose to use a unified description of Cloud resources, using an extension of BPMO ontology, to annotate the process models.
- Exploitation of knowledge: We propose a framework that allows business process providers to be aware of the characteristics of its allocated Cloud resources and relations between them.

Configurable Resource Allocation for Multi-tenant Process Development in the Cloud

Contents

| | | |
|------------|---|------------|
| 5.1 | Introduction | 92 |
| 5.2 | Configurable Cloud Resource Allocation | 93 |
| 5.2.1 | Configurable Resource Assignment Operator | 93 |
| 5.2.2 | Configurable Resource Elasticity Operator | 95 |
| 5.2.3 | Configurable Resource Sharing/Batching Operator | 97 |
| 5.3 | Optimization of Cloud resource allocation using Genetic Algorithms | 98 |
| 5.3.1 | Motivating Example | 99 |
| 5.3.2 | Cloud-specific Resource Properties (CRP) Problem | 102 |
| 5.3.3 | Problem Formalization using Genetic Algorithm | 106 |
| 5.3.3.1 | Genome Encoding | 106 |
| 5.3.3.2 | Fitness Function considering Green properties | 108 |
| 5.3.3.3 | Fitness Function considering QoS properties | 110 |
| 5.4 | Evaluation and validation | 111 |
| 5.4.1 | Supporting Configurable Resource Allocation | 112 |
| 5.4.2 | Experimentation | 113 |
| 5.4.2.1 | Structural Complexity experiments | 113 |
| 5.4.2.2 | Solving optimal cloud resource allocation | 115 |
| 5.5 | Threats to Validity | 118 |
| 5.6 | Conclusion | 118 |

5.1 Introduction

This chapter presents the main contribution of our thesis which addresses the lack of support for cloud-specific resource configuration where different allocation alternatives need to be explicitly defined. To this end, we introduce our approach for supporting configurable resource allocation for multi-tenant process development in the Cloud.

As mentioned in the Chapter 1, models and mechanisms for specifying the control-flow perspective of configurable processes are well defined unlike concepts in the resource perspective. Indeed, different approaches for configurable process modeling have been proposed so far, mainly with a focus on configuring the control-flow [38]. Even though the concept of configurable process models is highly complementary to cloud computing, there has been hardly any uptake in that area. The problem is apparently that specifics of cloud computing, specifically in how resources can be configured and integrated, are hardly considered in configurable process modeling. The few proposals on extending configuration to resources [29, 30, 43] do not cover Cloud features such as elasticity or multi-tenancy and focus on human resources and their dependencies [41, 42, 45].

In this chapter, we propose process configuration modeling mechanisms for cloud computing. More specifically, we define a novel approach for modeling configurable processes with configurable cloud resource allocation operators that allow to explicitly model resource allocation alternatives in multi-tenant process models. Our model takes into account two main Cloud features that are elasticity and shareability. Afterwards, in order to select the optimal resource allocation alternative or variant we propose an approach that, using a genetic algorithm, permits the extraction of the configuration that best fits to the tenant requirements. These requirements are divided into two types (i) the requirements related to Cloud features i.e., elasticity and shareability, and (ii) the requirements related to the quality of service (QoS).

We start the chapter by presenting our approach for supporting configurable Cloud resource allocation in configurable process models. To do so, we explain in details our proposed configurable resource operators that take into account the Cloud features i.e., elasticity and shareability (Section 5.2). Thereafter, we present our genetic-based approach that aims at selecting optimal cloud resource configuration allocation w.r.t Cloud resource properties related to the Cloud features, and process non functional properties associated to the QoS characteristics (Section 5.3). Next, we present our associated validation and experiments in order to show the effectiveness of our approaches (Section 5.4). Afterwards, we show some threads to validity that arise (Section 5.5). Finally, we conclude the chapter (Section 5.6).

The work in this chapter was published in conference proceedings [164, 184].

5.2 Configurable Cloud Resource Allocation

In this section, we present our *configurable cloud resource allocation approach* for multi-tenant business processes development. The allocation of cloud resources takes into account two main parameters: (1) the desired resources and their properties and (2) the desired resource behavior. Therefore, we identify three main operators related to the configuration of the resource properties and behavior: (i) configurable resource assignment operator denoted as A^c (detailed in Section 5.2.1), (ii) configurable resource elasticity operator denoted as E^c (detailed in Section 5.2.2) and (ii) configurable resource sharing/batching operator denoted as $(S/B)^c$ (detailed in Section 5.2.3). An excerpt of a configurable process model with the configurable resource allocation operators is depicted in Figure 5.1 and is explained in the following sections.

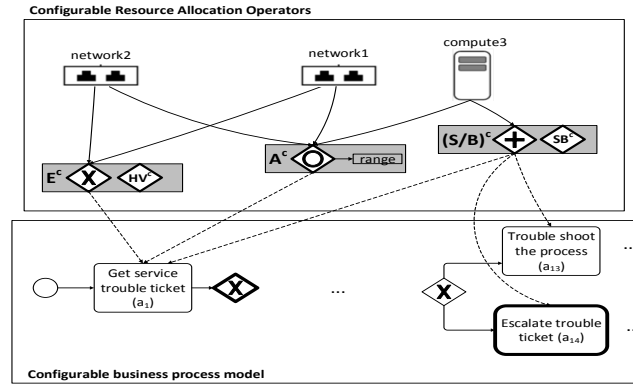


Figure 5.1: Configurable resource allocation operators

5.2.1 Configurable Resource Assignment Operator

The configurable resource assignment operator A^c allows the modeling of a *variable number of resources allocated to a specific activity*. For instance, in our running examples in Figure 1.6 and Figure 1.7, the activity a_1 needs either (i) a network resource “network1” and a compute resource “compute3” or (ii) a network resource “network2” and a compute resource “compute3”. Therefore, through A^c , we model a design-time choice in the configurable process that allows the tenants to select one of the available options. To do so, we define two main parameters for A^c : (i) a configurable type and (ii) a range (see A^c in Figure 5.1).

The configurable type can be either a configurable OR (OR^c), a configurable AND (AND^c) or a configurable XOR (XOR^c). These connectors have the same behavior as the configurable control flow connectors (see Section 2.1.2). A configurable type allows to model the behavior of the resource assignment. It is configured in the same way as the configurable connectors of the control-flow perspective (see Section 2.1.2), i.e. OR^c , AND^c and XOR^c can change their types while preserving their behaviors

| Parameters | | Configuration constraints |
|-------------------|-----------|---------------------------|
| Configurable type | OR^c | refer to Table 2.1 |
| | AND^c | refer to Table 2.1 |
| | XOR^c | refer to Table 2.1 |
| Range | $range_C$ | $min = 0, max = R_C $ |
| | $range_N$ | $min = 0, max = R_N $ |
| | $range_S$ | $min = 0, max = R_S $ |

Table 5.1: Configurable assignment parameters and configuration constraints

(see Table 2.1) and/or restrict the number of allocated resources. In our example in Figure 5.1, the activity a_1 is connected to the cloud resources “network2”, “network1” and “compute3” through an OR^c . A tenant may configure the OR^c to an XOR associated to “network2” and “network1” in order to specify that either “network2” or “network1” can be allocated to a_1 while “compute3” is not needed. The allocation decision between “network1” and “network2” is therefore left to the run-time depending on the runtime environment requirements, availability of the resources, etc.

The second operator parameter (i.e. range) imposes an additional constraint on the configuration choice. It is specified by the cloud process provider as a configuration guideline for the tenants. A range specifies the minimal and maximal number of the resources that are recommended to be allocated from each type ($range_C$ for compute, $range_N$ for network and $range_S$ for storage). For instance, a cloud process provider recommends that at least one compute and one network resources are allocated to the activity a_1 . This corresponds to set the minimum of $range_C$ and $range_N$ to 1. By default, the range minimum is set to 0 and the range maximum is set to the total number of the resources from a specific type. The configuration of the connectors in the configurable type should respect these constraints. For example, having the minimum $min_{range_C} = 1$ and $min_{range_N} = 1$, the aforementioned configuration of the OR^c to an XOR associated to the resources “network1”, “network2”, and “compute3” is not valid. The cause is when we have an XOR means that either we have the $min_{range_C} = 1$ or $min_{range_N} = 1$ but not at the same time.

Table 5.1 summarizes the configurable resource assignment parameters and their configuration constraints. The configurable type follows the configurable connectors from the control flow perspective. Its configuration constraints are the same as presented in Table 2.1. Each of the range parameters has a minimum min (set by default to 0) and a maximum max . We denote by $|R_C|$, $|R_N|$ and $|R_S|$ the number of compute, network and storage resources respectively provided for a specific activity. For instance, in order to derive the resources allocated to the activity a_1 in the process variant in Figure 1.6, the configurable resource assignment operator in the process in Figure 5.1 is configured as following:

- the configurable type OR^c is configured to an AND associated to the resources

$network_1$ and $compute_3$;

- This configuration does not violate the range that we suppose defined by the cloud process provider as follows: $range_C$ ($min = 1, max = 2$); $range_N$ ($min = 1, max = 1$); $range_S$ ($min = 0, max = 0$).

The resource assignment operator is the main operator in the configurable resource allocation modeling. It allows to define the pool of resources that may be allocated to the process activities. Once it is specified, the configurable sharing/batching (see Section 5.2.3) and the configurable elasticity (see Section 5.2.2) operators can be used to model the resource shareability and elasticity properties.

5.2.2 Configurable Resource Elasticity Operator

The Cloud infrastructure provides two types of elasticity, vertical and horizontal, in order to take into account the run-time workload. The *vertical elasticity* is the possibility to scale up and down by adding or removing resources to an existing activity in order to increase its capacity. The *horizontal elasticity* is the possibility of adding or removing instances of activities with their consumed resources.

During resource allocation, an organization may have different requirements regarding the anticipation of its activities workload, and thus may request different elasticity configurations. For instance, in a specific organization, an activity may require a network resource of at least 100 *Mbit/s* but may go to 600 *Mbit/s* during pick hours. At allocation time, a network resource of size 100 *Mbit/s* which can scale up to a 600 *Mbit/s* if vertical elasticity is selected. In a second organization, the same activity requires a network resource of at least 100 *Mbit/s* but may go to a maximum of 150 *Mbit/s*. The organization requests an horizontal elasticity that adds activities' instances to reach a 150 *Mbit/s*.

In order to model the variability at the elasticity level, our proposed *configurable resource elasticity* operator E^c takes into account two configuration parameters: (i) the set of resources to be elastic and (ii) the way they scale up and down (i.e. elasticity type). Table 5.2 summarizes the configurable resource elasticity parameters and configuration constraints. Similarly to the *configurable resource assignment* and *configurable resource sharing/batching* operators, the first parameter (configurable type) can be either an OR^c , XOR^c or AND^c and is used to model the number of resources to be elastic. For instance in our example in Figure 5.1, either “network1” or “network2” can be elastic (they are connected through an XOR^c). An organization may configure the XOR^c to a “sequence” associated to “network2” in order to specify that only “network2” can be elastic. The second parameter (configurable elasticity type) specifies the elasticity behavior. Four elasticity types: (i) H (i.e. horizontal), (ii) V (i.e. vertical), (iii) HV (i.e. hybrid) and (iv) HV^c (i.e. configurable hybrid) are defined from which only HV^c is configurable and can be configured to H , V or HV .

| Parameters | | Configuration constraints |
|------------------------------|---------|---------------------------|
| Configurable type | OR^c | refer to Table 2.1 |
| | AND^c | refer to Table 2.1 |
| | XOR^c | refer to Table 2.1 |
| Configurable elasticity type | HV^c | H, V, HV |
| | V | - |
| | H | - |
| | HV | - |

Table 5.2: Configurable elasticity parameters and configuration constraints

We note that the elasticity in Cloud computing is bounded in practice due to many causes such as outages, network bottlenecks, etc however it is not true in theory [185]. We also notice that the use of *configurable resource elasticity* operator E^c automatically involves the use of *configurable resource assignment* operator A^c . Besides the configuration constraints in Table 5.2, additional configuration guidelines can be specified by the cloud provider regarding the configuration of the elasticity type. These guidelines assist the tenants for selecting the right configuration of the configurable HV^c . They are derived according to the maximal capacity that is ensured by the cloud provider during the scale up (vertical or horizontal elasticity) which is specified in the Service Level Agreement (SLA) [186]. In case a resource has a configurable resource elasticity type HV^c , two parameters C_H and C_V are specified in the SLA which correspond to the maximal capacities ensured by the cloud provider if the configurations H or V are selected respectively. C_H corresponds to the maximal ensured capacity by adding activities' instances while C_V corresponds to the maximal ensured capacity by adding resources' instances to increase the activity capacity. We denote by C_a the maximal capacity that may be required by an activity "a" for a tenant process variant. The configuration guidelines of the configurable HV^c type are defined in Equation (5.1).

$$HV^c = \begin{cases} H & \text{if } C_a \leq C_H \wedge C_H = \min(C_H, C_V) \\ V & \text{if } C_a \leq C_V \wedge C_V = \min(C_H, C_V) \\ HV & \text{if } (C_a > C_H \wedge C_a > C_V) \wedge (C_a \leq C_V + C_H) \end{cases} \quad (5.1)$$

where $\min(C_H, C_V)$ returns the minimal capacity. The configuration H is recommended to the tenant in case (1) the maximal capacity required by its activity is less than or equal to the maximal capacity ensured by the cloud provider in the horizontal elasticity and (2) the capacity of the horizontal elasticity is less than that of the vertical elasticity. Respectively, the configuration V is recommended in case the same conditions are valid for C_V . The configuration HV is recommended in case C_a is greater than C_V and C_H but is less than or equal to their sum. For example, suppose that a tenant specifies that the activity a_1 in the process in Figure 5.1 requires a maximal capacity of 100 *uc* (i.e. $C_a = 100$ *unit-of-capacity* where *unit-of-capacity* can be a storage, compute or network related units). The cloud provider

specifies that a maximal capacity of $200 uc$ can be ensured for the vertical elasticity (i.e. $C_V = 200 uc$) and a maximal capacity of $150 uc$ can be ensured for the horizontal elasticity (i.e. $C_H = 150 uc$). Since, $C_a \leq C_H$, $C_a \leq C_V$ and $C_a \leq C_V + C_H$ then H , V and HV are potential configurations. However, as C_H is the minimal ensured capacity, the configuration H is recommended.

5.2.3 Configurable Resource Sharing/Batching Operator

The resource shareability represents one of the important features in cloud environments. According to security, availability and scalability issues in the process, an allocated resource may or may not be shareable between multiple activities, between multiple instances of the same activity or both. A resource shared between multiple activities is referred to as *shareable* and can be consumed by more than one activity instance at the same time within the same process instance [69]. A resource shared between multiple instances of the same activity is referred to as *batch* and can be utilized by multiple instances of the same activity within multiple process instances [86, 90]. A *hybrid* resource is shareable and batch.

As different tenants sharing the configurable process may have different requirements, the shareability of a resource should account for variability. For instance, in our running examples in Figure 1.6 and Figure 1.7, the resource “compute3” is shareable between multiple instances of two activities in the first process (a_1 and a_{13}) (i.e. it is shared and batch) while it is shared between three activities in the second process (a_1 , a_{13} and a_{14}). Therefore, we define the *configurable resource sharing* operator denoted as $(S/B)^c$ which allows to model the variability according to (i) the number of instances/activities sharing the corresponding resource and (ii) the way the activities share this resource (i.e. in a shareable, batch or hybrid manner) (see $(S/B)^c$ operator in Figure 5.1).

Table 5.3 summarizes the configurable resource sharing/batching parameters and their configuration constraints. The first parameter (configurable type) is similar to the configurable type in the configurable resource assignment operator. It can be either an OR^c , AND^c or XOR^c and allows to model the behavior of the resource shareability. Referring to our example in Figure 5.1, an AND^c is used to connect the resource “compute3” to the activities a_1 , a_{13} and a_{14} . Since an AND^c can be only configured to an AND with possible restricted branches, one can configure the type by selecting only a subset of the activities to share the corresponding resource. For example, the AND^c can be configured to an AND associated to a_1 and a_{13} in order to specify that only a_1 and a_{13} may share “compute3”. The second parameter (configurable shareability type) allows to define the way the activities share the resource. Four shareability types: (i) SB (i.e. hybrid), (ii) S (i.e. shareable), (iii) B (i.e. batch) and (iv) SB^c (configurable hybrid) are defined from which only SB^c is configurable and can be configured to S , B or SB .

For instance, in order to derive the shareability configuration of the resource “com-

| Parameters | | Configuration constraints |
|--------------------------------|---------|---------------------------|
| Configurable type | OR^c | refer to Table 2.1 |
| | AND^c | refer to Table 2.1 |
| | XOR^c | refer to Table 2.1 |
| Configurable shareability type | SB^c | S, B, SB |
| | S | - |
| | B | - |
| | SB | - |

Table 5.3: Configurable sharing/batching parameters and configuration constraints

pute3” in the process variant in Figure 1.6, the configurable shareability operator in Figure 5.1 is configured as follows:

- AND^c is configured to an AND associated to a_1 and a_{13} ;
- The configurable shreability type SB^c is configured to a SB (as “compute3” in Figure 1.6 is hybrid, i.e. it is shared between multiple instances and activities)

We note that the use of *configurable resource sharing* operator $(S/B)^c$ automatically implies the use of *configurable resource assignment* operator A^c .

5.3 Optimization of Cloud resource allocation using Genetic Algorithms

As observed before, there is a clear need of managing cloud resource allocation in configurable process models. The configuration, the integration, and the optimization of such resources are hardly taken into consideration in such process models. The existing approaches [5,29,30,42] which have extended configuration to resources do not take into account the cloud resource properties that depict requirements in terms of elasticity and shareability. For instance, the process provider defines that an activity needs an elastic storage resource and requires that this resource cannot be shared with other activities. Furthermore, the previous proposals in such context do not take into consideration the QoS properties of business processes. Hence, the tenant needs to be assisted to optimize the according requirements e.g., response time or cost.

While optimizing resource allocation has been studied in the last years, the existing proposals [162] have not tackled the problem considering both Cloud and QoS perspectives. On the one hand, some works [28,83,84] consider only cloud resource scheduling and neglect the QoS properties. On the other hand, other researches [147–149] consider QoS properties to reduce costs while omitting cloud features such as elasticity and shareability that should be properly handled to ensure a correct resource allocation.

Therefore, we build upon the configurable process model enriched with the configurable resource operators that is developed in our previous approach (Section 5.2) to reach the next step which is to generate the process variants. Applying process individualization phase on such model is a tedious and complicated task. Therefore, we choose to use genetic algorithms as a suitable technique to deal with such combinatorial problem. Our objective is to alleviate this task by selecting the optimal process variant that involves the best resource alternative which best fits to the process provider's requirements. These requirements take into account two types: (i) the Cloud features i.e., elasticity and shareability, and (ii) the QoS properties including the ecological properties.

We start by presenting a motivating example used throughout the approach (Section 5.3.1). Then, we introduce our model for optimal Cloud resource allocation (Section 5.3.2). Next, we formalize our problem according to the steps of genetic algorithm (Section 5.3.3).

5.3.1 Motivating Example

Our example is the same configurable service supervision process shown in Figure 1.5 but with its potential Cloud resources to be allocated. This allocation refers to the link between process activities and resources using our proposed configurable resource operators (defined in Section 5.2). It is depicted in Figure 5.2. For instance, the resource s_9 is allocated to the activity a_{13} through the Assignment operator A^c (Figure 5.2). After, we take two process variant examples generated from this latter process which is shared between Orange affiliates.

Once an affiliate decides to use this process, he can fine-tune the process according to his requirements in terms of control-flow elements at first then in terms of resources related to the active activities. Thus, for each activity in the process variant, the affiliate expresses its needs in terms of Cloud resource allocation and according to them he configures the configurable resource operators (i.e., A^c , E^c , and S/B^c). These requirements reflect the desired properties for a resource in terms of elasticity and shareability. For instance, the tenant may require for a_1 : vertical elastic and shared compute resources, and vertical elastic and not shared network resources. In accordance with the tenant requirements, we notice that the variant 1 depicted in Figure 1.6 is acceptable since the resource parameters related to the concerned activity conforms these needs. Nevertheless, the variant 2, shown in Figure 1.7, is not acceptable because it does not fulfill the requirements since the network resource $n1$, that is allocated to a_1 , could be shared according to its *resource behavior*.

It is possible to have more variants beside the variant 1 that can also fit to the tenant requirements and are acceptable. The choice between these "acceptable" variants can be made according to the QoS requirements. More details about the QoS properties are presented in sections 5.3.3.2 and 5.3.3.3.

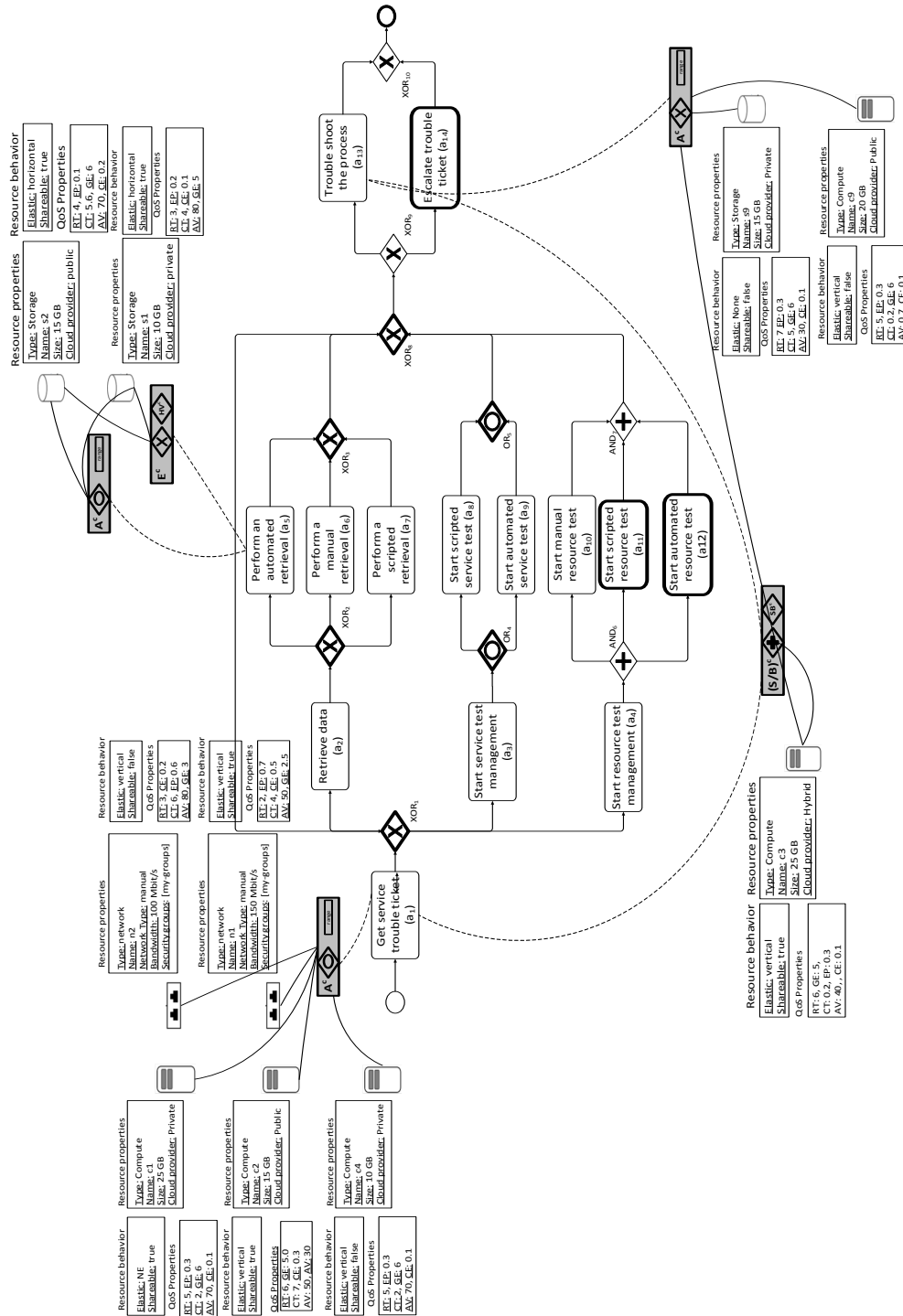


Figure 5.2: A configurable service supervision process with configurable resource operators

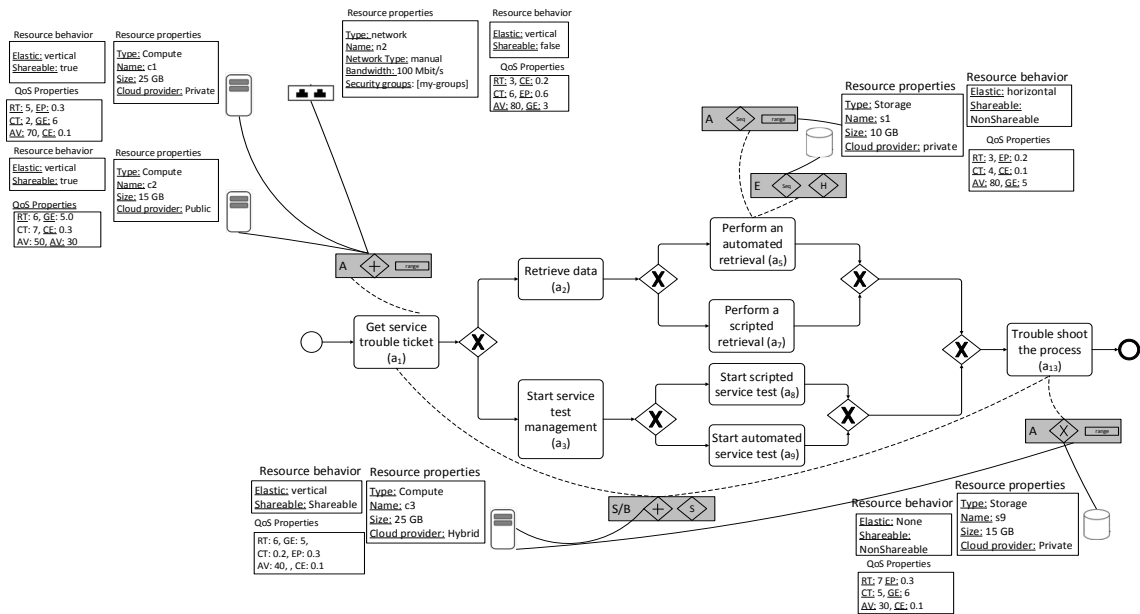


Figure 5.3: Variant 1: A process derived from the configurable process in Figure 5.2 with its allocated cloud resources

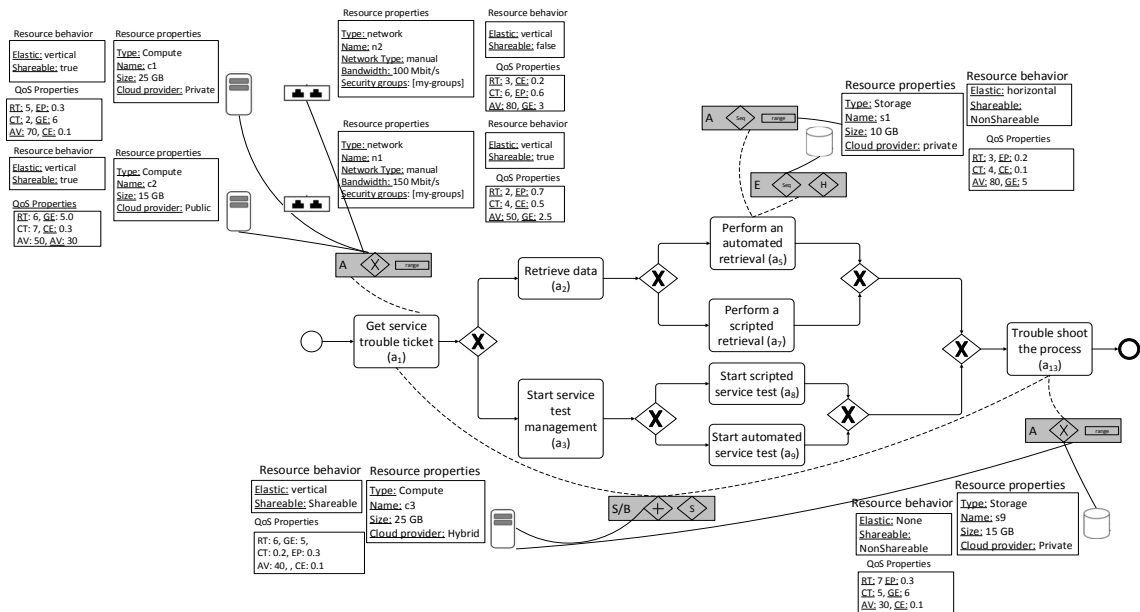


Figure 5.4: Variant 2: A process derived from the configurable process in Figure 5.2 with its allocated cloud resources

Therefore, the challenge of this approach is twofold: (1) to ensure the selection of acceptable variants that fit to the tenant requirements in terms of elasticity and

shareability, and then (2) to look for the one that best optimizes the QoS properties.

5.3.2 Cloud-specific Resource Properties (CRP) Problem

In this section, we formalize our optimization problem that aims at selecting the optimal cloud resource allocation in configurable process models. To do so, we define the set of requirements related to cloud-specific resource properties (CRP). As mentioned before, in a cloud context, resources are characterized by their dynamic and flexible behavior. To ensure a correct resource configuration, we identify process variants that fit to a tenant's requirements in terms of elasticity and shareability referred to as CRP requirements. To do so, we formally define the cloud resources, available for a configurable business process, including states regarding the elasticity and shareability (Definition 5.3.1). Then, we specify the configurable resource operators (Definition 5.3.2). Afterwards, we present a definition of a configurable business process from a resource perspective (Definition 5.3.3).

Definition 5.3.1 (Cloud resource). *A cloud resource is a tuple denoted as $CR = (id, T, RES, RSS)$ where:*

- *id is its unique identifier;*
- *T is the type of the cloud resource whose possible values are {compute, storage, network};*
- *RES (Resource Elastic State) is the state of the resource regarding the elasticity property that is provided from the resource operator E^c . The possible values are {none, vertical, horizontal};*
- *RSS (Resource Shareability State) is the state of the resource regarding the shareability property that is provided from the resource operator $(S/B)^c$. The possible values are {nonshareable, shareable, batch};*

Each resource is characterized by a set of attributes. For instance, the resource c_4 which is linked to a_1 represents a tuple CR where $id = c_4$, $T = \{compute\}$, $RES = \{vertical\}$, and $RSS = \{shareable\}$ (Figure 5.2).

Definition 5.3.2 (Configurable Cloud resource operator). *A configurable cloud resource operator is denoted as $RO = (id, OA, OR, range, CT, PT)$ where:*

- *id is its unique identifier;*
- *$OA \subseteq A$ is a set of business process activities assigned to RO ;*
- *OR is a set of cloud resources allocated to RO ;*
- *range is the maximum number of assigned resources to OA ;*

- CT is the configurable type where $CT = \{OR^c, XOR^c, AND^c\}$
- PT is the cloud property type. In case of A^c operator, $PT = \{none\}$, if E^c operator then $PT = \{HV^c\}$, or if S/B^c operator then $PT = \{SB^c\}$.

For example, the activity a_5 in Figure 5.2 is linked to the operator E^c in order to consume two resources s_1 and s_2 . This operator is defined as a tuple RO where $id = 2$, $OA = \{a_5\}$, $range = 5$, $OR = \{s_2, s_1\}$, $CT = \{XOR^c\}$, and $PT = \{HV^c\}$.

Definition 5.3.3 (Configurable business process (BP)). *A configurable business process is a process graph as defined in Definition 2.1.5 with adding our configurable resource operators. It is denoted as CBP and it depicts a tuple $CBP = (id, N, E, T, L, I, CR_p, RO_p, I, K)$ where:*

- id, N, E, T, L, I are as defined in Definition 2.1.5;
- CR_p is the set of cloud resources;
- RO_p is the set of configurable resource operators;
- $I \subseteq CR_p \times RO_p$ is the set of edges connecting cloud resources with configurable operators;
- $K \subseteq RO_p \times A$ is the set of edges connecting activities with configurable cloud resource operators;

Considering the configurable BP (Figure 5.2), it can be defined as a tuple denoted as CBP where: $id = 1$, $A = \{a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, a_9, a_{10}, a_{11}, a_{12}, a_{13}, a_{14}\}$, $CR_p = \{c_2, c_4, n_1, n_2, s_1, s_2, c_3, s_9\}$, and $RO_p = \{A_1^c, (S/B)_2^c, A_3^c, A_4^c, E_5^c\}$.

Next, we specify the CRP requirements that are expressed by process tenant. These CRP requirements are defined per BP activity and refer to the accepted states of the consumed cloud resources (Definition 5.3.4).

Definition 5.3.4 (CRP requirements). *The CRP requirements can be represented as a tuple $CRP = (id_A, RRT, AES, ASS, Cr)$ that depicts the requested needs by the tenant per activity, where:*

- id_A is the identifier of the activity;
- RRT (Requirement Resource Type) is the resource type where $RRT = \{compute, network, storage\}$;
- $AES \in CR.RES$ (Acceptable Elastic State) is the required state regarding the elasticity;
- $ASS \in CR.RSS$ (Acceptable Sharability State) is the required state regarding the shareability;

Table 5.4: CRP requirements

| id_A | a_1 | | | a_5 | | | a_{13} | | |
|--------|-----------|--------------|--------------|--------------|-----------|--------------|-----------|--------------|--------------|
| | CRP^c | CRP^n | CRP^s | CRP^s | CRP^n | CRP^c | CRP^c | CRP^s | CRP^n |
| RRT | compute | network | storage | storage | network | compute | compute | storage | network |
| AES | Vertical | Vertical | none | Horizontal | none | none | Vertical | None | None |
| ASS | Shareable | nonShareable | nonShareable | nonShareable | Shareable | nonShareable | Shareable | nonShareable | nonShareable |
| Cr | 10GB | 100Mbit/s | none | 20GB | 110Mbit/s | none | 15GB | 10GB | none |

- Cr is the requested capacity.

The tenant defines its CRP requirements per activity. Each activity may have three CRP requirements per resource type i.e., CRP^c for compute, CRP^s for storage, and CRP^n for network resources. Table 5.4 depicts for each activity a_1 , a_5 , and a_{13} the according tenant requirements. For instance, a_5 requires horizontally elastic and non shareable storage resources, and non elastic and shareable network resources. We note that a_5 could have two storage resources where the sum of their capacities do not exceed 20GB.

Algorithm 1 Generation of resource process variants

Require: CBP

Ensure: RAV

```

1:  $O \leftarrow GetresourceOperators(RO)$            ▷ Extract the configurable resource operators.
2: for all  $o$  in  $O$  do                               ▷ Iterate for each operator in  $O$ .
3:    $OR \leftarrow GetAllocatedResourceSet(o)$        ▷ Get the allocated resources to the operator.
4:   if  $CanConfiguredTo(o, \wedge)$  then             ▷ Check if the operator can be configured to  $\wedge$ .
5:      $R \leftarrow NewResourceSet()$                ▷ Initial an empty resource set.
6:     for each  $r$  in  $OR$  add  $r$  to  $R$              ▷ Add all allocated resources to  $R$ .
7:     add  $R$  to  $RAV$                                ▷ Add a set of allocated resources to  $RAV$ .
8:   end if
9:   if  $CanConfiguredTo(o, \times)$  then           ▷ Check if the operator can be configured to  $\times$ .
10:    for each  $r$  in  $OR$  add  $r$  to  $RAV$          ▷ Add individual allocated resource to  $RAV$ .
11:  end if
12:  if  $CanConfiguredTo(o, \vee)$  then             ▷ Check if the operator can configure to  $\vee$ .
13:     $i = 0$ 
14:    while  $i < SizeOf(OR)$  do
15:       $i = i + 1$ 
16:       $R \leftarrow combine(OR, i)$            ▷  $combine$  is a function that returns a set of allocated
resources following the logic of  $\vee$  operator
17:      add  $R$  to  $RAV$ 
18:    end while
19:  end if
20: end for

```

By configuring the resource operators, we may obtain several possible variants for

resource allocation for each activity. These variants may fit to the CRP requirements (i.e., acceptable) or not (i.e., not acceptable). Since our aim is to look for the best resource allocation variant, we only select at this stage the acceptable variants.

Algorithm 1 allows to generate all the possible resource variants from a configurable business process according to the behavior of the resource operators. To do so, our algorithm input is the configurable BP (CBP) that includes the used configurable resource operators (RO) and the pool of cloud resources (CR_p). First, we extract the configurable resource operators (i.e., OR^c , AND^c , or XOR^c) (Line 1). Then, based on RO behavior we select the appropriate allocated resources and store the according variant to RAV (Line 4-19). For example, OR^c can be configured to AND (\wedge), XOR (\times), and OR (\vee), thus it yields all conditions (line 4, 9, and 12). Finally the algorithm output is the set of the possible resource process variants stored in RAV .

Algorithm 2 Acceptable resource process variant

Require: RAV, CRP

Ensure: RAV

```

1: for all  $rav$  in  $RAV$  do                                ▷ Iterate for each resource allocation variant in  $RAV$ .
2:   for all  $r$  in  $rav$  do                                    ▷ Iterate for each resource in  $rav$ .
3:      $RES_r \leftarrow GetRES(r)$                              ▷ Get resource RES.
4:      $RSS_r \leftarrow GetRSS(r)$                              ▷ Get resource RSS.
5:      $O_r \leftarrow GetAssignedOperators(r)$  ▷ Get a set of operators assigned with resource
      $r$ .
6:     for all  $o$  in  $O_r$  do                                    ▷ Iterate for each operator in  $O_r$ .
7:        $a \leftarrow GetAssignedActivity(o)$                     ▷ Get an activity assigned with operator  $o$ 
8:        $AES_a \leftarrow GetAES(a)$                              ▷ Get activity AES.
9:        $ASS_a \leftarrow GetASS(a)$                              ▷ Get activity ASS.
10:      if  $RES_r \neq AES_a \vee RSS_r \neq ASS_a \vee C_r > C_a$  then ▷
      Verify whether the CRP properties ( $RES_r, RSS_r, C_r$ ) of a resource are compliant with
      properties ( $AES_a, ASS_a, C_a$ ) of its assigned activity.
11:        remove  $rav$  from  $RAV$  ▷ Remove the non-compliance resource variant  $rav$ .
12:        break
13:      end if
14:    end for
15:  end for
16: end for
17: return  $RAV$                                              ▷ Return accepted resource allocation variant.

```

Afterwards, algorithm 2 takes as input these obtained variants (RAV) and aims at selecting, from them, the acceptable ones that fit the CRP requirements. To this end, the algorithm inputs are the resource process variants (RAV) and the cloud requirements (CRP). We first extract for each variant the allocated resources and then check if the resource properties (Line 2-4) match with the CRP requirements provided by the assigned activity (Line 6-10). After we remove the variants that are not compliant (Line 11). Finally, we obtain the acceptable resource process variants

(RAV).

Once the acceptable variants are selected, the next step consists in assisting the tenant in order to get a process variant satisfying QoS constraints using a genetic algorithm (section 5.3.3).

5.3.3 Problem Formalization using Genetic Algorithm

At this stage, we use a genetic-based algorithm to solve the cloud resource allocation problem in configurable business process models. To do so, we specify the parameters for our genetic algorithm.

5.3.3.1 Genome Encoding

To solve such problem which is tedious and complex task when having a large amount of available cloud resources, we use Genetic algorithm. Genetic algorithm [187] is a powerful tool to deal with combinatorial problems. Furthermore, it has been successfully applied in many other research domains. Concretely, genetic algorithm is an optimal solution search in which the solutions of the problems are encoded in the form of arrays which represent individuals. The set of these individuals, which constitute a population, is randomly created to represent different points in the search space. Each individual is evaluated by a *fitness function* that represent the degree of goodness. Operators (e.g., crossover, and mutation) are applied to generate new individuals. This generation continues until the presented maximum number of generations is achieved or the specific conditions are satisfied, then the output is the individual with the best fitness value as an optimal solution [188].

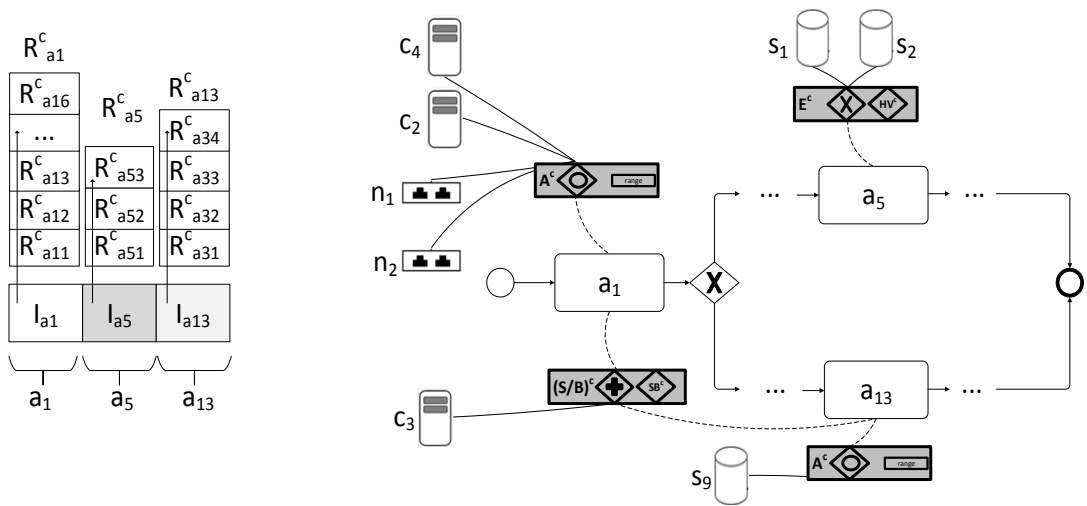


Figure 5.5: Configurable business process example

According to Definition 5.3.5, our configurable BP model contains a set of activities A and a set of cloud resources OR . Based on configurable resource operators, each activity a_i can be allocated to multiple configuration of resources R_A^c . For example, the activity a_1 (see Figure 5.5) can be allocated to the following resource configurations $R^{c a_1} = \{\{c_2, c_3\}, \{c_2, n_1\}, \{c_2, n_2\}, \{c_2, c_4\}, \{c_4, n_2\}, \{c_4, c_2, n_1\}, \text{etc}\}$.

Definition 5.3.5 (Configuration of resource allocation). *A configuration of resource allocation for a CBP is denoted as $R_A^c = \{R^{c a_1}, R^{c a_2}, \dots, R^{c a_k}\}$ where:*

- $A = \{a_1, a_2, \dots, a_n\}$ is the set of activities;
- $OR = \{r_1, r_2, \dots, r_m\}$ is the set of cloud resources;
- $R^{c a_i} = \cap \{r_1, r_2, \dots, r_j\}$ is the set of edges connecting two activities;

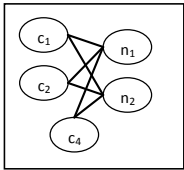

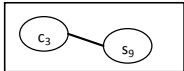
| a_i | Resource configuration | Bit representation | |
|----------|---|---------------------------------|-----|
| a_1 |  | $R_{a11}^c = \{c_1, n_1\}$ | 000 |
| | | $R_{a12}^c = \{c_1, n_2\}$ | 001 |
| | | ... | ... |
| | | $R_{a13}^c = \{c_1, c_2, n_2\}$ | 101 |
| | | ... | ... |
| a_5 |  | $R_{a21}^c = \{s_1\}$ | 000 |
| a_{13} |  | $R_{a31}^c = \{c_3\}$ | 000 |
| | | $R_{a32}^c = \{s_9\}$ | 001 |
| | | $R_{a33}^c = \{c_3, s_9\}$ | 010 |

Figure 5.6: Cloud resource configuration represented as bits

We encode the cloud resource allocation problem of the configurable BP with the genome depicted at the left in Figure 5.5. The genome represents an integer array where the number of its elements is equal to the number of activities that consume at least one resource. Each element refers to an activity a_i that contains an index I_{a_i} to the array of its cloud resource configurations $R^{c a_i}$. Then, we apply the standard two-point crossover operator to generate the population of genome arrays. Such operator randomly selects an activity a_i (i.e., an index in the genome array) and replaces the index of resource configuration I_{a_i} corresponding to its cloud resource configuration array $R^{c a_i}$ by another configuration. In our example, the maximum number of resource configurations for one activity is 6. Thus, three bits are enough to express every cloud resource configuration (see Figure 5.6). One of the possible

generated population is $I_{a1} = 010$, $I_{a5} = 000$, and $I_{a13} = 001$ which represents cloud resource configurations $R_{a13}^c = \{c_2, n_1\}$ of the activity a_1 , $R_{a51}^c = \{s_1\}$ of a_5 , and $R_{a32}^c = \{c_2\}$ of a_{13} , respectively.

To ensure the correctness of shareability property in the genome construction, we extract from the variants RAV (i.e., the acceptable set of resource allocation variants) that are obtained in section 5.3.2, the choices of the configurable resource operators S/B^c . For instance, if the configurable type of the S/B^c operator is configured to an XOR (Figure 5.5), the bit dedicated to the according resource c_3 should be either 1 for a_1 or a_{13} . Yet, at this stage we do not handle the constraints related to the elasticity resource operators E^c .

In the following, we compute the fitness function in two ways, i.e., with respect to two types of constraints: (i) green properties (section 5.3.3.2), and (ii) QoS properties (section 5.3.3.3).

5.3.3.2 Fitness Function considering Green properties

Once the acceptable variants are selected, the next step consists in assisting the tenant in order to get an energy efficient process variant. To this end, we describe in this section the ecological characteristics related to the environmental impact, referred to as green properties (GP). Then, we define the related constraints which are specified as a set of restrictions to respect within Service Level Agreements (SLA) constraints.

Several GP or green metrics have been proposed such as carbon footprint or CO2 emissions [155,189], CPU utilization [161], etc. In our work, we take into account three main metrics: Green Efficiency (GE), Energy Productivity (EP), and CO2 emissions (CE) which description is presented in Table 5.5. For instance, GE depicts the part of energy consumed by the resource provider that is produced by green energy sources.

Table 5.5: Green properties Description

| Green Metric | Description |
|--------------------------|--|
| Green Efficiency (GE) | The part of energy consumed by the resource provider that is produced by green energy sources. |
| Energy Productivity (EP) | Ratio between the output of the resource (i.e., number of consumption times) in a certain time interval and the energy consumed. |
| CO2 emissions (CE) | Quantity of CO2 provided by the resource. |

Accordingly, we define related constraints as follows. The two first constraints denote that the GE and EP of a given BP should be greater than minimal thresholds $gmax$ and $emax$. While the last constraint means that the CE should be smaller than maximal threshold referred to as $cmax$.

$$\begin{aligned}
 GE(BP) &\geq gmax \\
 EP(BP) &\geq emax \\
 CE(BP) &\leq cmax
 \end{aligned}
 \tag{5.2}$$

Based on these constraints, we define a green fitness (GF) to enable the assessment of the consistency level of the green properties for a given BP. It is described as follows.

$$GF(BP) = \alpha_1 * (GE(BP) - gmax) + \alpha_2 * (EP(BP) - emax) + \alpha_3 * (cmax - CE(BP)) \quad (5.3)$$

Where α_1 , α_2 and α_3 are defined as the weighting factors that belong to the interval $[0,1]$. They represent the user preferences w.r.t the property ($\alpha_1 + \alpha_2 + \alpha_3 = 1$).

Furthermore, the fitness function needs to maximize the function GF (see Equation 5.3). The values of Green metrics of cloud resources are illustrated in Table 3. Using our genetic algorithm and our fitness function, the genome g having $I_{a1} = 010$, $I_{a5} = 000$, and $I_{a13} = 001$, we can compute fitness function $GF(g) = 0.275$.

Table 5.6: Green metrics of cloud resources

| Resource ID | Green Efficiency (GE) | Energy Productivity (EP) | CO2 emissions (CE) |
|-------------|-----------------------|--------------------------|--------------------|
| c_1 | 5.0 | 0.8 | 0.3 |
| c_2 | 6.0 | 0.7 | 0.1 |
| c_3 | 5.5 | 0.7 | 0.2 |
| n_1 | 2.5 | 0.7 | 0.5 |
| n_2 | 3.0 | 0.6 | 0.2 |

Algorithm 3 Efficient resource allocation variant construction's algorithm

Require: RAV, δ

Ensure: bit

```

1:  $bit \leftarrow GenerateAnInitialBit(RAV)$            ▷ Generate an initial bit representation
2:  $bit \leftarrow GENOME(bit, \delta)$                  ▷ Start genetic algorithm.
3: return  $bit$                                      ▷ Return an optimal cloud resource allocation variant.
4: procedure  $GENOME(bit, \delta)$                    ▷ Check if the stop condition is respected
5:   while  $GF(bit) < \delta$  do
6:      $bit \leftarrow Crossover()$                  ▷ Apply the crossover operation to the bit
7:      $bit \leftarrow Mutation()$                  ▷ Apply the mutation operation to the bit
8:      $GENOME(bit, \delta)$                          ▷ Recursive call for the next population.
9:   end while
10:  return  $bit$                                    ▷ Return an optimal cloud resource allocation variant.
11: end procedure

```

It is necessary to define a stop condition for genetic algorithm. Hence, we consider the following condition:

$$GF(g) \geq \delta \quad (5.4)$$

where δ is the weighting quality factor (positive real value). The value of δ is determined by experimentations. A solution represented by a genome is feasible only if it respects this stop condition. The steps of our genetic-based algorithm are illustrated through the algorithm 3.

Firstly, this algorithm input is a set of the acceptable resource variants RAV and a δ value. Firstly, we generate an initial bit representation bit based on the maximum number of cloud resources in RAV (line 1). For example, an initial bit representation for cloud resource allocation variants in Figure 5.6 is 9 bits (000 000 000). We start our genetic algorithm with an initial bit in line 2 and return a bit represented optimal allocation in line 3. In our genetic algorithm (line 4-11), we compute the fitness function GF from bit (line 5). If the result is less than the δ value, we apply the crossover operator for obtaining next bit representation by taking into account previous populations to produce a new population (line 6-7). The mutation operator randomly selects an activity (i.e., a position in the genome) and randomly replaces the bit representation with another possible cloud resource allocation variant. Thus, we recursively call the genetic algorithm function for the next population (line 8). If the result is satisfy with the δ value, the algorithm will return the resource allocation variant of the bit representation bit (line 9).

5.3.3.3 Fitness Function considering QoS properties

The same steps, previously followed, are used to define our fitness function that takes into account non-functional properties. Therefore, we consider a set of important QoS properties: cost (CT), response time (RT) and availability (AT). Next, we specify appropriate constraints as shown in Equation 5.5. The two first constraints indicate that the CT and RT of a given BP should be less than maximal thresholds $cmax$ and $rmax$. Whereas the last constraint shows that the AT should be greater than minimal threshold referred to as $amax$.

$$\begin{aligned} CT(BP) &\geq cmax \\ RT(BP) &\geq rmax \\ AT(BP) &\leq amax \end{aligned} \tag{5.5}$$

Thereafter, we define the fitness function (FF) to enable the assessment of these QoS properties for a given BP described as follows.

$$\begin{aligned} FF(BP) &= \alpha_1 * (cmax - CT(BP)) + \alpha_2 * (rmax - RT(BP)) \\ &+ \alpha_3 * (AT(BP) - amax) \end{aligned} \tag{5.6}$$

Where α_1 , α_2 and α_3 are defined as the weighting factors that belong to the interval $[0,1]$. They represent the user preferences w.r.t the QoS properties ($\alpha_1 + \alpha_2 + \alpha_3 = 1$).

The fitness function needs to maximize the function FF (Equation 5.6). Then we define the stop condition in this case as follows.

$$FF(g) \geq \delta \quad (5.7)$$

where δ is the weighting quality factor. The value of δ is determined by experiments. A solution represented by a genome is feasible only if it respects this stop condition.

Algorithm 4 Resource allocation variant construction's algorithm

Require: RAV, δ

Ensure: bit

```

1:  $bit \leftarrow GenerateAnInitialBit(RAV)$            ▷ Generate an initial bit representation
2:  $bit \leftarrow GENOME(bit, \delta)$                  ▷ Start genetic algorithm.
3: return  $bit$                                        ▷ Return an optimal cloud resource allocation variant.
4: procedure  $GENOME(bit, \delta)$                      ▷ Check if the stop condition is respected
5:   while  $FF(bit) < \delta$  do
6:      $bit \leftarrow Crossover()$                    ▷ Apply the crossover operation to the bit
7:      $bit \leftarrow Mutation()$                    ▷ Apply the mutation operation to the bit
8:      $GENOME(bit, \delta)$                            ▷ Recursive call for the next population.
9:   end while
10:  return  $bit$                                      ▷ Return an optimal cloud resource allocation variant.
11: end procedure

```

Algorithm 4 shows the steps of our genetic-based algorithm. It follows the same instructions as the previous one but with taking into account the new parameters of the fitness function FF .

5.4 Evaluation and validation

In this section, we present the validation and experiments that we have conducted to show the effectiveness of our proposals on supporting the Cloud resource variability in configurable process models. Our purpose is to demonstrate that our approach is feasible and accurate in real use-cases.

Firstly, we implemented a proof of concept that is an extension of the *Signavio* editor aiming at integrating our proposed configurable Cloud resource operators in order to allow the process tenants to easily configure their allocated resources.

Secondly, we have conducted experiments using real world datasets of configurable business processes from France Telecom/Orange labs, which is a french telecom industrial partner .

- To support configurable resource allocation we extend Signavio editor in order to allow the process users to easily configure their allocated resources through the use of the proposed configurable resource operators (Section 5.4.1).

- To support Cloud resource configuration in multi-tenant process models, we used a real dataset of business processes from France Telecom/Orange labs i.e., (i) process variants, and (ii) cloud resources. We assess the quality of our model in terms of structural complexity and make a comparison with two previous configurable process models presented in the literature (Section 5.4.2.1).
- To optimize the cloud resource allocation process variants, we evaluate the performance of our genetic algorithm and then make a comparison between our approach and other well-known mathematical optimization program Linear Integer Programming (LIP) [190] (Section 5.4.2.2).

5.4.1 Supporting Configurable Resource Allocation

We develop a second extension to *Signavio* as a proof of concept to validate our approach presented in Chapter 5. More details on our application can be found on line¹.

We have added two main functionalities to the first proof of concept that are described as follows:

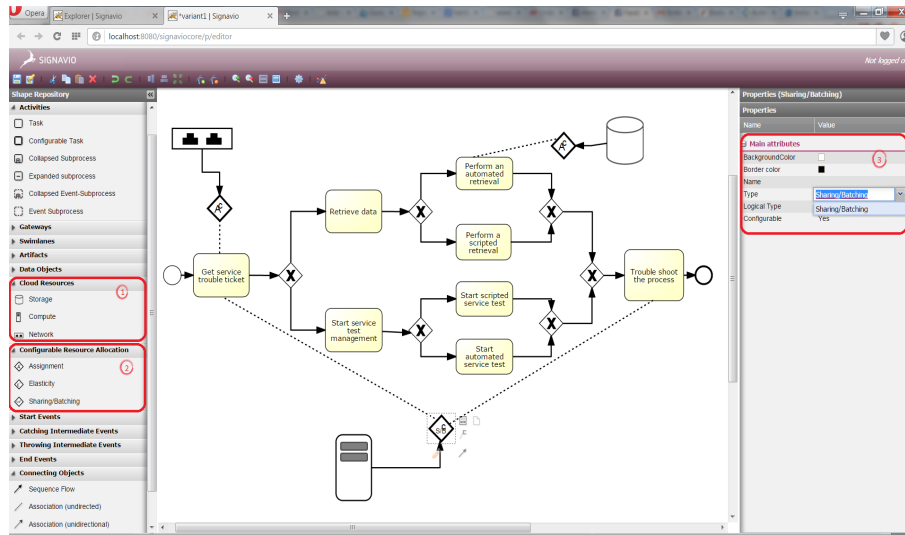


Figure 5.7: Application Screenshot: Configurable Resource Allocation

1. *Configurable resource allocation operators*: This functionality allows to allocate the cloud resources to activities using the configurable operators presented in Section 5.2. The three configurable operators (i) assignment for A^c , (ii) elasticity

¹<http://www-inf.it-sudparis.eu/SIMBAD/tools/Configurable-RA-BPM>

for E^c and (iii) sharing/Batching for $(S/B)^c$ (Area 2 in Figure 5.7) can be used to link the process activities to their allocated cloud resources.

2. *The configuration choices:* We have added the different configurable parameters (e.g. configurable type, configurable elasticity type, etc.) so that the user can specify its proper values. The configuration choices can be also specified (Area 3 in Figure 5.7).

5.4.2 Experimentation

We present, in this section, the experiments that we performed to evaluate our two main approaches. First, we show the experiments for supporting configurable cloud resource allocation in process models (Section 5.4.2.1). To this end, we used a real dataset of business processes from French Telecom/Orange Labs. Second, we present the evaluation of our genetic-based approach (Section 5.4.2.2). To do so, we assessed the quality of our solution as well as comparing it with a popular existing technique.

5.4.2.1 Structural Complexity experiments

In order to evaluate the usefulness and effectiveness of our approach for supporting configurable resource allocation in cloud-based process models, we performed experiments using a real dataset of business processes from Orange Labs. Different variants of business processes for VoIP assurance in France are defined and used by Orange. These variants and their allocated resources are manually and separately described. In total, there are 28 variants of the same process using about 30 different resources. Some activities have the same allocated resources in multiple variants, while others have different allocated resources and different needs for shareability and elasticity. In order to consolidate their expertise in telecommunication domain, Orange experts were interested in constructing one consolidate configurable model that also depicts the different resource allocation strategies.

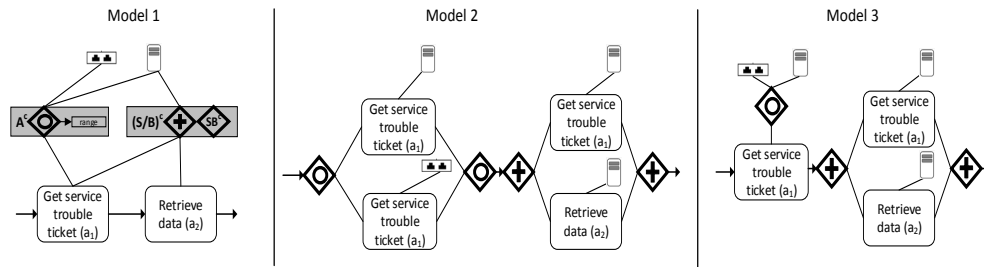


Figure 5.8: fragments of configurable processes of the three models

To construct the configurable model, we proceeded in three different ways. First, using our approach, we designed a configurable process model that depicts the vari-

ability both at the control flow and resource flow levels. Second, we modelled a configurable process model with a basic approach that does not consider the variability at the resource level. Thus, whenever an activity has different resource allocation possibilities, it is duplicated in the model in a choice block to express that there exist different resource allocation possibilities and so one should be selected. Third, we designed the same configurable process model using the approach introduced by La Rosa et al. in [5] which is close to ours but does not consider the variability at the shareability and elasticity levels. Therefore, when such a variability occurs (e.g., an activity has the same allocated resources in different variants but with different needs for elasticity and shareability) the same strategy as in the second approach (i.e. duplication of activities) is used. Figure 5.8 shows three process fragments from Model 1, Model 2 and Model 3. An activity a_1 is assigned to a variable number of resources (network and compute). The compute resource can be shared between a variable number of the activities (instances) a_1 and a_2 . According to our approach (represented by Model 1), a_1 is linked to the compute and network resources with a configurable *OR* via the configurable assignment operator. The compute resource is shared and therefore is linked to a_1 and a_2 with a configurable *AND* via the configurable Sharing/Batching operator. In the basic approach (represented by Model 2), there are two duplications. The first one is to model the configurable allocation. It is represented in the model by an activity a_1 assigned to the compute resource, and another activity a_1 assigned to the network resource which are connected by a configurable *OR*. The second one is to model the configurable shareability. It is represented in the model by the activities a_1 and a_2 assigned to the compute resource and connected through a configurable *AND* that represents the configurable shareability choice. In the approach of La Rosa et al. (represented by Model 3), there is only one duplication to model the configurable shareability. The configurable allocation is supported and can be modelled as in our approach.

| Complexity metric | | Model 1 | Model 2 | Model 3 |
|-------------------|-------------|---------|---------|---------|
| CFC | CFC_c | 28 | 128 | 39 |
| | CFC_r | 25 | - | 14 |
| ACD | ACD_c | 3.30 | 7.37 | 5.61 |
| | ACD_r | 2.11 | - | 1.64 |
| CNC | CNC_c | 0.56 | 1.18 | 0.61 |
| | CNC_r | 0.51 | - | 0.78 |
| density | $density_c$ | 0.02 | 0.01 | 0.01 |
| | $density_r$ | 0.04 | - | 0.03 |

Table 5.7: Structural Complexity metrics for different approaches

Thereafter, we assessed the quality of the three models in terms of their structural complexity. We computed the well known complexity metrics proposed in the literature: CFC (Control Flow Complexity), ACD (Average Connector Degree), CNC

(Coefficient of Network Connectivity) and density. The CFC [191] metric evaluates the complexity of the process with respect to the presence of gateways OR, AND and XOR. The ACD [192] metric generates the number of nodes that a connector has as an average. The CNC [193] gives the ratio of edges to nodes. Whereas the density [194] metric relates the number of edges to the number of maximum edges that can exist among nodes.

The above metrics have been proposed to assess the complexity of the control flow perspective in business processes. We also use these metrics to compute the complexity of the resource flow perspective since we are using control-flow like operators (i.e. XOR, OR and AND). The obtained values for the three configurable process models are summarized in Table 5.7. Model 1 refers to the configurable process model constructed with our approach; Model 2 is the configurable process model constructed with a basic approach that does not take the variability at the resource level; and Model 3 is the configurable process model constructed using the approach in [5]. For Model 1 and Model 3, we separately compute the complexity metrics at the control flow (referred to as $[\text{metric}]_c$) and resource flow (referred to as $[\text{metric}]_r$) perspectives. This is a logical choice since the resource and control-flow perspectives are separately modeled.

The results show that the metrics of Model 1 have noticeably low values compared to values of Model 2. For instance, even by summing the CFC_c (28) and CFC_r (25) of Model 1, the result remains smaller than the CFC_c (128) of Model 2. Hence, separately modelling the control-flow and resource-flow variability decreases the complexity of the model. We also notice that the density $density_c$ (0.02), and $density_r$ (0.04) of Model 1 are greater than the density $density_c$ (0.01) of Model 2. However, as stated in [195], the density metric is negatively correlated with the complexity of the model.

By comparing the metrics' values of Model 1 and 3, we notice that Model 1 has better complexity values for the control-flow while Model 3 has better complexity values for the resource-flow. This can be explained by the fact that Model 1 fully supports the resource variability modelling (i.e. allocation, shareability and elasticity). Therefore, we do not need to do duplications in the control flow and hence we obtained better complexity values for the control flow. Whereas, Model 3 is less expressive and only supports the resource variability modelling (i.e. allocation). So, it has better complexity values for the resource flow. Since we did duplications in the control flow to model the variability in the shareability and elasticity, we obtained worst complexity values for the control flow.

5.4.2.2 Solving optimal cloud resource allocation

To show the effectiveness of our approach, we conduct experiments using a real dataset of process models from Orange Labs. We start by studying the impact of the value of the quality of the solution δ . Then, we evaluate the quality of our approach by

comparing our genetic-based approach with Linear integer programming. We extend the configurable business process model in the previous section by adding more cloud resource operators and assigned cloud resources. The experiments were performed on a computer with Intel, 3.0 GHz, 8 GB of RAM, Windows 7 enterprise edition. Our genetic-based approach was implemented in Java using a library².

Impact of the Quality Factor

To assess the quality of our genetic algorithms presented in both Sections 5.3.3.2 and 5.3.3.3 following two ways, we proceed in the same manner. We first vary the quality of the solution δ from 0 to 20. The weight of the fitness function, α_1 , α_2 , and α_3 are 0.33 for simplicity sake. Thereafter, we identify the number of feasible cloud resource allocation variants for each case.

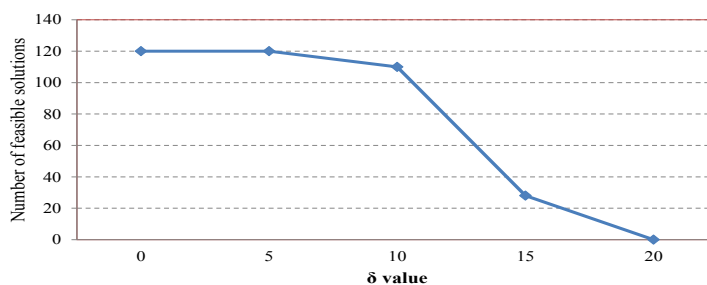


Figure 5.9: The impact of the of δ on the quality of the solution

Figure 5.9 shows the impact of δ on the number of feasible solutions. The results show that the number of feasible resource allocation variants decreases when δ increases. From the δ value is 20, there is a risk to reject feasible resource allocation variant. To conclude, the quality of the solution is influenced by δ .

Comparing genetic-based approach with Linear Integer Programming

To compare our genetic-based approach with Linear Integer Programming, we use the same dataset as the parameters that we follow for the two visions of our approach (Sections 5.3.3.2 and 5.3.3.3). The Linear Integer Programming (LIP) is a broadly used mathematical optimization program [196].

We study, herein, the growth of the computation time comparing to the number of consumed resources per process variant. To do so, we generate 5 variants from our configurable process model (Figure 5.2) having number of assigned cloud resources per activity from 4 to 20. In addition, we customize configurable resource operators

²<http://jenetics.io/>

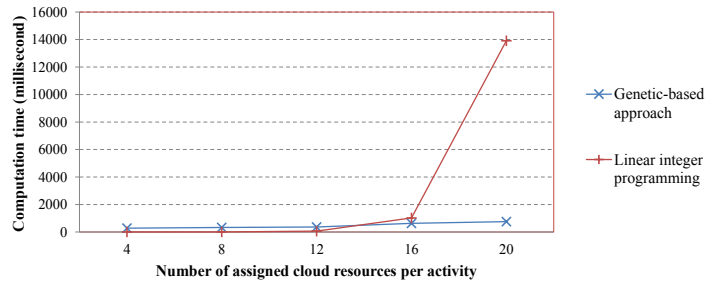


Figure 5.10: Comparing Genetic-based approach using QoS properties with LIP

to inclusive choice (*OR*) to maximize the complexity. For example, having 8 cloud resources assigned to an activity can have 255 possible resource allocation variants.

Figure 5.10 and 5.11 depicts the results of our comparison using QoS properties and green properties, respectively. Both Figures show that when the number of assigned cloud resources per activity is small (4-12), Linear integer programming outperforms our genetic-based approach. However, when the number of cloud resources are higher (16 and 20), our genetic-based approach is able to keep its timing performance almost constant while linear integer programming performs exponential growth. For instance, if the activity allocates 20 resources, the computation time of our genetic algorithm is 700ms (Figure 5.10) and 800ms (Figure 5.11), whereas the computation time related to LIP algorithm reaches 14000ms (Figure 5.10) and 12000ms (Figure 5.11). Therefore, the LIP algorithm can be efficient only when the problem is not complex. We can conclude that our genetic-based approach should be preferred than LIP in the scenario that we have a large number of cloud resources can be assigned to process activities, i.e., large-scale service-based business processes.

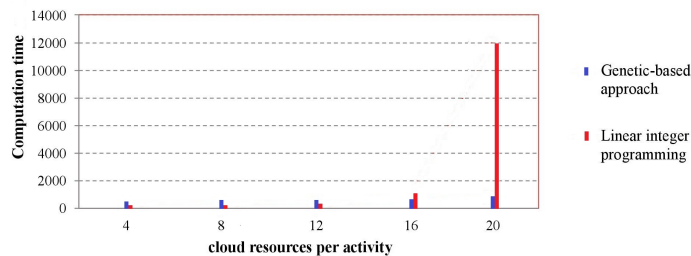


Figure 5.11: Comparing Genetic-based approach using green properties with LIP

5.5 Threats to Validity

Some potential threats to validity exist in our thesis. First, we have been interested in IaaS resources since they are the raw resources on which all others (i.e. PaaS and SaaS resources) are built. However, our study can be easily extended to consider the PaaS and SaaS resources.

Second, we have shown the feasibility of our study through real datasets from an industrial partner. Yet, the study requests a larger dataset that involve more process variants which represent the whole configuration possibilities in order to further evaluate the effectiveness of our approach.

Third, the proposed configurable resource operators as well as dependencies among cloud resources should formally be described, which are of a high importance in a multi-tenant environment. Nevertheless, we aim, in future work, at extending our study so that we define these dependencies in a flexible way so that tenants can customize them depending on their needs.

Finally there are other important metrics related to the environmental impact, which are of a high importance, that we do not take into account such as CPU utilization, carbon footprint, etc. In addition, our approach for optimizing the cloud resource allocation can be compared to other popular algorithms, in addition to LIP, such as the Ant Colony Optimization. In fact, research on cloud resources' management in BPM still at its beginning stage.

5.6 Conclusion

In this chapter, we answered the three questions mentioned in our thesis problematic (see section 1.2.2) which are: *How to allow users to configure their choices in terms of cloud resources?*, *How to integrate the cloud resource variability in configurable process models w.r.t the cloud properties i.e., elasticity or shareability?*, and *How to select the optimal cloud resource allocation?*.

To allow users to configure their choices in terms of cloud resources, we proposed an approach for configurable cloud resource allocation in multi-tenant business processes. Our aim is to shift the cloud resource allocation from the tenant side to the cloud process provider side for a centralized resource allocation management. Through configuration, different tenants can easily derive their allocated resources.

To integrate the cloud resource variability in configurable process models w.r.t the cloud properties i.e., elasticity or shareability, we have extended configurable process models with new resource operators. These resource operators take into account the characteristics of resources in the Cloud: configurable resource assignment operator (A^c), configurable resource elasticity operator (E^c), and configurable resource sharing/batching operator ($(S/B)^c$).

To select the optimal cloud resource allocation, we have proposed an approach that, using genetic algorithm, aims at ensuring an efficient configurable resource al-

location in cloud-based BPs. It guarantees, on the one hand, the selection of the process variant which best fits to the process tenant requirements, and optimizes on the other hand the green and QoS requirements.

Our principles presented in Section 1.4.1 are respected:

- **Exploitation of knowledge:** We exploit the context of resources allocated to process activities in addition to the cloud properties. We allow to process tenants to be aware of green properties and QoS properties.
- **Balanced computation complexity:** Our approach extends configurable business process models with new resource operators. Thus, it seems becoming more complex. Hence, we conduct experimentations to assess the complexity of our model (Sectionxx).
- **Fine-grained results:** To not confuse process users, we have proposed operators that are specific to resources and different from the control-flow operators.

Conclusion and Future Works

The research problem of our thesis is expressed by this interrogation: *How to support Cloud resource allocation in configurable business process models?* We presented in details, in previous chapters, our solutions to answer the raised question. In this chapter, we first summary our work in section 6.1 then we present the future work in section 6.2.

6.1 Contributions

Configurable process models are gaining momentum since they allow various organizations to derive, from a customizable process model, their suitable process variants according to their requirements. This configuration is more flexible and important if the deployment is realized in Cloud environments. Therefore, organizations become more interested in adopting such Cloud-based process models. In such environment, such organizations need to support large amount of cloud resources and configure their resource allocation that fit to their specific requirements. Not only supporting the resource allocation in such process models has been an interesting topic over the last years. But also supporting the variability of resource allocation constitutes an appealing challenge.

Many researches have been involved in both academics and industry. They differently address these challenges. On the one hand, several researches working on the resource perspective in process models have been proposed. They propose to support the resource perspective by extending process modeling languages. They provide formal definitions for the allocated resources through the use of semantic web and social computing. Nevertheless, such approaches only take into account objects and especially human resources and do not consider the particular resource type that are Cloud resources. On the other hand, many researches working on the variability in configurable process models have been introduced. They develop configurable process modeling languages to allow process variability modeling. They propose configuration support systems in order to assist process users to derive their suitable individual variants. However, such proposals lack of supporting the variability at the level of the resource selection as well as its optimization regarding the QoS requirements.

To address these challenges, we proposed in this thesis an approach for supporting the design and configuration of the Cloud resource allocation in configurable process models in a fine-grained way.

We proposed to integrate the structure of Cloud resources according to the architecture of OCCI standard by extending the BPMN definition as a process modeling language. Thereafter, we propose to ensure the interoperability of Cloud resources by using semantic definitions. To do so, we reused and extended ontologies from the European project SUPER to uniformly describe heterogeneous resources. The obtained process models are stored in a shared knowledge base to be reused afterwards.

We proposed a semantic framework to capture social dependencies connecting business process components and particularly Cloud resources, and resolve resource-based conflicts using SWRL rules.

We proposed to extend configurable process models with configurable resource operators in order to support the variability of resources. These configurable resource operators take into account the requirements of Cloud process providers in terms of elasticity and shareability. Through configuration, they allow different tenants to easily derive their allocated resources.

Once our configurable model involving the resource variability is built, we proposed to optimally select the Cloud resource allocation to the process variants. For this end, we adopt a genetic algorithm as the most suitable mechanism for such complicated problem. From the proposed configurable resource operators, we derive all the possible variants of cloud resource allocation that respect the Cloud properties (i.e., elasticity and shareability). Thereafter, we apply a genetic algorithm by encoding such variants as genomes which depict individuals. We apply genetic operators (i.e., crossover, and mutation) to randomly generate new individuals or process variants. Each individual is then assessed by a fitness function which represents the degree of goodness. Finally, we identify the individual with the best fitness value as an optimal solution.

To validate our approach, four proofs of concepts were implemented. A validation of our main proposed ontology *So-CloudPrO* is realized, then three extensions of *Signavio* are implemented: (i) A support of Cloud resource descriptions, (ii) An application that populates a knowledge base of cloud resources, and (iii) A support of configurable resource allocation. The first extension aims at assisting the design of process models by allocating Cloud resources to appropriate activities. Its implementation is based on the ontologies and semantic rules developed in Chapter 4. The second extension shows the population of the Cloud resource knowledge base discussed in Chapter 4. The third extension allows the process tenants to configure their allocated resources through the use of our proposed configurable resource operators introduced in Chapter 5. In addition, experiments have been conducted, using real datasets from Orange Labs, to demonstrate that our approach is feasible and accurate in real use-cases.

The principles presented in Section 1.4.1 have been respected:

- Heterogeneous data modeling: We propose to adopt and extend existing ontologies to model and describe heterogeneous Cloud resources.
- Exploitation of knowledge: We extract existing knowledge about Cloud resources that are shared within an organization as a data-source to design process models. Further, we exploit relations that exist between the allocated Cloud resources in process models. Besides, we optimize the Cloud resource allocation to process variants using shared process models within an organization.
- Balanced computation complexity: The computation complexity of the proposed algorithms is polynomial hence the NP-complete problem is not confronted. The computation time is acceptable for tenants.
- Fine-grained results: We exploit the relations between resources and activities, and assist the design of the process model by separating the configuration of the resource elements from the control-flow elements.

6.2 Future work

In the future work, we intend to enhance the support quality of our current work and to tackle other new research perspectives based on our work.

Regarding our first contribution presented in Chapter 4, we are currently targeting to further evaluate the effectiveness of our approach by conducting a case study with a large dataset of SAP reference model [197]. We also intend to compute the Precision and Recall values on the results to estimate their quality.

Moreover, we are extending our semantic definitions to cover the variability aspect at the level of Cloud resources. In this way, we aim at ensuring the interoperability between Cloud providers and process models for tenants that can use different modeling languages and configuration mechanisms. We also attempt to formally define the dependencies among Cloud resources in a flexible way so that tenants can customize them depending on their requirements.

Regarding our second contribution presented in Chapter 5, we intend to verify the correctness of the generated process variants with respect to the depending relation between the configuration rules of the resource elements and the configuration rules of the control-flow elements. We also plan to develop an algorithm that permits an automatic configuration and generation of correct process models from a configurable model with Cloud resources. Therefore, we aim at implementing such algorithm in a proof-of-concept prototype [17].

As research on Cloud resources' management in BPM still at its beginning stage, we could follow other new perspectives. For instance, we aim at extending our approach to take into account the PaaS and SaaS resources. Besides, we could more focus on the ecological vision in order to further optimize the environmental impact.

In such context, related challenges which are of high importance constitute the other steps of resource orchestration i.e., Resource control, Resource Deployment, and Resource Monitoring (Section 1.2). More specifically we first plan to specify monitoring techniques in order to manage the dynamic change of resources to match new requirements.

Appendices

List of Publications

Conference Proceeding

1. Emna Hachicha and Walid Gaaloul: *Towards Resource-aware Business Process development in the Cloud*, In 29th International Conference on Advances Information Networking and Applications, AINA, Gwangju, South Korea, March 14-27, 2015 (Ranking: B).
2. Emna Hachicha, Walid Gaaloul and Zakaria Maamar: *Social-based Semantic Framework for cloud resource management in Business Processes*. In 13th International Conference on Services Computing, SCC, San Francisco, USA, June 27-July 2, 2016 (Ranking: A).
3. Emna Hachicha, Nour Assy, Walid Gaaloul and Jan Mendling, *A Configurable Resource Allocation in Multi-tenant Process Development in the Cloud*, In 28th International Conference on Advanced Information Systems Engineering, Caise, Ljubljana, Slovenia, June 13-17, 2016 (Ranking: A).
4. Emna Hachicha, Karn Yongsiriwit and Walid Gaaloul, *A Configurable Resource Allocation in Multi-tenant Process Development in the Cloud*, In 28th International Conference on Advanced Information Systems Engineering, Caise, Ljubljana, Slovenia, June 13-17, 2016 (Ranking: A).
5. Emna Hachicha, Karn Yongsiriwit, Mohamed Sellami and Walid Gaaloul, *Genetic-based Configurable Cloud Resource Allocation in QoS-aware Business Process Development*, In 24th IEEE International Conference on Web Services, June 25 - June 30, 2017, Honolulu, Hawaii, USA (Ranking: A).

Proof of Concepts

1. “Supporting Cloud resource Descriptions”, extension of Signavio process editor at <http://www-inf.it-sudparis.eu/SIMBAD/tools/So-CloudPr0/>
2. “Cloud resource Knowledge Base”, a ProM plugin at <http://www-inf.it-sudparis.eu/SIMBAD/tools/linked-cr/>
3. “Supporting Configurable Resource Allocation”, extension of Signavio at <http://www-inf.it-sudparis.eu/SIMBAD/tools/Configurable-RA-BPM>

Bibliography

- [1] Owl-s. Owl-s: Semantic markup for web services.
- [2] Yun Lin and John Krogstie. Semantic annotation of process models for facilitating process knowledge management. *IJISMD*, 1(3):45–67, 2010.
- [3] Arne Sølvberg. Conceptual modeling in a world of models. In *Entwicklungsmethoden für Informationssysteme und deren Anwendung, EMISA '99, Fachtungung der Gesellschaft für Informatik, September 1999 in Fischbachau*, pages 63–77, 1999.
- [4] Liliana Cabral, Barry Norton, , and John Domingue. The business process modelling ontology. In *Proceedings of the 4th International Workshop on Semantic Business Process Management*, pages 9–16, 2009.
- [5] Marcello La Rosa, Marlon Dumas, Arthur H. M. ter Hofstede, and Jan Mendling. Configurable multi-perspective business process models. *Inf. Syst.*, 36(2):313–340, 2011.
- [6] Michael Adams, Moe Thandar Wynn, Chun Ouyang, and Arthur H. M. ter Hofstede. Realisation of cost-informed process support within the YAWL workflow environment. In *Asia Pacific Business Process Management - Third Asia Pacific Conference, AP-BPM 2015, Busan, South Korea, June 24-26, 2015, Proceedings*, pages 3–18, 2015.
- [7] Alexander Nowak and Frank Leymann. Green business process patterns - part II (short paper). In *2013 IEEE 6th International Conference on Service-Oriented Computing and Applications, Koloa, HI, USA, December 16-18, 2013*, pages 168–173, 2013.
- [8] Antonucci Y.L. Using workflow technologies to improve organizational competitiveness. *International journal of management*, 14(1):117–126, 1997.
- [9] Richard Lenz and Manfred Reichert. It support for healthcare processes - premises, challenges, perspectives. *Data Knowl. Eng.*, 61(1):39–58, April 2007.
- [10] Marlon Dumas, Wil M. van der Aalst, and Arthur H. ter Hofstede. *Process-aware Information Systems: Bridging People and Software Through Process Technology*. John Wiley & Sons, Inc., New York, NY, USA, 2005.
- [11] Wil M. P. van der Aalst, Arthur H. M. ter Hofstede, and Mathias Weske. Business process management: A survey. In *Business Process Management, International Conference, BPM 2003, Eindhoven, The Netherlands, June 26-27, 2003, Proceedings*, pages 1–12, 2003.

-
- [12] Mathias Weske. *Business Process Management - Concepts, Languages, Architectures, 2nd Edition*. Springer, 2012.
 - [13] Howard Smith and Peter Fingar. *Business process management: the third wave*. Springer, 2003.
 - [14] Wil M. P. van der Aalst and Kees M. van Hee. *Workflow Management: Models, Methods, and Systems*. MIT Press, 2002.
 - [15] Dimitrios Georgakopoulos, Mark Hornick, and Amit Sheth. An overview of workflow management: From process modeling to workflow automation infrastructure. *Distrib. Parallel Databases*, 3(2):119–153, April 1995.
 - [16] Peter M. Mell and Timothy Grance. Sp 800-145. the nist definition of cloud computing. Technical report, 2011.
 - [17] Michael Rosemann and Wil M. P. van der Aalst. A configurable reference modelling language. *Inf. Syst.*, 32(1):1–23, 2007.
 - [18] Wil M. P. van der Aalst. Business process configuration in the cloud: How to support and analyze multi-tenant processes? In *9th IEEE European Conference on Web Services, ECOWS 2011, Lugano, Switzerland, September 14-16, 2011*, pages 3–10, 2011.
 - [19] Alireza Khoshkbarforousha, Meisong Wang, Rajiv Ranjan, Lizhe Wang, Leila Alem, Samee Ullah Khan, and Boualem Benatallah. Dimensions for evaluating cloud resource orchestration frameworks. *IEEE Computer*, 49(2):24–33, 2016.
 - [20] Stefan Schulte, Christian Janiesch, Srikumar Venugopal, Ingo Weber, and Philipp Hoenisch. Elastic business process management: State of the art and open challenges for BPM in the cloud. *Future Generation Comp. Syst.*, 46:36–50, 2015.
 - [21] Marcello La Rosa, Wil M.P. van der Aalst, Marlon Dumas, and Fredrik P. Milani. Business process variability modeling : A survey, 2013.
 - [22] Luis Jesús Ramón Stroppi, Omar Chiotti, and Pablo David Villarreal. Defining the resource perspective in the development of processes-aware information systems. *Information & Software Technology*, 59:86–108, 2015.
 - [23] Luis Jesús Ramón Stroppi, Pablo David Villarreal, and Omar Chiotti. Extending the ws-humantask architecture to support the resource perspective of BPEL processes. *CLEI Electron. J.*, 16(1), 2013.
 - [24] Luis Jesús Ramón Stroppi, Omar Chiotti, and Pablo David Villarreal. Extending BPMN 2.0: Method and tool support. In *Business Process Model and*

- Notation - Third International Workshop, BPMN 2011, Lucerne, Switzerland, November 21-22, 2011. Proceedings*, pages 59–73, 2011.
- [25] Marcello La Rosa, Marlon Dumas, Arthur H. M. ter Hofstede, Jan Mendling, and Florian Gottschalk. Beyond control-flow: Extending business process configuration to roles and objects. In *Conceptual Modeling - ER 2008, 27th International Conference on Conceptual Modeling, Barcelona, Spain, October 20-24, 2008. Proceedings*, pages 199–215, 2008.
- [26] Marin Dimitrov, Alex Simov, Sebastian Stein, and Mihail Konstantinov. A BPMO based semantic business process modelling environment. In *Proceedings of the Workshop on Semantic Business Process and Product Lifecycle Management SBPM 2007, held in conjunction with the 3rd European Semantic Web Conference (ESWC 2007), Innsbruck, Austria, June 7, 2007*, 2007.
- [27] Liliana Cabral and John Domingue. Translating semantic web service based business process models. In *4th IEEE Asia-Pacific Services Computing Conference, IEEE APSCC 2009, Singapore, December 7-11 2009, Proceedings*, pages 1–6, 2009.
- [28] Philipp Hoenisch, Christoph Hochreiner, Dieter Schuller, Stefan Schulte, Jan Mendling, and Schahram Dustdar. Cost-efficient scheduling of elastic processes in hybrid clouds. In *8th IEEE International Conference on Cloud Computing, CLOUD 2015, New York City, NY, USA, June 27 - July 2, 2015*, pages 17–24, 2015.
- [29] Akhil Kumar and Wen Yao. Design and management of flexible process variants using templates and rules. *Computers in Industry*, 63(2):112–130, 2012.
- [30] Alena Hallerbach, Thomas Bauer, and Manfred Reichert. Capturing variability in business process models: the provop approach. *Journal of Software Maintenance*, 22(6-7):519–546, 2010.
- [31] Carlos Pedrinaci, John Domingue, Christian Brelage, Tammo van Lessen, Dimka Karastoyanova, and Frank Leymann. Semantic business process management: Scaling up the management of business processes. In *Proceedings of the 2th IEEE International Conference on Semantic Computing (ICSC 2008), August 4-7, 2008, Santa Clara, California, USA*, pages 546–553, 2008.
- [32] Liliana Cabral and et al. Process ontology stack. project ist 026850 super deliverable 1.5. In *March 2009*.
- [33] Remco M. Dijkman, Marlon Dumas, and Luciano García-Bañuelos. Graph matching algorithms for business process model similarity search. In *Business Process Management, 7th International Conference, BPM 2009, Ulm, Germany, September 8-10, 2009. Proceedings*, pages 48–63, 2009.

- [34] Agata Filipowska, Martin Hepp, Monika Kaczmarek, and Ivan Markovic. Organisational ontology framework for semantic business process management. In *Business Information Systems, 12th International Conference, BIS 2009, Poznan, Poland, April 27-29, 2009. Proceedings*, pages 1–12, 2009.
- [35] Martin Hepp and Dumitru Roman. An ontology framework for semantic business process management. In *eOrganisation: Service-, Prozess-, Market-Engineering: 8. Internationale Tagung Wirtschaftsinformatik - Band 1, WI 2007, Karlsruhe, Germany, February 28 - March 2, 2007*, pages 423–440, 2007.
- [36] Florian Gottschalk, Wil M. P. van der Aalst, Monique H. Jansen-Vullers, and Marcello La Rosa. Configurable workflow models. *Int. J. Cooperative Inf. Syst.*, 17(2):177–221, 2008.
- [37] W.M.P. van der Aalst, A. Dreiling, F. Gottschalk, M. Rosemann, and M.H. Jansen-Vullers. Configurable process models as a basis for reference modeling. *Business Process Management Workshops*, 2006.
- [38] Marcello La Rosa, Wil M.P. van der Aalst, Marlon Dumas, and Fredrik P. Milani. Business process variability modeling : A survey, 2013. ACM Computing Surveys.
- [39] Rajiv Ranjan, Boualem Benatallah, Schahram Dustdar, and Michael P. Papazoglou. Cloud resource orchestration programming: Overview, issues, and directions. *IEEE Internet Computing*, 19(5):46–56, 2015.
- [40] Dennis M. M. Schunselaar, H. M. W. Verbeek, Hajo A. Reijers, and Wil M. P. van der Aalst. YAWL in the cloud: Supporting process sharing and variability. In *Business Process Management Workshops - BPM 2014 International Workshops, Eindhoven, The Netherlands, September 7-8, 2014, Revised Papers*, pages 367–379, 2014.
- [41] Cristina Cabanillas, Alex Norta, Manuel Resinas, Jan Mendling, and Antonio Ruiz Cortés. Towards process-aware cross-organizational human resource management. In *Enterprise, Business-Process and Information Systems Modeling - 15th International Conference, BPMDS 2014, 19th International Conference, EMMSAD 2014, Held at CAiSE 2014, Thessaloniki, Greece, June 16-17, 2014. Proceedings*, pages 79–93, 2014.
- [42] Cristina Cabanillas, David Knuplesch, Manuel Resinas, Manfred Reichert, Jan Mendling, and Antonio Ruiz Cortés. Ralph: A graphical notation for resource assignments in business processes. In *Advanced Information Systems Engineering - 27th International Conference, CAiSE 2015, Stockholm, Sweden, June 8-12, 2015, Proceedings*, pages 53–68, 2015.

- [43] Marcello La Rosa, Marlon Dumas, Arthur H. M. ter Hofstede, and Jan Mendling. Configurable multi-perspective business process models. *Inf. Syst.*, 36(2):313–340, April 2011.
- [44] Alena Hallerbach, Thomas Bauer, and Manfred Reichert. Capturing variability in business process models: The provop approach. *J. Softw. Maint. Evol.*, 22(6-7):519–546, 2010.
- [45] Ejub Kajan, Noura Faci, Zakaria Maamar, Alfred Loo, Aldina Pljaskovic, and Quan Z. Sheng. The network-based business process. *IEEE Internet Computing*, 18(2):63–69, 2014.
- [46] Occi open cloud computing interface specification. <http://occi-wg.org/about/specification/>.
- [47] W. M. P. Van Der Aalst, A. H. M. Ter Hofstede, B. Kiepuszewski, and A. P. Barros. Workflow patterns. *Distrib. Parallel Databases*, 14(1):5–51, July 2003.
- [48] A. J. M. M. Weijters and W. M. P. van der Aalst. Rediscovering workflow models from event-based data using little thumb. *Integr. Comput.-Aided Eng.*, 2003.
- [49] OMG. Business process model and notation (bpmn) 2.0. <http://www.omg.org/spec/BPMN/2.0/>.
- [50] Ludmila Penicina. Towards the mapping of multidimensional bpmn models to process definition standards, 2010.
- [51] Michael Zur Muehlen and Jan Recker. How much language is enough? theoretical and practical use of the business process modeling notation. In *Proceedings of the 20th International Conference on Advanced Information Systems Engineering, CAiSE '08*, pages 465–479, Berlin, Heidelberg, 2008. Springer-Verlag.
- [52] M. Rosemann and W. M. P. van der Aalst. A configurable reference modelling language. *Inf. Syst.*, 32(1):1–23, 2007.
- [53] Florian Gottschalk, WilM.P. van der Aalst, and MoniqueH. Jansen-Vullers. Configurable process models — a foundational approach. In Jörg Becker and Patrick Delfmann, editors, *Reference Modeling*, pages 59–77. Physica-Verlag HD, 2007.
- [54] Marcello La Rosa, Hajo A. Reijers, Wil M.P. van der Aalst, Remco M. Dijkman, Jan Mendling, Marlon Dumas, and Luciano García-Bañuelos. Aprocure: An advanced process model repository. *Expert Systems with Applications*, 38(6):7029 – 7040, 2011.

- [55] Wil M. P. van der Aalst. Intra- and inter-organizational process mining: Discovering processes within and between organizations. In *The Practice of Enterprise Modeling - 4th IFIP WG 8.1 Working Conference, PoEM 2011 Oslo, Norway, November 2-3, 2011 Proceedings*, pages 1–11, 2011.
- [56] W. M. P. van der Aalst, A. P. Barros, A. H. M. ter Hofstede, and B. Kiepuszewski. *Advanced Workflow Patterns*, pages 18–29. Springer Berlin Heidelberg, Berlin, Heidelberg, 2000.
- [57] Super (semantics utilised for process management within and between enterprises) integrated project. In <http://projects.kmi.open.ac.uk/super/>.
- [58] Omg business process model and notation. In <http://www.omg.org/spec/BPMN/2.0.2/>.
- [59] Robert B. Bohn, John Messina, Fang Liu, Jin Tong, and Jian Mao. Nist cloud computing reference architecture. In *Proceedings of the 2011 IEEE World Congress on Services, SERVICES '11*, pages 594–596. 2011.
- [60] Mohamed Mohamed. *Generic monitoring and reconfiguration for service-based applications in the cloud. (Supervision et reconfiguration génériques des applications à base de service dans le nuage)*. PhD thesis, Telecom & Management SudParis, Évry, Essonne, France, 2014.
- [61] R. Nyren, A. Edmonds, A. Papaspyrou, and T. Metsch. Open cloud computing interface - core. <http://www.ogf.org/documents/GFD.183.pdf>, 2011.
- [62] T. Metsch and A. Edmonds. Open cloud computing interface - restful http rendering. <http://www.ogf.org/documents/GFD.185.pdf>, 2011.
- [63] T. Metsch and A. Edmonds. Open cloud computing interface - infrastructure. <http://www.ogf.org/documents/GFD.184.pdf>, 2011.
- [64] Wil M. P. van der Aalst. Workflow patterns. In *Encyclopedia of Database Systems*, pages 3557–3558. 2009.
- [65] Bartek Kiepuszewski, Arthur H. M. ter Hofstede, and Wil M. P. van der Aalst. Fundamentals of control flow in workflows. *Acta Inf.*, 39(3):143–209, 2003.
- [66] Chun Ouyang, Eric Verbeek, Wil M. P. van der Aalst, Stephan Breutel, Marlon Dumas, and Arthur H. M. ter Hofstede. Formal semantics and analysis of control flow in WS-BPEL. *Sci. Comput. Program.*, 67(2-3):162–198, 2007.
- [67] Andrea Burattin, Alessandro Sperduti, and Wil M. P. van der Aalst. Control-flow discovery from event streams. In *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2014, Beijing, China, July 6-11, 2014*, pages 2420–2427, 2014.

- [68] Petia Wohed, Wil M. P. van der Aalst, Marlon Dumas, Arthur H. M. ter Hofstede, and Nick Russell. Pattern-based analysis of the control-flow perspective of UML activity diagrams. In *Conceptual Modeling - ER 2005, 24th International Conference on Conceptual Modeling, Klagenfurt, Austria, October 24-28, 2005, Proceedings*, pages 63–78, 2005.
- [69] Nick Russell and et al. Workflow resource patterns: Identification, representation and tool support. In *Advanced Information Systems Engineering, 17th International Conference, CAiSE, Portugal, 2005*.
- [70] Luis Jesús Ramón Stroppi, Omar Chiotti, and Pablo David Villarreal. Extended resource perspective support for BPMN and BPEL. In *Proceedings of the XV Iberoamerican Conference on Software Engineering, Buenos Aires, Argentina, April 24-27, 2012*, pages 56–69, 2012.
- [71] Nick Russell and Wil M. P. van der Aalst. Work distribution and resource management in bpel4people: Capabilities and opportunities. In *Advanced Information Systems Engineering, 20th International Conference, CAiSE 2008, Montpellier, France, June 16-20, 2008, Proceedings*, pages 94–108, 2008.
- [72] Anastasiia Pika, Moe Thandar Wynn, Colin J. Fidge, Arthur H. M. ter Hofstede, Michael Leyer, and Wil M. P. van der Aalst. An extensible framework for analysing resource behaviour using event logs. In *Advanced Information Systems Engineering - 26th International Conference, CAiSE 2014, Thessaloniki, Greece, June 16-20, 2014. Proceedings*, pages 564–579, 2014.
- [73] Jianrui Wang and Akhil Kumar. A framework for document-driven workflow systems. In *Business Process Management, 3rd International Conference, BPM 2005, Nancy, France, September 5-8, 2005, Proceedings*, pages 285–301, 2005.
- [74] Stefan Schönig, Cristina Cabanillas, Claudio Di Ciccio, Stefan Jablonski, and Jan Mendling. Mining resource assignments and teamwork compositions from process logs. *Softwaretechnik-Trends*, 36(4), 2016.
- [75] Giray Havur, Cristina Cabanillas, Jan Mendling, and Axel Polleres. Resource allocation with dependencies in business process management systems. In *Business Process Management Forum - BPM Forum 2016, Rio de Janeiro, Brazil, September 18-22, 2016, Proceedings*, pages 3–19, 2016.
- [76] Giray Havur, Cristina Cabanillas, Jan Mendling, and Axel Polleres. Automated resource allocation in business processes with answer set programming. In *Business Process Management Workshops - BPM 2015, 13th International Workshops, Innsbruck, Austria, August 31 - September 3, 2015, Revised Papers*, pages 191–203, 2015.

- [77] Evert F. Duipmans, Luís Ferreira Pires, and Luiz Olavo Bonino da Silva Santos. Towards a BPM cloud architecture with data and activity distribution. In *16th IEEE International Enterprise Distributed Object Computing Conference Workshops, EDOC Workshops, Beijing, China, September 10-14, 2012*, pages 165–171, 2012.
- [78] MingXue Wang, Kosala Yapa Bandara, and Claus Pahl. Process as a service. In *IEEE IC on Services Computing, SCC, Miami USA 2010*, pages 578–585, 2010.
- [79] Stefan Schulte, Christian Janiesch, Srikumar Venugopal, Ingo Weber, and Philipp Hoenisch. Elastic business process management: State of the art and open challenges for {BPM} in the cloud. *Future Generation Computer Systems*, 46:36 – 50, 2015.
- [80] Philipp Hoenisch, Dieter Schuller, Stefan Schulte, Christoph Hochreiner, and Schahram Dustdar. Optimization of complex elastic processes. *IEEE Trans. Services Computing*, 9(5):700–713, 2016.
- [81] Souha Boubaker, Amel Mammam, Mohamed Graiet, and Walid Gaaloul. Formal verification of cloud resource allocation in business processes using event-b. In *30th IEEE International Conference on Advanced Information Networking and Applications, AINA 2016, Crans-Montana, Switzerland, 23-25 March, 2016*, pages 746–753, 2016.
- [82] Souha Boubaker, Walid Gaaloul, Mohamed Graiet, and Nejib Ben Hadj-Alouane. Event-b based approach for verifying cloud resource allocation in business process. In *2015 IEEE International Conference on Services Computing, SCC 2015, New York City, NY, USA, June 27 - July 2, 2015*, pages 538–545, 2015.
- [83] Philipp Hoenisch, Stefan Schulte, Schahram Dustdar, and Srikumar Venugopal. Self-adaptive resource allocation for elastic process execution. In *2013 IEEE Sixth International Conference on Cloud Computing, Santa Clara, CA, USA, June 28 - July 3, 2013*, pages 220–227, 2013.
- [84] Philipp Hoenisch, Stefan Schulte, and Schahram Dustdar. Workflow scheduling and resource allocation for cloud-based execution of elastic processes. In *2013 IEEE 6th International Conference on Service-Oriented Computing and Applications, Koloa, HI, USA, December 16-18, 2013*, pages 1–8, 2013.
- [85] Stefan Schulte, Philipp Hoenisch, Srikumar Venugopal, and Schahram Dustdar. Introducing the vienna platform for elastic processes. In *Service-Oriented Computing - ICSOC 2012 Workshops - ICSOC 2012, International Workshops ASC, DISA, PAASC, SCEB, SeMaPS, WESOA, and Satellite Events, Shanghai, China, November 12-15, 2012, Revised Selected Papers*, pages 179–190, 2013.

- [86] Luise Pufahl, Nico Herzberg, Andreas Meyer, and Mathias Weske. Flexible batch configuration in business processes based on events. In *Service-Oriented Computing - 12th International Conference, ICSOC 2014, Paris, France, November 3-6, 2014. Proceedings*, pages 63–78, 2014.
- [87] Kahina Bessai, Samir Youcef, Ammar Oulamara, Claude Godart, and Selmin Nurcan. Resources allocation and scheduling approaches for business process applications in cloud contexts. In *4th IEEE International Conference on Cloud Computing Technology and Science Proceedings, CloudCom 2012, Taipei, Taiwan, December 3-6, 2012*, pages 496–503, 2012.
- [88] Kahina Bessai, Samir Youcef, Ammar Oulamara, Claude Godart, and Selmin Nurcan. Multi-objective resources allocation approaches for workflow applications in cloud environments. In *On the Move to Meaningful Internet Systems: OTM 2012 Workshops, Confederated International Workshops: OTM Academy, Industry Case Studies Program, EI2N, INBAST, META4eS, OnToContent, ORM, SeDeS, SINCOM, and SOMOCO 2012, Rome, Italy, September 10-14, 2012. Proceedings*, pages 654–657, 2012.
- [89] Mourad Amziani, Tarek Melliti, and Samir Tata. Formal modeling and evaluation of stateful service-based business process elasticity in the cloud. In *On the Move to Meaningful Internet Systems: OTM 2013 Conferences - Confederated International Conferences: CoopIS, DOA-Trusted Cloud, and ODBASE 2013, Graz, Austria, September 9-13, 2013. Proceedings*, pages 21–38, 2013.
- [90] Luise Pufahl and Mathias Weske. Batch processing across multiple business processes based on object life cycles. In *Business Information Systems - 19th International Conference, BIS 2016, Leipzig, Germany, July, 6-8, 2016, Proceedings*, pages 195–208, 2016.
- [91] Martin Hepp, Frank Leymann, John Domingue, Alexander Wahler, and Dieter Fensel. Semantic business process management: A vision towards using semantic web services for business process management. In *2005 IEEE International Conference on e-Business Engineering (ICEBE 2005), 18-21 October 2005, Beijing, China*, pages 535–540, 2005.
- [92] Branimir Wetzstein, Zhilei Ma, Agata Filipowska, Monika Kaczmarek, Sami Bhiri, Silvestre Losada, Jose-Manuel Lopez-Cob, and Laurent Cicurel. Semantic business process management: A lifecycle based requirements analysis. In *Proceedings of the Workshop on Semantic Business Process and Product Lifecycle Management SBPM 2007, held in conjunction with the 3rd European Semantic Web Conference (ESWC 2007), Innsbruck, Austria, June 7, 2007*, 2007.
- [93] Branimir Wetzstein, Zhilei Ma, and Frank Leymann. Towards measuring key performance indicators of semantic business processes. In *Business Information*

- Systems, 11th International Conference, BIS 2008, Innsbruck, Austria, May 5-7, 2008. Proceedings*, pages 227–238, 2008.
- [94] Carlos Pedrinaci, John Domingue, and Ana Karla Alves de Medeiros. A core ontology for business process analysis. In *The Semantic Web: Research and Applications, 5th European Semantic Web Conference, ESWC 2008, Tenerife, Canary Islands, Spain, June 1-5, 2008, Proceedings*, pages 49–64, 2008.
- [95] Carlos Pedrinaci and John Domingue. Towards an ontology for process monitoring and mining. In *Proceedings of the Workshop on Semantic Business Process and Product Lifecycle Management SBPM 2007, held in conjunction with the 3rd European Semantic Web Conference (ESWC 2007), Innsbruck, Austria, June 7, 2007*, 2007.
- [96] Ana Karla Alves de Medeiros, Carlos Pedrinaci, Wil M. P. van der Aalst, John Domingue, Minseok Song, Anne Rozinat, Barry Norton, and Liliana Cabral. An outlook on semantic business process mining and monitoring. In *On the Move to Meaningful Internet Systems 2007: OTM 2007 Workshops, OTM Confederated International Workshops and Posters, AWeSOMe, CAMS, OTM Academy Doctoral Consortium, MONET, OnToContent, ORM, PerSys, PPN, RDDS, SSWS, and SWWS 2007, Vilamoura, Portugal, November 25-30, 2007, Proceedings, Part II*, pages 1244–1255, 2007.
- [97] Chiara Di Francescomarino, Marco Rospocher, Chiara Ghidini, and Andrea Valerio. The role of semantic annotations in business process modelling. In *18th IEEE International Enterprise Distributed Object Computing Conference, EDOC 2014, Ulm, Germany, September 1-5, 2014*, pages 181–189, 2014.
- [98] Mauro Dragoni, Piergiorgio Bertoli, Chiara Di Francescomarino, Chiara Ghidini, Michele Nori, Marco Pistore, Roberto Tiella, and Francesco Corcoglioniti. Modeling and monitoring processes exploiting semantic reasoning. In *Proceedings of the ISWC 2014 Posters & Demonstrations Track a track within the 13th International Semantic Web Conference, ISWC 2014, Riva del Garda, Italy, October 21, 2014.*, pages 121–124, 2014.
- [99] Chiara Di Francescomarino, Francesco Corcoglioniti, Mauro Dragoni, Piergiorgio Bertoli, Roberto Tiella, Chiara Ghidini, Michele Nori, and Marco Pistore. Semantic-based process analysis. In *The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part II*, pages 228–243, 2014.
- [100] Chiara Ghidini, Chiara Di Francescomarino, Marco Rospocher, Paolo Tonella, and Luciano Serafini. Semantics-based aspect-oriented management of exceptional flows in business processes. *IEEE Trans. Systems, Man, and Cybernetics, Part C*, 42(1):25–37, 2012.

-
- [101] Barry Norton, Liliana Cabral, and Jörg Nitzsche. Ontology-based translation of business process models. In *Fourth International Conference on Internet and Web Applications and Services, ICIW 2009, 24-28 May 2009, Venice/Mestre, Italy*, pages 481–486, 2009.
- [102] Oliver Thomas and Michael Fellmann. Semantic EPC: enhancing process modeling using ontology languages. In *Proceedings of the Workshop on Semantic Business Process and Product Lifecycle Management SBPM 2007, held in conjunction with the 3rd European Semantic Web Conference (ESWC 2007), Innsbruck, Austria, June 7, 2007*, 2007.
- [103] Oliver Thomas and Michael Fellmann. Semantic process modeling - design and implementation of an ontology-based representation of business processes. *Business & Information Systems Engineering*, 1(6):438–451, 2009.
- [104] Ivan Markovic. Advanced querying and reasoning on business process models. In *Business Information Systems, 11th International Conference, BIS 2008, Innsbruck, Austria, May 5-7, 2008. Proceedings*, pages 189–200, 2008.
- [105] Chiara Di Francescomarino, Chiara Ghidini, Marco Rospocher, Luciano Serafini, and Paolo Tonella. Reasoning on semantically annotated processes. In *Service-Oriented Computing - ICSOC 2008, 6th International Conference, Sydney, Australia, December 1-5, 2008. Proceedings*, pages 132–146, 2008.
- [106] Chiara Di Francescomarino, Chiara Ghidini, Marco Rospocher, Luciano Serafini, and Paolo Tonella. Semantically-aided business process modeling. In *The Semantic Web - ISWC 2009, 8th International Semantic Web Conference, ISWC 2009, Chantilly, VA, USA, October 25-29, 2009. Proceedings*, pages 114–129, 2009.
- [107] Liliana Cabral, John Domingue, Enrico Motta, Terry R. Payne, and Farshad Hakimpour. Approaches to semantic web services: an overview and comparisons. In *The Semantic Web: Research and Applications, First European Semantic Web Symposium, ESWS 2004, Heraklion, Crete, Greece, May 10-12, 2004, Proceedings*, pages 225–239, 2004.
- [108] Lamia Youseff, Maria Butrico, and Dilma Da Silva. Towards a unified ontology of cloud computing. In *in Proc. of the Grid Computing Environments Workshop (GCE08)*, 2008.
- [109] Yong Beom Ma, Sung-Ho Jang, and Jong Sik Lee. Ontology-based resource management for cloud computing. In *Intelligent Information and Database Systems - Third International Conference, ACIIDS 2011, Daegu, Korea, April 20-22, 2011, Proceedings, Part II*, pages 343–352, 2011.

- [110] Amir Vahid Dastjerdi, Sayed Gholam Hassan Tabatabaei, and Rajkumar Buyya. An effective architecture for automated appliance management system applying ontology-based cloud discovery. *Cluster Computing and the Grid, IEEE International Symposium on*, 00:104–112, 2010.
- [111] Francesco Moscato, Rocco Aversa, Beniamino Di Martino, Teodor-Florin Fortis, and Victor Ion Munteanu. An analysis of mosaic ontology for cloud resources annotation. In *Federated Conference on Computer Science and Information Systems - FedCSIS 2011, Szczecin, Poland, 18-21 September 2011, Proceedings*, pages 973–980, 2011.
- [112] Giuseppe Di Modica and Orazio Tomarchio. A semantic framework to support resource discovery in future cloud markets. *IJCSE*, 10(1/2):1–14, 2015.
- [113] Giuseppe Di Modica and Orazio Tomarchio. A semantic model for utility driven discovery of cloud resources. In *Sixth International Conference on Complex, Intelligent, and Software Intensive Systems, CISIS 2012, Palermo, Italy, July 4-6, 2012*, pages 822–827, 2012.
- [114] Simeon Edosomwan and al. The history of social media and its impact on business. *JAME*, 16(3):1–7, 2011.
- [115] IBM. Social business: Amplify the value of human connections. 2014.
- [116] Zakaria Maamar and al. A framework of enriching business processes life-cycle with tagging information. In *26th ADC 2015*.
- [117] Muhammad Z. C. Candra, Hong Linh Truong, and Schahram Dustdar. Provisioning quality-aware social compute units in the cloud. In *Service-Oriented Computing - 11th International Conference, ICSOC 2013, Berlin, Germany, December 2-5, 2013, Proceedings*, pages 313–327, 2013.
- [118] Said Elnaffar, Zakaria Maamar, and Quan Z. Sheng. When clouds start socializing: The sky model. *IJEER*, 9(2):1–7, 2013.
- [119] Vanilson Arruda Burégio, Zakaria Maamar, and Silvio Romero de Lemos Meira. An architecture and guiding framework for the social enterprise. *IEEE Internet Computing*, 19(1):64–68, 2015.
- [120] Wil M. P. van der Aalst and Minseok Song. Mining social networks: Uncovering interaction patterns in business processes. In *BPM 2004, Germanys*, pages 244–260.
- [121] Zakaria Maamar and al. Towards an approach for weaving preferences into web services operation. *JSW'12*.

-
- [122] Zakaria Maamar and al. Network-based conflict resolution in business processes. In *IEEE 10th International Conference ICEBE 2013*.
- [123] Wil M. P. van der Aalst, Hajo A. Reijers, and Minseok Song. Discovering social networks from event logs. *Computer Supported Cooperative Work*, 14(6):549–593, 2005.
- [124] Wei Zhe Low, Wil M. P. van der Aalst, Arthur H. M. ter Hofstede, Moe Thandar Wynn, and Jochen De Weerd. Change visualisation: Analysing the resource and timing differences between two event logs. *Inf. Syst.*, 65:106–123, 2017.
- [125] J.M. Bhat and N. Deshmukh. Methods for Modeling Flexibility in Business Processes. In *BPMDs '05*.
- [126] Mathias Weske. *Business Process Management - Concepts, Languages, Architectures, 2nd Edition*. Springer, 2012.
- [127] George M. Giaglis. A taxonomy of business process modeling and information systems modeling techniques”. *International Journal of Flexible Manufacturing Systems*, 13, 2001.
- [128] Jan Recker, Michael Rosemann, Wil M. P. van der Aalst, and Jan Mendling. On the syntax of reference model configuration - transforming the C-EPC into lawful EPC models. In *Business Process Management Workshops, BPM 2005 International Workshops, BPI, BPD, ENEI, BPRM, WSCOBPM, BPS, Nancy, France, September 5, 2005, Revised Selected Papers*, pages 497–511, 2005.
- [129] Irina Rychkova and Selmin Nurcan. Towards adaptability and control for knowledge-intensive business processes: Declarative configurable process specifications. In *44th Hawaii International International Conference on Systems Science (HICSS-44 2011), Proceedings, 4-7 January 2011, Koloa, Kauai, HI, USA*, pages 1–10, 2011.
- [130] Florian Gottschalk, Wil M. P. van der Aalst, Monique H. Jansen-Vullers, and Marcello La Rosa. Configurable workflow models. *Int. J. Cooperative Inf. Syst.*, 17(2):177–221, 2008.
- [131] August-Wilhelm Scheer and Markus Nüttgens. ARIS architecture and reference models for business process management. In *Business Process Management, Models, Techniques, and Empirical Studies*, pages 376–389, 2000.
- [132] OMG. Unified modelling language (uml). <http://www.omg.org/spec/uml/2.3/>.
- [133] Alena Hallerbach, Thomas Bauer, and Manfred Reichert. Managing process variants in the process life cycle. Technical report, University of Twente, 2007.

- [134] Manfred Reichert, Alena Hallerbach, and Thomas Bauer. Lifecycle management of business process variants. In *Handbook on Business Process Management 1, Introduction, Methods, and Information Systems, 2nd Ed.*, pages 251–278. 2015.
- [135] Florian Gottschalk, Wil M. P. van der Aalst, and Monique H. Jansen-Vullers. Mining reference process models and their configurations. In *On the Move to Meaningful Internet Systems: OTM 2008 Workshops, OTM Confederated International Workshops and Posters, ADI, AWeSoMe, COMBEK, EI2N, IWSSA, MONET, OnToContent + QSI, ORM, PerSys, RDDS, SEMELS, and SWWS 2008, Monterrey, Mexico, November 9-14, 2008. Proceedings*, pages 263–272, 2008.
- [136] Michael Arias, Eric Rojas, Jorge Munoz-Gama, and Marcos Sepúlveda. A framework for recommending resource allocation based on process mining. In *Business Process Management Workshops - BPM 2015, 13th International Workshops, Innsbruck, Austria, August 31 - September 3, 2015, Revised Papers*, pages 458–470, 2015.
- [137] W. M. P. van der Aalst. Petri net based scheduling. *Operations-Research-Spektrum*, 18(4):219–229, 1996.
- [138] Pinar Senkul and Ismail Hakki Toroslu. An architecture for workflow scheduling under resource allocation constraints. *Inf. Syst.*, 30(5):399–422, 2005.
- [139] Zhengxing Huang, Wil M. P. van der Aalst, Xudong Lu, and Huilong Duan. Reinforcement learning based resource allocation in business process management. *Data Knowl. Eng.*, 70(1):127–145, 2011.
- [140] Kahina Bessai and François Charoy. Business process tasks-assignment and resource allocation in crowdsourcing context. In *2nd IEEE International Conference on Collaboration and Internet Computing, CIC 2016, Pittsburgh, PA, USA, November 1-3, 2016*, pages 11–18, 2016.
- [141] Amina Ahmed-Nacer, Kahina Bessai, Samir Youcef, and Claude Godart. A multi-criteria based approach for web service selection using quality of service (qos). In *2015 IEEE International Conference on Services Computing, SCC 2015, New York City, NY, USA, June 27 - July 2, 2015*, pages 570–577, 2015.
- [142] Fangzhe Chang, Jennifer Ren, and Ramesh Viswanathan. Optimal resource allocation in clouds. In *IEEE International Conference on Cloud Computing, CLOUD 2010, Miami, FL, USA, 5-10 July, 2010*, pages 418–425, 2010.
- [143] Wanneng Shu, Wei Wang, and Yunji Wang. A novel energy-efficient resource allocation algorithm based on immune clonal optimization for green cloud computing. *EURASIP J. Wireless Comm. and Networking*, 2014:64, 2014.

- [144] Hwa-Min Lee, Young-Sik Jeong, and Haeng Jin Jang. Performance analysis based resource allocation for green cloud computing. *The Journal of Supercomputing*, 69(3):1013–1026, 2014.
- [145] Fangzhe Chang, Jennifer Ren, and Ramesh Viswanathan. Optimal resource allocation for batch testing. In *Second International Conference on Software Testing Verification and Validation, ICST 2009, Denver, Colorado, USA, April 1-4, 2009*, pages 91–100, 2009.
- [146] Molka Rekik, Khouloud Boukadi, Nour Assy, Walid Gaaloul, and Hanène Ben-Abdallah. A linear program for optimal configurable business processes deployment into cloud federation. In *IEEE International Conference on Services Computing, SCC 2016, San Francisco, CA, USA, June 27 - July 2, 2016*, pages 34–41, 2016.
- [147] Wei Zhe Low, Jochen De Weerd, Moe Thandar Wynn, Arthur H. M. ter Hofstede, Wil M. P. van der Aalst, and Seppe K. L. M. vanden Broucke. Perturbing event logs to identify cost reduction opportunities: A genetic algorithm-based approach. In *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2014, Beijing, China, July 6-11, 2014*, pages 2428–2435, 2014.
- [148] Wei Zhe Low, Seppe K. L. M. vanden Broucke, Moe Thandar Wynn, Arthur H. M. ter Hofstede, Jochen De Weerd, and Wil M. P. van der Aalst. Revising history for cost-informed process improvement. *Computing*, 98(9):895–921, 2016.
- [149] Moe Thandar Wynn, Hajo A. Reijers, Michael Adams, Chun Ouyang, Arthur H. M. ter Hofstede, Wil M. P. van der Aalst, Michael Rosemann, and Zahirul Hoque. Cost-informed operational process support. In *Conceptual Modeling - 32th International Conference, ER 2013, Hong-Kong, China, November 11-13, 2013. Proceedings*, pages 174–181, 2013.
- [150] Vasa Curcin, Thomas Woodcock, Alan J. Poots, Azeem Majeed, and Derek Bell. Model-driven approach to data collection and reporting for quality improvement. *Journal of Biomedical Informatics*, 52:151 – 162, 2014. Special Section: Methods in Clinical Research Informatics.
- [151] Moe Thandar Wynn, Wei Zhe Low, Arthur H. M. ter Hofstede, and Wiebe Nauta. A framework for cost-aware process management: Cost reporting and cost prediction. *J. UCS*, 20(3):406–430, 2014.
- [152] Elio Goettelmann, Karim Dahman, Benjamin Gâteau, and Claude Godart. A formal broker framework for secure and cost-effective business process deployment on multiple clouds. In *Information Systems Engineering in Complex Environments - CAiSE Forum 2014, Thessaloniki, Greece, June 16-20, 2014, Selected Extended Papers*, pages 3–19, 2014.

- [153] Ralph Mietzner, Andreas Metzger, Frank Leymann, and Klaus Pohl. Variability modeling to support customization and deployment of multi-tenant-aware software as a service applications. In *International ICSE Workshop on Principles of Engineering Service-Oriented Systems, PESOS 2009, 18-19 May 2009, Vancouver, BC, Canada*, pages 18–25, 2009.
- [154] Marty Poniatowski. *Foundations of Green IT: Consolidation, Virtualization, Efficiency, and ROI in the Data Center*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1st edition, 2009.
- [155] Aditya Ghose, Konstantin Hoesch-Klohe, Lothar Hinsche, and Lam-Son Lê. Green business process management: a research agenda. *Australasian J. of Inf. Systems*, 16(2), 2009.
- [156] Alexander Nowak, Frank Leymann, and Ralph Mietzner. Towards green business process reengineering. In *Service-Oriented Computing - ICSOC 2010 International Workshops, PAASC, WESOA, SEE, and SOC-LOG, San Francisco, CA, USA, December 7-10, 2010, Revised Selected Papers*, pages 187–192, 2010.
- [157] Konstantin Hoesch-Klohe, Aditya Ghose, and Lam-Son Lê. Towards green business process management. In *2010 IEEE International Conference on Services Computing, SCC 2010, Miami, Florida, USA, July 5-10, 2010*, pages 386–393, 2010.
- [158] Konstantin Hoesch-Klohe and Aditya K. Ghose. Carbon-aware business process design in abnoba. In *Service-Oriented Computing - 8th International Conference, ICSOC 2010, San Francisco, CA, USA, December 7-10, 2010. Proceedings*, pages 551–556, 2010.
- [159] Alexander Nowak, Frank Leymann, David Schumm, and Branimir Wetzstein. An architecture and methodology for a four-phased approach to green business process reengineering. In *Information and Communication on Technology for the Fight against Global Warming - First International Conference, ICT-GLOW 2011, Toulouse, France, August 30-31, 2011. Proceedings*, pages 150–164, 2011.
- [160] Alexander Nowak, Frank Leymann, Daniel Schleicher, David Schumm, and Sebastian Wagner. Green business process patterns. In *Proceedings of the 18th Conference on Pattern Languages of Programs, PLoP 2011, Portland, Oregon, USA, October 21-23, 2011*, pages 6:1–6:10, 2011.
- [161] Cinzia Cappiello, Sumit Datre, Mariagrazia Fugini, Paco Melia, Barbara Pernici, Pierluigi Plebani, Michael Gienger, and Axel Tenschert. Monitoring and assessing energy consumption and CO2 emissions in cloud-based systems. In *IEEE International Conference on Systems, Man, and Cybernetics, Manchester, SMC 2013, United Kingdom, October 13-16, 2013*, pages 127–132, 2013.

- [162] Alexander Nowak, Tobias Binz, Christoph Fehling, Oliver Kopp, Frank Leymann, and Sebastian Wagner. Pattern-driven green adaptation of process-based applications and their runtime infrastructure. *Computing*, 94(6):463–487, 2012.
- [163] Alexander Nowak, Frank Leymann, and David Schumm. The differences and commonalities between green and conventional business process management. In *IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing, DASC 2011, 12-14 December 2011, Sydney, Australia*, pages 569–576, 2011.
- [164] Emna Hachicha, Karn Yongsiriwit, and Walid Gaaloul. Energy efficient configurable resource allocation in cloud-based business processes (short paper). In *On the Move to Meaningful Internet Systems: OTM 2016 Conferences - Confederated International Conferences: CoopIS, C&TC, and ODBASE 2016, Rhodes, Greece, October 24-28, 2016, Proceedings*, pages 437–444, 2016.
- [165] Marco Brambilla, Piero Fraternali, and Carmen Karina Vaca Ruiz. Combining social web and BPM for improving enterprise performances: the bpm4people approach to social BPM. In *Proceedings of the 21st World Wide Web Conference, WWW 2012, Lyon, France, April 16-20, 2012 (Companion Volume)*, pages 223–226, 2012.
- [166] Zakaria Maamar and al. Social engineering of communities of web services. In *11th SAINT Germany, 18-21 July, Proceedings*, pages 100–109, 2011.
- [167] Zakaria Maamar and al. Towards a framework for weaving social networks into mobile commerce. *Int. J. Syst. Serv.-Oriented Eng.*, 2(3):32–46, July 2011.
- [168] Zakaria Maamar and al. SUPER: Social-based Business Process Management Framework. In *ICSOC*, 2014.
- [169] Emna Hachicha and Walid Gaaloul. Towards resource-aware business process development in the cloud. In *29th IEEE International Conference on Advanced Information Networking and Applications, AINA 2015, Gwangju, South Korea, March 24-27, 2015*, pages 761–768, 2015.
- [170] Emna Hachicha, Walid Gaaloul, and Zakaria Maamar. Social-based semantic framework for cloud resource management in business processes. In *IEEE International Conference on Services Computing, SCC 2016, San Francisco, CA, USA, June 27 - July 2, 2016*, pages 443–450, 2016.
- [171] Joos C. A. M. Buijs, Boudewijn F. van Dongen, and Wil M. P. van der Aalst. Towards cross-organizational process mining in collections of process models and their executions. In *Business Process Management Workshops - BPM 2011 International Workshops, Clermont-Ferrand, France, August 29, 2011, Revised Selected Papers, Part II*, pages 2–13, 2011.

- [172] Steffen Kächele, Christian Spann, Franz J. Hauck, and Jörg Domaschka. Beyond iaas and paas: An extended cloud taxonomy for computation, storage and networking. In *IEEE/ACM 6th International Conference on Utility and Cloud Computing, UCC 2013, Dresden, Germany, December 9-12, 2013*, pages 75–82, 2013.
- [173] Andy Edmonds, Thijs Metsch, Alexander Papaspyrou, and Alexis Richardson. Toward an open cloud standard. *IEEE Internet Computing*, 16(4):15–25, 2012.
- [174] George Papamarkos, Alexandra Poulouvasilis, and Peter T. Wood. Event-condition-action rules on RDF metadata in P2P environments. *Computer Networks*, 50(10):1513–1532, 2006.
- [175] Luis Jesús Ramón Stroppi and al. Extended resource perspective support for bpmn and bpel. In *Proceedings of the XV Iberoamerican Conference on Software Engineering'12*.
- [176] Luis Jesús Ramón Stroppi and al. A bpmn 2.0 extension to define the resource perspective of bp models. In *CIbSE 2011*.
- [177] David Hollingsworth. Workflow management coalition the reference model. 1995.
- [178] Alexandra Poulouvasilis and al. E-c-a rule languages for the semantic web. In *CT in Database Technology-Workshops and Reactivity on the Web'06*.
- [179] Alexander Maedche and Steffen Staab. Measuring similarity between ontologies. In *Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web, 13th International Conference, EKAW 2002, Sigüenza, Spain, October 1-4, 2002, Proceedings*, pages 251–263, 2002.
- [180] R. Porzel and R. Malaka. A task-based approach for ontology evaluation. In *Proc. of ECAI 2004 Workshop on Ontology Learning and Population*, Valencia, Spain, August 2004.
- [181] Marta Sabou and Miriam Fernández. Ontology (network) evaluation. In *Ontology Engineering in a Networked World.*, pages 193–212. 2012.
- [182] Super Specification: sbpmn and sepc to bpmo traslation. <http://www.ip-super.org/res/deliverables/m24/d4.5.pdf>.
- [183] Virtuoso jena provider. <http://virtuoso.openlinksw.com/dataspace/doc/dav/wiki/main/virtjenaprovider>.
- [184] Emna Hachicha, Nour Assy, Walid Gaaloul, and Jan Mendling. A configurable resource allocation for multi-tenant process development in the cloud.

- In *Advanced Information Systems Engineering - 28th International Conference, CAiSE 2016, Ljubljana, Slovenia, June 13-17, 2016. Proceedings*, pages 558–574, 2016.
- [185] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, and Matei Zaharia. A view of cloud computing. *Commun. ACM*, 53(4):50–58, April 2010.
- [186] Mariam Rady. Parameters for service level agreements generation in cloud computing - A client-centric vision. In *Advances in Conceptual Modeling - ER 2012 Workshops CMS, ECDM-NoCoDA, MoDIC, MORE-BI, RIGiM, SeCoGIS, WISM, Florence, Italy, October 15-18, 2012. Proceedings*, pages 13–22, 2012.
- [187] S. N. Sivanandam and S. N. Deepa. *Introduction to genetic algorithms*. Springer, 2008.
- [188] Darrell Whitley and Andrew M. Sutton. Genetic algorithms - A survey of models and methods. In *Handbook of Natural Computing*, pages 637–671. 2012.
- [189] Jan Recker, Michael Rosemann, and Ehsan Roohi Gohar. Measuring the carbon footprint of business processes. In *Business Process Management Workshops - BPM 2010 International Workshops and Education Track, Hoboken, NJ, USA, September 13-15, 2010, Revised Selected Papers*, pages 511–520, 2010.
- [190] Christos H. Papadimitriou and Kenneth Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, 1982.
- [191] Jorge Cardoso. Evaluating the process control-flow complexity measure. In *IEEE ICWS '05*.
- [192] Jorge Cardoso, Jan Mendling, Gustaf Neumann, and Hajo A. Reijers. A discourse on complexity of process models. In *Business Process Management Workshops '06*.
- [193] Beate List and Birgit Korherr. An evaluation of conceptual business process modelling languages. In *Proceedings of the 2006 ACM Symposium on Applied Computing (SAC), Dijon, France, April 23-27, 2006*, pages 1532–1539, 2006.
- [194] Hajo A. Reijers and Irene T. P. Vanderfeesten. Cohesion and coupling metrics for workflow process design. In *Business Process Management: Second International Conference, BPM 2004, Potsdam, Germany, June 17-18, 2004. Proceedings*, pages 290–305, 2004.

-
- [195] Jan Vogelaar, H. M. W. Verbeek, B. Luka, and Wil M. P. van der Aalst. Comparing business processes to determine the feasibility of configurable models: A case study. In *Business Process Management Workshops - BPM 2011 International Workshops, Clermont-Ferrand, France, August 29, 2011, Revised Selected Papers, Part II*, pages 50–61, 2011.
- [196] Srinivas Talluri and R. C. Baker. A multi-phase mathematical programming approach for effective supply chain design. *European Journal of Operational Research*, 141(3):544–558, 2002.
- [197] Gerhard Keller and Thomas Teufel. *Sap R/3 Process Oriented Implementation*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition, 1998.