TECHNISCHE UNIVERSITÄT BERLIN

A Generic Framework for Heterogeneous Resource Federation

von Diplom-Ingenieur
Sebastian Wahle

Von der Fakultät IV - Elektrotechnik und Informatik
der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften
- Dr.-Ing. -

genehmigte Dissertation

Promotionsausschuss
Vorsitzender: Prof. Dr. Axel Küpper (Technische Universität Berlin)
Gutachter: Prof. Dr. Thomas Magedanz (Technische Universität Berlin)
Gutachter: Prof. Dr. Paul Müller (Technische Universität Kaiserslautern)
Gutachter: Prof. Dr. Kurt Tutschku (Universität Wien)

Tag der wissenschaftlichen Aussprache: 15.11.2011

Berlin 2011
D 83

A Generic Framework for Heterogeneous Resource Federation

von Diplom-Ingenieur
Sebastian Wahle

Von der Fakultät IV - Elektrotechnik und Informatik
der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften
- Dr.-Ing. -

genehmigte Dissertation

Technische Universität Berlin
Fakultät IV - Elektrotechnik und Informatik
Lehrstuhl Architektur der Vermittlungsknoten
Franklinstrasse 28-29, 10587 Berlin

First Edition
Berlin, November 22, 2011

Supervisors:
Prof. Dr. Thomas Magedanz, thomas.magedanz@tu-berlin.de
Technische Universität Berlin, Lehrstuhl Architektur der Vermittlungsknoten
Franklinstraße 28-29, 10587 Berlin

Prof. Dr. Paul Müller, pmueller@informatik.uni-kl.de
Technische Universität Kaiserslautern, Fachbereich Informatik, AG ICSY
Paul-Ehrlich-Straße, Gebäude 34, 67653 Kaiserslautern

Prof. Dr. Kurt Tutschku, kurt.tutschku@univie.ac.at
Universität Wien, Future Communication, Fakultät für Informatik
Universitätsstraße 10/T11, A-1090 Vienna, Austria

To my wonderful sons Niklas and Maximilian.

Abstract

This doctorate thesis introduces a framework for information and communication technology resource federation. Resource federation in the context of this work is a concept for sharing and allowing access to heterogeneous resources across multiple administrative domains. Three design artifacts are delivered as the main research result: a federation model, a federation methodology, and a system instantiation.

The artifacts abstract from the types of resources to be federated. Examples of resources include general-purpose physical and virtual machines, software packages and systems, as well as specialized proprietary devices. Resource capabilities are exposed in a service oriented manner and can be assembled in meaningful combinations and configurations to satisfy the service consumer's requirements.

One of the potential user segments is the scientific community where resources are requested for running large scale research experiments that require a certain setup of heterogeneous resources. The benefit provided to users of the federation is the seamless access to a large collection of resources via unified interfaces. Resource providers benefit from an additional sales channel, increasing their average resource utilization, some sort of compensation for the provided services, and the access to additional user groups. A federation organization maintains relationships with all stakeholders and offers central services to ensure the overall operation.

Zusammenfassung

Die vorliegende Arbeit betrachtet ein generisches Framework zur Föderation von Ressourcen der Informations- und Kommunikationstechnologie. “Generic resource federation” ist, im Kontext dieser Arbeit, ein Konzept zur Bereitstellung und Freigabe heterogener Ressourcen über die Grenzen administrativer Domänen hinweg. Die hier präsentierte Forschungsleistung setzt sich aus der Entwicklung und Bereitstellung—dem Design—eines generischen Föderationsmodells, eines Satzes von Methoden zur flexiblen Föderation von Ressourcen, sowie einer Systeminstanziierung zusammen.

Hierbei abstrahieren die drei oben genannten “design artifacts” vom Typus föderierter Ressourcen. Beispiele verschiedener Ressourcentypen, die mit dem System föderiert werden können, sind physikalische und virtuelle Rechner, Softwarepakete und -systeme, sowie Spezialgeräte. Die Eigenschaften und Einsatzmöglichkeiten einzelner Ressourcen werden dienstorientiert zur Verfügung gestellt und können flexibel kombiniert werden, um den Anforderungen des Nutzers gerecht zu werden.

Eines der potentiellen Anwendungsgebiete der Ergebnisse ist die wissenschaftliche Forschungsgemeinschaft, die verschiedenste Ressourcen für die Durchführung umfangreicher Experimente benötigt. Der Mehrwert für die Nutzer des beschriebenen Systems ist der einfache Zugang zu einer Vielzahl unterschiedlicher Ressourcen mittels einheitlicher Schnittstellen. Die Anbieter von Ressourcen profitieren durch einen weiteren Vertriebskanal, eine erhöhte durchschnittliche Ressourcenauslastung, die Kompensierung ihrer angebotenen Dienste, sowie die Erschließung zusätzlicher Nutzergruppen. Eine Föderationsorganisation vertritt die unterschiedlichen Interessensgruppen nach außen und bietet zentrale Dienstleistungen, um einen fortlaufenden Betrieb zu gewährleisten.

Preface

Collective resource sharing, as for example driven by projects like SETI@home, has always been among the things that fascinated me. When I joined the Next Generation Network Infrastructures group at Fraunhofer FOKUS as a student in 2005, our team was providing IP Multimedia Subsystem (IMS) testbeds to customers worldwide and it was at that time that I began to understand the importance of testbeds and experimental facilities. However, the short lifetime of experimental infrastructure, the limited sharing of resources across projects, and the associated deployment and maintenance cost were astonishing to me. Also, I realized that today, in industrialized nations, the electric energy consumption of the Information and Communication Technology (ICT) sector is—as recent studies show—around 10% of the total consumption, which is a considerabe fraction. Those facts, combined with my university background as an industrial engenieer and my general interest in the field, led me to the conclusion that there is a demand for better solutions regarding collective resource sharing beyond the federation of Central Processing Unit (CPU) cycles.

It was this line of thinking that raised my interest in the Pan-European Laboratory (Panlab) project funded by the European Commission as part of the Sixth Framework Programme (FP6). In 2007, I took over the Panlab project lead for Fraunhofer FOKUS from a colleague and successfully completed the project with the Panlab team. In 2008, together with a set of core partners, we prepared a Panlab follow-on integrating project proposal for the Seventh Framework Programme (FP7) where I heavily contributed with my ideas about heterogeneous resource federation. The project got accepted and as a work package leader, I was responsible for the Teagle framework design and implementation. This provided a great foundation for bringing my vision of collective sharing of heterogeneous resources into life.

Looking back now, for the past 4 years I have been working with talented and inspiring people. In addition to my responsibilities regarding the Panlab projects, I headed the Evolving Infrastructures and Services group at Fraunhofer FOKUS supervising a team of scientist and students active in several scientific projects, and was a member of the Future Internet Research and Experimentation (FIRE) architecture board.

I owe a debt of gratitude to a number of people that assisted me during this journey and supported me in completing my PhD work. I would like to thank my supervisors Prof. Dr. Thomas Magedanz, Prof. Dr. Paul Müller, and Prof. Dr. Kurt Tutschku for the possibility to write this thesis and the guidance that they gave.

Also, I would like to thank Bogdan Harjoc and Konrad Campowsky for their excellent work regarding the Teagle implementation, as well as Stefan Harder and Nils Lüdicke for their support on the maintenance of the Teagle live installation and the different playgrounds.

Furthermore, I would like to thank Anastasius Gavras, Christos Tranoris, Shane Fox, Eamonn Power, Mariano Belaunde, and Konstantinos Koutsopoulos from the Panlab team for their contributions to the Teagle design and implementation as well as the numerous papers that we authored together.

I deeply thank Maria for her love and understanding, all the sleepless nights taking care of our sons, and for being as special to me as you are. You have an eternal place in my heart.

Also, special thanks goes to my parents, Claudia and Wolf-Dietrich Wahle. It is your impressive dedication and selfless devotion to your kids' well-being that enabled me to get this far.

Last but not least I would like to thank the European Commission for funding the FIRE initiative including the Panlab projects as well as the "Elterngeldstelle" for providing the "Elterngeld" that allowed me to take a few month off to concentrate on finishing this work.

Berlin, August 2011

Sebastian Wahle

Contents

Abstract vii

Zusammenfassung ix

Preface xi

List of Figures xvii

List of Tables xxi

1 Introduction 1
- 1.1 Motivation . . . 1
- 1.2 Scope . . . 3
  - 1.2.1 Definitions . . . 3
  - 1.2.2 Taxonomy . . . 4
- 1.3 Research Hypothesis . . . 5
- 1.4 Scientific Classification and Research Methodology . . . 7
- 1.5 Structure . . . 9

2 Federation Challenges 13
- 2.1 Federation Challenges Overview . . . 13
- 2.2 Categorization of Challenges and Requirements . . . 15
  - 2.2.1 User Challenges . . . 16
  - 2.2.2 Resource Provider Challenges . . . 18
  - 2.2.3 Federation Organization Challenges . . . 18
  - 2.2.4 Non-Functional Challenges . . . 20

3 State of the Art 21
- 3.1 Service Orientation as a Fundamental Design Pattern . . . 21
- 3.2 Infrastructure as a Service . . . 23
- 3.3 Cloud Computing . . . 24
  - 3.3.1 Overview . . . 24

3.3.2 Cloud Characteristics . . . 25
3.3.3 Standardization . . . 27
3.4 Grid Computing . . . 31
3.4.1 Overview . . . 31
3.4.2 Standardization . . . 32
3.5 Federated Identity Management . . . 33
3.6 Future Internet Experimental Facilities . . . 36
3.7 GENI Initiative . . . 39
3.7.1 PlanetLab . . . 41
3.7.2 The SFA Versions . . . 42
3.7.3 The ORCA-BEN Project . . . 44
3.7.4 ORBIT and OMF . . . 46
3.7.5 ProtoGENI and Emulab . . . 48
3.8 FIRE Initiative . . . 49
3.8.1 Onelab2 and PlanetLab Europe . . . 49
3.8.2 The FP6 Panlab and FP7 PII Projects . . . 50
3.8.3 Wisebed and the Open Federation Alliance . . . 52
3.9 G-Lab . . . 53
3.10 Summary . . . 54

4 Resource Federation Model **57**
4.1 Model Entities . . . 57
4.2 Federation Roles . . . 61
4.3 Federation Modes . . . 63
4.3.1 Identity Federation . . . 63
4.3.2 Control Framework Federation . . . 64
4.3.3 Federated Resource Description . . . 65
4.3.4 Federated Policy Description . . . 67
4.4 Federation Scenarios . . . 68
4.4.1 Central Operation . . . 68
4.4.2 Distributed Operation . . . 69
4.4.3 Recursive Scenario . . . 70
4.4.4 Consortium Operation . . . 71
4.5 Summary . . . 72

5 Resource Federation Methodology **73**
5.1 Information Model & Resource Description . . . 73
5.1.1 Resource Types and Instances . . . 74
5.1.2 Resource Naming Conventions . . . 75
5.1.3 Resource Configurations Grammar . . . 75
5.2 Resource Abstraction & Control Framework . . . 77
5.2.1 The Control Framework Service Primitives . . . 77
5.2.2 Resource Abstraction . . . 79
5.3 Resource Relationships . . . 81
5.3.1 Resource Hierarchies . . . 81
5.3.2 Configuration References . . . 83

5.4 Resource Orchestration & Provisioning 84
5.5 Cross-domain Resource Interconnection 88
5.6 Summary 90

6 Resource Federation Framework Instantiation **91**
6.1 Requirements 91
6.1.1 Notation 91
6.1.2 GRL-Based Stakeholder Requirements Definition and Analysis 94
6.2 Federation Framework Design Decisions 106
6.3 UCM-Based Use Case Analysis 107
6.3.1 Notation 107
6.3.2 Overview 109
6.3.3 The VCT Tool Startup Process 111
6.3.4 The VCT Design Process 112
6.3.5 Initiating the VCT Deployment 113
6.3.6 The Resource Orchestration Process 114
6.3.7 Interfacing With the Domain Manager Layer 114
6.3.8 Domain Layer Resource Management 115
6.4 System Overview and Reference Points 117
6.5 The Federation Layer 118
6.5.1 Teagle Architecture 119
6.5.2 Portal 120
6.5.3 Repository 122
6.5.4 VCT Tool 125
6.5.5 Policy Engine 128
6.5.6 Request Processor 133
6.5.7 Orchestration Engine 135
6.5.8 Gateway 138
6.6 The Domain Layer 139
6.6.1 Domain Manager Framework Architecture 140
6.6.2 Resources and Resource Adapters 141
6.6.3 Resource Adapter Description 143
6.7 Cross-Domain Interconnectivity 144
6.8 Summary 147

7 Evaluation **149**
7.1 Observational Evaluation: Field Study 149
7.1.1 The FP7 PII Project 150
7.1.2 The FP7 TEFIS Project 153
7.1.3 The FP7 BonFIRE Project 157
7.1.4 The FP7 NOVI Project 159
7.1.5 The BMBF G-Lab and G-Lab Deep Projects 160
7.1.6 The FP7 OpenLab Project 161
7.1.7 Field Study Summary and Further Analysis 162
7.2 Experimental Evaluation: A Controlled Experiment 165
7.2.1 First Experiment Run 166

7.2.2 Second Experiment Run . . . 170
7.2.3 Summary . . . 172
7.3 Comparison with Other Approaches . . . 173

8 Conclusion **177**
8.1 Conclusion and Impact . . . 179
8.2 Outlook . . . 181

A VCT Design and Provisioning **185**
A.1 Requesting Data from the Teagle Repository . . . 185
A.2 Virtual Resource Grouping Design and Configuration . . . 196
A.3 Booking the Virtual Resource Grouping . . . 202
A.4 Virtual Resource Grouping Deployment . . . 203
A.5 T_1 and Intra Domain Communication . . . 208

B Policy Evaluation Example **217**
B.1 Policy Definition . . . 217
B.2 Policy Evaluation and Enforcement . . . 220

C Information Model **225**

D Reference Point T1 Specifications **229**

E The RADL Meta Model and Syntax **233**

F VCT Specification & REP XML Schemata **235**

G Typographical Conventions **251**
G.1 Literature References . . . 251
G.2 Footnotes . . . 252
G.3 Acronyms . . . 252
G.4 Source Code and Formal Syntax Notation . . . 252
G.5 Quotations . . . 252
G.6 Important Terms . . . 253
G.7 Gender Considerations . . . 253

Acronyms **xxiii**

Bibliography **xxix**

List of Figures

1.1 Motivation for federation 2
1.2 A high level view on the main aspects and terminology of resource federation 5
1.3 Business, organizational, information technology, and information system boundaries 8
1.4 Information system research framework 8
1.5 Thesis chapters overview 10

2.1 Federation stakeholders and relationships. The different stakeholder roles allow to group challenges and requirements accordingly. 16

3.1 Primary roles in a service oriented architectural design 22
3.2 Principal concepts of the SOA reference model defined by OASIS 22
3.3 The cloud stack showing all layers of the XaaS concept: Infrastructure, Platform, and Software as a Service 24
3.4 High level conceptual overview of grid computing 31
3.5 Collaborative model – central operation and governance of a CoT by a single governing organization 33
3.6 Left: Consortium model – A multi-party contract between the founding organizations determines rules and governance of the CoT. Right: Centralized Model – A single founding organization contracts other CoT members where the central entity can broker n-party relationships. 34
3.7 Federated identity management roles and relationships 35
3.8 Examples of FDIM terminology and the mapping from an identity hold by a *real person* to a digital identity represented by a *persona* 36
3.9 The different attribute types that can be associated to a person are temporary, persistent, intrinsic, and extrinsic attributes. 37
3.10 GENI system overview 39
3.11 The second GENI spiral provided a better interoperability of the clusters and their control frameworks. 41
3.12 PlanetLab node architecture and the concept of slicing distributed nodes . . . 42

3.13 The ORCA actors and interactions with arrows illustrating the leasing protocols. 45
3.14 The OMF architecture 46
3.15 High level view on the ProtoGENI federation approach 48
3.16 The Panlab roles include a Panlab office, a Panlab partner, and a Panlab customer role. The Teagle framework provides a set of middleware components to control the distributed testbed resources. 50
3.17 Overview of the Wisebed client, federator, and testbed backend interaction . 52
3.18 The G-Lab vision: Future Internet research studies are accompanied by experimentation performed on the G-Lab facility. Through the tight coupling of research and facility operation, the platform might evolve into the Future Internet itself. 53

4.1 Overview of the federation model entities: SET, M, REG, R, GW. The figure also shows the abstract concepts of a domain, and a virtual resource grouping that may span the border of several domains. 58
4.2 Gateways in domain A and domain B provide inter-domain resource connectivity. 60
4.3 The federation stakeholder roles: a user interested in resource services offered by resource providers interacts with the providers that are collectively represented by a federation organization. All stakeholder groups engage in legal, operational, and technical relationships. 61
4.4 Identity Federation 64
4.5 Control framework federation: harmonized domain manager interfaces 65
4.6 Harmonized resource description 65
4.7 Policy federation 67
4.8 Operation by a central organization: authentication and authorization is aligned. Also, the federation relies on a common control framework and resource model. A central registry simplifies resource discovery and the support of advanced tools. 68
4.9 In distributed scenarios, less federation relevant aspects are harmonized across the federation stakeholders compared with the central operation. 69
4.10 The recursive scenario with multiple federation levels n. In case of $n = 1$, the subscripted index has been omitted. 71

5.1 An information model allows to structure data across the federation 74
5.2 Resource model: resource types, instances, and their relation with virtual resource groupings 76
5.3 Resource model: key-value pairs are used to express the resource configuration options and the actual configuration data values. 77
5.4 Resource abstraction method: the reference points T_1 and T_2 80
5.5 Resource relationships 86
5.6 From a virtual resource grouping specification to resource orchestration and provisioning 87
5.7 Interconnecting resources provided by federated domains using virtual overlay networking techniques 89

6.1 The URN abstract grammar given as a UML model including links, metadata, and the GRL and UCM model element classes. 92
6.2 The GRL intentional element type symbols used in GRL based figures 92

6.3 The GRL abstract grammar given as a UML model including the important intentional elements: actor, goal, softgoal, and task. 93
6.4 The contribution types and their respective symbols 94
6.5 Overview of the most important stakeholder goals and correlations with the highest impact on the system design 95
6.6 GRL evaluation: the information model impact 99
6.7 GRL evaluation: the control framework impact 101
6.8 GRL evaluation: the impact of identity management 103
6.9 GRL evaluation: the policy impact 105
6.10 The UCM abstract grammar 107
6.11 The UCM path and path nodes abstract grammar 108
6.12 A high level use case map showing the path for deriving a provisioned virtual resource grouping across multiple stubs associated to the different Teagle system components. 109
6.13 The stub *start VCT tool*: the successful VCT tool startup path 111
6.14 The stub *start VCT tool*: error while loading resource information 111
6.15 The stub *design VCT spec*: successful virtual resource grouping specification design 112
6.16 The stub *initiate VCT deployment*: the request processor forwards the VCT specification to the orchestration engine. 113
6.17 The stub *deploy VCT*: resource orchestration workflow generation and execution 114
6.18 The stub *forward requests*: the Teagle gateway shall act as single point of communication between the federation layer (the Teagle system) and the domain layer (the domain managers). 115
6.19 The stub *CRUD resources*: successful domain layer resource configuration results in interactions between the domain manager, the resource adapter, and the resource. 116
6.20 Framework instantiation overview and reference points 117
6.21 The Teagle system 119
6.22 VCT design using the VCT tool – the opened resource configuration dialog for the resource `XenNode-756406707` (a virtual machine) allows to specify the number of CPUs and the amount of memory to be available to this virtual machine. 125
6.23 The booking process from an operational and legal point of view 128
6.24 The Teagle portal policy specification overview page 129
6.25 The Teagle portal policy evaluation overview page 131
6.26 The orchestration engine system components and interfaces 136
6.27 An example structure of a state machine script used for the parallel execution of resource provisioning tasks 138
6.28 The domain manager implementation architecture 140
6.29 The reference point T_2 concept and instantiation 143
6.30 The use of the resource adapter language and the EC2 example resource adapter 144
6.31 The interconnection gateway instantiation exposing interfaces on the reference points T_2, I_1, I_2, and U_2 145
6.32 VLAN-based collision domain isolation 147

7.1 The TEFIS high level architecture . . . 154
7.2 The TEFIS domains . . . 155
7.3 Open call results demonstrating the impact on the European ICT research community. In both sub-figures the ordinate plots the number of submitted proposals. . . . 156
7.4 The BonFIRE domains and the respective resources provided to the facility . 157
7.5 The first release of the BonFIRE architecture . . . 158
7.6 Recursive federation approach building upon Teagle, PLC, and SFAv2 . . . 160
7.7 Architecture for federating PLC/SFAv2 based resources with Teagle-controlled domains . . . 161
7.8 Usage of the different federation methods and scenarios described by this thesis. The abscissa plots the section where the respective method/scenario is described. The ordinate of both sub-figures plots the corresponding number of projects as revealed by this field study. . . . 165
7.9 Controlled experimental evaluation: the initial virtual resource grouping design 166
7.10 Stress-test metrics: the abscissa of all sub-figures plots the elapsed time (ISO 8601 extended format: [hh]:[mm]:[ss]). The ordinate plots the respective measurement. . . . 169
7.11 Controlled experimental evaluation: the *modified* virtual resource grouping design. The migrated CSCF services are now hosted on an OCCI-based cloud resource provided by a second domain. . . . 170
7.12 Stress-test metrics of the second run: the abscissa of all sub-figures plots the elapsed time (ISO 8601 extended format: [hh]:[mm]:[ss]). The ordinate plots the respective measurement. Subfigure (e) plots the CPU load of the new cloud resource hosting the migrated PCSCF and SCSCF services. . . . 172

8.1 Thesis impact . . . 180
8.2 Outlook: Cross-domain and cross-layer federation applied to the NGN field . 183

A.1 The VCT tool showing the virtual resource grouping before the booking has been initiated . . . 196
A.2 Abstract representation of the virtual resource grouping shown by the VCT tool in figure A.1 . . . 197
A.3 VCT tool showing the virtual resource grouping after successful resource provisioning. The resource instance IDs have changed from design time IDs to run time IDs. . . . 215

B.1 The policy overview page on the Teagle portal . . . 217
B.2 The policy definition page on the Teagle portal . . . 218
B.3 The VCT tool acting as policy a enforcement point . . . 220

C.1 The key classes of the information model . . . 226
C.2 The full information model . . . 227
C.3 The policy model . . . 228

E.1 The RADL meta-model . . . 233

List of Tables

2.1 User challenges 17
2.2 Resource provider challenges 18
2.3 Federation organization challenges 19
2.4 Non-functional challenges 20

3.1 Node types and node hardware 53
3.2 G-Lab sites and node count 54

5.1 Examples of user requests and respective provisioning actions (non-exhaustive list) 87

6.1 Mapping of Teagle repository operations to HTTP methods 123
6.2 Domain level resource and resource adapter identifiers 142

7.1 Observational evaluation methods 150
7.2 The federated Panlab resources 151
7.3 Observational evaluation: usage of the artifacts in multiple projects 163
7.4 Experimental evaluation methods 165
7.5 Comparison of different federation approaches 175

A.1 The Teagle and domain manager service endpoints used for this use case . . . 185
A.2 Packet trace of the VCT tool booking request and request processor response 203
A.3 Packet trace of the OE engine orchestration request and the respective response 207

1 Introduction

1.1 Motivation

Resource federation is a concept for *sharing resources* beyond the boundaries of *administrative domains*. Federations aim at implementing a *common service* drawing upon the resources committed by participating organizations. This enhances the *utility* of resource offerings significantly. Federating arbitrary resources across multiple administrative domains and on multiple federation levels, involves many technical, operational, and legal issues making it an interesting *interdisciplinary research field*.

While the concept can be applied to a number of fields such as identity management, networking, trust, and security, in the context of this work, the federation of *Information and Communication Technology (ICT) resources* is investigated. Generally, a federation is understood to be "an organization or group within which smaller divisions have some degree of internal autonomy" (Oxford definition). Merriam-Webster defines *federal* as: "(a) formed by a compact between political units that surrender their individual sovereignty to a central authority but retain limited residuary powers of government; (b) of or constituting a form of government in which power is distributed between a central authority and a number of constituent territorial units." This concept of federation, clearly stemming from a political background, is applied in the context of this thesis' research in sharing hardware and software resources across the borders of individual administrative (and usually organizational) domains. This means that the resource providers allow—to a certain extend—for remote resource control giving away some of their local autonomy to external entities.

Now, let's take a closer look at the problems to be solved by applying the resource federation concept. With the speed of network convergence and technology evolution, covering all fields from telecommunications to the Future Internet, today's ICT systems require sophisticated heterogeneous *experimental infrastructures* (or testbeds) to design, integrate, and test new solutions. The problem is that infrastructure lifetime—the time an infrastructure remains at technology's cutting edge—has decreased dramatically, making investments in expensive isolated research infrastructure more costly and risky than they were already. This is especially true for complex cross-layer and cross-technology testbeds. Also, gathering the required experts and know-how to build and maintain such complex infrastructures is a major challenge for the providers. Using federation mechanisms, two important phenomena can be observed: (1) access to a *single resource* can be given to *more users* (local and federation

users) and (2) a *single user* has access to *more resources* (local and federated resources). This allows, on the one hand, to increase the average resource utilization and, therefore, to increase the return on infrastructure investments. On the other hand, it allows to increase the *scale of testing and experimentation* and to include *unique infrastructural resources* or configuration properties that are available remotely and can be accessed through federation.

Figure 1.1 ([1], p. 168)[1] motivates the use of federation mechanisms to support ICT research. The research (and in particular the network research) process usually begins using *formal models* to study the subject of investigation analytically. However, the complexity of todays ICT systems increasingly complicates the construction of realistic models.

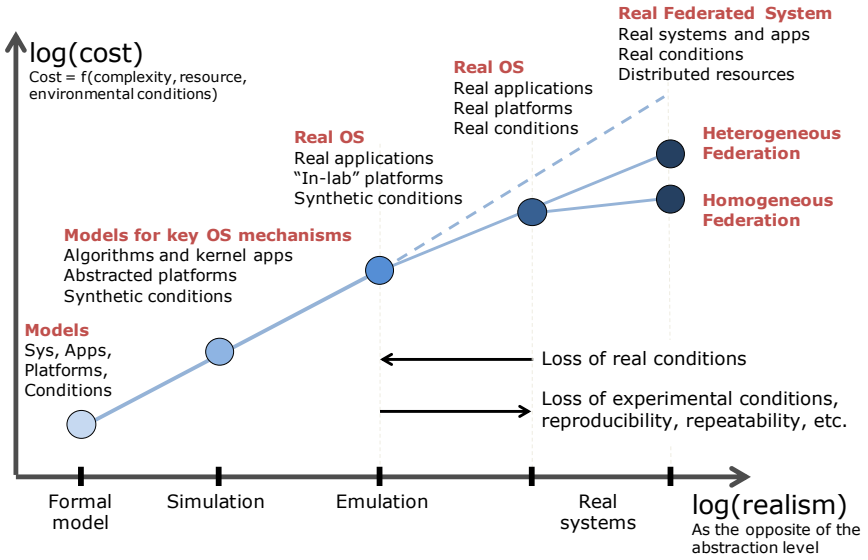


Figure 1.1 – Motivation for federation

Using *simulation and emulation techniques*, the subject of investigation can be studied in a laboratory environment and under controlled conditions. This allows for repeatability of experiments and the reproducibility of results. However, to further increase the realism of experiments, real systems have to be used. Here, federation helps to increase the scale of experiments and to increase the diversity of available resources. Also, federation has a positive influence on the cost that are associated with experimentation and testing (see figure 1.1) because existing resources and entire facilities are shared and re-used instead of being re-build. This is particularly important for industrial players that seek to cut costs due to the increasing global competition and market forces.

In addition to the field of experimental facilities where resource federation positions itself as a key concept, it can also have positive effects in a much broader sense. Modern societies are concerned about ICT infrastructure energy consumption and need to improve the way ICT resources are provisioned and maintained. As shown by recent studies, the electric energy consumption of industrialized nations' ICT sectors is around 10%. Reusing infrastructure, services, and data across the silos of individual organizations, as motivated by federation principles, helps to reduce energy consumption and over-provisioning. Therefore, the hope is that the work presented here will not only contribute to push the frontiers of science a little further but to impact the ICT sector and our society as a whole.

[1] Adapted from original figures published in [2], p. 4-5

1.2 Scope

1.2.1 Definitions

The following definitions reflect the title of this thesis and shall help to ease the understanding of its rather abstract context.

Definition of *Generic*: The Merriam Webster dictionary defines the adjective *generic* as "1 a: relating to or characteristic of a whole group or class [...]" [3].

For the scope of this work, this means that the framework (see next item for the definition of a framework) is defined not to be tailored towards a certain application field. The resulting system can be used for highly heterogeneous resources and within different contexts.

Definition of *Framework*: The Merriam Webster dictionary defines the noun *framework* as "1 a: a basic conceptional structure (as of ideas) [...], b: a skeletal, openwork, or structural frame [...]" [4].

For the scope of this work, a framework is defined to be (i) a methodical foundation as well as (ii) the resulting system to enable *resource federation*. Here, a *system* is to be understood as a set of elements that interact with each other to serve a common purpose and that form a conceptual unit that differs from its surroundings.

Definition of *Resource*: The Merriam Webster dictionary defines the noun *resource* as "1 a: a source of supply or support: an available means [...] e: a source of information or expertise [...]" [5].

Also, Wikipedia differentiates several contexts of resources, i.e. *computer science* and *Web*. In computer science, Wikipedia defines *resource* as: "A resource, or system resource, is any physical or virtual component of limited availability within a computer system. Every device connected to a computer system is a resource. Every internal system component is a resource. Virtual system resources include files, network connections and memory areas." [6]

In the context of the World Wide Web (WWW), Wikipedia describes a *resource* as follows. "The concept of resource is primitive in the Web architecture, and is used in the definition of its fundamental elements. The term was first introduced to refer to targets of Uniform Resource Locators (URLs), but its definition has been further extended to include the referent of any Uniform Resource Identifier (RFC 3986), or Internationalized Resource Identifier (RFC 3987) [...]" [7].

For the scope of this work, a resource is defined as anything that can be controlled by software and that is meant to be federated (shared) with other entities within a federation. Working examples include computers, virtual machines, and devices. On higher abstraction layers this includes software, services, and abstract concepts such as a domain.

Definition of *Federation*: The Merriam Webster dictionary defines the noun *federation* as "1: an encompassing political or societal entity formed by uniting smaller or more localized entities: as a: a federal government b: a union of organizations [...]" [8].

The adjective *federal* is defined by Merriam Webster as: "[...] 2 a: formed by a compact between political units that surrender their individual sovereignty to a central authority but retain limited residuary powers of government b: of or constituting a form of government in which power is distributed between a central authority and a number of constituent territorial units [...]" [9].

Those definitions originate from a political context. However, the phenomenon of sharing or surrendering individual sovereignty to an external party is also an important aspect of federation in a technical context.

Wikipedia addresses this aspect by defining *federation* in information technology (IT) as: "[...] multiple computing and/or network providers agreeing upon standards of operation in a collective fashion. The term may be used when describing the inter-operation of two distinct, formally disconnected, telecommunications networks that may have different internal structures. The term may also be used when groups attempt to delegate collective authority of development to prevent fragmentation [...]" [10].

For the scope of this work, the following definition of *federation* is applied: Resource federation is a concept for sharing resources beyond the boundaries of administrative (usually organizational) domains. Federations aim to implement a common service drawing upon the resources committed by participating organizations. This mechanism can follow a recursive model. A federation in itself might be viewed as one administrative domain that is federated on yet another overarching level. This model can be applied with any meaningful granularity to fit specific federation contexts.

1.2.2 Taxonomy

Figure 1.2 shows the main aspects of resource federation to categorize the different approaches and target fields for federation. The political context has not been reflected deliberately to keep the focus on the technical domain.

Several fields strive for federation today as the benefits have been recognized by the respective communities. Latest example is the cloud computing domain where resource and data sharing as well as virtual machine migration across cloud domain providers has been identified as a major use case with yet unexplored industrial impact. But also in the identity federation field, considerable achievements have been reached in recent years allowing for concepts like Single Sign-On (SSO) and contextualized persona across the identity silos of individual organizations.

As characterized by the definitions outlined in subsection 1.2.1, the work presented in this thesis aims at providing a generic framework. Such frameworks can be broken down further by differentiating between *homogeneous* and *heterogeneous* federation. The differentiating factor is mainly the types of resources that are federated. Ultimately, the way how federation is achieved i.e. which federation scenario (e.g. centralized, loose coupling, recursive) has been applied to model and implement the federation, is a strong categorization attribute determining the *level of integration* between the participating entities. The work described here, targets a hybrid approach where several federation services have been centralized for the first federation layer to allow for a tight coupling of resource between known peers and enable a "seed of trust". As sharing resource across administrative boundaries requires the peers to trust each other, this first federation level serves to establish the necessary trust relationships

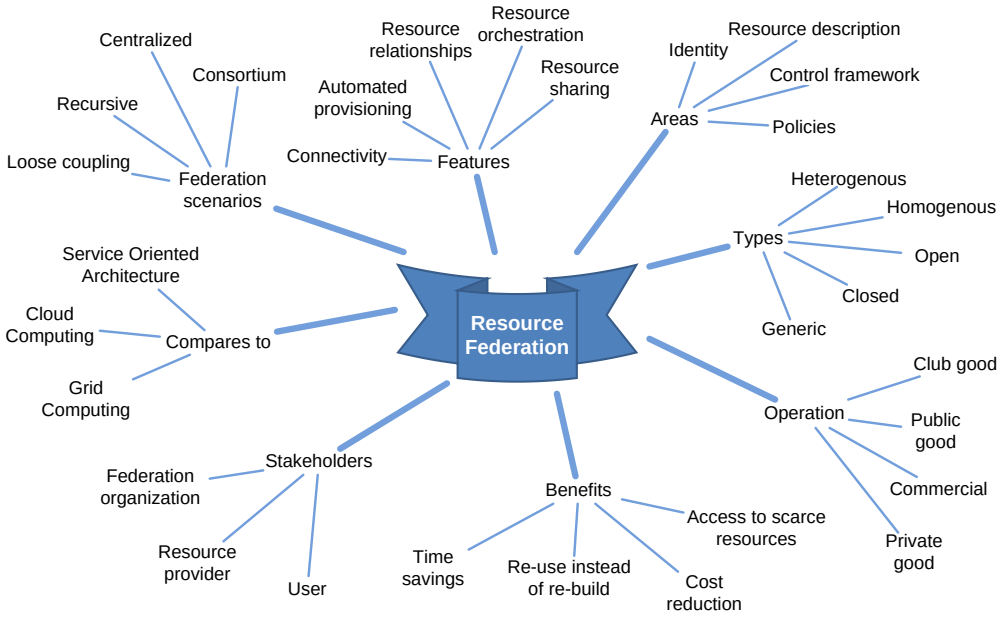


Figure 1.2 – A high level view on the main aspects and terminology of resource federation

as a nucleus for a larger federation. However, the core federation model has been designed to allow for recursion in order to serve more complex federation scenarios.

Further important aspects related to resource federation are the stakeholders, the operation, the features, the areas, etc. (see figure 1.2). Those terms and the concepts behind them will mostly be introduced and explained in chapter 4 (the federation model) and chapter 5 (the federation methods).

1.3 Research Hypothesis

The key scientific questions and the chosen approach to address them have been summarized using a *single sentence*. It postulates this work's research hypothesis:

1. To *effectively* and *efficiently federate heterogeneous resources* across the boundaries of administrative domains, enabling *flexible cross-domain resource collaboration*, ...
2. ... a *federation model* can be designed that allows to define a *generic resource federation methodology* and instantiate an according *system solution*, ...
3. ... building upon a *federated resource control framework* and using *resource description, abstraction, orchestration*, and *provisioning techniques*.

Although, the above sentence is poor from a writing style point of view, it has been formed to give a *condensed and concise statement* of what this thesis is about. The upcoming chapters will reference this hypothesis to make clear, at any stage, what is currently being addressed. Further details on the the individual aspects of each of the hypothesis' parts are given in the following enumeration.

1. The first part of the hypothesis demands *effectiveness* and *efficiency*. This means that the artifacts (see next section for an explanation of the term *artifact*) provided by this thesis have to be effective and efficient in addressing the problem of generic resource federation. Further, this part mandates that the approach shall work in a *cross-domain* and *cross-layer* (derived by the fact that heterogeneous resources are targeted) fashion.

   Here, the main contribution of this work is to apply the concept of federation to heterogeneous resources—as opposed to a homogeneous approach as for example taken by most Grid Computing efforts or PlanetLab—and enable resource collaboration, i.e. provide higher tier federation logic such as resource relationships and orchestration as opposed to simply providing resource access.

2. The second part of the hypothesis demands the design of three *artifacts*: a federation *model*, a federation *method*, and a system solution (an *instantiation*)[2].

   Here, the main contribution of this work is the design of the three artifacts.

   (i) Regarding the model, no other—to the best of my knowledge—federation model exists that provides both the flexibility and generic applicability of the model proposed by this thesis[3]. Most federation approaches that exist today can be modeled using the model presented here.

   (ii) With respect to the methods, the resource description and abstraction framework that allows the definition and implementation of abstract resource relationships enables automated resource orchestration and provisioning in a cross-domain and cross-layer fashion. Entire virtual resource groupings[4] can be described in terms of their resource relationships and provisioning dependencies. Such a generic approach to resource management that can be applied to any type of resource[5] and allows for recursion is not covered—to the best of my knowledge—by the existing literature.

   (iii) Regarding the instantiation, *Teagle* and the domain manager framework[6] considerably impact a number of currently running projects[7] and already enabled and stimulated further developments[7] (by providing a foundation for those) which marks an important contribution to the target scientific community. The prototypes have mostly been released as open source and have been implemented by several people. The main contribution of the author was the design and the software development coordination.

3. The third part of the hypothesis mandates how to approach the other parts and which are the main concepts to follow.

   Here, the main contribution of this work is to apply resource description, abstraction, provisioning, and interconnection concepts to the problem of generic resource federation. Although information modeling as well as service orchestration and abstraction are well known concepts, the stringent application of those aspects in the context of cross-domain and cross-layer resource federation is—to the best of my knowledge—not covered in depth

[2] See next section for a clarification of the terms: artifact, model, method, and instantiation

[3] The strongest competitor in this regard is the Slice-based Federation Architecture (SFAv2) which emerged around the same time but defines less model entities and scenarios.

[4] As introduced on page 61

[5] As long as the resource can be described using the resource description method presented in section 5.1

[6] See sections 6.5 and 6.6

[7] See figure 8.1

by existing literature. The advantage of the chosen approach is the ability to implement higher tier federation logic building upon heterogeneous distributed resources[8]. As an example, the concept enables the federation user (e.g. the experimenter) to change the configuration of the resources associated to his the virtual resource grouping *at experiment runtime* allowing for a new type of the experiments that may recursively influence themselves.

1.4 Scientific Classification and Research Methodology

It is important to note that large parts of the work presented in this thesis are dedicated to *engineering and applied science*—putting research results into practice across several projects—rather than foundational research that is usually targeted by the natural sciences. The subject of investigation can be subsumed to different and partially overlapping scientific disciplines, namely informatics, software engineering, economics, organizational science, and information system (IS) research.

The main results of this work are a *federation model*, a *federation method*, and an according *system design, instantiation, and evaluation*. The model, the method, and the instantiation are *artifacts* in terms of *design science* [11]. Design science is problem-solving oriented and has its roots in the engineering disciplines. In contrast to the natural sciences, it investigates the problems and phenomena created by humans in order to reach certain targets. The goal is to create and evaluate *artifacts* to solve organizational problems. In this sense, *design* is understood as the deliberate organization of resources in order to reach a goal.

Similarly, since its inception in the 1970s, the IS research field has positioned itself in face of neighborhood disciplines, namely the organizational science and information science to study the implications of interaction between organizational entities, humans, and IT. The study of IT-based systems (the so-called *IT artifact*) is at the very core of the IS discipline. Design science research results in IS, are artifacts that can be described by models, software, and even natural language specifications. The artifacts are classified as *constructs*, *models*, *methods*, and *instantiations*.

Constructs denote the language to define problems and solutions.

Models use constructs to describe problems and solutions.

Methods define the process to solve problems.

Instantiations are the actual system implementations that are able to demonstrate how the three other artifacts are effectively and efficiently used to solve problems in a specific context.

Figure 1.3 (adapted from [12], p. 476) shows the organizational and IS design activities and their relation to business and IT strategies. The focus of this thesis is on the IS infrastructure and IS design aspects of a generic resource federation framework. Business, organizational, as well as legal aspects have been considered, but have not been investigated in depth.

The research has been based on a clear framework and methodology that is shown in figure 1.4 ([11], p. 80). The target was the design and implementation of the *artifacts*

[8] E.g., the same way higher level programming languages can make use of the underlying machine resources through Application Programming Interfaces (APIs).

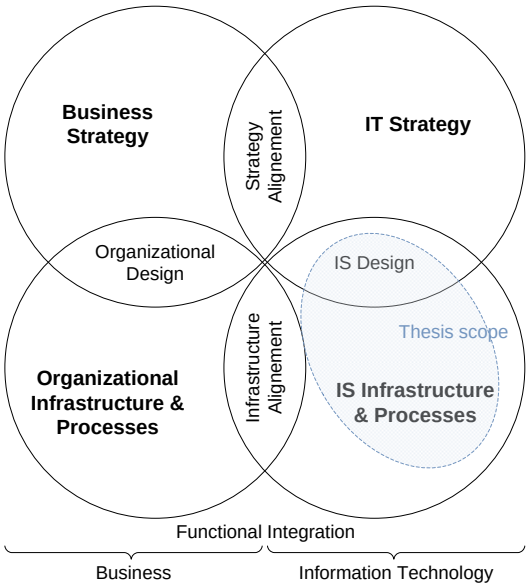


Figure 1.3 – Business, organizational, information technology, and information system boundaries

mentioned above rather than the development of an abstract *theory*. Methods from different evaluation method classes (observational and experimental) have been used to asses and refine the results, enabling a circular design spiral to derive the model, the methods, and the framework instantiation. As shown by figure 1.4, research should address relevant business needs. Resulting solutions and systems have to be applicable to the targeted environment (people, organizations, technology). This can be subsumed under the term *research relevance*.

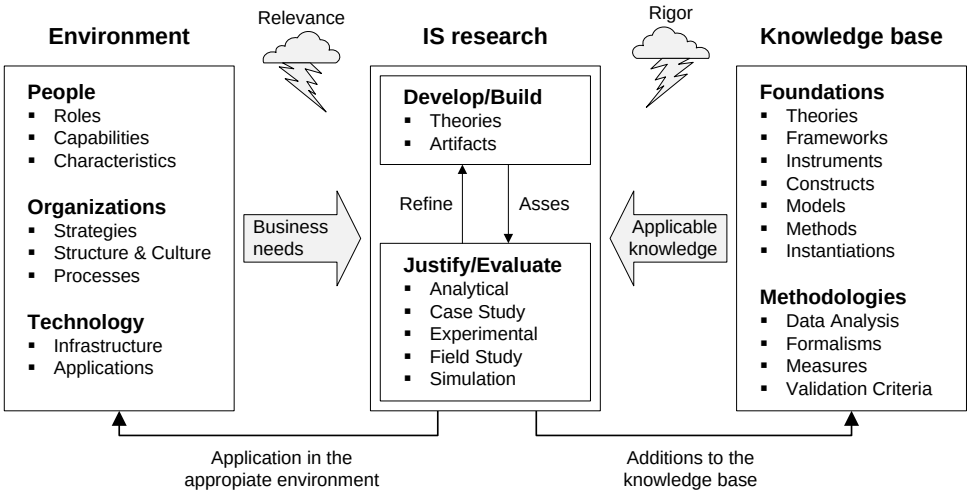


Figure 1.4 – Information system research framework

Today, an impressive *knowledge base* (foundations and methodologies) is available to be applied to the subject of research. The research results themselves extend the knowledge base. By consistently applying existing foundational and methodological knowledge, *research rigor* is achieved. The difference between routine engineering or system building and IS research

lies in the extension of the knowledge base by addressing important unsolved problems.

To derive meaningful research results, an established set of research guidelines defined by Hevner et al. ([11], p. 83) was adapted and extended[9]:

Problem Relevance: The objective is to develop a technology-based solution to a relevant problem.

Design as an Artifact: Design science research must produce a viable artifact in the form of a construct, a model, a method, or an instantiation.

Evaluation: The utility, quality, and efficiency of an artifact must be rigorously demonstrated via accepted and well-executed evaluation methods.

Research Contributions: Effective research must provide clear and verifiable contributions in the areas of the artifact, foundations, and/or methodologies.

Research Rigor: Design science research relies upon the application of rigorous methods in both the construction and evaluation of the design artifact.

Design as a Search Process: The search for an effective artifact requires utilizing available means to reach desired ends while satisfying laws in the problem environment. By executing several design/evaluation circles, the solution space can be narrowed to derive a useful, effective, and high quality problem solution.

Results Dissemination and Adoption: Ultimately, the value and success of architectural design artifacts is best measured by the level of adoption within the target field. An important aspect in this regard is the sufficient publication of research results as well as practical tutorials attended by the target audience.

The above mentioned guidelines have been respected and applied throughout the entire research process for this thesis. The next section outlines the thesis structure summarizing each of the upcoming chapters.

1.5 Structure

This thesis contains eight chapters. The chapters have been aligned with the research methodology introduced by the previous section. The main steps and the according chapters of this thesis are shown in figure 1.5. Each chapter is briefly introduced in the following subsections.

Federation Challenges

This chapter describes the problem to be solved by this thesis. It lists a number of high level challenges and requirements to be meet be the artifacts (federation model, method, and instantiation) provided by this work. The challenges are grouped according to the federation stakeholders. In terms of IS and design science research methodology, the chapter demonstrates the *problem awareness*.

[9] In order to increase the readability, no quotation marks have been used to indicate word for word quotations in this listing. Most of the listing of guidelines is to be considered as original work produced and published by Hevner et al.

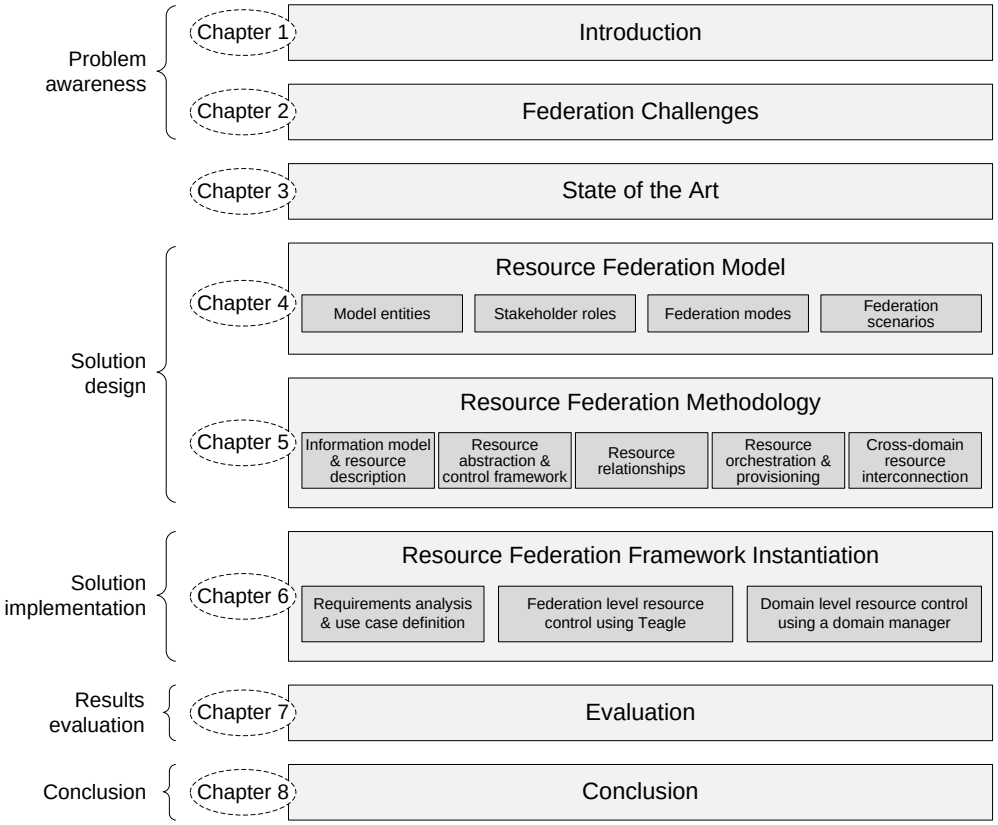


Figure 1.5 – Thesis chapters overview

State of the Art

The third chapter introduces important concepts, technologies, initiatives, and projects in the field of federation and service orientated infrastructure paradigms. Among the discussed fields are the broad research areas of Grid Computing, Cloud Computing, Federated Identity Management, as well as Future Internet experimental facilities.

Resource Federation Model

This chapter defines the design artifact *federation model* that is used to characterize the problem of generic resource federation, the involved stakeholders, different federation modes, and resulting federation scenarios. The model defines five conceptual entities and two abstract concepts. Those are used to explain the different federation areas of identity management, federated control frameworks, resources descriptions, and policy handling. The scenarios are constructed making use of the model and different manifestations of the federation modes. Most federation approaches that exist today can be classified and described using the model introduced by this chapter.

Resource Federation Methodology

This chapter defines the design artifact *federation methodology* and outlines a set of *methods* and *processes* enabling generic resource federation across different resource provider domains.

The methods build upon the federation model defined in chapter 4 and describe the process of deriving functional *virtual resource groupings* form a set of heterogeneous distributed resources. The methods address the key areas of resource description, abstraction, orchestration, provisioning, and interconnection. The framework instantiation builds upon the federation model and the federation methods.

Resource Federation Framework Instantiation

This chapter provides detailed insights into the federation framework *instantiation* covering both the *federation* as well as the *domain* layers. The system components are build based on a stakeholder requirements analysis. The analysis reveals partly *conflicting requirements*. Particularly, some of the federation user goals are conflicting with several resource provider goals. GRL evaluation strategies are used to analyze the impact of critical system decision aspects on the different stakeholder goals. A UCM-based use case analysis helps to follow and understand the intended system interactions. From the requirement and use case analysis, important *design decisions* are derived such as the implementation of a common information model and control framework. The resulting prototype system called *Teagle* includes a number of components such as a design tool, a policy engine, a request processor, an orchestration engine, etc. that collectively perform *inter domain resource management*. On the domain layer, a domain manager framework hosting different resource adapters deals with resource abstraction and *intra domain management*.

Evaluation

This chapter describes important *evaluation results*. Different evaluation methods (observational method: a field study & experimental method: a controlled experiment) are used to determine if the resource federation framework (including the model, the methods, and the instantiation) is suited to effectively solve the problem. Some early evaluation results from different projects have been used to refine the artifacts. This research spiral approach allowed to narrow the problem space and improve the overall artifact design. The chapter also provides a comparison with similar approaches.

Conclusion and Outlook

This chapter concludes the thesis and shows that the presented work has been disseminated through a number of publications and tutorials. This led to the adoption of some of the research results presented by this thesis by other projects and also resulted in additional developments based on this work. Towards the end of the chapter, an outlook into possible extensions of the presented work is given.

2 Federation Challenges

Scientific research aims at generating knowledge and solve existing problems of a particular research area. This chapter targets a characterization of the problems to be solved by the results presented in this thesis. Hence, the major challenges of generic heterogeneous resource federation and the high level requirements of different stakeholders that are likely to be involved in a federated environment are outlined. In this context and with specific emphasis on design science, Hevner et al. constitute:

> *A design artifact is complete and effective when it satisfies the requirements and constraints of the problem it was meant to solve.* [11], p. 85

With this in mind, the artifacts delivered by this thesis (the federation model, methods, and framework instantiation) have been designed, build, and evaluated.

2.1 Federation Challenges Overview

In section 1.3, a research hypothesis was given using a *single sentence*. The hypothesis will be used to structure the challenges addressed by this section. It is repeated here for the convenience of the reader:

1. To *effectively* and *efficiently federate heterogeneous resources* across the boundaries of administrative domains, enabling *flexible cross-domain resource collaboration*, ...
2. ... a *federation model* can be designed that allows to define a *generic resource federation methodology* and instantiate an according *system solution*, ...
3. ... building upon a *federated resource control framework* and using *resource description, abstraction, orchestration*, and *provisioning techniques*.

From the first block, it becomes clear that different administrative domains will be involved in a federated system. Therefore, the artifacts delivered by this thesis will need to take into account the concept of a *domain* marking the boundaries of resource ownership. This will lead to the definition of different stakeholders interacting within the system (i.e. the federation) in order to enable cross-domain collaboration.

Furthermore, the aspect of heterogeneity plays a crucial role in resource federations. Besides resource heterogeneity, federations usually face the challenge that many processes regarding resource control, access, and usage are heterogeneous across the federation stakeholders. This will need to be taken into account and will considerably impact the design of the artifacts.

From the third block of the research hypothesis, a number of challenges can be derived. Resource federations need to enable sufficient resource abstraction, to allow higher federation layers to work with distributed resources and provide generic services but at the same time enable users to utilize specific features offered by resources. Mechanisms such as resource orchestration, flexible resource hierarchies, cross-domain configuration dependency resolution, as well as automatic resource deployment and provisioning are challenging aspects to be addressed by this work.

However, a number of challenges derive from the intended operational area. A very large application field where federation mechanisms play a major role today is Future Internet (FI) experimental facility resource federation. As motivated by figure 1.1, federation can help to cut down the costs of large scale research experiments and increase the range of possible experiments by introducing resource heterogeneity. In this context, Paul et al. consider *federation* to be an active field of research:

> *Federation research looks at the methods to unify multiple diverse testbeds designed to serve diverse experimental contexts and realistic experimental environment.* [13], p. 28

While the scope of this work mostly covers technical challenges, federation introduces a number of legal and socio-economic issues. Providing access to resources governed by a specific organization usually requires to respect several organization-wide policies and guidelines. Depending on the type of organization, such guidelines might restrict or even completely prohibit joining a federation and allowing external access to the resources provided by that organization. On the technical side, Paul et al. summarize some of the high level experimental facility federation challenges:

> *The technical challenges involve problems such as (1) homogenization of diverse contexts to facilitate easy deployment of experiments, (2) fair and efficient sharing of scarce resources, and (3) interoperability of security protocols.* [13], p. 29

In Europe, the working group on modular federation of experimental testbed facilities, the so-called *wise men group*, has been set up to derive an outline of common principles for a collaboration and high level federation architecture. The group produced a report that states additional goals and challenges in experimental facility federation:

> *A federation of test-beds aims at creating a physical and logical interconnection of several independent experimental facilities or testbeds to provide a larger-scale, more diverse and higher performance platform for accomplishing tests and experiments.* [14], p. 2

This makes clear that *diversity* in terms of federated resources is a major challenge when interconnecting independent facilities in addition to scale and performance. Several additional aspects are considered by Crowcroft et al.:

> *The major goal of test-bed federation is to enable experiments on Future Internet research that are otherwise not possible, and this in a cost efficient and experimenter friendly way. The overall results for federated test-beds should also ensure interoperability, transparency and neutrality of the basic infrastructures.* [14], p. 2

This statement touches two distinct areas: the federation shall be cost efficient while user friendly, as well as inter-operable while respecting neutrality of the infrastructures. Some of those challenges result in requirements and goals that are contradicting across the different stakeholders (e.g. user vs. resource provider goals, see also figure 2.1) interacting within a federation. While the instantiated federation (i.e. the deployed system in conjunction with an accompanying organization) usually has clear objectives and target usage areas, a generic framework that can serve different application areas must take into account a broader spectrum of challenges, thereby increasing the possibilities of requirement contradictions.

A way to approach this issue is to shift the focus to a higher abstraction level (i.e. a *meta level*), leaving the definition of concrete details to the specific system to be deployed in a particular context. This context (i.e. the usage area) defines all the environment variables and detailed requirements that have been abstracted on the *meta framework level* (for a detailed analysis of stakeholder requirements and how to deal with contradicting system requirements see also section 6.1.2).

2.2 Categorization of Challenges and Requirements

Figure 2.1 shows a high level overview of the roles and relationships typically found in resource federations. The different roles are discussed in greater detail in section 4.2. For now, it can simply be assumed that a *user* interested in some resource capabilities that are offered by *resource providers* interacts with the providers who are collectively represented by a *federation organization*.

The federation organization and the participating resource providers engage in legal, operational, and technical relationships. The focus of this work is on the technical mechanisms to enable cross-domain resource federation. The challenges of such high level framework are summarized in the following listing. They are further broken down into stakeholder role dependent challenges and requirements and are explained in more detail in the following sections 2.2.1 to 2.2.4. Among the high level challenges are:

- the definition of federation stakeholder roles and federation-wide identity management means,
- the design of mechanisms for shared access to heterogeneous resources across federated domains,
- the design of description techniques for highly heterogeneous resources,
- federation-wide resource discovery mechanisms,
- the configuration, reservation, and provisioning of heterogeneous resources,
- resource interconnection across organizational boundaries and administrative realms,
- the specification of resource orchestration, dependency resolution, and flexible deployment procedures,

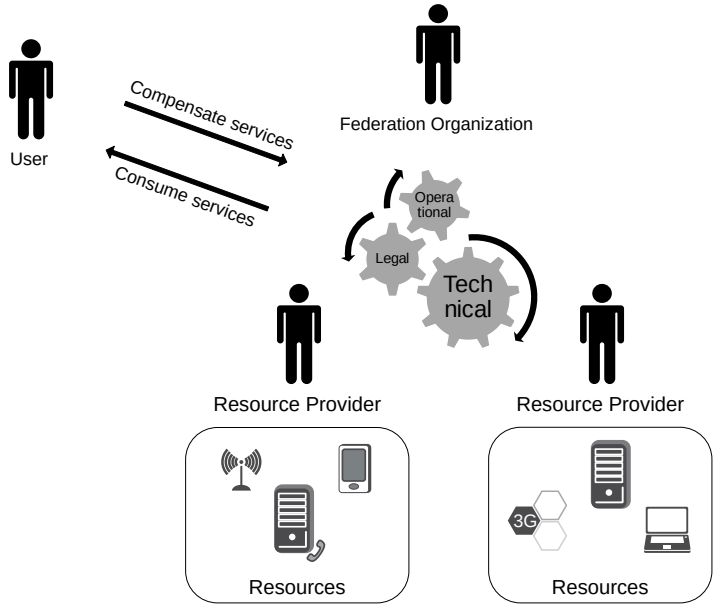


Figure 2.1 – Federation stakeholders and relationships. The different stakeholder roles allow to group challenges and requirements accordingly.

- the definition of resource security means including control framework interfaces as well as resource access and usage policies,
- resource abstraction and life-cycle support,
- openness, high usability, and flexibility using several abstraction layers.

From a resource provider perspective, the author started to identify benefits and challenges of contributing experimental resources to a pan-European federation already in 2006/2007 focusing on IP Multimedia Subsystem (IMS) and Next Generation Network (NGN) resources and the processes needed to market them as a technology oriented playground [15, 16, 17].

In the following sections, the challenges are detailed and grouped according to the different stakeholder roles. As a result, the high level requirements that particularly influence the design of the *federation model* and the *federation methods* have been collected. However, more detailed requirements are needed for the design of the *system instantiation*. Those are outlined and discussed in chapter 6. Also, the high level challenges and requirements are structured differentiating between functional and non-functional ones. The functional ones are further broken down into user challenges (section 2.2.1), resource provider challenges (section 2.2.2), and federation organization challenges (section 2.2.3). The non-functional challenges are detailed in section 2.2.4.

2.2.1 User Challenges

Table 2.1 lists and describes the federation user challenges: (1) resource lookup, (2) access heterogeneous resources, (3) resource control, (4) identification services, (5) tools and platform API support, (6) resource orchestration and automated provisioning.

Table 2.1 – User challenges

Requirement	Description
Resource lookup	In most cases, users will request to browse through the federation offerings in order to identify resources serving their needs. Sufficiently detailed information should be provided for users to select and request desired resources.
Access and reserve heterogeneous resources	This assumes that the user contacts the federation (e.g. represented by a federation organization) in order to obtain access to specific resources. Such access might be requested to be exclusive or shared. It is further assumed that the user requests resource access and reservation to use the resources in order to satisfy certain resource usage requirements (e.g. development, prototyping, testing, etc.). However, the type of usage is not limited by definition. Therefore, from a conceptual system design perspective, the types of resources that can be requested should also not be limited.
Resource control	The distributed resources offered by the federation shall allow to be controlled remotely enabling extensive and flexible resource usage. Such control primarily includes resource configuration. The providers of the resources define to which extend resource control is granted.
Identification services	As the users will interact with various federation entities and services provided by several stakeholders (e.g. several resources from different providers as well as different federation organization services and tools), they need to be authenticated and authorized at several points. This needs to function seamlessly and in a SSO fashion to ease the interaction with the federated platform.
Tools and platform API support	Once the user has requested and obtained access to a set of federated resources, usually, tools and platform APIs are required to interact with the resources, speed up routine tasks, and obtain meaningful results with reasonable efforts.
Resource orchestration and automated provisioning	In order to derive a collection of federated resource instances that are configured and set up according to specific user needs, some sort of resource orchestration is required that allows to resolve configuration dependencies and execute the resource provisioning. Although, this can often be achieved by manual configuration, automatic orchestration simplifies complex resource configuration and deployment tasks considerably and allows to realize federated resource provisioning more quickly and less error-prone.

2.2.2 Resource Provider Challenges

Table 2.2 lists and describes the resource providers challenges: (1) resource life cycle support, (2) resource abstraction and platform support for heterogeneous resources, (3) resource security and open standards, and (4) domain and resource specific policies.

Table 2.2 – Resource provider challenges

Requirement	Description
Resource life cycle support	Resource providers need a means to un-/register, describe, and re-/configure the resources offered to the federation.
Resource abstraction and platform support for heterogeneous resources	As the types of resources that providers can offer are not limited by definition (the only exceptions are illegal or offensive resources), resource providers rely on advanced platform support to deal with technical challenges regarding resource heterogeneity. An extensive resource abstraction framework is required that is able to deal with a wide variety of resource types.
Resource security and open standards	Offering resources across organizational boundaries to external users is a delicate issue for many providers. In some cases, it might even be completely prohibited by corporate policies. Therefore, resource security, including identification services, is a key requirement to attract a critical mass of resource providers. By relying on open standards to access and control domain resources, the providers can choose to deploy existing control framework implementations or implement their own solutions. This increases the trust in the federation concept and a specific platform instantiation.
Domain and resource specific policies	Resource providers have the need to restrict resource access and usage to their specific requirements (e.g. lab conditions, corporate policies, local usage, etc.). Also, in many cases, resource providers will offer resources that are: (1) exclusively dedicated to federation users, and (2) shared between federation users, local users, and other remote users. Therefore, resource providers require the possibility to define global and domain specific policies regulating the usage of their resources.

2.2.3 Federation Organization Challenges

Table 2.3 lists and describes the federation organization challenges: (1) technical federation infrastructure and tools, (2) cross-domain resource interconnection mechanisms, (3) funding & business model, (4) organizational structure, and (5) create incentives for users and resource providers. Although the focus of this work is on the technical issues, some key business and organizational challenges have been included due to their high impact on the technical side and to allow for a more holistic view on the topic.

Table 2.3 – Federation organization challenges

Requirement	Description
Technical federation infrastructure and tools	The federation organization must ensure that its services remain attractive to both the users and the providers. If users can obtain the same service provided by the federation organization from a single resource provider directly, then there is no need for the federation and the organization itself. The technical infrastructure to provide federation services (e.g. consistent identification, orchestrated resource collections, resource heterogeneity, generic control framework, etc.) must be flexible and adaptive enough to support the organization on a daily and on a long term basis.
Cross-domain resource interconnection mechanisms	In order to offer sets of federated resource instances, the federation must provide resource interconnection capabilities. If such connections are based upon public Internet, sufficient security mechanisms must be enforced. This impacts a number of technical aspects and will considerably influence the design artifacts.
Funding & business model	The federation organization acts like a resource broker and governing authority between the resource providers and the users. In order to sustain its operation, it requires funding. Among the options are: (1) a fee-based system where either users and/or providers pay certain fees for using federation services, (2) public funding, or (3) a combination of the other two. The type and amount of fee (money vs. resource contributions) can vary according to the type of provider (industry vs. academia).
Organizational structure	In combination with the type of funding and the business model, the overall organizational approach needs to follow a well designed structure. Different options are: (1) a market based/commercial approach, (2) a public good model, and (3) a club model. The organizational structure will influence the system instantiation with considerable impact on the other stakeholders.
Create incentives for users and resource providers	Although the choice of the organizational structure should have been taken in view of the incentives for resource providers to join the federation, the federation organization must ensure that this works for a critical mass of providers and federation users. The federation will only be attractive to many users, if the pool of federated resources is large, partly unique, and feature rich.

2.2.4 Non-Functional Challenges

Table 2.4 lists and describes the non-functional challenges: (1) usability, (2) platform extensibility, (3) performance, (4) security, (5) scalability, and (6) cross-domain aspects. Sometimes, the boundaries between functional and non-functional requirements are somewhat fuzzy. For example, it could be argued that security or cross-domain aspects are in fact functional requirements. Therefore, it should be kept in mind that some of the following challenges can apply to multiple stakeholders in different manifestations.

Table 2.4 – Non-functional challenges

Requirement	Description
Usability	The design artifacts should allow for easy use by all stakeholders. As resource heterogeneity is a major design criterion, the way how the users (i.e. the stakeholders) interact with the platform and make use of federated resources need to support different usage styles.
Platform extensibility	To stay attractive over time, most federations are assumed to undergo a constant evolution in terms of resource offerings. This requires the architecture design and platform engineering to allow for extensibility in terms of the federated resource types.
Performance	The system needs to perform sufficiently well to support the overall federation operation. This affects various platform entities. In terms of resource management, real time resource provisioning is *not* a requirement.
Security	The framework needs to be secure to avoid resource misuse and fraud. This boils down to the basic principles of information security: confidentiality, integrity, availability, authenticity, and non-repudiation. However, depending on the specific target system instantiation, functional security aspects might arise, e.g. experiment isolation.
Scalability	The framework will have to deal with changing amounts of resources and platform interactions while meeting the performance requirement. A realistic measure for this type of system performance is to observe the behavior for a critical mass of resources, providers, and users. The critical mass in turn is determined by the overall operation (i.e. whether the federation is sustainable given a certain number of users and providers).
Cross-domain aspects	The resources to be federated are distributed across multiple administrative domains. Still, federation users shall be enabled to work with such resources independently of the physical location and any administrative constraints. This results in a trade-off between resource consumer and resource provider interests and preferences. A balance must be found between the partly conflicting interests of the different stakeholders within the federated system.

3 State of the Art

Based on the challenges of generic resource federation that have been identified and discussed in the previous chapter, this chapter is dedicated to the concepts, existing principles, and initiatives that represent the state of the art in this field. The first sections cover the broad fields of service oriented architectures and distributed resource access, while the later sections specifically look at federation aspects and provide a greater level of detail in terms of current technologies, international initiatives, and projects.

3.1 Service Orientation as a Fundamental Design Pattern

The term *Service Oriented Architecture (SOA)* appeared around 1996. It reflects an important trend in IT to organize and use distributed systems in a service oriented manner. In a SOA, services are designed, implemented, and grouped together to support business processes. Through orchestration of service building blocks (those can be atomic services or federated and orchestrated services themselves) high level processes can be supported dynamically re-using building blocks in different contexts.

The Organization for the Advancement of Structured Information Standards (OASIS) published a reference model for SOA in 2006 and defines the term as follows:

> *Service Oriented Architecture is a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains. It provides a uniform means to offer, discover, interact with and use capabilities to produce desired effects consistent with measurable preconditions and expectations. [18], p. 29*

This introduces the important aspect of distributed control and ownership of capabilities that are provided by means of services. In SOAs, the primary roles are *service provider* and *service consumer*. A service is seen as "[...] the means by which the needs of a consumer are brought together with the capabilities of a provider." ([18], p. 29)

Figure 3.1 shows the basic relationships between the conceptual entities: capability, service, service provider, and service consumer.

Figure 3.1 – Primary roles in a service oriented architectural design

The simple interaction model shown in figure 3.1 is enriched by the following additional concepts: (1) execution context, (2) real world effect, (3) interaction, (4) visibility, (5) contract & policy, and (6) service description as depicted in figure 3.2. All terms are defined and explained in detail in [18].

Furthermore, the Object Management Group (OMG) specified a SOA modeling language (SoaML) which provides a Unified Modeling Language (UML) profile and metamodel [19] targeting SOA service design. It allows the specification of service systems as well as service interfaces and implementations. The goal is to enable an Model Driven Architecture (MDA) based approach to derive design artifacts automatically. MDA is as software development concept that uses formal models to separate functionality and design aspects from implementation and technology details. The model defines the information structure and model entity relationships that can then be translated into any target machine interpretable language.

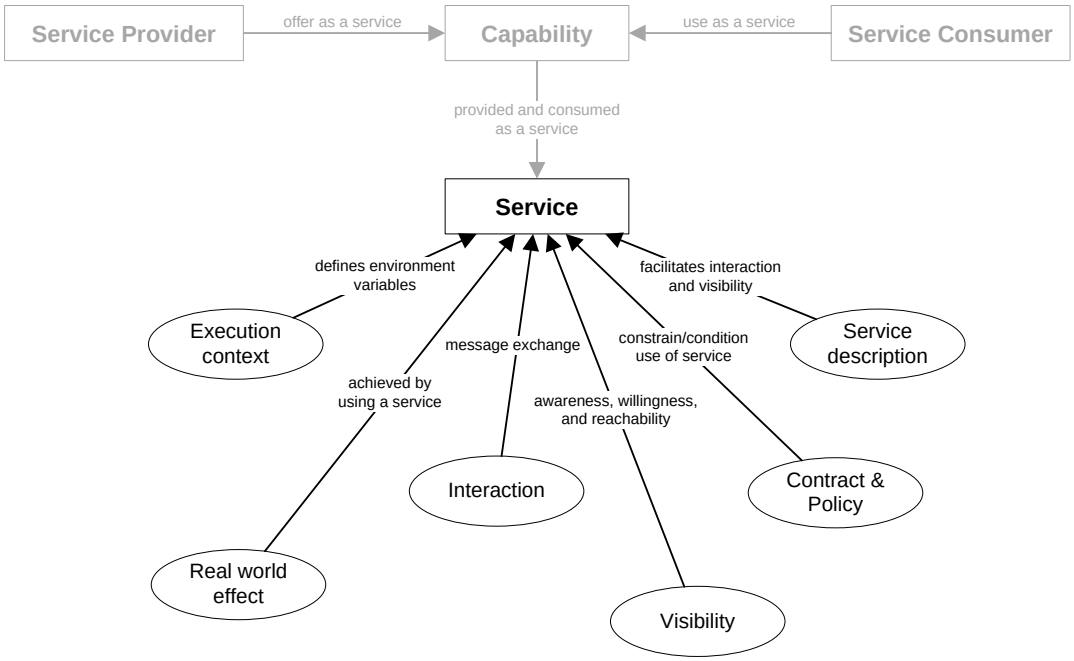


Figure 3.2 – Principal concepts of the SOA reference model defined by OASIS

Lately, the exposure of network capabilities via SOA mechanisms [20] has become interesting in the telecommunications operator (Telco) domain to expand the scope of NGN service consumption and improve network and service management [21]. Specific testbeds [22] [23] to enable technology and service design evaluation and early prototyping can help operators and third party service developers to gain access to critical infrastructure and obtain feedback on solutions which are going to be introduced to the market.

The OASIS and OMG specifications as well as the development of service oriented testbed

and experiment facilities [24] show that the concept of service orientation has become a key concept in terms of architecture design and derived artifacts. Therefore, it will considerable impact this work which aims at federating distributed capabilities allowing to implement a common high level service across multiple resource providers. Hence, the basic service oriented design aspects shown in figure 3.2 (i.e. interaction, description, policy description) have influenced the design of the federation model and methods and the implemented system.

3.2 Infrastructure as a Service

Although the concept of Infrastructure as a Service (IaaS) is part of the Everything as a Service (XaaS) approach that is discussed in section 3.3 on cloud computing, one separate section is dedicated to IaaS, since it is more related to this thesis rather than other parts of the cloud stack (i.e. Platform as a Service (PaaS) and Software as a Service (SaaS)).

There is currently no generally accepted definition for IaaS. However, many related articles refer to the definition published by the National Institute of Standards and Technology (NIST), a United States Department of Commerce agency. This IaaS definition is also referred to by the Distributed Management Task Force (DMTF) in most of their cloud computing related white papers (e.g. [25][26]):

> *The capability provided to the consumer is to provision processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, deployed applications, and possibly limited control of select networking components (e.g., host firewalls).* [27], p. 3

In a wider and less cloud computing oriented sense, IaaS refers to the concept of outsourcing infrastructure operations to an external entity. Many industry players and especially small-to-medium sized enterprises (SMEs) have recognized that owning a data center requires considerable investments in terms of procurement, personnel, and maintenance (particularly electricity) costs. Also, it is difficult for smaller players to react on resource demand changes flexibly (resource scalability and elasticity). Additional challenges are infrastructure availability, reliability, and security. Therefore, acquiring ICT infrastructure as an outsourced services has become increasingly interesting and profitable to support the operation of various routine ICT applications, such as home pages and web servers, databases, mail systems, etc.

As demonstrated, using IaaS offerings can be attractive from the perspective of individual organizations. But also from a global point of view it is interesting in terms of the overall ICT sector energy consumption. For example, concentrating ICT resource maintenance in large data centers allows for the installation of energy efficient professional cooling systems, advanced virtualisation technology, and high speed secure networking. Also, down-scaling effects (switching off unused systems) can be exploited more effectively. This has a positive effect on the overall ICT sector energy consumption and carbon footprint as shown by recent studies. Fan et al. argue that considerable power and energy savings (30% peak power savings, 50% energy savings) are possible for datacenters by reducing idle power consumption [28].

Figure 3.3 shows the IaaS layer at the bottom of the entire cloud stack including network, storage, and compute resources. To which extend and in which combinations control over resources is granted to the user, depends on the provider. Especially network configurations might compromise security measures. Here, providers need to find a balance between flexible, highly configurable IaaS offerings and overall security considerations. Examples for IaaS offerings are Amazon S3 (storage) [29] and Amazon EC2 (compute) [30].

3.3 Cloud Computing

3.3.1 Overview

On top of the IaaS layer, the cloud stack defines two additional layers: the PaaS layer and the SaaS layer. Collectively, this is often refered to as XaaS or *aaS as acronmys for *Everything as a Service*.

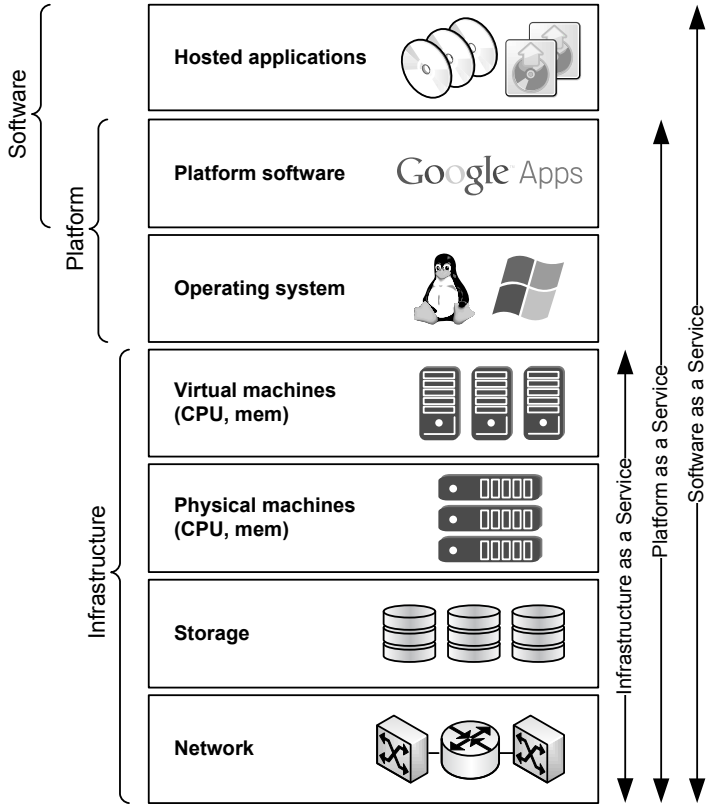


Figure 3.3 – The cloud stack showing all layers of the XaaS concept: Infrastructure, Platform, and Software as a Service

Figure 3.3 shows the different layers and the related service concepts where PaaS includes the platform as well as the infrastructure layer while SaaS includes all the layers. NIST defines cloud computing 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 or service provider interaction. This cloud model promotes availability and is composed of five essential characteristics, three service models, and four deployment models.* [27], p. 2

The service type on the lowest layer has been discussed in section 3.2. The NIST definitions for PaaS and SaaS respectively are as follows:

> *Cloud Platform as a Service (PaaS). The capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages and tools supported by the provider. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly application hosting environment configurations.* [27], p. 2

> *Cloud Software as a Service (SaaS). The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure. The applications are accessible from various client devices through a thin client interface such as a web browser (e.g., web-based email). The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings.* [27], p. 2

3.3.2 Cloud Characteristics

In addition to the clearly separated service types, the Cloud computing paradigm is characterized by a number of aspects [31].

Cloud Deployment There are three main deployment types: (1) *public clouds*, (2) *private clouds*, and (3) *hybrid clouds*.

In *public clouds*, services are offered beyond the hosting organization. In contrast, *private clouds* are used only by the owner or lessee (in a leasing context) and services are not exposed to external users.

Hybrid clouds aim at balancing between the two other extremes and make use of private and public cloud resources. This is usually done to circumvent certain issues in public clouds (e.g the disclosure of sensitive data).

In addition, some articles refer to *community clouds* and *special purpose clouds*. However, those can be seen as special subtypes of the three main cloud deployment types.

Cloud capabilities Among the most important capabilities of cloud computing system are: (1) *elasticity*, (2) *quality of service*, and (3) *high availability*.

The term *elasticity* describes one of the main features of cloud environments allowing to scale resources up and down flexibly in order to adapt to changing requirements. These requirements are usually passed down from the application layer. The aim is to respond to varying load situations, for example to guarantee certain response times of cloud applications. Although elasticity features are mainly delivered by virtualization techniques and technologies and can principally be seen as independent from the cloud

computing concept, scalability is perceived today as one of the very core benefits provided by cloud computing systems. "One can distinguish between horizontal and vertical scalability, whereby *horizontal scalability* refers to the amount of instances to satisfy e.g. changing amount of requests, and *vertical scalability* refers to the size of the instances themselves and thus implicit to the amount of resources required to maintain the size. Cloud scalability involves both (rapid) up- and down-scaling." ([31], p. 13)

Quality of Service (QoS) denotes the requirement of ensuring a certain measurable service level. A good example is the application response time. It can only be assured if the underlying infrastructure is performing as planned and is able to scale if needed. Therefore, the elasticity capability of cloud systems helps to guarantee a certain QoS. Other cloud capabilities can be subsumed under the overarching QoS term, such as *reliability* which is strongly connected to high availability.

High availability denotes the ability of cloud system to ensure a very high availability of services running within the cloud. This is achieved by means of data replication on distributed resources, service redundancy, load-balancing, and fault tolerance.

Technological aspects A number of the key cloud system capabilities are achieved by building upon the following technologies and techniques: (1) *virtualization*, (2) *multi-tenancy*, and (3) *metering*.

Virtualization is the most important technology that enabled the fast ascend of the cloud computing concept. It is commonly used to virtually separate operating platforms (virtual machines) from the underlying hardware platform. Hardware resources can thereby be shared by several virtual machines running on the same physical machine. For example, a machine that runs a Linux derivate may also host a Windows virtual machine in such a way that it is transparent to the Windows machine that it is currently sharing the hardware with another operating environment. The system creating and managing virtual machines on a target hosting hardware platform is usually referred to as a *hypervisor*. There different types of virtualization techniques and technologies and the concept can be applied to different target technologies such as operating platforms, networks, and storage devices.

Multi-tenancy is a critical aspect in today's cloud systems and generates a number of security, privacy, and legal issues. This stems form the fact that "[...] the location of code and/or data is principally unknown and the same resource may be assigned to multiple users (potentially at the same time). This affects infrastructure resources as well as data/applications/services that are hosted on shared resources but need to be made available in multiple isolated instances." ([31], p. 15)

Metering is necessary to allow for pricing, charging, and billing. A major benefit—but also a requirement—of cloud computing is that virtual machine instances can be acquired and destroyed easily and in a short time. This requires cloud providers to be able to charge and bill with a very high degree of automation. Therefore, metering is a prerequisite and has a notable impact on the economic aspects of cloud computing discussed in the next paragraph.

Economic aspects Among the most important economic aspects are: (1) *outsourcing and cost reduction*, (2) *pay-per-use*, and (3) *time to market*.

Outsourcing and cost reduction are among the main industrial drivers for cloud computing. Acquiring infrastructure, platform, and software as outsourced services instead of hosting all the necessary systems in-house is a common practice in industry today. The concept is applied in many fields ranging from call center business to human resources. The aim of the outsourcing organization is usually to be able to concentrate on the core business rather than dealing with all the problems related to the operation and maintenance. By acquiring this as a professional service, positive effects on the core business are expected, leading to overall *cost reduction* and improved business results.

Another economic aspect of cloud computing is the *pay-per-use* model. Here, customers only pay for the resources actually used. This differs from flat-rate models where the access to services is charged independent of the amount of resources actually consumed. The *pay-per-use* model allows for great transparency and comparability in terms of infrastructure cost. This is especially interesting for smaller industrial players (SMEs) that are more concerned about infrastructure investments (capital expenditures (CAPEXs)) and related operational expenditures (OPEXs).

Furthermore, utilizing cloud services can bring a positive impact on the *time to market*. Building upon externally acquired resources, innovative ideas and new products can be launched in beta status or early adoption phase without costly infrastructure related investments. Again, this is especially beneficial for SMEs as in this way, new ideas and products can be tried out and tested in real markets very fast. If something is successful and needs further supporting infrastructure or platform support, cloud computing systems allow to scale up and down as needed relying on the mechanism of elasticity as introduced on page 25. This lowers the overall risks associated with innovative product development and market introduction as CAPEXs and OPEXs can be reduced. But also for larger players such as the Telcos, there are opportunities to adopt the cloud paradigm [32] to improve infrastructure operation with a positive impact on expenditures.

3.3.3 Standardization

As the concept of cloud computing started to receive broad attention from industry and academia only a couple of years ago, standardization efforts are still in an early stage. Nearly all well-known standard development organizations (SDOs) in the field shaped cloud computing working groups and technical committees. It remains to be seen which standards will receive sufficient industry adoption. The following subsections shall provide an overview of the most important standardization organizations and the cloud standards produced by them.

Open Grid Forum

The Open Grid Forum (OGF) is "[...] an open community committed to driving the rapid evolution and adoption of applied distributed computing." [33] Although the OGF started as a community of users, developers, and vendors in order to drive standardization for grid computing, the Open Cloud Computing Interface Working Group (OCCI-WG) [34] was initiated. Its focus is mainly on the IaaS layer of the cloud, producing the Open Cloud Computing Interface (OCCI) specification [35] that defines an IaaS API. Today, OCCI provides a remote management API with a strong focus on interoperability allowing for tasks

such as resource deployment, autonomic scaling, and monitoring. The OCCI website states that the "[...] current release of the Open Cloud Computing Interface is suitable to serve many other models in addition to IaaS, including e.g. PaaS and SaaS." [34]

Storage Networking Industry Association

Since 1997, the Storage Networking Industry Association (SNIA) is non-profit trade association, forming and sponsoring technical work groups to address the challenges of the storage and information management market [36]. The SNIA Cloud Storage Initiative (CSI) drives the Cloud Data Management Interface (CDMI) standard [37] to promote elastic, on-demand Cloud storage. CDMI allows to tag "[...] data with special metadata (data system metadata) that tells the cloud storage provider what data services to provide that data (backup, archive, encryption, etc) [...]" [38]. Using the CDMI, users can move "[...] data from cloud vendor to cloud vendor without the pain of recoding to different interfaces." [38] In addition, the CSI has published a number of work-in-progress drafts targeting a storage reference model [39], storage use cases [40], storage management, etc. Furthermore, there is a SNIA software development project that implements the CDMI.

Object Management Group

Since 1989, the OMG is an international not-profit computer industry consortium [41]. The focus of the specifications produced by the OMG is on modeling standards such as UML and MDA. However, the different OMG task forces develop standards for various technologies. The OMG Cloud Services Specification Project (CSSP) has been started to address the challenges of cloud computing. The project aims at producing reference documents and models, harmonize standards with SOA, and provide implementation guidance and service specifications. Among the measurable objectives of the project are (1) to coordinate OMG efforts in the cloud field to avoid standards duplication, (2) to develop a cloud ontology, (3) to develop a cloud type specification, and (4) to develop a *cloudML* meta-model in the style of the OMG Service oriented architecture Modeling Language (SoaML) [42]. However, at the time of writing, no concrete specifications are available. Member submissions such as [43] are available via the CSSP webpage.

Organization for the Advancement of Structured Information Standards

OASIS is a "[...] not-for-profit consortium that drives the development, convergence and adoption of open standards for the global information society." [44] OASIS was founded in 1993, at that time under the name SGML (Standard Generalized Markup Language), and changed its name in 1998 to OASIS. Within the OASIS, there is a technical committee (TC) that deals with identity management (IDM) issues in the cloud. The OASIS IDCloud TC [45] aims at identifying gaps in existing IDM standards. It performs risk analyses and produces guidelines for mitigating identified vulnerabilities. At the time of writing the planned specifications are in a very early stage. A draft version of a use case document has already been published [46].

Distributed Management Task Force

The DMTF [47] was founded in 1992 and produces standards focusing on inter-operable IT management. The DMTF Cloud Management Working Group [48] targets standards that

improve cloud management interoperability between service providers and users/developers. Besides a couple of whitepapers in the area of cloud management (e.g. [25], [26]) and related use cases, this group produced the Open Virtualization Format (OVF) as part of the Virtualization Management (VMAN) specifications[1]. The VMAN initiative was launched in 2008 to deliver virtual appliance interoperability standards. OVF is a packaging format[2] developed to enable the portability of virtual appliances across different platforms. One of the major benefits is the easy and less error-prone deployment across different virtualization platforms.

Open Cloud Consortium

The Open Cloud Consortium (OCC) is: "[...] a member driven organization that:

1. manages testbeds for cloud computing, including the Open Cloud Testbed and the Intercloud Testbed;
2. manages cloud computing infrastructure, such as the Open Science Data Cloud;
3. develops reference implementations, benchmarks and standards.

The Open Cloud Consortium is organized into different working groups." [49] At the time of writing no public standards were available. However, in addition to the OCC testbed[3], several cloud related projects and implementations have been published, such as [50] and [51].

The Open Group

The Open Group is "a vendor- and technology-neutral consortium, whose vision of Boundaryless Information Flow will enable access to integrated information within and between enterprises based on open standards and global interoperability." [52] The Cloud Work Group within the Open Group wants "[...] to create a common understanding among buyers and suppliers of how enterprises of all sizes and scales of operation can include Cloud Computing technology in a safe and secure way in their architectures to realize its significant cost, scalability and agility benefits. It includes some of the industry's leading cloud providers and end-user organizations, collaborating on standard models and frameworks aimed at eliminating vendor lock-in for enterprises looking to benefit from cloud products and services." [52] The list of Open Groups members includes large players such as Hewlett Packard, IBM, Oracle, and SAP. However, no concrete standards are available via the Open Group Cloud Work Group webpage at the time of writing. Member submissions such as [53] have been published.

Cloud Computing Interoperability Forum

The Cloud Computing Interoperability Forum (CCIF) is a consensus building non-profit community seeking to establish a common framework/ontology, and standardized cloud computing reference architectures enabling two or more cloud platforms to exchange information in an unified manor [54]. At the time of writing no public standards were available, but there is an active API development project called the *Unified Cloud Interface Project*[4] that aims at

[1] http://dmtf.org/standards/vman
[2] http://dmtf.org/standards/ovf
[3] http://opencloudconsortium.org/testbed
[4] http://code.google.com/p/unifiedcloud

creating a standardized cloud interface based on the Resource Description Framework (RDF). The target is the unification of various cloud APIs relying on a semantic cloud data model.

TM Forum

The TM Forum (formerly teleManagement Forum) (TMF) is an industry association providing industry standards and expertise to enable the creation, delivery and monetization of digital services [55]. A Cloud Services Initiative[5] has been launched by the TMF. "The centerpiece of this initiative is an ecosystem of enterprise customers, cloud service providers and technology suppliers who collaborate to define a range of common approaches, processes, metrics and other key service enablers. The Enterprise Cloud Leadership Council (ECLC) which consists of a group of enterprise customers serves as the anchor for this ecosystem." [55] The TMF offers case studies, presentations, webinars, whitepapers, and training via their website. However, many documents are only accessible for TMF members.

Cloud Security Alliance

The Cloud Security Alliance (CSA) is a non-profit organization working on best practices for providing security assurance within and education on the use of cloud computing [56]. For example, the CSA issues the CSA Certificate of Cloud Security Knowledge[6]. The CSA was founded in late 2008 and produced a number of whitepapers and technical reports since then, such as the Cloud Controls Matrix[7], a set of security guidance best practices[8], an identity & access management whitepaper [57], as well as a report on the top threads in cloud computing [58].

European Telecommunications Standards Institute

The European Telecommunications Standards Institute (ETSI) is a European non-profit organization that "produces globally-applicable standards for Information and Communications Technologies (ICT), including fixed, mobile, radio, converged, broadcast and internet technologies.." [59]. The technical committee GRID of ETSI was established in 2006 and looks at grid and cloud computing interoperability. However, no cloud standards have been published by the time of writing. For more details on grid standards see also section 3.4.2.

International Telecommunication Union Telecommunication Standardization Sector

The Telecommunication Standardization Sector of ITU (ITU-T) publishes standards for telecommunications on behalf of the International Telecommunication Union (ITU). "ITU-T Recommendations are defining elements in information and communication technologies (ICTs) infrastructure. Whether we exchange voice, data or video messages, communications cannot take place without standards linking the sender and the receiver." [60] The Focus Group on Cloud Computing (FG Cloud)[9] was established in February 2010 and aims at

[5] http://www.tmforum.org/EnablingCloudServices/8006/home.html
[6] http://www.cloudsecurityalliance.org/certifyme.html
[7] http://www.cloudsecurityalliance.org/cm.html
[8] https://wiki.cloudsecurityalliance.org/guidance/index.php/Main_Page
[9] http://www.itu.int/en/ITU-T/focusgroups/cloud/Pages/default.aspx

analyzing the cloud computing standardization requirements from a telecommunication point of view based on collaboration with worldwide cloud computing communities.

3.4 Grid Computing

3.4.1 Overview

Grid Computing evolved from the concept of *metacomputing* [61]. Metacomputing aims at integrating distributed computing resources using high speed network links to form a *metacomputer* (a distributed supercomputer). In a broader sense, *grid computing*, as well as cloud computing, can be seen as a form of *utility computing*. In utility computing, a service provider offers IT resources as a metered service. The service consumer is charged based on the amount, time, or other criteria of resource consumption.

> *A grid is a system that is concerned with the integration, virtualization, and management of services and resources in a distributed, heterogeneous environment that supports collections of users and resources (virtual organizations) across traditional administrative and organizational domains (real organizations).* [62], p. 8

The concept of *resources* and *services* can be compared to the equivalent terms in SOAs. Resources provide capabilities and are accessed through services. Grids differentiate between *real* and *virtual* organizations. *Real* organizations provide the resources to be accessed by *virtual* organizations that represent sets of individuals and/or institutions engaging in collaborative *problem-solving*.

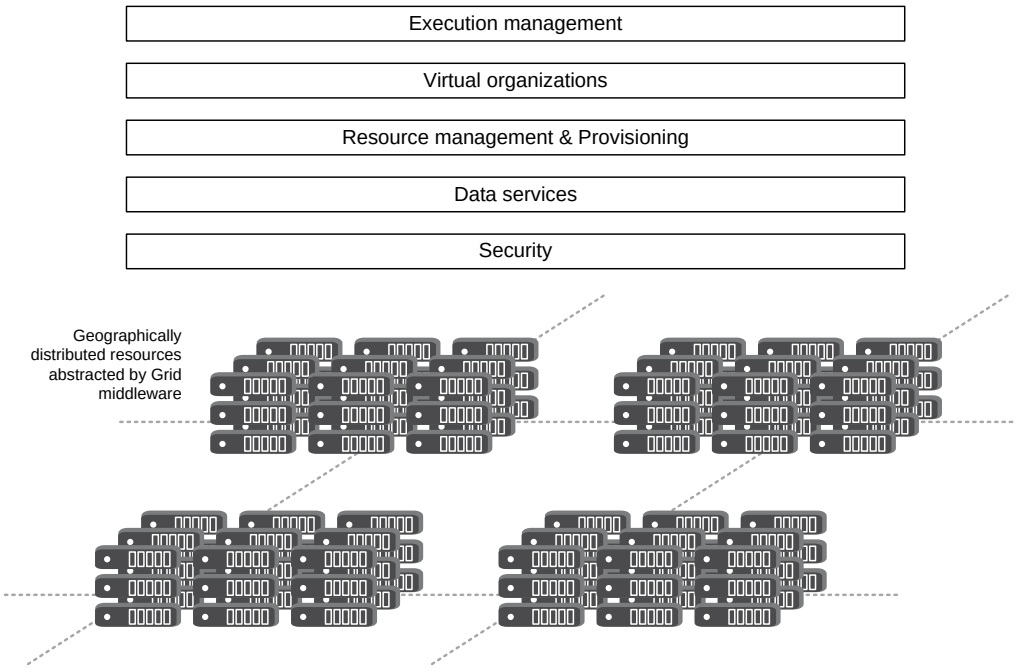


Figure 3.4 – High level conceptual overview of grid computing

Typically, resources that are shared in grid computing systems are computational and storage resources to solve computing intensive problems. Heterogeneity usually refers to the operating environments in which the target resource types are embedded. Through grid middleware, the different operating environments are homogenized in order to offer higher level services and overlay concepts (i.e. virtual organizations) as shown in figure 3.4.

The *problem-solving* oriented approach is a key characteristic of grid computing. Grids can be effectively used today for collective *job* execution. A job is a "[...] user-defined task that is scheduled to be carried out by an execution subsystem." ([62], p. 9) Job execution usually involves job scheduling, management, resource allocation, deployment, and configuration. More general requirements of grid systems are resource discovery and query, common management capabilities, resource virtualization, as well as standard protocols and schemas to ease system interoperability. Additional aspects are data services such as data storage, access, transfer, update, and persistency. Also, security, QoS, service level agreements (SLAs), scalability, and ease of use are central points in modern grid systems.

The Open Grid Services Architecture (OGSA) [63] provides a collections of standards that intend to standardize most of such aspects. OGSA defines the services and their exposed interfaces, the resource states needed to provide the services, and service interaction in a SOA style. OGSA builds upon several other specifications in the field of Web Services (WSs). Among the most important WS related standards are the OASIS Web Services Resource Framework (WSRF) and the World Wide Web Consortium (W3C) Web Services Management standard families, for example WS-Resource [64] or WS-Transfer [65].

A more detailed overview of SDOs and their standards in the field of grid computing is given in the following section.

3.4.2 Standardization

Open Grid Forum

As the name suggests, the OGF is one of the key SDOs in the field of grid standards in addition to their involvement in cloud standardization as described in section 3.3.3. The OGF grid standards are heavily based on SOA standards in line with the WSRF model provided by OASIS. Among the most important standards published by the OGF is the Open Grid Services Architecture (OGSA) [63].

European Telecommunications Standards Institute

The grid activities within ETSI are lead by the *GRID technical committee (TC)*[10]. The TC GRID ams at inter-operable applications and services in grids. The main (but not exclusive) focus is on: (1) resource and service access, (2) middleware, protocols, and APIs, and (3) security. Also, TC GRID initiates plugtests and workshop events to analyze latest solutions and trends in the field. Among other documents and standards, the TC GRID published an extensive multi-part technical report on grid computing in 2009 [66].

Organization for the Advancement of Structured Information Standards

Many of the standards published by other SDOs in the grid domain such as OGF build upon the work performed by OASIS, a consortium that "[...] produces more Web services standards

[10] http://www.etsi.org/WebSite/Technologies/GRID_CLOUD.aspx

than any other organization [...]" ([66], p. 52). Most relevant in the context of grid computing are the OASIS SOA reference model[11] and WSRF[12] family standards.

World Wide Web Consortium

The W3C creates Web standards and guidelines called *W3C Recommendations*. Although the W3C is not directly involved in grid standardization, its standards such as the *Extensible Markup Language (XML)* as well as the *Semantic Web* (e.g. RDF, Web Ontology Language (OWL), and SPARQL) and *Web Services* (e.g. Simple Object Access Protocol (SOAP) and Web Services Architecture) families have considerable impact on other SDOs in the grid domain.

3.5 Federated Identity Management

Identity federations are a specific case of federated systems in which multiple organizations agree to allow access to resources based on agreed processes for the handling of user identities. The most obvious benefit of *federated identity management (FIDM)* is Single Sign-On (SSO) allowing to re-use digital identity information across different services.

A key concept in *FIDM* is the separation of resource access and authorization from identity registration and authentication. Organizations that agree to federate in such a way are only concerned about user identities issued by themselves. For such identities they perform authentication if required by their own systems as well as upon request by other *trusted* organizations.

Trust is a central aspect in identity federation. A common term in FIDM is a *Circle of Trust (CoT)* which collectively represents the federation partners that agree to trust each other and allow the access to resources for users that are known and authenticated by other *CoT* members. This form of authentication delegation is useful as federating organizations only need to keep their affiliated user records and associated credentials up to date for users that have initially registered with them. For external users, the authentication decision is delegated to the issuing organization as it is expected that authentication can happen there based on up-to-date credentials.

Different organizational models have been investigated by the *Liberty Alliance* that lead to different CoT arrangements . The different models are shown in figure 3.5 ([67], p. 4) and figure 3.6 ([67], p. 5-6).

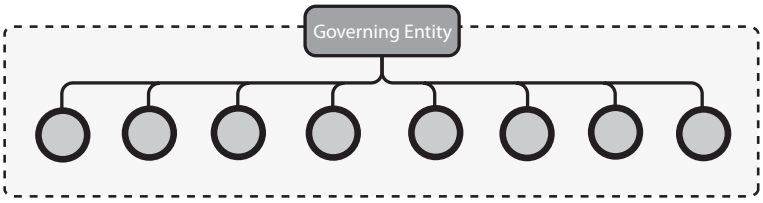


Figure 3.5 – Collaborative model – central operation and governance of a CoT by a single governing organization

[11] http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=soa-rm
[12] http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsrf

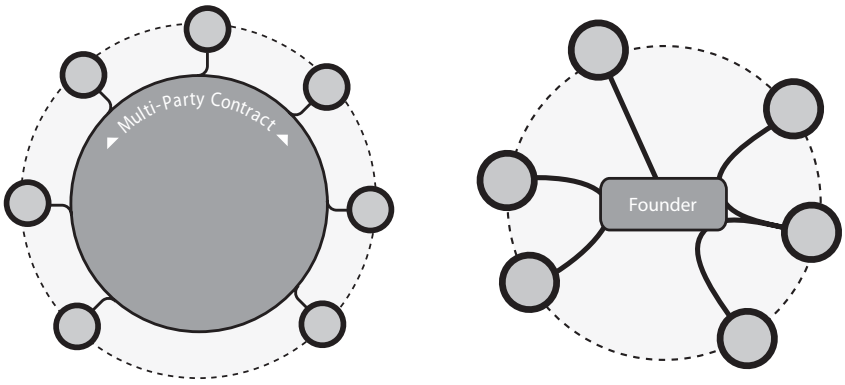


Figure 3.6 – Left: Consortium model – A multi-party contract between the founding organizations determines rules and governance of the CoT.
Right: Centralized Model – A single founding organization contracts other CoT members where the central entity can broker n-party relationships.

Authorization decisions whether to allow or deny access to particular resources are usually taken based on different identity attributes according to the policies of the authorizing organization. Therefore, the federating organizations usually agree on specific identity attributes shared across organizations (e.g. full name, role, etc.). Such agreements might also be established in an ad-hoc manner where the authorizing organization requests the authentication of certain attributes, which, in turn, might be accepted or rejected by the authenticating organization. This approach is usually followed in very large federations where initial agreements among all CoT members are not feasible or are hard to achieve. In such situations, trust is established by relying on accepted and standardized protocols and mechanisms. Generally, scalability is a difficult problem in identity federations following the CoT approach. This will be addressed again after the introduction of some additional FIDM terminology.

Most FIDM systems rely on the following entities: (1) user or principal, (2) identity provider, (3) service provider. The relationships between those are shown in figure 3.7 and are further exemplified in figure 3.8.

The user/principal wishes to access resources at a services provider and has an identity. The identity is issued and guaranteed by the Identity Provider (IdP). An example of an IdP is a governmental institution issuing passports and other official identity documents for principals. The service provider offers resources via services consumed by the principal and needs to authenticate the principal to decide whether to provide or deny resource access. Authentication and authorization decisions might (but do not have to) be delegated. In any case, whether or not the service provider authenticates/authorizes the principal itself, it needs to trust the identity issued by the IdP.

In digital IDM systems, users can have several *persona* and associated credentials, attached to the same *identity*. Merriam Webster defines identity as follows:

> *[...] 2 a : the distinguishing character or personality of an individual : individuality [...]* [68]

Whereas a persona is:

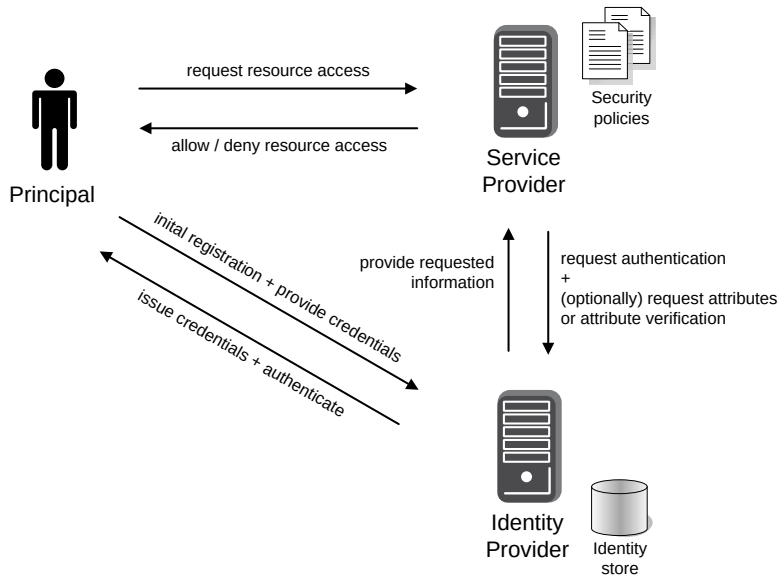


Figure 3.7 – Federated identity management roles and relationships

> *[...] an individual's social facade or front that [...] reflects the role in life the individual is playing [...]* [69]

Usually, individuals take on different persona depending on the social context in which they are acting, for example: *"at work"*, *"at home"*, or *"online"*. The context affects the way individuals are acting while their identity stays the same. A practical example of what makes up an identity is the information printed on individuals passports: full name, nationality, birth date, etc. However, how and why this information is important in different contexts, impacts the persona that are attached to the same identity.

For example, the birth date of an individual stays the same, but in different contexts, it is unequally important if an individual is *"over 18"*, *"over 21"*, or *"under 40"*. Which attributes that belong to an identity are presented and the way in which they are presented in different contexts make up different persona belonging to the same identity. Figure 3.9 ([70], p. 441) shows the different forms of attributes.

Such examples show that the selection of the attributes that are important for authorization decisions is not always easy. This is one on the disadvantages of FIDM systems. The definition of organization-wide authorization policies as well as the initial investments in FIDM system infrastructure are considerable hurdles for organizations to join an identity federation. In addition, the initial trust establishment among circle of trust members can be difficult. This is influenced by the fact that different organizations might have different security requirements based on the different levels of risk associated to unauthorized resource access. Therefore, identity federations are commonly observed among similar organizations, e.g. universities. For example, there is a global identity federation of universities based on the eduroam[13] system. While developing the ideas for the federation framework presented in this thesis, some identity aspects and the impact on the Telco domain have been investigated [71] by the author.

[13] http://www.eduroam.org

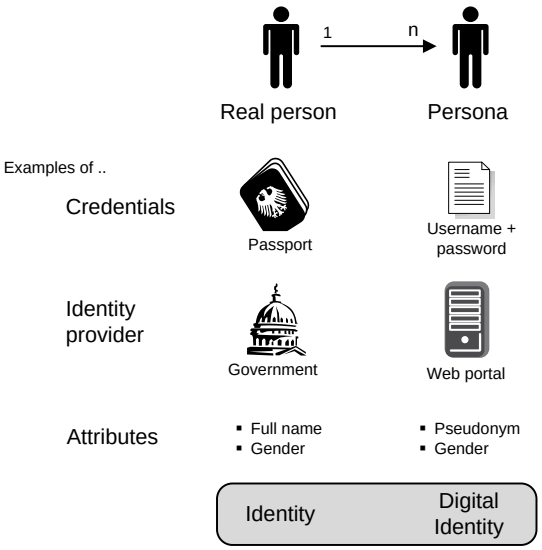


Figure 3.8 – Examples of FDIM terminology and the mapping from an identity hold by a *real person* to a digital identity represented by a *persona*

Another two important groups of federation related IDM standards are the Web Services Federation Language (WS-Federation) and Public Key Infrastructures (PKIs). WS-Federation allows to federate different security realms, "[...] such that authorized access to resources managed in one realm can be provided to security principals whose identities are managed in other realms. While the final access control decision is enforced strictly by the realm that controls the resource, federation provides mechanisms that enable the decision to be based on the declaration (or brokering) of identity, attribute, authentication and authorization assertions between realms." ([72], p. 6)

PKIs allow to create, manage, validate, and revoke digital certificates that bind a public key to an identity. In that way users can be authenticated at third party services if the service provider trusts the issuing certification authority. In this regard, the PKI concept allows for secure data communication in *trust federations*.

3.6 Future Internet Experimental Facilities

The current Internet has a number of limitations resulting from its constant evolution as a global system. Numerous patches have been applied over time to improve various aspects such as security, management, and others. Therefore, the Internet has evolved into as a huge patchwork that faces issues like high complexity, security threads (e.g. phishing, pharming, viruses, etc.), high management cost, and the lack of mobility and privacy support.

Due to such limitations, the discussion on the architecture of a *Future Internet (FI)* has started. In a so-called *clean slate* approach, scientists worldwide re-consider the design principles of the Internet and investigate alternative solutions. The idea is to forget about current deployments and how interoperability with legacy systems can be achieved and to see what could be improved in terms of architecture design if it would be done again *from scratch*. One of the first initiatives into this direction was the United States of America (U.S.) Future Internet Design (FIND) initiative. "The philosophy of the program is to help conceive

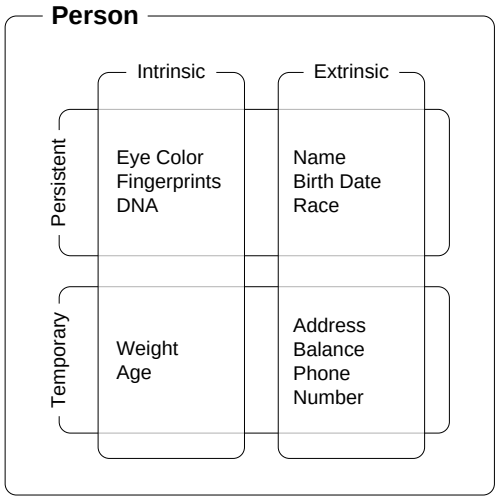


Figure 3.9 – The different attribute types that can be associated to a person are temporary, persistent, intrinsic, and extrinsic attributes.

the future by momentarily letting go of the present." [73]

The drawback of the clean slate approach is obvious. Millions have been invested in the infrastructure that was necessary to build the current Internet. Without demonstrating a path of evolution, the ideas that are developed in a clean slate way will not be adopted by the infrastructure operators anytime in the near future. A possible solution to this dilemma is the deployment and operation of FI experimental facilities. Such facilities allow for the trial of clean slate concepts and technologies in addition to the productive networks that provide the Internet backbone today. This idea of deploying revolutionary concepts and systems in an experimental environment and to allow for successive adoption is not new, as this is how the Internet started in the first place. The scientific deployments of the Advanced Research Projects Agency Network (Arpanet) and the WWW at the European Organization for Nuclear Research (CERN) show that the scientific communities have been the early adopters of revolutionary technology all along.

However, also *evolutionary* approaches to the design of a FI rely on large scale experimental facilities. Today's FI research activities can be structured into several categories, namely the *Internet of Services (IoS)*, *Internet of Things (IoT)*, *Internet of Media and Content (IoM/C)*, as well as the supporting networking infrastructure (*Network of the Future (NoF)*).

Internet of Things (IoT) "[...] more and more 'things' around us are electronically addressable and controllable. This trend is expected to continue and further increase. With IPv6, even today there is a huge addressing space available and many more things will be addressable in the future. Also, with the increased use of semantics and information abstraction, things are expected to self-organize and behave more intelligently." ([24], p. 102)

Internet of Services (IoS) "In recent years service oriented computing has heavily influenced the way of providing internet-scale applications. A most prominent example is the emergence of Web2.0 and the related application mashup phenomenon. Service orientation and service orchestration will be the dominating design paradigm for the

Future Internet. The idea behind service orientation is to encapsulate atomic functionality and provide it as a service. Such services can be requested by humans or other services. This leads to a coupling of atomic functionalities into larger service blocks (or service chains) where a service can be composed by any number of other services. [...] In this way, service orchestration leads to composite applications that are a specific aggregation of a number of services that are offered by different independent sites. [...]" ([24], p. 102)

Internet of Media and Content (IoM/C) "Digital media is increasingly replacing analog media. This will eventually result in a complete digitalization of media (for example video) and content (for example books and magazines). Today it is difficult to cope with the amount of digital information being produced worldwide both by organizations (for example press) as well as end users (user generated content). This is mainly due to the lack of semantic interpretation of information and proper service discovery functionalities. A lot of research activities are currently under way to improve current approaches of semantic service and content annotation (tagging) and it is expected that inference systems will intelligently process the constantly increasing amount of available information and present it in an understandable and optimized way to different classes of requesters." ([24], p. 102)

Network of the Future (NoF) "[...] In the future, a network of networks will support a wide variety of nomadic and mobile interoperable devices, innovative services, ICT tools and applications, content formats and delivery modes. A new generation of telecom infrastructure, network and internet technologies will be used in the coming years as fundamental building blocks, supporting health, environment, government, transport, entertainment and education to name just a few examples. The Future network will provide underlying support for the Future Internet, enabling new business models and a multiplicity of devices, networks, and service providers [...]" [74].

In order to address the wide range of research aspects related to both *revolutionary* and *evolutionary* FI research, experimental facilities are needed. Such facilities allow experimentation with and testing of innovative concepts and prototypes. In this context *experimentally-driven research* is defined as

> *[...] visionary multidisciplinary research, defining the challenges for and taking advantage of experimental facilities [...] by means of iterative cycles of research, oriented towards the design and large-scale experimentation of new and innovative paradigms for the Future Internet - modelled as a complex distributed system. Therefore, research is expected to address all the associated aspects in a holistic vision, at all relevant levels and layers. [...]* [2], p.2

Figure 1.1 shows that the usage of large scale experimental facilities is often the final step of an experimentally-driven research process. Usually, new ideas are initially modeled using formal models and are then simulated using simulation techniques. As it becomes increasingly difficult to model and simulate the real world due to the increasing complexity of today's systems ([75], p. 1), the level of experiment realism is gradually increased using emulation techniques and real testbeds or experimental facilities. However, with an increasing level of realism, the associated costs and experiment complexity are also increased. To support

experimenters along the entire chain from the abstract model to a real large scale deployment, various experimental facilities have been established in the past years. Federation among such facilities can cut down costs and allow for additional types of experiments that are very hard and costly to perform without federation.

The following sections cover different experimental facility initiatives from a federation point of view.

3.7 GENI Initiative

The Global Environment for Network Innovations (GENI) initiative [76] funded by the U.S. National Science Foundation (NSF) started in 2007. The program is managed by the GENI Project Office (GPO) which is run by the private company BBN Technologies. The GPO defines the overall GENI system architecture (see figure 3.10 ([77], p. 17)) and requirements [78], identifies risks, and supports the GENI project prototypes integration.

The GENI architecture includes several building blocks and functions such as "[...] slices, components, aggregates, and a clearinghouse. Components are the offered resources that are independently owned and operated. Components can be organized into aggregates, which are groups of resources owned and administered as an ensemble by some organization. Aggregates and its components are available for experiments via a control framework run by a clearinghouse (there will be multiple clearinghouses which will federate). A slice is a set of slivers (a sliver is a part of a specific resource) spanning a set of network components, plus associated users that are allowed to access those slivers for the purpose of executing an experiment." ([79], p. 275)

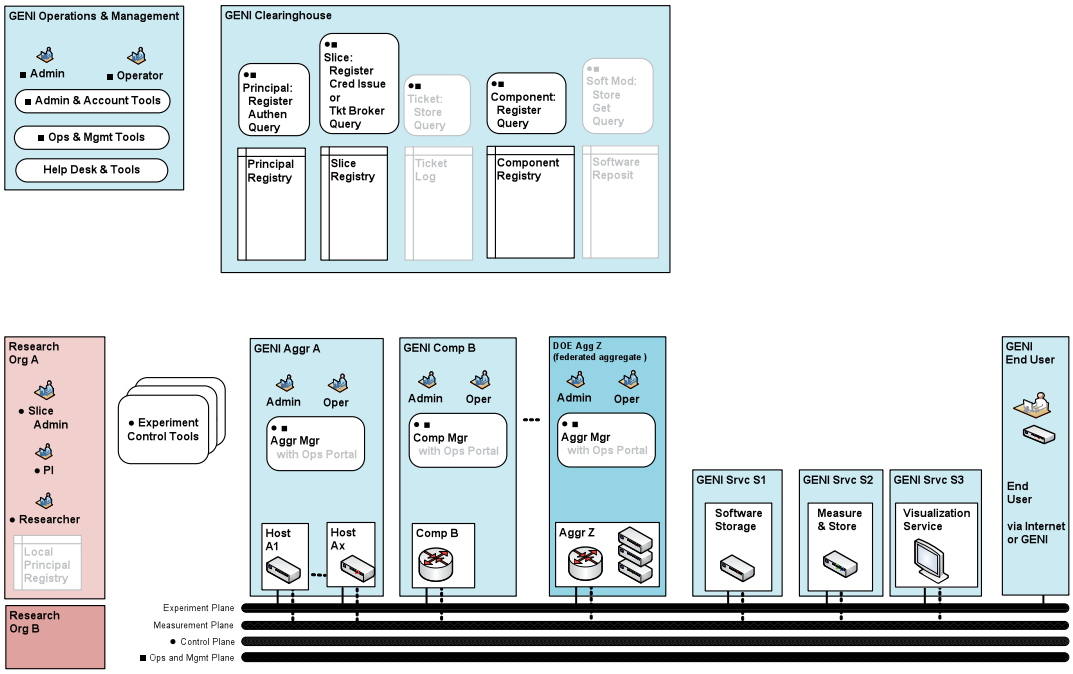


Figure 3.10 – GENI system overview

The GENI projects are funded in so-called spirals that typically last 12 month and are

organized in clusters. The first spiral featured 5 clusters that built upon competing control frameworks which are briefly introduced in the following. A full overview and comparison has been published in [79].

Cluster A: DETER control framework "TIED (Trial Integration Environment built on DETER) builds upon an architecture (DETER Federation Architecture, DFA) for creating experimental environments across multiple cooperating Emulab-based testbeds. [...] The architecture involves domain-specific testbed and experiment descriptions, domain specific splitters, a canonical experiment description, and a federator component. The experimenter defines an experiment using a domain-specific creation tool. The domain specific description is transformed into a topology description in a standard language (currently an Emulab topology description language based on ns simulator language called Canonical Experiment Description Language (CEDL)). The federator divides the experiment among testbeds (based on requirements, constraints and available resources), creates sub-experiments on each testbed and interconnects these to form the federated experiment. The resources are allocated by the federator through Emulab allocation interfaces. A distributed implementation of the federation system is enabled by WSDL interface descriptions and SOAP as well as XML-RPC remote procedure calls." ([79], p. 275) and [80]

Cluster B: PlanetLab control framework This control framework relies on the PlanetLab Central software (PLC) and the Slice-based Facility Architecture (SFAv1) and SFAv2. Resources are described in terms of a resource specification (RSpec). An in-depth discussion on PlanetLab is provided by section 3.7.1.

Cluster C: ProtoGENI control framework In GENI spiral 1, ProtoGENI relied on "[...] the Emulab software, and the Slice-Based Facility Architecture. It is also leveraging PLC and is mainly driven by University of Utah, Flux Research Group. [...] The name Emulab refers to the Emulab project, a facility (testbed) and a software. The Emulab project maintains the facility. The Emulab facility uses the software, which is also used by numerous other testbeds to control their infrastructure. [...] Resources in ProtoGENI are described by XML documents following RELAX NG Compact syntax (a schema language for XML). Currently [in spiral 1], ProtoGENI is using RSpec with descriptions for nodes, links, interfaces, and some metadata. In Emulab, resource mapping is realized by a tool called assign which maps virtual resources to local nodes and VLANs. A format called vtop/ptop (virtual topology/physical topology) is used for the definition of common data structures representing topologies." ([79], p. 275) A more detailed discussion of Emulab and ProtoGENI is provided by section 3.7.5.

Cluster D: ORCA control framework In spiral 1, the ORCA framework relied on "[...] a Java-based resource leasing toolkit called Shirako. It serves as Java reference implementation of a GENI clearinghouse (a broker component), aggregate managers (domain authorities), and experiment control tools. The involved entities (so-called actors) exchange digitally signed SOAP messages. All actors provide a web control portal interface for users and operators. The ORCA framework provides [in spiral 1] all essential clearinghouse functions (such as the design of a desired infrastructure setup) and enables a loose federation of substrate providers. A substrate provider offers resources to several clearinghouses. However, inter-clearinghouse federation (broker federation) is currently

[in spiral 1] not supported. The resources themselves are managed by a site/domain authority with the help of component agents (to configure and control the resource). ORCA does not impose any particular structure on shared resources." ([79], p. 275) and [81] A more detailed discussion of the ORCA framework is provided by section 3.7.3.

Cluster E: ORBIT control framework For spiral 1, ORBIT "[...] relies on the OMF (ORBIT Management Framework) [...] OMF is used by several testbed sites in Australia, Europe, and the US. ORBIT (Open Access Research Testbed for Next-Generation Wireless Networks) is a radio grid testbed for reproducible evaluation of wireless network protocols. OMF specifically aims at helping users to describe (in a high-level domain-specific language) and control an experiment, manage the testbed, and collect reproducible measurements. The OMF architecture includes experiment controllers (send instructions to resource controllers), aggregate managers (overall testbed control, communicate with resource managers), resource managers (control resources) and resource controllers (control only the part of a resource committed to an experiment)" ([79], p. 275 - 276) and [82]. OMF is now called Control and Management Framework [83]. Section 3.7.4 provides more details on ORBIT and OMF.

GENI spiral 2 ran from October 2009 until September 2010 and added 33 new projects to the total number of GENI projects. In spiral 2, an important steps towards the convergence of the competing GENI clusters and their respective control frameworks was achieved by defining the SFAv2 draft document [84]. Section 3.7.2 provides further insights into the SFAv2. Figure 3.11 ([85], p. 12) depicts the GENI spiral model in general and locates the second spiral phase as part of the overall process. At the time of writing, GENI is in its third spiral.

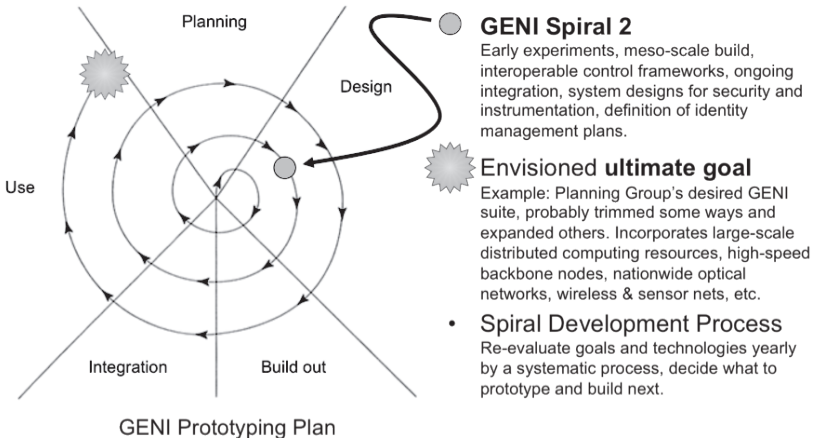


Figure 3.11 – The second GENI spiral provided a better interoperability of the clusters and their control frameworks.

3.7.1 PlanetLab

PlanetLab [86] was started in 2002 and provides access to PlanetLab nodes worldwide. A PlanetLab node is a computer that runs the PlanetLab Central (PLC) software. The nodes are provided by participating organizations, mostly universities and research institutes. Collectively, the PlanetLab nodes provide a global platform for scientific experiments. [87]

PlanetLab nodes "[...] run a minimal version of a UNIX operating system and are divided into virtual containers. The resources bundled by such a container are called sliver. PLC groups these slivers into slices. [...] PLC allows the party owning a slice to grant individual users access to it, who can use the slices to run experiments [...]" ([88], p. 931). Figure 3.12 (based on [88], p. 930) shows this concept. At the time of writing, PlanetLab provides access to 1090 nodes at 513 sites[14]. In order to create a slice, the experimenter has to be affiliated with one of the participating institutions. Participating institutions have to provide at least two PlanetLab nodes to the global resource pool and sign a membership agreement. Therefore, PlanetLab follows a *club good* model as individuals can be excluded from the consume of PlanetLab resources (as apposed to public goods) and usually do not rival (non-rivalrous good, as opposed to private goods).

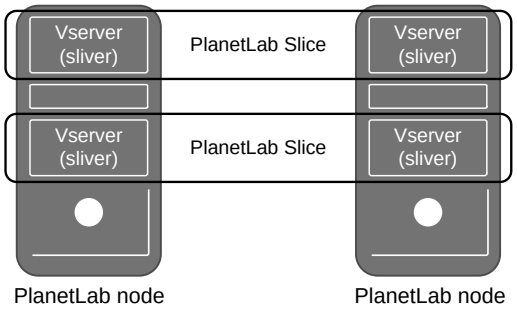


Figure 3.12 – PlanetLab node architecture and the concept of slicing distributed nodes

The pool of PlanetLab nodes is currently controlled by three authorities: (1) PlanetLab USA[15], (2) PlanetLab Europe[16], and (3) PlanetLab Japan[17]. The three authorities form the global PlanetLab federation. The PlanetLab control framework supports SFAv1 as introduced in the next subsection and also seeks to support SFAv2 as it matures.

3.7.2 The SFA Versions

In its first version SFA stood for slice-based *facility* architecture and is specified in [89]. In this thesis it is referred to as *SFAv1*. In the second version of SFA, a change in the name appeared. SFA stands now for slice-based *federation* architecture. It is specified by [84] and is referred to as *SFAv2* in this thesis. The entities defined by SFAv2 are closely aligned with those of PlanetLab. The key SFAv2 roles are: (1) the management authority (MA), (2) the slice authority (SA), and (3) the user.

The management authority "[...] is responsible for some subset of substrate components: providing operational stability for those components, ensuring the components behave according to acceptable use policies, and executing the resource allocation wishes of the component owner." ([84], p. 4)

The slice authority "[...] is responsible for one or more slices. It names and registers the slices and enables users to access and control their slices. The SA must provide a

[14] https://www.planet-lab.org/db/pub/sites.php
[15] http://planet-lab.org
[16] http://planet-lab.eu
[17] http://planet-lab.jp

contact interface to obtain information about the slice or to respond to any perceived misbehavior by the slice. MAs have the right to select which SAs are empowered to create slices on their resources." ([84], p. 4)

The user "[...] is a person playing one or more roles in a facility – a researcher that wishes to run an experiment or service in a slice, an operator that manages some part of the substrate, a PI [principle investigator] at an institution that conducts research on the facility, or an owner that contributes resources to a facility." ([84], p. 4)

In addition to those roles, SFAv2 defines several functions that are very similar to those of PlanetLab as shown in figure 3.12. The main functions are: (1) components, (2) aggregates, and (3) slices & slivers.

Components "Components are the primary building block of the architecture. For example, a component might correspond to an edge computer, a customizable router, or a programmable access point. A component encapsulates a collection of resources, including physical resources (e.g., CPU, memory, disk, bandwidth) logical resources (e.g., file descriptors, port numbers), and synthetic resources (e.g., packet forwarding fast paths). These resources can be contained in a single physical device or distributed across a set of devices, depending on the nature of the component. A given resource can belong to at most one component." ([84], p. 4)

Aggregate "Components are grouped into aggregates. All of the components of an aggregate are under the authority of the same MA, which also governs the aggregate. Each aggregate is controlled via an aggregate manager (AM), which exports a well-defined, remotely accessible interface. If an aggregate contains only a single component, then the AM may be called a component manager (CM). The AM/CM defines the operations available to user- level services to manage the allocation of component resources to different users and their experiments." ([84], p. 4)

Slices & slivers "It may be possible to multiplex (slice) component resources among multiple users. This can be done by a combination of virtualizing the component (where each user acquires a virtual copy of the component's resources), or by partitioning the component into distinct resource sets (where each user acquires a physical partition of the component's resources). In both cases, we say the user is granted a sliver of the component. Each component must include hardware or software mechanisms that isolate slivers from each other, making it appropriate to view a sliver as a 'resource container'." ([84], p. 4)

Furthermore, SFAv2 defines the format of identifiers for SFA entities and a couple of additional data types. It also specifies several interfaces for communication with the core SFA objects. The slice interface allows provisioning and controlling a slice, while the component management interface supports operations like rebooting the component and querying its status. The registry interface allows to read and write registry records, SFA defines the format of such records as well as authorization and access control.

Although the exact form of a resource specification (Rspec) is not specified by SFAv2, it draws some restrictions on the format of an Rspec by defining fields for the resource reservation time. The Rspec format that is currently used in the deployments of SFAv2 is similar to the

one used in PlanetLab in terms of its description capabilities. Essentially, an Rspec is an XML file listing resources and their characteristics. An example is shown in listing 3.1.

Listing 3.1 – An Rspec example showing one authority with one site and 3 nodes. Some nodes are already bound to a slice.

```
<RSpec type="SFA">
  <network name="fraunhofer" slice="my_experiment">
    <sliver_defaults/>
    <site id="s1">
      <name>fokus</name>
      <node id="n11">
        <hostname>fiesta.fokus.fraunhofer.de</hostname>
        <bw_limit units="kbps">10000</bw_limit>
        <sliver/>
      </node>
      <node id="n12">
        <hostname>doko.fokus.fraunhofer.de</hostname>
        <bw_limit units="kbps">10000</bw_limit>
      </node>
      <node id="n007">
        <hostname>spice.fokus.fraunhofer.de</hostname>
        <sliver/>
      </node>
    </site>
  </network>
</RSpec>
```

The example shows the `fraunhofer` site with 3 nodes where the two nodes with the tag `<sliver/>` are bound to the slice `my_experiment`. The node `n12` is still available.

3.7.3 The ORCA-BEN Project

ORCA [90] is the name for a software framework, a project, and a platform for distributed resource management. It is also one of the GENI control frameworks represented by GENI cluster D[18] (see also section 3.7). The *Breakable Experimental Network (BEN)* is a disruptive network research facility managed by Renaissance Computing Institute (RENCI). Jointly, the ORCA and BEN teams work on extending the ORCA-BEN facility and their *resource leasing* concept. A resource lease is a contract between the resource consumer, the resource provider, and a broker. The contract assures the holder access to a specific resource (or a set of resources) for a specific period of time. The ORCA framework includes several other projects that are advertised using ORCA as an umbrella:

- the Secure Highly Available Resource Peering (SHARP) framework [91],
- the Shirako resource leasing system [92],
- the Automat portal [93],
- and the Cluster-on-Demand (COD) platform [94].

[18] https://geni-orca.renci.org/trac

In the following, for the sake of simplicity, it is referred to ORCA as a collection of the projects mentioned above, without specifying which specific subsystem is responsible for a given functionality. The ORCA architecture defines the following actor roles: (1) an authority, (2) a service manager, and (3) a broker. The ORCA actors and their interactions are shown in figure 3.13 ([90], p. 5).

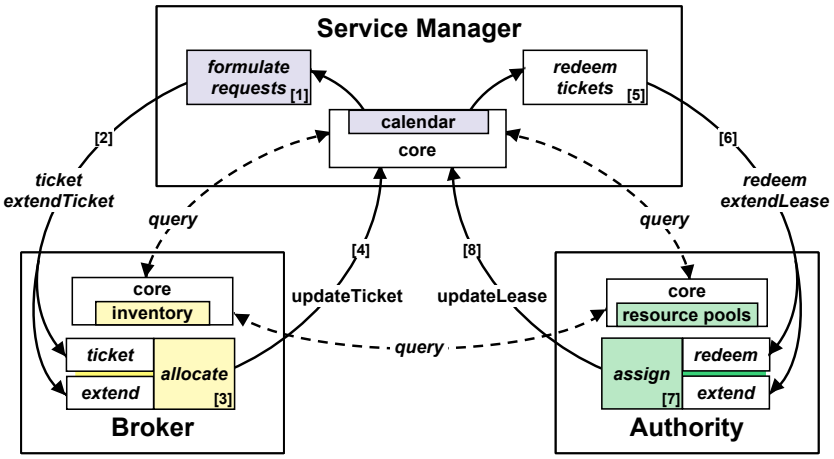


Figure 3.13 – The ORCA actors and interactions with arrows illustrating the leasing protocols.

The following list gives a short description of the ORCA actors as defined by [90]. Note that the authority and service manager can be mapped to the aggregate manager (AM) and slice manager respectively, as defined by GENI and the SFAv2. The ORCA team adopted some of the GENI terminology for the ORCA actors. Several instances of the actors can be active at the same time in an ORCA-based system.

Authority "An authority actor controls access to some subset of the substrate components. It corresponds directly to the aggregate manager (AM) in GENI. Typically, an authority controls some set of infrastructure resources in a particular site, autonomous system, transit domain, administrative domain, or component aggregate comprising a set of servers, storage units, network elements, or other components under common ownership and control." ([90], p. 5)

Service manager "This actor is responsible for creating, configuring, and adapting one or more slices. It runs on behalf of the slice owners to build each slice to meet the needs of a guest that inhabits the slice." ([90], p. 5)

Broker "A broker mediates resource discovery and arbitration by controlling the scheduling of resources at one or more substrate providers over time. It may be viewed as a service that runs within a GENI clearinghouse. A key principle in Orca is that the broker can have specific allocation power delegated to it by one or more substrate authorities, i.e., the substrate providers 'promise' to abide by allocation decisions made by the broker with respect to their delegated substrate. This power enables the broker to arbitrate resources and coordinate allocation across multiple substrate providers, as a basis for federation and scheduling of complex slices across multiple substrate aggregates. Brokers exercise this power by issuing tickets that are redeemable for leases." ([90], p. 5-6)

Furthermore, ORCA defines actor interfaces and authorization policy mechanisms. The identity system is based on Shibboleth[19] and attribute-based access control (ABAC) (see also section 3.5). ORCA relies on an advanced semantic resource description approach[20] based on OWL ontologies and the network description language (NDL). NDL is a schema (and an RDF-based ontology) initially proposed and developed by the University of Amsterdam, that allows describing network elements and network topologies. The ORCA way of describing resources is a potential candidate for resource descriptions in GENI beyond the currently used Rspec that is mainly influenced by PlanetLab and ProtoGENI.

3.7.4 ORBIT and OMF

The Control and Management Framework (OMF) is a testbed control framework initially developed for the Open Access Research Testbed for Next-Generation Wireless Networks (ORBIT)[21] [82] that provides an experimental facility for wireless technologies. However, today OMF provides tools can be used with different testbed technologies. Ott et al. summarize: "The cOntrol and Management Framework (OMF) is a suite of software components, which provides management, control, and measurement tools & services to users and operators of networking testbeds. [...] Through active development and extensions at NICTA, it has now evolved into an open source framework, which supports heterogeneous wired and wireless resources." ([83], p. 1-2)

Being an integral part of the ORBIT control framework within the GENI cluster E (see section 3.7), as well as the FP7 project Onelab2 (see section 3.8.1), OMF plays an important role for the experimental facilities in the U.S. as well as in Europe. Figure 3.14 ([83], p. 3) shows the OMF system architecture. It has to be noted that the dashed-line rounded boxes indicate functions that have not yet been developed but are expected to be included in future OMF releases. The key OMF system aspects and functions are: (1) an experiment controller, (2) an experiment description, (3) a resource manager/controller, and (4) a collection of measurement libraries.

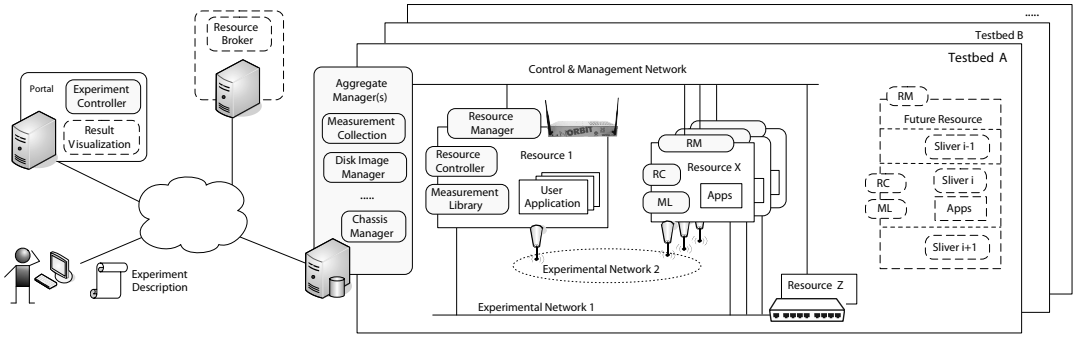


Figure 3.14 – The OMF architecture

The experiment controller is provided by an OMF portal and orchestrates the experiment based on an experiment description. Resource access is controlled through time slot reservation [83].

[19] http://shibboleth.internet2.edu/
[20] https://geni-orca.renci.org/trac/wiki/NDL-OWL
[21] http://www.winlab.rutgers.edu/docs/focus/ORBIT.html

The experiment description is formalized by means of a domain specific language—the OMF Experiment Description Language (OEDL). This allows to define all the details regarding an experiment such as: resource requirements, resource configurations, and a state machine describing experiment execution actions.

The resource manager/controller is part of the management plane and is responsible for configuring the resources according to the experiment description. Resource controllers execute actions on the actual resource as defined in the experiment description state machine. Therefore, this entity actually performs the experiment tasks directly on a node.

The measurement libraries allow recording metrics and experiment result storage.

Users may install custom Operating System (OS) images on OMF nodes. All system entities can communicate via the *control & management network* as shown in figure 3.14. Running experiments with allocated resources use the *experimental networks* for experiment traffic.

The key advantages provided by OMF is the experiment description language and the automated experiment execution based on the description. However, it remains to be seen if the OMF teams are successful in extending the descriptions beyond wireless resources in order to allow for more heterogeneous experiments. An example experiment description is shown in listing 3.2 ([83], p. 4). Currently, OMF has been released in version 5.3 allowing for federation between ORBIT and other testbeds relying on SFAv2.

Listing 3.2 – An example OMF experiment description

```
# Part 1 - Describe the resources required for this experiment
defGroup('source', [5]) {|node|
   node.prototype("test:proto:udp sender", {
      'destinationHost' => '192.168.0.10',
      'localHost' => '192.168.0.5'})
}
allGroups.net.w0 { |w|
   w.type = ''g'' # Use 802.11g
   w.mode = ''ad-hoc'' # Set interface in ' ad-ho' mode
   w.channel = ''6'' # Use channel 6
   w.essid = ''simple'' # Set SSID
   w.ip = ''%192.168.0.%i'' # Set the IP address to 192.168.0.i (i =5)
}
# Part 2 - Describe the state-machine and tasks to execute
# Here we only define one state 'whenAllInstalled' and associate 8 tasks to it
whenAllInstalled() { |node|
   wait 20
   info("Start all the applications")
   allGroups.startApplications
   wait 30
   info("Stop all the applications on 'source' group")
   group('source'). stopApplications
   info("Now Stop the experiment")
   Experiment.done
}
```

3.7.5 ProtoGENI and Emulab

The ProtoGENI project provides a "prototype implementation and deployment of GENI, led by the Flux research group at the University of Utah, and is largely based on [the] Emulab software. ProtoGENI is the Control Framework for GENI Cluster C, the largest set of integrated projects in GENI."[22] Emulab is a network emulation testbed based on the same-named software[23]. The software is used by projects worldwide to carry out network experiments in a controlled simulated/emulated environment.

ProtoGENI is a key facility within GENI that provides a nationwide, high-speed backbone on Internet2's wave infrastructure[24] that connects multiple wireless networks, edge clusters, and user programmable components, including standard PCs and servers as well as NetFPGA[25] cards. In addition, the facility includes a number of software installations (e.g. enhanced Emulab, PlanetLab, VINI[26]). To use ProtoGENI services, users must register an account on Emulab via the Emulab web interface. ProtoGENI builds upon a certificate-based PKI. The ProtoGENI interface runs over Secure Sockets Layer (SSL) that users can access presenting their SSL certificates. ProtoGENI services can be used via XML-RPC based interfaces. At the time of writing, ProtoGENI offers a component manager API, a clearinghouse, a slice authority API, and a slice embedding service. All services are described and documented on the ProtoGENI web[27].

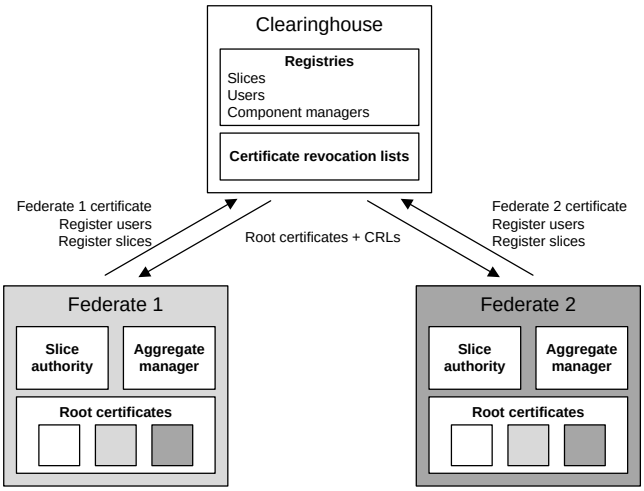


Figure 3.15 – High level view on the ProtoGENI federation approach

Resources (*components* in ProtoGENI terminilogy) are advertised, requested, and described by means of a ProtoGENI Rspec[28]. The Rspec details components in terms of resources and constraints and has lately been discussed as the format to be adopted by all GENI clusters for cross-cluster communication. From a federation perspective, ProtoGENI offers a loose federation based on trust relationships and a clearinghouse with central registries as shown by

[22] http://www.protogeni.net/trac/protogeni
[23] http://www.emulab.net
[24] http://www.internet2.edu
[25] http://www.netfpga.org
[26] http://www.vini-veritas.net
[27] http://www.protogeni.net/trac/protogeni
[28] https://www.protogeni.net/trac/protogeni/wiki/RSpec

Figure 3.15. The registries centrally hold data about component mangers, users, and slices. Therefore, resource can be discovered via the clearinghouse API[29]. Trust is established by centrally distributing root certificates and certificate revocation lists (CRLs). As each member of the ProtoGENI federated facility provides a local Emulab installation, user management is performed locally. Therefore, all users hold an encrypted SSL certificate that has been issued by their local Emulab installation (for example Federate 1 and Federate 2 in figure 3.15) and that is used for authentication at any site within the federation. The certificates convey the identity information for authentication. However, authorization is performed based on user privileges. Privileges are bound to user credentials which are XML documents that are provided along with a request and that have been digitally signed and can be traced to one of the trusted root authorities either directly or following the certificate chain.

3.8 FIRE Initiative

The Future Internet Research and Experimentation (FIRE) initiative [95] has been established by the European Commission (EC) as part of the seventh framework programme for research and technological development (FP7). FIRE aims at supporting Future Internet (FI) research in Europe by driving the concept of *experimentally driven research* [2]. Furthermore, FIRE seeks to provide a large scale European *experimental facility*. Federation plays a crucial role in this approach in order to gradually connect and extend existing testbeds providing FI technologies and know-how.

In contrast to GENI, FIRE is not centrally steered and organized by some sort of FIRE office equivalent to the GPO. Instead, FIRE funds a number of research projects that act independently from each other. However, a number of support actions (e.g. FIREworks[30] and FIRESTATION[31]) seek to align the outcomes of those projects and facilitate collaboration. The drawback from this approach is that although the individual projects might be successful, the overall FIRE facility lacks agreed procedures and technologies for federation. The EC recognized this and established the *working group on modular federation of FIRE facilities* that published a report in 2009 [14]. However, at the time of writing the FIRE facility does not build upon a consolidated federation architecture. Individual projects drive federation with selected peers within and outside of Europe (for example Onelab2 and PlanetLab, see section 3.8.1). The following subsections provide an insight into the federation approaches of the most important FIRE projects.

3.8.1 Onelab2 and PlanetLab Europe

The FP7 integrating project (IP) *Onelab2* [96] started in 2008 and provides a federated facility for FI research. Building upon PlanetLab software, Onelab2 operates the PlanetLab Europe (PLE)[32] that is federated with PLC and PlanetLab Japan (PLJ) allowing for mutual resource sharing.

In addition to the PlanetLab based facilities, Onelab2 provides a testbed for wireless

[29] http://www.protogeni.net/trac/protogeni/wiki/ClearingHouseAPI2
[30] http://www.ict-fireworks.eu
[31] http://ict-fire.eu
[32] https://www.planet-lab.eu

resources (i.e. the NITOS tetsbed[33]), the DIMES testbed[34] for distributed topology measurement, and the ETOMIC advanced network measurement system[35]. Furthermore, Onelab2 includes partners that are actively involved in the OMF and SFAv2 developement. In fact, the PLC-PLE federation is based on the PlanetLab SFAv2 implementation that has been considerably advanced by partners of the Onelab2 project.

3.8.2 The FP6 Panlab and FP7 PII Projects

The idea of a pan-European laboratory that provides experimental resources across multiple sites in Europe started in 2006 with the sixth framework programme (FP6) specific support action (SSA) *Panlab*. The Panlab SSA produced a number of initial concepts and ideas[36] around testbed federation, but no prototypes were delivered at that time. The development of Panlab prototypes started in 2009 driven by the FP7 IP Pan-European laboratory infrastructure implementation (PII) [97]. Figure 3.16 [98, 99] shows a high level view on the PII architecture and roles.

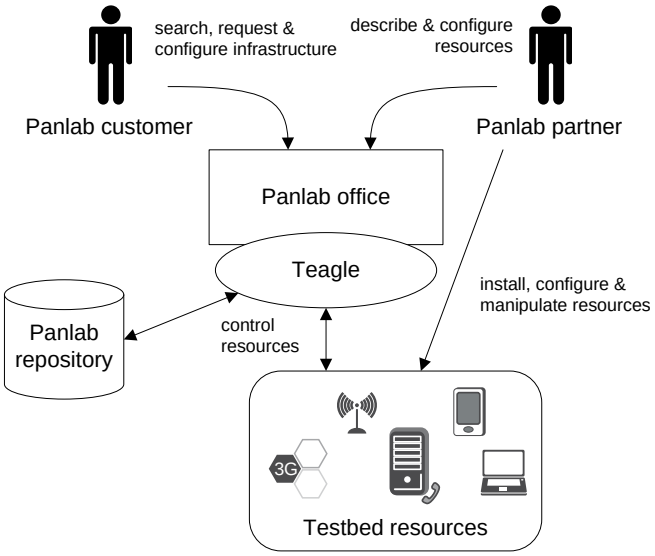


Figure 3.16 – The Panlab roles include a Panlab office, a Panlab partner, and a Panlab customer role. The Teagle framework provides a set of middleware components to control the distributed testbed resources.

The Panlab partner "[...] is a provider of infrastructure resources/testbed components (e.g., hardware, software, virtualized resources) necessary to support the testing services requested by the customer. Partners interact with the Panlab office to offer requested testbed resources to customers. Collectively, the Panlab partners represent the Panlab federation." ([1], p. 168)

The Panlab customer "[...] has access to specific infrastructure and functionality necessary to perform testing and experimentation according to its needs. Customers are typically

[33] http://nitlab.inf.uth.gr/NITlab/index.php/testbed
[34] http://www.netdimes.org/new/
[35] http://www.etomic.org/
[36] http://www.panlab.net/objectives/panlab-ssa.html

interested in carrying out R&D activities using resources provided by Panlab partners. They are supported by the Panlab office in order to implement and evaluate new technologies, products, or services drawing upon the large resource pool available through the entire Panlab federation." ([1], p. 168)

The Panlab office "[...] realizes a brokering service, serving Panlab partners and customers by coordinating legal and operational processes, the provisioning of the infrastructures and services to be used for testing and experimentation, and the interconnectivity of the various partner test sites and customers." ([1], p. 168)

An important architectural concept is the Panlab *Virtual Customer Testbed (VCT)*. A VCT is a collection of resources configured for a specific Panlab customer. The resources may be distributed across several Panlab partners. The so-called *Teagle* framework provides tools and mechanisms to define VCTs and provision them. At the time of writing, Teagle implements:

- a model-based repository,
- a creation environment (the VCT tool),
- an orchestration engine,
- a web portal,
- a policy engine, and
- a gateway.

"The repository holds data about available resource types and instances, and can be queried by other Teagle components such as the VCT tool. The Panlab customer can launch the VCT tool [...] from the Teagle portal and use it to define a VCT. From a list of available resources, selected elements can be dropped, connected, and configured on the tool workbench. During the VCT design, the tool interacts with the policy engine to indicate impossible or forbidden testbed layouts or configurations. When the VCT design has been finished, the tool stores the VCT definition in the repository, and the booking/scheduling procedure can be initiated via the VCT tool or the Teagle portal." ([1], p. 170)

During the booking procedure, Teagle retrieves the VCT specification from the repository, triggers policy evaluation, and sends the full specification to the orchestration engine. Once the orchestration engine has received a valid VCT specification, it starts to assemble a provisioning workflow resolving dependencies and sorting provisioning steps. Upon execution of the workflow, provisioning requests are sent to the Panlab partner domains in order to provision the requested resources as demanded by the user's VCT specification. As a lot of the Panlab architecture and prototypes rely on concepts defined by this thesis, more details on specific Teagle functions are explained in the chapters 4, 5, and 6. Also, a number of publications give additional insight such as [100], [101], and [88]. In addition to the technical architecture and prototypes that enable the Panlab testbed federation, PII delivered[37] a number of operational [102] and legal [103] requirements and process specifications that enable the operation of the Panlab office.

[37] http://www.panlab.net/publications/pii-deliverables.html

3.8.3 Wisebed and the Open Federation Alliance

The European project *Wisebed* [104] started in 2008 and ended in January 2011. The project established a federated environment for wireless sensor networks (WSNs) with more than 2000 nodes. Furthermore, the project initiated the *Open Federation Alliance (OFA)*[38]. However, the OFA consortium's standard drafts are not public at the time of writing and the OFA impact is unclear. Figure 3.17 ([105], p. 4) shows the Wisebed federation system entities and their interactions.

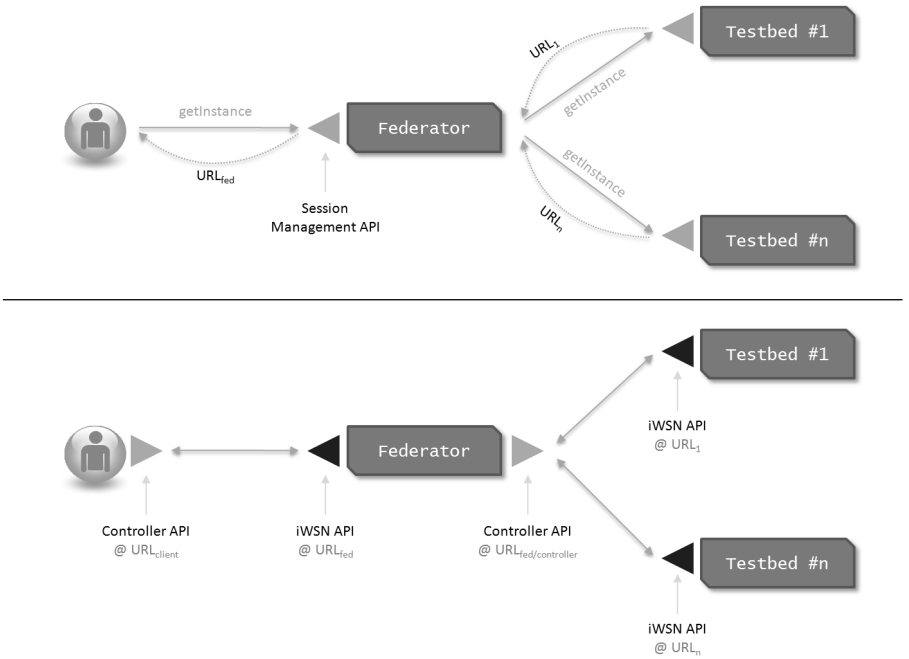


Figure 3.17 – Overview of the Wisebed client, federator, and testbed backend interaction

The testbeds implement Web services for exposed capabilities. The *federator* component is "[...] a system that federates two or more testbeds. It provides both testbed services and a controller API. It connects to two or more testbeds via their testbed services, and exposes a unified view of the resulting federated testbed via its testbed services." ([106], p. 8) A controller can use the federated services to control an experiment run on the Wisebed platform. The *iWSN API* [106] also supports communication between testbed services, directly forming a virtual link (not shown in figure 3.17).

One of the main advantages of the Wisebed implementation is that it provides considerable abstraction capabilities. Clients can connect to the federated testbed abstracting from the fact that the service is actually provided by a distributed facility. This allows to build large federations flexibly and according to the experimenters needs. In addition, Wisebed provides several specifications and implementations such as a wireless node API[39], the iWSN federator [105], a wireless testbed management architecture called *TARWIS*[40], a reservation system[41],

[38] http://wisebed.eu/ofa
[39] http://www.wisebed.eu/images/stories/deliverables/TR/TR-2010-CTI-TUBS-UZL-NodeApi-V0.9.pdf
[40] http://www.wisebed.eu/images/stories/deliverables/TR/TR-2010-TARWIS-API-2.X.pdf
[41] http://www.wisebed.eu/images/stories/deliverables/TR/TR-RS-API-V1.pdf

an authentication and authorization framework building upon *Shibboleth*[42], as well as a resource description format called *WiseML*[43].

3.9 G-Lab

The German-Lab (G-Lab) initiative [107] is funded by the German Federal Ministry of Education and Research (BMBF) and includes a total number of 10 German national projects. G-Lab started in 2008 with a first phase project that provides the G-Lab infrastructure and is composed of seven working groups: FI architectures, routing and address schemes, wireless networks and mobility, monitoring and management concepts, QoS and security, SOA and service composition, and experimental facility. With the G-Lab second phase, nine additional projects joined the G-Lab initiative.

G-Lab aims at driving FI research in Germany in terms of network and architecture research studies (e.g. [108], [109]) supported by an experimental facility. The facility is shaped according to the users needs. This shall ensure that the facility fits the experimenters demands and does not develop independently from the actual user community. A strong correlation is foreseen between the research studies and the facility as shown by figure 3.18 ([110], p. 2). The idea behind this approach is that the facility might evolve into something like the Future Internet itself.

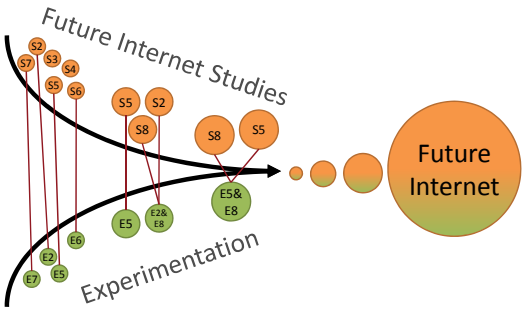


Figure 3.18 – The G-Lab vision: Future Internet research studies are accompanied by experimentation performed on the G-Lab facility. Through the tight coupling of research and facility operation, the platform might evolve into the Future Internet itself.

From a technical infrastructure perspective, G-Lab is an *extended private Planetlab* that relies on the PLC software and can officially only be used by G-Lab project participants. The facility hardware is shown in table 3.1 (adapted from [110], p. 3).

Table 3.1 – Node types and node hardware

Node Type	CPU	RAM	Disk	Network
Head node	2x Xeon Quad E5450 3.0 GHz	16 GB	16x 146 GB SAS	4x 1 GBit
Network node	2x Xeon Quad L5420 2.5 GHz	16 GB	4x 146 GB SAS	8x 1 GBit
Normal node	2x Xeon Quad L5420 2.5 GHz	16 GB	4x 146 GB SAS	4x 1 GBit

[42] http://www.wisebed.eu/images/stories/deliverables/TR/TR-AA-RS-API-AA-RS.pdf
[43] http://dutigw.st.ewi.tudelft.nl/wiseml/wiseml_schema.pdf

The infrastructure is distributed across six universities in Germany: University of Kaiserslautern, University of Würzburg, Karlsruhe Institute of Technology, University of Munich, University of Darmstadt, and Universitiy of Berlin. The node distribution is shown in table 3.2 ([110], p. 3). The sites are connected by public Internet and are able to deploy *default and custom boot images*. The selection of a specific boot image depends on the type of experiment to be executed.

Table 3.2 – G-Lab sites and node count

Site	Head nodes	Network nodes	Normal nodes
University of Kaiserslautern	1	2	56
University of Würzburg	1	2	22
Karlsruhe Institute of Technology	1	2	22
University of Munich	1	2	22
University of Darmstadt	1	2	22
Universitiy of Berlin	1	2	12

Also, federation with other PlanetLab based facilities as well as non-PlanetLab testbeds is envisaged but not yet officially established. The BMBF project G-Lab Deep[44] is looking into federation aspects of the G-Lab platform from a conceptual perspective. Furthermore, a topology management tool called *ToMaTo*[45] [111] has been developed at the University of Kaiserslautern that allows to create and manage virtual network topologies and that is available to G-Lab users. Using virtualization technologies, experimenters can execute isolated experiments using customized subsets of the available G-Lab facility resources. The topologies can be composed of *devices* (OpenVZ or KVM based nodes) that run the experimental software as well as *network components* that interconnect devices according to the topology layout specified by the user. For interconnecting distributed nodes, the public Internet is used, however, private overlays can be specified to emulate network characteristics such as packet loss, delay, and bandwidth limitations.

3.10 Summary

This section covered existing concepts and initiatives relevant in the context of this thesis. As described in the upcoming chapters, the design of the artifacts follows general concepts in the field of information systems such as the concept of service orientation and the collective resource sharing paradigm. Also, it has to be noted that some of the presented approaches such as most of the FIRE and GENI project results have emerged in the course of the past 4 years in parallel to the work described by this thesis. Therefore, some overlap in functionality is expected. This essentially shows that others are moving into a similar direction, proving the work to be relevant for the target community. Section 7.3 and in particular table 7.5 compare this thesis with other federation approaches in detail. Some of the most important unique characteristics of this thesis include: model-based federated resource description of

[44] G-lab Deep project website: http://www.g-lab-deep.de
[45] Available under GNU Affero GPL version 3 at https://github.com/dswd/ToMaTo/wiki

heterogeneous resources, support of higher tier federation logic (e.g. abstract cross-domain resource relationships and orchestration), an advanced model-based resource request and configuration design tool, support for automated dynamic inter- and intra-domain resource deployment decisions, as well as cross-layer and cross-domain resource dependency resolution and automatic distributed resource configuration.

4 Resource Federation Model

This chapter introduces one of the artifacts delivered by this thesis: *the federation model*. It provides the basis for the upcoming chapters as it defines the constructs (i.e. the model entities) used by the federation methods introduced in chapter 5 and the federation framework instantiation described in chapter 6. First, the model entities are introduced that are then used by the following sections to model and discuss different federation scenarios.

4.1 Model Entities

Models are important in architecture and system design as they allow to conceptually approach a problem and define the constructs and terms needed for a systematic problem solution. Hevner et al. define a model as follows:

> *Models use constructs to represent a real world situation—the design problem and its solution space (Simon 1996)*[1]*. Models aid problem and solution understanding and frequently represent the connection between problem and solution components enabling exploration of the effects of design decisions and changes in the real world.* [11], p. 78-79

In section 1.3 this thesis' research hypothesis was given. It is repeated here for the convenience of the reader:

1. To *effectively* and *efficiently federate heterogeneous resources* across the boundaries of administrative domains, enabling *flexible cross-domain resource collaboration*, ...
2. ... a *federation model* can be designed that allows to define a *generic resource federation methodology* and instantiate an according *system solution*, ...
3. ... building upon a *federated resource control framework* and using *resource description, abstraction, orchestration*, and *provisioning techniques*.

This section addresses one of the core parts of the hypothesis: *the federation model*. It introduces the model entities and high level interactions as shown in figure 4.1 (based on [112],

[1] Simon, H. A.: The Sciences of the Artificial (3rd ed.), MIT Press, Cambridge, MA, 1996.

p. 52). A more detailed description of the entity interactions—representing the federation processes (i.e. the methods)—is given in chapter 5.

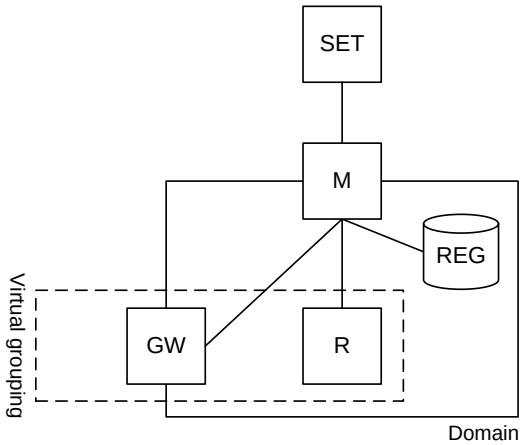


Figure 4.1 – Overview of the federation model entities: SET, M, REG, R, GW. The figure also shows the abstract concepts of a domain, and a virtual resource grouping that may span the border of several domains.

The model defines five conceptual entities (*SET*, *M*, *REG*, *R*, *GW*) and two abstract concepts (*domain* and *virtual resource grouping*). A line between two entities represents bi-directional communication between those entities. The model has been proposed to serve a FIRE federation framework [112]. It's entities are described below.

The M entity represents a *domain manager* function. Typically, this is a software component that exposes domain capabilities via a well defined *generic interface* acting as a *resource abstraction* layer. The manager interface defines generic resource control commands that can be used by higher layer entities (see SET entity) to interact with resources. Such generic commands are for example the create, read, update, delete (CRUD) commands (see section 5.2) marking the communication enabling service primitives. If the semantics of such service primitives are defined and consistently implemented across all participating domains, higher tier logic can build upon this resource abstraction and control layer.

The domain managers forward requests received on their domain-external interface to resources and registries after applying general and domain specific security measures (e.g. domain specific resource access and usage policies).

The R entity represents a resource. A resource is part of a domain and can be controlled remotely via the domain manager (the M entity). Resources exist as *types* and *instances*. One can think of the relationship between types and instances as in object oriented programming where an object is an instance of a specific class. The type defines resource attributes that are common across all instances of a specific type. Everything that does not classify as a *type*, is an *instance*.

Resources are described according to specific models or languages that are optimally understood across all participating domains. The use of domain specific languages and models might require a mapping to an overarching information model that represents the

characteristics of the entire system. If such mappings cannot be constructed with reasonable efforts, this might prevent higher tier federation logic (e.g. resource orchestration and automatic resource provisioning) relying on common resource descriptions.

Examples of resources that can be provided by the federated domains are physical and virtual machines, data, software, arbitrary devices, as well as abstract concepts such as user accounts. Here, the difference between resource types and instances can be exemplified. A resource type *virtual machine* can be offered by several domains. However, one concrete virtual machine that is deployed in a specific domain would be a resource *instance*. Resources are registered either manually or automatically in *registries* (see REG entity).

The REG entity represents a domain registry that holds records for all resources of a domain. Optimally, the data hold by the registries are structured according to a common federation information model. Registries are accessed by domain managers that may choose to expose domain specific resource data on their public interface after applying the necessary security measures, i.e. authentication and authorization. The data hold by registries might have been entered manually or might have been aggregated automatically depending on the registry implementation.

It has to be noted that the registry is basically a *concept*. How domains choose to implement this is irrelevant from the federation-level perspective as long as certain data formats and semantics have been agreed. In fact, the federation level is likely to dispose of another registry. Here, even advanced concepts such as syntactical and semantical data mapping might be applied if needed. See also the recursive federation scenario described in section 4.4.3.

The GW entity represents a gateway component. Strictly speaking, it can be seen as just another resource that is controlled by a domain manager. However, as it provides critical resource inter-connection services, it has been made a separate model entity.

It has to be noted that for some federation scenarios (e.g. the recursive scenario, see section 4.4.3), treating the gateway as a simple resource can considerably simplify the scenario construction and visualization. Therefore, the GW entity is not always specifically outlined or visualized in upcoming descriptions and figures. In such cases it can be assumed to be represented by the *R entity*.

As shown in figure 4.2, via the gateways, inter-domain connections can be established and secured. If the gateways should be considered to be part of a virtual resource grouping or not, depends on the system view and the intended level of resource configuration granularity (see [1] and section 5.5).

While from a domain perspective the gateways play a crucial role in terms of network topology and security, from a higher layer federation perspective, the gateways might be considered transparent. Therefore, a resource request might simply ask for the resources R_A and R_B (see figure 4.2) to be interconnected, abstracting from lower level details and leaving it to the platform to choose the proper interconnection settings. Which level of configuration granularity to expose to upper federation layers (and ultimately the users) depends on the different stakeholders preferences and operational principles.

The SET entity communicates with one or more domains via the domain-external interface

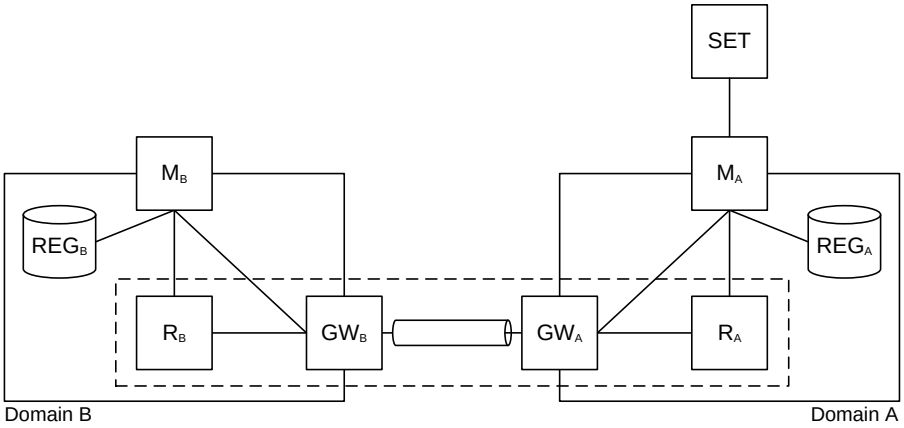


Figure 4.2 – Gateways in domain A and domain B provide inter-domain resource connectivity.

of the domain managers. This allows the *SET* entity to control distributed resources across the federation. SET entities usually provide the entry point for the federation users and can result in rather complex frameworks themselves. For example, they might rely on their own registries that can be populated with information requested from various domains. Also, they might implement higher tier federation logic such as federation policies and federated resource orchestration mechanisms. Furthermore, they might provide a wide variety of supporting tools such as graphical creation environments, resource monitoring tools, and experiment execution tools.

In a recursive scenario (see section 4.4.3) the SET entity might play the role of a domain manager (M entity) on the next upper federation level. This depends on the M/SET entity software component and API design and implementation. For a visualization of this aspect see also figure 4.10.

In addition to the basic model entities described above, the federation model defines two additional abstract concepts:

A domain is represented in figure 4.1 and figure 4.2 by the large solid lined rectangle. It marks an administrative domain boundary. Usually, all resources provided by a domain are governed by the same organization. In case of a recursive federation scenario, a domain might represent a federation of domains. Therefore, in such case, the previous statement is somewhat blurred as resources provided by a federation are usually owned by several organizations and unless they are collectively governed by a single *federation organization*, the initial statement does not hold true. For a discussion of resource governance in different federation scenarios, see section 4.4.

Resources within a domain can be controlled via the according *M entity* (i.e. the domain manager). In figure 4.2, the resources R_A and R_B are part of different administrative domains and can be controlled via their respective domains mangers M_A and M_B. Principally, a single organization can operate several domains. However, in most cases, each organization operates a single domain that conceptually groups all resources provided by that organization. Therefore, although not mandatory, the domain A and the domain B in figure 4.2 are likely to be operated by different organizations.

A virtual resource grouping is a collection of resources that have been configured to be part of a virtual environment. This is indicated by the dotted rectangles in figure 4.1 and figure 4.2. Virtual resource groupings may span the borders of several domains if resources from different domains are configured to be part of this grouping.

In many cases, virtual resource groupings will rely on public links between the involved domains. In such cases, overlay networks—e.g. using virtual private network (VPN) connections—can help to satisfy security requirements and to separate network traffic (for example separating concurrent experiments that are run on the same physical substrate). However, any interconnection option mutually supported by the *GW entity* provided by each of the involved domains can be configured. For example, a dedicated optical link with certain link properties (e.g. bandwidth) may be switched between R_A and R_R via their respective gateways GW_A and GW_B.

Virtual resource groupings are requested by users of the federation to be accessed by them or to be used for other purposes. For example, the execution of a research experiment might require a virtual resource grouping in a specific setup. Or an interoperability event organizer might contact the federation and book a certain resource grouping for the event participants to work with the resources during the event. There are many other examples and use cases where federation users have a need for a specific infrastructure setup and request an according resource provisioning from the federation.

4.2 Federation Roles

The different federation relationships have already been touched in section 2.1 and have been illustrated in figure 2.1. Figure 4.3 shows the different *federation roles* as defined by the model. Three distinct stakeholder groups can be differentiated: (1) the federation user, (2) a federation organization, and (3) the resource providers.

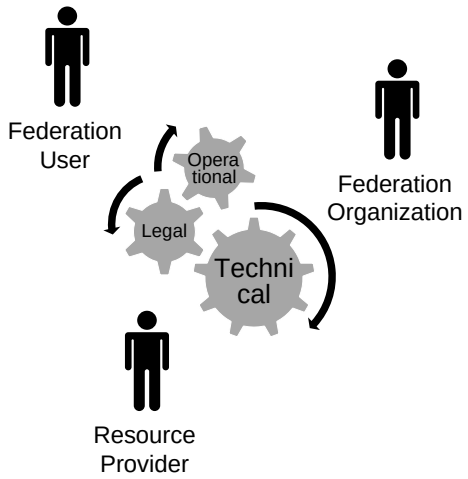


Figure 4.3 – The federation stakeholder roles: a user interested in resource services offered by resource providers interacts with the providers that are collectively represented by a federation organization. All stakeholder groups engage in legal, operational, and technical relationships.

All of the stakeholders engage in legal, operational, and technical relationships. The focus of this work is on the technical mechanisms to enable federation. The roles are introduced by the following listing.

Resource providers supply the resources that are offered across the federation and are usually represented by an organization such as a university, a research institute, or a company. Providers are usually restricted by corporate guidelines impacting the way resources of a specific provider can be accessed and used. In most cases, this results in specific resource usage and access policies that are evaluated and enforced upon resource access requests (see also section 4.3.4 for federated policy aspects).

Resource providers engage in legal and operational agreements with each other, with users, and especially with the federation organization. Such agreements regulate all processes needed to establish the necessary trust and business relationships among the federation partners. The terms and conditions can vary considerably, depending on the chosen federation mode(s) and specific federation scenario (see section 4.3 and section 4.4 respectively).

The Federation organization represents the resource providers and serves their interests (but not exclusively). In many cases the organization is formed by a subset of resource provider organizations (see also section 4.4.4). It provides services to both the resource providers and the users.

Such services include a resource brokering service where resources offered by the individual resource providers are offered to users through the federation organization. Legal and operational agreements define the terms and conditions of such processes. Both the users and the providers benefit from channeling certain tasks through the federation organization. A practical example is the setup of a virtual resource grouping spanning three resource provider domains. The user requesting such setup benefits from negotiating (and maintaining a legal relationship) with a single entity, i.e. the federation organization, instead of three different resource provider organizations. The providers benefits from stable and agreed processes, terms and conditions, and technical requirements that should be designed to make the setup of the requested virtual resource grouping a routine task from a legal, operational, and technical perspective and reduce the overall overhead. In a well designed and implemented federation instantiation, this will result in a *"win-win-win"*[2] situation where all of the three stakeholders mutually benefit from getting engaged with and in the federation.

Furthermore, the federation organization might provide higher tier federation tools that build upon the abstraction layer realized by the technical federation infrastructure. Examples of such tools include virtual resource grouping design environments, monitoring and measurement tools, experiment planing tools, etc. Also, resource lookup and search tools might be provided to ease working with the large pool of resources collectively provided by the resource providers (i.e. *the federation*).

Federation user The user requests and consumes the services provided by the organization and the providers to satisfy a certain infrastructure need. Which type of users are

[2]Definition WIN-WIN: "[...] advantageous or satisfactory to all parties involved <a *win-win* situation> <a *win-win* deal>". Merriam Webster dictionary entry: http://www.merriam-webster.com/dictionary/win-win

mainly targeted depends on the specific federation instantiation. In many cases, users include individuals from industry and academia. The services consumed by the users are compensated. Again, the type of compensation depends on the concrete federation instantiation. Different models known from the economic sciences are possible for the type of collective resource sharing realized by generic resource federations: e.g. federated resources as a *private good* or a *club good*.

If the federation is for example instantiated as a *club*, compensation might be realized in terms of a *fee*. The fee might take different forms, such as a resource contribution (e.g. initially, when joining the club) or a certain amount of money (e.g. a flatrate or resource usage based fee). Also, resource providers might agree to be federation users themselves. Therefore, part of the resource provider *compensation* might actually be the *consume* of federated resources.

We have seen that many aspects of how the stakeholders engage, depend on the federation modes and scenarios. Choosing among the options created through the different modes and scenarios will result in a concrete system instantiation. The next section introduces the different federation modes.

4.3 Federation Modes

From a technical point of view, in order to enable effective federation, four distinct areas need to be addressed:

1. identity management,
2. control framework design and interfaces,
3. resource description and discovery, and
4. policy description and enforcement.

The extend of how much those aspects are aligned across the different federation stakeholders and especially across the different resource providers, determines the different federation scenarios resulting in different system instantiations. The different aspects of the four important areas enumerated above are discussed in detail in the following subsections.

4.3.1 Identity Federation

As individuals and federation tools on behalf of such, shall be enabled to access and work with distributed resources, resource providers need to be able to authenticate an entity requesting to use and/or to control resources. Figure 4.4 demonstrates the identity federation mechanism details.

Generally, identity federation requires a trust relationship between the federated domains or at least the possibility to establish such a relationship, for example via a trust broker. To ease the scenario, we can assume that domain A and domain B shown in figure 4.4 have previously established a trust relationship. Specifically, the domain A trusts the domain B and accepts identities issued by domain B. Technically, this usually involves trust anchors (e.g. using common trusted root certificates). For example, as shown in figure 4.4, domain A can verify that a certain identity has been issued by domain B, represented by trust anchor_{B}.

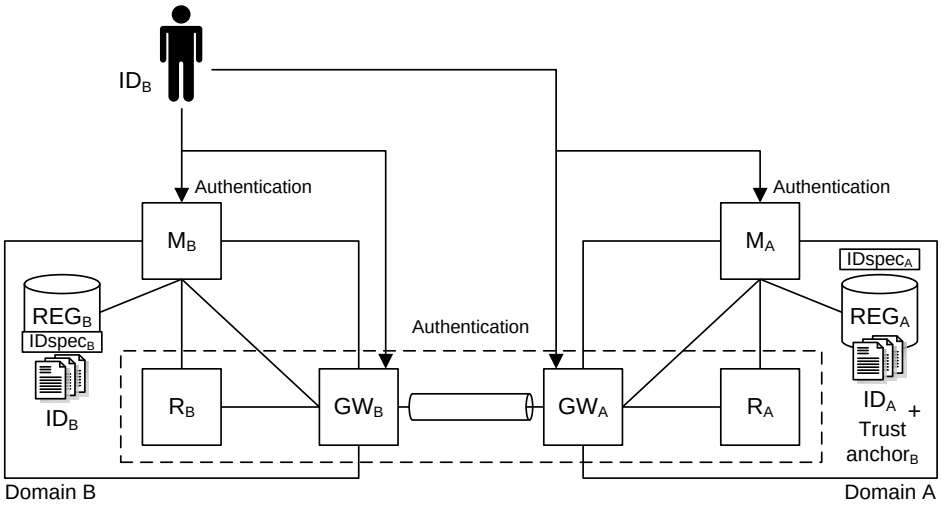


Figure 4.4 – Identity Federation

Authentication takes places at domain managers and gateways where the identity of the requester is checked. Authorization may be enabled using policy mechanisms (see subsection 4.3.4). Domain B holds data records for all identities issued at domain B. Such information might be stored "domain-centrally" in the domain registry. Upon requesting resource access either at the domain manager or the gateway, those components can compare the identity information presented by the requester with the domain records and take authentication decisions.

The format of how identity information is conveyed is defined by an $IDspec$. This means that the security handles (i.e. credentials) that are for example issued by domain B in figure 4.4 follow a format defined by $IDspec_B$. The more heterogenous $IDspec_A$ and $IDspec_B$ are, the more difficult it will be for domain A and domain B to federate on the identity layer. As we have seen in the state of the art section, many implementations of federated ID systems make use of cryptography and public/private key mechanisms. This allows to use the public key of the trust anchor to verify the validity of digitally signed objects. A tangible example for an $IDspec$ would be the format of X.509 digital certificates[3] and their cross-domain use in terms of a Public Key Infrastructure (PKI).

4.3.2 Control Framework Federation

Federating on control framework layer requires either the harmonization of the control framework interface semantics, message formats, and protocols or a mapping between those. The problem of the latter option is the poor scalability.

The control framework interfaces (i/f) are shown in figure 4.5 on the north side of the domain managers: i/f M_A and i/f M_B. Harmonizing the interfaces allows to reduce the overhead needed to generate and communicate the configuration of a virtual resource grouping VG to the federated domains. A VG specifies a virtual resource grouping while a $Vspec$ defines the format of its description. VGs are structured according to the format defined by a

[3]RFC 5280: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, [113]

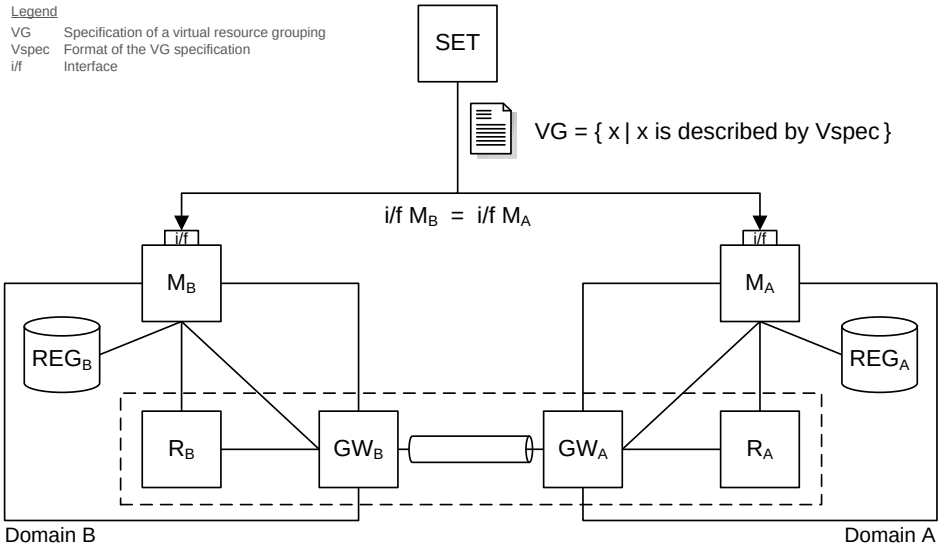


Figure 4.5 – Control framework federation: harmonized domain manager interfaces

Vspec including the resources that are part of the grouping and their configuration and may contain additional parameters (e.g. a virtual resource grouping lifetime).

For the formal definition of a VG, see figure 4.6 and the next subsection.

4.3.3 Federated Resource Description

This subsection explains how resources can be described within and across domains in terms of the federation model. Figure 4.6 illustrates the federated resource description mechansims.

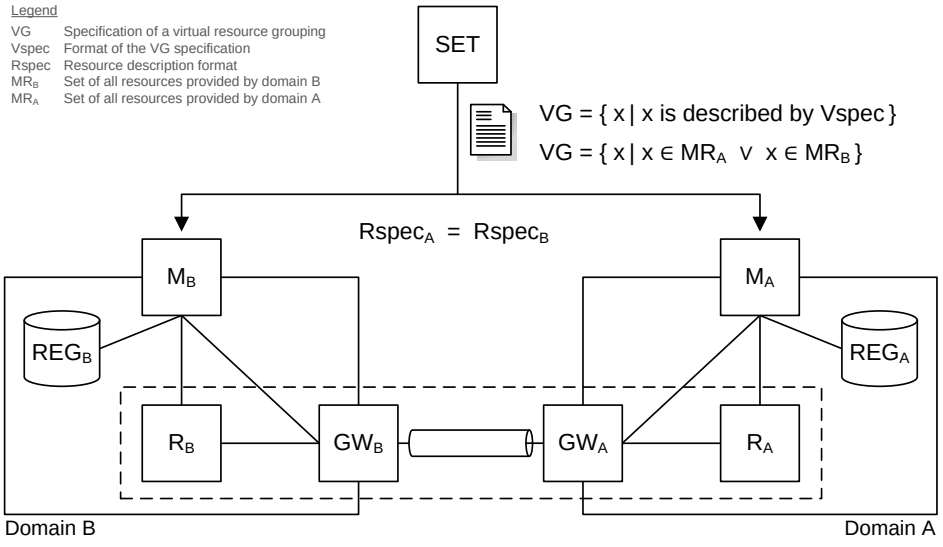


Figure 4.6 – Harmonized resource description

All resources provided by a domain are formally described following a specified *resource description*. The format of such formal description shall be denoted by $Rspec_x$. For example,

the resources provided by domain A are described following the format defined by $Rspec_A$.

This allows us to characterize all resources provided by a specific domain in terms of their description. Therefore, we can define the set of resources provided by domain A and domain B as:

$$MR_A = \{x \mid x \text{ is described by } Rspec_A \text{ and is provided by domain A}\} \tag{4.1}$$

$$MR_B = \{x \mid x \text{ is described by } Rspec_B \text{ and is provided by domain B}\} \tag{4.2}$$

In case of harmonized resource descriptions across the participating domains, the format of the description used in different domains is the same or a mapping between the different descriptions can be derived. However, as with the control interfaces, the mapping approach provides limitations in terms of scalability. Harmonized resource descriptions across domain A and B require:

$$Rspec_A = Rspec_B \tag{4.3}$$

Furthermore, a virtual resource grouping and the specification of such can be defined making use of the definitions outlined above. If a virtual resource grouping is denoted by VG we can say that:

$$VG = \{x \mid x \in MR_A \vee x \in MR_B\} \tag{4.4}$$

where:

$$VG = \{x \mid x \text{ is described by } Vspec\} \tag{4.5}$$

and a $Vspec$ is the format of the specification of a virtual resource grouping.

As stated before, resources usually exist as types and instances. In case of *instances*, resources can never be the same:

$$R_{Ai} \neq R_{Bi} \tag{4.6}$$

However, specific resource *types* can be used in several domains, so that:

$$R_{At} = R_{Bt} \tag{4.7}$$

In this case and given that resource descriptions are harmonized across the domains A and B, the following applies:

$$Rspec_A(R_{At}) = Rspec_B(R_{Bt}) \qquad Rspec_A(R_{At}) = Rspec_A(R_{Bt}) \tag{4.8}$$

$$Rspec_B(R_{At}) = Rspec_B(R_{Bt}) \qquad Rspec_B(R_{At}) = Rspec_A(R_{Bt}) \tag{4.9}$$

To elaborate in more detail on virtual resource groupings and specifically on formula 4.4, it can be said that *only* in case of *types*, the following holds true:

$$VG = \{x \mid x \in MR_A \vee x \in MR_B\} \tag{4.10}$$

while for resource instances it must be:

$$VG = \{x \mid x \in MR_A \dot{\vee} x \in MR_B\} \tag{4.11}$$

This means that no resource instance that is part of MR_A and that is at the same time part of MR_B can be part of a VG. This is because such resource instances do not exist in terms of the model.

4.3.4 Federated Policy Description

Although the harmonization of resource usage and access policies is not a mandatory requirement in order to enable federation, it provides a number of benefits to the federation users.

After successful authentication of the requester (see figure 4.4), the system needs to decide whether or not to allow the request. This authorization is based on policies that are defined locally by the domain authority or globally for the entire federation or both.

Figure 4.7 shows the policy enforcement points of domain A and domain B. Both domains store a number of domain-specific policies in their respective registries. Decisions can be

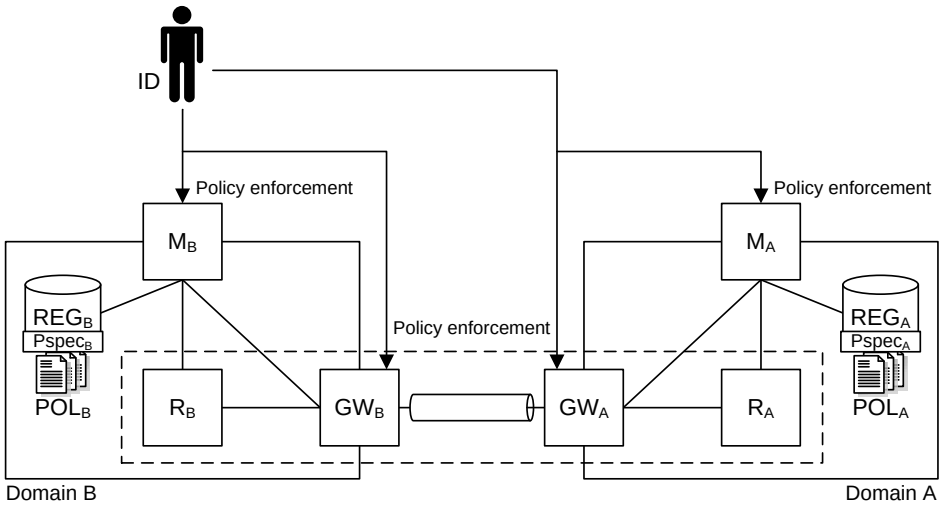


Figure 4.7 – Policy federation

based on the requester's identity, the issuing authority, or general attributes like time and date of the request, lifetime of the requested resources, intended resource usage, etc. In a federated system in which all domains define their policies independently, resource providers have maximum flexibility and can enforce a high level of security tailored towards their specific requirements (e.g. corporate guidelines, etc.). However, this approach will result in a best effort system, where the deployment of complex virtual resource groupings with extensive intended resource usage is likely to fail due to numerous policy evaluations and the high probability of request-denying policy enforcements at different sites.

Therefore, global federation policies are preferable from a federation user perspective enabling extensive and flexible resource configuration possibilities across the entire federated system. However, not only the policies themselves but also the format of how to describe such policies play a crucial role in large federations. The format of such policy description is defined as *Pspec*.

The harmonization of *Pspecs* is difficult as policies often rely on the *IDspec* and the *Rspec* to express *who* is allowed to do *what*. Consequently, if the Rspecs and IDspecs are not harmonized across the federation, the definition of a Pspec across multiple federated domains is a challenging task. In addition to the technical aspects, the federation-wide alignment of Pspecs involves legal and operational aspects that tend to be affected by corporate guidelines. Most of today's resource federations are operated without a global alignment of policies or the format of their description.

4.4 Federation Scenarios

Having discussed the different federation modes (identity, control framework, resource description, and policies), this section will introduce several federation scenarios. The scenarios are essentially different manifestations of federation variables defined by the federation model and the different federation modes. This demonstrates the flexibility and generic nature of the federation model allowing to model different federation contexts.

4.4.1 Central Operation

One of the characteristics of the central scenario is that the federated platform is operated by a single organization. Resource providers have bilateral agreements with this central federation organization. In such scenario there is a rather tight integration of the resource providers which allows to offer very stable services based on federated resource capabilities. As opposed to a more loose coupling of services, it also allows to standardize most of the fields discussed in section 4.3, namely federated identity management, a federated control framework, as well as a common description for resources across the participating domains. Also, most policies (at least on a federation level) tend to be harmonized in central federation scenarios. An example for such a central federation operation is the Panlab federation (see sections 3.8.2 and 7.1.1).

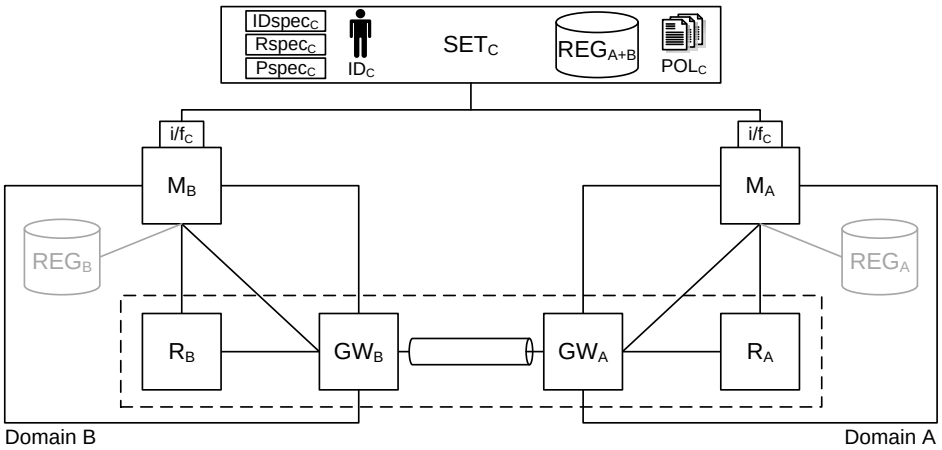


Figure 4.8 – Operation by a central organization: authentication and authorization is aligned. Also, the federation relies on a common control framework and resource model. A central registry simplifies resource discovery and the support of advanced tools.

Therefore, although the different items are not mandatory, the central scenario can be characterized as follows:

1. Trust relationships between resource providers and the federation organization exist. Resource providers trust and accept the identities issued by the central organization. Such identities are represented by digital objects compliant to an IDspec$_C$.

2. Higher tier federation logic and tools can rely on a common control framework: i/f M_A = i/f M_B = i/f$_C$ (see figure 4.8).

3. Resource descriptions are harmonized across most providers: $Rspec_A = Rspec_B = Rspec_C$.

   In addition, central registries might be supplied as shown in figure 4.8. The domains are likely to maintain their own registries. However, the data hold in the central registry can be considered to be a superset of the local records.

4. Resource usage policies and authorization decisions tend to be harmonized across the participating resource providers, based on the trust relationships and common formats for resource descriptions and identities/credentials.

The strength of the central operation is that on top of the harmonized identity, resource, and policy handling, advanced tools and services can be build that provide a wide range of features and functionality (e.g. easy resource discovery, lookup, scheduling, etc.) On the other hand, for very large federations it is unlikely to achieve a great level of harmonization due to the extensive standardization efforts needed and the time required to agree on all sort of details. Also, the operation of central services is risky from a service availability point of view as specific components can provide bottlenecks and potential targets for strategic cyber attacks.

4.4.2 Distributed Operation

The distributed scenario is characterized by the fact that the different domains and the capabilities/services provided by them are loosely coupled. An organization steering the federation might exist but has less influence on the other stakeholders. The level of integration between the resource provider organizations is lower compared with the central operation. It can be measured in terms of how many areas (as introduced by section 4.3) and to which extend within each area, the processes and specifications (IDspec, Rspec, Pspec, and resource control) are aligned.

Figure 4.9 visualizes the distributed scenario and potential mappings that might be required by the SET_C entity.

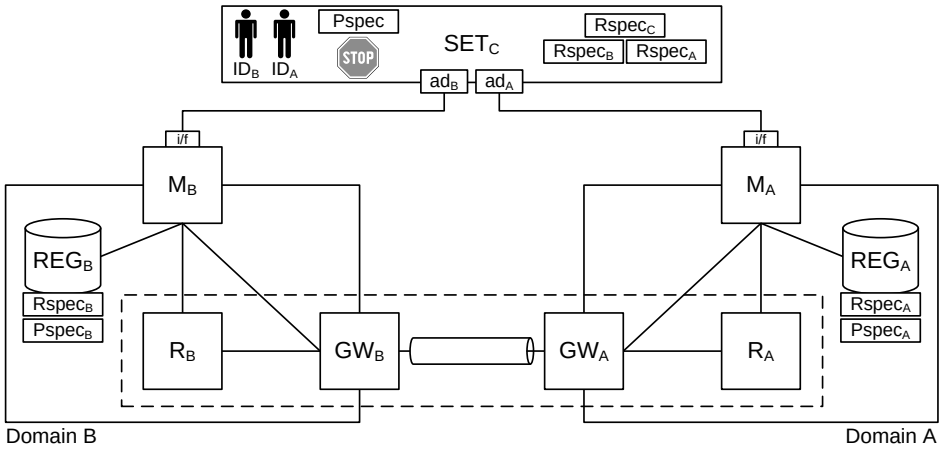


Figure 4.9 – In distributed scenarios, less federation relevant aspects are harmonized across the federation stakeholders compared with the central operation.

Regarding resource descriptions, for most distributed scenarios, it can be assumed that:

$$Rspec_A \neq Rspec_B \tag{4.12}$$

This means that any higher tier federation logic and tools, represented by SET_C cannot rely on a common understanding about resources that are offered by the different domains. To circumvent this issue, a common (meta-)model might be constructible that maps the entities and constructs used by the domain specific resource description languages and models to a set of common entities in order to derive a certain level of harmonization, even if this can only be achieved on a higher abstraction level.

Regarding the control framework, the federation might face the following situation:

$$\text{i/f } M_A \neq \text{i/f } M_B \tag{4.13}$$

Again, a mapping is most probably required in such situations in order to effectively federate. Such mappings can be realized by adapter component implementations that translate the service primitives and their semantics between different control frameworks. This is indicated in figure 4.9 by the two boxes ad_A and ad_B.

Regarding identity and policy handling, the users of tools provided by SET_C will need to be prepare to use different identities and the according credentials when accessing the different domains. Also policy evaluation and enforcement can be very heterogeneous.

In how far the outlined issues regarding a distributed system instantiation and operation prevent the overall federation to provide usable and useful services, needs to be decided in respect of the intended use cases. In most cases there will be a trade off between harmonization (central scenario) and heterogeneity (distributed scenario). While harmonizing all the different aspects of federation requires extensive and lengthy standardization efforts as well as considerable commitment from the resource providers, the distributed approach can be realized much faster. In case a tighter integration is needed at a later stage, adapters and mappings can be realized upon demand. However, in some cases this might not be possible because the different system instantiations are simply too heterogeneous. Especially in large federations this results in big efforts in terms of adapter/mapping implementation. For most system instantiations it is critical to find a balance between a totally centralized (harmonized) approach and a totally distributed (heterogeneous) one.

4.4.3 Recursive Scenario

Recursion is defined as "[...] the determination of a succession of elements (as numbers or functions) by operation on one or more preceding elements according to a rule or formula involving a finite number of steps [...]" [114].

This concept known from mathematics or functional programming is applied here to construct a scenario for very large federations that can grow both horizontally and vertically. The horizontal extent is measured in terms of the number of domains on the same federation level (the federation level is denoted by n, see figure 4.10). The vertical extend is measured in terms of the number of federation levels.

As shown in figure 4.10, the SET entity on federation level n conceptually acts as M entity on federation level $n + 1$.

$$SET_{C_n} = M_{A_{n+1}} \tag{4.14}$$

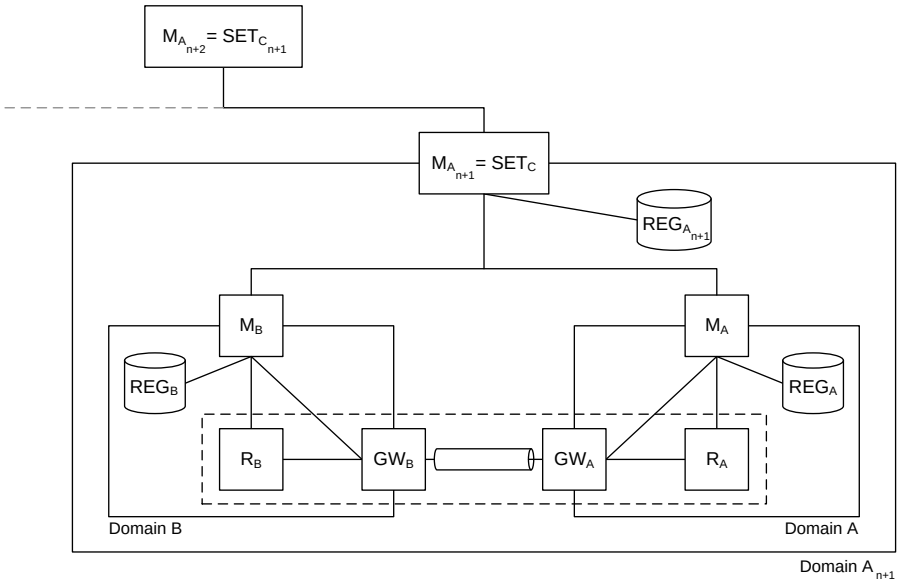


Figure 4.10 – The recursive scenario with multiple federation levels n. In case of $n = 1$, the subscripted index has been omitted.

The resources controlled on level $n+1$ are then actually *domains* that can be controlled by their respective M entities on level n. It has to be noted that in case of $n = 1$, the subscripted index has been omitted on all entity labels in figure 4.10 for the sake of readability.

Registries on a higher federation level may have records for resources of the next lower federation layer. In order to communicate with or obtain information on specific resources that are located several federation levels below the current level, lower level entities might be used as brokers acting on behalf of a requesting higher layer entity. This can result in a chain of requests, in order to arrive at the intended resource on the intended level.

Such mechanisms require a certain level of harmonization in the fields of identity management and resource description. At minimum, a federation-wide naming system, for example using domain prefixes (like DNS) is required to allow communication across the levels of such a hierarchic system. Also, a federation-wide (hierarchic) identity management system like a PKI will greatly simplify the authentication and authorization processes from an implementation perspective.

Regarding policy design and enforcement, as recursive scenarios tend to become rather complex, resource providers will be very careful with respect to authorization and are likely to choose rather restrictive policies.

4.4.4 Consortium Operation

This scenario can be located somewhere in between the central and the distributed operation. The main characteristic is that the federation is jointly operated by a *consortium*. The consortium members are mostly the resource providers represented by their respective organizations that engage in multilateral relationships.

Such federations tend to be smaller compared to the other scenarios. The providers have greater influence on the multilateral agreements and the enforcement of their interests.

Technically, any level of harmonization in the four critical fields (see the federation modes in section 4.3) is possible. How much process and technology alignment will, in the end, exist across the resource providers, depends on the use cases and requirements for a specific system instantiation. On the other hand it depends on what is actually agreeable within the consortium, i.e. how homogeneous the interests of the participating organizations are.

4.5 Summary

This chapter provided a description of a federation model that defines several model entities and different stakeholder types that interact in a federated system. Also, a number of federation modes and scenarios were described. Through the model, the upcoming chapters can rely on a common terminology with respect to the federation framework functions. This will aid the solution understanding and allows to structure and streamline the solution characterization throughout the upcoming chapters. The next chapter will describe a second design artifact—a set of federation methods—targeting resource description, abstraction, orchestration, and interconnection.

5 Resource Federation Methodology

BUILDING large resource federations, results in complex systems involving many system-internal component interactions, as well as interactions across system boundaries. In addition, different stakeholders are involved introducing multi-lateral organizational relationships and various interactions between individuals and IT systems. This chapter introduces the next artifact delivered by this thesis to address those aspects: a set of *federation methods*. Methods are among the four artifact types characterized by Hevner et al. as introduced on page 7 and are defined as:

> *Methods define processes. They provide guidance on how to solve problems, that is, how to search the solution space. These can range from formal, mathematical algorithms that explicitly define the search process to informal, textual descriptions of "best practice" approaches, or some combination.* [11], p. 79

The upcoming sections cover the federation methodology and provide descriptions of five key areas: (1) information model & resource description, (2) resource abstraction & control framework, (3) resource relationships, (4) resource orchestration & provisioning, and (5) cross-domain resource interconnection. Therefore, this chapter targets the second and third part of the research hypothesis postulated in section 1.3.

5.1 Information Model & Resource Description

Resource federations are characterized by the large number of distributed system component interactions and relationships. To efficiently and effectively process and store data across distributed entities in such complex and multi-faceted systems, is it crucial to structure the data consistently allowing for machine interpretation and coherent data handling. Information models allow to specify the data semantics for conceptual elements (entities, relationships, operations, etc.) of a specific field structuring information of a specific knowledge domain. This allows different system components to rely on a common understanding regarding the subject of discourse.

Figure 5.1 demonstrates this for the network domain A. The information model is used to structure the data held by the domain registry. Any other system component that uses

the same information model can interpret such structured data based on the data semantics specified by the model.

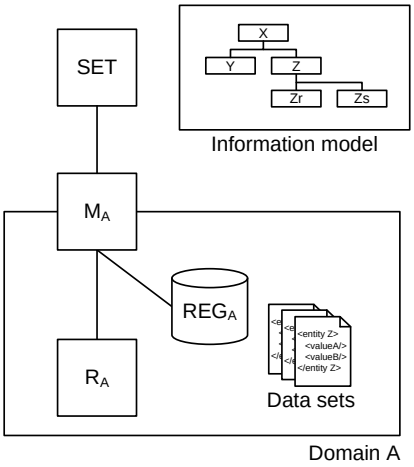


Figure 5.1 – An information model allows to structure data across the federation

The problem with common information models is that they are hard to establish and to agree upon in large federations including many different resource providers. Some of the successful resource sharing initiatives like PlanetLab have reached a global scale with hundreds of resource providers making it difficult to negotiate common agreements across all providers. Therefore, the approach outlined by this work is to keep the overhead for resource providers to a minimum while still relying on a common information model. To achieve this, the modeling focus will be kept on resource configuration aspects. The deployment of virtual resource groupings across federated domains primarily requires configuration actions to be carried out on behalf of a federation user. Therefore, restricting the description of resources to common resource types and their supported configuration parameters will help to satisfy this requirement while minimizing federation-wide agreements in terms of extensive information modeling. The following subsections explain the conventions that need to be respected following the proposed federation methodology with respect to resource modeling and description.

5.1.1 Resource Types and Instances

Resources may exist as *types* and as *instances*. An instance can be instantiated from a type. For example, a resource type might be a virtual machine type with several configuration parameters such as the number of Central Processing Units (CPUs) or a variable amount of memory. ([115], p. 4)

An instantiated virtual machine (representing a resource *instance*) has defined parameters and is an instances of the type virtual machine. *Create* commands are typically executed upon *types* resulting in the creation of a new *instance*. *Delete*, *update*, and *read* commands are typically executed upon *instances* (see also the CRUD commands and resource abstraction covered in section 5.2). ([115], p. 4)

5.1.2 Resource Naming Conventions

Resource instances shall be uniquely identified using an *identifier* derived from the type name. Domain managers are responsible for assigning such an identifier to each resource instance under their control and ensuring the uniqueness of identifiers within their own domain. ([115], p. 4)

An identifier shall not be restricted in length and may consist of any printable American Standard Code for Information Interchange (ASCII) character. Also, they shall consist of a *prefix* and a *local name*. Prefix and local name are separated by a dash character. The last dash found in an identifier is regarded as the separator. Consequently, local names cannot contain a dash character. ([115], p. 4)

Listing 5.1 – Example identifiers on domain manager level

```
1 /node-0/apache-1
2 /node-1/apache-1
```

The two examples identifiers shown in listing 5.1 represent two different instances of an Apache web server software package, although they have the same local name `"1"`. However, they have different prefixes `/node-0/apache` and `/node-1/apache`. The prefixes also illustrate the concept of resource hierarchy where both Apache software instances are hosted by another machine (`node-0` and `node-1`, see also section 5.3).

It has to be noted that on the federation level, an additional prefix is needed in order to map resource instances to domains. A prefix per domain is used, e.g. `fokus.` for the Fraunhofer FOKUS domain and its managed resources. ([115], p. 4)

Listing 5.2 – Example identifier on a federation level

```
fokus./node-0/apache-1
```

5.1.3 Resource Configurations Grammar

The figures 5.2 ([116], p. 1193) and 5.3 ([116], p. 1194) show the relevant information modeling with respect to resources and their configuration. The classes shown in those figures are part of a larger information model given in annex C on page 227 which has been used for the system instantiation. As stated previously, resources exist as *types* and *instances*. This is represented by the two model classes *ResourceSpec*, representing a resource type, and *ResourceInstance*, representing a resource instance. Resource types have associated cost and are defined by their configurations parameters which are represented by the class *ConfigParam*. Configuration parameters "[...] can either be a single parameter (*ConfigParamAtomic*) or a composition of configuration parameters (*ConfigParamComposite*) [...]" ([116], p. 1194).

The class *ResourceInstance* represents resources that have been instantiated from a *ResourceSpec*. Therefore, resource instances are modeled to contain "[...] the actual configuration data values, as opposed to the definition of the configuration parameters. The configuration values are associated with a resource instance through the link with *ConfigurationBase*." ([116], p. 1194) This is depicted in figure 5.3 where *ResourceSpec* has an association with *ConfigParam* and *ResourceInstance* has an association with *ConfigurationBase*. The model class arrangement is made of two composite patterns that form a key-value pair allowing for

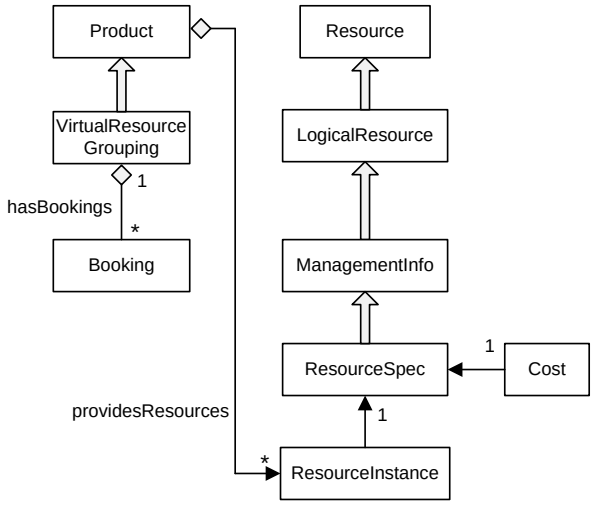


Figure 5.2 – Resource model: resource types, instances, and their relation with virtual resource groupings

great flexibility to define and express resource configurations. "*ConfigParam* (*ConfigParamComposite*, *ConfigParamAtomic*) is a set of configuration parameters where *ConfigurationBase* (*Configuration*, *Configlet*) is a set of values for those parameters." ([116], p. 1194)

Configuration parameters are named, typed and can specify a default value. Possible types are:

- Strings,
- Integers,
- Floating point values,
- Booleans, and
- References to other resource instances (see section 5.3).

Arrays and dictionaries[1] of the mentioned types shall be possible. Attributes shall either be both readable and writable, read only, or write only. One attribute that all resource types have in common is the `"id"` attribute, which identifies a resource instance uniquely as explained on page 75. As shown by the model in figure 5.2, a collection of resource instances with specific configuration form a virtual resource grouping which can be booked by users.

The conventions described in this section allow for the naming of federated resources as well as the structured expression of resource configurations. This allows to abstract from heterogeneous resources and to discover and control them across domains. Furthermore, is allows SET entities to communicate resource configuration aspects across distributed domains and rely on a common information plane enabling higher tier federation logic such as abstract resource relationships and resource orchestration services. To support resource providers in describing their resources, tools can be offered to ease the description process and lower the *entry barrier* for new providers (see also section 6.5.2).

[1]Here, a *dictionary* means an array that uses strings as index values instead of numerical values.

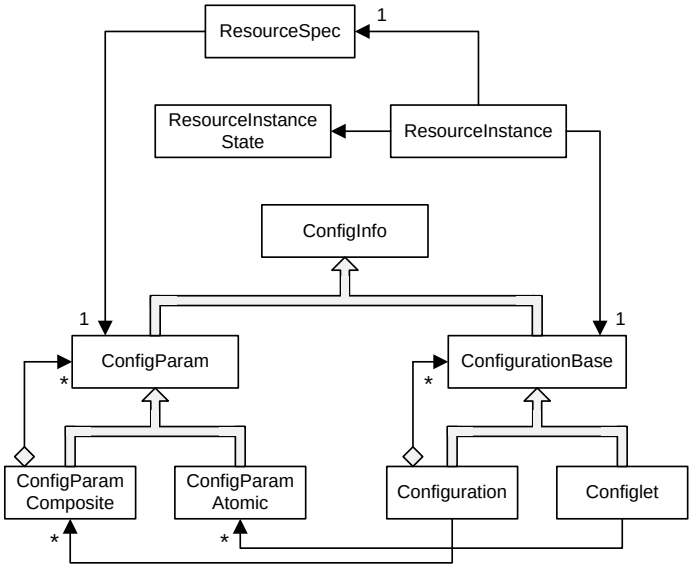


Figure 5.3 – Resource model: key-value pairs are used to express the resource configuration options and the actual configuration data values.

5.2 Resource Abstraction & Control Framework

In order to enable remote resource control, this work describes a resource abstraction and control framework that makes use of the information modeling explained in the previous section. The M entity defined by the federation model (see page 58) shall expose a control interface *i/f M* that can be used to control resources inside a particular domain. In the following, the interface semantics will be specified, enabling resource abstraction and effective cross-domain resource control. The interface i/f M shall represent a *service access point (SAP)* supporting the following four service request primitives:

- Create (a resource instance from a specific type),
- Read (a resource configuration),
- Update (a resource configuration), and
- Delete (a resource instance).

In the following, it will be referred to those requests as the CRUD (create, read, update, delete) commands. A full specification of a CRUD interface for the federation system instantiation is given in annex D.

5.2.1 The Control Framework Service Primitives

The Create Service Primitive Definition

Via a *create* request, a new resource instance of a specific type can be requested to be created in a given configuration. The resource can also be requested to be the child of a parent resource. For details regarding resource hierarchies, please see section 5.3. Listing 5.3 shows a notation of the create request. The resource type name `typename` identifies a resource type that is an instance of the class *ResourceSpec*.

Listing 5.3 – The create request primitive

```
CREATE (parent_id: Identifier, typename: TypeName, config: Configuration,
    vrg: GroupingName) : Identifier
```

In order to execute a create operation, the resource configuration is passed via i/f M in terms of a document that details the configuration parameters. This shall be expressed using a set of key-value pairs as defined by the information model (see page 75). How this information is encoded and how the configuration document is transported, those are implementation details that will be further specified in chapter 6. A specification of the data types and the format is given by the listings provided by annex D on page 229 and page 230. Generally, a resource instance should be assigned to a specific virtual resource grouping. Upon successful creation of the instance, an identifier shall be returned that conforms to the convention described in section 5.1.2.

An example configuration that can be passed along with a create request is shown in the listing 5.4. In this example, the resource `fokus./PortGroup-707316526` is to be created with the given configuration where one of the configuration parameters is a reference to resource `fokus./VLAN-1234`.

Listing 5.4 – Example resource configuration

```
<fokus.PortGroup>
  <id>fokus./PortGroup-707316526</id>
  <state>unprovisioned</state>
  <configuration>
    <vlan type="reference">fokus./VLAN-1234</vlan>
    <name type="string">internal6</name>
  </configuration>
</fokus.PortGroup>
```

The Read Service Primitive Definition

Via the *read* request, the configuration of a specific resource can be retrieved. Listing 5.5 shows a specification of this request.

Listing 5.5 – The read request primitive

```
READ (identifier: Identifier): Configuration
```

Also, i/f M shall support to obtain the identifiers of a specific set of resources. The supported modes shall be:

- Receive the identifiers of all resources of a specific type (see listing 5.6).
- Receive the identifiers of all child resources of a specific parent (see listing 5.7).
- Receive the identifiers of all root resources (see listing 5.8).

Combinations of the above options shall be possible (e.g. to obtain all resource of a specific type that are child resources of a specific parent).

Listing 5.6 – Listing all resources of a specific type

```
READ (typename: TypeName): { Identifier }
```

Listing 5.7 – Listing all child resources of a specific parent

```
READ (parent_id: Identifier): { Identifier }
```

Listing 5.8 – Listing all root resources

```
READ ('\'): { Identifier }
```

The Update Service Primitive Definition

Via the *udpate* command a re-configuration of an existing resource instance can be requested. The new configuration shall not have to include all parameters of the resource instance. It shall be sufficient to include only the parameters that are to be changed. Upon success, the full configuration of the resource instance shall be returned. ([115], p. 4)

Listing 5.9 – The update request primitive

```
UPDATE (id: Identifier, config: Configuration): Configuration
```

The Delete Service Primitive Definition

Via the *delete* command, the deletion of an existing resource instance can be requested. It is up to the M entity implementation to decide if the instance shall actually be deleted or not. Therefore, this request can rather be viewed as an indication that a certain resource instance is not needed by the requester (usually on the federation level) anymore. ([115], p. 4)

Listing 5.10 – The delete request primitive

```
DELETE (identifier: Identifier): None
```

The request primitives as introduced above define the abstract resource control framework on the federation layer. How resource control is handled inside the domains may be implementation specific and shall be transparent to higher tier federation entities. However, in the following a resource abstraction method is described that can be implemented within the domains to manage heterogeneous resources.

5.2.2 Resource Abstraction

Figure 5.4 shows the resource abstraction concept and the two reference points[2] T_1 and T_2. The control framework service primitives described in the previous section define the reference point T_1. The implementation of the domain manager and its i/f M can be domain specific but the conventions for T_1 need to be respected by all i/f M implementations in order to enable effective federation as explained in section 4.3.2. Sufficient security mechanisms (especially authentication and authorization) need to be implemented to prevent unauthorized resource access (see section 6.6 for a system instantiation implementing a domain manager interface on reference point T_1).

The reference point T_2 on the other hand is not strictly specified. This means that resource providers are not restricted in how to control resources inside their domain. Still,

[2] A reference point is a "[...] conceptual point at the conjunction of two non-overlapping functional groups (source: ITU-T I.112)." ([117], p. 25)

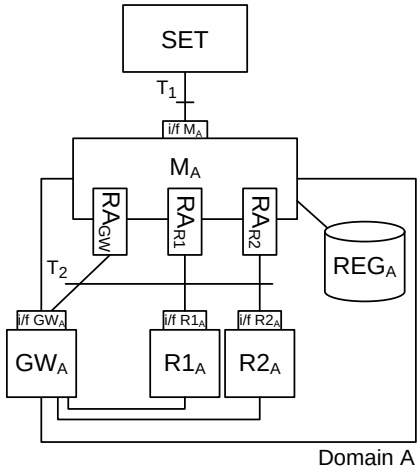


Figure 5.4 – Resource abstraction method: the reference points T_1 and T_2

this work proposes a method that enables effective resource abstraction using the concept of a *resource adapter (RA)*. An RA acts like a device driver in an operating system providing a resource API to the domain manager implementation.

Although T_2 is not strictly specified, resource management standards may be used to implement this reference point such as the OCCI specifications [35] or the 3rd Generation Partnership Project (3GPP) integration reference point (IRP) standards series[3]. Generally, resource providers should be advised to base their RA implementations on open standards as this allows the re-usability of RA implementations across domains. Also, the federation organization might opt to provide a domain manager framework and RA implementations to give new resource providers a kick-start.

However, RA implementations might be considered to be an asset worth protecting. This is likely to occur in commercially oriented federations that follow a market paradigm. In such federations, resource providers—although they are partners—compete against each other in attracting as many federation customers (i.e. users) as possible. In less competitive federations, RA implementations might indeed be shared between resource providers in which case open standards can help to establish a stable T_2 reference point implementation fast and efficiently. Thus, in case two domains A and B use the same domain manager implementation as well as the concept of RAs, and in case open standards exist that define resource control interfaces so that:

$$\text{i/f R1}_\text{A} = \text{i/f R2}_\text{B} \tag{5.1}$$

resource adapter implementations can be used across domains:

$$\text{RA}_{\text{R1}_\text{A}} = \text{RA}_{\text{R2}_\text{B}} \tag{5.2}$$

Building upon both the information modeling described in section 5.1 and the resource abstraction and control mechanisms (i.e. the CRUD primitives and reference points T_1 and T_2) described in this section, higher tier federation logic can be implemented at SET entities providing added value to the federation users. This is described in the following sections.

[3]For example 3GPP TS 32.601 V8.1.0: "Telecommunication management; Configuration Management (CM); Basic CM Integration Reference Point (IRP): Requirements", September 2009.

5.3 Resource Relationships

The key advantage of information modeling and resource descriptions combined with a resource abstraction and control framework as described by the previous sections, is the possibility to build systems making use of higher tier federation logic. This section on resource relationships (such as resource hierarchies and configuration references) introduces a method to design and deploy virtual resource groupings that rely on inter-resource dependencies. Such dependencies allow federation users to express infrastructure requests in a very flexible and elegant way. This can speed up the virtual resource grouping design process and allows for advanced mechanisms on the federation layer such as *resource orchestration* and *automated resource provisioning* as introduced in section 5.4.

5.3.1 Resource Hierarchies

Resource hierarchies are defined by *parent-child relationships*. A typical example is a piece of software that is installed on a virtual machine (VM) whereas the VM is hosted on a physical machine managed by a hypervisor. All three resources can be considered as resource *instances* in terms of the resource model introduced in section 5.1.1. In this example, one could specify a virtual resource grouping containing the three resources with the following relationships between the resources:

- a parent-child relationship between the physical machine and the virtual machine with the physical machine being the parent of the VM (and the VM being the child resource).
- a parent-child relationship between the VM and the software with the VM being the parent resource (and the software being the child).

The following expressions might help to clarify this for domain A where the federated resources offered by this domain are represented by MR_A (see also page 66):

$$MR_A = \{R_{A_x} |\ x = \{1, 2, 3\}\} \tag{5.3}$$

Here, the physical machine (PM), the virtual machine (VM), and the software (SW) are the federated resources provided by this domain:

$$R_{A_1} = PM \qquad R_{A_2} = VM \qquad R_{A_3} = SW \tag{5.4}$$

Furthermore, if p denotes the function that assigns a resource y to be the parent resource of x and if c denotes the function that assigns a resource z to be the child resource of x,

$$p : MR_A \rightarrow MR_A,\ x \mapsto y \tag{5.5}$$

$$c : MR_A \rightarrow MR_A,\ x \mapsto z \tag{5.6}$$

the following example relationships might be defined:

$$p(R_{A_3}) = R_{A_2} \qquad p(R_{A_2}) = R_{A_1} \tag{5.7}$$

which implies:

$$c(R_{A_1}) = R_{A_2} \qquad c(R_{A_2}) = R_{A_3} \tag{5.8}$$

This allows to express more complex relationships and to abstractly define the hierarchical resource relationships of entire virtual resource groupings. For example, it can be deduced:

$$\begin{aligned} p(p(R_{A_3})) &= p(p(SW)) \\ &= p(VM) \\ &= PM = R_{A_1} \end{aligned} \tag{5.9}$$

Note, that for the scope of this work only intra-domain parent-child relationships are defined, so that the following parent function p would be illegitimate:

$$p : MR_A \rightarrow MR_B,\ x \mapsto y \tag{5.10}$$

The possibility to express parent-child relationships as part of the virtual resource grouping design process, allows the user to specify a desired infrastructure in detail. Also, it opens up for another layer of abstraction. In case the user does not specify a full chain of parent-child relationships, such decisions can be taken according to some logic on the federation layer or on the domain layer. For example, the user might specify the following virtual resource grouping:

$$VG = \{R_{A_x} \in MR_A \mid x = \{2, 3\}\} \tag{5.11}$$

with

$$R_{A_2} = \text{fokus./vm-1} \qquad R_{A_3} = \text{fokus./mysql-1} \tag{5.12}$$

and

$$p(R_{A_3}) = R_{A_2} \tag{5.13}$$

In this example, the user has not specifically specified a resource R_{A_1} to be the host for the VM. This situation creates additional deployment flexibility on the domain layer. The domain where the resources shall be deployed has been specified using the federation level identifier `fokus.` representing the Fraunhofer FOKUS domain. However, within the Fraunhofer domain, the VM can be deployed on any available physical host. This allows the domain to take a decision on how to allocate physical machine resources to this virtual resource grouping.

Also, instead of specifying resource *instances* that are tied to a specific domain, resource *types* could have been specified by the user without choosing any particular domain. Such virtual resource grouping specifications allow for even greater deployment flexibility as the parent resource *and* the according domain can be selected using some logic on the federation and/or the domain layer.

How to determine the deployment domain in such cases can be realized according to different global federation strategies or according to user preferences. For example, the domain could be chosen according to the most economic prize in terms of money that the user would be charged by the resource provider for hosting the virtual resource grouping. Many other deployment strategies (e.g. best performance, lowest energy consumption, best network connectivity, fair allocation, etc.) are possible depending on the description granularity of the federated resources. Such global strategies need to be negotiated as part of the federation agreements between resource providers and SET entities.

5.3.2 Configuration References

This section describes how resource configurations can be expressed to depend on other resources and their specific configurations. This shall also work across different resource provider domains. As stated before, one of the possible resource configuration parameter types is a *reference* to another resource instance. The referring resource can use the reference to query configuration parameters from the referenced instance. An *intra domain* reference relationship r_{intra} is defined with

$$r_{intra} : MR_A \rightarrow MR_A,\ x \mapsto y \tag{5.14}$$

that can be used to express that the configuration of resource x depends on the configuration of resource y where x and y are provided by the same domain.

In case x and y are provided by different domains, an *inter domain* configuration reference can be defined:

$$r_{inter} : MR_A \rightarrow MR_B,\ x \mapsto y \tag{5.15}$$

For example, a virtual resource grouping might contain resources of two distinct types: (1) a VM resource and (2) a shared storage resource. The storage shall be available through the Network File System (NFS) protocol[4]. In this example, a VM resources instance shall use a shared storage resource instance as its data storage. The listings 5.11 and 5.12 show example resource type specifications for the VM and the shared storage resource types:

Listing 5.11 – Virtual machine resource type specification

```
<resourceSpec>
  <name>vm</name>
  <attribute name="cpus" type="integer" default="2" />
  <attribute name="memory" type="integer" default="512" />
  <attribute name="storage" type="reference" />
</resourceSpec>
```

Listing 5.12 – Shared storage resource type specification

```
<resourceSpec>
  <name>shared_storage</name>
  <attribute name="size" type="integer" default="10000" />
  <attribute name="uri" type="string" />
</resourceSpec>
```

Once the VM and the storage resources have been instantiated, they could be configured—from a *federation layer* perspective—as shown in listing 5.13 and 5.14.

Listing 5.13 – Shared storage resource instance configuration

```
<shared_storage>
  <id type="string">storage-1</id>
  <size type="integer">20000</size>
  <uri type="string">nfs://11.235.81.3/example</uri>
</shared_storage>
```

[4]RFC 5661: Network File System (NFS) Version 4 Minor Version 1 Protocol, [118]

Listing 5.14 – VM resource instance configuration

```
<vm>
  <id type="string">vm-11</id>
  <cpus type="integer">4</cpus>
  <memory type="integer">2048</memory>
  <storage type="reference">storage-1</storage>
</vm>
```

In listing 5.14, a `storage` parameter is given. It specifies which storage instance is to be used as data storage for the VM instance. On the *domain layer*, the domain manager and essentially the resource adapter (RA) that is responsible for configuring the VM instance can use this reference to query information needed for configuring the VM to access the shared storage resource (in this example, using the NFS Uniform Resource Identifier (URI): `nfs://11.235.81.3/example`).

This example showed how resource configurations can depend on the configuration of remote resources that are offered by a resource provider belonging to the same federation. This method, combined with hierarchical resource relationships (as described in section 5.3.1) allows to specify and provision complex virtual resource groupings where part of the complexity can be abstracted on the federation layer. This, in turn, enables the federation to expose different levels of resource configuration granularity serving the different needs of a wide range of federation users.

5.4 Resource Orchestration & Provisioning

The previous sections already outlined extensive means for remote users to control resources inside distributed domains and define abstract resource relationships. This section aims to provide additional value for federation users in terms of higher tier federation logic that shall be implemented in system components and tools provided by SET entities.

If the resource federation model and the set of methods described here are applied to experimental facilities and testbeds, such concepts allow to provide *testbeds as a service* strongly supporting an experimentally driven research approach. In this context, *resource orchestration* allows federation users to receive a custom environment tailored to satisfy their well-defined abstract request without having to go through all the necessary resource deployment procedures manually. Based on the information model, *resource discovery* mechanisms can be implemented that allow the user to learn about available resources and their configuration options. This enables users to express a request for a set of resources (i.e. a virtual resource grouping as defined in section 4.1). Tools can be designed and implemented to guide the user in the selection and virtual resource grouping definition process. For example, such tool has been designed and implemented as part of the system instantiation described in section 6.5.4). The resource orchestration method outlined here builds upon 2 key aspects: (1) resource orchestration script compilation and (2) resource orchestration script execution.

Resource orchestration script compilation It is assumed that the user derives a specification of a virtual resource grouping *VG*. This specification serves as input for the orchestration script compilation. Clearly separating the VG specification and the orches-

tration script enables to replace the orchestration logic and technology without changing the VG specification. For example, the orchestration could be a Business Process Execution Language (BPEL)[5] based solution or a state machine based workflow engine depending on the specific system requirements. The link between the VG specification and the executable orchestration script is performed by the script compilation.

The compilation shall be realized based on the following orchestration algorithm that defines two central rules:

1. If w is a parent of x then w must be provisioning before x.
2. If y references z then z must be provisioned before y.

Those rules allow to derive a sorted sequence of provisioning requests (i.e. CRUD requests) and resolve inter-resource configuration dependencies. The sequencing shall be performed by computing a dependency graph of the required actions. Child resources (or *contained* resources) depend on the parent resource (or the *container*). Hence, the container has to be created first. A referencing resource depends on the resource that is being referenced. Hence, the referenced resource needs to be created first.

As an example, consider the following virtual resource grouping VG to be defined as:

$$\begin{aligned} VG &= \{R_{A_x} \in MR_A \mid x = \{1, 2, ..., 5\}\} \\ &= \{A, B, C, D, E\} \end{aligned} \tag{5.16}$$

Further, let a *parent-child* relationship

$$p : MR_A \rightarrow MR_A,\ x \mapsto y \tag{5.17}$$

be defined so that x is the parent of y (see also section 5.3.1) and let a *reference* relationship

$$r : MR_A \rightarrow MR_A,\ x \mapsto y \tag{5.18}$$

be defined so that x is referenced by y (see also section 5.3.2).

Furthermore, let's assume that the user specified the following resource relationships as part of the VG specification:

$$p(B) = A \qquad p(C) = B \qquad r(E) = B \qquad r(D) = C \tag{5.19}$$

Graphically, this could be represented as shown by figure 5.5.

From this resource relationship specification, several valid provisioning sequences are possible such as:

$$\{A, E, D, B, C\} \qquad \{A, E, B, D, C\} \qquad \{E, D, A, B, C\} \tag{5.20}$$

Thus, several sequences can be derived that respect the orchestration rules defined above. Which precise order to derive may be impacted by the *provisioning mode*. In an *sequential* mode, all resource configuration requests are ordered so that the execution steps will

[5] Web Services Business Process Execution Language Version 2.0, OASIS Standard, [119]

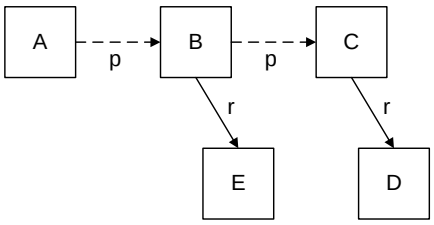


Figure 5.5 – Resource relationships

be performed one after the other. In a *parallel* mode, several configuration steps can be performed at the same time. This requires an asynchronous communication mode with the domain managers using notifications about the status of pending provisioning actions.

In the above example, the provisioning actions for the following resources can be performed in parallel in a first step $\{A, E, D\}$ as those resources themselves do not depend on other resources. At least this is true for the federation layer. If E and D depend on other resources (e.g. on a potential container) on the domain layer, is up to the providing domain and is abstracted on the federation layer. However, further actions might be necessary on the domain layer to find or instantiate an appropriate parent resource for E and/or D. Please refer to section 6.6 for a description of a domain manager system that is able to handle such domain-internal dependencies.

Once $\{A, E\}$ have been provisioned successfully, the provisioning of $\{B\}$ can be initiated. In a final step, resource $\{C\}$ that depends on $\{B, D\}$, can be provisioned.

It has to be noted that it is possible to design virtual resource groupings that contain cyclic relationships. Such designs are problematic at the resource orchestration and provisioning stage as a clear provisioning order cannot be determined using the rules above. However, the practical experienced gained from implementing the described method (see section 6.5.7) shows that restricting virtual resource grouping designs to layouts that do not contain cyclic relationships still leaves sufficient design freedom to the federation user. The tools supporting the user in deriving a valid virtual resource grouping specification may implement respective warning messages, if a cyclic layout has been detected by the system.

Resource orchestration script execution Once the orchestration script has been compiled it shall be executed by invoking the actions defined by the script. The execution shall either result in a set of actions performed sequentially until the last action has been performed or be reflected by a state machine. State machines may implement waiting conditions and asynchronous events (i.e. notifications).

The key advantage that a state machine provides is the possibility for parallel execution of resource configuration actions. Depending on the nature and complexity of the requested virtual resource grouping, parallel execution can reduce the overall provisioning time of a virtual resource grouping considerably.

The resource orchestration script execution results in the step by step provisioning of the virtual resource grouping. It provides the missing link between the VG specification and the *deployed* custom VG infrastructure. The infrastructure provisioning is realized according to

the configuration specified by the federation user. There are different possibilities of what the user may request in terms of resource instances resulting in different provisioning options. Table 5.1 shows the different cases and the resulting provisioning commands. Combinations of the different cases are possible resulting in more complex orchestration and provisioning actions.

Table 5.1 – Examples of user requests and respective provisioning actions (non-exhaustive list)

User Request	Provisioning Action
New resource instance	CREATE
Existing resource instance with new configuration	UPDATE
Delete resource instance	DELETE
Existing virtual resource grouping with new instances	multiple CREATE + multiple UPDATE
Instances removed from an existing virtual resource grouping	multiple DELETE + multiple UPDATE
Deletion of a virtual resource grouping	multiple DELETE

Figure 5.6 depicts the entire chain of events in order to derive a deployed virtual resource grouping starting from a VG specification derived by the user. The specification is passed to the SET entity where the orchestration actions are performed. The specification might have been derived by the user manually or using specific tools. Such tools might also by supplied by a SET entity.

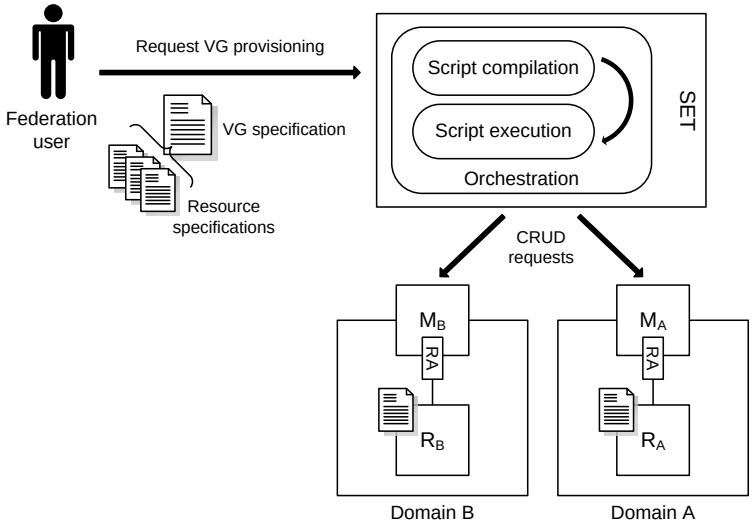


Figure 5.6 – From a virtual resource grouping specification to resource orchestration and provisioning

The VG specification shall use the resource descriptions (i.e. resource types, identifiers, and configuration values) as explained in section 5.1. The specification is passed to the SET entity where it shall be processed by an orchestration entity. In a two-step process (resource orchestration script compilation and execution), the orchestration entity initiates

the resource provisioning. As shown in table 5.1, depending on how the VG specification is composed, a series of CRUD requests results. From the resource identifiers, the domains that are responsible for the resources in question can be derived. Therefore, the CRUD requests can be directed to the correct domain manager (i.e. the M entity). The domain managers are responsible for forwarding the requested configuration to the respective resources. Standard or proprietary resource adapters can be used for intra-domain communication with different resources. Once the resource configuration has been performed, responses are propagated back to the user indicating success of failure of the requested resource deployment. Annex A provides example traces of control messages and configuration documents (e.g. the VG specification, intra-domain resource configuration, etc.) generated throughout this process.

In case the provisioning fails, the orchestration entity shall perform a *rollback* operation. This involves reverting any requested and successfully performed resource configurations of existing instances (UPDATE) or a deletion of newly created resources (one DELETE for every successful CREATE). In short, the following general strategy shall be applied:

- The rollback of a CREATE operation is a DELETE operation.
- The rollback of an UPDATE operation is an UPDATE operation with the original resource configuration.
- The rollback of a DELETE operation is a CREATE operation with the original resource configuration.

5.5 Cross-domain Resource Interconnection

This section describes a method for interconnecting resources belonging to a specific virtual resource grouping across provider domains so that they can be accessed by a federation user as part of a single virtual environment. Figure 5.7 (adapted from [115], p. 7) demonstrates this concept based on the gateway model entity as introduced in section 4.1 using four federated domains.

Each domain provides a gateway (i. e. the federation model GW entity) and several resources that have a direct physical connection to the gateway of their respective domain. The method presented here allows communication between the resources of the same virtual resource grouping. Other resources that are not part of the same grouping are excluded from grouping-internal communication. This behavior can be disabled by assigning publicly reachable addresses to resources. In such cases, the resource itself shall implement sufficient security mechanisms to avoid resource misuse. It is up to the resource provider to allow or deny public resource exposure.

Figure 5.7 shows several resources that have been instantiated to be part of two distinct virtual resource groupings and that are interconnected across the domains A and B:

$$VG_1 = \{R2_A, R1_B\} \qquad VG_2 = \{R3_A, R2_B\} \tag{5.21}$$

The resources $R2_A$ and $R1_B$ that are both part of VG_1 are connected by three different network segments. The first segment is the connection between $R1_B$ and GW_B. The second segment is the network path between GW_B and GW_A. The third segment is the domain A internal connection between GW_A and $R2_A$. Collectively, the three network segments enable

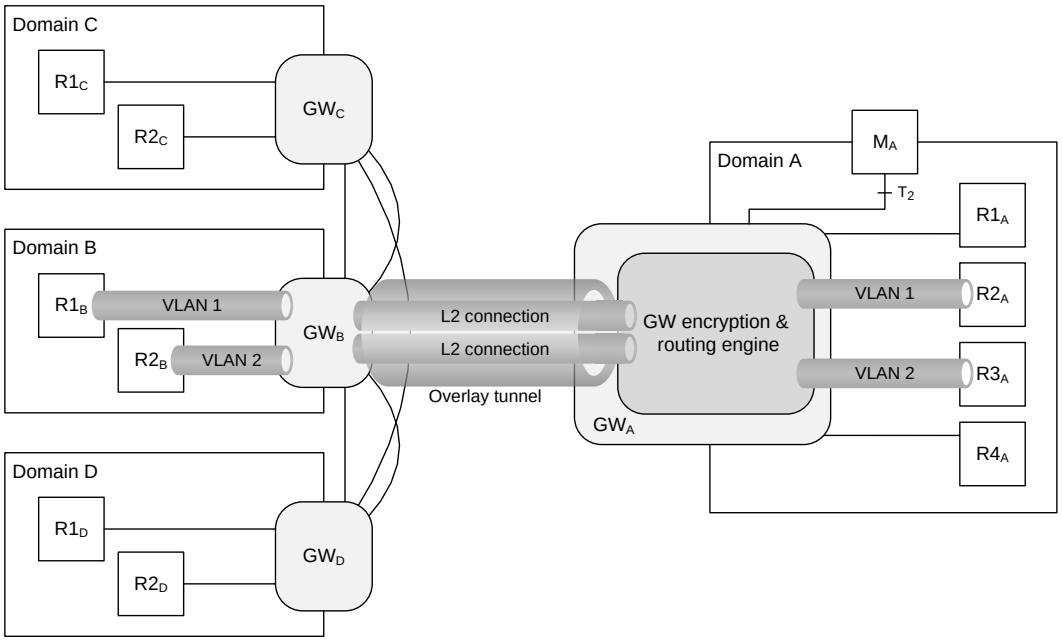


Figure 5.7 – Interconnecting resources provided by federated domains using virtual overlay networking techniques

data packets exchange between those two resources. Similar segments can be identified for the resources $R2_B$ and $R3_A$ that both belong to VG_2.

Using this network constellation, no communication shall be possible between two resources being part of different virtual resource groupings (e.g. between $R1_B$ and $R3_A$). The gateways might establish site-to-site virtual network overlay tunnels between all participating domains in a *full mesh network topology*. Alternatively, a *star topology* might be established using a central virtual network concentrator (e.g. maintained by the federation organization). Which network topology is chosen as an inter-domain connection layout is implementation dependent and might be negotiated between the federated domains. Also, it depends on the size of the federation (i.e. the total number of participating domains) to avoid the so-called *n-square problem*. In a full mesh of n nodes, each node connects to $n-1$ pears, resulting in a total number of $n(n-1)/2$ connections.

The tunnels can for example be established using VPN technology[6]. Essentially, the gateways realize *collision domain isolation*. A *collision domain* is "[...] an isolated network segment where data packets are sent on a shared channel." ([115], p.7) Inside the domains, the respective resource providers are responsible to ensure isolation. Virtual local area network (VLAN) techniques and technologies (e.g. IEEE 802.1Q VLAN technology[7]) can be used to ensure collision domain isolation across the entire network path between resources of different domains. Following this concept, several virtual *layer 2 connections*[8] can be established via the network overlay tunnel between two gateways enabling communication between resources

[6]RFC 4026: Provider Provisioned Virtual Private Network (VPN) Terminology, [120]

[7]IEEE Standard for Local and metropolitan area networks: Virtual Bridged Local Area Networks, 802.1Q, [121]

[8]Here, the term *layer 2* refers to the second Open Systems Interconnection model (OSI model) layer as defined by [122].

of the same virtual resource grouping using protocols on layer 2 and above. The GW entity shall establish, maintain, and protect the mapping of local collision domain communication to external wide area network connections. Also, each gateway performs routing and en-/decryption of data plane packets between those channels. If required, QoS rules may be enforced on routing decisions (e.g. limitations on the maximum throughput rate). Also, the gateways shall act as a firewall enforcing restrictions defined by the resource provider governing the respective domain. ([115], p. 9)

By default, the domains are foreseen to be interconnected via public Internet. Therefore, the virtual overlay network tunnel between domains represents an Internet overlay. However, if any domain disposes of more advanced interconnection facilities such as optical fiber equipment and the gateways are implemented and installed accordingly, real QoS reservations are possible. However, for the scope of this work, a public Internet connection is assumed as the default connection for any federated domain.

The north side of any gateway is a control and communication layer facing the domain manager (and the respective gateway RA implementation) to communicate on reference point T_2 (see figures 5.4 and 5.7). Using suitable domain manager and resource adapter implementations, this allows to expose configuration capabilities of the gateways to SET entities. How much configuration options are exposed on the federation layer depends on the configuration granularity requested by the targeted federation users. As such decisions can impact the domain-internal network topology design and domain specific security policies, they might be regulated by federation-wide agreements depending on how much policy alignment across the federation can be achieved.

Authorized users (using the federation credentials) shall be able to access resources part of a virtual resource grouping by connecting to the virtual network that connects the resources belonging to the same grouping. This requires an integration with the federation IDM system. Layer Two Tunneling Protocol (L2TP)[9] based connections can be used to give direct access to a specific virtual resource grouping as if the federation user would work from within a domain providing a gateway. See section 6.7 for a system instantiation implementing such solution.

5.6 Summary

Targeting the second and third part of the research hypothesis postulated in section 1.3, this chapter introduced a number of methods enabling generic resource federation across different resource provider domains. The methods build upon the federation model defined in chapter 4 and describe various processes required to derive functional virtual resource groupings from a set of heterogeneous distributed resources. The methods address the key areas of resource description, abstraction, orchestration, provisioning, and interconnection.

The next chapter describes a system instantiation that builds upon the federation model and the federation methods.

[9] RFC 2661: Layer Two Tunneling Protocol “L2TP”, [123]

6 Resource Federation Framework Instantiation

This chapter introduces the specification and implementation of the third design artifact: the *framework instantiation*. The federation model and the federation methods defined in chapter 4 and chapter 5 respectively are applied here to derive a system instantiation that can be evaluated in a real life deployment. In fact, the federation framework instantiation has been used and evaluated in several projects as discussed in chapter 7. In terms of the applied research methodology, an instantiation is regarded as defined by Hevner et al.:

> *Instantiations show that constructs, models, or methods can be implemented in a working system. They demonstrate feasibility, enabling concrete assessment of an artifact's suitability to its intended purpose. They also enable researchers to learn about the real world, how the artifact affects it, and how users appropriate it.* [11], p. 79

Starting with a specification of the system requirements, the system context and the goals of the different stakeholders are defined in section 6.1, followed by a use case analysis and some framework design decisions. Section 6.4, 6.5, and 6.6 address the aspects of a cross-domain system implementation and operation using a framework called *Teagle*. Federation and domain level issues are described in individual sections but are linked via a common federation control framework. Section 6.7 provides insights into cross-domain resource connectivity aspects.

6.1 Requirements

6.1.1 Notation

For the definition of the system requirements that are strongly related to the different stakeholder goals, the user requirements notation (URN) has been used. The URN has been specified by the ITU-T in the recommendations Z.150 (URN language requirements and framework) [124] and Z.151 (URN language definition) [125]. According to those specifications, the URN is "[...] intended for the elicitation, analysis, specification, and validation of requirements. URN combines modelling concepts and notations for goals and intentions (mainly for non-functional requirements and quality attributes) and scenarios (mainly for operational requirements, functional requirements, and performance and architectural reasoning). In

particular, URN has concepts for the specification of goals, non-functional requirements, rationales, behaviour, scenarios, and structuring." ([125], p. 1)

URN defines two different sub-notations: a *goal* sub-notation that is called goal-oriented requirements language (GRL) as well as a *scenario* sub-notation that is called use case map (UCM). GRL allows to graphically model stakeholder goals, believes, tasks, etc. in order to capture requirements that are difficult to describe using other means. UCMs allow to describe functional requirements for software systems providing language constructs for flows, actors, events, etc. Figure 6.1 ([125], p. 8) shows the abstract grammar used by the URN specification.

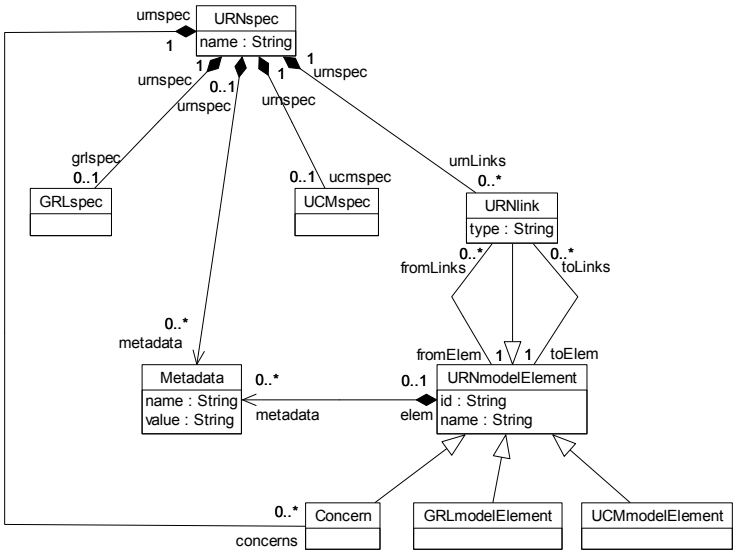


Figure 6.1 – The URN abstract grammar given as a UML model including links, metadata, and the GRL and UCM model element classes.

Goal-Oriented Requirements Language

The GRL grammar is shown in figure 6.3 ([125], p. 18). In particular, for GRL based figures, this thesis makes extensive use of the intentional elements: *goal*, *softgoal*, *actor*, and *task*.

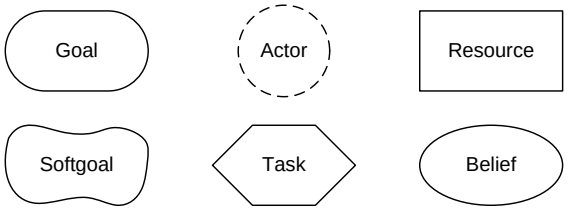


Figure 6.2 – The GRL intentional element type symbols used in GRL based figures

A *goal* is defined as "[...] a condition or state of affairs in the world that the stakeholders would like to achieve. How the goal is to be achieved is not specified, allowing alternatives to be considered. A goal can be either a business goal or a system goal [...]" ([125], p. 22). The goal intentional element type is graphically represented by a rounded rectangle.

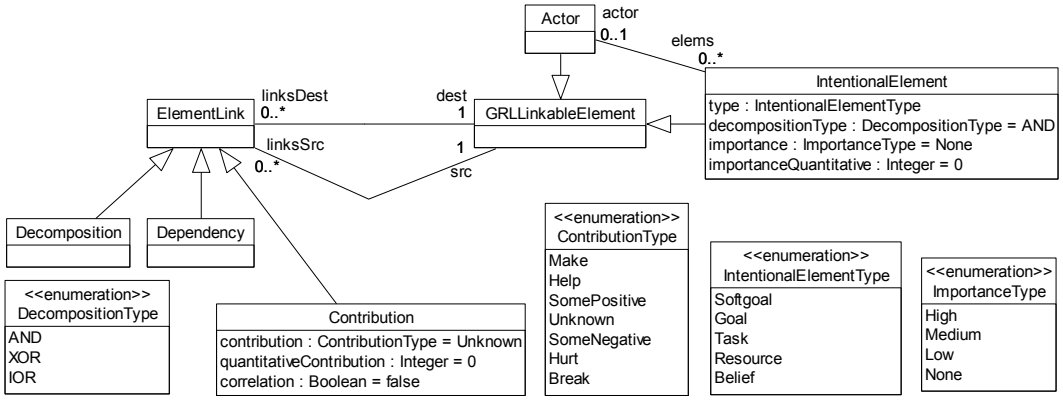


Figure 6.3 – The GRL abstract grammar given as a UML model including the important intentional elements: actor, goal, softgoal, and task.

On the other hand a *softgoal* is "[...] a condition or state of affairs in the world that the actor would like to achieve, but unlike in the concept of (hard) goal, there are no clear-cut criteria for whether the condition is achieved, and it is up to subjective judgement and interpretation of the modeller to judge whether a particular state of affairs in fact achieves sufficiently the stated softgoal. Softgoals are often used to describe qualities and non-functional aspects such as security, robustness, performance, usability, etc." ([125], p. 22) A softgoal is graphically represented by a rounded form as shown in figure 6.2.

An *actor* is represented by a circle while the actor boundary is indicated by a dashed line. An actor "[...] has intentions and carries out actions to achieve its goals by exercising its know-how. Actor definitions are often used to represent stakeholders as well as systems. Actor definitions may contain intentional elements." ([125], p. 18)

Furthermore, a *task* "[...] specifies a particular way of doing something. When a task is part of the decomposition of a (higher-level) task, this restricts the higher-level task to that particular course of action. Tasks can also be seen as the solutions in the target system, which will address (or operationalize) goals and softgoals. These solutions provide operations, processes, data representations, structuring, constraints, and agents in the target system to meet the needs stated in the goals and softgoals." ([125], p. 22) Tasks are graphically represented by a hexagon (see figure 6.2).

For a description of the other GRL intentional elements (resource and belief), please refer to [125]. In addition to elements, GRL specifies relationships between elements that are expressed as an *ElementLink*. Such relationships are important to enable GRL *evaluation strategies*. The different types of links are: *contribution*, *dependency*, and *decomposition*. This work primarily uses *contributions*.

A *contribution* is a link that "[...] defines the level of impact that the satisfaction of a source intentional element has on the satisfaction of a destination intentional element. If the impact is qualitative (positive or negative, sufficient or insufficient [...]), then contribution will be used in goal model evaluations. The impact can be also quantitative (value in [-100, 100]) in which case quantitative contribution will be used in goal model evaluations [...] " ([125], p. 25). The different contributions and the according symbols are shown in figure 6.4.

Contributions are used in GRL *evaluation strategies* to evaluate how well a certain goal has been achieved. "GRL strategies are sets of initial evaluation values given to some

Figure 6.4 – The contribution types and their respective symbols

intentional elements in a GRL model. These evaluation values, which can be quantitative or qualitative, are satisfaction levels that can then be propagated to the other intentional elements in the GRL model through the various decomposition, contribution, and dependency links connecting them. Evaluations are used to assess how well goals in a model are achieved in a given context, which enables the selection of alternatives that represent appropriate trade-offs amongst the often conflicting goals of the stakeholders/actors involved." ([125], p. 32-33)

In section 6.1.2, the GRL is used to analyze different manifestations of the *federation model* entities in order to evaluate how the different stakeholder goals are effected by the choice of certain values for system variables (e.g. common control framework vs. heterogeneous control interfaces). This will allow to define the system requirements and use cases using UCMs in section 6.3 in order to derive a well-justified contextualized *federation system instantiation*. This demonstrates the value of the *federation model* allowing for a structured analysis of the impact on the federated system caused by the modification of key variables.

6.1.2 GRL-Based Stakeholder Requirements Definition and Analysis

The different stakeholders roles have been defined in section 4.2. Three different roles were introduced: the *federation user*, the *federation organization*, and the *resource provider*. Figure 6.5 shows the stakeholders and some of their goals, softgoals, and tasks. Also, the most important contributions on specific intentional elements are shown. To keep the figure as simple as possible, the focus of the analysis has been placed on those intentional elements and their respective contributions that have a major impact on the critical design decisions for the federation system instantiation.

The Federation User Goals

Easy to use This is a softgoal and can be understood as a non-functional system requirement. The system shall be as easy to use as possible while providing the desired functionality defined by the other user goals.

Federation-wide resource discovery The user requires to learn about the resources offered by the federated domains. This is a prerequisite for controlling and accessing resources. Such dependencies can be expressed with the GRL, but are omitted in figure 6.5 to avoid overloading the figure.

Control resources across domains This is one of the critical goals for a user to actually use the system. By controlling resources across federated domains, he shall be enabled to design a virtual infrastructure consisting of distributed resources offered by different resource providers.

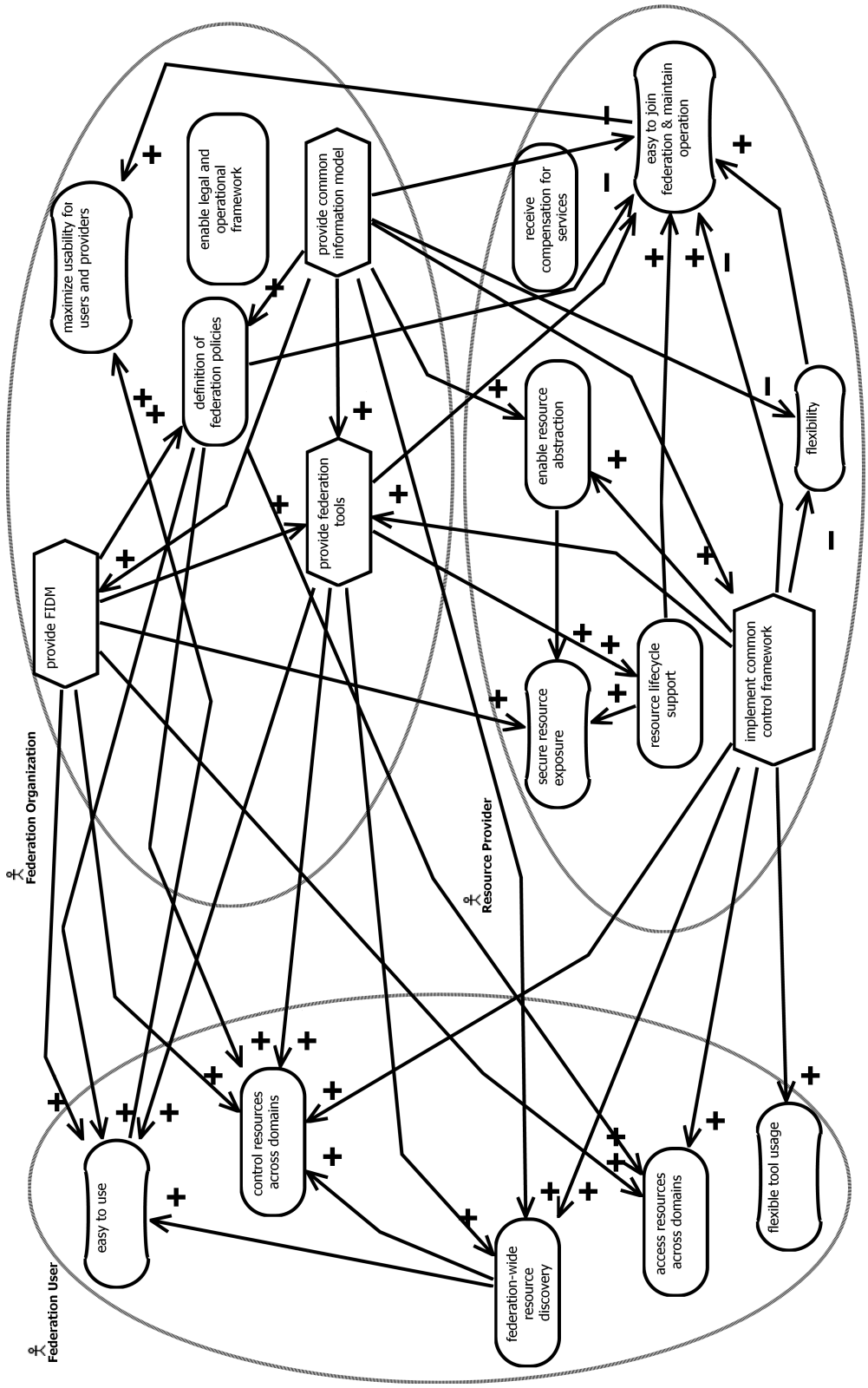


Figure 6.5 – Overview of the most important stakeholder goals and correlations with the highest impact on the system design

Access resources across domains This is also one of the critical goals for a user interacting with the federation framework. The purpose of controlling resources to design a desired virtual resource grouping is to access such resources and execute desired actions.

Flexible tool usage Although one of the federation organization goals is to provide tools to work with federated resources, it is likely for users to be familiar with existing tools (e.g. simulation/emulation tools, monitoring tools, etc.). Therefore, they might require to connect such tools to the federation system. This requirement was actually among the early feedback received from users of the system instantiation described in section 6.5. Therefore, it has been added as an important federation user intentional element after the first design/evaluation circle (see also *design as a search process*, section 1.4).

The Federation Organization Goals and Tasks

Provide federated identity management (FIDM) Identity management provides the basis for numerous tasks that need to be performed in a distributed system such as authentication and authorization. Therefore, the impact of this task on the other stakeholder goals is significant.

Provide federation tools Providing tools allows users and providers to interact with the federation increasing the overall system usability for the other stakeholders.

Provide common information model An information model that is agreed upon and accepted across the participating domains provides a number of benefits as all system entities can rely on a common understanding in terms of terminology. For example, common resource descriptions—based on a common information model—allow for the design and use of powerful federation tools that can implement abstract logic independent of specific domains.

Definition of federation policies Being one of the critical design variables for the federation framework this goal allows the analysis of global policies on the entire system (see also policy analysis starting on page 104).

Maximize usability for users and providers Representing a service broker, the federation organization serves the needs and requirements of the other two stakeholders. Therefore, an important goal is to ensure that the entire system usability is maximized. During GRL evaluation (see pages 98 - 104), this softgoal is a good indicator for the overall framework performance in terms of usability and stakeholder satisfaction for a given specific system configuration (i.e. a specific manifestation of system variables).

Enable legal and operational framework This thesis deals primarily with the technical aspects of resource federation. However, this goal has been mentioned here to demonstrate that there are other aspects besides technical ones that need to be considered when establishing a resource federation. From the early experiences gained during the technical work on the federation framework design and the numerous interactions with resource providers, it became clear that legal and operational aspects need to be dealt with and that such issues can become equally complex in large federations. Besides technical work, the Panlab project addressed such questions[1].

[1] http://www.panlab.net/organizational-framework.html

The Resource Provider Goals and Tasks

Secure resource exposure The resource providers are interested in offering their resources to users in a secure manner. Corporate security guidelines usually regulate the conditions under which resources can be exposed (if at all) to externals. Therefore, the federation framework needs to be both flexible and restrictive in terms of resource exposure at the same time in order to accommodate the different resource provider needs. This will ultimately result in a well-balanced trade-off between local and global resource usage policies. In addition, security depends on additional aspects (that have not been modeled to avoid overloading the figures), such as: accepted security standards and routines, hard- and software security, etc.

Implement a common control framework As discussed in section 4.3.2, this task is one of the most important variables in terms of the federation model and the overall federation framework. Figure 6.5 shows that the task has considerable impact on intentional elements of the other stakeholders but also on a number of resource provider goals such as resource abstraction, flexibility, and how easy it is for resource providers to join and operate the federation infrastructure. The analysis shown in figure 6.7 reveals that the task has both positive and negative correlations with other very important goals.

Enable resource abstraction As providers will offer heterogeneous resources, resource abstraction capabilities and their implementation in terms of middleware components are important to support a wide range of resource types. The goal correlates with other crucial goals such as security, common information model, and control framework.

Flexibility Resource providers are interested in keeping things as simple as possible in order to stay flexible in terms of federation commitment as well as implementation and maintenance efforts. Therefore, flexibility is a usability indicator for resource providers. The analysis starting on page 98 reveals that the evaluation of this softgoal is countercyclical compared with many other goals due to the negative correlation with the control framework. Therefore, sufficient attention needs to be payed to this softgoal if a high priority is assumed.

Easy to join federation and maintain operation In order to attract as many resource providers as possible, joining the federation should be as easy as possible for resource providers. Also, the operation of the necessary federation infrastructure should be realizable for resource providers with reasonable efforts. Unfortunately, there is a trade-off between resource provider and federation user usability. This is discussed in more detail in the analysis beginning on page 98.

Resource life cycle support Offering resources to the federation users requires the providers to manage the resource life cycle. Providers should be supported by the federation infrastructure (i.e. the middleware implementation) and tools to realize this goal.

Receive compensation for services Although not a technical issue and therefore not a target of the analysis, this goal has been added here to demonstrate that there must be a clear business strategy to attract resource providers. The compensation does not

necessarily have to be monetary. For example, academic resource providers might agree to join a federation simply in order to gain access to a scientific community. However, for any federation framework instantiation that is expected to operate in a target business environment such aspects need to be taken into account besides the technical approach.

This subsection summarized the different stakeholder goals, softgoals, and tasks (i.e. the stakeholder GRL intentional elements). The next subsections analyses the correlations between those elements using qualitative GRL evaluation strategies.

Information Model Impact Analysis

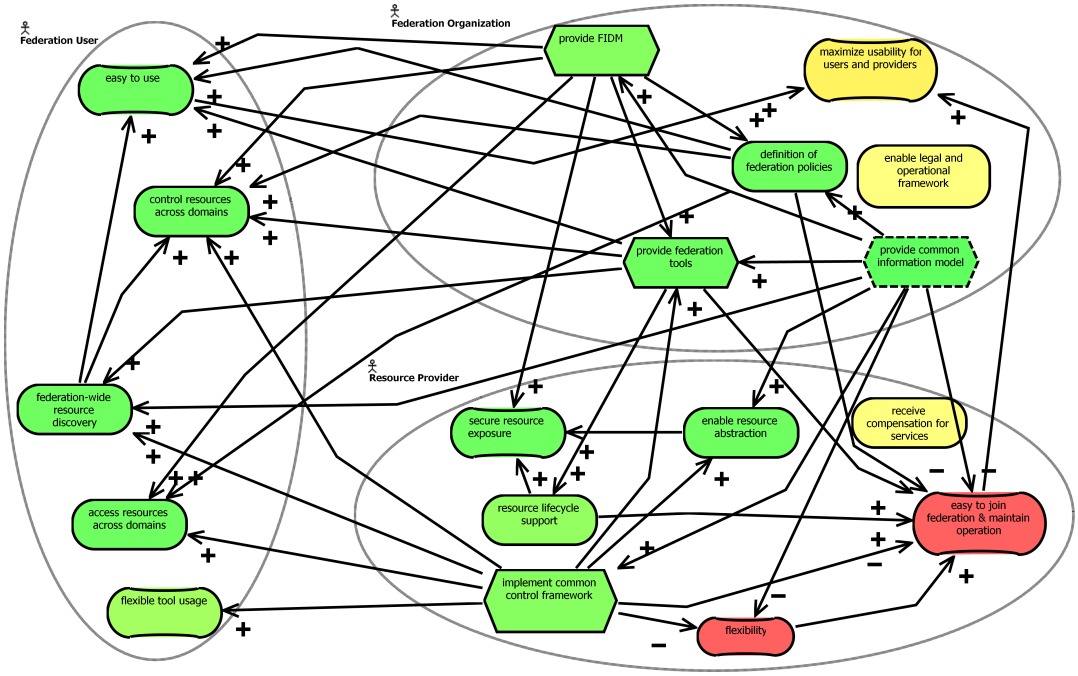
This section analyses the impact of a *common information model* on the system of stakeholder goal correlations. A common information model allows to derive common resource descriptions across the domains that participate in the federation. Also, a common information model supports the design of feature-rich federation tools and stable federation processes.

Figure 6.6 shows the result of this analysis. Sub-figure (a) illustrates a situation where a common information model is provided while sub-figure (b) shows the opposite case. To derive those figures, the goal *provide common information model* has been set to "satisfied" for sub-figure (a) while it has been set to "denied" in case of sub-figure (b).

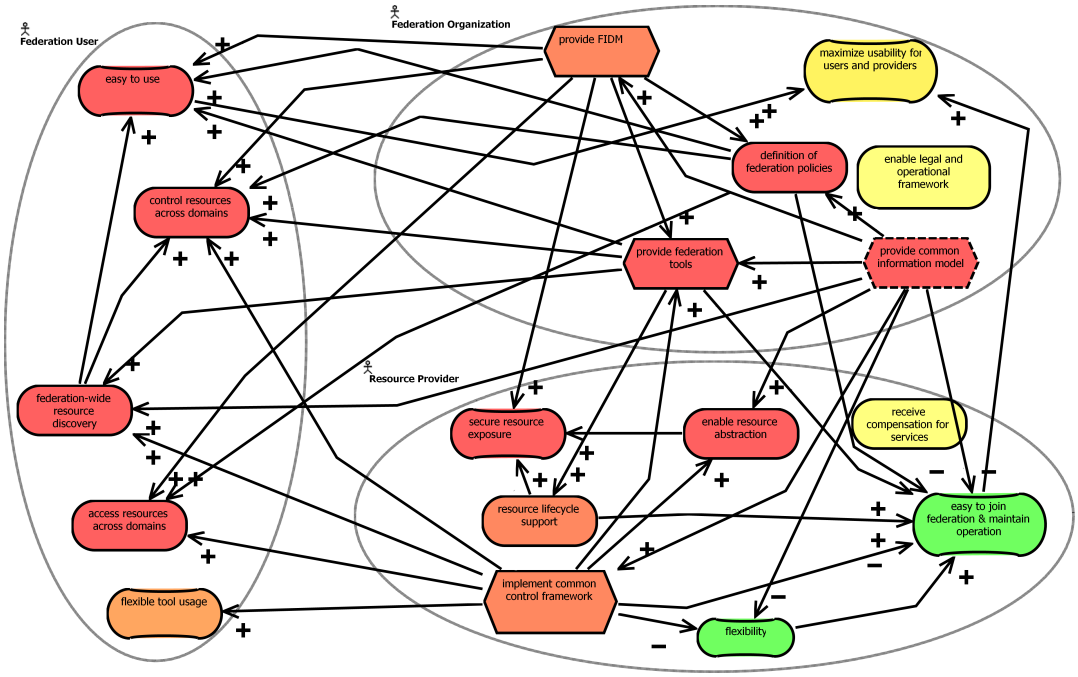
It can be observed that providing a common information model helps to achieve all of the federation user goals either directly (e.g. a common model directly impacts *resource discovery*) or indirectly via another goal (e.g. a common model has a positive impact on the design of *federation tools* which has a positive impact on *controlling resources across domains*). On the other hand, providing a common model negatively impacts two resource provider goals: *flexibility* and *easy to join federation and maintain operation*. This observation can be explained as follows: describing all resources that a provider intends to expose to the federation in terms of a common model, can result in significant overhead for providers. This reduces the flexibility and easiness to join a federation. Also, although having a common model helps to implement a common control framework, the need to have such a control framework in the first place, has a negative impact on resource providers flexibility and the easiness of joining the federation and operating the necessary control framework components.

Another interesting observation is the impact of the common model on the federation organization goal *maximize usability for users and providers*. It can be observed that this goal is equally evaluated for both cases shown in the sub-figures of figure 6.6. This is because it is impacted by federation user and provider goals with contrary evaluation. Therefore, when designing the federation system instantiation, there is a trade-off between federation user and provider goals satisfaction.

It remains to be noted that the positive correlation between the common model and most of the federation organization and user goals clearly suggests the design decision of providing a common model. However, it will be important to provide sufficient other incentives for resource providers to join the federation in order to lessen the negative impact of the common model on the provider goals *flexibility* and *easy to join federation and maintain operation*.



(a) The federation organization provides a common information model.



(b) The federation organization does not provide a common information model.

Figure 6.6 – GRL evaluation: the information model impact

Control Framework Impact Analysis

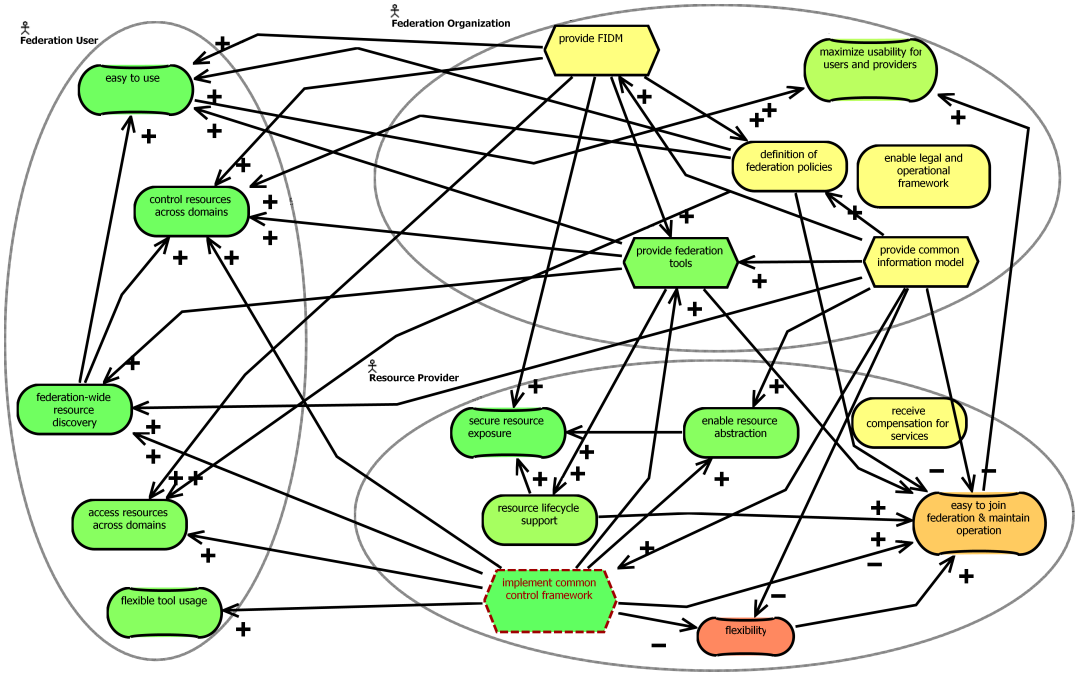
Figure 6.7 shows the GRL evaluation of the resource provider task *implement a common control framework*. Sub-figure (a) shows the situation in which a common control framework is implemented across all resource providers, while sub-figure (b) shows the opposite situation.

Whether or not a common control framework exists mainly impacts federation user goals such as *control resources across domains* and *resource discovery*. Therefore, having a common control framework, indirectly leads to a high usability of the federation framework from a user perspective.

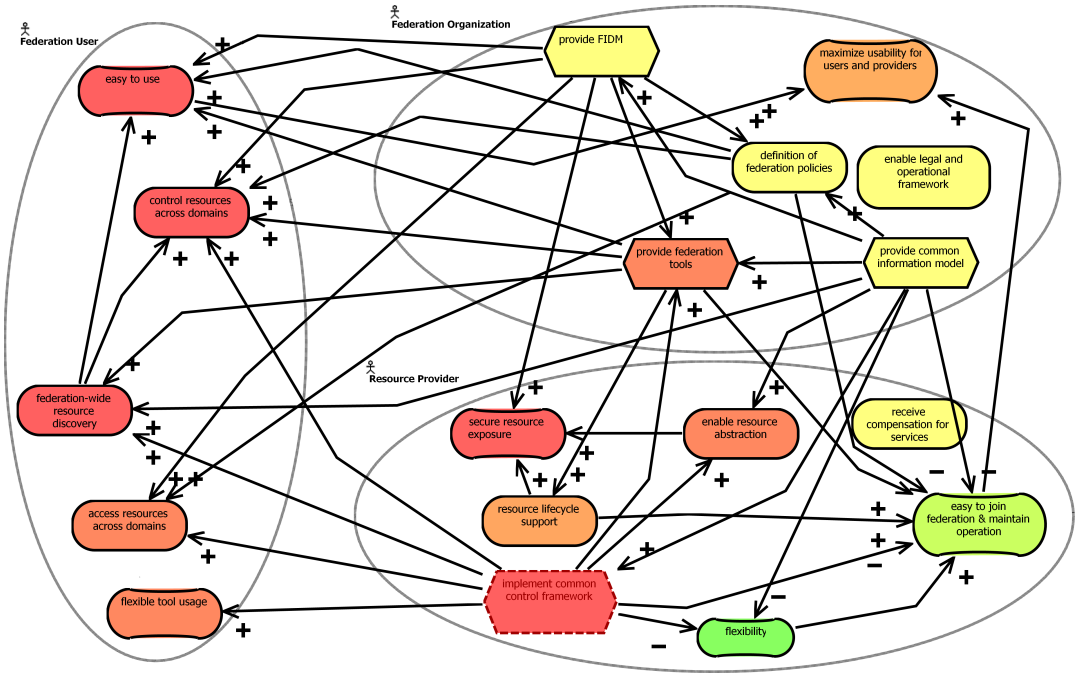
The federation organization goal *maximize usability for users and providers* is also influenced positively but to a lesser degree. This is interesting compared to the results of the common information model impact analysis where the effect of the choice of having a common information model or not, did not have a very clear influence on this goal. However, in case of the common control framework, it can be concluded that having such framework positively influences the overall usability for users and providers. This is due to the lesser negative impact of the control framework on the provider goal *easy to join federation and maintain operation*. To judge if this is a realistic result, the experiences gained up to date from prototyping and running a federation framework (see also the field study in section 7.1), can be considered. Indeed, as mentioned in the previous section, describing all provider resources and processes in terms of a common model can be considered as a considerable hurdle for resource providers to join the federation. On the other hand, for implementing a common control framework it can be expected that sufficiently detailed specifications and even ready-to-use software components are available to resource providers. Therefore, joining a federation framework that implements a common control framework will in most cases—at least from a technical point of view—simply result in installing and configuring the required software components (i.e. the M entity in terms of the federation model, see page 58) at the border of a resource provider domain.

Thus, the results obtained from this analysis are in line with the feedback received from users and providers so far. However, relying on existing prototype implementations might not be acceptable for some providers. This insight was gained from implementing the domain manager control framework software (see 6.6) and the early feedback received from some resource providers. It appears that especially large industrial players only trust in in-house developments whenever components are security relevant.[2] Therefore, detailed specification and the use of open standards is important to ease such developments.

[2]In our specific case an external implementation of the M entity was refused by a large industrial resource provider. Any security critical implementation would have to be an in-house development. Relying on an external M entity implementation allowing for resource exposure was considered to be an unacceptable security risk for this specific organization.



(a) The federation relies on a common control framework.



(b) The federation cannot rely on a common control framework.

Figure 6.7 – GRL evaluation: the control framework impact

Identity Management Impact Analysis

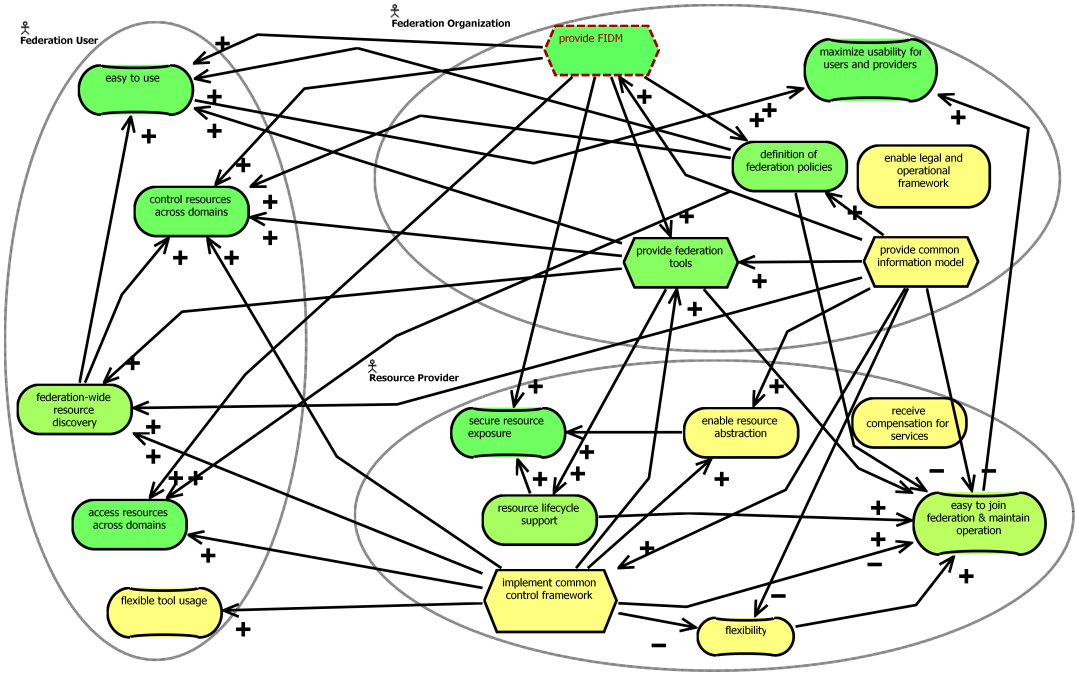
This section analyses the impact of a federated identity management (FIDM) system. Figure 6.8 graphically shows the result of this analysis comparing (a) a situation where the federation framework can rely on a federated identity management system and (b) a situation where such system is absent.

The analysis clearly shows that the operation of a FIDM system greatly impacts many relevant goals in a positive way. Nearly all federation user goals are positively influenced including the important goal *easy to use*.

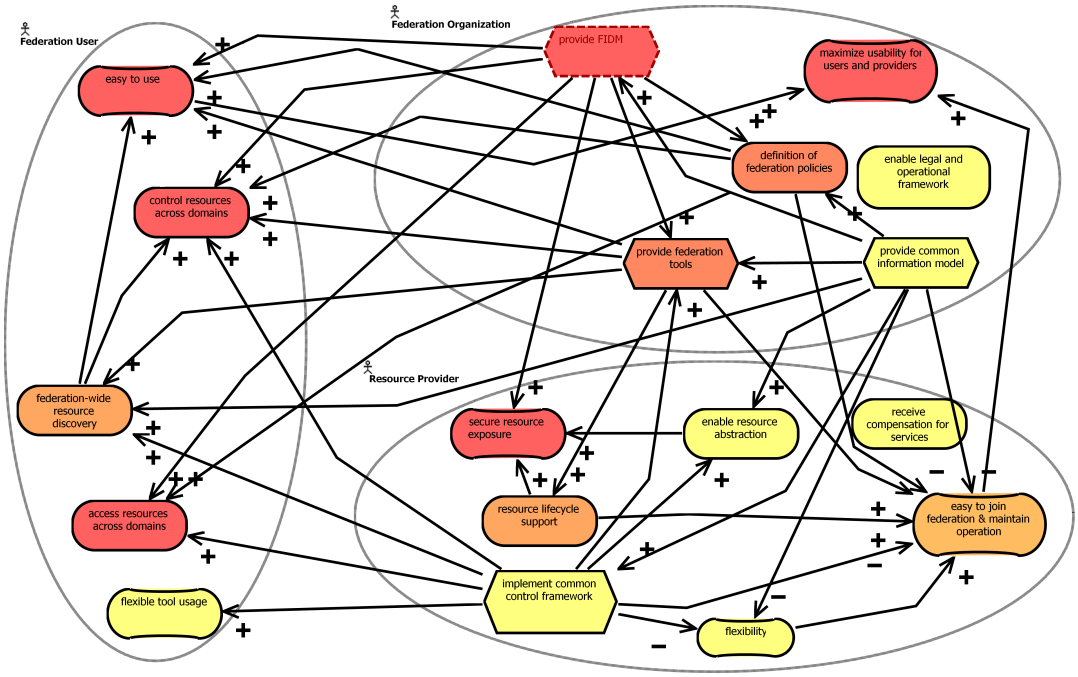
On the provider side, the goal *secure resource exposure* is directly impacted, while *resource life cycle support* and *easy to join federation and maintain operation* are indirectly influenced via the the federation organization goal *provide federation tools*.

As the two goals *easy to use* and *easy to join federation and maintain operation* directly impact the organization goal *maximize usability for users and providers*, the latter is positively impacted if *provide FIDM* is satisfied.

This leads to the conclusion that providing a FIDM system is beneficial for all stakeholders. Sub-figure (b) confirms this assessment as not a single goal that is impacted by the qualitative FIDM evaluation is satisfied in case a FIDM system is *not* provided.



(a) The federation relies on federated identity management mechanisms.



(b) The federation cannot rely on federated identity management mechanisms.

Figure 6.8 – GRL evaluation: the impact of identity management

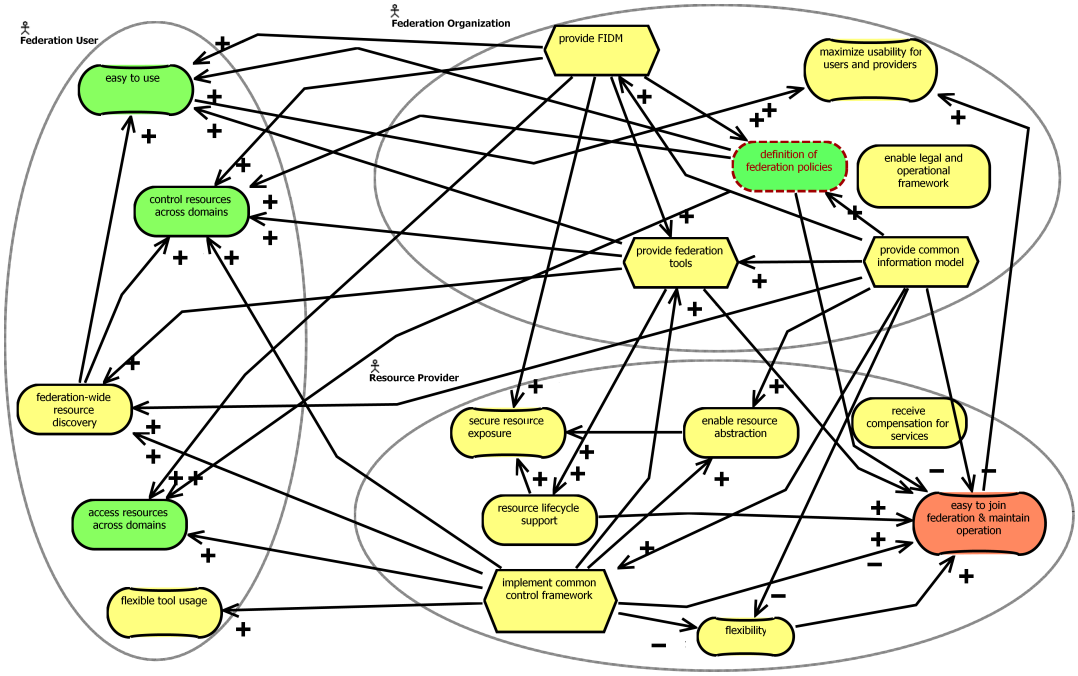
Policy Impact Analysis

This section analyses the impact of global federation policies and the mechanisms to define such. Figure 6.9 depicts the result of this analysis comparing (a) a situation where the federation can rely on global policies definition and (b) a situation where those are absent.

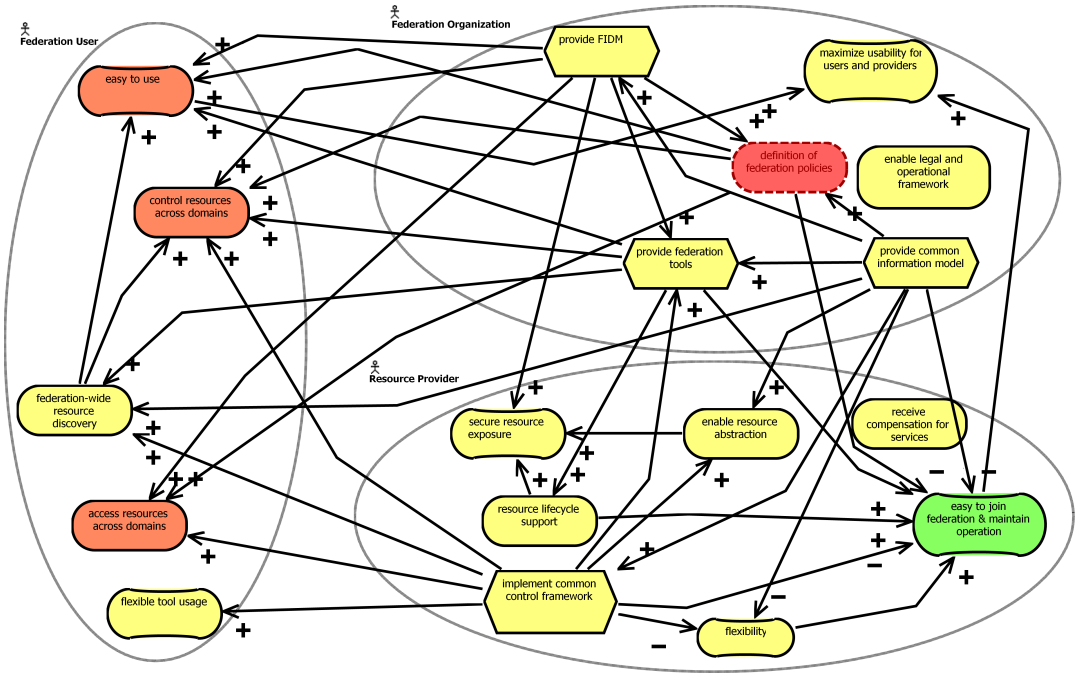
The evaluation reveals that policy related aspects seem to have the least impact on the overall federation framework. Fewer goals and tasks are impacted by the qualitative evaluation of the goal *definition of federation policies*. While the goal has a direct positive correlation with some user goals (e.g. control / access resources across domains), there is a direct negative correlation with the resource provider goal *easy to join federation and maintain operation*. The negative correlation is based on the fact that considerable efforts might be needed to analyze and define domain-specific resource provider policies that are in line with the global federation policies and the policy description format used to express them. Also, there is the possibility of conflicts between domain-specific (i.e. organization-specific) policies and global policies. Therefore, a federation-wide alignment of policies has a negative impact on the resource provider goal *easy to join federation and maintain operation*.

As shown in sub-figure (b) of figure 6.9, the absent of global policies makes the federation more attractive to resource providers while such situation makes the federation *less* attractive for the users. As the providers, in turn, depend to some extend on the users and their positive valuation, the decision has a number of side effects that especially need to be taken into account for a business analysis.

For example, there is a contribution of the user goal *easy to use* on the provider goal *receive compensation for services* which has not been modeled here to keep the analysis on the technical aspects and to avoid overloading the figures. However, the following causal relationship exists: if the users are content because the system works well for them, they will be willing to pay more for the services provided by the resource providers and the federation as a whole. Therefore, if global policies make the user's life easier, this is beneficial for the providers although they have to accept the additional burden of global policy definition and alignment. Such correlation can be observed for other goals as well, such as the *common control framework*. To model all of those side effects and especially those that mainly affect legal and business goals, would have overloaded the analysis and the resulting figures. Thus, the analysis has been restricted to technical aspects. Nevertheless, for a holistic view, such side effects must not be disregarded.



(a) The federation relies on global policies.



(b) The federation cannot rely on global policies.

Figure 6.9 – GRL evaluation: the policy impact

6.2 Federation Framework Design Decisions

Based on the analysis of the different stakeholder goals and their correlations presented in the previous section and based on the federation model and methods presented in chapter 4 and 5, this sections summarizes the design decisions for the federation framework instantiation. The decisions lead toward a high level system design that will be detailed further in section 6.3 where a detailed use case will be discussed. The federation framework instantiation will rely on the following:

A common control framework will be the basis for all resource control operations. Each domain shall expose a domain manager interface i/f M so that for domain A and domain B:

$$\text{i/f } M_A = \text{i/f } M_B \tag{6.1}$$

Such interface shall allow for controlling resources inside the domain from remote locations. The decision is based on the GRL evaluation shown in figure 6.7 which revealed a positive impact of a common control framework on most stakeholder goals. The control framework interface will have to be specified and then implemented across all domains participating in the federation. This will essentially implement the control framework method described in section 5.2.

Common identity management across the federation will by used to allow for stable authentication and authorization decisions at resource provider domain interfaces as well as federation services offered by the federation organization. In terms of the federation model, this decision leads to situation where for any two domains A and B that are part of the federation, trust relationships have been established and the following holds true:

$$IDspec_A = IDspec_B \tag{6.2}$$

This decision is based on the GRL evaluation shown in figure 6.8 which revealed that providing a FIDM system has a highly positive impact on the different stakeholder goals. It has to be noted that as the federation grows, the above conditions are unlikely to hold true for all participating domains. In such case, trust brokering mechanisms need to be used to establish trust relationships between the stakeholders as well as trust anchors, as described in section 4.3.1. However, for the federation framework instantiation described in the following, it is assumed that the federation can rely on trust relationships between the resource providers and the federation organization and that *IDspecs* have been aligned across the federation.

A common information model will provide accepted and agreed terminology across all federation stakeholders and system entities. Most importantly, this will allow the common control framework and any federation tools to rely on common resource descriptions so that for any two federated domains A and B:

$$Rspec_A = Rspec_B \tag{6.3}$$

The GRL evaluation depicted in figure 6.6 shows that a common model has some *negative* influence on some of the resource provider goals. However, the *positive* impact on many other important federation user and organization goals outweighs this drawback. It will

be important to support the resource providers with useful tools in order to ease the process of adapting to the common information model and common resource descriptions as much as possible.

Both global and local resource usage and access policies The analysis shown in figure 6.9 unveiled that this decision has less impact on the overall federation framework compared to the other decisions in terms of the total number of stakeholders goals that are influenced. Therefore, the design decision with respect to policies enables a hybrid approach: although the federation framework instantiation shall allow for the definition of global policies, it shall also be allowed for resource providers to define, evaluate, and enforce domain-specific policies. The reasoning behind this decisions it to facilitate a first set of prototype implementations without heavy restrictions in terms of policy mechanisms. Those shall be evaluated based on early feedback from resource providers with respect to how policies are defined and used to restrict resource usage and access.

The next section will introduce a high level use case that defines a number of federation system entities interacting with each other.

6.3 UCM-Based Use Case Analysis

6.3.1 Notation

While the GRL is primarily used to analyze the different stakeholder goals and their correlation in terms of different evaluation strategies, *use case maps (UCMs)* allow for a detailed analysis of functional system requirements and use cases. UCMs provide excellent means to communicate system message flows to developers and other important stakeholders by supporting the definition and visualization of scenario execution. This provides developers with the means to analyze the impact of alternative execution paths and communicate them graphically. Figure 6.10 ([125], p. 53) shows the UCM specification in terms of its abstract grammar.

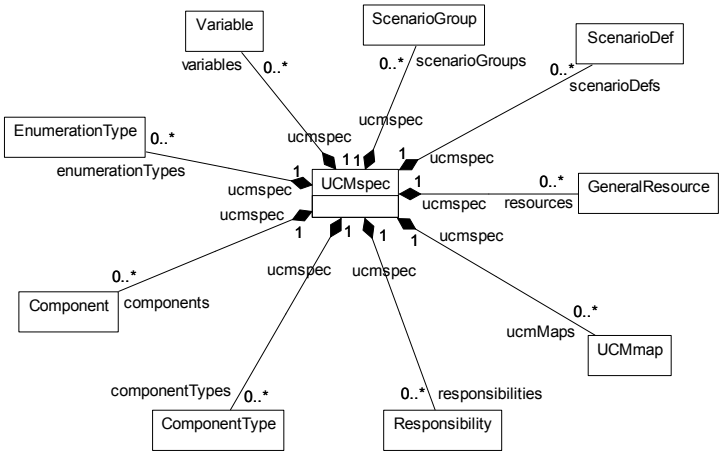


Figure 6.10 – The UCM abstract grammar

In UCMs, *paths* are used to show the system entities interactions. The concept is similar to UML sequence diagrams. The path may cross the boundaries of subsystems to show

the communication flow within the system (see for example figure 6.12). Subsystems are drawn as boxes and are containers for other component or *path nodes* such as *stubs* and *responsibilities*. Such path nodes are used to indicate some action. "Path nodes may express actions, alternatives (choice points and merge points), and concurrency (parallel branching points and synchronization points) as well as the beginning and end of scenarios." ([125], p. 59)

The path nodes primarily used in this thesis are *responsibilities*, *stubs*, and *orForks*. A full overview of the different path nodes is shown in figure 6.11 ([125], p. 55). A short description is given in the listing below.

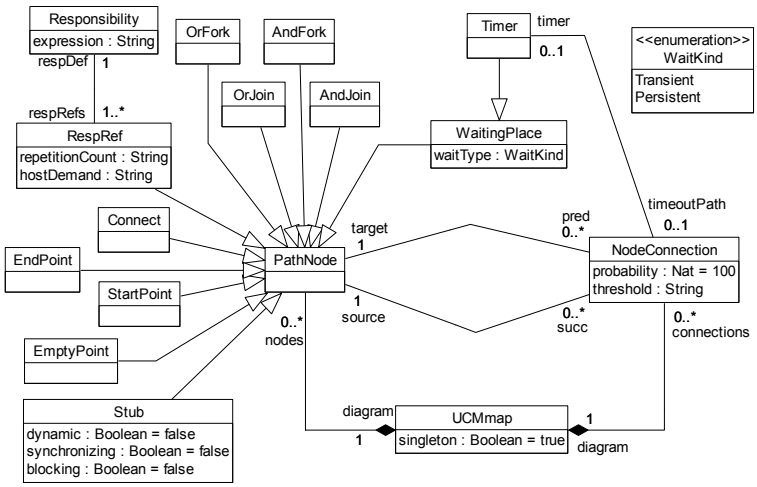


Figure 6.11 – The UCM path and path nodes abstract grammar

A responsibility "[...] is a reusable definition of a scenario activity representing something to be performed (operation, action, task, function, etc.) or in other words a step in the scenario." ([125], p. 63)

It defines actions "[...] either informally through its name or more formally with the help of its expression. The actions of the expression may make use of globally defined variables. When the traversal of a scenario path reaches a responsibility reference (RespRef), the expression defined in the associated responsibility definition is interpreted. This may change the values of variables in the global data model of the URN specification." ([125], p. 64)

A stub "[...] represents hierarchical structuring of UCM specifications through containment of plug-in maps. When the traversal of a UCM path reaches a stub, the traversal continues with the plug-in maps of the stubs. When the traversal reaches an end point on a plug-in map, the traversal may return to the map of the stub (i.e., the parent map) and proceed past the stub. The exact binding of the parent map to the plug-in map is specified with the help of PluginBindings, ComponentBindings [...], InBindings, and OutBindings." ([125], p. 81)

An orFork "[...] represents a choice point in the UCM specification with at least two alternative, outgoing branches. Each alternative, outgoing branch (i.e. NodeConnection)

has a condition. When arriving at the OR-fork during traversal of the UCM path, the conditions are evaluated. If exactly one condition evaluates to true, the alternative branch with that condition is chosen and the traversal continues along that branch. If no condition or more than one condition evaluates to true (non-determinism), then the traversal stops and a warning is generated [...]" ([125], p. 69).

The UCM notation is used in the following sections to outline a detailed use case and discuss the various subsystem interactions.

6.3.2 Overview

This use case analysis describes the system interactions that are necessary to design and provision a *virtual resource grouping*. It is assumed that a known user (i.e. a user that has previously registered with the federation organization) starts to design a new virtual resource grouping. Such grouping is referred to as a *Virtual Customer Testbed (VCT)*. Figure 6.12 shows the high level path through the entire *Teagle* system. Teagle represents a SET entity in terms of the federation model introduced in chapter 4 and provides several system components such as a portal, a request processor, an orchestration engine, and others. A description of the Teagle inner components is given below and in section 6.5.

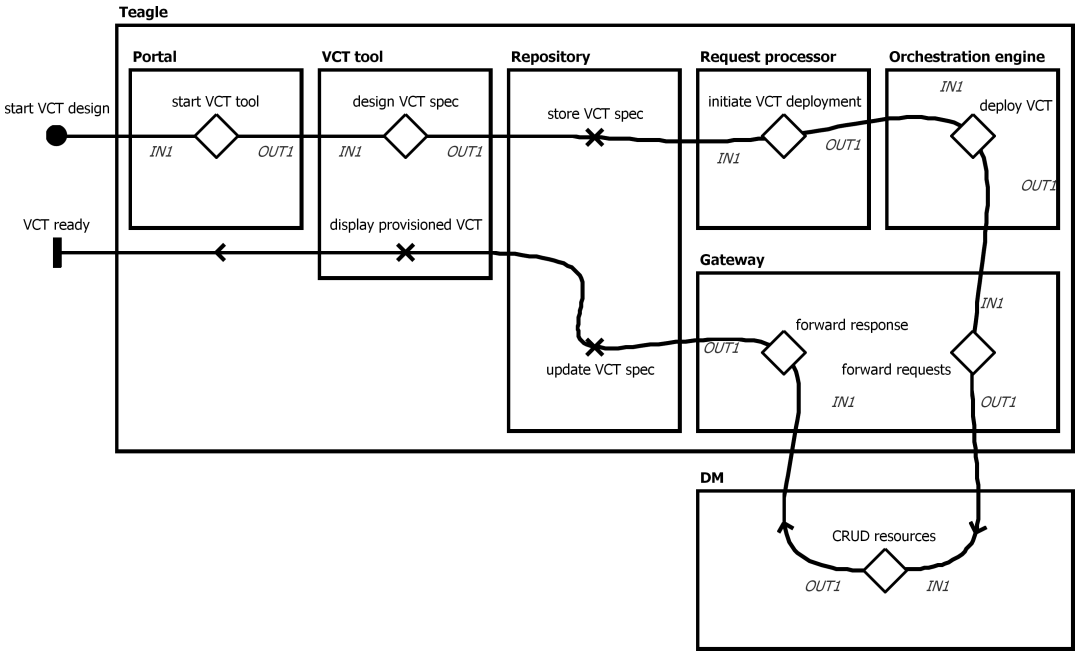


Figure 6.12 – A high level use case map showing the path for deriving a provisioned virtual resource grouping across multiple stubs associated to the different Teagle system components.

Virtual Customer Testbed A VCT is a virtual resource grouping in terms of the federation model introduced in chapter 4. It represents a logical container for interconnected distributed resource instances requested by a federation user. The VCT specification can either be derived manually or using tools such as the *VCT tool* that can be accessed via the *portal*.

Portal The portal is operated by the federation organization and offers a number of services to federation users and resource providers such as for example resource lookup, resource registration, domain manager (DM) registration, user account management, access to tools, policy definition, and others.

VCT tool This is a federation tool and is essentially a design environment that supports federation users in configuring and deploying a VCT (i.e. a virtual resource grouping). Although the specification of a VCT can be derived without such tools, providing graphical design environments that allow for resource selection via drag and drop, configuration wizards, and save/load features, can significantly simplify working with federated resources. The output of the VCT tool is a virtual resource grouping specification in terms of the federation model as defined on page 66. An example VCT specification derived using the VCT tool is given in annex A as part of the sections A.2 and A.4.

Repository The repository plays the role of a registry (i.e. the REG entity) in terms of the federation model described in chapter 4. It allows other Teagle framework components to store and retrieve data about resources, users, VCTs, etc.

Request processor This component handles incoming requests for VCTs. It processes them, performs federation level policy enforcement and forwards valid VCT specifications to the orchestration engine (OE).

Orchestration engine The main task of this component is to derive an instantiated set of resources (i.e. a deployed VCT) based on the user-designed virtual resource grouping specification received from the request processor. This requires to orchestrate the different resource provisioning tasks into a sequence of steps that can be executed (see also section 5.4 on resource orchestration). The critical part is to resolve the dependencies between resources and to derive the correct order (i.e. a workflow) of executable steps. Upon execution of such workflow, requests are send to all the domains that provide the resources selected by the user. Those requests contain the proper configuration instructions to perform the VCT deployment matching the user request.

Gateway The gateway acts as an ingress/egress point for communication between the other Teagle components (mainly the OE, the request processor (RP), and the registry) and the federated domains and their managers.

Domain manager This component represents the M entity in terms of the federation model. It manages all resources exposed by a specific domain and receives incoming resource configuration requests on interface T_1 (see figure 5.4). It communicates with the resources and enforces configurations requested by other federation entities on interface T_2 (see figure 5.4). Also, it implements security means to ensure secure resource exposure.

Resource Resources are located within a resource provider domain and can be controlled remotely using the resource abstraction capabilities provided by the domain managers.

Policy engine Although not shown in the high level overview, the policy engine (PE) plays an important role for federation policy evaluation and enforcement. For example, figure 6.15 shows some PE interactions during the VCT design process.

The following subsections describe—step by step—the entire path through the Teagle system and its subcomponents as depicted in figure 6.12.

6.3.3 The VCT Tool Startup Process

The user shall start to design a VCT by logging into the Teagle portal. Figure 6.13 shows the actions aggregated by the stub *start VCT tool* shown in figure 6.12. The *red line* shows the actions along the path of a successful startup of the VCT tool.

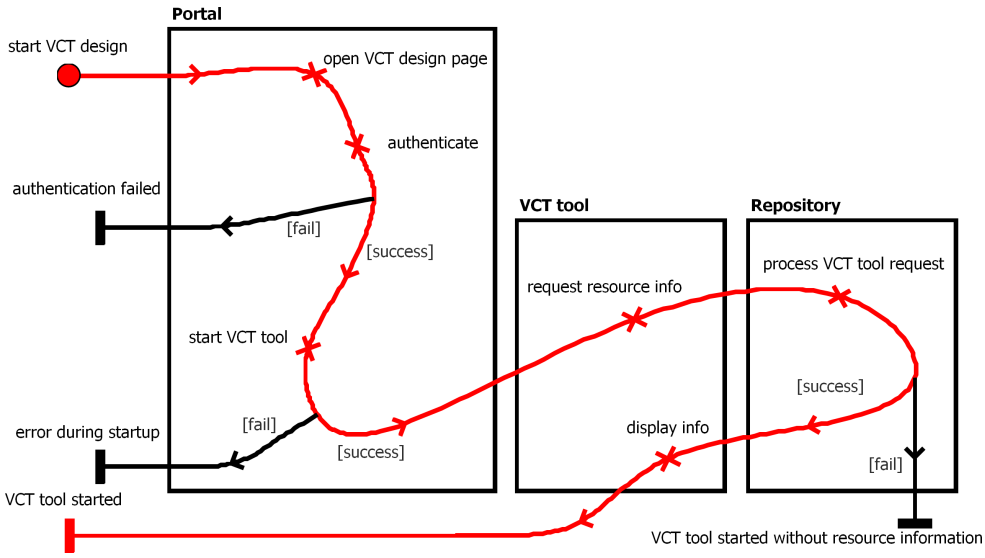


Figure 6.13 – The stub *start VCT tool*: the successful VCT tool startup path

A number of errors can occur along the path. For example, as shown in figure 6.14, the repository might be unavailable. In this case, the resource and VCT information stored and provided by the repository, cannot be loaded and displayed by the VCT tool. The other errors and activities are mostly self-explaining. The next subsection deals with the VCT design once the tool is up and running and has successfully loaded available resource information from the repository.

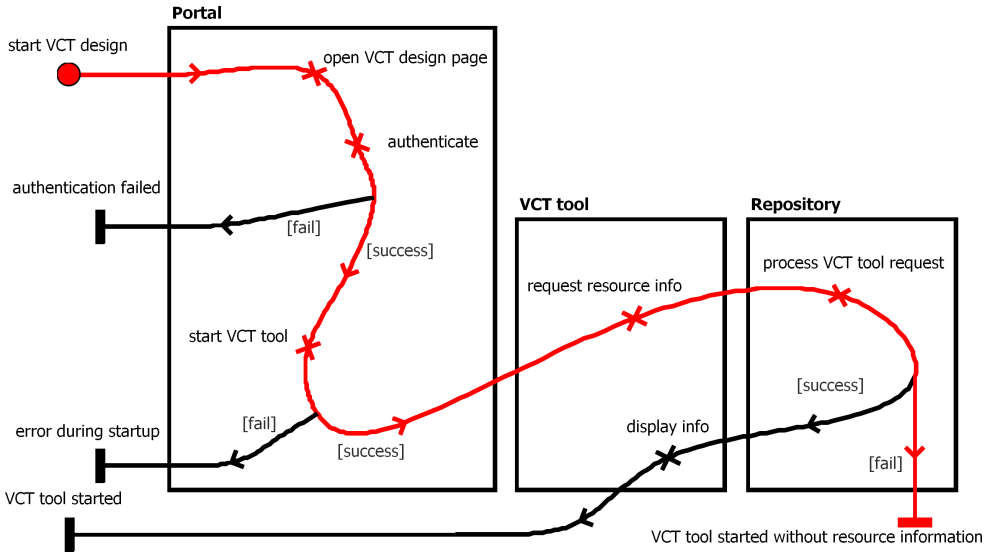


Figure 6.14 – The stub *start VCT tool*: error while loading resource information

6.3.4 The VCT Design Process

Figure 6.15 shows the system path for a successful VCT design process. Once the VCT tool has started and has successfully loaded the information received from the repository, the user shall be able to browse through the descriptions of available resources offered by the federation and select desired resources by dragging and dropping them onto a workbench. The resource descriptions shall be based on a common information model in line with the design decisions outlined in section 6.2.

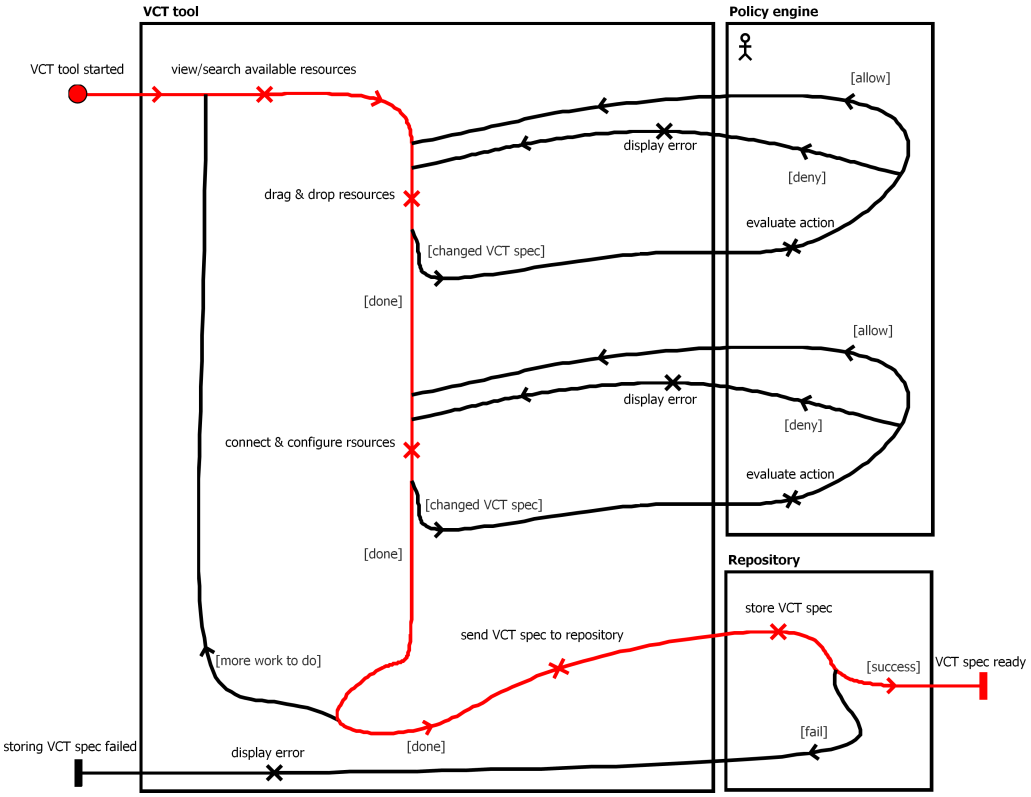


Figure 6.15 – The stub *design VCT spec*: successful virtual resource grouping specification design

Figure 6.15 shows two cycles that involve the Teagle policy engine (PE). The first cycle starts at the responsibility *drag & drop resources*. If new resources have been added to the workbench, the VCT tool shall consult the PE if any policy exists that denies such action. For example, specific users might not be allowed to use certain resources based on their affiliation. For a detailed description of the policies that can be evaluated and enforced by Teagle please refer to section 6.5.5.

A second very similar cycle is shown in figure 6.15 starting after the responsibility *connect & configure resources*. Here, the policy evaluation and enforcement shall ensure that only legitimate resource configurations are specified by the user. For example, meaningless parent-child relationships of specific resource types can be restricted. This allows to check the validity of VCTs on a very abstract level increasing the usability of the VCT tool and the resource provider flexibility in terms of policy definitions.

Once the user has passed the design/evaluation cycles, the VCT shall be stored in the repository. If this can be performed with success, the VCT specification process is completed

and the VCT is ready for deployment. The deployment will result in the provisioning of all resource instances specified during the VCT design process. The next subsection explains this step.

6.3.5 Initiating the VCT Deployment

Figure 6.16 shows the path along the initiation of the VCT deployment. The VCT tool shall enable the user to request the VCT deployment. Upon user requests the tool shall notify the request processor (RP) about any new VCT to be deployed. The RP shall fetch the VCT specification from the repository. Alternatively, if any other tool than the VCT tool is used, the notification may include a valid VCT specification that has been derived using the other tool. VCT specifications may also be derived manually. The VCT tool shall support loading such specifications. For XML schema definitions of a VCT specification please see annex F.

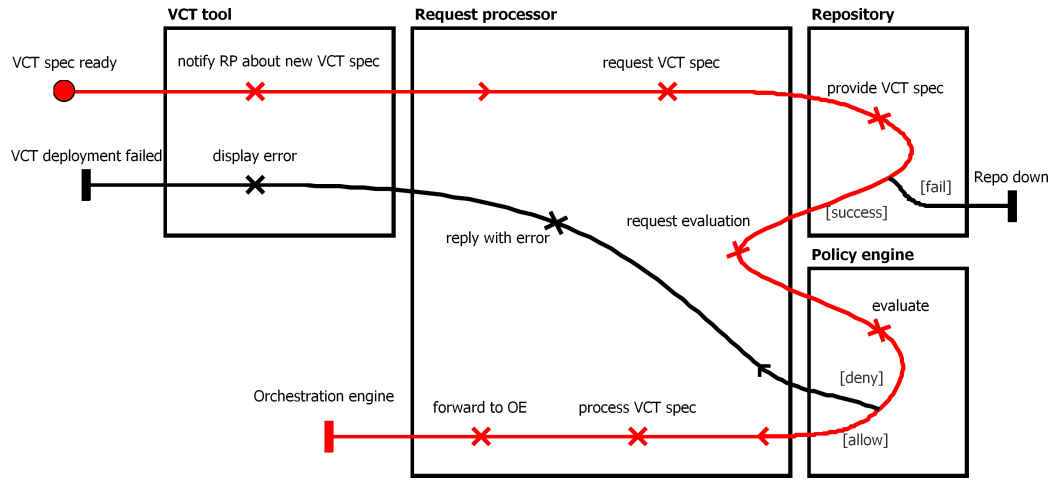


Figure 6.16 – The stub *initiate VCT deployment*: the request processor forwards the VCT specification to the orchestration engine.

Once the RP obtained the VCT specification, it shall request its evaluation by the policy engine. This step shall be performed even if the VCT specification has been derived using the VCT tool (and has already been evaluated by the PE during the VCT design process). This is due to the fact that the VCT tool will most likely run on the user's machine and can therefore not be trusted by the Teagle system. In case the VCT specification has been derived using the VCT tool, the PE should allow further processing by the RP.

At this stage, the RP shall further analyze and process the VCT specification. This shall be a hook for any higher tier federation logic. For example, the RP might select the domains where resource instances shall be provisioned in case the user has simply specified desired resource *types* but has not specified in which domain those should be instantiated. Such decisions might be taken by the RP on behalf of the user.

Finally, the RP shall forward the processed specification to the OE where a series of resource provisioning steps will be derived and executed.

6.3.6 The Resource Orchestration Process

Figure 6.17 shows a number of steps to derive an executable resource orchestration script (i.e. a workflow) that determines the order in which resource provisioning requests (CRUD requests) will be send to the federated domains. The orchestration engine shall implement a Teagle component that performs this task in line with the resource orchestration method described in 5.4.

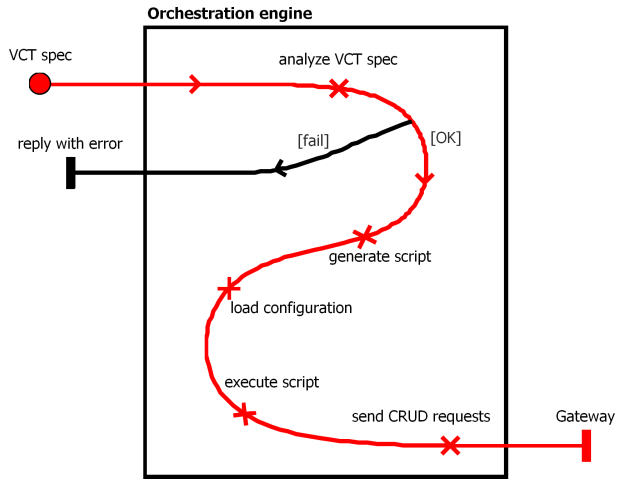


Figure 6.17 – The stub *deploy VCT*: resource orchestration workflow generation and execution

As a first step, the orchestration engine shall analyze the VCT specification validating its syntax and semantics. In case that this step terminates with success, a script shall be generated that resolves dependencies between resources. Generally, child resources depend on the parent resource and referencing resources depend on the resources that are being referenced. Those relationships determine the dependency graph and therefore the order of resource provisioning requests.

Next, the configuration shall be loaded. This is required to determine any *pre-existing resource instances* and replace any *design-time* resource identifiers with the *final* resource instance identifiers as defined in section 5.1.2.

Next, the workflow shall be executed using either a parallel or sequential mode as explained in section 5.4. This shall result in a series of CRUD requests that are sent to all the involved domains via reference point T_1. The requests shall be channeled via a gateway that implements security measures and acts as a single point of communication between the Teagle system and the federated domains on reference point T_1 (see page 79).

6.3.7 Interfacing With the Domain Manager Layer

Figure 6.18 shows the Teagle gateway subsystem and the series of steps that shall be executed before valid CRUD requests will leave the Teagle system boundaries. The red line marks the UCM scenario path of a successful VCT specification and deployment.

The gateway has been designed to keep all T_1 reference point communication technology and security relevant aspects in a single component. Conceptually, the OE could implement the steps performed by the gateway but a dedicated gateway component facilitates a clear

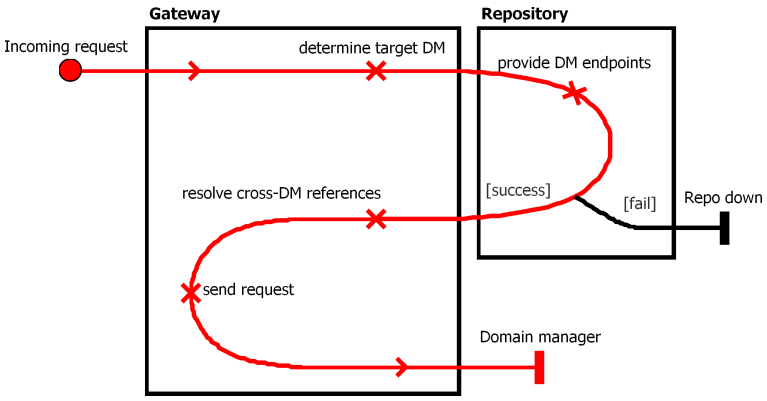


Figure 6.18 – The stub *forward requests*: the Teagle gateway shall act as single point of communication between the federation layer (the Teagle system) and the domain layer (the domain managers).

separation of concerns relieving the OE from dealing with T_1 reference point security and transmission technology implementation issues.

The gateway shall resolve any Teagle-internal domain handles and replace the DM identifiers with the T_1 interface DM endpoints. The DMs shall expose a T_1 interface based on Web Services (e.g. based on SOAP[3]) that shall be secured using SSL/TLS[4]. Furthermore, the gateway shall request DM client authentication. The federation organization shall operate a certification authority (CA) allowing to sign the certificates used on reference point T_1 and to establish trust relationships between the DM and gateway components.

At this stage, the complex VCT configuration involving multiple resources from different domains has been broken into atomic provisioning requests that are directed towards the responsible DMs. The requests will be send in the right order to allow for resources depending on other resources to be instantiated when the request arrives at the respective DM. Each provisioning request (i.e. CRUD request) carries the configuration to by enforced by the domain manager layer. See annex D for a specification of the T_1 interface.

6.3.8 Domain Layer Resource Management

Figure 6.19 shows the path along successful resource configuration on the domain layer. Using the resource abstraction method introduced in section 5.2, the configuration conveyed by each CRUD request arriving at a particular DM shall be applied to the respective resources of that domain.

Upon receiving a CRUD request, the DM shall analyze the request, determine which CRUD action to perform, determine which RA to contact, and forward the request to the RA. The format of this request and the format of the contained configuration information is not specified by the federation framework. The RA may physically reside on the DM machine or at a remote location inside the same domain and shall enforce the requested configuration on reference point T_2 (see figure 5.4).

The transmission technology to convey such information and the specific resource configuration execution processes, depend on the respective resource and the RA implementation.

[3]SOAP Version 1.2 Part 1: Messaging Framework (Second Edition), W3C Recommendation, [126]
[4]RFC 5246: The Transport Layer Security (TLS) Protocol Version 1.2, [127]

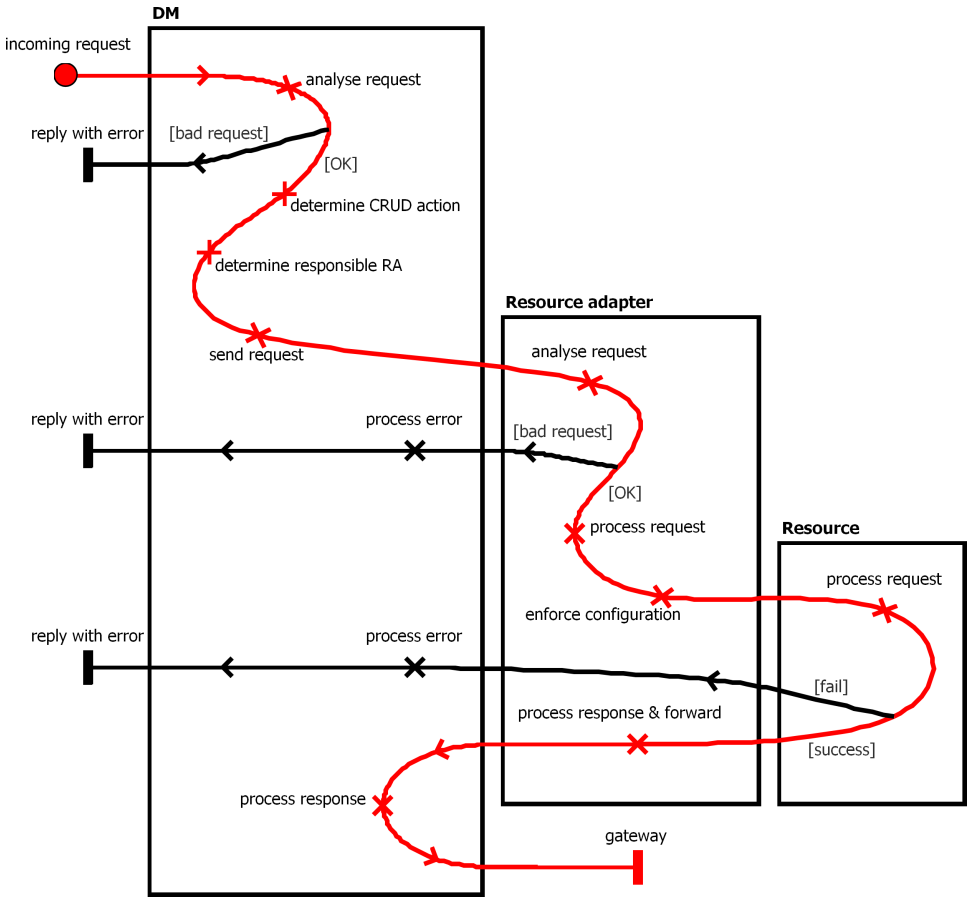


Figure 6.19 – The stub *CRUD resources*: successful domain layer resource configuration results in interactions between the domain manager, the resource adapter, and the resource.

The federation framework shall not specify any of such details in order to maximize the range of supported resource types and the resource provider flexibility. Please refer to section 6.6.2 for concrete examples.

The result of the configuration action is reported back to the DM. In a successful case, the DM will update the domain registry with the new resource status and report the new resource configuration back to the Teagle layer (i.e. to the Teagle gateway). In case of errors, the respective messages and error codes shall be reported back along the entire chain of components. Also, the DM is responsible for enforcing the original resource status. This only applies to domain-internal "atomic" CRUD actions. Cross-domain dependencies and VCT configuration failures are handled by the Teagle layer using a rollback function as described in section 5.4.

This section covered a detailed system use case. The next sections will outline the architecture and implementation details of the federation framework system (i.e. the design artifact *instantiation*) based on the GRL requirements analysis introduced in section 6.1, the resulting design decisions discussed in section 6.2, and the UCM-based use case analysis presented in this section.

6.4 System Overview and Reference Points

Figure 6.20 (adapted from [115], p. 3) shows all reference points of the federation framework instantiation.

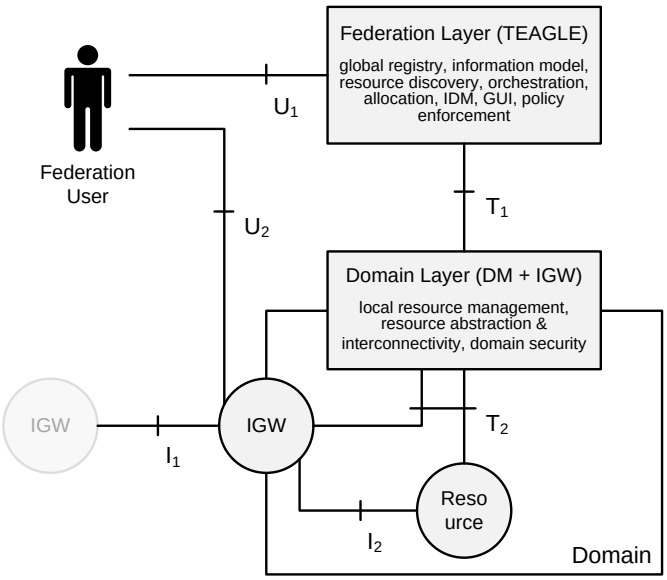


Figure 6.20 – Framework instantiation overview and reference points

Reference point U_1 enables the user to interact with the Teagle framework. Teagle represents a SET entity in terms of the federation model and allows the user to manage his account and identity related information, discover resources and obtain detailed information on them, define Virtual Customer Testbeds (VCTs), and request the resource allocation according to the VCT specification. Such functions are provided by Teagle components such as the VCT tool and the Teagle portal. For resource providers, U_1 enables additional functions such as to register resources, to register domains, and to define resource usage and access policies.

Reference point U_2 allows a user to access a successfully deployed VCT. According to the resource interconnection method introduced in section 5.5, an interconnection gateway enables the user to *"dial into"* his VCT. Technically, this is done using a VPN client or VPN aware router to connect to the virtual network connecting the resource of a specific VCT. In case of public resources, user can directly address them without the need of VPN technology. For many resources, Secure Shell (SSH)[5] access can be used. However, other access methods and technologies can be required for specific resources.

Reference point T_1 has already been mentioned a couple of times (see for example section 5.2 and annex D). It allows: to create resource instances of a specific configuration, to obtain information on existing resource instances, to update such instances, and to delete them. Furthermore, in the opposite direction, domain managers can notify the

[5]The IETF WG Secure Shell (secsh) standardized the SSH protocol in a series of request for comments (RFC): http://datatracker.ietf.org/wg/secsh.

Teagle layer about new resource types and instances, report VCT provisioning results, and provide status information.

Reference point $\mathbf{T_2}$ represents the communication between the domain manager and the resource. To enable controlling highly heterogeneous resources, the concept of resource adapters (RAs) is used as proposed by the federation method described in section 5.2.2. RAs allow to translates the normative T_1 instructions into resource specific commands. Several RAs have been implemented that support standard management protocols and techniques such as OCCI [35] and the Service Provisioning Markup Language (SPML) [128] as well as remote interaction standards such as SSH[5] and Extensible Markup Language Remote Procedure Call (XML-RPC)[6].

Reference point $\mathbf{I_1}$ defines "[...] a gateway-to-gateway connection, a data path between two [domains]. Communication between the gateways uses the data plane only, there is no control message exchange since the secure tunneling methods are stateless and use pre-shared secrets. [...] Using the I1 interface, all gateways can form a meshed [domain] interconnection network that is dynamically created without exchange or stateful control." ([24], p. 105)

Reference point $\mathbf{I_2}$ enables "an isolated (e.g. by Virtual Local Area Network (VLAN) techniques) link layer access to all associated testbed components and Quality of Service (QoS) based traffic shaping for traffic that is routed from the testbed component through the gateway. To achieve a secure and isolated connection between a gateway and testbed components, the latter need to be directly connected to the gateway or link layer by a switching device that honors link layer isolation methods like IEEE 802.1q (VLAN). As long as the associated testbed component is assigned to a VCT, the associated gateway shall have exclusive link layer access for reliability and security reasons." ([24], p. 105-106)

In the following sections 6.5 and 6.6, the architecture and implementation of the federation system is discussed on the two conceptual layers: (1) the federation layer and (2) the domain layer.

6.5 The Federation Layer

The federation layer is represented and implemented by the *Teagle* framework. Teagle represents a SET entity in terms of the federation model introduced in section 4.1 and implements:

- a web portal that used the repository interface (REP, see below) to enable registering and listing resources and DMs, displaying VCT instances, and managing policies. Also, the portal allows to start the VCT tool and provides general information, news, and tutorials around the federation framework.

- a model-based repository that consists of several registries (users, resources, configurations, etc.).

[6] http://www.xmlrpc.com/spec

- a VCT design environment (i.e. the VCT tool) that enables the setup and configuration of VCTs using available federation resources.
- a policy engine (PE) that "[...] allows the evaluation of policies that resource providers can define. The engine also allows for global federation policies." ([1], p. 169)
- a request processor (RP) that exposes an API that can be called by the VCT tool or other tools to request the deployment of a VCT.
- an orchestration engine (OE) that "[...] generates an executable workflow for resource provisioning. The engine receives a VCT definition from the request processor." ([1], p. 169)
- a gateway that "[...] handles bidirectional communication between domain managers [...] and Teagle internal entities." ([1], p. 169)

6.5.1 Teagle Architecture

Figure 6.21 (adapted from [1], p. 170) shows the Teagle architecture including several subcomponents and internal reference points.

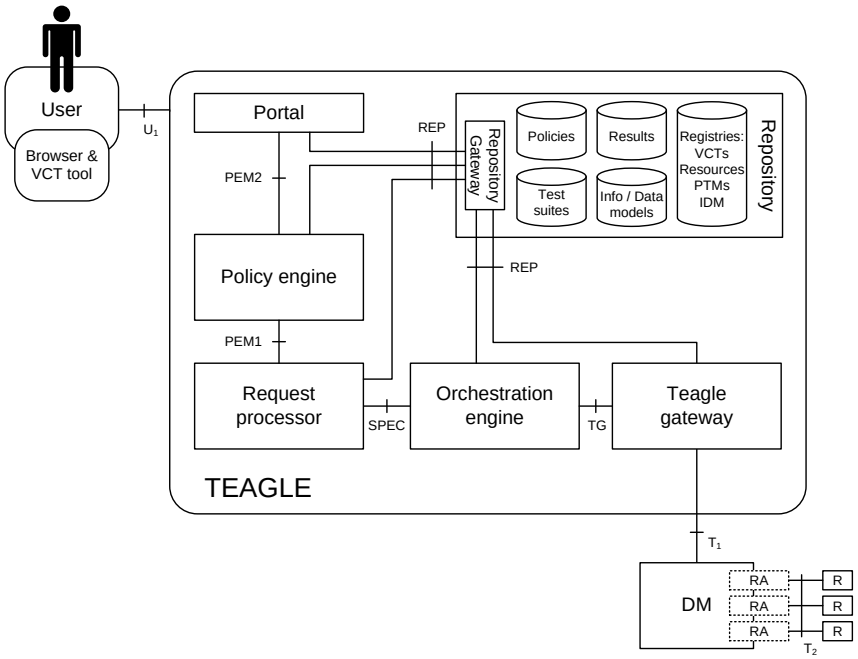


Figure 6.21 – The Teagle system

Reference point REP defines the communication with the Teagle repository. Here, information about resource types and instances, virtual resource groupings, bookings, users, etc. can be obtained by the other Teagle components as XML documents. The sections 6.5.3 and A.1 outline further details and examples on this reference point, its implementation, and message flows.

Reference point PEM1 is used for policy evaluation. The requester may ask for the evaluation of *organization*, *resource*, or *user* policies. The policy engine will reply with an `ALLOW` or `DENY` response. More information on this reference point and the policy evaluation process is given in the sections 6.5.5 and B.2.

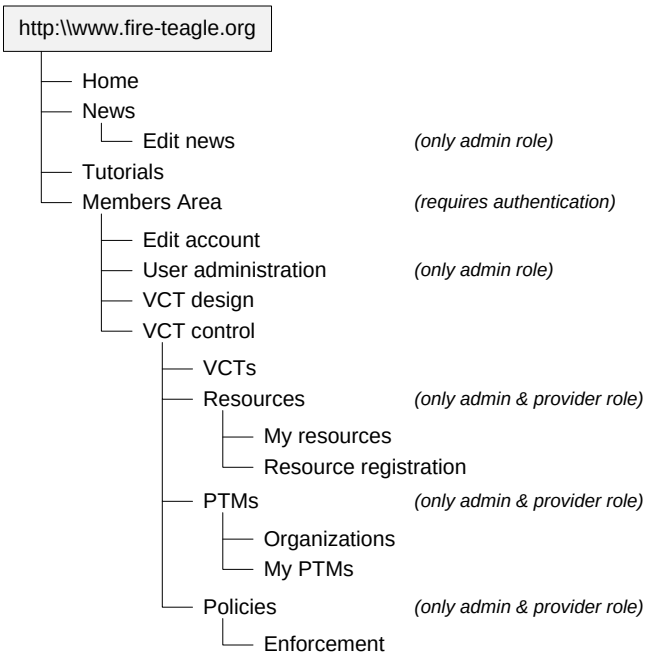
Reference point PEM2 defines a policy management interface. In Teagle it is used by the portal to manage global federation and provider-defined policies. It is introduced in detail in section 6.5.5. Annex B gives an example.

Reference point SPEC allows to request the orchestration and provisioning of a virtual resource grouping. The according interface exposed by the OE is currently only used by the request processor. The orchestration and provisioning result is reported back as soon as the result is available. More on this reference point and its implementation can be found in section 6.5.7 and A.4.

Reference point TG allows to communicate resource provisioning requests to the federated domains. This is used by the OE during the virtual resource grouping deployment process by calling the Teagle Gateway (TGW) interface exposed on reference point TG. In turn, the TGW may notify the requester about resource provisioning events. More details are provided by the sections 6.5.7 and 6.5.8.

6.5.2 Portal

The Teagle portal acts as an entry point for users to interact with the federation. It provides services to federation users and resource providers. The following listing shows a sitemap of the portal which is available online[7].



[7]Teagle portal website: `http:\www.fire-teagle.org`

The public information includes news and tutorials pages. To gain access to the *members area*, users have to create an account and accept the federation organization terms and conditions or use the credentials linked with an existing account. The portal implements three distinct roles:

- a *user* role,
- an *admin* role, and
- a *provider* role.

Users can manage their accounts and identities used within the federation via the portal. All user and identity related data such as personas, attributes, and credentials are stored in the Teagle repository according to the common information model implemented by the repository. This allows the different Teagle components to retrieve relevant information when needed. As an example consider the policy engine that needs to obtain the organization that a specific user is affiliated with. Only Teagle-internal components are authorized to retrieve such information.

After the user has been authenticated at the Teagle portal, the functions that are available in the members area are adapted depending on the user's role. Some pages are exclusively accessible by an administrator, e.g. the user and news administration pages. The VCT design page allows to launch the VCT tool as a Java Web Start application using the Java™ Network Launching Protocol (JNLP)[8]. The VCT control section allows to list existing VCTs and resources (all roles), register new resource types and *pre-defined* resource instances (only administrators and resource providers), register new organizations and according domain managers (only administrators and resource providers), and define resource usage policies (only administrators and resource providers).

The resource registration page implements a *description wizard* that guides the user through the definition of new resource types and pre-defined instances. Pre-defined instances are resource instances that are of an existing *resource type* and that are usually provided in a *pre-defined*, basic configuration. For example, physical machines or specific devices, such as sensor nodes, are typically provided as instances with a given configuration (e.g. CPU, memory, network parameters) that can not be changed by the user. Still, such instances can be used in VCTs and allow for resource relationships (e.g. to be referenced by other resource instances or act as a parent resource, see also section 5.3). The resource description wizard also implements *tooltips* that pop up automatically upon selection of input fields increasing the usability for unexperienced users.

The domain manager registration page allows the registration of new domains and the resource types supported by this domain. This allows Teagle to select domains during the VCT deployment process on behalf of the user. Users can always specify the domain to deploy a specific resource. However, in some cases, users are not interested in the specific location of a given resource. In this case, Teagle can select the domain based on different criteria (overall VCT performance, resource pricing, etc.). Such higher tier federation logic is possible due to the federation model and methods (e.g. common resource description, common control framework) defined in chapter 4 and 5 providing the basis for the federation framework instantiation.

[8] Java™ Network Launching Protocol & API Specification (JSR-56), version 6.0.18, Sun Microsystems, Inc., [129]

The policy related pages allow the definition and display of organization, resource, and user policy rules (see also section 6.5.5). The policy enforcement overview page displays recent policy evaluations including the originator, target, operation, time, and decision. All information processed by the portal pages (e.g. resources, domain managers, policies, etc.) is structured using the model implemented by the Teagle repository (see section 6.5.3). Most portal pages simply provide a user fronted to the federation data held by the repository.

With regard to the technologies used for the implementation, the portal builds upon Java Server Pages (JSP)[9] hosted by an Apache Tomcat container[10]. Additionally, a MySQL[11] database allows to store news items and other textual website elements.

6.5.3 Repository

The Teagle repository is a critical component in the Teagle design. It serves as a central information hub and storage facility for many other Teagle components. The information stored and provided upon request is structured according to an information model as proposed by the method introduced in section 5.1. The basis for this model provided an existing model, the *Directory Enabled Networks - next generation (DEN-ng)* information model [131].

"The DEN-ng information model [...] that is rooted in the area of network management and autonomic networking was used and extended to allow the description of resources as managed entities, their life cycle, as well as associated policies. In terms of DEN-ng, resource entities provide a service and have a certain configuration attached that can be defined and altered using the federation tools that are exposed to the experimenter via Teagle. Resources can exist as physical or logical resources where resource providers can define a list of resource instances as specific subtypes based on the model to represent their federation offerings." ([1], p. 171)

Annex C shows the most relevant classes of the information model. A number of root classes such as *Application*, *PhysicalResource*, and *LogicalResource* are defined by the original model and have been kept to enable interoperability with the DEN-ng core model. An initial prototype of the repository was implemented based on the *eXist* open source XML database[12] that allowed to "issue select/update queries directly on the database, using XML query languages (currently XPath[13] and XUpdate[14])." ([101], p. 83) The most recent Teagle repository implementation "[...] has been realized as a number of applications running as contexts on an application server. Each application has its own data storage facility and exposes an HTTP-based RESTful interface with a number of Representation State Transfer (REST) resources. The repository only deals with storage and retrieval of data on behalf of client applications. This allows the architectural entities that collectively represent the Teagle framework, to develop independently of the repository but to rely on a common data model." ([1], p. 171)

The RESTful interfaces of the repository exposed on reference point REP are available

[9] JSR-000245 JavaServer Pages™ Specification, version 2.2, Sun Microsystems, Inc., [130]
[10] Apache Tomcat website: http://tomcat.apache.org
[11] MySQL website: http://www.mysql.de/
[12] The eXist-db open source native XML database project website: http://exist.sourceforge.net
[13] The XML Path Language (XPath) 2.0, W3C Recommendation, [132]
[14] At the time of writing, the XUpdate language specification as published by the XML:DB-Project [133] is somewhat outdated and is expected to be replaced by the XQuery Update Facility, a W3C Recommendation published in [134].

online[15]. Four Hypertext Transfer Protocol (HTTP)[16] methods can be used to communicate with the repository on reference point REP (see figure 6.21): POST, GET, PUT, and DELETE. Table 6.1 shows the mapping of those methods to the repository operations.

Table 6.1 – Mapping of Teagle repository operations to HTTP methods

Repository Operation	HTTP Method
Create a repository item (e.g. the VCT tool requests to store a resource configuration that is part of a new VCT definition).	POST
Read an existing repository item (e.g. the request processor reads a resource configuration that is part of an existing VCT specification).	GET
Update an existing repository item (e.g. the RP updates an existing resource specification by replacing a design time resource identifier with a final identifier value).	PUT
Delete an existing resource (e.g. a resource type specification is deleted by a resource provider using the Teagle portal resource management page).	DELETE

For example, to update the VCT specification with `id="61"`, the VCT tool might send the following HTTP request (consisting of the request line, several header fields, and the entity body):

Listing 6.1 – Update request

```
PUT /repository/rest/vct/61 HTTP/1.1
Authorization: Basic dGVzdHVzZXI6dGVzdA==
Content-Type: text/xml
User-Agent: Java/1.6.0_24
Host: 193.175.132.210:8080
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Connection: keep-alive

<?xml version="1.0" encoding="UTF-8"?>
<vctInstance>
   <shared>false</shared>
   <commonName>seb_diss</commonName>
   <description></description>
   <user>2</user>
   <state.id>8</state.id>
   <hasConnections>3</hasConnections>
   <hasConnections>2</hasConnections>
   <providesResources>144</providesResources>
   <providesResources>149</providesResources>
```

[15]The repository URL is constructed as follows: http://'host-name':'port'/repository/rest/'rest-resource', e.g. `http://repos.pii.tssg.org:8080/repository/rest/resourceSpec`. To access the repository interface, a valid Teagle account is required that can be obtain via the Teagle website `http://www.fire-teagle.org`

[16]RFC 2616: Hypertext Transfer Protocol – HTTP/1.1, [135]

```
20    <providesResources>147</providesResources>
21    <providesResources>145</providesResources>
22 </vctInstance>
```

The request line determines the addressed resource (`/vct/61`) and the operation to be performed (`PUT`). HTTP version 1.1 as defined by [135] is to be used along with a `Basic` authentication scheme as defined by [136]. The `User-Agent` header field indicates that a Java application issued the request. The latest VCT tool version is implemented in Java. The request was traced on an in-house Teagle installation at Fraunhofer FOKUS. Therefore, the `Host` field carries a FOKUS network address that cannot be reached from outside the FOKUS network (`193.175.132.210`) and a non standard HTTP port (`8080`). The XML document carried in the message body part of the request is compliant with the VCT instance schema defined for reference point REP as shown by listing F.13. Also, from the model shown in annex C, it becomes clear that the information carried in the message body can be understood and interpreted on the repository side as the elements map the classes and relations defined by the information model. This demonstrates the power of a common information model as all Teagle entities that implement the model can rely on a common understanding. The repository might respond to the request with the following answer:

Listing 6.2 – Update response

```
1  HTTP/1.1 200 OK
2  Server: Apache-Coyote/1.1
3  Content-Type: text/xml;charset=utf-8
4  Transfer-Encoding: chunked
5  Date: Thu, 05 May 2011 14:28:36 GMT
6
7  <?xml version="1.0" encoding="UTF-8"?>
8  <vct id="61">
9     <commonName>seb_diss</commonName>
10    <description></description>
11    <hasBookings />
12    <hasConnections>
13       <connection id="2" />
14       <connection id="3" />
15    </hasConnections>
16    <providesResources>
17       <resourceInstance id="149" />
18       <resourceInstance id="147" />
19       <resourceInstance id="145" />
20       <resourceInstance id="144" />
21    </providesResources>
22    <shared>false</shared>
23    <state id="8" />
24    <user id="2" />
25 </vct>
```

The HTTP entity body `Content-type` must be `text/xml` with a `UTF-8` character encoding. Generally, all information passed to the repository in the message body has to be compliant with the REP reference point XML schemata given by annex F. Also, all informa-

tion received from the repository can be expected to be compliant to those schemata. This ensures that all information regarding federated resources can be equally interpreted across all Teagle components, including domain managers and resource adapters. The latest repository implementation driven by the Panlab project is available as open source under Apache v2.0 license[17].

6.5.4 VCT Tool

The VCT tool can be launched by registered users from the Teagle portal as a Java application using JNLP. Figure 6.22 shows a screenshot of the VCT design view of the tool. During the

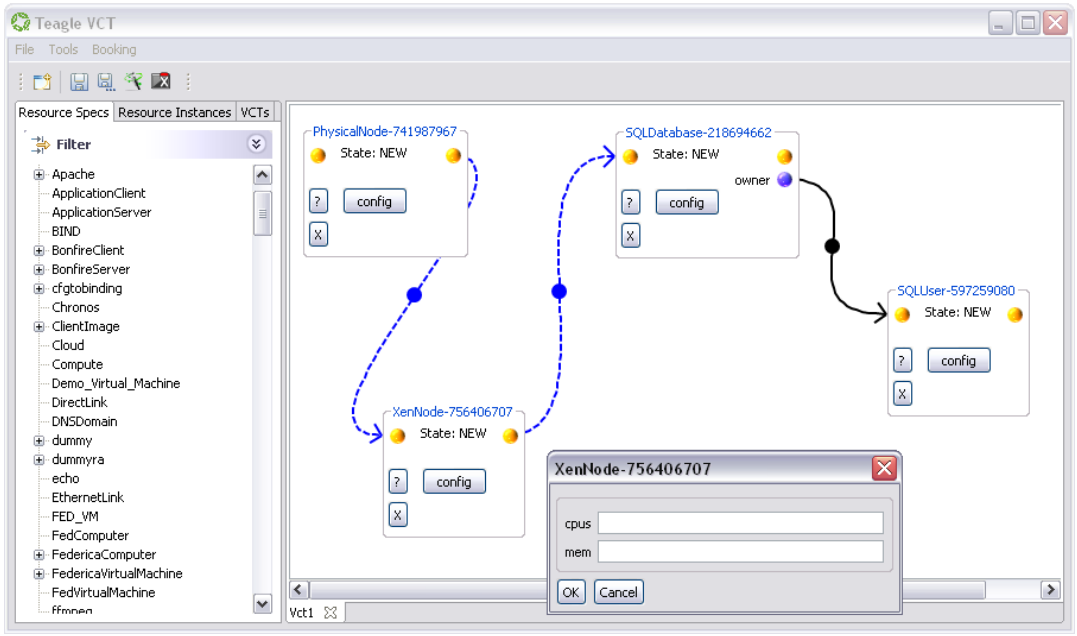


Figure 6.22 – VCT design using the VCT tool – the opened resource configuration dialog for the resource `XenNode-756406707` (a virtual machine) allows to specify the number of CPUs and the amount of memory to be available to this virtual machine.

VCT startup process the tool consults the Teagle repository and retrieves available resources and stored VCTs from previous sessions. Available resource can be dragged and dropped from the list of resources on the left onto a workbench. Here, resources can be interconnected defining resource relationships as proposed by the method described in section 5.3. In addition, the tool provides the following functions:

Resource Specs This tab allows to view available resource *types*. A *filter* option allows to reduce the total number of displayed resources. A resource type can be offered by several domains. In this case, a specific deployment domain can be selected from the available options. If such selection is omitted, Teagle will choose an appropriate domain to deploy the resource instance derived from the chosen resource type.

[17]The Teagle repository source code SVN location: `http://svn.panlab.net/PII/repos/Software/sources/Repository/CoreRepository`

Resource Instances This tab displays available resource instances to be used in the VCT design. This can be either resource instances created previously by this user as part of another VCT specification, or pre-defined resource instances that have been made available by a resource provider to be used in a given configuration. An example for a resource instance that is typically used across multiple VCTs, is a virtual appliance (i.e. a virtual machine image). Images are often re-used across several VCT instances, for example to test the performance of an application based on different physical and virtual machine configurations.

VCTs This tab lists the available VCTs that can be loaded on the workbench, for example to re-configure the VCT or re-use selected resource instances for a new VCT specification.

File Menu

- New VCT – opens a new VCT workbench tab
- Save – stores the currently opened VCT specification on the repository
- Save as – stores the currently opened VCT specification on the repository prompting for another VCT name
- Exit – closes the VCT tool

Tools Menu

- Console – opens a console for debugging purposes
- Preferences – allows to specify the credentials that will be used to communicate with the Teagle platform a well as the repository Uniform Resource Locator (URL) to be used

Booking Menu

- Book – starts the booking of the currently opened VCT, saving the VCT is required before the booking can be initiated.
- Export to XML – allows to save the VCT specification as an XML document
- Refresh booking state – updates the booking status with the latest information obtained from the OE
- Show booking state – shows the latest booking status
- Clone all resources as unprovisioned – allows to *clone* all resources of the currently opened VCT resulting in a new set of resource instances that have the same configuration

The following subsections explain how the VCT tool can be used to define and provision a valid VCT.

Setting Configuration Parameters

Figure 6.22 shows a typical configuration dialog. This dialog can be opened by the user by clicking the *config* button of a resource. The dialog lists the available configuration parameters to be set by the user. For example, the user may specify the number of CPUs and the

amount of memory to be available to a XEN node (a virtual machine controlled by a XEN hypervisor[18]). Invalid parameter values are rejected during the provisioning of a VCT[19].

Determine Resource Relationships

The tool allows to define resource relationships by drawing a line between two resources. A dotted line (see figure 6.22) represents a *parent-child* relationship as defined in section 5.3 that is also referred to as a *containment* relationship. For example, the dotted between the resource `PhysicalNode-741987967` and the resource `XenNode-756406707` defines a parent-child relationship between those resources with the physical node being the parent (i.e. the container). During the provisioning phase, this information is used by the OE to derive a valid order of provisioning steps.

A solid line between two resources represents a *configuration reference* as introduced in section 5.3. Figure 6.22 shows a configuration reference between the resource `SQLDatabase-218694662` and the resource `SQLUser-597259080`. This specifies the database to be configured with the given database user. On the domain manager layer this means that the resource adapter (RA) responsible for `SQLDatabase-218694662` will obtain a reference to the resource `SQLUser-597259080` in order to query the user configuration (in this case a user name) and update/configure the database accordingly.

Policy Evaluation

During the VCT design, policy evaluation is performed whenever a new resource has been placed on the workbench or has been connected to another resource. This allows to restrict the design to meaningful resource combinations and enable VCT validity checks on different abstraction levels. The VCT tool requests policy evaluation from the Teagle policy engine (PE) whenever required and enforces the PE decision. Section 6.5.5 gives detailed insights into the policy evaluation process handled by the Teagle PE.

Booking a VCT

Once the user has finished and saved the VCT design, the *booking* feature may be used by selecting the respective menu entry/button to initiate the VCT deployment. The tool presents a booking summary, listing all the resources that are part of this VCT, their configuration, and the associated cost. It is assumed that resource providers receive a compensation for providing the resources. However, this depends on the operational model followed by the federation (club good vs. public good, fully commercial vs. public funding, etc.). Functions such as accounting, charging, and billing would have to be revised and extended once the prototypes are to be used in a commercial context, because most of the legal and contractual aspects of the booking process have only been investigated superficially.

Also, before the booking can be completed, several *operational steps* should be performed involving representatives from the federation organization and the resource providers. This is not the main focus of this work but it is mentioned here to remind the reader of the non-technical processes required to establish a functional federation. The Panlab project (see section 3.8.2 and 7.1.1) has done some work regarding legal and operational requirements

[18] XEN community website: http://xen.org

[19] Future versions of this tool should implement additional usability features to ease the parameter specification during the VCT design time.

besides the technical aspects. Essentially, the user has to sign a resource usage contract with the federation organization that acts on behalf of the resource providers. Such processes are likely to require human intervention. Figure 6.23[20] shows the legal and operational processes that are needed to create a booking after the VCT design has been completed using the Teagle VCT tool.

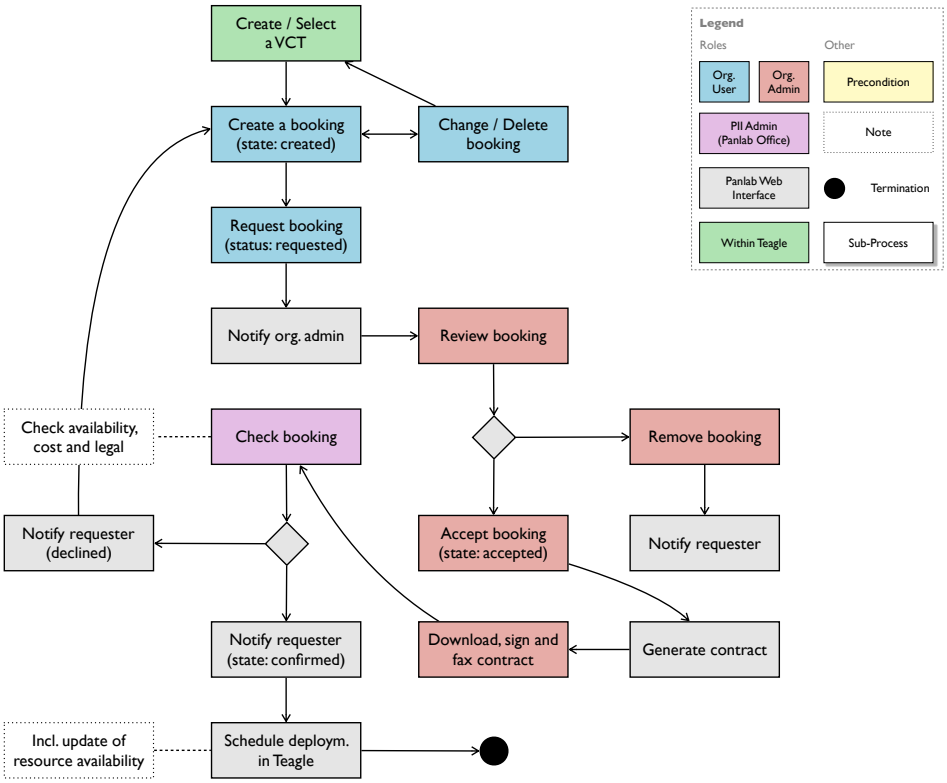


Figure 6.23 – The booking process from an operational and legal point of view

Once the operational and legal procedures have been completed and the booking summary is accepted in the VCT tool, the VCT specification (that has already been stored in the repository upon saving) is assigned the status "booking initiated" and the request processor is notified that a new VCT specification is ready for deployment. The request processor will then initiate the resource provisioning.

The latest version of the VCT tool implementation is available as open source under an Apache v2.0 license[21].

6.5.5 Policy Engine

The policy engine is a central component in the Teagle design as is allows to flexibly restrict the usage of the federated resources. This allows different resource providers to adapt and customize the way users interact with the resources exposed by their domain. The PE connects

[20] Adapted from Panlab work on legal and operational processes, http://trac.panlab.net/trac/wiki/PWI

[21] The Teagle source code SVN location: http://svn.panlab.net/PII/repos/Software/sources/Repository/CoreRepository

to other Teagle components via two reference points: PEM1 and PEM2 (see figure 6.21). Those are discussed in detail in the following two subsections.

Policy Definition

The Teagle portal allows resource providers to define policies and manage them through the policy engine using a Teagle portal web interface as shown in figure 6.24.

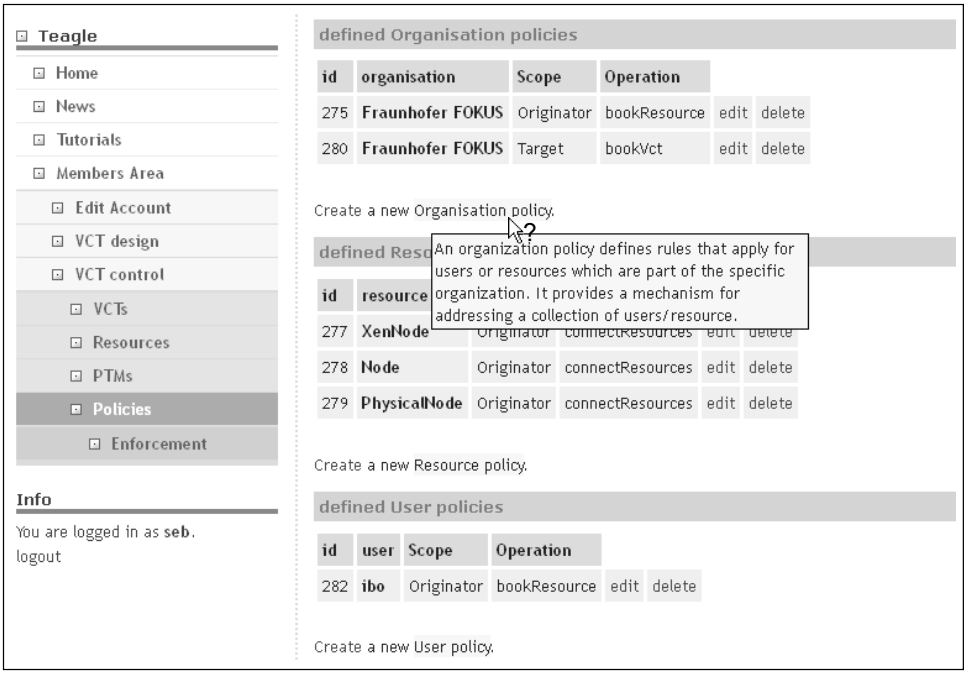


Figure 6.24 – The Teagle portal policy specification overview page

The portal implements the PEM2[22] reference point to manage the policies edited by the portal users and communicate with the policy engine. The PEM2 policy management interface has been specified by the Open Mobile Alliance (OMA)[23] and defines the messages and parameters exchanged over this interface. It is compliant to the XML configuration access protocol (XCAP)[24]. The portal acts as a *management requestor* in terms of the PEM2 specification, which abstracts from the concrete format that is used to express the policies (i.e. the policy expression language (PEL)) that are communicated on the PEM2 reference point. Therefore, multiple PELs can be supported as long as they can be mapped to the policy model used by Teagle repository. Figure C.3 (annex C) shows the data model used to store policies in the Teagle repository. It is based on the DEN-ng core specification of the policy class and extends it according to the needs of the PE implementation. The model defines the following policy types:

- UserAccessPolicy: holds user related policies that are defined and evaluated based on the identity of the requester.

[22]Policy Evaluation, Enforcement and Management - Management Interface (PEM-2), Open Mobile Alliance (OMA) specification, [137]
[23]Open Mobile Alliance (OMA) website: http://www.openmobilealliance.org
[24]RFC 4825: The Extensible Markup Language (XML) Configuration Access protocol (XCAP), [138]

- OrganisationAccessPolicy: holds organization related policies that are defined and evaluated based on the affiliation of entities.
- ResourceAccessPolicy: holds resource related policies that are defined and evaluated based on the requested resource types.
- RoleAccessPolicy: holds user role related policies that are defined and evaluated based on the role that a specific user has.

Regarding the information model classes, "[e]ach policy references a specific type of entity (User, Organisation, Resource and Role) using ResourceReference. This association defines the policy subject. The AccessControlPolicy is defined by a set of ECAPolicyRule. An ECAPolicyRule is composed of a PolicyEvent, PolicyCondition and a PolicyAction." ([139], p. 21)

Currently, two different PELs have been implemented: a *ruleset framework*[25] and the *Drools engine rule language*. The latter is implemented by the Drools Expert rule engine that is part of the open source project JBoss Drools[26]. This allowed for an open source version of the policy engine that has been published together with the other Teagle components. The Drools engine rule language makes use of object oriented design patterns while the ruleset framework is based on the declarative XML-based *common policy format*[25]. Listing 6.3 and 6.4 compare the different formats using a simplified example policy definition.

Listing 6.3 – Drools engine rule language

```
rule "exampleRule"
   when
      r : PEInputRequest(targetIdentities contains "HSS", targetIdentityType == "
          resource", event=="bookResource")
      actionMng: DrlActionsManager()
      eval (getOrganization(r.originatorIdentity) == "Fraunhofer")
   then
      DrlAction action = new DrlAction("denyRequest");
      action.addAttribute("message", "You are not allowed to book this resource.");
      actionMng.execute(drools.getRule(), action)
end
```

Listing 6.4 – Common Policy Format

```
<ruleset name="/Resource/HSS/Target/bookResource">
   <rule id="1">
      <conditions>
         <originatorIdentity>
            <many/>
         </originatorIdentity>
         <validity>
            <from>2011-04-01T10:00:00.000Z</from>
            <until>2012-04-01T10:00:00.000Z</until>
         </validity>
         <constraints>
```

[25] As suggested by the OMA PEEM Policy Expression Language Technical Specification [140] and RFC 4745: Common Policy: A Document Format for Expressing Privacy Preferences, [141]

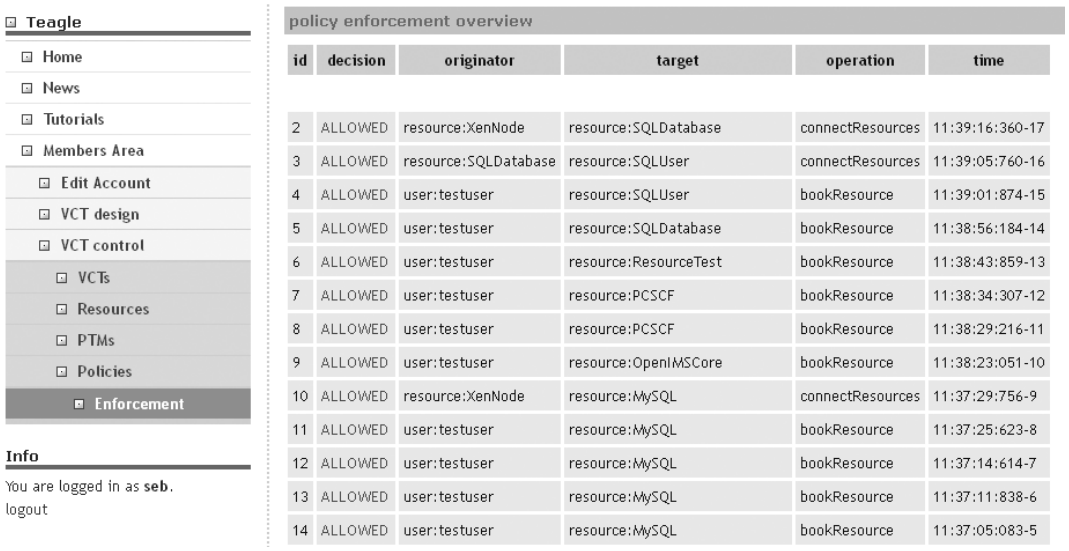
[26] JBoss Drools project website http://www.jboss.org/drools

```
12          <operator name="equal" operandsType="boolean" match="noregx" >
13            <operand1>fn:belongsToOrganization(fn:getOriginatorIdentity(),"
                  Fraunhofer")</operand1>
14            <operand2>false</operand2>
15          </operator>
16        </constraints>
17      </conditions>
18      <actions/>
19    </rule>
20  </ruleset>
```

The example shows that the information carried by the different PELs is essentially the same: users affiliated with `Fraunhofer` are not allowed to `bookResource` of type `HSS`. Different installations of the Teagle framework might choose which format (or even both) to support based on the federation organization and resource providers preferences and the software licenses of the different PE implementations[27].

Policy Evaluation

The VCT tool and the request processor connect to the policy engine via the PEM1[28] reference point to request policy evaluation. The evaluation results are visualized on the Teagle portal as shown in figure 6.25.



id	decision	originator	target	operation	time
2	ALLOWED	resource:XenNode	resource:SQLDatabase	connectResources	11:39:16:360-17
3	ALLOWED	resource:SQLDatabase	resource:SQLUser	connectResources	11:39:05:760-16
4	ALLOWED	user:testuser	resource:SQLUser	bookResource	11:39:01:874-15
5	ALLOWED	user:testuser	resource:SQLDatabase	bookResource	11:38:56:184-14
6	ALLOWED	user:testuser	resource:ResourceTest	bookResource	11:38:43:859-13
7	ALLOWED	user:testuser	resource:PCSCF	bookResource	11:38:34:307-12
8	ALLOWED	user:testuser	resource:PCSCF	bookResource	11:38:29:216-11
9	ALLOWED	user:testuser	resource:OpenIMSCore	bookResource	11:38:23:051-10
10	ALLOWED	resource:XenNode	resource:MySQL	connectResources	11:37:29:756-9
11	ALLOWED	user:testuser	resource:MySQL	bookResource	11:37:25:623-8
12	ALLOWED	user:testuser	resource:MySQL	bookResource	11:37:14:614-7
13	ALLOWED	user:testuser	resource:MySQL	bookResource	11:37:11:838-6
14	ALLOWED	user:testuser	resource:MySQL	bookResource	11:37:05:083-5

Figure 6.25 – The Teagle portal policy evaluation overview page

The policy evaluation is performed by the PE using the *callable mode* specified by [142] which allows to call the PE on interface PEM1 to request policy evaluation. "The request is based on a flexible template compatible with the OMA PEM-1 specification and provides information regarding the specific parameters and values. The Policy Engine exposes a web

[27]The ruleset framework implementation is available under a Fraunhofer FOKUS binary license, while the Drools engine based implementation is available as open source.

[28]Policy Evaluation, Enforcement and Management Callable Interface (PEM-1), Open Mobile Alliance (OMA) specification, [142]

service interface for incoming PEM-1 messages thus the underlying protocol is SOAP." ([139], p. 22) Listing 6.5 shows an example policy evaluation result. In this case, the request is denied. More details on the policy evaluation including PEM1 example messages are provided by annex B.

Listing 6.5 – Example policy evaluation XML document

```
<?xml version='1.0' encoding='UTF-8'?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
   <S:Body>
      <S:Fault xmlns:ns4="http://www.w3.org/2003/05/soap-envelope">
         <faultcode>S:Server</faultcode>
         <faultstring>DENIED: 290/resource/mysql/Originator/connectResources:
             r1:Action denyRequest has been enforced: This operation is not allowed!</
             faultstring>
         <detail>
            <denyPolicyResponseException:denyPolicyResponseException
                xmlns:denyPolicyResponseException="http://www.openmobilealliance.org/
                wsdl/PEM1/v1_0/faults" xmlns="http://www.openmobilealliance.org/wsdl/
                PEM1/v1_0/faults" xmlns:ns2="http://www.openmobilealliance.org/wsdl/
                PEM1/v1_0/local.xsd">
               <statusCode>2401</statusCode>
               <statusText>DENIED: 290/resource/mysql/Originator/connectResources:
                   r1:Action denyRequest has been enforced: This operation is not
                   allowed!</statusText>
            </denyPolicyResponseException:denyPolicyResponseException>
         </detail>
      </S:Fault>
   </S:Body>
</S:Envelope>
```

Policies contain a set of rules and are identified and evaluated using the following information:

- "Identity of the request (e.g. originatorID, targetId) or a group of identities (e.g. organization)
- Scope of the identity which represents the relationship of the identity to the operation (e.g. 'originator' or 'target' of the operation). In case one does want to ignore the scope of the identity, one can use the 'all' scope. This scope can be used only in combination with the 'all' identity which applies to all identities [...]
- Operation that triggered the evaluation request (e.g. bookResource)" ([139], p. 23)

In addition to the scope, rules specify the constrains under which they apply. "Rule constraints are processed in case the rule scope matched. Compared with the rule scope, the rule constrains define the expected values of the input request parameters in order for the rule evaluation to be successful. In case the constraints are fulfilled the evaluation process will continue with the next rule from the set. The associated actions to this rule will be gathered in a set which will be subsequently executed. Otherwise, if the constraints are not fulfilled the evaluation process will be ceased and a 'deny' action will be triggered. The Policy Engine

starts enforcement of the collected set of actions as soon as the evaluation of the rules has ended. An exclusive action is represented by the 'deny' action which will drop the enforcement of all other actions." ([139], p. 23)

The VCT tool and the RP request policy evaluation on interface PEM1 to restrict the usage of resources. At design time, the VCT tool requests policy evaluation to adapt the set of available resources that is displayed to the user and to restrict the number of possible resource combinations. If a user is not allowed to book a specific resource, it is not displayed in the list of available resources. Also, upon connecting two resources, policy evaluation is requested to check if any restrictive policies apply. This gives the user a feedback on possible resource combinations at design time. For example, to define an abstract resource such as an IMS domain to be the parent of a physical machine does not make sense. However, the definition of such relationship as a configuration reference can make sense in order to configure the physical machine to be part of a specific IMS domain. Such restrictions can be enforced defining specific policies that will be enforced by the VCT tool. If another tool has been used to design a VCT, the request processor will request policy evaluation from the PE and act as policy enforcement point instead of the VCT tool. This is required from a security perspective, as the VCT tool runs on the user's machine and might have been compromised or forged and cannot be trusted.

6.5.6 Request Processor

The RP has been implemented as a lightweight HTTP servlet[29] and is available under the Apache License, Version 2.0. It can be deployed in any servlet container that implements the Java Servlet API[30]. The VCT tool, once a VCT has been designed and stored, allows to request to book this VCT. Upon selecting this option, the tool notifies the RP about a new VCT to be fetched from the repository and to be deployed. From this point, the RP is responsible for the following tasks:

- Authenticate and authorize the request – This is done based on an authorization cookie that is written by the Teagle portal into the VCT tool JNLP start file (see page 121). This requires the portal and the RP to run within the same container[31]. In a more distributed setup, that also allows third-party tools to connect to the Teagle framework, the RP would need to implement its own authentication/authorization service or rely on a service provided by its own container (e.g. HTTP Digest [136]). Alternatively, the entire Teagle deployment environment can rely on an authentication and authorization layer that shields all Teagle services (e.g. the portal, repository, request processor) provided by the Teagle API.

- Obtain the VCT specification from the Teagle repository – This is done based on the `commonName` of this VCT in combination with the name of the `user` (see also the listings F.13 and F.26) using the *read* operation of the REST interface exposed by the repository on reference point REP (see also section 6.5.3).

- Request policy evaluation from the PE and enforce the PE decisions – Once the RP has obtained the full VCT specification from the repository, it requests policy decisions

[29] Java™ Servlet Specification, version 3.0, Sun Microsystems, Inc., [143]
[30] http://java.sun.com/developer/onlineTraining/Servlets/Fundamentals/servlets.html
[31] In the current setup this is a Tomcat servlet container that hosts the portal JSPs as well as the RP servlet.

for all resources and resource relationships from the policy engine. Essentially, those steps have already been performed by the VCT tool, if the VCT tool has been used to derive the VCT specification. However, this time it is a Teagle-internal evaluation that can be trusted from a RP point of view. The RP enforces the decisions by the policy engine. If a *deny* decision is returned by the policy engine, the VCT deployment process is stopped and a booking validation error is returned.

- Update the resource status of the requested resource instances – There are three different resource instance states: (1) `new`, (2) `unprovisioned`, and (3) `provisioned`. The VCT design may contain new resource instances or already provisioned ones. Later, during the orchestration stage, for new resources a *create* will be performed, while for already provisioned resources an *update* might be performed depending on whether any resource configuration modification is requested or not. The RP will at least assign the state `unprovisioned` to all resource instances. Once the booking has been completed successfully, the states are updated to `provisioned`. Resources that already have the state `provisioned` will *not* be reset to `unprovisioned`.

- Forward the processed VCT specification to the OE – This step involves producing an XML document that can be processed by the OE. Listing 6.6 shows an example[32]. The OE responds with a success or failure message including detailed debugging information. The next two steps are only performed in a successful case.

- Update the VCT state – In case of successful booking, the RP updates the state of the affected VCT to `booked`. In the asynchronous mode that is used for large VCTs and if the OE is using a parallel orchestration script compilation mode (see section 5.4), the state `inprogress_wait` is assigned until the entire VCT deployment process has finished. The resource states of all resources contained by a booked VCT will be updated to `provisioned`.

- Update all VCT entity identifiers from design time identifiers to run time identifiers – Upon successful resource orchestration and deployment, the OE returns the runtime identifiers of all VCT entities. The RP updates the original VCT specification hold by the repository accordingly by updating the `commonName` of all affected resources.

  Example of a *design time* identifier: `XenNode-823480093`
  Example of a *run time* identifier: `/fokus/pnode-0/XenNode-3`

  The run time identifier includes the resource hierarchy as well as a long term ID assigned by the according domain manager. See also section 5.1.2 on resource identifiers.

Listing 6.6 – VCT specification example forwarded from the RP to the OE

```
1 <testbed>
2   <components>
3     <fokus_ptm.resources.PortGroup>
4       <id>fokus_ptm.resources.PortGroup-707316526</id>
5       <state>unprovisioned</state>
6       <configuration>
```

[32] At the time of writing, the format of this specification is not yet final. It is expected that this will be subject to changes, including the according XML schema documents.

```
7         <vlan type="reference">\$dynid(fokus_ptm.resources.VLAN-94625491)</vlan>
8         <name type="string"/>
9       </configuration>
10    </fokus_ptm.resources.PortGroup>
11    <fokus_ptm.resources.VLAN>
12      <id>fokus_ptm.resources.VLAN-94625491</id>
13      <state>unprovisioned</state>
14      <configuration>
15        <name type="string"/>
16      </configuration>
17    </fokus_ptm.resources.VLAN>
18  </components>
19  <connections>
20    <connection>
21      <src>
22        <id>fokus_ptm.resources.PortGroup-707316526</id>
23      </src>
24      <dst>
25        <id>fokus_ptm.resources.VLAN-94625491</id>
26      </dst>
27      <type>references</type>
28    </connection>
29  </connections>
30 </testbed>
```

6.5.7 Orchestration Engine

The orchestration engine (OE) implements the methods described in section 5.4. It receives a Virtual Customer Testbed (VCT) specification from the Teagle request processor (RP) and executes the VCT deployment. Before starting the deployment, a work flow is generated that specifies the order of the deployment steps to be executed. After the execution has finished (or has been aborted due to an error) a report is generated and returned back to the entity that requested the VCT deployment (usually this is the Teagle RP as described in the previous section). Figure 6.26 (adapted from [101], p. 81) shows an overview of the OE system implementation revealing the OE-internal components and all OE interfaces. The OE implements *three external* (SPEC, REP, and TG) and *four internal* (COMP, GEN, CALL, and EXE) reference points. On the reference points SEPC and TG, it exposes a respective interface that can be called by other *Teagle-internal* components (such as the RP and the TGW). Depending on the Teagle framework environment deployment, requests on those interfaces may be authenticated using HTTP Digest. However, only Teagle-internal components should dispose of valid credentials and should be authorized to access those interfaces.

Reference point SPEC On the interface exposed on this references point, the OE can be called using an HTTP REST API with several distinct operations: (1) a VCT registry service, (2) an orchestration service, and (3) a deployment service. The *registry* service can be used to register a new VCT for later deployment. The request must contain a valid VCT specification and VCT identifier that are known to the Teagle repository.

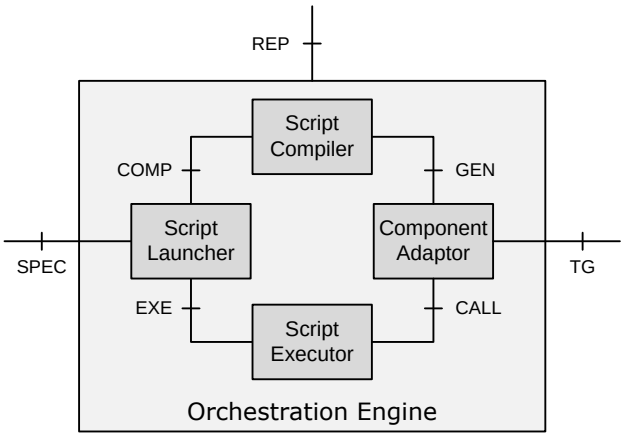


Figure 6.26 – The orchestration engine system components and interfaces

The *orchestration* and *deployment* services can be called providing a valid VCT identifier and a federation user identifier on whose behalf the respective service is requested.

The *deployment* service can be requested in two different modes. A *synchronous mode* returns the deployment result as soon as it is ready. In case of lengthy deployment procedures this might cause the connection to time out before the OE response has been received. Therefore, an *asynchronous mode* allows the OE to return a provisional result response, which contains an URL that will contain the final result, once the booking has been completed.

Reference point REP The OE may use the interface exposed by the Teagle repository on this reference point to request information on federation users, VCTs, resource specifications, etc. (see section 6.5.3 for more information on the repository interface exposed on REP). The OE itself does not expose an interface on this reference point.

Reference point TG The OE is called on the interface exposed on this reference point by the TGW. In turn, the OE may call the according TGW interface on reference point TG. The TGW maintains service endpoints for all domain managers registered with the Teagle portal (and stored in the Teagle repository) relieving the OE from such task. Also, the TGW implements different communication and security measures to communicate with domain managers on reference point T_1 shielding the OE from this complexity. Furthermore, the TGW may use the OE TG interface to notify the OE about resource provisioning events. Such notifications follow the format:

```
<notification requestId="requestID">
  <uuid>resourceID</uuid>
  <component>target component for this notification</component>
  <event>eventID</event>
  <vctId>vctID</vctId>
  <status>returnStatus</status>
  <reason>message</reason>
  <details>logfile</details>
</notification>
```

OE-internal reference point COMP On this reference point the script launcher connects to the script compiler via an interface with the following signature:

```
1 ScriptCompiler :: compile(scriptId:String, scriptSpec:XMLDoc) : status
```

The `scriptId` is an identifier for unique identification of the resource orchestration script in the orchestration system. The `scriptSpec` is an XML document that represents the VCT requested by the user. The script compiler accepts HTTP POST messages. ([144], p. 45)

OE-internal reference point GEN "The GEN reference point represents the relationship between the script compiler component and the component adaptors. Component adaptors are generated from the service interface descriptors exposed by [domain managers and the TGW]. In some cases - when using non standard interfaces - the generated code will need to be manually edited." ([144], p. 46)

OE-internal reference point CALL This reference point represents the connection between the script executor and the component adaptor. "It is simply materialized by the presence of an operation call within the executable script, where the operation call refers to an operation of an existing component adapter. An instantiation of the given component adaptor is required previous to the call. This instantiation is normally done at the beginning of the script." ([144], p. 46)

OE-internal reference point EXE On this reference point, the script launcher connects to the script executor via an interface with the following signature:

```
ScriptExecutor :: execute(scriptId:String) : status
```

The `scriptId` is an identifier for unique identification of the script in the orchestration system. The script executor accepts HTTP POST messages. ([144], p. 45)

Generally, it has to be noted that real time resource interactions were not intended. Therefore, the performance of the VCT deployment and the associated resource provisioning in terms of the time required to perform those tasks was not a very important design criterion. Still, different orchestration strategies have been specified to improve the VCT provisioning time based on the feedback received by early system users. It was observed that for complex VCTs (e.g. including several virtual machines and different virtual appliances) the resource provisioning can easily require 20 minutes and more. To improve the usability, a parallel resource provisioning schema using state machines has been investigated. As described in section 5.4, a parallel resource provisioning scheme reduces the overall time needed to perform the full set of resource management operations. "Resource booking can often be parallelized and then synchronized. This is particularly useful when dealing with inter-domain VCTs. In this case the compilation step generates an event loop: at each execution of the loop launching conditions are computed for each resource node [...]. The launching conditions are initially computed from the dependency graph. The conditions depend on state variables that are updated each time the booking of a resource completes" ([139], p. 37). Figure 6.27 (adapted from [139], p. 37) shows an example structure of a generated state-machine script.

The OE has been implemented on top of the Spatel Engine framework [145] which "[...] is based on SOA principles and finite state machines for service orchestration. All components inside Teagle (such as the Teagle repository and the Teagle gateway) and outside the Teagle

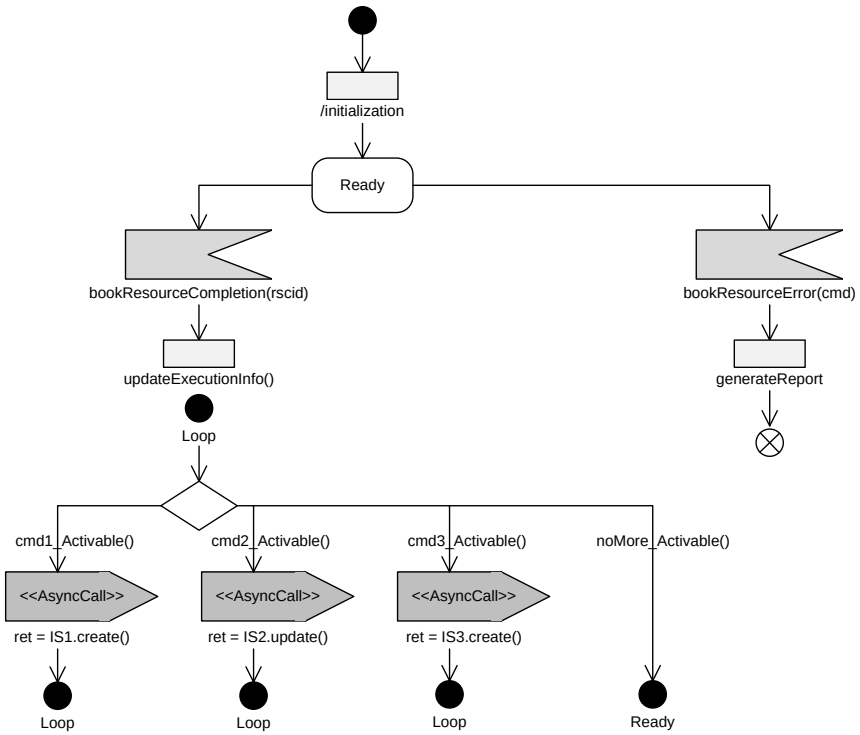


Figure 6.27 – An example structure of a state machine script used for the parallel execution of resource provisioning tasks

boundaries (such as the list of registered PTMs) are seen as Web Services exposing a list of operations. An orchestration generated to instantiate a VCT is itself seen as a service exposing the operation in charge of realizing the provisioning (the orchestration script)." ([139] , p. 35)

Therefore, VCT specifications that have been registered with the OE on reference point SPEC and that have been compiled by the script compiler are available as OE services. The logic of those services is represented by the orchestration script that is stored and may be executed again at a later stage. This provides additional services to the users as any individual VCT booking is now executable *as a service* enabling for example the re-provisioning of VCT configurations requested in the past. This behavior implements the vision of *testbed as a service* where a testbed is a virtual environment flexibly composed of individual infrastructural resources. The latest version of the OE is part of the Teagle open source release and is available under the Apache License, Version 2.0.

6.5.8 Gateway

The Teagle gateway handles bi-directional communication between the domain layer and the federation layer on reference point T_1 (see figure 6.21). It also implements the Teagle-internal reference points REP and TG. The gateway relieves the other Teagle components from implementing the T_1 communication and security mechanisms. Also, the domain managers can rely in a single communication point for addressing Teagle components. The gateway maintains the communication endpoints for all the registered domain managers and forwards resource management requests accordingly. Furthermore, the gateway inspects the resource

provisioning requests and resolves any *cross-domain configuration references*. This is done "[...] by replacing the data in the element containing the reference with the data provided by the referenced resource after querying it with a specific query type ('reference'). The data obtained by issuing this kind of query are resource specific and can be translated by the Resource Adapter that is the recipient of the orchestration request." ([139], p.38) Note that intra-domain configuration references are handled domain internally without the interference of the gateway.

Also, in complex VCT specifications *bi-directional configuration references* can occur. Such cases are also filtered by the TGW "[...] leading to an update to both ends of a reference type association. In this case both resources are queried, in the proper order, and the obtained information is passed to the other end. The order of updates is following the execution order of an orchestration, this means that if references exist in a configuration of a create request, only when the create actions ha[ve been] completed will the Teagle Gateway update the other ends of the reference associations that were enumerated in the create configuration." ([139], p.38)

Additionally, the gateway supports an event reporting feature that can be used by domain managers to notify Teagle components. The gateway routes the notifications to the intended Teagle-internal component. Notifications can inform the federation layer about the result of pending resource management operations, the domain manager status, resource registration, and resource availability.

The communication on reference point T_1 is secured by making use of SSL/TLS[33]. The certificates used on T_1 are signed by the federation organization. This allows to establish a trust relationship between the Teagle gateway and the domain managers. The gateway requests *client authentication* from the domain managers. Also, for an even tighter integration of the federation layer and the domain layer, the gateway and domain managers can be configured to support a *heart-beat* mechanism. This allows the federation layer to rely on a real-time overview of the federation and the currently available resources. The domain manager status can be one of the following: (1) OK, (2) SSL problem, and (3) Offline.

The gateway has been implemented as a Java application and is part of the latest Teagle open source release that is available under Apache License, Version 2.0. It integrates a Grizzly servlet web server[34] and a Jersey engine[35] to support RESTful Web Services, as well as an Axis-1.4 engine[36] with embedded Jetty web server[37] to support SOAP-based Web Services.

6.6 The Domain Layer

The instantiation of the *federation layer* allows a federation user to abstractly define a desired virtual environment drawing upon higher tier federation logic such as resource relationships, resource orchestration, and automated deployment. On the other hand, the user may request the according resource provisioning based on the abstract virtual environment definition. On the *domain layer*, the resource provisioning is executed in terms of lower layer resource management. The management entities on this layer may take decisions on behalf of the

[33] RFC 5246: The Transport Layer Security (TLS) Protocol Version 1.2, [127]
[34] Grizzly project website: http://grizzly.java.net
[35] Jersey project website: http://jersey.java.net
[36] Apache Web Services project website: http://ws.apache.org/axis
[37] Jetty web server project website: http://jetty.codehaus.org/jetty

user whenever specific resource configuration details have been left undefined. To exemplify this, consider that a user requests a virtual machine for which the domain, the parent (i.e. a specific physical machine), and a number of CPUs have been explicitly specified, but the amount of memory that shall be available for this machine has not been defined. In such case, the federation layer directs the request toward the requested domain, but leaves it to the domain layer to handle the resource management of the parent resource instance (i.e. the VM container) and the requested VM. Particularly, the amount of memory assigned to the VM may be decided on the domain layer, for example based on the total amount of currently available memory. This shows the importance of the domain layer in terms of resource abstraction dealing with a set of highly heterogeneous resources.

6.6.1 Domain Manager Framework Architecture

The domain layer instantiation that is discussed here has been realized as a combined Java (for Java RAs) and Python (the DM core) implementation. It is based on and extended from a diploma thesis [146] supervised by the author. Figure 6.28 shows an architectural overview of the core elements of the domain manager (DM) instantiation.

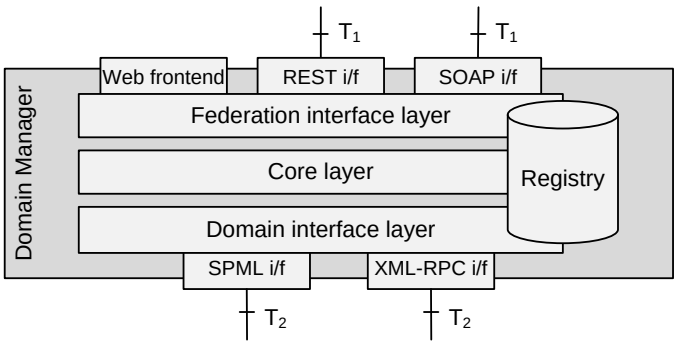


Figure 6.28 – The domain manager implementation architecture

The DM resides at the border of a domain and therefore exposes interfaces on two reference points: (1) the T_1 reference point and (2) the T_2 reference point. The DM T_1 interface is a public interface that is used by the federation layer to request resource management operations. The DM T_2 interface is a domain-internal interface that is used to communicate with domain resources.

Two distinct DM layers implement the federation and the domain interfacing: (1) the *federation interface layer* and (2) the *domain interface layer*. In those layers, several concrete interface implementations have been realized that use different protocols and transport mechanisms for data communication. However, the reference point communication semantics remain the same on each layer, independent from the transport technologies and mechanisms used for a specific interface implementation. On T_1, a REST-based as well as a SOAP-based DM interface have been implemented. On T_2, an SPML-based and an XML-RPC-based interface have been implemented. Resource abstraction *may* or *may not* rely on a remote manager configuration as shown in figure 6.29. Additionally, a Hypertext Markup Language (HTML)-based frontend has been implemented that allows to perform domain administration tasks such as adding or removing resource specifications or configuring the DM framework. Therefore, it provides a frontend for the DM registry but also allows to

perform provisioning operations directly on the domain level without having to go through the federation layer. This is mainly useful for testing and debugging of RA implementations.

The DM *core layer* deals with connecting the federation and the domain layer and implements the necessary resource management logic. Here, the decision is taken to which resource a specific management request should be forwarded and if the request needs to be modified or re-fined based on the resource type or instance in question. It implements a T_2 reference point client following the structural *facade design pattern*[38]. In this case, the T_2 client implementation wraps the DM registry and resource adapter (that can reside locally or remotely, see figure 6.29 and section 6.6.2) functions. Also, the core layer deals with *intra-domain* configuration references. While *cross-domain* (or *inter-domain*) references need to be resolved on the federation layer (the TGW provides this functionality, see section 6.5.8), references between resources of the same domain are resolved on the domain layer.

The DM registry "[...] keeps track of which resource adapter is responsible for which resource types at a certain point in the resource hierarchy, how it is to be reached and which internal communication protocols it follows." ([146], p. 47). It allows the operations[39] shown in listing 6.7 ([146], p. 55) to store and retrieve data held for the domain resources.

Listing 6.7 – The domain manager framework registry operations

```
resolve(id: QRAI): URI

list(id: RAI): <(RAI, URI)>

register(id: RAI, payload: URI): void

unregister(id: RAI): void
```

The resource identifiers used for the DM instantiation and for identifying data sets stored by the registry, follow the resource naming convention defined for the entire federation framework as discussed in section 5.1.2. The table 6.2 shows examples of the different identifier types.

6.6.2 Resources and Resource Adapters

In order to support heterogeneous resources, the domain manager framework instantiation makes use of the *resource abstraction method* introduced in section 5.2. It implements so-called *resource adapters (RAs)* to control resources that are highly heterogeneous in terms of their supported communication and control mechanisms. In this sense, RAs act like device drivers on an Operating System (OS) by translating the abstract resource control commands received from the domain manager framework *core layer* into resource specific commands. The domain manager core layer acts like a forwarding proxy dealing with T_1 reference point security, intra-domain logic (e.g. intra-domain reference resolution), and the mapping of resource identifiers.

The domain manager framework instantiation allows for several communication schemes on reference point T_2. In the most trivial case, the RA implementation plugs into the core domain manager framework, communicating either with local or remote resources directly. Here, the domain manager SPML-based provisioning system can be used (see figure

[38]The pattern can be used to provide "[...] a unified interface to a set of interfaces in a subsystem. Facade defines a higher-level interface that makes the subsystem easier to use." ([147] p. 208)

[39]Resource adapter identifier (RAI), qualified resource adapter identifier (QRAI)

Table 6.2 – Domain level resource and resource adapter identifiers

Identifier	Example	Comment
RII	`pnode-0`	As defined in section 5.1.2
RAI	`pnode-0` (if responsible for this *instance*) or `pnode` (if responsible for this *type*)	Superset of RIIs, every valid RII is also an RAI. ([146], p. 51) If a resource adapter is responsible for resource instances of a certain type, the instance identifier is omitted while the typename is kept.
QRAI	`/pnode-0/vnode-3/mysql`	Contains multiple identifiers separated by a slash operator. Used to uniquely identify a resource adapter.
URI	`xmlrpc+http://vnode-3.ptm.fokus.de:8003`	RFC 3986[40] compliant URI syntax. The *scheme part* identifies the communication protocol `xmlrpc+http` indicating that the resource adapter uses XML-RPC over HTTP. ([146], p. 54)

6.29). Alternatively, an XML-RPC-based mechanism can be used. This requires a *manager* component to run on the remote resource providing yet another level of abstraction that allows to deal with all sort of intra-domain resource connectivity and elasticity requirements. Figure 6.29 (adapted from [101], p. 84) demonstrates this using different types of remote resources as examples.

The core domain manager implements a *domain hub* that keeps track of all remote managers and communicates with them via XML-RPC over HTTP. The managers offer a run-time environment for RAs and usually dispose of a set of RAs that are accessed as native objects (e.g. plain old Java objects (POJOs) in case of a Java manager and a Java RA implementation). Also, managers provide the resource adapters under their control with client capabilities so that those can access other domain services themselves. "By doing so, managers completely hide any handling of network protocols, locality of resources or transmission formats from resource adapters and their respective implementers." ([146], p. 59-60)

The domain manager framework aims at enabling the easy implementation of resources and the according adapters. This addresses the resource provider requirements discussed in section 6.1.2. Implementors of resource adapters may use "[...] platform specific mechanisms and semantics being fully oblivious to any external communication or transmission requirements. Usually, this simply involves deriving an appropriately named class from a common base class and providing implementations for some abstract methods. However, a set of convenience classes [...] use means of reflection and introspection to facilitate a number of common tasks. This allows implementers to focus on the actual task of resource management while writing only a minimum of interface code." ([146], p. 49)

[40]RFC 3986: Uniform Resource Identifier (URI): Generic Syntax, [148]

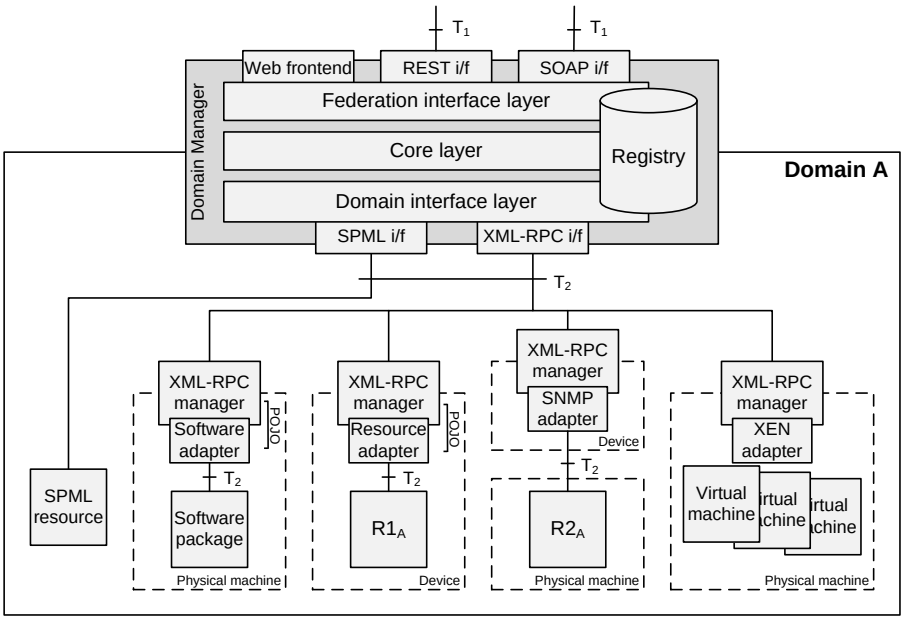


Figure 6.29 – The reference point T_2 concept and instantiation

6.6.3 Resource Adapter Description

To simplify the resource provider task of resource abstraction and implementation even further, a resource adapter description language (RADL) [149] has been developed. The language is a syntax for the definition of resource adapters. It abstracts from programming languages and implementation paradigms using a meta-model. "RADL is an attempt to describe an RA in a way that decouples it from the underlying implementation code. RADL's textual syntax aims to simplify the description of an RA rather than having to write code in Java or other target languages. We anticipate that one can define an RA in RADL without even knowing the target language." ([149], p. 7)

The RADL meta model has been defined in *Ecore*, which is a variant of the Meta Object Facility (MOF) [150] defined by OMG, using the Eclipse Modeling Framework (EMF)[41]. It is given in annex E on figure E.1. In terms of RADL, an RA is "[...] an aggregation of some parameters, particularly the `BindingParam` and `ConfigurationParam` [see figure E.1]. The class `Protocol` wraps the concept of the API configuration. Currently, four APIs have been examined: SSH, HTTP, XML-RPC, and a Java class. The `ConfigurationParams` are passed together with the `BindingParams` to the resource to be configured." ([149], p. 8)

The developed prototype allows to model an RA using an Eclipse-based editor[42] that supports the RADL syntax. Immediate syntax evaluation and error-detection increase the usability of the RADL editor. From the abstract RA model expressed in RADL, Java code can be generated using a *model-to-code transformation*. This allows to tailor the resulting code toward a specific domain manager framework implementation and publish the new resource to the Teagle repository. Such model-to-code (or model-to-text) transformation has been implemented using Xpand[43]. Several adapters have been generated using RADL including

[41] EMF website: `http://www.eclipse.org/modeling/emf/?project=emf` [151]
[42] Developed at the University of Patras, `http://trac.panlab.net/trac/wiki/RADL`
[43] A statically-typed template language used for model transformation and validation [152]

a simple RA for Amazon VM instances using the EC2 API [30]. The RADL code for this adapter is given in the annex in listing E.1. Figure 6.30 (adapted from [149], p. 14) shows the process of how a resource provider can use RADL to define the Amazon EC2 RA that is made available by a target domain manager (DM) and registered with Teagle. The federation user may now use the tools provided by Teagle (e.g. the VCT tool) to configure and deploy a desired EC2 VM. Also, figure 6.30 shows the communication of the RA with the Amazon domain.

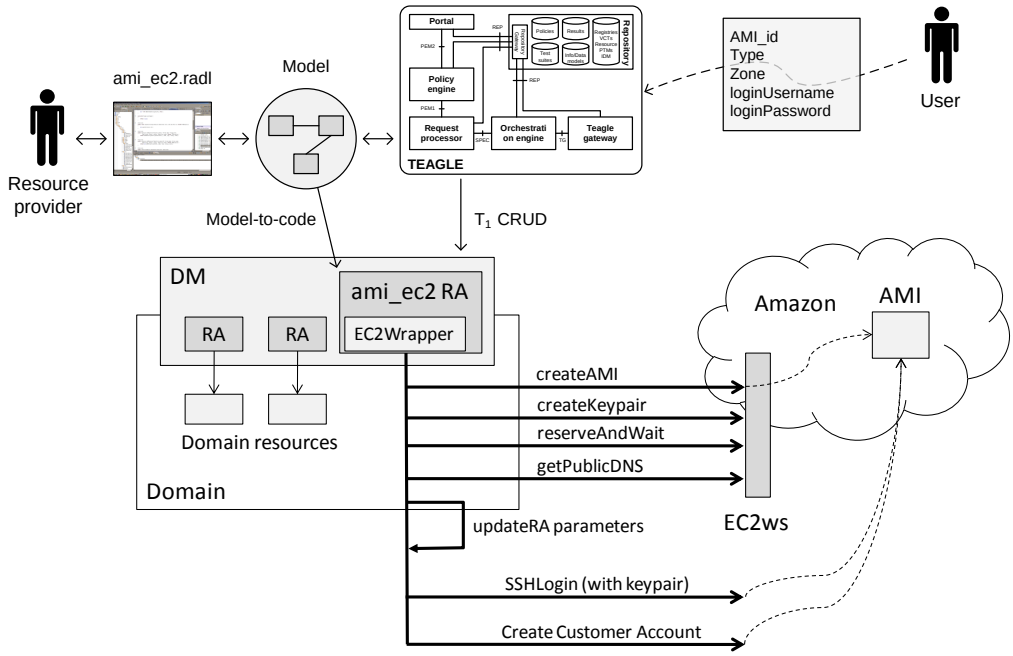


Figure 6.30 – The use of the resource adapter language and the EC2 example resource adapter

It has to be noted here that this is a specific case, because Amazon (in terms of a resource provider organization) is most likely *not* part of the federation on the same federation level as the domain that implements the RADL-based EC2 RA. Therefore, this scenario is somewhat close to the recursive federation scenario introduced in section 4.4.3 because the resource provider acts as a resource broker on level $n + 1$ while Teagle reside on a $n + 2$ federation level. This demonstrates the power of the recursive scenario where resource brokering can occur across multiple federation levels. However, for this concept to function in a *dynamic* manner and in a commercial context, flexible SLA mechanisms would need to be in place. In the rather static settings of the prototypes described here and in section 7.1.5, and given that fact that the users of those instantiations are mostly academics, less emphasis has been put on dynamic SLA negotiation. However, interesting work into this direction based on the details described here has been performed recently [153].

6.7 Cross-Domain Interconnectivity

The previous section explained how resources can be described, discovered, configured, orchestrated, and provisioned to derive a virtual resource grouping. This section describes

how the resources that are part of such virtual grouping can be connected across domains using an interconnection gateway (IGW) and how the users can access them. The presented IGW instantiation is shown in figure 6.31 ([115], p. 8) and has been implemented by the PII project [97].

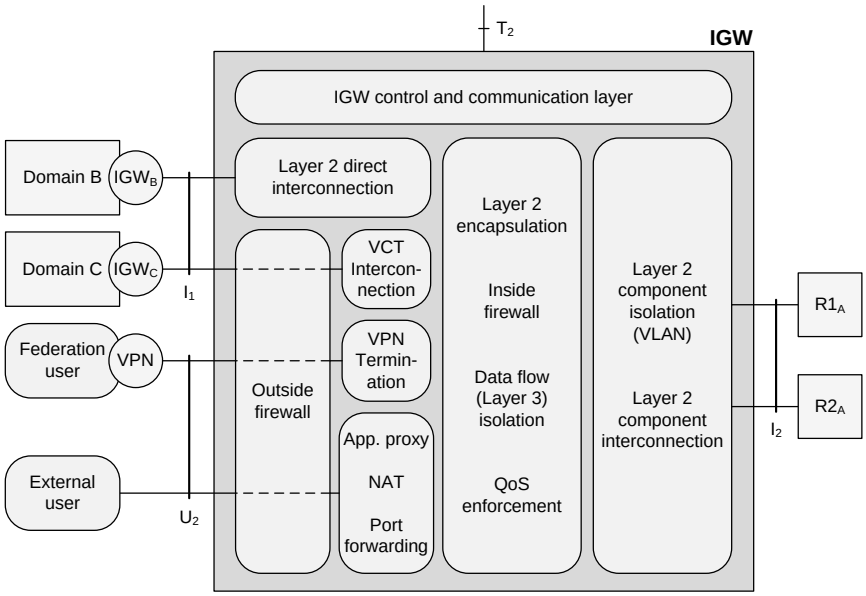


Figure 6.31 – The interconnection gateway instantiation exposing interfaces on the reference points T_2, I_1, I_2, and U_2

The IGW communicates on four different reference points: (1) T_2, (2) I_1, (3) U_2, and (4) I_2, and represents a connection state machine that controls connections established with other federated domains.

Reference Point T_2

The T_2 interface is used by the IGW resource adapter (RA) to control the IGW configuration by creating or deleting intra- and inter-domain communication states. The communication is based on SNMPv3[44] using command/reply messages and event updates.

Reference Point I_1

On I_1, the IGW communicates with other peering IGWs using a stateless low-overhead tunneling approach. Such virtual network connections are established using pre-shared keys when resources of two target domains have been configured and booked to be part of the same virtual resource grouping (i.e. the same VCT). IGWs dispose of a list of federation partner IGW endpoints, received during the initial setup process. Upon connecting to a remote IGW, a file of other IGW peers known to the remote IGW is obtained. This allows to discover new IGW endpoints without relying entirely on the Teagle layer for remote domain IGW discovery. "On the IGWs internal connection state machine, active VCTs are lists of tuples consisting of the other IGW's external address and the collision domain(s) associated with

[44]RFC 3411: An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks, [154]

the specific VCT behind it. Each interconnection state can be expanded by adding more interconnections. [...] They use the same VPN tunnel but are separated during the routing and filtering process. This guarantees an on-demand automatic IGW-to-IGW meshing [...] without using proprietary inter-IGW protocols." ([115], p. 8)

The IGW instantiation also supports a direct layer 2 interconnection on reference point I_1. This has been build to enable an alternative to the default connection via public Internet. If supported by the domains on both ends, such setup can be used to provide real QoS reservations for example relying on asynchronous transfer mode (ATM) or fibre optic links. However, by the time of writing, this setup has not been tested as the required interconnection facilities were missing.

Reference Point U_2

On U_2, users can connect to the resources that are part of a specific virtual resource grouping by using a VPN dial-in feature. This L2TP-based on-demand tunneling [123] approach allows users to connect to resources as if they would be working from within a domain that disposes of a *full IGW*. Additionally, users can *directly* connect to *public* resources provided by a domain. However, an IGW firewall (outside firewall, see figure 6.31) as well as network address translation (NAT) and application layer proxying help resource providers to secure their domain and customize the IGW according to their local policies and network topology.

Reference Point I_2

On reference point I_2, the IGW connects to the domain resources and performs collision domain isolation (see also section 5.5). Generally, the IGW's main task is to "[...] interconnect, keep, and protect the mapping of local collision domain communication to external VPN interconnection. Therefore, it functions like an IP-based trunking device [...] communicating on data planes separated by collision domains on the internal side and VPN-based access on the external side. Routing of data plane packets in-between these secure channels is done by the interconnection engine." ([115], p. 9) The channels are en/de-capsulated, en/de-crypted, and filtered by the IGW. An internal IP-based firewall allows to restrict resource access as demanded by local policies (the inside firewall, see figure 6.31). Collision domain channel isolation is achieved making use of IEEE 802.1Q virtual bridged local area networks [121]. To secure the entire intra-domain path from the resource to the IGW, only devices that support such technology must be used. Otherwise, no full separation between parallel sessions can be guaranteed. Figure 6.32 ([115], p. 10) shows a concept for connecting several virtual resources via virtual local area networks (VLANs).

This concept relies on an Ethernet switching resource `ETH-sw-fabric` that can be exposed to higher federation layers as any other resource via T_2 and T_1. To exemplify the concept, the following description lists several example network segments identified by their VLAN number as shown in figure 6.32:

`vlan#10` This network segment is a management network providing connectivity to the domain manager (DM) (that connects to the Teagle gateway on T_1) and the management operations of the `dom0` (domain zero)[45] and physical hosts.

[45] Represents the first and specially privileged domain started by the hypervisor

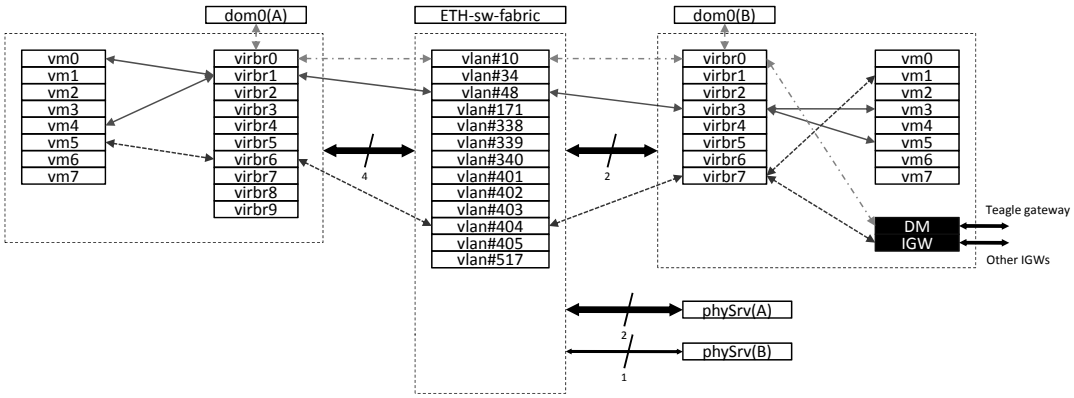


Figure 6.32 – VLAN-based collision domain isolation

`vlan#48` This network segment interconnects the virtual machines `vm0` and `vm4` on `dom0(A)` with `vm3` and `vm5` on `dom0(B)` as part of a single Ethernet segment. The virtual bridges `dom0(A):virbr1` and `dom0(B):virbr3` provide the link between the different zero domains started by the hypervisors running on the physical hosts A and B.

`vlan#404` This network segment connects `dom0(A):vm5` via `dom0(A):virbr6` to `dom0(B):vm1` via `dom0(B):virbr7` as part of a single Ethernet segment. Also, the IGW is a member of this segment via `dom0(B):virbr7`. This allows to connect the resources of `vlan#404` with remote resources offered by other domains running a peering IGW.

The user may define network topologies using multiple VLANs as a shared medium to connect multiple resources on a domain. Essentially, the concept allows to expose the entire domain-internal topology to the federation user. "However, this flexibility comes with a significant complexity in configuring the network layer." ([115], p. 9)

As an alternative to the Ethernet switching resource, the container of virtualized resources may assign VLAN tags transparently providing the isolation capability. In this case, such resources must be connected to the IGW via a direct link in order to support full isolation. By the time of writing, the IGW has been implemented and released as a pre-configured redhat[46] virtual machine image.

6.8 Summary

This chapter provided a detailed overview of the *federation framework instantiation* covering both the *federation* and the *domain layer*. The system components have been build driven by the different *stakeholder requirements* and an according requirement *analysis*. The analysis revealed partly conflicting requirements. Particularly, some of the federation user goals are conflicting with several resource provider goals. *GRL evaluation strategies* have been used to analyze the impact of critical system design aspects on the different stakeholder goals. From this analysis, important *design decisions* have been derived such as the implementation of a common information model and a common control framework.

The resulting framework prototype including a number of system components such as a design tool, a request processor, a domain manager, etc. have been introduced and discussed

[46] Red Hat, Inc. website: `http://www.redhat.com`

in detail. References to detailed technical information provided by the annex have been given where appropriate. Additionally, an *UCM-based use case analysis* helped to follow and understand the intended system interactions. The next chapter will evaluate the framework instantiation and the other design artifacts using information system research and design science *evaluation methods*.

7 Evaluation

Result evaluation is a crucial step of the overall research process. This chapter aims at evaluating the design artifacts delivered by this thesis. As described in section 1.4, the research methodology that was followed to produce the presented research results, explicitly defines an *evaluation* step. Hevner et al. characterize this important aspect as follows:

> *The utility, quality, and efficacy of a design artifact must be rigorously demonstrated via well-executed evaluation methods.* [11], p. 85

Several evaluation method categories exist, where each category groups a number of specific methods. Among the known methods are observational, analytical, experimental, testing, and descriptive methods. To evaluate this work's design artifacts, *observational* and *experimental* evaluation methods have been used. For the observational evaluation, a *field study* has been performed to analyze how the design artifacts are used and further developed in a total number of six scientific research projects. For the experimental evaluation, a *controlled experiment* has been executed to analyze the behavior of the instantiation in a laboratory environment.

Some of the projects targeted by the field study, such as the FP7 PII project, heavily drove the development of the artifacts in the first place. *Early feedback* from those projects and their users has been incorporated into the artifact design, making the design process "[...] an iterative and incremental activity [...]" ([11], p. 85). In section 7.3, the solution presented here is compared with other approaches.

7.1 Observational Evaluation: Field Study

Among the observational evaluation methods are *field studies* and *case studies*. Table 7.1 ([11], p. 86) explains the difference. The field study has been chosen to perform an observational evaluation because the artifacts are, in fact, used in multiple projects. Therefore, sufficient data could be gathered to allow for an evaluation of the "utility, quality, and efficacy" of the design artifacts.

The fact that the artifacts are used in multiple projects, demonstrates the impact on the research community. Generally, only those research results that are considered to be of high quality and that are effective in solving the problem, are repeatedly used across

Table 7.1 – Observational evaluation methods

Category	Evaluation methods
Observational	Case Study: Study artifact in depth in business environment
	Field Study: Monitor use of artifact in multiple projects

different projects. The projects that have been investigated for this field study are: FP7 PII, FP7 TEFIS, FP7 BonFIRE, FP7 NOVI, BMBF G-Lab, and FP7 OpenLab. The following subsections introduce those projects and explain how the artifacts are used. Also, table 7.3 shows a summary and comparison of the different aspects.

7.1.1 The FP7 PII Project

The Pan-European laboratory infrastructure implementation (PII) project[1] followed a predecessor project called *Panlab*. Therefore, the federation build by the PII project is usually called the *Panlab federation*. By the time of writing, the Panlab federation operates across a total number of 15 domains. It fully follows the federation model presented in chapter 4 and implements a number of methods outlined in chapter 5. Until today, a *Panlab office* acting as a federation organization has not yet been established although an operational and a legal framework have been developed. Therefore, it is unclear how the Panlab federation can be sustained after the end of the EC-funded PII project. Generally, the sustainability of federated infrastructure from a business perspective is subject to further study. The FIRE architecture board[2] has established a working group on this issue.

The federation scenario used in PII is a *central scenario* as characterized by section 4.4. Additionally, a *consortium operation* is planned but not yet established. The PII federation relies on a common information model and resource descriptions, and implements a resource abstraction and control framework. Resource orchestration, provisioning, and interconnection mechanisms allow for abstract federation level resource control. Therefore, all of the federation methods described in chapter 5 have been used to establish the Panlab federation. This allows to control a highly heterogeneous set of resources across the 15 Panlab domains. Among the resources that can be controlled by the time of writing are physical, virtual, and logical resources as summarized by table 7.2. This demonstrates the effectiveness of the design artifacts in dealing with a high heterogeneity of the federated resources.

[1]The Panlab projects website: http://www.panlab.net

[2]The FIRE STATION project website: http://www.ict-fire.eu/home/firestation.html, FIRE STATION runs the FIRE architecture board.

Table 7.2 – The federated Panlab resources

Resource	Resource type	Description
Physical server	Physical resource	Usually acts as a container for child resources, supports configuration of the host name.
System user	Logical resource	Can be configured as a child of a physical server.
XEN node	Virtual resource	Acts as a virtual node, supports configuration of a specific image, host name, number of CPUs, and amount of memory.
X-CSCF	Software resource	Deploys P-, C-, I-CSCF software to a node. Supports various configuration parameters specific to this software.
HSS	Software resource	Supports various configuration parameters specific to this software.
Open IMS core	Software resource	Supports various configuration parameters specific to this software.
MySQL server	Software resource	Can have multiple database resources as references.
SQL database	Logical resource	Should reference an SQL database server resource.
SQL user	Logical resource	Should reference an SQL database resource.
XDMS	Software resource	Supports various configuration parameters specific to this software.
DNS domain	Logical resource	Should reference a DNS server resource.
IMS domain	Logical resource	Should reference an IMS core resource.
Apache	Software resource	Web server, supports various configuration parameters specific to this software.
Bind	Software resource	DNS server, supports various configuration parameters specific to this software.
Compute resource	Virtual resource	Deploys a virtual machine using OCCI.
Storage resource	Virtual resource	Deploys a storage resource using OCCI.
Network resource	Virtual resource	Allows network configuration using OCCI.
Cloud	Logical resource	Compromises OCCI compute, storage, and network resources.
AMI EC2	Virtual resource	Creates an Amazon machine instance using the EC2 service.
Federica computer	Virtual resource	Allowing to deploy a FEDERICA project virtual machine resource.
Chronos	Logical resource	Allows to control the Chronos application for creating dynamic network connections within the HPDMnet network.
FFMpeg	Software resource	Provides FFMpeg computing capabilities to record, convert, and stream audio and video.
VideoLan	Software resource	Video playing and streaming resource.
GSN VM	Virtual resource	Allows to deploy a Green Star virtual machine.
GSN monitor	Software resource	Provides power monitoring capabilities for GSN resources, should reference GNS VM resources.

A number of use case descriptions have been published[3] that demonstrate the *applicability* of the *design artifacts* to different usage areas. The following listing gives an overview of the different use cases that have been implemented by Panlab stakeholders.

Testing a VOIP user agent This use case[4] introduces the experimenters requirements in testing a voice over IP (VOIP) application and setting up a desired environment. The use case description document contains the virtual resource grouping design and explains the used resources and the involved federation domains. The design environment instantiation and the process of how to derive a functional virtual resource grouping is evaluated. Two important aspects were revealed by this evaluation: (1) parallelization of provisioning actions would be useful to speed up the virtual resource grouping deployment time, (2) improvement of the RA development process and tools would be desirable. The first issue impacted the design of the orchestration engine (see section 6.5.7), while the latter led to the development of RADL (see section 6.6.3).

Stress test the Open IMS core This use case[5] explains and evaluates how the federation model, methods, and the Teagle framework representing the system instantiation can be used for *stress testing* of NGN components. Additionally, some resources that act as the parent resources for the NGN software components are provided as OCCI-based cloud resources. A dedicated section discusses the results in more detail (see section 7.2). The main lessons learned from this evaluation was that network resources become increasingly important to perform service layer experiments in cloud environments as the application performance heavily relies on the inter-domain network path. Here, best effort Internet seems not to be enough to enable scenarios such as *cross-domain virtual machine and/or service migration*. This result impacted the design of the IGW which provides a concept for including other connectivity facilities beyond layer 3 public Internet.

Testing adaptive admission control and resource allocation algorithms This is a use case[6] that exploits the capability of the design artifacts to enable *programmatic resource control*. The experiment requires to regulate system resources such as CPU and memory size of a virtualized resource while the experiment is running. This led to the development of the Federation Computing Interface (FCI)[7] which can be considered as work that is based on the results of this thesis but goes beyond this thesis' scope.

Testing enhanced Web TV services over mobile phones This use case[8] explains and evaluates how the design artifacts can be used to federate WebTV and IMS services

[3] Use case section on the Panlab website: http://www.panlab.net/use-cases.html

[4] Christos Tranoris, *Testing a VOIP user agent*, University of Patras, http://www.panlab.net/fileadmin/documents/Use-Cases/VOIP_user_agent.pdf

[5] Sebastian Wahle, Thomas Magedanz, and Konrad Campowsky, *Getting started with Teagle - A FIRE testbed federation tool*, Fraunhofer Institute FOKUS & Technische Universität Berlin, http://www.panlab.net/fileadmin/documents/Use-Cases/Stress_test_openimscore.pdf

[6] Christos Tranoris, *Testing adaptive admission control and resource allocation algorithms*, University of Patras, http://www.panlab.net/fileadmin/documents/Use-Cases/Adaptive_admission_control_and_resource_allocation.pdf

[7] Federation Computing Interface, http://trac.panlab.net/trac/wiki/FCI

[8] Denis Mischler, Sergio Morant, Michel Corriou, and George Korinthios, *Testing enhanced Web TV services over mobile phones*, http://www.panlab.net/fileadmin/documents/Use-Cases/Web_TV_services_over_mobile_phones.pdf

across several domains including the end user and resource provider as most important system stakeholders. The results show that it is challenging but possible to federate industrial resources provided by a telecommunications operator and provide them to federation users.

Testing end-to-end self-management in a wireless FI environment This use case[9] describes an experiment that includes advanced wireless resources (a WiMAX base station and several WiMAX clients). While, initially, the WiMAX air interface has been set up manually, the design artifacts have been used at later stages to provision the WiMAX base station and the subscriber stations. If sufficient demand from the community is observed, those services are expected to be operated on a commercial basis.

EzWeb application over TID SDPLabs This use case[10] shows how service delivery platform (SDP) resources and open Telco APIs can be exposed to federations. The design artifacts are used to provide Telefónica I+D SDPLabs resources and enable the design of virtual resource groupings making use of a Short Message Service (SMS) sending capability via a configuration reference. The results from implementing and evaluating this use case scenario show that it is possible to model and sufficiently abstract telecommunication provider resources to federate third party SDP messaging capabilities. Users have been enabled to send SMS messages over the Telefónica life network demonstrating the applicability of the design artifacts to yet another specific type of resources and the possibility to include industrial resource providers.

It can be concluded that the Panlab projects drove the development of large parts of the research presented by this thesis and made extensive use of the results, demonstrating the *utility* and *quality* of the design artifacts. Also, additional work beyond the Panlab project objectives has emerged from the results, showing the *efficacy* of the artifacts in addressing the problem of generic resource federation.

7.1.2 The FP7 TEFIS Project

The TEFIS project[11] is an FP7 project launched in June 2010. It aims at providing a federated facility building upon a set of heterogeneous testbeds to support Future Internet experimentation combined with user-oriented living labs. In detail, TEFIS provides (taken and adapted from the TEFIS DoW[12], p. 2):

- An open platform to integrate and use heterogeneous testbeds based on a connectors model and exposed as a service.
- Integration of six complementary experimental facilities, including network and software testing facilities, as well as user-oriented living labs.

[9] Jussi Mäkinen et al., *Testing end-to-end Self-Management in a Wireless Future Internet Environment*, http://www.panlab.net/fileadmin/documents/Use-Cases/Testing_end-to-end_Self-Management_in_a_Wireless_Future_Internet_Environment.pdf

[10] Lourdes Calvo Val, *EzWeb application over TID SDPLabs: "PII Message Sender"*, Telefónica I+D, http://www.panlab.net/fileadmin/documents/Use-Cases/EzwebApplicationOverTIDSDPLabs.pdf

[11] TEFIS project website: http://www.tefisproject.eu

[12] TEstbed for Future Internet Services (TEFIS), Large-scale integrating project (IP), Call FP7-ICT-2009-5, April 2010

- A platform to share expertise and best practices.
- Core services for flexible management of experimental data and underlying testbed resources during the execution of a user-defined experiment workflow.
- A single access point to testbeds instrumented with a large number of tools to support users throughout the whole lifecycle of an experiment (compilation, integration, deployment, dimensioning, evaluation, monitoring, etc.) and allow them to work together by sharing expertise.

The TEFIS architecture is shown in figure 7.1 ([155], p. 7).

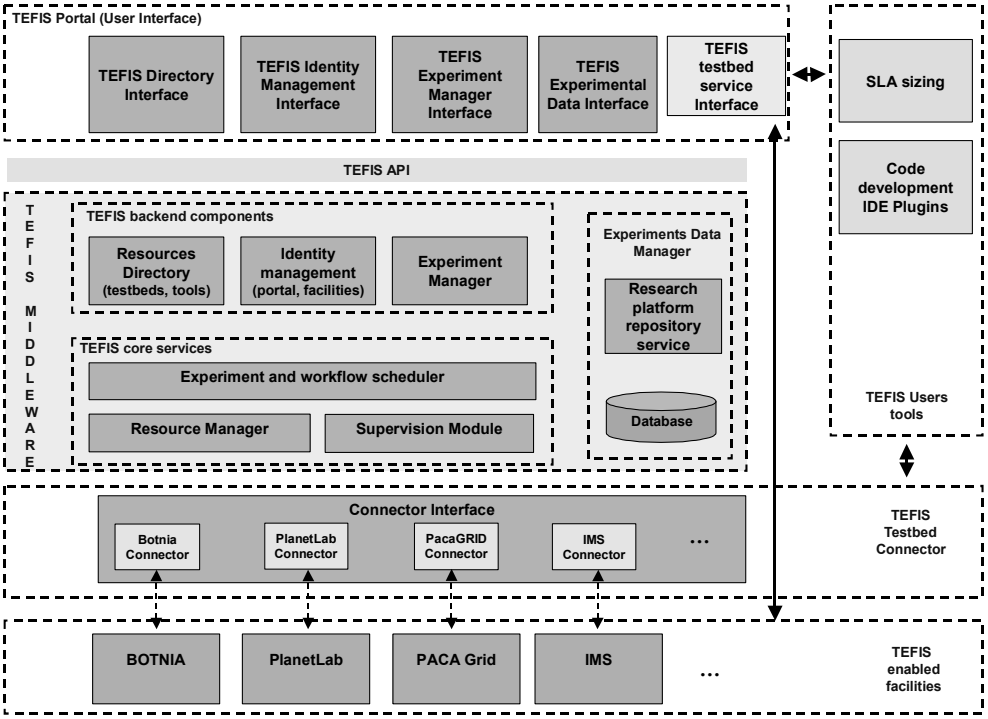


Figure 7.1 – The TEFIS high level architecture

The different TEFIS domains (where each domain represents a dedicated TEFIS test site) are summarized by the following listing and table 7.2 ([155], p. 39).

BOTNIA The BOTNIA domain is a Living Lab serving as a facility for research, development, and innovation for the creation and refinement of ICT-based services. The site supports users in the generation of new knowledge, methods, and tools, for open user-centric research and innovation. ICT-based products and services are experimentally developed in real-life contexts with real users. ([155], p. 29)

PlanetLab The PlanetLab TEFIS domain relies on PlanetLab resources governed by PlanetLab Europe which is the European portion of PlanetLab. It was funded by the European Commission's FIRE unit through the FP7 OneLab2 project. ([155], p. 12)

PACA Grid This TEFIS domain provides a set of machines deployed within the INRIA Sophia Antipolis network with the objective to provide computational resources for

		PlanetLab	PACA Grid	ETICS	IMS	BOTNIA	Kyatera
Need Identification						X	
Business Model definition		X	X		X	X	X
Software design						X	
Software Development				X	X		
Testers & quality		X	X	X	X	X	X
System engineering (Network, SLAs)		X	X		X	X	X
Targeted users	Network operators				X		X
	Service Developers			X	X	X	
	Researchers	X	X				X

Figure 7.2 – The TEFIS domains

applications requiring relevant computing effort. Examples include scientific simulations, prototypes for the evaluation of algorithms, financial computations, and image processing. ([155], p. 19)

IMS This domain is operated by Software Quality Systems, S.A. (SQS) and provides an IMS testbed for validating and testing applications over IMS. The test site disposes of a real IMS network as well as a simulation environment. ([155], p. 25)

Kyatera The Kyatera domain provides an optical network connecting several research institutes in the state of São Paulo in Brazil. It is still in deployment phase.

ETICS The ETICS domain provides solutions for software development lifecylce management. The ETICS system is designed to simplify the development process while improving the quality, reliability, and interoperability of distributed complex systems. The official ETICS system instance is hosted by CERN, using CERN machines. The instance that is made available to the TEFIS project is hosted by Engineering R&D. ([155], p. 21)

Regarding the *utility* of the design artifacts presented by this thesis it can be said that several of them are used by the TEFIS project:

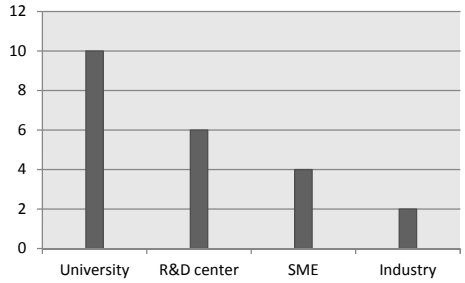
- The TEFIS federation concept, architectural elements, and stakeholders can fully be modeled with the model presented in chapter 4. The federation follows the *central scenario* where resources from six different domains are accessible via the TEFIS portal and the TEFIS API.
- From a methodological point of view, TEFIS relies on several of the federation methods presented in chapter 5. A *common model and resource descriptions* are used that define

the data format of the information stored by the TEFIS *resource directory*. Also, figure 7.1 demonstrates that the architecture relies on the concept of *resource abstraction* where *resource connectors* are used to deal with heterogeneous resources.

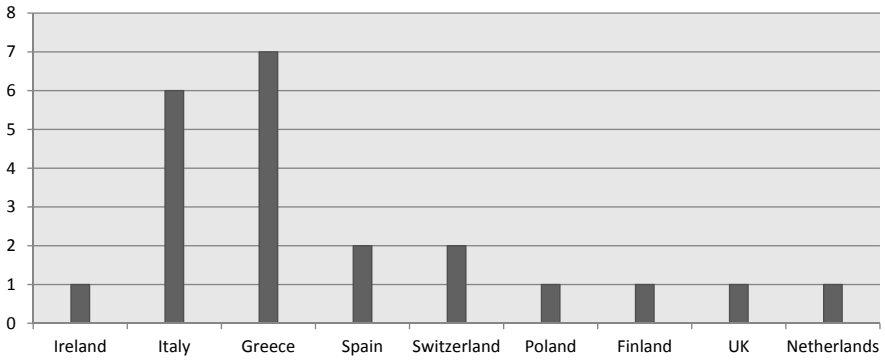
- The Teagle and domain manager instantiation presented in chapter 6 are used and extended by TEFIS on the TEFIS middleware and connectors levels. In addition, the Teagle portal capabilities are extended by adding an experiment manager that is able to organize and execute experiments and collect the experiment execution results. Also, the Teagle resource model has been extended by keywords to allow for automatic matching of experimentation requirements to Teagle-controlled resources.

From the above it can concluded that TEFIS makes extensive use of the research results presented here. This allowed the project to build upon a considerable knowledge base and existing prototypes speeding up the progress during the first year of the project.

Furthermore, TEFIS organized an *open call for experiments* where selected experiments are co-funded by the European Commission. 22 proposals have been submitted to the open call and are currently being evaluated. The winners of this evaluation will gain access to the TEFIS facility—that is heavily based on the design artifacts presented in this thesis—in order to execute their intended experiments. This demonstrates the impact of the artifacts on the European ICT research community and the TEFIS project stakeholders in particular.



(a) Proposing organization types



(b) Nationalities of the proposing organizations

Figure 7.3 – Open call results demonstrating the impact on the European ICT research community. In both sub-figures the ordinate plots the number of submitted proposals.

Figure 7.3 shows some specifically interesting results from the TEFIS open call as it allows for additional conclusions on the usefulness of the artifacts. Subfigure (a) shows which organization types are primarily interested in using federated resources. It is not surprising that especially academic users are interested in using federated and public infrastructure as the subject of investigation is mostly of academic and less of commercial interest. In turn, industrial users tend to experiment with ideas, products, and services closer to the market making intellectual property and data privacy considerations an important aspect of the experimental facility selection process.

Subfigure (b) demonstrates that users from all across Europe are attracted to the federated facility provided by TEFIS with a clear emphasis on Italy and Greece. In Greece public funding for ICT experiments is currently very hard if not impossible to acquire explaining the high number of Greek proposals.

7.1.3 The FP7 BonFIRE Project

The BonFIRE project[13] is an FP7 FIRE project that started in June 2010 and has a duration of 42 months. It aims at providing a federated testbed for cloud computing related research with a specific focus on exploring the interactions between the service layer and the underlying network infrastructures. The offered resources are provided by five domains as depicted in figure 7.4[14].

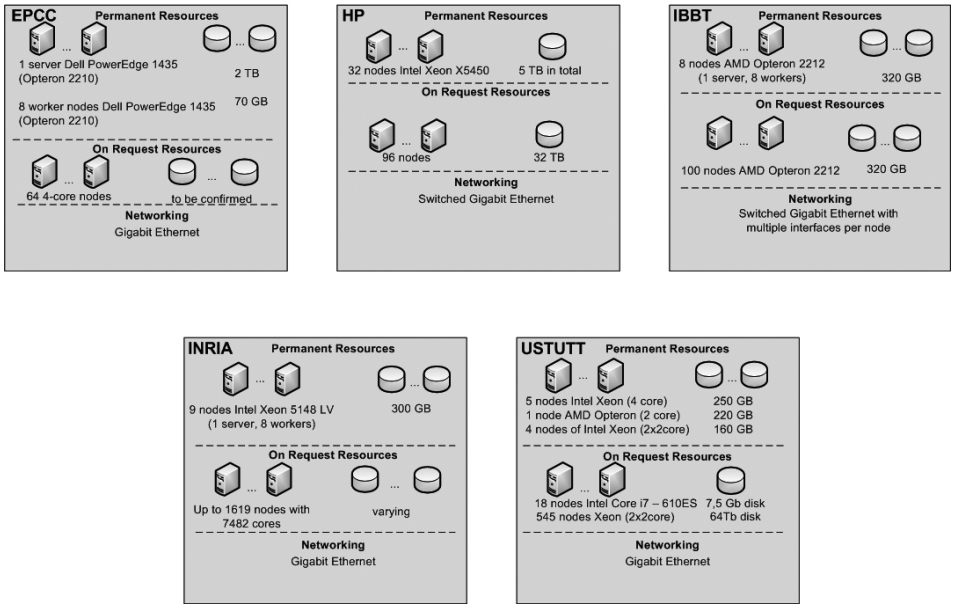


Figure 7.4 – The BonFIRE domains and the respective resources provided to the facility

The BonFIRE project adapted some of the artifacts presented by this thesis while others were deliberately build from scratch. The BonFIRE federation builds upon the *central federation scenario* and most of the federation framework components and stakeholders can be modeled with the *federation model* presented in chapter 4. However, the *gateway* model entity is currently not used as the deployed resources are expected to be accessed via public

[13] BonFIRE project website: http://www.bonfire-project.eu
[14] http://www.bonfire-project.eu/content/testbeds

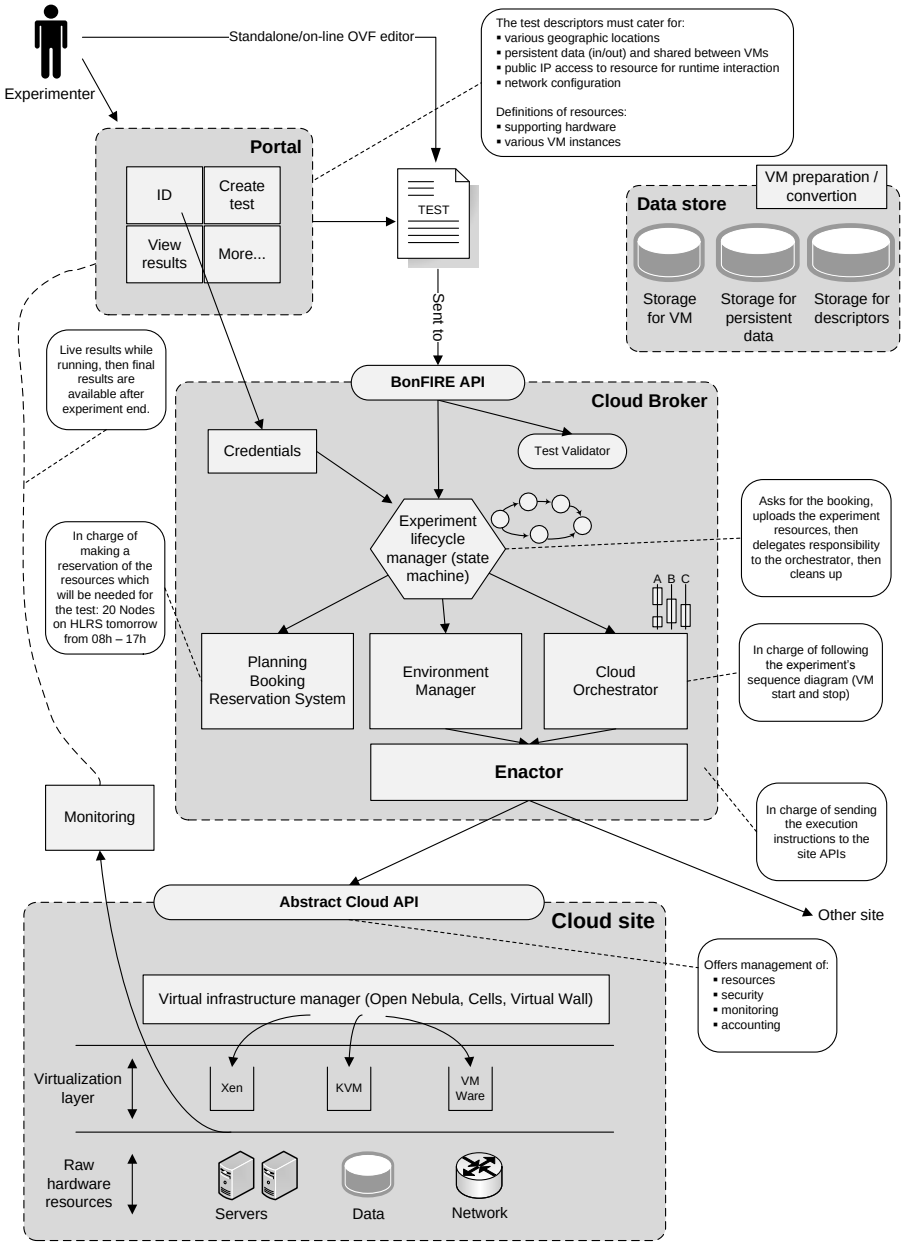


Figure 7.5 – The first release of the BonFIRE architecture

interfaces for the first BonFIRE releases. However, private network connections might be added to the BonFIRE facility at a later stage.

From the methods presented in chapter 5, BonFIRE currently uses three: (1) the facility builds upon common resource descriptions, (2) uses a common resource abstraction and control framework, and (3) implements the possibility for central resource brokering using the resource orchestration and provisioning method. Figure 7.5 ([156], p. 22) shows the BonFIRE architecture where the cloud broker and the portal play the role of a *SET entity*, the data store represents a *REG entity*, and the enactor combined with the cloud API component represent an *M entity*.

The BonFIRE architecture was heavily influenced by that of Teagle, capitalizing on lessons learned by the Panlab projects and the details presented in this thesis. Furthermore, Teagle's DEN-ng based data model and graphical portal components will influence similar components of the BonFIRE release 2. It is currently also under investigation whether it would be desirable and feasible to adopt the Teagle VCT tool in BonFIRE.

However, it has to be noted that large parts of the federation framework instantiation presented in chapter 6 were not used by BonFIRE because of licensing issues and the intended commercial exploitation of the BonFIRE results. Initially, the Teagle components were planned to be released under a more restrictive license to allow for a commercial exploitation by a potential federation organization formed by several core Panlab partners. This was somewhat contradicting the BonFIRE exploitation plans and was a strong motivation for BonFIRE—on the one hand—to learn from the Teagle architecture design and implementation but—on the other hand—to build the cloud broker instantiation from scratch for the first BonFIRE release[15]. Now, as the Teagle components mostly have been released under an Apache License, Version 2.0 that explicitly allows for commercial exploitation, the situation has changed and the BonFIRE consortium currently considers using some of the Teagle components for the second release which is planned for late 2011. This line of reasoning is backed by the fact that BonFIRE intends to federate with selected domains governed by external facilities such as Panlab, NOVI, and TEFIS. Here, Teagle can play an important role as its components are used in all of those projects enabling interoperability.

7.1.4 The FP7 NOVI Project

The FP7 NOVI project[16] is part of the FIRE initiative and started in September 2010 with a duration of 30 months. The NOVI objective is to investigate "[...] efficient approaches to compose virtualized e-Infrastructures towards a holistic Future Internet (FI) cloud service. Resources belonging to various levels, i.e. networking, storage, and processing are in principle managed by separate yet interworking providers. NOVI will concentrate on methods, information systems, and algorithms that will enable users with composite isolated slices, baskets of resources, and services provided by federated infrastructures." [157]

NOVI envisions an architecture that is able to deal with both hierarchical and peer-to-peer federation concepts. Therefore, it is argued that NOVI makes use of the *central scenario* as well as the *distributed scenario*. Furthermore, NOVI relies on the federation methods presented in sections 5.1, 5.2, and 5.4 building upon resource description data models and abstraction techniques, as well as a resource brokering concept. Lymberopoulos et al.—representing the NOVI drivers—cite this thesis' work and state their plans to build upon it: "Indeed, our approach can be viewed as a generalization of the recent work for inter-working among Teagle and SFA, as reported in [27][17]." ([158], p. 7)

At the time of writing, it has not yet been decided which parts of the federation framework instantiation (Teagle) will finally be used for the NOVI implementation. This might also depend on the progress of other projects that are currently contributing to the SFAv2 and Teagle integration, such as the BMBF project G-Lab Deep that is discussed in the next section.

[15] Released in December 2010
[16] NOVI project website: http://www.fp7-novi.eu
[17] Refers to [88]

7.1.5 The BMBF G-Lab and G-Lab Deep Projects

The BMBF phase-one G-Lab project has already been introduced in section 3.9. In the second phase, several smaller projects have been launched to build upon the facility provided by the phase-one project. One of those projects is the BMBF G-Lab Deep project[18]. G-Lab Deep started in September 2009 with a duration of 30 months and aims at investigating a cross-layer composition approach that brokers between application and network layer entities. As a part of this, one of the work packages deals with cross-domain resource federation. Here, the concepts and prototypes delivered by this thesis are used.

In particular, the federation established by G-Lab Deep builds upon the *central scenario* and the *recursive scenario* making use of four of the federation model entities (*SET, REG, M, and R*) and *four* of the methods defined in chapter 5 (see also table 7.3). The *resource interconnection method* described in section 5.5 is currently not used as the main resources federated in G-Lab Deep are *PlanetLab* resources that rely on public links. Essentially, the G-Lab phase-one facility is a *private PlanetLab* installation that is partly extended with additional resources if needed by the phase-two projects. However, interesting concepts are emerging from G-Lab Deep to build secure overlay connections on top of the federated PL resources under the control of Teagle. This would allow for cross-facility packet tracking use cases as envisioned by [159]. Therefore, method 5.5 and the GW model entity might be used by G-Lab Deep at a later stage of the project.

Also, one of the main outcomes of G-Lab Deep is the combination of Teagle with the Slice-based Federation Architecture (SFAv2). This has been published in [88] and further work is currently performed to drive this idea further and provide stable prototypes. A recursive facility architecture has been designed following the scenario described in section 4.4.3. Figure 7.6 ([160], p. ii) demonstrates this.

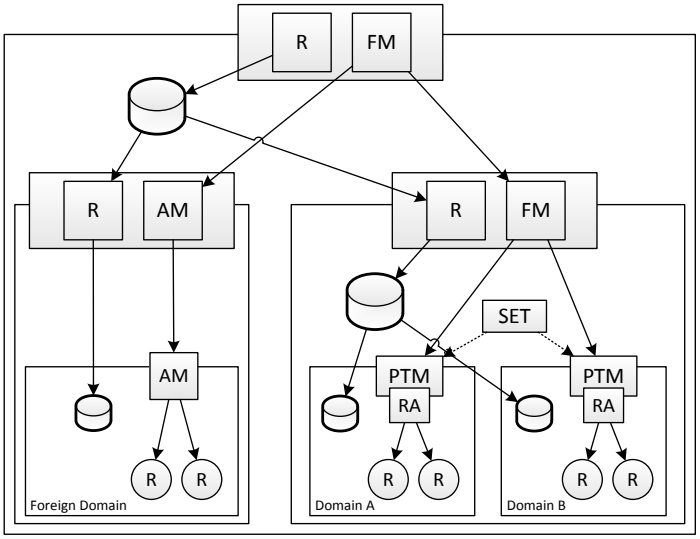


Figure 7.6 – Recursive federation approach building upon Teagle, PLC, and SFAv2

Here, the recursive approach allows federated domains to be part of yet another overarching federation. "All federating domains and all federations of federated domains expose SFA

[18]G-lab Deep project website: http://www.g-lab-deep.de

enabled interfaces which are accessed by the next higher level in the recursive design. The components for the framework are based on principals the SFA defines but with additional functionality in the Federation Manager (FM) regarding the handling of several different federation partners with potentially different resource descriptions. The knowledge about resources is distributed. Every domain has its own set of resource descriptions but the upper level repository knows at least the names of the available resources in the different underlying domains." ([160], p. ii)

In order to implement such an approach the existing Teagle (as described in chapter 6) and SFA prototypes[19] are used and combined as shown by figure 7.7 ([160], p. iii).

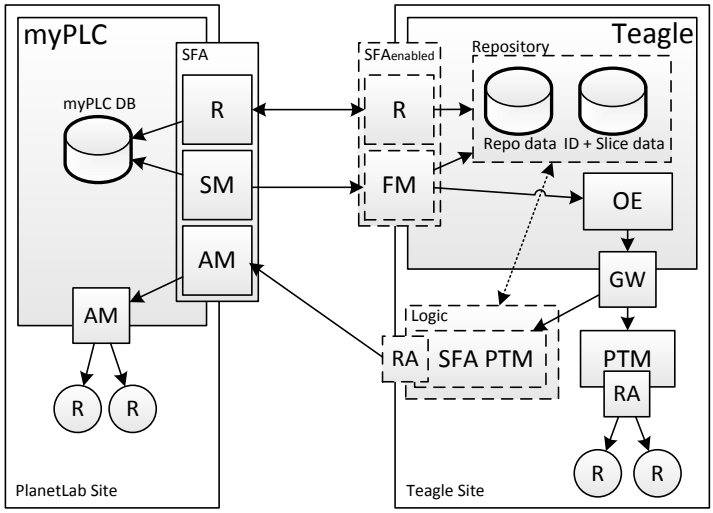


Figure 7.7 – Architecture for federating PLC/SFAv2 based resources with Teagle-controlled domains

The dotted-lined components are currently under development and will extend the framework instantiation as described in chapter 6 in the near future. "The existing Teagle repository will be expanded to store additional information about slices and keep the user certificates which are needed to authenticate the Teagle users against the other federation parties." ([160], p. iv)

The FM will be designed with the recursive use case in mind. For the development, the Fraunhofer FOKUS Teagle playground and selected G-Lab resources (SFA/PLC-based VMs) will be used.

7.1.6 The FP7 OpenLab Project

The FP7 OpenLab[20] project is an ICT Call 7 FIRE initiative project that will start in September 2011. Although the project has not started at the time of writing, it is included here as some sort of outlook. The description of work foresees a very interesting integration and consolidation of different federation approaches (mainly PlanetLab, OMF, SFAv2, and the artifacts presented by this thesis (federation model, methods, and Teagle)). Therefore, some insights into how the artifacts are planned to be used are given here.

[19]SFA overview webpage: http://svn.planet-lab.org/wiki/SFAGuide
[20]OpenLab project website: http://www.ict-openlab.eu

One of the issues related to resource federation and a global adoption today is that different frameworks have emerged at the same time. As an example, while the results of this thesis were produced, similar frameworks have emerged such as the SAFv2, the ORCA framework, the OMF, and the ProtoGENI/Emulab federation framework. Although all of those approaches have slightly different scopes and user communities justifying their existence, global resource federation across several of those facilities becomes difficult due to heterogeneous concepts and technologies. Therefore, the OpenLab project addresses the important aspects of federation framework interoperability. In particular, the aim is to derive "[...] a recursive federation model and implementation that allows for federation at any granularity (e.g. domain, federation, meta-federation). Today, we operate MyPLC-, OMF-, Teagle-, and SFA-based facilities that are federations in themselves and that offer resources provided by multiple sites. Our vision is to enable a federation of federations based on a recursive model, where the same APIs are offered at any level. [...] In order to federate heterogeneous resources across administrative boundaries, we need to address resource description (RSpec), policy description, negotiation, and enforcement (PSpec), identity management (ISpec), and control framework interfaces (CSpec). Today, the different frameworks handle such issues independently by using different mechanisms and technologies. This task aims at specifying and implementing the minimal common denominator that would allow the frameworks to inter-operate in terms of the R-, P-, I-, and CSpec." ([161], p. 28-29)

This shows the high relevance of the results presented by this thesis for the upcoming FP7 OpenLab project. It is clear that further work will build upon this thesis' design artifacts, demonstrating the impact on the experimentally-driven Future Internet research community that requires federated large scale experimental facilities.

7.1.7 Field Study Summary and Further Analysis

Table 7.3 summarizes the field study by comparing the different projects in terms of the number of federated domains, the federation model entities, the scenario, the methods, the federated resources, and the tools and prototypes used.

Table 7.3 – Observational evaluation: usage of the artifacts in multiple projects

Project	Number of domains	Federation model entities[21]	Federation scenario[22]	Federation methods[23]	Federated resources	Tools & prototypes
PII	15	SET, REG, M, R, GW	4.4.1, 4.4.4 planned	5.1, 5.2, 5.3, 5.4, 5.5	Physical nodes, virtual nodes (XEN), various software resources (e.g. IMS, DNS, databases, web servers, etc.), Federica VMs, OCCI and-based cloud resources, GSN and HPDMnet resources	Teagle, FCI[24]
TEFIS	6	SET, REG, M, R, GW	4.4.1, 4.4.4[25]	5.1, 5.2, 5.3, 5.4, 5.5	ProActive PACA Grid, AQCM[26], Emulated and full IMS, Botnia Living Lab resources, Kyatera high capacity optical network resources, PlanetLab resources, software	Experiment manager, Teagle, PLC, iRODS[27]
NOVI	> 5	SET, REG, M, R	4.4.2, 4.4.4[25]	5.1, 5.2, 5.4	Virtualized network infrastructure, PlanetLab resources, Federica resources, GEANT resources	SFAv2, Federica control plane implementation, Teagle (not yet decided)

To be continued on the next page ...

[21] As described in chapter 4
[22] As described in section 4.4
[23] As described in the respective sections of chapter 5
[24] Federation Computing Interface (FCI), as published at http://trac.panlab.net/trac/wiki/FCI
[25] only PLE domains
[26] Automatic build, test and quality certification (Automated Quality Certification Model (AQCM)), as presented at ISO WG as evaluation module for any distributed software exploiting distributed resources
[27] integrated Rule Oriented Data System (iRODS), https://www.irods.org/index.php

... table 7.3 continued from the previous page

G-Lab Deep	2[28]	SET, REG, M, R	4.4.1, 4.4.3	5.1, 5.2, 5.3, 5.4	Virtual machines & appliances, software, private PlanetLab resources, ToMaTo-based topologies[28], OpenFlow resources planned	PLC, SFAv2, Teagle, ToMaTo[29]
BonFIRE	5	SET, REG, M, R	4.4.1	5.1, 5.2, 5.4	Cloud resources, specific focus on IaaS, PaaS, and custom virtual appliances, IBBT Virtual Wall resources. Federation with other facilities is planned to include fixed network resources.	OpenNebula[30], CloudBroker
OpenLab[31]	> 5	SET, REG, M, R, GW	4.4.2, 4.4.3, 4.4.4	5.1, 5.2, 5.4	Wireless (e.g., Wi-Fi, WiMAX, 3G, LTE) & wired resources, PlanebLab/SFA-based resources, Panlab resources	SFAv2, PLC, Teagle, OMF

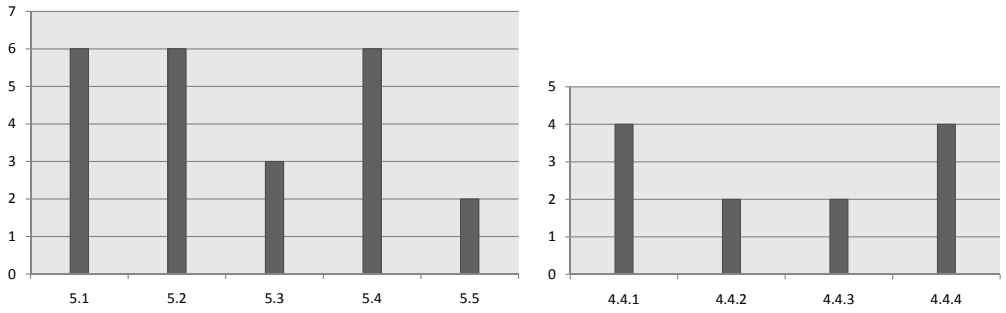
[28] The overall G-lab experimental facility builds upon more domains, as described in section 3.9, but for the work performed in the G-Lab Deep project, only the domains of Fraunhofer FOKUS and the University of Kaiserslautern are involved until now.

[29] Topology Management Tool (ToMaTo), [111]

[30] http://www.opennebula.org

[31] At the time of writing, this project has not yet started, but the description of work foresees a very interesting integration and consolidation of different federation approaches (mainly PlanetLab, Orbit, SFAv2, and the artifacts presented by this thesis (federation model, methods, and Teagle)). Therefore, the OpenLab project has been included in this field study. A website is already available at http://www.ict-openlab.eu.

Further, figure 7.8 compares the methods introduced in chapter 5 and the federation scenarios introduced in section 4.4 in terms of the total number of projects that make use of them. Here, all six projects use a *resource modeling* or *description* approach together with some sort of *resource abstraction* and *control framework*. Therefore, a potential federation interoperability approach as planned by the OpenLab project would have to target those aspects and homogenize as much as possible regarding resource description and abstraction techniques and technologies or work towards an advanced mapping approach. Regarding the federation scenario, most projects seem to favor a *centralized* approach with four out of six projects investigating a central solution. Please note that regarding the consortium scenario (4.4.4), this is only planned in case of Panlab and in case of the other projects it only applies to a subset of resources, namely the PlanetLab Europe (PLE) resources.



(a) Methods (b) Scenarios

Figure 7.8 – Usage of the different federation methods and scenarios described by this thesis. The abscissa plots the section where the respective method/scenario is described. The ordinate of both sub-figures plots the corresponding number of projects as revealed by this field study.

7.2 Experimental Evaluation: A Controlled Experiment

Among the experimental evaluation methods are *controlled experiments* and *simulation*. Table 7.4 ([11], p. 86) explains the difference. This section describes the use of the controlled

Table 7.4 – Experimental evaluation methods

Category	Evaluation methods
Experimental	Controlled Experiment: Study artifact in controlled environment for qualities (e.g., usability)
	Simulation: Execute artifact with artificial data

experiment method to evaluate some of the design artifacts. The term *experiment* is used here in a double meaning because we experiment with the process of utilizing the artifacts to execute an experiment. This means that several experiment runs will be executed upon federated infrastructure that is provided by making use of the design artifacts. The federation methods (based on the federation model) and the federation system instantiation *Teagle* will

be used to provide the required virtual infrastructure to *stress test* an IMS core relying on different federated infrastructure configurations.

7.2.1 First Experiment Run

Figure 7.9 shows the initial virtual resource grouping configuration that has been designed using the VCT tool[32]. This will be the system under test (SUT).

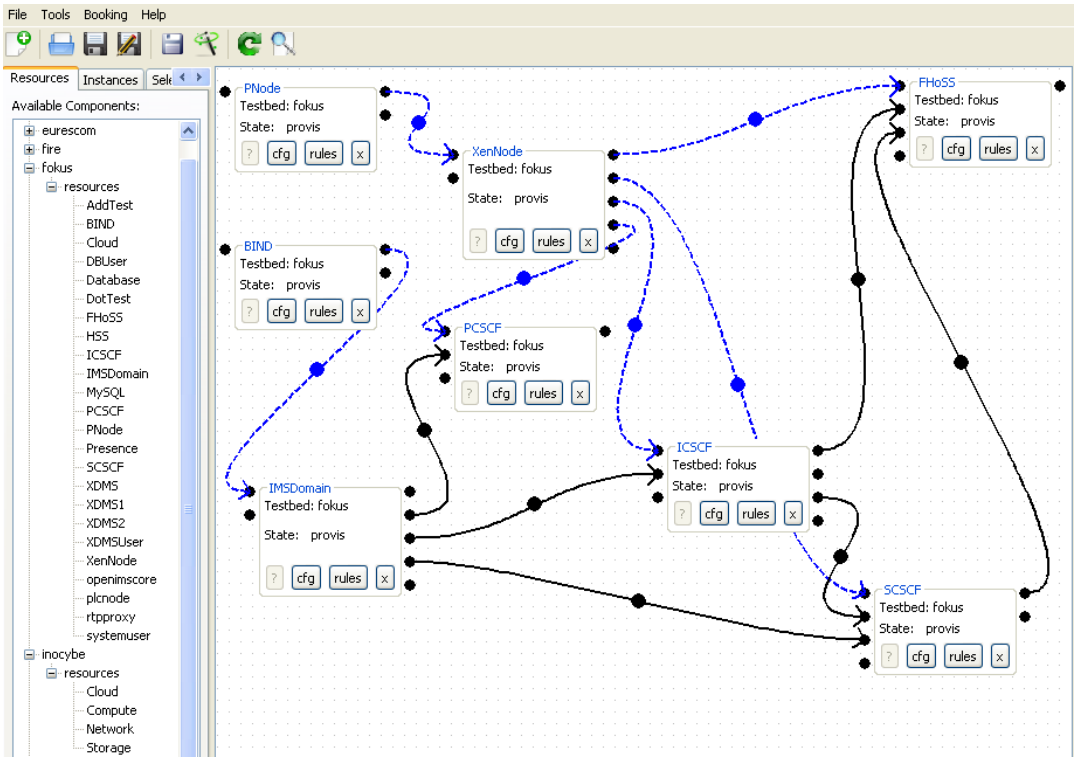


Figure 7.9 – Controlled experimental evaluation: the initial virtual resource grouping design

This virtual resource grouping (VG) can be formally described as follows:

$$VG = \{R_{fokus_x} \in MR_{fokus} \mid x = \{1, 2, ..., 8\}\} \tag{7.1}$$

with:

$$\begin{aligned} R_{fokus_1} &= \text{PNode} & R_{fokus_2} &= \text{XenNode} & R_{fokus_3} &= \text{FHoSS} \\ R_{fokus_4} &= \text{PCSCF} & R_{fokus_5} &= \text{ICSCF} & R_{fokus_6} &= \text{SCSCF} \\ R_{fokus_7} &= \text{BIND} & R_{fokus_8} &= \text{IMSDomain} \end{aligned} \tag{7.2}$$

where the following hierarchical resource relationships apply (represented by the dotted lines

[32]Please note that this version of the VCT tool displays the *resource type* instead of the full resource identifier as opposed to the version used to generate e.g. figure A.1. In order to see the full identifier, the `cfg` button would have to be clicked. Also, this version does not yet support the display of configuration reference semantics.

in figure 7.9):

$$\begin{aligned} p(R_{fokus_2}) &= R_{fokus_1} & p(R_{fokus_3}) &= R_{fokus_2} \\ p(R_{fokus_4}) &= R_{fokus_2} & p(R_{fokus_5}) &= R_{fokus_2} \\ p(R_{fokus_6}) &= R_{fokus_2} & p(R_{fokus_8}) &= R_{fokus_7} \end{aligned} \tag{7.3}$$

and where the following intra-domain resource reference relationships apply (represented by the solid lines in figure 7.9):

$$\begin{aligned} r_{intra}(R_{fokus_5}) &= R_{fokus_3} & r_{intra}(R_{fokus_6}) &= R_{fokus_3} \\ r_{intra}(R_{fokus_8}) &= R_{fokus_4} & r_{intra}(R_{fokus_8}) &= R_{fokus_5} \\ r_{intra}(R_{fokus_8}) &= R_{fokus_6} & r_{intra}(R_{fokus_5}) &= R_{fokus_6} \end{aligned} \tag{7.4}$$

Remember, that p denotes a function that assigns a resource y to be the parent resource of x and r denotes a function that assigns a resource x to rely on the configuration of resource z. Hierarchical relationships currently do not work across domains, resulting in:

$$p : MR_A \rightarrow MR_A,\ x \mapsto y \tag{7.5}$$

while configuration references are implemented to work within a single domain:

$$r_{intra} : MR_A \rightarrow MR_A,\ x \mapsto z \tag{7.6}$$

or across domains:

$$r_{inter} : MR_A \rightarrow MR_B,\ x \mapsto z \tag{7.7}$$

This virtual resource grouping has been booked according to the procedures described in chapter 5 and 6, going through the entire process of virtual resource grouping design, orchestration, and deployment performed by Teagle and the underlying federated domains running the DM and RA software. Based on this instantiation of the federation framework, an open source IMS core[33] has been provisioned on a virtual machine (XenNode) as defined by the equations 7.3. The configuration of the IMS components has been performed using the VCT tool and is realized by the references defined in the equations 7.4. This is possible through the method of federated resource abstraction as described in section 5.2 and the abstraction layer implementation (DM and RAs) as described in section 6.6.1 and 6.6.2 for all involved resources. The abstract resource of an *IMS domain* is handled using a DNS server (R_{fokus_7}) for which a parent relationship has been left unspecified. Therefore, it is up to the *domain* to choose a proper container. In this case, the *PNode* is chosen by the DM to host the BIND server, but this decision is abstracted on the Teagle layer. Please note that in the following—in order to shorten the description of the experiment—only those resource configurations that primarily impact the results of the experiment are given.

The IMS core has essentially been configured with default values[34]. The `XenNode` resource instance has been configured using the VCT tool with:

```
1 <cpu type="int">1</cpu>
2 <mem type="int">384</mem>
```

[33] Fraunhofer FOKUS open source IMS core project website: http://www.openimscore.org

[34] Open source IMS core configuration: http://www.openimscore.org/installation_guide#step4

This is important because the XEN resource, running a Gentoo Linux, hosts all of the IMS services (P-CSCF, S-CSCF, I-CSCF, and HSS) which will create heavy load on this machine, once the stress-testing procedure will be started. The machine that executes the load generation is in this case (although possible) *not* managed by Teagle in order to simplify the scenario. For the load generation, a Lua script[35] is used that is executed using SIPNuke[36]. A shortened form of the Lua script (omitting the function definitions) is shown in listing 7.1.

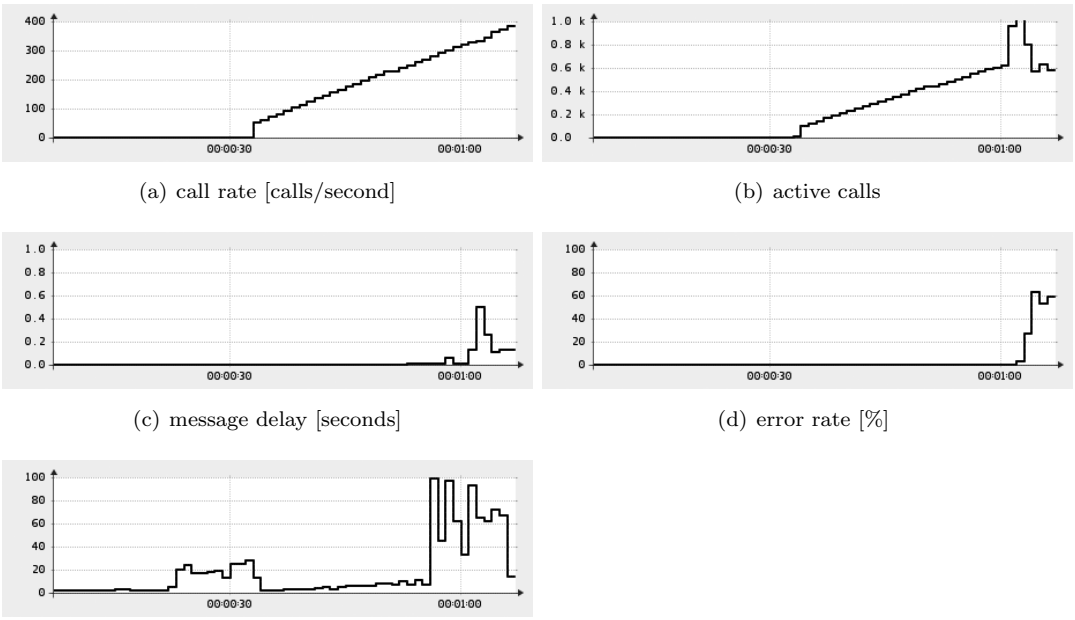
Listing 7.1 – Lua script (shortened)

```
local cfg = {
   format = {user="sip:user%d@open-ims.test", password="user%d"},
   destination = {ip="193.174.152.197", port=20001},
   userCount = 50,
   from = 50,
   to = 400,
   step = 10,
   duration = 1,
   regExpires = 300,
   regRate = 5,
   answer = true
}

log.warn"* creating endpoints"
endpoints = Endpoints{count=cfg.userCount, format=cfg.format, destination=cfg.destination}
for i,e in ipairs(endpoints) do
   e.on_method.INVITE = {cb.fork, handle_incoming_call, e}
end

log.warn"* registering"
reg_all(cfg.regExpires)

log.warn"* calling"
local stamp = event.now()
local uac_id = 0
for delay, rate in rate_step(cfg.from, cfg.to, cfg.step, cfg.duration) do
   local uac = endpoints[1 + uac_id % #endpoints]
   local uas = endpoints[1 + (uac_id+1) % #endpoints]
   uac_id = uac_id + 1
   cb.fork(dial_call, uac, uas.user_info.impu)
   stamp = stamp + delay
   sync.sleep(stamp - event.now())
   stat_show{rate=rate, calls=threads}
end

while threads > 0 do
   event.run()
   stat_show{calls=threads}
end

log.warn"* unregistering"
reg_all(0)
```

Once the stress-test is initiated, the script registers a total number of 50 IMS user identities with a registration rate of five users/second and initiates calls between them with

[35]The Lua scripting language website: http://www.lua.org
[36]SIPNuke project website: http://www.sipnuke.org

a duration of one second. The call rate is steadily increased by 10 calls/second from 50 calls/second to a maximum of 400 calls/second, looping through all of the registered users. This creates considerable CPU[37] load and memory usage on the XEN machine (R_{fokus_2}). This is monitored and logged using local processes. Figure 7.10 shows the results of the first experiment run.



(a) call rate [calls/second]

(b) active calls

(c) message delay [seconds]

(d) error rate [%]

(e) CPU load XEN node [%]

Figure 7.10 – Stress-test metrics: the abscissa of all sub-figures plots the elapsed time (ISO 8601 extended format: [hh]:[mm]:[ss]). The ordinate plots the respective measurement.

From those, it can be observed, that at about 280 calls/seconds, the CPU load on the XEN machine reaches 100% (see sub-figure (e)). Also, the IMS services run out of memory resulting in unstable and unpredictable IMS core message processing. At this stage, high delays and error rates are observed as shown by sub-figures (c) and (d) and the experiment had to be terminated manually by stopping the Lua script execution and the IMS core services processes. The conclusion at this stage is that this installation of the IMS core cannot support call rates much higher than 250 calls/second due to the under-performing underlying virtual infrastructure. This will be improved by re-provisioning the virtual resource grouping in a different configuration for the second experiment run.

Although this experiment has been executed primarily to demonstrate, evaluate, and experiment with this thesis' artifacts, such stress-testing execution would be useful to analyze the IMS core behavior, e.g. to determine the optimal IMS core services configuration in different load situations and for different underlying (virtual) infrastructure configurations.

[37]The CPU controlled by the XEN hypervisor is an Intel® Xeon® CPU X3363 @ 2.83GHz (Quad Core). Also, the physical machine disposes of 8GB of RAM.

7.2.2 Second Experiment Run

The virtual resource grouping has been re-configured using Teagle to incorporate additional cloud computing resources provided by a second domain. Figure 7.11 shows a VCT tool screenshot of the modified virtual resource grouping design.

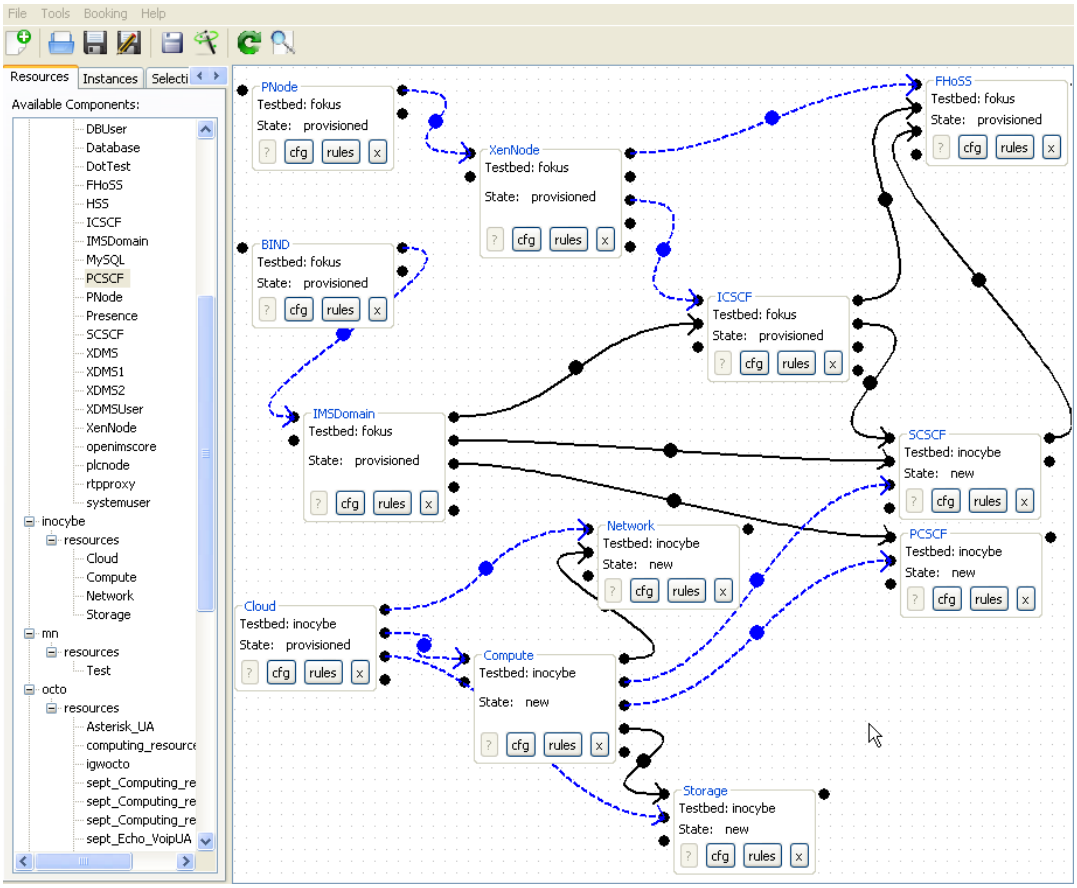


Figure 7.11 – Controlled experimental evaluation: the *modified* virtual resource grouping design. The migrated CSCF services are now hosted on an OCCI-based cloud resource provided by a second domain.

Following this layout, the most resource consuming IMS services, the P-CSCF and the S-CSCF, have been migrated to an OCCI-based compute resource while the other IMS components still reside on the small XEN machine. The cloud resources are provided using RA implementations following the emerging OCCI standard [35] to control network, compute, and storage resources. Therefore, the modified VCT VG_{mod} can be formally described as follows:

$$VG_{mod} = \{R_{fokus} \in MR_{fokus} \ \wedge \ R_{inocybe} \in MR_{inocybe}\} \tag{7.8}$$

with the new resources

$$\begin{aligned} R_{inocybe_1} &= \text{Cloud} & R_{inocybe_2} &= \text{Network} \\ R_{inocybe_3} &= \text{Compute} & R_{inocybe_4} &= \text{Storage} \\ R_{inocybe_5} &= \text{SCSCF} & R_{inocybe_6} &= \text{PCSCF} \end{aligned} \tag{7.9}$$

for which the following hierarchical resource relationships apply:

$$\begin{aligned} p(R_{inocybe_2}) &= R_{inocybe_1} & p(R_{inocybe_3}) &= R_{inocybe_1} \\ p(R_{inocybe_4}) &= R_{inocybe_1} & p(R_{inocybe_5}) &= R_{inocybe_3} \\ p(R_{inocybe_6}) &= R_{inocybe_3} \end{aligned} \tag{7.10}$$

and for which the following *intra*-domain resource reference have been defined:

$$r_{intra}(R_{inocybe_3}) = R_{inocybe_2} \qquad r_{intra}(R_{inocybe_3}) = R_{inocybe_4} \tag{7.11}$$

as well as the following *inter*-domain references:

$$\begin{aligned} r_{inter}(R_{inocybe_5}) &= R_{fokus_3} & r_{inter}(R_{fokus_8}) &= R_{inocybe_5} \\ r_{inter}(R_{fokus_8}) &= R_{inocybe_6} & r_{inter}(R_{fokus_5}) &= R_{inocybe_5} \end{aligned} \tag{7.12}$$

This means that the resource demanding services, the PCSCF and the SCSCF, have been migrated to a different network domain but are still part of the same IMS domain defined by R_{fokus_8}. The migration has not been performed in a live fashion meaning that the services have been restarted between the first and the second Lua script execution.

The OCCI compute resource, relying on an Intel® Xeon® CPU X3363 @ 2.83GHz (Quad Core) processor, runs a Gentoo Linux and has been configured with the following parameters:

```
1 <cpu type="int">4</cpu>
2 <mem type="int">4096</mem>
```

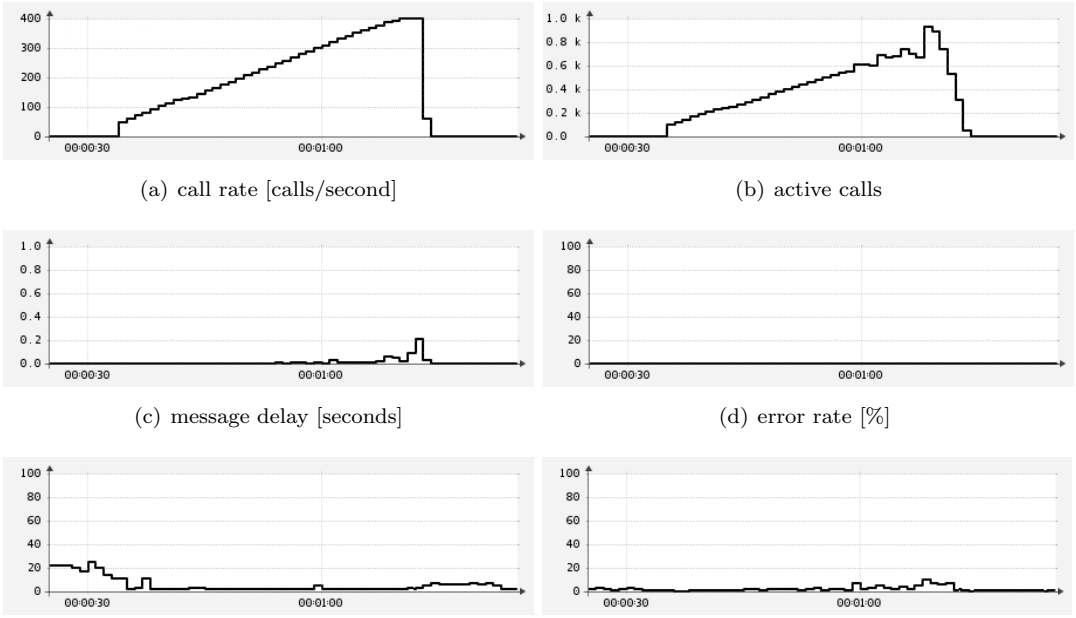
Compared with the XEN node configuration from the first experiment execution, this is far more available processing power and memory leading to the assumption that the IMS services should perform much better during the second experiment run.

Note, that for the second domain *inocybe*, two different network options have been used. For the results shown by figure 7.12 and its sub-figures, the two domains *inocybe* and *fokus* were interconnected using a FOKUS in-house Gigabit-Ethernet network. However, the experiment was also executed using the same resource types (but different *instances*) hosted by our partners from Inocybe[38] in Canada using the public Internet as a means to connect the relevant resources of both domains. Before coming back to the network issues, let's take a look at the results from the second experiment execution shown in figure 7.12.

Note, that the charts do not plot the entire experiment execution time. The initial XEN node CPU load shown by sub-figure (e) results from registering the 50 IMS users. As the HSS (R_{fokus_3}) still resides on the XEN node as initially configured, it creates some load on this machine (also compare with figure 7.10, sub-figure (e)) during the IMS user registration process. For formatting reasons and to avoid redundant information, the charts start around the 25 seconds mark. At this time, the user registration at the HSS is already ongoing.

As in the first run, the call rate is increased from 50 to 400 calls/second. After the calls have been terminated and the call rate has dropped back to zero calls/second, the user de-registration is performed resulting in some CPU load on the XEN machine around the one minute and 20 seconds mark. The experiment terminates successfully and a call rate of 400 calls/second could be handled by the cloud compute resource with a maximum CPU usage of

[38]Inocybe Technologies Inc. website: http://www.inocybe.ca/frontpage/home



(a) call rate [calls/second]

(b) active calls

(c) message delay [seconds]

(d) error rate [%]

(e) CPU load XEN node [%]

(f) CPU load OCCI compute node [%]

Figure 7.12 – Stress-test metrics of the second run: the abscissa of all sub-figures plots the elapsed time (ISO 8601 extended format: [hh]:[mm]:[ss]). The ordinate plots the respective measurement. Subfigure (e) plots the CPU load of the new cloud resource hosting the migrated PCSCF and SCSCF services.

below 20% around the one minute and 15 seconds mark. No errors and only small message delays have been observed.

As stated before, the second experiment run has additionally been performed in a different setup using the public Internet to connect the two resource provider domains *inocybe* and *fokus*. For this to work, our partners at Inocybe ran a DM equipped with the same RAs that were used at the *fokus* domain. Using a public address space allowed to directly interconnect the IMS services. The execution showed that the public network could not support call rates above 100 calls/second without high delays. In fact, applying the XEN machine configuration (1 CPU, 384 MB memory) to the OCCI compute resource did not result in the IMS services to crash as in the first experiment run because the network turned out to be the bottleneck instead of the CPU or memory. This is an important result as it impacts the vision of heterogeneous resource federation as a whole. Unless there are experimental networks that can be dynamically controlled by a federation just as any other resource in terms of dedicated network path setup and connection parameter configuration, many experiments are heavily restricted or bound to fail. This conclusion lead us to look at federating network resources in more detail, as described in [112] on page 57/58 and investigate alternative connectivity solutions using OpenFlow [162].

7.2.3 Summary

The configuration of a full-fledged IMS core system is a time-consuming and error-prone process. On top, the cross-domain migration of individual IMS services and the necessary re-

configuration of the entire environment can delay the collection of testing and experimentation data considerably. The generic resource federation model and methods presented by this thesis and the Teagle-provided services relying on federated resources, enable a user to work with stable default resource configurations and adapt those at any desired level of granularity (as long as this is supported by the RA implementations). This allows to speed up the entire experimentation process and therefore allows to cut down the associated cost. This shows that the concept and the accompanying implementation can handle complex cross-layer and cross-domain resource federation scenarios providing a clear benefit to the involved stakeholders. The IMS stress-testing experiment was chosen here as an example because the required RA implementations were already available. In a similar manner, any other technology that can be controlled using the resource abstraction concept, could be targeted. In this sense, federations and their system implementations have to adapt constantly to changing user demands.

7.3 Comparison with Other Approaches

This section compares the results of this thesis with other relevant approaches as summarized by table 7.5. All of the other six approaches that are part of this analysis have been introduced in chapter 3. The following description shortly explains the different comparison criteria and gives some concluding remarks:

Resource description This criterion marks the approach taken to describe the federated resources. The analysis shows that very different paradigms and technologies are used for describing federated resources. The applicability of those to highly heterogeneous resources is not given for most approaches as their focus is limited to rather homogeneous resources.

Resources This indicates the level of heterogeneity of the federated resource types. It can be observed that most other approaches target a considerably smaller scope of federated resource types compared with the resources that can be federated using the concepts and system prototypes described by this thesis.

Resource discovery This indicates how the federated resources can be discovered. Most approaches choose a similar concept where both domain and central resource registries are used to collect information about resources available across the federation.

Resource request tools This lists the different tools that are available to express a request for federated resources. In case no specific tools are developed, all of the compared approaches specify an interface for 3rd party tools to interact with. This thesis enables both.

Experiment description and control As most of the compared frameworks have been designed and built to support scientific experiments, this criterion looks at the methods and tools that are available to describe and control the execution of an experiment. Only two of the other approaches (OMF and DFA) address this challenging task and both of them primarily deal with rather homogeneous resource types. This thesis did not explicitly address experiment description and control due to the high resource heterogeneity which results in a very high number of potential experiment types and usage cases. However, interestingly, an additional development has emerged based

on this thesis' results called FCI[39] which aims at closing this gap making use of the information model presented here as well as model-to-model transformations, enabling a flexible processing of the resource information available through the Teagle repository.

Policy support This criterion analyses each approach in terms of the possibility to express resource specific or global policies. Most approaches support domain specific policies without global alignment. This thesis defines a central policy engine that deals with resource specific, provider specific, and global policies.

Resource control This investigates the control framework concepts and technologies. Most approaches, including this thesis, use an XML-based control messaging framework following the CRUD paradigm.

Abstract resource relationships This criterion looks at the resource abstraction capabilities of the different frameworks. Although all of the approaches allow for resource abstraction, only few of them allow for the definition of abstract resource relationships. With the possibility to express p, r_{intra}, and r_{inter} resource relationships (see section 5.3), this thesis proposes a new way to flexibly define virtual resource groupings and interact with a pool of federated resources.

Resource orchestration This analyses in how far the virtual resource grouping derivation and configuration process can be automated using resource orchestration capabilities. Most of the other approaches rely on a manual binding of resource instances to a specific virtual resource grouping. Based on the abstract resource relationships discussed in the previous paragraph, dynamic dependency resolution, and the specification of a dedicated resource orchestration engine, this thesis proposes a unique approach to generically orchestrate federated resources into virtual groupings.

Dynamic resource deployment decisions This criterion indicates in how far dynamic decisions can be taken during the resource deployment process. This is essentially a feature that relies on extensive resource abstraction and structured resource description. It allows to satisfy a more or less fuzzy request by taking specific inter- or intra-domain decisions. To the best of my knowledge, this is not addressed by any of the other approaches, making it a unique characteristic of the artifacts presented here allowing to dynamically take global (inter-domain) and local (intra-domain) federated resource deployment decisions whenever the relevant details have been left unspecified by the requester.

The table 7.5—spanning across the following two pages—compares the SFAv2, OMF, DFA, ProtoGENI, ORCA, and Wisebed approaches with the concepts and prototypes proposed by this thesis.

[39] Federation Computing Interface, http://trac.panlab.net/trac/wiki/FCI

Table 7.5 – Comparison of different federation approaches

	SFAv2	**OMF**	**DFA**	**ProtoGENI**	**ORCA**	**Wisebed**	**This thesis**
Resource description	out of scope	extended PlanetLab RSpec	network simulator (ns), Emulab ptop/vtop format	ProtoGENI RSpec v2	NDL-OWL	WiseML	model-based approach, extended DEN-ng IM
Resources	out of scope	homogeneous: wireless	rather homogeneous: nodes, links	rather homogeneous: nodes, links	rather heterogeneous: nodes, storage, network	homogeneous: wireless	heterogeneous
Resource discovery	aggregate registries	aggregate registries, OMF inventory	centralized	centralized, global knowledge due to small number of federating entities	broker mediates resource discovery using an inventory	site registry and aggregated registry	domain registries and central registry, Teagle repository
Resource request tools	out of scope, AM interface can be used	3rd party tools can be used (e.g. SFI)	Security Experimentation EnviRonment (SEER)	users interact with ProtoGeni XMLRPC servers using python or perl	Web portal, Automat	TARWIS	VCT Tool
Experiment description and control	out of scope	OMF Experiment Description Language (OEDL)	ns/Emulab formats, Canonical Experiment Description Language (CEDL)	out of scope	out of scope, can be integrated, e.g. using GUSH experiment controller	operations component supporting runtime updates	out of scope but an additional development (FCI) based on this thesis' work is addressing this
Policy support	out of scope, aggregate specific policies possible, sfatables concept under investigation	different OMF sites have different access and reservation policies, better alignment is planned	site usage policy DB and management tools	individual site policies, no global alignment	balance of local autonomy and global coordination, no central point of trust or control	resource-specific custom policies	resource-specific, provider-specific, and global policies, Teagle policy engine

To be continued on the next page ...

... table 7.5 continued from the previous page

Resource control	CRUD, XML-RPC	several, depending on the deployed AM; REST, XML-RPC	SOAP, XML-RPC	XML-RPC	SOAP, WS-Security, XML-RPC	proprietary, SOAP	T_1 interface, CRUD, REST, SOAP, T_2 interface, XML-RPC
Abstract resource relationships	out of scope	out of scope	ns and Emulab topology semantics	Emulab semantics, otherwise out of scope	possible based on semantic resource descriptions and arbitrary levels of abstraction; currently not implemented	out of scope	abstract resource hierarchies and configuration references possible, see section 5.3
Resource orchestration	manual slice-based binding, otherwise out of scope	out of scope, but time- and location-based experiment orchestration possible	topology description language	out of scope	leasing core orchestrates workflow to acquire and manage leased resources, orchestration of end-to-end slice configuration and stitching planned	detailed instantiation information of nodes and links is part of the resource request, otherwise out of scope	cross-domain and cross-layer orchestration based on resource model and abstract relationships, Teagle orchestration engine
Dynamic resource deployment decisions	out of scope, abstracts from dynamic intra-domain decisions	out of scope	out of scope	out of scope	dynamic intra-domain decisions foreseen, global knowledge only includes resource leases	out of scope	both inter-domain (Teagle) and intra-domain (DM) decisions supported

8 Conclusion

Aristotle postulated in his Metaphysics that *"the whole is more than the sum of its parts"*. It is precisely this philosophical standpoint of *holism* or *emergentism* that justifies the efforts spent on researching the architecture and properties of complex systems. The generic framework for heterogeneous resource federation presented by this thesis classifies as such a complex system. As the main research results, this thesis delivers three artifacts: a federation model, a set of federation methods, and a framework instantiation.

The *federation model* defines five conceptual entities[1], two abstract concepts[2], and three roles[3]. It is used to define four federation modes[4] as well as four federation scenarios[5]. All of this has been introduced in chapter 4.

The *federation methods* build upon the federation model and postulate five important processes to solve the problem of federating heterogeneous resources across different provider domains. In particular, the methods address the problems of information modeling and resource description[6], resource abstraction and control[7], resource relationships[8], resource orchestration and provisioning[9], as well as cross-domain resource interconnectivity[10]. All of this has been introduced in chapter 5

The *framework instantiation* includes the federation system design, instantiation, and evaluation. The system design and instantiation have been described in chapter 6 including an extensive stakeholder requirements analysis. The result of those efforts that have been driven by multiple projects manifests itself in a software framework implementation called *Teagle*. Several people contributed to this open source implementation[11] that is used in a number of currently running projects and is planned to be extended by upcoming activities (see also outlook in section 8.2). Also, the implementation includes a resource abstraction and

[1] SET, M, REG, GW, and R as described in section 4.1

[2] *Domain* and *virtual resource grouping* as described in section 4.1

[3] *Resource provider*, *federation organization*, and *federation user* as described in section 4.2

[4] *Identity federation*, *control framework federation*, *federated resource description*, and *federated policy description* as described in section 4.3

[5] *Central scenario*, *distributed scenario*, *recursive scenario*, and *consortium operation*, see section 4.4

[6] See section 5.1

[7] See section 5.2

[8] See section 5.3

[9] See section 5.4

[10] See section 5.5

[11] Including the components: Teagle portal (section 6.5.2), repository (section 6.5.3), VCT Tool (section 6.5.4), PE (section 6.5.5), RP (section 6.5.6), OE (section 6.5.7), and TGW (section 6.5.8).

management framework[12] that can be installed by resource providers to join the federation. A system evaluation has been introduced in chapter 7 including a field study representing an observational evaluation and a controlled experiment representing an experimental evaluation.

One of the main lessons learned from the research performed as part of this thesis is that generic resource federation concepts indeed constitute a feasible approach for collective resource sharing but seem to be primarily *applicable in an academic environment*. All of the initiatives that make use of the artifacts presented here, receive public funding and primarily attract academic users. Fraunhofer FOKUS contributes to those initiatives by providing open testbeds and prototypes that can be federated with other facilities. On the other hand, FOKUS delivers testbeds to industrial organizations worldwide. In fact, it is probably fair to say that—in addition to being part of *open federations*—we also run a "testbed export" business because our in-house environments are replicated at our industrial customer's premises to serve joint R&D activities. While some of those installations are federated and opened to third parties (e.g. to the local academia and developer communities), many of them are *closed industrial environments* where testing and experimental results are produced for and restricted to company-internal use.

This clearly demonstrates two extremes: open federation vs. closed in-house setups. A potential explanation for this phenomenon can be given by looking at the different market forces that affect academic users on the one side and industrial users on the other side. While academic users are usually restricted in terms of available resources to perform complex and large scale experiments, industrial users are more likely to rely upon a better resource supply in terms of testbeds and experimental facilities. Furthermore, academic users are obliged to produce high quality research results and publish the derivation process while industrial users usually work towards the release of new and innovative products. This shows the clear difference between the two user groups. While academic users benefit from using public federated infrastructure and the potential reproducibility of their results once they have been published, it is quite the opposite for industrial users. Therefore, the following conclusion can be derived: the closer to the market we get in terms of testing and experimentation, the more difficult it becomes to keep federated infrastructure attractive. Here, considerable efforts are necessary to unsure extremely high standards of security including privacy and experiment data confidentiality to establish the necessary level of trust between an industrial user and the federation. On the other hand, more and more industrial players realize the power of openness, capitalizing on effects like open developer communities, user generated content, and the "perpetual beta".

The above is well-reflected by the ongoing discussions in the FIRE architecture board around the *sustainability* of the FIRE facility. Most of the currently running or recently ended FIRE projects struggle with the question of how to ensure the *sustainability* of the experimental infrastructure build in the course of the respective projects once the funding provided by the EC has come to an end.

Such questions go beyond the technical scope of this work and have not been investigated in depth. However, based on the experience gained while addressing the main topics of this thesis, the following answer can be given. The federation organization should be build as one of the stakeholder roles described in section 4.2 together with a group of strong and committed resource providers that are well-known for providing testing and experimentation

[12]The domain manager (see section 6.6.1), the resource adapters (see section 6.6.2), and the interconnection gateway (see section 6.7)

facility services on a commercial basis. Building such a federation would open up an additional sales channel (through the federation) for the resource providers without taking high risks associated with joining the federation. Once a strong and stable nucleus of federation partners has been established, including the necessary legal, operational, and trust relationships, the federation organization needs to build and provide the technical federation infrastructure. This is an important step that would probably require additional investments in order to enable the transition from prototype components to a solution that is able to support productive commercial operation. Given that the above steps can be performed, a small but strong federation nucleus could be well-prepared to extend the initial scope of the federation by including additional partners in order to extend the pool of available resources upon user demand. In this way, the federation could evolve in a *demand-driven way* building upon a stable foundation. Existing and future infrastructural investments could be justified and pay back through the additional federation sales channel and the enlarged user community.

8.1 Conclusion and Impact

In the introduction, the following hypothesis was formulated using a single sentence:

1. To *effectively* and *efficiently federate heterogeneous resources* across the boundaries of administrative domains, enabling *flexible cross-domain resource collaboration*, ...
2. ... a *federation model* can be designed that allows to define a *generic resource federation methodology* and instantiate an according *system solution*, ...
3. ... building upon a *federated resource control framework* and using *resource description, abstraction, orchestration*, and *provisioning techniques*.

In the course of the research work performed for this thesis, it was tried to validate the hypothesis. Based on the chapters 4, 5, 6, and 7, the following conclusions can be drawn:

1. Regarding the first part of the hypothesis, it can be concluded that the artifacts presented here indeed allow for effective cross-domain resource federation. Using the artifacts, it is possible to model and describe virtual resource groupings drawing upon resources provided by multiple administrative domains. By abstractly defining resource dependencies, arbitrary combinations of federated resources can be provisioned efficiently relying on a resource deployment automatism. Therefore, it can be concluded that the first part of the hypothesis can be corroborated.

2. Regarding the second part, it can be concluded that it was possible to design an abstract and generic resource federation model that could be used to define several federation modes and scenarios and to abstractly describe them. The model serves to classify different federation approaches and compare them in terms of the conceptual and architectural layout. This allows for a structured approach to the problem of generic resource federation. In addition, the federation model serves as the basis for the definition of a generic resource federation methodology. Both the federation model and the methodology allowed to approach and implement a complex system solution in a well-structured manner. Therefore, it can be concluded that the second part of the hypothesis can also be corroborated.

3. Regarding the third part of the hypothesis, it can be concluded that the concepts of resource description, abstraction, and orchestration could by applied to federated resources in order to control them in a cross-domain and cross-layer fashion. However, one limitation of the presented solution is that the design of virtual resource groupings is mostly performed by the user. This means that the resource selection process to derive a functional infrastructure, in most cases, relies on some design actions carried out by the user. Although the domain managers dispose of a certain degree of flexibility regarding *intra-domain* resource selection and configuration decisions, most of the *inter-domain* choices have to be taken by the user. Here, future work could build upon the presented work putting specific emphasis an advanced higher tier federation logic, e.g. by relying on semantic technologies, to enable more intelligent inter-domain resource selection and configuration choices. However, the results presented here, already allow for common resource control, intelligent intra-domain choices, and automated cross-domain provisioning. Therefore, it can be concluded that the third part of the hypothesis can also be corroborated.

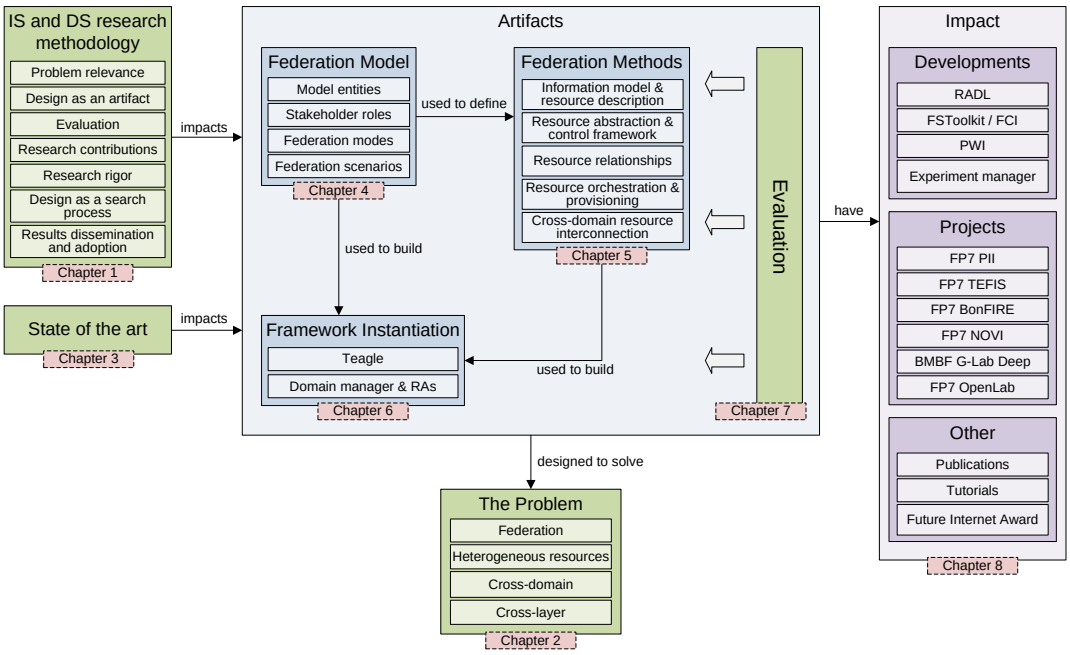


Figure 8.1 – Thesis impact

Figure 8.1 visualizes the process of how the results were produced and what the impact is today. Generally, the work was based upon a widely accepted research methodology originating from the information system (IS) and design science (DS) fields. Together with the state of art they provided the starting point for the design and development of this thesis' artifacts. The artifacts impact each other, have been designed to solve the problem, and were evaluated using observational and experimental evaluation methods. On the right hand side of figure 8.1, the impact of this work is shown. The following four main achievements can be listed:

Artifact usage (projects and developments): The artifacts and especially the Teagle framework are adapted in multiple projects, even beyond the FP7 Panlab projects (e.g.

FP7 TEFIS, BMBF G-Lab Deep). Furthermore, additional developments currently emerge that are based on the results presented here such as the FSToolkit/FCI[13], RADL[14], PWI[15], and an experiment manager[16]. More on the emerging developments and other upcoming activities can be found in the outlook section 8.2.

Publications: A total number of 21 publications related to the topics of this thesis have been published between May 2007 and May 2011: 5 journal articles[17], 3 book chapters[18], 9 conference papers[19], and 4 workshop papers[20].

Talks & tutorials: In addition to the various talks given at scientific conferences related to the topics covered by the publications, 3 half-day tutorials[21] have been given focusing on federation concepts and the Teagle framework instantiation. Also, a couple of demonstration videos[22] have been published to explain some of the Teagle functions.

Future Internet Award: "The FP7 project PII was selected among 32 submissions to win the Future Internet Forum (FIF) award for the best initiative that is currently running and nearing completion. The pan-European testbed federation platform prototyped by PII builds upon a federation framework and the resource brokering system 'Teagle' whose development is lead by Fraunhofer FOKUS. According to the Judging Panel, Panlab was 'innovate and exemplar' and would have a 'high impact'. The award was officially presented at the closing plenary of the Future Internet Assembly (FIA) in Ghent on December 17, 2010."[23]

The next section will outline important extensions to the presented work driven by currently running and upcoming activities in 2011 and 2012.

8.2 Outlook

The Teagle VCT tool provides the user with the possibility to configure a desired virtual resource grouping. However, often users are interested in what can be done with such infrastructure instead of the details of how it is composed and configured. Specifically, experimentally driven research demands experimental infrastructure that supports any specific type of experiment. Therefore, instead of defining an infrastructure that enables the experiment, the users are increasingly expected to define an experiment and ask the platform to provision an according infrastructure automatically. This issue is currently addressed by an experiment manager designed and developed by the FP7 TEFIS project. To fit the TEFIS needs, the resource description method and repository instantiation used by Teagle are currently extended by enriching the resource model and the descriptions hold by the repository. This will

[13] FSToolkit website: http://nam.ece.upatras.gr/fstoolkit/trac, FCI project website: http://trac.panlab.net/trac/wiki/FCI

[14] As published in [149]

[15] As published at http://trac.panlab.net/trac/wiki/PWI

[16] The experiment manager is driven by the FP7 TEFIS project. At the time of writing no public information is available but is expected soon.

[17] [1], [17], [21], [79], [101]

[18] [99], [112], [163]

[19] [15], [16], [20], [22], [88], [98], [100], [115], [149]

[20] [24], [116], [164], [165]

[21] [166], [167], [168]

[22] As published on the Teagle website: http://www.fire-teagle.org/tutorials.jsp

[23] Cited from: http://www.fire-teagle.org/news.jsp?news_id=42

allow the experiment manager to match the experiment definition to experimental resources. Also, the experiment manager is expected to extend the current Teagle portfolio with a more advanced resource reservation system. Currently, resource reservation is following a rather simple best effort model. However, to speed up the virtual resource grouping specification process, more advanced combinational use of resource availability as well as resource and provider policy information would be desirable.

In addition, the upcoming FP7 OpenLab project will also look into how the existing approaches in the field of experiment description (e.g. OMF) can be consolidated and better integrated with resource control, management, and brokering frameworks such as Teagle or the SFAv2.

Furthermore, to better support the user in the entire research process, current activities at Fraunhofer FOKUS concentrate on extending the Teagle simulation/emulation and monitoring capabilities. As motivated by figure 1.1, experimentally driven research usually begins with a formal model and the simulation of a possible solution. To further increase the level of experiment realism, emulation techniques allow to replace certain parameters of the simulation with real values. However, the next step from a simulation/emulation to a real large scale facility is not an easy one. Usually, moving a specific experiment from the laboratory environment to a real facility is very time consuming. Here, Teagle is envisioned to help by supporting the user in the entire chain from a formal model to a large scale facility. Precisely, ongoing work concentrates on integrating ns[24] into Teagle. This shall allow the user to load an ns script in Teagle and find suitable Teagle-controlled resources to perform the simulated/emulated experiment with a virtual resource grouping provisioning by Teagle. This activity also nicely fits the FP7 TEFIS experiment manager efforts given that a stable integration can be achieved.

For a more industry-oriented outlook of this work consider figure 8.2 that shows how federation can be applied in a cross-domain (horizontal federation) and cross-layer (vertical federation) fashion to the broad field of NGNs. In section 7.2, a federation experiment dealing with the *migration of IMS services across domains* has been discussed. Such scenarios are expected to become increasingly important in the future, driven by emerging paradigms such as network convergence and cloud computing. Therefore, federation aspects and the dynamic negotiation with resource providers across all layers of the ICT technology stack will be key for service providers in the mid and long term future. Advanced industry-driven networking solutions such as the 3GPP Evolved Packet Core (EPC)[25] as well as academic approaches such as concurrent multipath transmissions [169], [170] that will influence future commercial deployments will be essential to ensure QoS and Quality of Experience (QoE) in highly heterogeneous and potentially federated environments.

A challenge for large federations in particular will be to find the right balance between a tight resource provider integration including strict SLAs, and a best-effort-oriented approach. In the latter case, resources will appear and disappear based on their current availability, and upper layers will need to adapt dynamically. The key question will be what impact such high variability of resource availability will have on the QoS and QoE perceived by the user. Also, what is an acceptable time scale to reserve/lease virtual (network) resources, given that a certain signaling overhead is required to acquire those? What is an acceptable degree of *resource variability* and what are the consequences for the services based upon

[24]The discrete-event network simulator, http://en.wikipedia.org/wiki/Ns_%28simulator%29
[25]OpenEPC project website: http://www.openepc.net

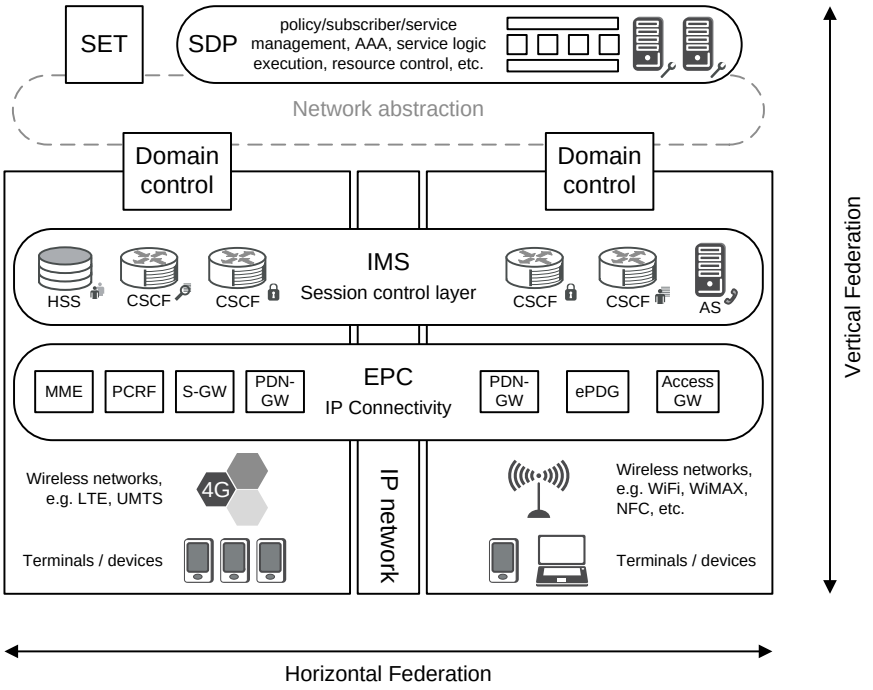


Figure 8.2 – Outlook: Cross-domain and cross-layer federation applied to the NGN field

such resources? Also, data federation and data interoperability is just about to emerge with the advance of e-Government and the integration of utility providers (e.g. smart grid/metering, home automation, IoT, etc.). This thesis cannot answer all of those questions and standardization is expected to drive industrial use cases and large scale commercialization. However, further research into such directions could be based upon the resource control and abstraction framework and the federation aspects and formal modeling presented by this thesis. Nonetheless, it has to be noted again, that the technical questions are only a subset of the aspects that need to be investigated as particularly legal and regional regulation issues can become the show stoppers for open and dynamic federation. This makes this area still a very attractive field for interdisciplinary research.

Last but not least, some interesting industrial exploitation possibilities are currently discussed with ETSI. The ETSI Plugtest^{TM} unit is specializing in running interoperability test events. Here, considerable efforts are spend on manually preparing and setting up the test environments required for a specific Plugtest event. Teagle provides the means to specify and provision any testbed environment setup, given that it disposes of resource adapter implementations controlling the required system. Also, re-configuration and re-provisioning can be handled effectively. Therefore, Teagle is discussed to be used for such events allowing to reduce the testbed preparation and re-configuration costs by automating manual configuration and deployment steps as well as routine processes.

Therefore, the hope is that the work presented here will continue to serve and impact scientific activities in both industry and academia.

A

VCT Design and Provisioning

This annex provides detailed insight into the full process of virtual resource grouping (a VCT in Teagle terminology) design and provisioning as defined by the UCM-based use case introduced in section 6.3. Numerous listings explain and demonstrate the Teagle internal communication as well as the message exchange with the federated domains.

It has to be noted that all Teagle instance components as well as the domain manager instance that have been used to gather the data presented in this section have been hosted on the same virtual machine. Therefore, all communication is taking place within the same physical host (193.175.132.210)[1]. Table A.1 shows the configuration of the different services and the corresponding ports to facilitate the understanding of the data presented here.

Table A.1 – The Teagle and domain manager service endpoints used for this use case

Service	Endpoint
Teagle Portal	http://193.175.132.210:8080/teagle
Teagle repository	http://193.175.132.210:8080/repository/rest
Request processor (RP)	http://193.175.132.210:8080/reqproc
Policy engine (PE)	http://193.175.132.210:8080/openpe/services/PolicyEngineService
Orchestration engine (OE)	http://193.175.132.210:80//teagle-site
Domain manager (DM)	http://193.175.132.210:8000

Initially, it is assumed that the user has accessed the Teagle portal using his Teagle credentials consisting of a username and a corresponding password. Further, it is assumed that he has started the VCT tool following a link in the VCT design section of the Teagle portal.

A.1 Requesting Data from the Teagle Repository

During the VCT tool startup process (see also section 6.3.3), the tool contacts the Teagle repository and loads available information for existing resource types, resource instances, and

[1] A public Fraunhofer FOKUS network address but restricted in terms of external access

virtual resource groupings (Virtual Customer Testbeds (VCTs)). Such requests are `HTTP GET` requests as for example shown in listing A.1. All data in form of XML documents provided by the repository are structured according to the model shown in annex C.

Listing A.1 – VCT tool request to load available information on existing VCTs

```
GET /repository/rest/vct HTTP/1.1
Content-Type: text/xml
Cache-Control: no-cache
Pragma: no-cache
User-Agent: Java/1.6.0_20
Host: 127.0.0.1:8080
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Connection: keep-alive
```

Listing A.2 – Teagle repository response on existing VCTs (shortened, only selected VCTs are shown)

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Type: text/xml;charset=utf-8
Transfer-Encoding: chunked
Date: Tue, 28 Jun 2011 11:53:15 GMT

<?xml version="1.0" encoding="UTF-8"?>
<list>
<vct id="63">
   <commonName>federicatest</commonName>
   <description></description>
   <hasBookings />
   <hasConnections>
      <connection id="6" />
      <connection id="5" />
      <connection id="7" />
   </hasConnections>
   <providesResources>
   <resourceInstance id="184" />
      <resourceInstance id="171" />
      <resourceInstance id="179" />
      <resourceInstance id="174" />
   </providesResources>
   <shared>false</shared>
   <state id="7" />
   <user id="1" />
</vct>
...
<vct id="91">
   <commonName>nirvana</commonName>
   <description></description>
   <hasBookings />
   <hasConnections>
      <connection id="35" />
      <connection id="34" />
      <connection id="33" />
      <connection id="36" />
   </hasConnections>
   <providesResources>
      <resourceInstance id="349" />
      <resourceInstance id="350" />
```

```
      <resourceInstance id="342" />
      <resourceInstance id="345" />
   </providesResources>
   <shared>false</shared>
   <state id="2" />
   <user id="1" />
</vct>
...
<vct id="108">
   <commonName>DissertationVCT</commonName>
   <description></description>
   <hasBookings />
   <hasConnections>
      <connection id="82" />
      <connection id="81" />
      <connection id="80" />
      <connection id="83" />
   </hasConnections>
   <providesResources>
      <resourceInstance id="496" />
      <resourceInstance id="493" />
      <resourceInstance id="288" />
      <resourceInstance id="499" />
      <resourceInstance id="489" />
      </providesResources>
   <shared>false</shared>
   <state id="6" />
   <user id="2" />
</vct>
...
```

Further, person data is loaded by the tool as shown in listings A.3 and A.4 to associate the VCTs to the different Teagle users. The `<user id="2"/>` tag from the VCTs listing above corresponds to the respective user entry in listing A.4. Note, that the password hashes are only exemplary and have been replaced by random data.

Listing A.3 – VCT tool person request

```
GET /repository/rest/person HTTP/1.1
Content-Type: text/xml
Cache-Control: no-cache
Pragma: no-cache
User-Agent: Java/1.6.0_20
Host: 127.0.0.1:8080
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Connection: keep-alive
```

Listing A.4 – Teagle repository person response (shortened)

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Type: text/xml;charset=utf-8
Transfer-Encoding: chunked
Date: Tue, 28 Jun 2011 11:53:15 GMT

<?xml version="1.0" encoding="UTF-8"?>
<list>
   <person id="1">
```

```
      <email />
      <fullName>Administrator</fullName>
      <organisations />
      <password>1176569fcc1771beb2405c0c56bf9a55</password>
      <personRoles><personRole id="1" /></personRoles>
      <userName>root</userName>
   </person>
   <person id="2">
      <email />
      <fullName>testuser</fullName>
      <organisations><organisation id="1" />
      </organisations>
      <password>b6a74bbe603abb62312f7cc97bab7008</password>
      <personRoles><personRole id="4" />
      </personRoles>
      <userName>testuser</userName>
   </person>
</list>
```

The following listings show the data related to the VCT with `<vct id="108">` and `<commonName>DissertationVCT</commonName>` that is obtained by the VCT tool from the repository. The repository responses can carry large XML documents. Those have been cut to avoid lengthy listings and to enable easier reading and comprehension of the different XML documents. Therefore, only the data relevant to `<vct id="108">` is shown in the following.

Listing A.2 shows that `<vct id="108">` has the following connections:

Listing A.5 – Connections of `<vct id="108">`

```
   <hasConnections>
      <connection id="82" />
      <connection id="81" />
      <connection id="80" />
      <connection id="83" />
   </hasConnections>
```

Therefore, the following listing only displays the XML for those connections, although the response to the request `GET /repository/rest/connection` would carry *all* the connections from *all* VCTs known to the repository. This has been cut to the relevant pieces: only connection `80`, `81`, `82`, `83`. This "XML cutting method" is applied to all of the following listings (connections, resourceInstances, resourceSpecs, configlets, etc.) in this section!

Listing A.6 – Connections request and response (shortened, only for `vct id="108"`)

```
GET /repository/rest/connection HTTP/1.1
...

HTTP/1.1 200 OK
...
<?xml version="1.0" encoding="UTF-8"?>
<list>
   <connection id="80">
      <destination id="160" />
      <rules></rules>
      <source id="159" />
      <type enumType="ConnectionType">CONTAINS</type>
   </connection>
   <connection id="81">
```

```
15      <destination id="162" />
16      <rules></rules>
17      <source id="161" />
18      <type enumType="ConnectionType">CONTAINS</type>
19    </connection>
20    <connection id="82">
21      <destination id="164" />
22      <rules></rules>
23      <source id="163" />
24      <type enumType="ConnectionType">CONTAINS</type>
25    </connection>
26    <connection id="83">
27      <destination id="166" />
28      <rules></rules>
29      <source id="165" />
30      <type enumType="ConnectionType">CONTAINS</type>
31    </connection>
32  </list>
```

From this information further pieces can be obtained. For example `<connection id="80">` lists `<destination id="160" />` and `<source id="159" />`. For those items the data hold by the repository also needs to be requested. This is shown in the listings A.8 and A.9.

Listing A.7 – Resource instances request and response (shortened, only for `vct id="108"`)

```
1   GET /repository/rest/resourceInstance HTTP/1.1
2   ...
3
4   HTTP/1.1 200 OK
5   ...
6   <?xml version="1.0" encoding="UTF-8"?>
7   <list>
8     <resourceInstance id="288">
9       <commonName>//test/pnode-0</commonName>
10      <configurationData />
11      <description>Physical Host pnode-0</description>
12      <geometry id="114" />
13      <resourceSpec id="275" />
14      <shared>false</shared>
15      <state id="9" />
16    </resourceInstance>
17    <resourceInstance id="489">
18      <commonName>//test/pnode-0/mysql-QFgT9sbgdN2VoDIq</commonName>
19      <configurationData />
20      <description>Deploy mysql</description>
21      <geometry id="190" />
22      <resourceSpec id="255" />
23      <shared>false</shared>
24      <state id="9" />
25    </resourceInstance>
26    <resourceInstance id="493">
27      <commonName>//test/pnode-0/fhoss-b9qwiyGlRULvpCb7</commonName>
28      <configurationData>
29        <configlet id="490" />
30        <configlet id="492" />
31        <configlet id="491" />
32      </configurationData>
```

```
      <description>Fraunhofer HSS</description>
      <geometry id="191" />
      <resourceSpec id="293" />
      <shared>false</shared>
      <state id="9" />
   </resourceInstance>
   <resourceInstance id="496">
      <commonName>//test/pnode-0/mysql-QFgT9sbgdN2VoDIq/database-HSSdb
      </commonName>
      <configurationData>
         <configlet id="495" />
         <configlet id="494" />
      </configurationData>
      <description>http://193.175.132.210:8000/pnode-0/</description>
      <geometry id="192" />
      <resourceSpec id="263" />
      <shared>false</shared>
      <state id="9" />
   </resourceInstance>
   <resourceInstance id="499">
      <commonName>//test/pnode-0/mysql-QFgT9sbgdN2VoDIq/dbuser-HSSdbuser
      </commonName>
      <configurationData>
         <configlet id="498" />
         <configlet id="497" />
      </configurationData>
      <description>dbuser for mysql database</description>
      <geometry id="193" />
      <resourceSpec id="259" />
      <shared>false</shared>
      <state id="9" />
   </resourceInstance>
</list>
```

Listing A.8 – Destinations request and response (shortened, only for `vct id="108"`)

```
GET /repository/rest/dst HTTP/1.1
...

HTTP/1.1 200 OK
...
<?xml version="1.0" encoding="UTF-8"?>
<list>
   <dst id="160">
      <identifier></identifier>
      <pos>0</pos>
      <resourceInstance id="493" />
      <side>0</side>
   </dst>
   <dst id="162">
      <identifier></identifier>
      <pos>0</pos>
      <resourceInstance id="489" />
      <side>0</side>
   </dst>
   <dst id="164">
      <identifier></identifier>
      <pos>0</pos>
```

```
      <resourceInstance id="496" />
      <side>0</side>
   </dst>
   <dst id="166">
      <identifier></identifier>
      <pos>0</pos>
      <resourceInstance id="499" />
      <side>0</side>
   </dst>
</list>
```

Listing A.9 – Sources request and response (shortened, only for `vct id="108"`)

```
GET /repository/rest/src HTTP/1.1
...

HTTP/1.1 200 OK
...
<?xml version="1.0" encoding="UTF-8"?>
<list>
   <src id="159">
      <identifier></identifier>
      <pos>0</pos>
      <resourceInstance id="288" />
      <side>0</side>
   </src>
   <src id="161">
      <identifier></identifier>
      <pos>0</pos>
      <resourceInstance id="288" />
      <side>0</side>
   </src>
   <src id="163">
      <identifier></identifier>
      <pos>0</pos>
      <resourceInstance id="489" />
      <side>0</side>
   </src>
   <src id="165">
      <identifier></identifier>
      <pos>0</pos>
      <resourceInstance id="489" />
      <side>0</side>
   </src>
</list>
```

Listing A.10 – Geometry request and response (shortened, only for `vct id="108"`)

```
GET /repository/rest/geometry HTTP/1.1
...

HTTP/1.1 200 OK
...
<?xml version="1.0" encoding="UTF-8"?>
<list>
   <geometry id="114">
      <h>100</h>
      <w>200</w>
      <x>132</x>
```

```
      <y>733</y>
    </geometry>
    <geometry id="190">
      <h>0</h>
      <w>0</w>
      <x>394</x>
      <y>419</y>
    </geometry>
    <geometry id="191">
      <h>0</h>
      <w>0</w>
      <x>267</x>
      <y>20</y>
    </geometry>
    <geometry id="192">
      <h>0</h>
      <w>0</w>
      <x>567</x>
      <y>185</y>
    </geometry>
    <geometry id="193">
      <h>0</h>
      <w>0</w>
      <x>706</x>
      <y>528</y>
    </geometry>
</list>
```

Listing A.11 – ResourceSpecs request and response (shortened, only for `vct id="108"`)

```
GET /repository/rest/resourceSpec HTTP/1.1
...

HTTP/1.1 200 OK
...
<?xml version="1.0" encoding="UTF-8"?>
<list>
    <resourceSpec id="255">
      <commonName>mysql</commonName>
      <configurationParameters id="264" />
      <cost id="12" />
      <description>|http://193.175.132.210:8000/pnode-0/|Deploy mysql
      </description>
      <isInstantiable />
      <keywords />
    </resourceSpec>
    <resourceSpec id="259">
      <commonName>dbuser</commonName>
      <configurationParameters id="258" />
      <cost id="13" />
      <description>|http://193.175.132.210:8000/pnode-0/|dbuser for mysql
        database</description>
      <isInstantiable />
      <keywords />
    </resourceSpec>
    <resourceSpec id="263">
      <commonName>database</commonName>
      <configurationParameters id="262" />
```

```
      <cost id="14" />
      <description>|http://193.175.132.210:8000/pnode-0/|</description>
      <isInstantiable />
      <keywords />
   </resourceSpec>
   <resourceSpec id="275">
      <commonName>pnode</commonName>
      <configurationParameters id="274" />
      <cost id="15" />
      <description>|http://193.175.132.210:8000/pnode-0/|</description>
      <isInstantiable />
      <keywords />
   </resourceSpec>
   <resourceSpec id="293">
      <commonName>fhoss</commonName>
      <configurationParameters id="292" />
      <cost id="16" />
      <description>|http://193.175.132.210:8080/repository/|Fraunhofer HSS
      </description>
      <isInstantiable />
      <keywords />
   </resourceSpec>
</list>
```

Listing A.12 – ConfigParamComposite request and response (shortened, only for `vct id="108"`)

```
GET /repository/rest/configParamComposite HTTP/1.1
...

HTTP/1.1 200 OK
...
<?xml version="1.0" encoding="UTF-8"?>
<list>
   <configParamComposite id="258">
      <commonName>specConfigParams</commonName>
      <configParams>
         <configParamAtomic id="257" />
         <configParamAtomic id="256" />
      </configParams>
      <description></description>
   </configParamComposite>
   <configParamComposite id="262">
      <commonName>specConfigParams</commonName>
      <configParams>
         <configParamAtomic id="260" />
         <configParamAtomic id="261" />
      </configParams>
      <description></description>
   </configParamComposite>
   <configParamComposite id="264">
      <commonName>specConfigParams</commonName>
      <configParams />
      <description></description>
   </configParamComposite>
   <configParamComposite id="274">
      <commonName>specConfigParams</commonName>
      <configParams />
      <description></description>
```

```
33    </configParamComposite>
34    <configParamComposite id="292">
35       <commonName>specConfigParams</commonName>
36       <configParams>
37          <configParamAtomic id="289" />
38          <configParamAtomic id="290" />
39          <configParamAtomic id="291" />
40       </configParams>
41       <description></description>
42    </configParamComposite>
43 </list>
```

Listing A.13 – ConfigParamAtomic request and response (shortened, only for `vct id="108"`)

```
1  GET /repository/rest/configParamAtomic HTTP/1.1
2  ...
3
4  HTTP/1.1 200 OK
5  ...
6  <?xml version="1.0" encoding="UTF-8"?>
7  <list>
8     <configParamAtomic id="256">
9        <commonName>name</commonName>
10       <configParamType>string</configParamType>
11       <defaultParamValue></defaultParamValue>
12       <description>username</description>
13    </configParamAtomic>
14    <configParamAtomic id="257">
15       <commonName>password</commonName>
16       <configParamType>string</configParamType>
17       <defaultParamValue></defaultParamValue>
18       <description>user password</description>
19    </configParamAtomic>
20    <configParamAtomic id="260">
21       <commonName>name</commonName>
22       <configParamType>string</configParamType>
23       <defaultParamValue></defaultParamValue>
24       <description>database name</description>
25    </configParamAtomic>
26    <configParamAtomic id="261">
27       <commonName>owner</commonName>
28       <configParamType>reference</configParamType>
29       <defaultParamValue></defaultParamValue>
30       <description>database owner</description>
31    </configParamAtomic>
32    <configParamAtomic id="289">
33       <commonName>ims_domain</commonName>
34       <configParamType>string</configParamType>
35       <defaultParamValue>openims.test</defaultParamValue>
36       <description></description>
37    </configParamAtomic>
38    <configParamAtomic id="290">
39       <commonName>port</commonName>
40       <configParamType>int</configParamType>
41       <defaultParamValue>20003</defaultParamValue>
42       <description></description>
43    </configParamAtomic>
44    <configParamAtomic id="291">
```

```
      <commonName>rdbms</commonName>
      <configParamType>reference</configParamType>
      <defaultParamValue></defaultParamValue>
      <description></description>
   </configParamAtomic>
</list>
```

Listing A.14 – Configlet request and response (shortened, only for `vct id="108"`)

```
GET /repository/rest/configlet HTTP/1.1
...

HTTP/1.1 200 OK
...
<?xml version="1.0" encoding="UTF-8"?>
<list>
   <configlet id="490">
      <commonName>port</commonName>
      <configurationParametersAtomic>
         <configParamAtomic id="290" />
      </configurationParametersAtomic>
      <description></description>
      <paramValue>20003</paramValue>
   </configlet>
   <configlet id="491">
      <commonName>ims_domain</commonName>
      <configurationParametersAtomic>
         <configParamAtomic id="289" />
      </configurationParametersAtomic>
      <description></description>
      <paramValue>openims.test</paramValue>
   </configlet>
   <configlet id="492">
      <commonName>rdbms</commonName>
      <configurationParametersAtomic>
         <configParamAtomic id="291" />
      </configurationParametersAtomic>
      <description></description>
      <paramValue>//test/pnode-0/mysql-QFgT9sbgdN2VoDIq</paramValue>
   </configlet>
   <configlet id="494">
      <commonName>name</commonName>
      <configurationParametersAtomic>
         <configParamAtomic id="260" />
      </configurationParametersAtomic>
      <description>database name</description>
      <paramValue>HSSdb</paramValue>
   </configlet>
   <configlet id="495">
      <commonName>owner</commonName>
      <configurationParametersAtomic>
         <configParamAtomic id="261" />
      </configurationParametersAtomic>
      <description>database owner</description>
      <paramValue>//test/pnode-0/mysql-QFgT9sbgdN2VoDIq/dbuser-HSSdbuser
      </paramValue>
   </configlet>
   <configlet id="497">
```

```
50      <commonName>password</commonName>
51      <configurationParametersAtomic>
52        <configParamAtomic id="257" />
53      </configurationParametersAtomic>
54      <description>user password</description>
55      <paramValue>dbuserpwd</paramValue>
56    </configlet>
57    <configlet id="498">
58      <commonName>name</commonName>
59      <configurationParametersAtomic>
60        <configParamAtomic id="256" />
61      </configurationParametersAtomic>
62      <description>username</description>
63      <paramValue>HSSdbuser</paramValue>
64    </configlet>
65  </list>
```

A.2 Virtual Resource Grouping Design and Configuration

Figure A.1 shows another VCT layout that will be used in the following. The user has dragged and dropped the resources into the workbench and has defined several resource relationships by connecting the respective boxes.

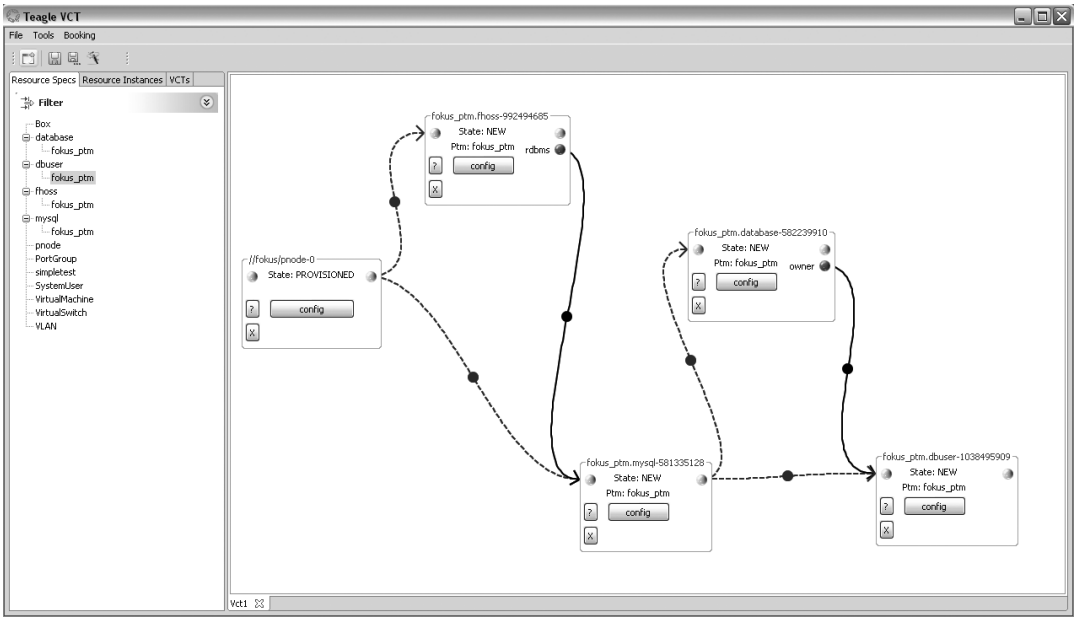


Figure A.1 – The VCT tool showing the virtual resource grouping before the booking has been initiated

The artifacts delivered by this thesis — the federation model methods, specifically the method of abstractly defining resource relationships as defined in section 5.3 — allow to formulate this virtual resource grouping as follows:

$$VG = \{R_{fokus_x} \in MR_{fokus} \mid x = \{1, 2, 3, 4, 5\}\} \tag{A.1}$$

with

$$\begin{aligned} R_{fokus_1} &= \text{fokus/pnode-0} \\ R_{fokus_2} &= \text{fokus/fhoss} \\ R_{fokus_3} &= \text{fokus/database} \\ R_{fokus_4} &= \text{fokus/mysql} \\ R_{fokus_5} &= \text{fokus/dbuser} \end{aligned} \tag{A.2}$$

where `fokus/pnode-0` is an existing instance (indicated by the trailing dash and resource index `0`) while the other resource identifiers refer to resource types to be instantiated for this virtual resource grouping.

Also, if p denotes a function that assigns a resource y to be the parent resource of x and if r denotes a function that assign a resource x to rely on the configuration of resource z,

$$p : MR_{fokus} \rightarrow MR_{fokus},\ x \mapsto y \tag{A.3}$$

$$r : MR_{fokus} \rightarrow MR_{fokus},\ x \mapsto z \tag{A.4}$$

then, the resource relationships for this virtual resource grouping can be formulated as:

$$\begin{aligned} p(R_{fokus_2}) &= R_{fokus_1} \\ p(R_{fokus_4}) &= R_{fokus_1} \\ p(R_{fokus_3}) &= R_{fokus_4} \\ p(R_{fokus_5}) &= R_{fokus_4} \\ r(R_{fokus_2}) &= R_{fokus_4} \\ r(R_{fokus_3}) &= R_{fokus_5} \end{aligned} \tag{A.5}$$

This is visualized by figure A.2.

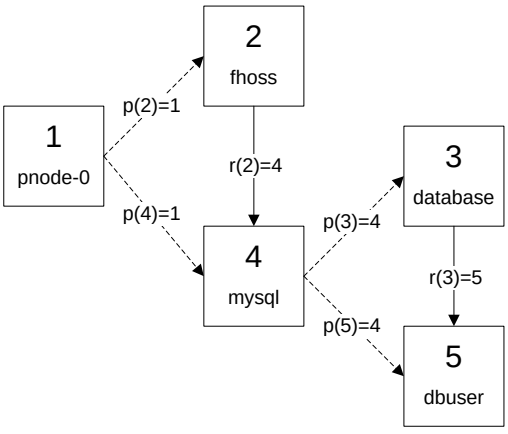


Figure A.2 – Abstract representation of the virtual resource grouping shown by the VCT tool in figure A.1

This demonstrates how entire federations and the different resources and their relationships can be modeled and abstractly formulated to allow for automated provisioning and

higher tier federation logic. For example, this dependency graph is in the following resolved by the orchestration engine to derive a functional environment. Later changes to the setup requested by the user can easily be configured and instantiated. Also, the rollback to previous installations can be realized in a structured and automated fashion.

During the virtual resource grouping design, several policy evaluations are performed (see also 6.3.4). For the sake of brevity this is omitted here. However, a detailed policy evaluation example is given in annex B. Each time a resource is placed on the workbench a policy evaluation is performed to check if the user is allowed to book the respective resource. Also, each time two resources are connected by the user (using either a *containment*[2] or a *configuration*[3] reference), a policy evaluation is performed to check if connecting those two resources is allowed. Therefore, resource providers and the Teagle framework in general can operate on any desired level of resource integration. For tight resource integration, several restrictive policies can be defined that only allow for a very limited set of resource combinations. Alternatively, the usage of resources can be left open, defining none or only very few policies.

The open style has the disadvantage of having a higher resource provisioning failure rate. This is because the user is likely to define virtual resource groupings that can not be deployed by the platform due to a meaningless or erroneous design. This results in a best effort system rejecting meaningless requests. In such case, the usability might be poor for unexperienced users. On the other hand, the open style requires less administrative overhead to maintain the policy definitions, avoid deadlock situations, etc.

Once the user has finished the design of the desired VCT and selects the *save* feature from the VCT tool menu, the tool send numerous `HTTP PUT` requests to the Teagle repository carrying XML documents with the VCT configuration to store the user-designed VCT in the repository. For the VCT shown in figure A.1 those requests are shown by listing A.15. The VCT will have the `<vct id="112">` as well as the common name `<commonName>EvaluateVCT</commonName>`. The repository replies with a `200 OK` optionally attaching any updated data. The resource instance states are at this stage `id="4"` which stands for `UNPROVISIONED`. Please note that the repository responses as well as cost related data are omitted in the following listing for the sake of brevity.

Listing A.15 – VCT tool requests to store the VCT layout and configuration in the repository

```
PUT /repository/rest/configlet/521/ HTTP/1.1
...
<configletInstance>
  <configurationParametersAtomic>
    <configParamAtomic>261</configParamAtomic>
  </configurationParametersAtomic>
  <paramValue>fokus_ptm.dbuser-1038495909</paramValue>
  <commonName>owner</commonName>
  <description>database owner</description>
</configletInstance>

PUT /repository/rest/configlet/522/ HTTP/1.1
...
<configletInstance>
  <configurationParametersAtomic>
    <configParamAtomic>260</configParamAtomic>
```

[2]Synonymously used for the parent-child relationship represented by function p

[3]Represented by function r

```
  </configurationParametersAtomic>
  <paramValue>DissDB</paramValue>
  <commonName>name</commonName>
  <description>database name</description>
</configletInstance>

PUT /repository/rest/geometry/202/ HTTP/1.1
...
<geometryInstance>
  <x>541</x>
  <y>179</y>
  <w>0</w>
  <h>0</h>
</geometryInstance>

PUT /repository/rest/resourceInstance/523/ HTTP/1.1
...
<resourceInstanceInstance>
   <commonName>fokus_ptm.database-582239910</commonName>
   <description>http://193.175.132.210:8000/pnode-0/</description>
   <state.id>9</state.id>
   <resourceSpec>
      <resourceSpec>263</resourceSpec>
   </resourceSpec>
   <geometry>
      <geometry>202</geometry>
   </geometry>
   <configurationData>
      <configlet>521</configlet>
      <configlet>522</configlet>
   </configurationData>
   <shared>false</shared>
</resourceInstanceInstance>

PUT /repository/rest/configlet/524/ HTTP/1.1
...
<configletInstance>
  <configurationParametersAtomic>
    <configParamAtomic>256</configParamAtomic>
  </configurationParametersAtomic>
  <paramValue>DissUser</paramValue>
  <commonName>name</commonName>
  <description>username</description>
</configletInstance>

PUT /repository/rest/configlet/525/ HTTP/1.1
...
<configletInstance>
  <configurationParametersAtomic>
    <configParamAtomic>257</configParamAtomic>
  </configurationParametersAtomic>
  <paramValue>DBpwd</paramValue>
  <commonName>password</commonName>
  <description>user password</description>
</configletInstance>

PUT /repository/rest/geometry/203/ HTTP/1.1
...
```

```
<geometryInstance>
  <x>764</x>
  <y>445</y>
  <w>0</w>
  <h>0</h>
</geometryInstance>

PUT /repository/rest/resourceInstance/526/ HTTP/1.1
...
<resourceInstanceInstance>
  <commonName>fokus_ptm.dbuser-1038495909</commonName>
  <description>dbuser for mysql database</description>
  <state.id>9</state.id>
  <resourceSpec>
    <resourceSpec>259</resourceSpec>
  </resourceSpec>
  <geometry>
    <geometry>203</geometry>
  </geometry>
  <configurationData>
    <configlet>524</configlet>
      <configlet>525</configlet>
  </configurationData>
  <shared>false</shared>
</resourceInstanceInstance>

PUT /repository/rest/geometry/201/ HTTP/1.1
...
<geometryInstance>
  <x>413</x>
  <y>452</y>
  <w>0</w>
  <h>0</h>
</geometryInstance>

PUT /repository/rest/resourceInstance/520/ HTTP/1.1
...
<resourceInstanceInstance>
  <commonName>fokus_ptm.mysql-581335128</commonName>
  <description>Deploy mysql</description>
  <state.id>9</state.id>
  <resourceSpec>
    <resourceSpec>255</resourceSpec>
  </resourceSpec>
  <geometry>
    <geometry>201</geometry>
  </geometry>
  <configurationData/>
  <shared>false</shared>
</resourceInstanceInstance>

PUT /repository/rest/configlet/516/ HTTP/1.1
...
<configletInstance>
  <configurationParametersAtomic>
    <configParamAtomic>290</configParamAtomic>
  </configurationParametersAtomic>
  <paramValue>20003</paramValue>
```

```
  <commonName>port</commonName>
  <description></description>
</configletInstance>

PUT /repository/rest/configlet/517/ HTTP/1.1
...
<configletInstance>
  <configurationParametersAtomic>
    <configParamAtomic>289</configParamAtomic>
  </configurationParametersAtomic>
  <paramValue>openims.test</paramValue>
  <commonName>ims_domain</commonName>
  <description></description>
</configletInstance>

PUT /repository/rest/configlet/518/ HTTP/1.1
...
<configletInstance>
  <configurationParametersAtomic>
    <configParamAtomic>291</configParamAtomic>
  </configurationParametersAtomic>
  <paramValue>fokus_ptm.mysql-581335128</paramValue>
  <commonName>rdbms</commonName>
  <description></description>
</configletInstance>

PUT /repository/rest/geometry/200/ HTTP/1.1
...
<geometryInstance>
  <x>229</x>
  <y>41</y>
  <w>0</w>
  <h>0</h>
</geometryInstance>

PUT /repository/rest/resourceInstance/519/ HTTP/1.1
...
<resourceInstanceInstance>
   <commonName>fokus_ptm.fhoss-992494685</commonName>
   <description>Fraunhofer HSS</description>
   <state.id>9</state.id>
   <resourceSpec>
      <resourceSpec>293</resourceSpec>
   </resourceSpec>
   <geometry>
      <geometry>200</geometry>
   </geometry>
   <configurationData>
      <configlet>516</configlet>
      <configlet>517</configlet>
      <configlet>518</configlet>
   </configurationData>
   <shared>false</shared>
</resourceInstanceInstance>
```

Once the VCT has been successfully stored, the VCT is ready for booking. The booking details are given by the next section.

A.3 Booking the Virtual Resource Grouping

This section lists the Teagle communication once the booking of `<vct id="112">` has been requested by the user (see the use case description in the sections 6.3.5 and 6.3.6 for details regarding the processes). The request send by the VCT tool to the request processor (RP) is shown in listing A.16.

Listing A.16 – Booking request from the VCT tool to the request processor (RP)

```
POST /reqproc HTTP/1.1

Content-Type: text/plain
User-Agent: Java/1.6.0_26
Host: 193.175.132.210:8080
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Connection: keep-alive
Content-Length: 133

<string-array>
   <string>VctRegistry</string>
   <string>setVct</string>
   <string>testuser</string>
   <string>EvaluateVCT</string>
</string-array>
```

The request processor can now fetch the VCT specification based on the VCT common name `EvaluateVCT` from the Teagle repository, request policy evaluation, enforce policy decisions and forward the request to the orchestration engine (OE). The next listing A.17 shows the response from the RP to the VCT tool *after* the provisioning has been *successfully* been performed.

Listing A.17 – Booking response from the RP

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Length: 803
Date: Tue, 28 Jun 2011 11:53:52 GMT

<return>
  <status>0</status>
  <message></message>
  <log>http://localhost/teagle-site/tmp/log_exec_testuser_EvaluateVCT.4159.txt</log>
  <result>
    <idmapping>
      <mapping designid="fokus_ptm.resources.//fokus/pnode-0" runtimeid="//fokus/pnode-0"/>
      <mapping designid="fokus_ptm.mysql-581335128" runtimeid="//fokus/pnode-0/mysql-
          d3S2XvB1oc5sBrwP"/>
      <mapping designid="fokus_ptm.fhoss-992494685" runtimeid="//fokus/pnode-0/fhoss-
          GGnvy1NaZHZ0k7cF"/>
      <mapping designid="fokus_ptm.database-582239910" runtimeid="//fokus/pnode-0/mysql-
          d3S2XvB1oc5sBrwP/database-Tifup_vo5gMU0W2B"/>
      <mapping designid="fokus_ptm.dbuser-1038495909" runtimeid="//fokus/pnode-0/mysql-
          d3S2XvB1oc5sBrwP/dbuser-sMq_tBwt4yryueEp"/>
    </idmapping>
  </result>
</return>
```

The status `0` indicates a successful virtual resource grouping deployment. A log file is available at the given URL `http://localhost/teagle-site/tmp/log_exec_testuser_EvaluateVCT.4159.txt`. Also, the response includes the full *runtime IDs* for all the requested resource instances. Those will be displayed by the tool after reloading the VCT as shown by figure A.3.

From the collected packet trace files, it can be observed that there is a delay of about 37 seconds between the VCT tool request and the RP response. An excerpt of the entire trace file is given in table A.2 showing only the tool request (issued by `10.147.69.191`[4]) and the RP response (issued by `193.175.132.210`[5]).

Table A.2 – Packet trace of the VCT tool booking request and request processor response

No.	Time	Source	Destination	Protocol	Info
792	155.260937	10.147.69.191	193.175.132.210	HTTP	POST /reqproc HTTP/1.1
794	192.045274	193.175.132.210	10.147.69.191	HTTP	HTTP/1.1 200 OK

Listing A.18 shows the time stamps in a full date format as well as the delta between `frame 792` (the request) and `frame 794` (the response). This delta of `36.784291000` seconds represents the *virtual resource grouping deployment time*.

Listing A.18 – Frames, time stamps, and time delta

```
1 POST /reqproc HTTP/1.1
2 frame 792
3 frame.time == "Jun 28, 2011 13:53:16.121301000"
4
5 HTTP/1.1 200 OK
6 frame 794
7 frame.time == "Jun 28, 2011 13:53:52.905638000"
8 frame.time_delta == 36.784291000
```

The next sections gives insight into the Teagle internal communication that was performed during this time delta.

A.4 Virtual Resource Grouping Deployment

Upon receiving the booking request, the RP retrieves the VCT specification from the Teagle repository. This results in a number of `HTTP GET` requests similar to ones shown in the listings A.3 to A.14. Once the full VCT specification is known to the RP, it requests policy evaluation for all resources and configurations that are part of the virtual resource grouping. Again, this is omitted here, annex B shows examples of such policy evaluation. If the VCT tool has been used to generate the VCT specification, the policy evaluation should not fail as evaluation has already been performed during the VCT design. Essentially, the RP performs the same exact evaluation as the tool as the same Java code realizing the evaluation is used for the implementation of both the tool and the RP. However, as the VCT tool runs on the user's machine, it cannot by trusted. Therefore, the double evaluation is necessary. In case of a

[4]As the VCT tool is a Java Web start application residing on the user's machine, this IP address belongs to the machine executing the VCT tool.

[5]The virtual machine hosting the entire Teagle and domain manager setup

policy evaluation failure, the VCT deployment process is aborted and an error is reported instead of the `200 OK` shown in listing A.17.

Once the policy evaluation and enforcement has been completed, the RP forwards the request to the orchestration engine (OE) listening on `port 80`. This is shown in listing A.19.

Listing A.19 – VCT provisioning request from the RP to the OE

```
POST /spatelrunner/cgi-bin/clientMyRD.py HTTP/1.1
User-Agent: Jakarta Commons-HttpClient/3.1
Host: localhost
Content-Length: 4218
Content-Type: multipart/form-data; boundary=sHoTSL9RNuMF3NW3rI1iC0yDJ8BjShyh9

--sHoTSL9RNuMF3NW3rI1iC0yDJ8BjShyh9
Content-Disposition: form-data; name="step"
Content-Type: text/plain; charset=US-ASCII
Content-Transfer-Encoding: 8bit
xdocmd

--sHoTSL9RNuMF3NW3rI1iC0yDJ8BjShyh9
Content-Disposition: form-data; name="op"
Content-Type: text/plain; charset=US-ASCII
Content-Transfer-Encoding: 8bit
putVCTSpec

--sHoTSL9RNuMF3NW3rI1iC0yDJ8BjShyh9
Content-Disposition: form-data; name="siteid"
Content-Type: text/plain; charset=US-ASCII
Content-Transfer-Encoding: 8bit
teagle

--sHoTSL9RNuMF3NW3rI1iC0yDJ8BjShyh9
Content-Disposition: form-data; name="appid"
Content-Type: text/plain; charset=US-ASCII
Content-Transfer-Encoding: 8bit
teagle.repository

--sHoTSL9RNuMF3NW3rI1iC0yDJ8BjShyh9
Content-Disposition: form-data; name="serviceid"
Content-Type: text/plain; charset=US-ASCII
Content-Transfer-Encoding: 8bit
VCTRegistry

--sHoTSL9RNuMF3NW3rI1iC0yDJ8BjShyh9
Content-Disposition: form-data; name="clientid"
Content-Type: text/plain; charset=US-ASCII
Content-Transfer-Encoding: 8bit

--sHoTSL9RNuMF3NW3rI1iC0yDJ8BjShyh9
Content-Disposition: form-data; name="v_vctid"
Content-Type: text/plain; charset=US-ASCII
Content-Transfer-Encoding: 8bit
testuser_EvaluateVCT

--sHoTSL9RNuMF3NW3rI1iC0yDJ8BjShyh9
Content-Disposition: form-data; name="v_vctfile"; filename="booking.xml"
Content-Type: application/octet-stream; charset=ISO-8859-1
Content-Transfer-Encoding: binary
```

```
<testbed>
  <components>
    <database>
      <id>fokus_ptm.database-582239910</id>
      <state>unprovisioned</state>
      <configuration>
        <owner type="reference">$dynid(fokus_ptm.dbuser-1038495909)</owner>
        <name type="string">DissDB</name>
      </configuration>
    </database>
    <dbuser>
      <id>fokus_ptm.dbuser-1038495909</id>
      <state>unprovisioned</state>
      <configuration>
        <name type="string">DissUser</name>
        <password type="string">DBpwd</password>
      </configuration>
    </dbuser>
    <mysql>
      <id>fokus_ptm.mysql-581335128</id>
      <state>unprovisioned</state>
      <configuration/>
    </mysql>
    <fhoss>
      <id>fokus_ptm.fhoss-992494685</id>
      <state>unprovisioned</state>
      <configuration>
        <port type="int">20003</port>
        <ims_domain type="string">openims.test</ims_domain>
        <rdbms type="reference">$dynid(fokus_ptm.mysql-581335128)</rdbms>
      </configuration>
    </fhoss>
  </components>
  <connections>
    <connection>
      <src>
        <id>fokus_ptm.database-582239910</id>
      </src>
      <dst>
        <id>fokus_ptm.dbuser-1038495909</id>
      </dst>
      <type>references</type>
    </connection>
    <connection>
      <src>
        <id>fokus_ptm.fhoss-992494685</id>
      </src>
      <dst>
        <id>fokus_ptm.mysql-581335128</id>
      </dst>
      <type>references</type>
    </connection>
    <connection>
      <src>
        <id>fokus_ptm.resources.//fokus/pnode-0</id>
      </src>
      <dst>
        <id>fokus_ptm.fhoss-992494685</id>
```

```
110       </dst>
111       <type>contains</type>
112     </connection>
113     <connection>
114       <src>
115         <id>fokus_ptm.mysql-581335128</id>
116       </src>
117       <dst>
118         <id>fokus_ptm.database-582239910</id>
119       </dst>
120       <type>contains</type>
121     </connection>
122     <connection>
123       <src>
124         <id>fokus_ptm.resources.//fokus/pnode-0</id>
125       </src>
126       <dst>
127         <id>fokus_ptm.mysql-581335128</id>
128       </dst>
129       <type>contains</type>
130     </connection>
131     <connection>
132       <src>
133         <id>fokus_ptm.mysql-581335128</id>
134       </src>
135       <dst>
136         <id>fokus_ptm.dbuser-1038495909</id>
137       </dst>
138       <type>contains</type>
139     </connection>
140   </connections>
141 </testbed>
```

The OE instantiation is based on an existing implementation of a SPATEL engine [145] exposing everything as a service. The same request could have been issued from a web interface exposed by the OE. This is why the request is of `Content-Type: multipart/form-data;`. This was rather an implementation choice as we were building upon exiting software rather than a design driven feature. The essential information is carried in the XML part of the request which is the VCT specification for the virtual resource grouping `EvaluateVCT`. Alternatively, the OE — being a Teagle internal component — could have loaded the specification from the repository itself.

As everything is exposed as a service in the OE, it now calls itself (the launcher) with the reference to the VCT deployment service. This is shown in listing A.20.

Listing A.20 – VCT deployment launcher

```
1 GET /spatelrunner/cgi-bin/clientMyRD.py?step=xdocmd&op=deployVCT&siteid=teagle&appid=teagle.
      orchestrationengine&serviceid=Launcher&clientid=&v_vctid=testuser_EvaluateVCT&options=d0&
      userId= HTTP/1.1
```

The launcher then initiates the orchestration which sends the T_1 provisioning requests and assembles the results. Note that in this use case implementation, the Teagle Gateway (TGW) has been omitted. This was done to simplify the entire process. Also, as the resources are all provided by the Fraunhofer domain, cross-domain reference resolution and transport layer security was not required. Listing A.21 shows the orchestration request and response in a

shortened form.

Listing A.21 – VCT deployment launcher

```
GET /spatelrunner/cgi-bin/xclientMyRD.py?step=docmd&op=orchestrate&siteid=teagle&appid=teagle.
    testbeds.usersebastian&serviceid=testuser_EvaluateVCT&clientid=&options=d0&userId= HTTP
    /1.1
...

HTTP/1.1 200 OK
...
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<return>
  <status>0</status>
  <message></message>
  <log>http://localhost/teagle-site/tmp/log_exec_testuser_EvaluateVCT.4159.txt</log>
  <launchparams>
 </launchparams>
 <result><idmapping>
    <mapping designid="fokus_ptm.resources.//fokus/pnode-0" runtimeid="//fokus/pnode-0"/>
    <mapping designid="fokus_ptm.mysql-581335128" runtimeid="//fokus/pnode-0/mysql-
        d3S2XvB1oc5sBrwP"/>
    <mapping designid="fokus_ptm.fhoss-992494685" runtimeid="//fokus/pnode-0/fhoss-
        GGnvy1NaZHZ0k7cF"/>
    <mapping designid="fokus_ptm.database-582239910" runtimeid="//fokus/pnode-0/mysql-
        d3S2XvB1oc5sBrwP/database-Tifup_vo5gMU0W2B"/>
    <mapping designid="fokus_ptm.dbuser-1038495909" runtimeid="//fokus/pnode-0/mysql-
        d3S2XvB1oc5sBrwP/dbuser-sMq_tBwt4yryueEp"/>
</idmapping>
</result>
</return>
```

Note that the XML returned upon this service call is forwarded to RP which relays the information to the VCT tool as already discussed and shown in listing A.17. Table A.3 shows the relevant trace file excerpt demonstrating the communication delay induced by T_1 reference point communication.

Table A.3 – Packet trace of the OE engine orchestration request and the respective response

No.	Time	Source	Destination	Protocol	Info
19	0.730277	127.0.0.1	127.0.0.1	HTTP	GET /spatelrunner/cgi-bin/xclientMyRD.py? step=docmd&op=orchestrate&siteid=teagle& appid=teagle.testbeds.usersebastian&serviceid =testuser_EvaluateVCT HTTP/1.1
22	27.543308	127.0.0.1	127.0.0.1	HTTP/XML	HTTP/1.1 200 OK

Listing A.22 shows the time stamps in a full date format as well as the delta between `frame 19` (the request) and `frame 22` (the response). This delta of `26.813010000` seconds represents the time needed for the communication on T_1 and the intra domain resource provisioning.

Listing A.22 – Frames, time stamps, and time delta

```
GET /spatelrunner/cgi-bin/xclientMyRD.py? ...
frame 19
frame.time == "Jun 28, 2011 13:53:20.854038000"
```

```
4
5  HTTP/1.1 200 OK
6  frame 22
7  frame.time == "Jun 28, 2011 13:53:47.667069000"
8  frame.time_delta == 26.813010000
```

The T_1 and intra domain communication performed during the time delta is discussed in the next section.

A.5 T_1 and Intra Domain Communication

First, the OE retrieves the information on the parent pnode0, using the T_1 *read* operation resulting in a `HTTP GET` towards the domain manager (DM) responsible for the domain `fokus`. Remember that the Teagle installation and the DM reside on the same machine, therefore, the request is directed to *localhost* but to port `8000` where the DM service is listening.

Listing A.23 – T1 request

```
1  GET /rest/fokus_ptm.resources.//fokus/pnode-0 HTTP/1.0
2  Host: localhost:8000
3  User-Agent: Python-urllib/1.17
```

The request triggers intra domain requests on reference point T_2 to assemble the information requested in T_1.

Listing A.24 – Domain internal request and response

```
1  POST /RPC2 HTTP/1.0
2  Host: ptm:8000
3  User-Agent: xmlrpclib.py/1.0.1
4  Content-Type: text/xml
5  <?xml version='1.0'?>
6  <methodCall>
7     <methodName>get_configuration</methodName>
8     <params>
9        <param>
10          <value>
11             <string>/pnode-0</string>
12          </value>
13       </param>
14    </params>
15 </methodCall>
16
17 HTTP/1.0 200 OK
18 Date: Tue, 28 Jun 2011 11:53:20 GMT
19 Server: WSGIServer/0.1 Python/2.6.5
20 Content-type: text/xml
21 <?xml version='1.0'?>
22 <methodResponse>
23    <params>
24       <param>
25          <value>
26             <struct>
27                <member>
28                   <name>public_ip</name>
29                   <value>
```

```
                  <string>193.175.132.210</string>
               </value>
            </member>
            <member>
               <name>hostname</name>
               <value>
                  <string>pnode-0</string>
               </value>
            </member>
         </struct>
      </value>
   </param>
</params>
</methodResponse>
```

Now, the reply on T_1 can be given, resulting in the following response:

Listing A.25 – T1 response

```
HTTP/1.0 200 OK
Date: Tue, 28 Jun 2011 11:53:20 GMT
Server: WSGIServer/0.1 Python/2.6.5
Content-Type: application/xml

<?xml version="1.0" encoding="utf-8"?>
<pnode>
   <configuration>
      <identifier type="string">//fokus/pnode-0</identifier>
      <public_ip type="string">193.175.132.210</public_ip>
      <hostname type="string">pnode-0</hostname>
   </configuration>
</pnode>
```

Further, the Teagle layer now requests the provisioning of the mysql, fhoss, database, and dbuser resources. For the sake of brevity only the *mysql* related provisioning messages are listed in the following. The T1 mysql provisioning request is:

Listing A.26 – T1 request

```
POST /rest//fokus/pnode-0 HTTP/1.0
Content-Type: application/x-www-form-urlencoded
Host: localhost:8000
User-Agent: Python-urllib/1.17
<mysql>
   <context>
      <vctId>testuser_EvaluateVCT</vctId>
   </context>
   <configuration/>
</mysql>
```

This triggers several domain internal requests and responses:

Listing A.27 – Domain internal request and response

```
POST /RPC2 HTTP/1.0
Host: ptm:8000
User-Agent: xmlrpclib.py/1.0.1
Content-Type: text/xml
<?xml version='1.0'?>
```

```
<methodCall>
   <methodName>add_resource</methodName>
   <params>
      <param>
         <value>
            <string>/pnode-0</string>
         </value>
      </param>
      <param>
         <value>
            <string>mysql</string>
         </value>
      </param>
   </params>
</methodCall>

HTTP/1.0 200 OK
Date: Tue, 28 Jun 2011 11:53:31 GMT
Server: WSGIServer/0.1 Python/2.6.5
Content-type: text/xml
<?xml version='1.0'?>
<methodResponse>
   <params>
      <param>
         <value>
            <string>/pnode-0/mysql-d3S2XvB1oc5sBrwP</string>
         </value>
      </param>
   </params>
</methodResponse>
```

This determines the identifier (`/pnode-0/mysql-d3S2XvB1oc5sBrwP`) that is used from now on to identify the fresh *mysql* instance. Note that the identifier also encodes the parent-child relationship between `pnode-0` and `mysql-d3S2XvB1oc5sBrwP`. Further, the domain internal communication includes the provisioning of the according *resource adapter* — which is considered a resource itself from the DM perspective — as shown in the following listing.

Listing A.28 – Domain internal resource adapter instantiation request and response

```
POST /RPC2 HTTP/1.0
Host: ptm:8000
User-Agent: xmlrpclib.py/1.0.1
Content-Type: text/xml
<?xml version='1.0'?>
<methodCall>
   <methodName>add_resource</methodName>
   <params>
      <param>
         <value>
            <string>/pnode-0/pyromanager-0</string>
         </value>
      </param>
      <param>
         <value>
            <string>pythonresourceadapter</string>
         </value>
      </param>
```

```
19      <param>
20        <value>
21          <struct>
22            <member>
23              <name>adapter_class</name>
24              <value>
25                <string>sMySQLAdapter.MySQLAdapter</string>
26              </value>
27            </member>
28            <member>
29              <name>adapter_parent</name>
30              <value>
31                <string>r/pnode-0/mysql-d3S2XvB1oc5sBrwP</string>
32              </value>
33            </member>
34          </struct>
35        </value>
36      </param>
37    </params>
38  </methodCall>
39
40  HTTP/1.0 200 OK
41  Date: Tue, 28 Jun 2011 11:53:31 GMT
42  Server: WSGIServer/0.1 Python/2.6.5
43  Content-type: text/xml
44  <?xml version='1.0'?>
45  <methodResponse>
46    <params>
47      <param>
48        <value>
49          <string>/pythonresourceadapter-/pnode-0/mysql-d3S2XvB1oc5sBrwP\#MySQLAdapter.
                MySQLAdapter</string>
50        </value>
51      </param>
52    </params>
53  </methodResponse>
```

The new resource (together with its adapter) is registered at the *domain registry* as shown in listing A.29.

Listing A.29 – Domain internal resource registration

```
1   POST /RPC2 HTTP/1.0
2   Host: ptm:8000
3   User-Agent: xmlrpclib.py/1.0.1
4   Content-Type: text/xml
5   <?xml version='1.0'?>
6   <methodCall>
7     <methodName>register</methodName>
8     <params>
9       <param>
10        <value>
11          <string>/pnode-0/mysql-d3S2XvB1oc5sBrwP/dbuser</string>
12        </value>
13      </param>
14      <param>
15        <value>
16          <string>http://193.175.132.210:10002</string>
```

```
        </value>
      </param>
   </params>
</methodCall>

HTTP/1.0 200 OK
Date: Tue, 28 Jun 2011 11:53:31 GMT
Server: WSGIServer/0.1 Python/2.6.5
Content-type: text/xml
```

Now, the response is send by the DM to the Teagle layer on T_1. The extra information (*port 3307*) is added by the DM as it is in charge of the domain and the mysql port has not been requested by the federation layer. This shows the domain level control that is left to the DM whenever specifics are not mandated by the federation layer.

Listing A.30 – T1 response

```
HTTP/1.0 200 OK
Date: Tue, 28 Jun 2011 11:53:31 GMT
Server: WSGIServer/0.1 Python/2.6.5
Content-Type: application/xml

<?xml version="1.0" encoding="utf-8"?>
<mysql>
   <configuration>
      <identifier type="string">//fokus/pnode-0/mysql-d3S2XvB1oc5sBrwP</identifier>
      <started type="boolean">True</started>
      <public_ip type="string">127.0.0.1</public_ip>
      <port type="integer">3307</port>
   </configuration>
</mysql>
```

This concept allows the resource provider to decide (during the initial resource registration) how much configuration options to expose to the federation user. Options that are not exposed but are needed for a resource to function properly, must be handled by the respective resource adapter. Also, from a federation organization point of view this concept allows to satisfy both users that want to configure resources at a high level of granularity as well as those users that want to stay on a more abstract level. Generally, any details and resource specifics that are not explicitly specified by the user are left to the domain (the respective resource adapter) to decide upon.

At this stage, further similar T_1 and domain internal messages are send regarding the provisioning of the other resources (fhoss, database, and dbuser). *Parent-child relationships* are realized by deploying the child resources within the respective parent. *configuration references* are realized as shown by the following listings.

Listing A.31 – T1 configuration reference request and response

```
POST /rest//fokus/pnode-0 HTTP/1.0
Content-Type: application/x-www-form-urlencoded
Host: localhost:8000
User-Agent: Python-urllib/1.17
<fhoss>
   <context>
      <vctId>testuser_EvaluateVCT</vctId>
```

```
  </context>
  <configuration>
    <port type="int">20003</port>
    <ims_domain type="string">openims.test</ims_domain>
    <rdbms type="reference">//fokus/pnode-0/mysql-d3S2XvB1oc5sBrwP</rdbms>
  </configuration>
</fhoss>

HTTP/1.0 200 OK
Date: Tue, 28 Jun 2011 11:53:47 GMT
Server: WSGIServer/0.1 Python/2.6.5
Content-Type: application/xml
<?xml version="1.0" encoding="utf-8"?>
<fhoss>
  <configuration>
    <identifier type="string">//fokus/pnode-0/fhoss-GGnvy1NaZHZ0k7cF</identifier>
    <peers type="object-array" />
    <diameter_port_number type="integer">20000</diameter_port_number>
    <started type="boolean">True</started>
    <db type="reference">//fokus/pnode-0/mysql-d3S2XvB1oc5sBrwP/database-jWbgqX8NRzgsSxT9</db
        >
    <dbuser type="reference">//fokus/pnode-0/mysql-d3S2XvB1oc5sBrwP/dbuser-e1io3UIS5sUgDR9U</
        dbuser>
    <ims_domain type="string">openims.test</ims_domain>
  </configuration>
</fhoss>
```

On the domain layer configuration references are resolved by querying (using the method `get_attribute`) a referenced resource for its configuration. In our example the resource adapter responsible for the *fhoss* resource retrieves the *mysql* resource configuration to deploy the *fhoss* resource. Among the required configuration details, the port of the mysql installation is required, therefore, it queries the `port` of the mysql instance as shown by the following listing:

Listing A.32 – Domain internal configuration reference resolution

```
POST /RPC2 HTTP/1.0
Host: ptm:8000
User-Agent: xmlrpclib.py/1.0.1
Content-Type: text/xml
<?xml version='1.0'?>
<methodCall>
  <methodName>get_attribute</methodName>
  <params>
    <param>
      <value>
        <string>/pnode-0/mysql-d3S2XvB1oc5sBrwP</string>
      </value>
    </param>
    <param>
      <value>
        <string>port</string>
      </value>
    </param>
  </params>
</methodCall>

```

```
HTTP/1.0 200 OK
Date: Tue, 28 Jun 2011 11:53:31 GMT
Server: WSGIServer/0.1 Python/2.6.5
Content-type: text/xml
<?xml version='1.0'?>
<methodResponse>
   <params>
      <param>
         <value>
            <int>3307</int>
         </value>
      </param>
   </params>
</methodResponse>
```

This allows to instantiate resources depending on the configuration of a remote resource. Intra-domain references are handled by the DM, cross-domain references need to be resolved on the federation layer which is done by the TGW.

Once all provisioning requests have successfully been processed, the result is passed back via the OE engine to the RP. The RP relays this to the user and also updates all the repository entries for resource instances, configlets, VCT, etc. with the runtime resource identifiers. Also, the RP updates the VCT state to `<state.id>6</state.id>` which stands for `BOOKED`. Updating the repository results in numerous `HTTP PUT` requests similar to the ones shown in the listing A.15. However, as an example, listing A.33 shows the request updating the data help for the VCT with `<vct id="112">`:

Listing A.33 – Updating the repository data for VCT 112

```
PUT /repository/rest/vct/112/ HTTP/1.1
Content-Type: text/xml
Cache-Control: no-cache
Pragma: no-cache
User-Agent: Java/1.6.0_20
Host: 127.0.0.1:8080
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Connection: keep-alive
Content-Length: 571
<vctInstance>
   <shared>false</shared>
   <commonName>EvaluateVCT</commonName>
   <state.id>6</state.id>
   <description></description>
   <providesResources>280</providesResources>
   <providesResources>523</providesResources>
   <providesResources>520</providesResources>
   <providesResources>526</providesResources>
   <providesResources>519</providesResources>
   <user>
      <person>2</person>
   </user>
   <hasConnections>90</hasConnections>
   <hasConnections>92</hasConnections>
   <hasConnections>91</hasConnections>
   <hasConnections>93</hasConnections>
   <hasBookings />
```

```
</vctInstance>

HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Type: text/xml;charset=utf-8
Transfer-Encoding: chunked
Date: Tue, 28 Jun 2011 11:53:49 GMT
<?xml version="1.0" encoding="UTF-8"?>
<vct id="112">
   <commonName>EvaluateVCT</commonName>
   <description></description>
   <hasBookings />
   <hasConnections>
      <connection id="91" />
      <connection id="93" />
      <connection id="90" />
      <connection id="92" />
   </hasConnections>
   <providesResources>
      <resourceInstance id="523" />
      <resourceInstance id="520" />
      <resourceInstance id="280" />
      <resourceInstance id="519" />
      <resourceInstance id="526" />
   </providesResources>
   <shared>false</shared>
   <state id="6" />
   <user id="2" />
</vct>
```

Once everything has been updated, the user is notified about the successful VCT deployment. Also, the URL of a log file provided by the OE is forwarded. Figure A.3 shows the VCT tool with `<vct id="112">` opened. Note that all resources shows the full runtime IDs and display the correct resource status `PROVISIONED`.

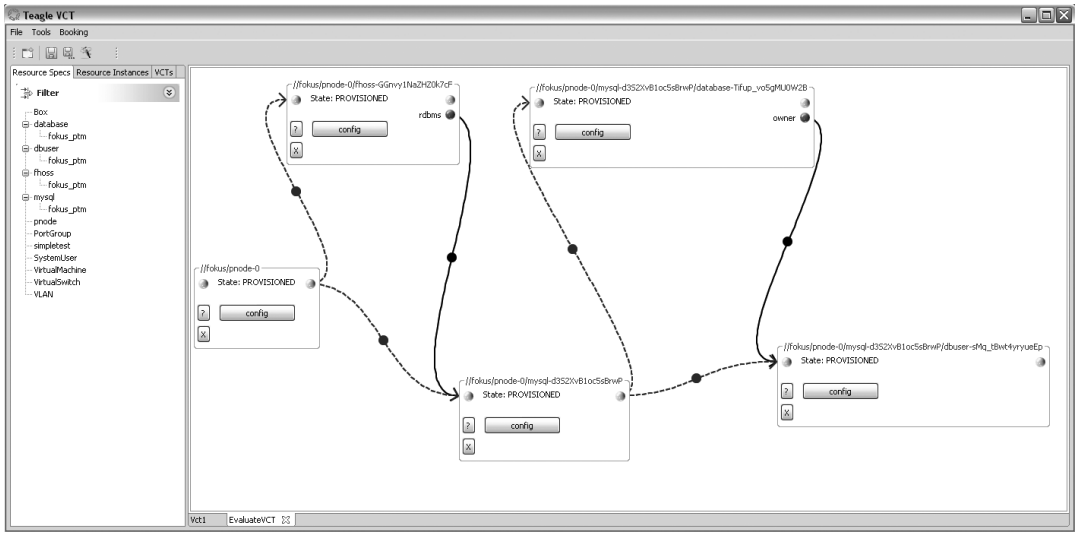


Figure A.3 – VCT tool showing the virtual resource grouping after successful resource provisioning. The resource instance IDs have changed from design time IDs to run time IDs.

B

Policy Evaluation Example

This annex shows the resource provider and VCT tool user views on the policy definition and evaluation process.

B.1 Policy Definition

Figure B.1 shows the policy overview page on the Teagle portal that can be accessed by resource providers. Here, the *resource policy* with the `id 290` has been created among other policies.

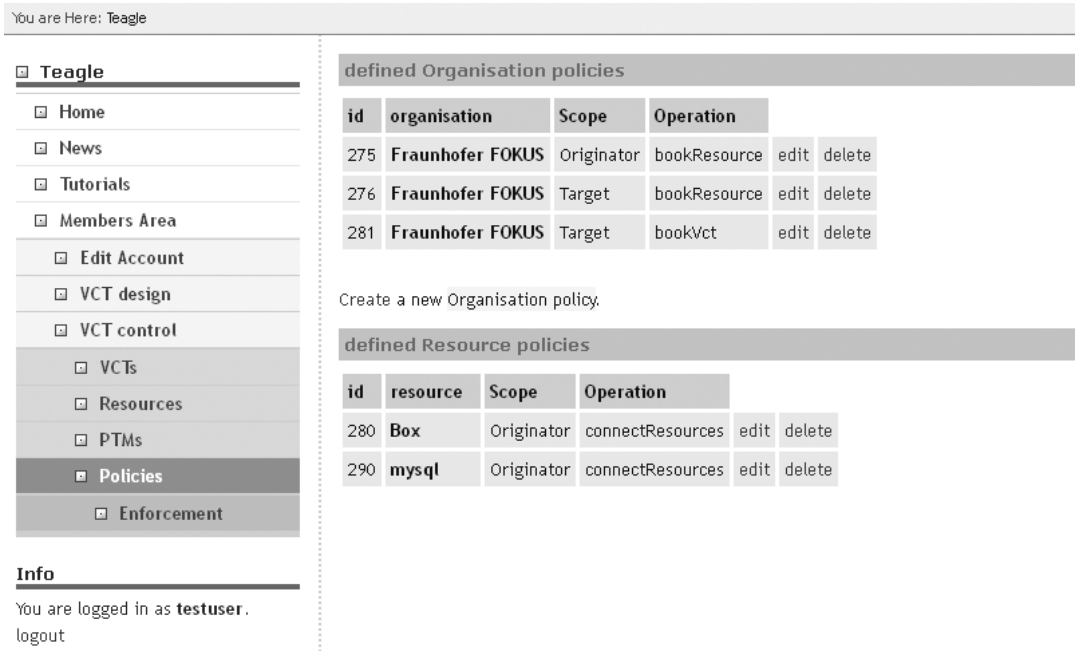


Figure B.1 – The policy overview page on the Teagle portal

By clicking the `edit` link, the user is taken to the policy definition page that is shown by figure B.2. The policy consists of the three scope identifiers `identity`, `scope`, `operation`, and

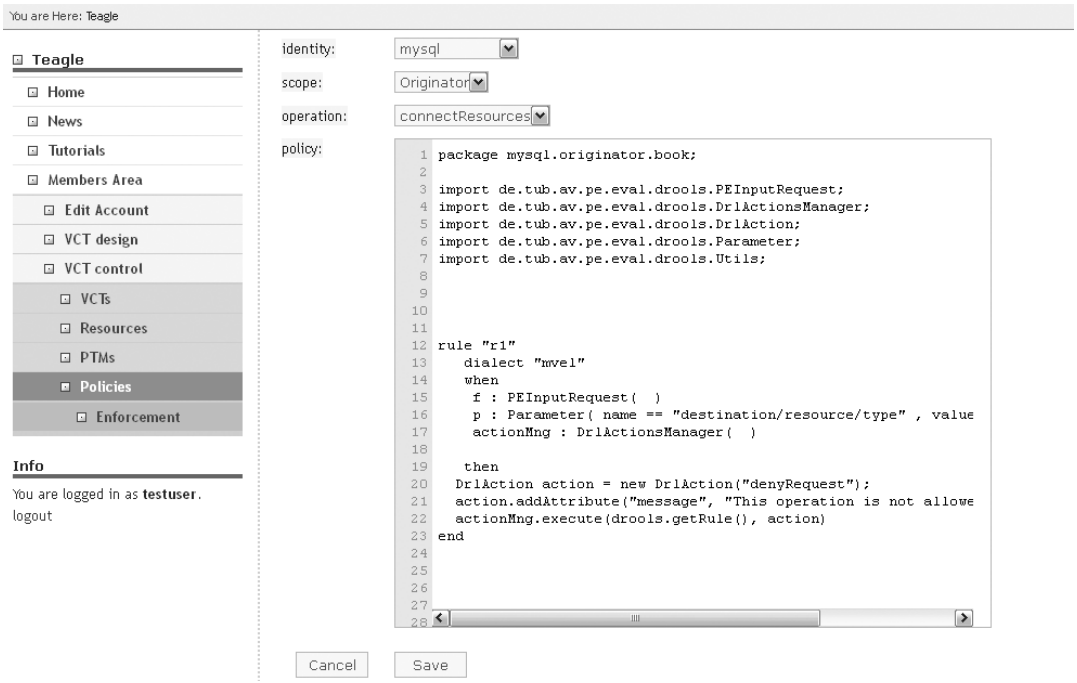


Figure B.2 – The policy definition page on the Teagle portal

the policy definition itself that is expressed using the Drools engine rule language[1]. Listing B.1 shows this policy.

Listing B.1 – Example resource policy

```
1  package mysql.originator.connectResource;
2
3  import de.tub.av.pe.eval.drools.PEInputRequest;
4  import de.tub.av.pe.eval.drools.DrlActionsManager;
5  import de.tub.av.pe.eval.drools.DrlAction;
6  import de.tub.av.pe.eval.drools.Parameter;
7  import de.tub.av.pe.eval.drools.Utils;
8
9  rule "r1"
10   dialect "mvel"
11   when
12      f : PEInputRequest( )
13      p : Parameter( name == "destination/resource/type" , value == "database" )
14      actionMng : DrlActionsManager( )
15   then
16      DrlAction action = new DrlAction("denyRequest");
17      action.addAttribute("message", "This operation is not allowed!");
18      actionMng.execute(drools.getRule(), action)
19 end
```

The package declaration groups a set of policy rules. The declaration must be unique for a given *knowledge base*. "A knowledge base consists of the packages and associated rules which have been loaded into the engine. The sources of the rules may be a file, a database etc. and it is the application job which integrates this engine to provide the rules as streams."

[1] JBoss Drools project website http://www.jboss.org/drools

([139], p. 28)

The import statements can be compared to Java import statements:

- `de.tub.av.pe.eval.drools.PEInputRequest` defines the fact type identifying the request.
- `de.tub.av.pe.eval.drools.DrlActionsManager` defines the actions manager used for action execution.
- `de.tub.av.pe.eval.drools.DrlAction` can be instantiated for rule actions and executed by the *DrlActionsManager*.
- `de.tub.av.pe.eval.drools.Parameter` defines the fact type containing the *name* and *value* properties.
- `de.tub.av.pe.eval.drools.teagle.Utils` represents static utility methods.

"A rule is identified by an `id` which is unique in the package and consists of attributes, conditions and actions. Using the attributes of a rule one can define the dialect to be used (e.g. `mvel` which exposes functions like `eval`, `modify`, etc.)" ([139], p. 28)

"The condition part of the rule starts with `when` and consists of conditional elements. The most used conditions elements are the patterns elements. A pattern matches against the input facts of the specific type. [...] A fact represents the information based on which rules conditions are asserted. The [engine] defines a specific type of fact `PEInputRequest` that maps to the identifiers of an incoming request (e.g. originator, targets, event) which consists of the following properties:

- originatorIdentity, respectively originatorIdentityType (user, resource)
- targetIdentities (as a list), respectively targetIdentityType (user, resource)
- event

For the event (request operation), its parameters are designed as facts of type Parameter." ([139], p. 28)

In the example shown in listing B.1, if a fact of type `Parameter` with property name equals `destination/resource/type` matches, then the rule will be asserted and the action `denyRequest` is reported to be enforced by the policy enforcement point. In case of Teagle this can be the VCT Tool, the request processor, or potentially a domain manager (although this has currently not yet been implemented).

"The *actions* part of a rule starts with the keyword `then`. After the assertion of the rules in the knowledge base is finished, the matching rules are added to an *Agenda*. Based on the rule attributes like priority [...] the associated actions are executed. The actions are deployed in a plugin manner. Therefore the constructor of the action (*DrlAction*) receives as input the name of the actions. Note [that] the registered actions are uniquely identified by their name. Attributes can be added to the action by calling `action.addAttribute(name, value)` [...] and executed by calling `actionMng.execute(drools.getRule(), action)`." ([139], p. 28)

Listing B.2 – Rule action, attributes, and execution

```
1 DrlAction action = new DrlAction("denyRequest");
2 action.addAttribute("message", "This operation is not allowed!");
3 actionMng.execute(drools.getRule(), action);
```

This demonstrated how policies can be defined using the Teagle portal and policy engine (PE) system instantiation. The next section explains the policy evaluation process and shows the interactions between the PE performing the policy evaluation and the VCT tool acting as policy enforcement point.

B.2 Policy Evaluation and Enforcement

Figure B.3 shows the VCT tool view as seen by the federation user in case a policy has been evaluated to *deny* the request. In such case, the VCT tool enforces the decision taken by the PE and displays an error message to the VCT tool user.

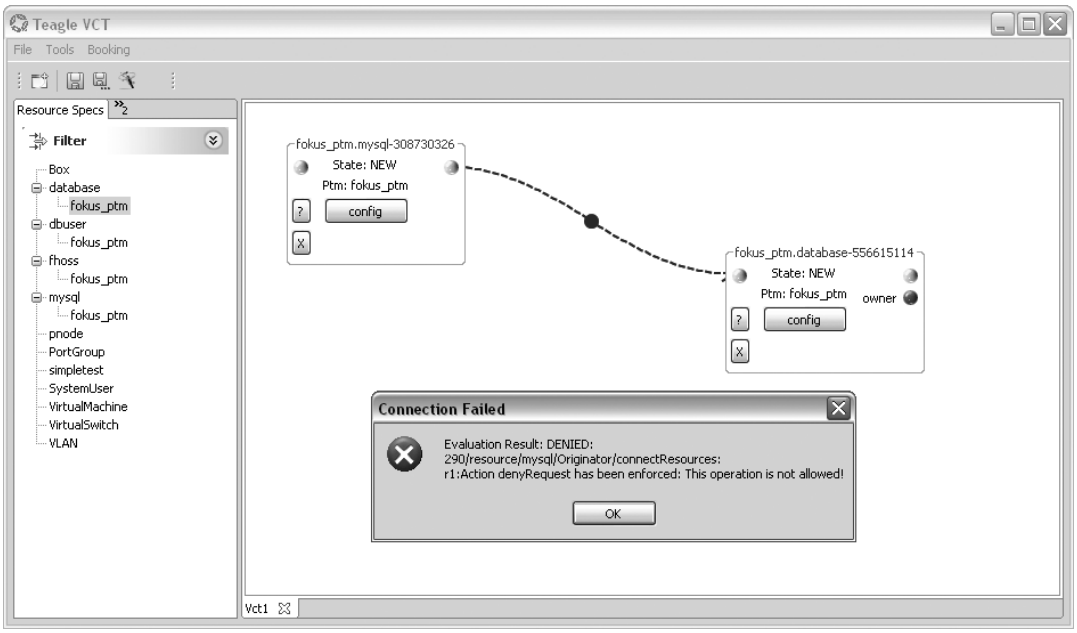


Figure B.3 – The VCT tool acting as policy a enforcement point

The policy evaluation requests send by the VCT tool to the PE are shown in the following listings. Listing B.3 shows the request form the VCT tool to the PE once the *database* resource has been placed on the VCT tool workbench. Listing B.4 shows the response. In this case of positive policy evaluation the user is allowed to continue working with the resource *database*.

Listing B.3 – Database resource booking policy evaluation request

```
1  POST /openpe/services/PolicyEngineService HTTP/1.1
2  Content-type: text/xml;charset="utf-8"
3  Soapaction: ""
4  Accept: text/xml, multipart/related, text/html, image/gif, image/jpeg, *; q=.2, */*;
5  User-Agent: JAX-WS RI 2.1.6 in JDK 6
6  Host: 193.175.132.210:8080
7  Connection: keep-alive
8  Content-Length: 1880
9
10 <?xml version="1.0" ?>
11 <S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
12   <S:Body>
```

```
13      <evaluatePolicy xmlns="http://www.openmobilealliance.org/wsdl/PEM1/v1_0/local.xsd"
            xmlns:ns2="http://www.openmobilealliance.org/wsdl/PEM1/v1_0/faults">
14        <callbackUrl></callbackUrl>
15        <timeStamp>2011-06-27T10:55:09.609+02:00</timeStamp>
16        <policyData>
17          <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
18          <ns2:policyInputData xmlns:ns2="urn:oma:xml:fokus:pem1-input-template">
19            <policyInputTemplate xsi:type="ns4:FOKUSSOAPInputTemplateType" id="0"
                  templateVersion="v1.0.0" templateID="FOKUSEnablerInputTemplate_ID1"
                  xmlns:ns4="urn:oma:xml:fokus:soap-pem1-input-template" xmlns:xsi="http://www.
                  w3.org/2001/XMLSchema-instance">
20              <requestMessage>true</requestMessage>
21              <policyIdentifiers>
22                <originatorID>testuser</originatorID>
23                <originatorIDType>user</originatorIDType>
24                <targetID>database</targetID>
25                <targetIDType>resource</targetIDType>
26              </policyIdentifiers>
27              <event name="bookResource">
28                <eventParameter name="user">testuser</eventParameter>
29                <eventParameter name="resource/provider">Fraunhofer FOKUS</eventParameter>
30                <eventParameter name="resource/type">database</eventParameter>
31                <eventParameter name="resource/price">100.0</eventParameter>
32                <eventParameter name="resource/ptm">fokus_ptm</eventParameter>
33              </event>
34            </policyInputTemplate>
35          </ns2:policyInputData>
36        </policyData>
37      </evaluatePolicy>
38    </S:Body>
39  </S:Envelope>
```

In this example case the policy engine replies positively. This means that the VCT tool user is allowed to book this resource. Therefore, the resource may be dragged to the workbench and may be used further in the VCT specification process. Although the user has not yet requested the booking, this is already checked to enable an early feedback.

Listing B.4 – Database resource booking policy evaluation response

```
1   HTTP/1.1 200 OK
2   Server: Apache-Coyote/1.1
3   Content-Type: text/xml;charset=utf-8
4   Transfer-Encoding: chunked
5   Date: Mon, 27 Jun 2011 08:55:08 GMT
6
7   <?xml version='1.0' encoding='UTF-8'?>
8   <S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
9     <S:Body>
10      <ns2:evaluatePolicyResponse xmlns="http://www.openmobilealliance.org/wsdl/PEM1/v1_0/
            faults" xmlns:ns2="http://www.openmobilealliance.org/wsdl/PEM1/v1_0/local.xsd">
11        <ns2:statusCode>2101</ns2:statusCode>
12        <ns2:statusText>ALLOWED: 286/user/testuser/Originator/*: r1:Action bookResource has
              been enforced</ns2:statusText>
13        <ns2:policyData>
14          <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
15          <ns2:policyOutputData xmlns:ns2="urn:oma:xml:fokus:pem1-output-template">
16            <policyOutputTemplate xsi:type="ns2:FOKUSOutputTemplateType" id="0"
                  templateVersion="v1.0.0" templateID="FOKUSEnablerInputTemplate_ID1"
```

```
                  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">;
17                <StatusCode>2101</StatusCode>
18                <StatusText>Successfully evaluated</StatusText>
19              </policyOutputTemplate>
20            </ns2:policyOutputData>
21          </ns2:policyData>
22        </ns2:evaluatePolicyResponse>
23    </S:Body>
24  </S:Envelope>
```

The listings B.5 and B.6 demonstrate a negative policy evaluation resulting in the error message displayed by figure B.3 because connecting a resource of type *mysql* to a resource of type *database* is prohibited by the policy shown in figure B.2 and listing B.1. This policy has only been defined to exemplify the concept and prototype implementation. In fact, connecting a *mysql* resource to a *database* resource should be allowed in most cases.

Listing B.5 – Mysql resource connection policy evaluation request

```
1   POST /openpe/services/PolicyEngineService HTTP/1.1
2   Content-type: text/xml;charset="utf-8"
3   Soapaction: ""
4   Accept: text/xml, multipart/related, text/html, image/gif, image/jpeg, *; q=.2, */*
5   User-Agent: JAX-WS RI 2.1.6 in JDK 6
6   Host: 193.175.132.210:8080
7   Connection: keep-alive
8   Content-Length: 2245
9
10  <?xml version="1.0" ?>
11  <S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
12    <S:Body>
13      <evaluatePolicy xmlns="http://www.openmobilealliance.org/wsdl/PEM1/v1_0/local.xsd"
            xmlns:ns2="http://www.openmobilealliance.org/wsdl/PEM1/v1_0/faults">
14        <callbackUrl></callbackUrl>
15        <timeStamp>2011-06-27T10:55:20.343+02:00</timeStamp>
16        <policyData>
17          <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
18          <ns2:policyInputData xmlns:ns2="urn:oma:xml:fokus:pem1-input-template">
19            <policyInputTemplate xsi:type="ns4:FOKUSSOAPInputTemplateType" id="0"
                  templateVersion="v1.0.0" templateID="FOKUSEnablerInputTemplate_ID1"
                  xmlns:ns4="urn:oma:xml:fokus:soap-pem1-input-template" xmlns:xsi="http://www.
                  w3.org/2001/XMLSchema-instance">
20              <requestMessage>true</requestMessage>
21              <policyIdentifiers>
22                <originatorID>mysql</originatorID>
23                <originatorIDType>resource</originatorIDType>
24                <targetID>database</targetID>
25                <targetIDType>resource</targetIDType>
26              </policyIdentifiers>
27              <event name="connectResources">
28                <eventParameter name="user">testuser</eventParameter>
29                <eventParameter name="connectionType">contains</eventParameter>
30                <eventParameter name="source/resource/type">mysql</eventParameter>
31                <eventParameter name="source/resource/provider">Fraunhofer FOKUS</
                      eventParameter>
32                <eventParameter name="source/resource/ptm">fokus_ptm</eventParameter>
33                <eventParameter name="destination/resource/type">database</eventParameter>
34                <eventParameter name="destination/resource/provider">Fraunhofer FOKUS</
```

```
                        eventParameter>
35                    <eventParameter name="destination/resource/ptm">fokus_ptm</eventParameter>
36                  </event>
37                </policyInputTemplate>
38              </ns2:policyInputData>
39            </policyData>
40          </evaluatePolicy>
41        </S:Body>
42      </S:Envelope>
```

Listing B.6 – Mysql resource connection policy evaluation response

```
1   HTTP/1.1 500 Internal Server Error
2   Server: Apache-Coyote/1.1
3   Content-Type: text/xml;charset=utf-8
4   Transfer-Encoding: chunked
5   Date: Mon, 27 Jun 2011 08:55:19 GMT
6   Connection: close
7
8   <?xml version='1.0' encoding='UTF-8'?>
9   <S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
10    <S:Body>
11      <S:Fault xmlns:ns4="http://www.w3.org/2003/05/soap-envelope">
12        <faultcode>S:Server</faultcode>
13        <faultstring>DENIED: 290/resource/mysql/Originator/connectResources: r1:Action
              denyRequest has been enforced: This operation is not allowed!</faultstring>
14        <detail>
15          <denyPolicyResponseException:denyPolicyResponseException
                xmlns:denyPolicyResponseException="http://www.openmobilealliance.org/wsdl/PEM1/
                v1_0/faults" xmlns="http://www.openmobilealliance.org/wsdl/PEM1/v1_0/faults"
                xmlns:ns2="http://www.openmobilealliance.org/wsdl/PEM1/v1_0/local.xsd">
16              <statusCode>2401</statusCode>
17              <statusText>DENIED: 290/resource/mysql/Originator/connectResources: r1:Action
                    denyRequest has been enforced: This operation is not allowed!</statusText>
18          </denyPolicyResponseException:denyPolicyResponseException>
19        </detail>
20      </S:Fault>
21    </S:Body>
22  </S:Envelope>
```

C

Information Model

This annex gives insight into the information model used by the system instantiation. It is also the basis for the Teagle portal productive system installation that is available online[1].

Figure C.1 shows the *key classes* of the information model using an UML class diagram notation. Figure C.2 shows the *full model*. Figure C.3 shows the *policy model*.

[1] http://www.fire-teagle.org/

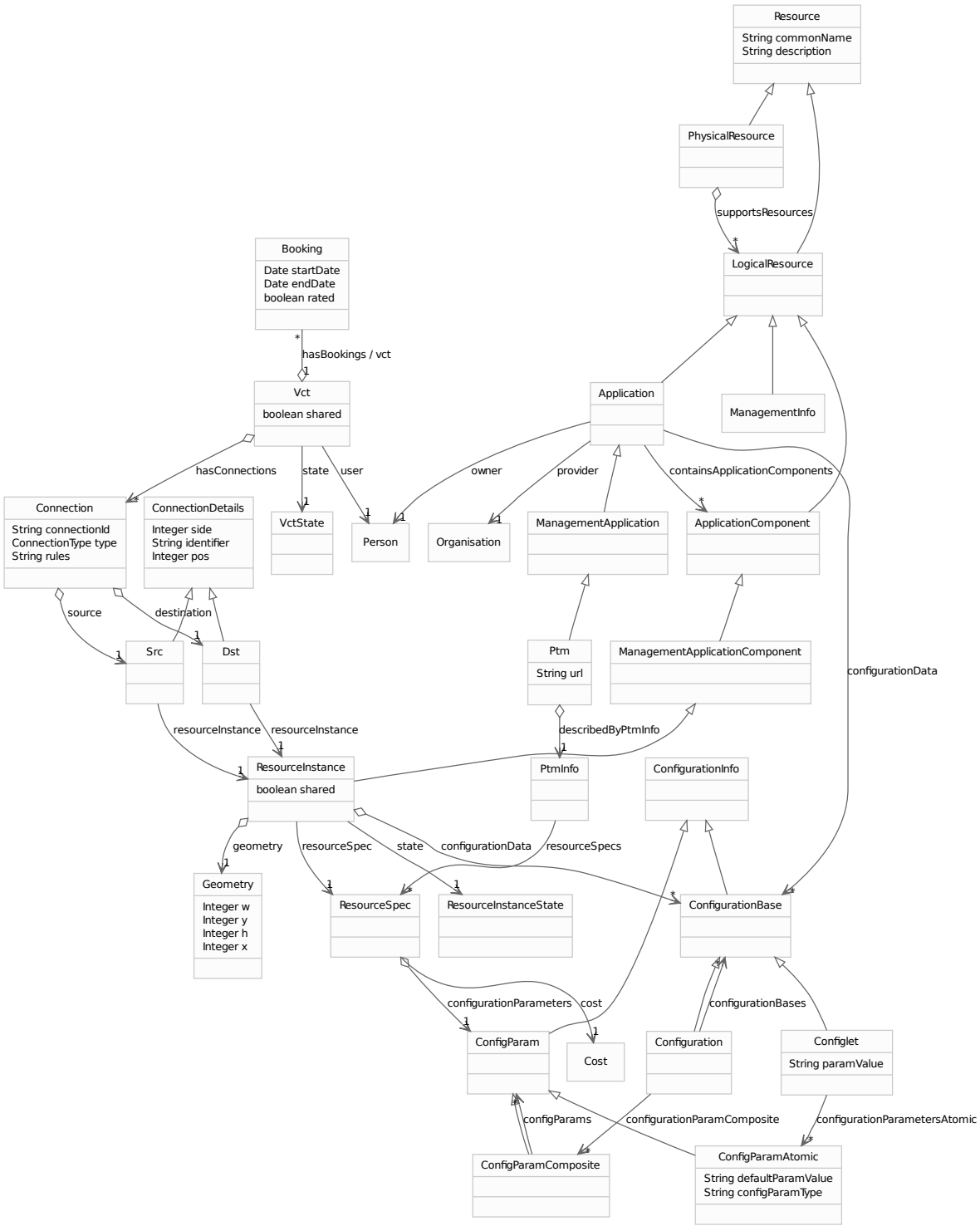


Figure C.1 – The key classes of the information model

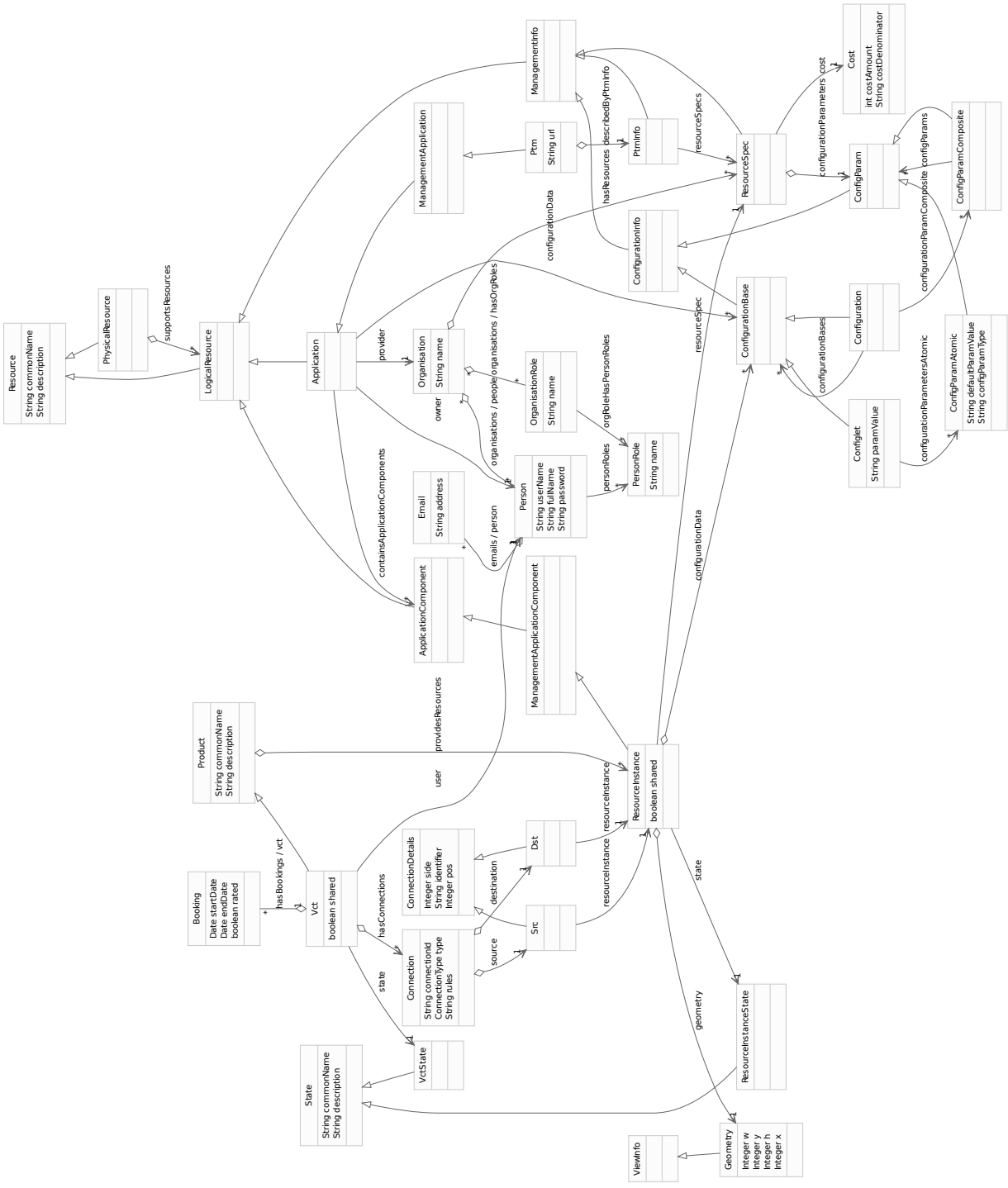


Figure C.2 – The full information model

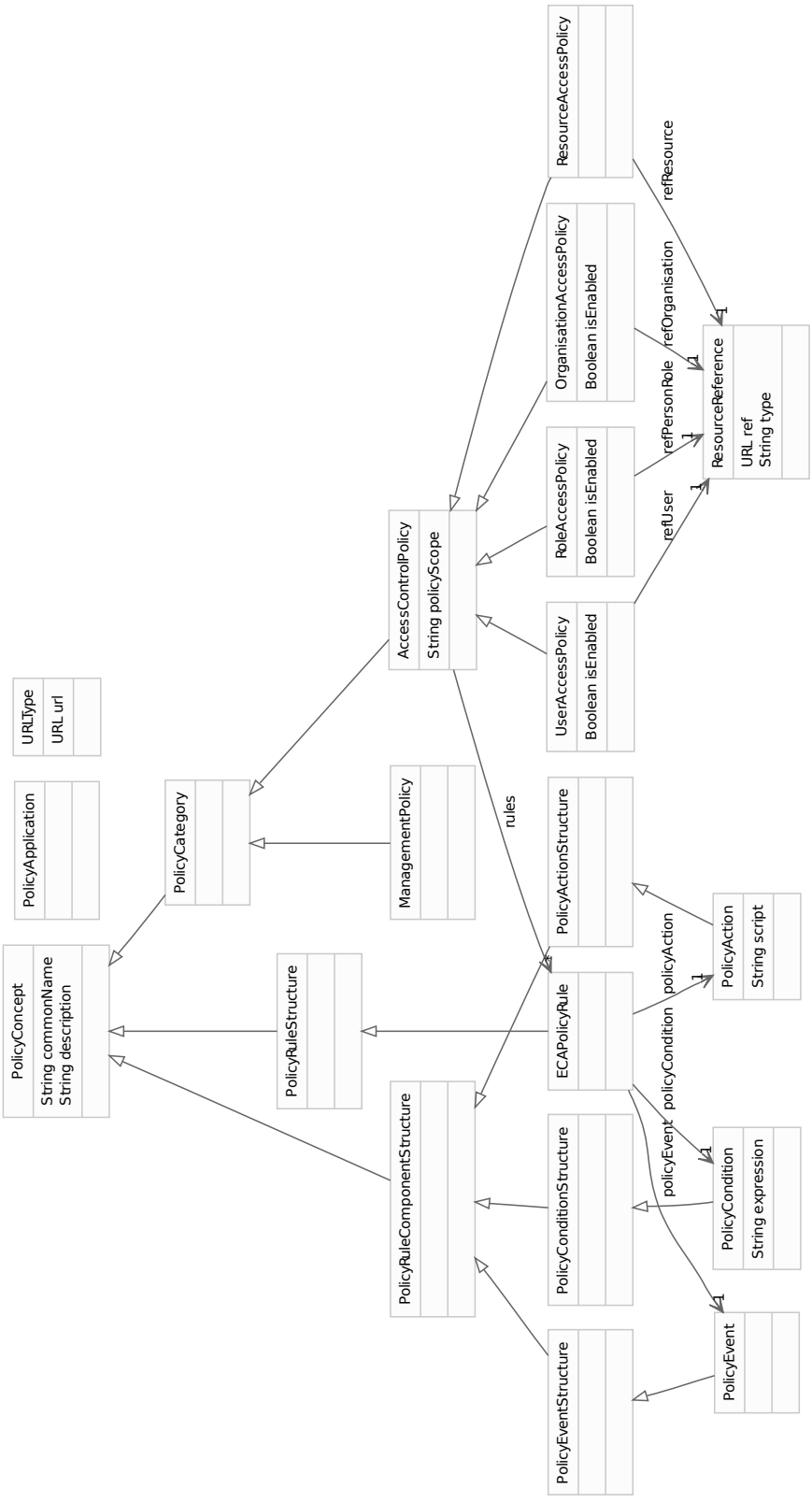


Figure C.3 – The policy model

D

Reference Point T1 Specifications

This annex specifies the reference point T_1 on which domain managers (DMs) expose an interface that allows to control resources inside a federated domain.

General Specifications

This section describes some basic data types used during the specification of T1 interface methods using the Extended Backus Naur Form (EBNF) [171].

Listing D.1 – General specifications

```
None = ;
Digit = "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9" ;
Integer = [ "-" ] Digit { Digit } ;
Float = [ Integer [ "." Digit { Digit } ] ] ;
Character = ? Any valid Unicode character ? ;
String = { Character } ;
Boolean = "true" | "false" ;
LowerCaseLetter = "a" | "b" | "c" | "d" | "e" | "f" | "g" | "h" | "i" | "j" | "k" | "l" | "m"
    | "n" | "o" | "p" | "q" | "r" | "s" | "t" | "u" | "v" | "w" | "x" | "y" | "z" ;
UpperCaseLetter = "A" | "B" | "C" | "D" | "E" | "F" | "G" | "H" | "I" | "J" | "K" | "L" | "M"
    | "N" | "O" | "P" | "Q" | "R" | "S" | "T" | "U" | "V" | "W" | "X" | "Y" | "Z" ;
Letter = UpperCaseletter | LowerCaseLetter ;
Whitespace = ? Any ASCII whitespace character ? ;
PrintableNoDash = Letter | Whitespace | " | "!" | "§" | "$" | "%" | "&" | "/" | "(" | ")" | "="
     | "?" | "\" | "´" | "‘" | "~" | "*" | "+" | "#" | "’" | "_" | "," | "." | ":" | ";" | "<"
     | ">" | "|" | "^" | "°" | "{" | "}" | "[" | "]" ;
Printable = PrintableNoDash | "-" ;
LocalName = PrintableNoDash { PrintableNoDash } ;
Identifier = Printable { Printable } "-" LocalName ;
TypeName = Letter { Letter | Digit | "_" } ;
ParameterName = Letter { Letter | Digit | "_" } ;
ParameterValue = String | Boolean | Float | Integer | Identifier | None
ParameterType = ("string" | "integer" | "float" | "boolean" | "reference") [ - ("array" | "map
    ")] ;
VCTName = Letter { Letter | Digit | "_" } ;
InstanceName = Printable { Printable } ;
Identifier = Printable { Printable } ;
Configuration = ? as defined by XML schema at http://www.fire-teagle.org/T1 ? ;
```

Configuration Data Format

The configuration data accepted and emitted by the T_1 interface methods is specified by the following XML schema which can also be found at the target namespace URL: http://www.fire-teagle.org/T1

Listing D.2 – Configuration data format

```
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns=http://www.w3.org/2001/XMLSchema targetNamespace="http://www.fire-teagle.org/T1"
     xmlns:tns="http://www.fire-teagle.org/T1" elementFormDefault="qualified">

   <element name="configuration">
     <complexType>
        <choice minOccurs="0" maxOccurs="unbounded">
           <element name="integer" type="tns:integer" nillable="true"/>
           <element name="string" type="tns:string" nillable="true"/>
           <element name="float" type="tns:float" nillable="true"/>
           <element name="boolean" type="tns:boolean" nillable="true"/>
           <element name="reference" type="tns:reference" nillable="true"/>
           <element name="array" type="tns:array" nillable="true"/>
           <element name="dict" type="tns:dict" nillable="true"/>
        </choice>
     </complexType>
   </element>

   <complexType name="integer">
     <simpleContent>
        <extension base="integer">
           <attribute name="name"/>
        </extension>
     </simpleContent>
   </complexType>

   <complexType name="string">
     <simpleContent>
        <extension base="string">
           <attribute name="name"/>
        </extension>
     </simpleContent>
   </complexType>

   <simpleType name="referenceValueType">
     <restriction base="string">
        <minLength value="1" />
     </restriction>
   </simpleType>

   <complexType name="reference">
     <simpleContent>
        <extension base="tns:referenceValueType">
           <attribute name="name"/>
        </extension>
     </simpleContent>
   </complexType>

   <complexType name="float">
```

```
49      <simpleContent>
50        <extension base="float">
51          <attribute name="name"/>
52        </extension>
53      </simpleContent>
54    </complexType>
55
56    <complexType name="boolean">
57      <simpleContent>
58        <extension base="boolean">
59          <attribute name="name"/>
60        </extension>
61      </simpleContent>
62    </complexType>
63
64    <complexType name="arrayValueType">
65        <choice minOccurs="0" maxOccurs="unbounded">
66          <element name="integer" type="integer" nillable="true"/>
67          <element name="string" type="string" nillable="true"/>
68          <element name="float" type="float" nillable="true"/>
69          <element name="boolean" type="boolean" nillable="true"/>
70          <element name="reference" type="tns:referenceValueType" nillable="true"/>
71        </choice>
72    </complexType>
73
74    <complexType name="array">
75        <complexContent>
76          <extension base="tns:arrayValueType">
77          <attribute name="name"/>
78        </extension>
79        </complexContent>
80    </complexType>
81
82    <complexType name="dictValueType">
83        <choice minOccurs="0" maxOccurs="unbounded">
84          <element name="integer" type="tns:integer" nillable="true"/>
85          <element name="string" type="tns:string" nillable="true"/>
86          <element name="float" type="tns:float" nillable="true"/>
87          <element name="boolean" type="tns:boolean" nillable="true"/>
88          <element name="reference" type="tns:reference" nillable="true"/>
89        </choice>
90    </complexType>
91
92    <complexType name="dict">
93       <complexContent>
94        <extension base="tns:dictValueType">
95          <attribute name="name"/>
96        </extension>
97      </complexContent>
98    </complexType>
99
100 </schema>
```

T1 Interface Methods

Some of the following subsections contain information published in [115], see also section 5.2.1.

add_resource

```
add_resource(parent_id: Identifier, typename: TypeName, [name: LocalName,] config:
    Configuration, vct: VCTName) : Identifier
```

The `add_resource` operation requests the instantiation of a given resource type with a given configuration as a child of the existing resource instance denoted by `parent_id` optionally specifying a local name. The `vct` parameter indicates which VCT this instance will be part of. Upon success, an identifier of an existing resource instance is returned.

get_resource

```
get_resource(identifier: Identifier): Configuration
```

The `get_resource` operation retrieves configuration information for the existing resource instance denoted by `identifier`.

delete_resource

```
delete_resource(identifier: Identifier): None
```

The `delete_resource` operation requests the deletion of the existing resource instance given by `identifier`. It is up to the DM to decide if the instance will actually be deleted or not. From a federation perspective this request can rather be viewed as an indication that a certain instance is not needed by Teagle anymore and will not further be referenced.

list_resources

```
list_resource(parent_id: Identifier, typename: TypeName): { Identifier }
```

The `list_resources` operation retrieves a list of all resource instances that are regarded as children of the instance denoted by `parent_id` and that are of the type determined by `typename`. If `parent_id` is omitted, all instances at the root of the resource hierarchy mst be listed. If `typename` is omitted, instances if all types must be listed.

update_resources

```
update_resource (id: Identifier, config: Configuration): Configuration
```

The `update_resource` operation requests the reconfiguration of an existing resource instance denoted by `id` with the configuration specified by `config`. This configuration does not have to include all parameters of the resource instance. It is sufficient to include only the parameters that are to be changed. Upon success, the full configuration of the resource instance is returned.

E

The RADL Meta Model and Syntax

Figure E.1 ([149], p. 8) shows the RADL meta model. It is used to specific an abstract syntax and enable the automatic code generation for a target domain manager framework implementation.

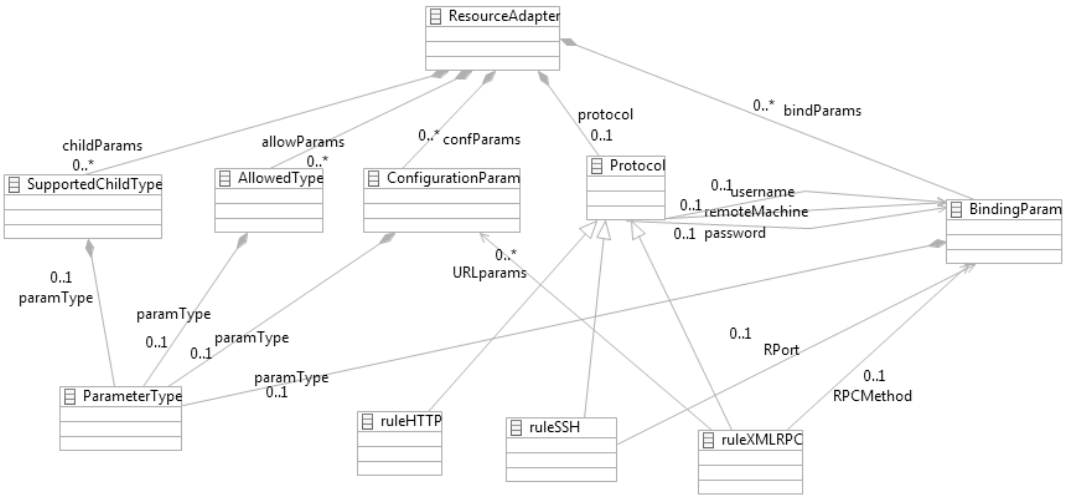


Figure E.1 – The RADL meta-model

Listing E.1 ([149], p. 12) shows a concrete example of a resource adapter definition using the RADL syntax.

Listing E.1 – RADL syntax example

```
1  Resource Adapter "ami_ec2_ra"
2  Configuration Parameters { // Visible Parameters to VCT user
3     String AMI_Id = "ami-2cb05345" description = "An AMI from Amazon list" ;
4     String accessKey; //Provider Amazon credentials
5     String secretKey; //Provider Amazon credentials
6     String InstanceType = "m1.small" description = "AMI type";
7     String AvailabilityZone = "us-east-1a" description ="AMI Region";
8     String PublicDnsName description = "ReadOnly. Available after creating VM";
9     String loginUsername;
10    String loginPassword;
```

```
11    Integer maxNumberOfInstances= "1"; //default is 1 VM instance
12  }
13  On Update {
14    ProcessOnAllConfigurationParametersComplete = YES;
15    RAProtocol Java EC2Wrapper(accessKey, secretKey ){
16      //Call the EC2Wrapper class
17      JExecute createAMInstances(AMI_Id, 1, maxNumberOfInstances, loginUsername, InstanceType,
            AvailabilityZone)
18      JAssign AvailabilityZone = getPublicDnsName()
19    }
20  }
```

F

VCT Specification & REP XML Schemata

This annex lists the XML schemata[1] documents defining a valid VCT specification. They can be used to derive valid REP reference point (see figure 6.21) requests and interpret the repository responses.

First, all *request* schemata are listed. Then, beginning at listing F.14 all *response* formats are given.

Listing F.1 – bookingInstance.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
   <xs:element name="bookingInstance">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="startDate"/>
            <xs:element ref="startDate_year"/>
            <xs:element ref="startDate_month"/>
            <xs:element ref="startDate_day"/>
            <xs:element ref="startDate_hour"/>
            <xs:element ref="startDate_minute"/>
            <xs:element ref="endDate"/>
            <xs:element ref="endDate_year"/>
            <xs:element ref="endDate_month"/>
            <xs:element ref="endDate_day"/>
            <xs:element ref="endDate_hour"/>
            <xs:element ref="endDate_minute"/>
            <xs:element ref="vct.id"/>
            <xs:element ref="rated"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
   <xs:element name="startDate" type="xs:string"/>
   <xs:element name="startDate_year" type="xs:integer"/>
   <xs:element name="startDate_month" type="xs:integer"/>
   <xs:element name="startDate_day" type="xs:integer"/>
   <xs:element name="startDate_hour" type="xs:integer"/>
   <xs:element name="startDate_minute" type="xs:integer"/>
   <xs:element name="endDate" type="xs:string"/>
   <xs:element name="endDate_year" type="xs:integer"/>
   <xs:element name="endDate_month" type="xs:integer"/>
   <xs:element name="endDate_day" type="xs:integer"/>
   <xs:element name="endDate_hour" type="xs:integer"/>
   <xs:element name="endDate_minute" type="xs:integer"/>
   <xs:element name="vct.id" type="xs:integer"/>
   <xs:element name="rated" type="xs:boolean"/>
</xs:schema>
```

Listing F.2 – configParamAtomicInstance.xsd

[1] in terms of XSD files as defined by the World Wide Web Consortium (W3C) in [172, 173]

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
  <xs:element name="configParamAtomicInstance">
    <xs:complexType>
      <xs:all>
        <xs:element ref="commonName"/>
        <xs:element ref="description"/>
        <xs:element ref="defaultParamValue"/>
        <xs:element ref="configParamType"/>
      </xs:all>
    </xs:complexType>
  </xs:element>
  <xs:element name="commonName">
       <xs:simpleType>
        <xs:restriction base="xs:string">
           <xs:minLength value="1"/>
        </xs:restriction>
      </xs:simpleType>
  </xs:element>
  <xs:element name="description" type="xs:string"/>
  <xs:element name="defaultParamValue" type="xs:string"/>
  <xs:element name="configParamType" type="xs:string"/>
</xs:schema>
```

Listing F.3 – configParamCompositeInstance.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
   <xs:element name="configParamCompositeInstance">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="commonName" minOccurs="1"/>
            <xs:element ref="description" minOccurs="1"/>
            <xs:element ref="configParams" minOccurs="0" maxOccurs="unbounded"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
   <xs:element name="description" type="xs:string"/>
   <xs:element name="commonName" type="xs:string"/>
   <xs:element name="configParams" type="xs:integer"/>
</xs:schema>
```

Listing F.4 – configletInstance.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
  <xs:element name="configletInstance">
    <xs:complexType>
      <xs:all>
        <xs:element ref="commonName"/>
        <xs:element ref="description"/>
        <xs:element ref="paramValue"/>
        <xs:element ref="configurationParametersAtomic" minOccurs="0"/>
      </xs:all>
    </xs:complexType>
  </xs:element>
  <xs:element name="commonName">
      <xs:simpleType>
         <xs:restriction base="xs:string">
            <xs:minLength value="1"/>
         </xs:restriction>
      </xs:simpleType>
  </xs:element>
  <xs:element name="description" type="xs:string"/>
  <xs:element name="paramValue" type="xs:string"/>
  <xs:element name="configurationParametersAtomic" type="decimal-or-empty"/>

  <xs:simpleType name="decimal-or-empty">
     <xs:union memberTypes="xs:decimal empty-string" />
  </xs:simpleType>

  <xs:simpleType name="empty-string">
     <xs:restriction base="xs:string">
      <xs:enumeration value="" />
```

```
    </xs:restriction>
  </xs:simpleType>
</xs:schema>
```

Listing F.5 – configurationInstance.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
    XMLSchema-instance" elementFormDefault="qualified">
  <xs:import namespace="http://www.w3.org/2001/XMLSchema-instance" schemaLocation="xsi.xsd"/>
  <xs:element name="configurationInstance">
    <xs:complexType>
      <xs:all>
        <xs:element ref="commonName" minOccurs="1"/>
        <xs:element ref="description" minOccurs="1"/>
        <xs:element ref="configurationParamComposite" minOccurs="0"/>
        <xs:element ref="configurationBases" minOccurs="0"/>
      </xs:all>
    </xs:complexType>
  </xs:element>
  <xs:element name="commonName" type="xs:string"/>
  <xs:element name="configurationParamComposite" type="xs:integer"/>
  <xs:element name="configurationBases" type="xs:integer"/>
  <xs:element name="description" type="xs:string"/>
</xs:schema>
```

Listing F.6 – connectionInstance.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
  <xs:element name="connectionInstance">
    <xs:complexType>
      <xs:all>
        <xs:element ref="ConnectionId"/>
        <xs:element ref="source.id"/>
        <xs:element ref="destination.id"/>
        <xs:element ref="type"/>
        <xs:element ref="rules"/>
      </xs:all>
    </xs:complexType>
  </xs:element>
  <xs:element name="ConnectionId" type="xs:integer"/>
  <xs:element name="source.id" type="xs:integer"/>
  <xs:element name="destination.id" type="xs:integer"/>
  <xs:element name="type">
   <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="REFERENCES"/>
        <xs:enumeration value="CONTAINS"/>
      </xs:restriction>
   </xs:simpleType>
  </xs:element>
  <xs:element name="rules" type="xs:string"/>
</xs:schema>
```

Listing F.7 – costInstance.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
  <xs:element name="costInstance">
    <xs:complexType>
      <xs:all>
        <xs:element ref="costAmount"/>
        <xs:element ref="costDenominator"/>
      </xs:all>
    </xs:complexType>
  </xs:element>
  <xs:element name="costAmount" type="xs:integer"/>
  <xs:element name="costDenominator" type="xs:string"/>
</xs:schema>
```

Listing F.8 – dstInstance.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
```

```
   <xs:element name="dstInstance">
      <xs:complexType>
         <xs:all>
            <xs:element ref="side"/>
            <xs:element ref="resourceInstance"/>
            <xs:element ref="identifier"/>
            <xs:element ref="pos"/>
         </xs:all>
      </xs:complexType>
   </xs:element>
   <xs:element name="side">
      <xs:simpleType>
         <xs:restriction base="xs:integer">
            <xs:enumeration value="0"/>
            <xs:enumeration value="1"/>
         </xs:restriction>
      </xs:simpleType>
    </xs:element>
   <xs:element name="resourceInstance" type="xs:integer"/>
   <xs:element name="identifier" type="xs:string"/>
   <xs:element name="pos">
      <xs:simpleType>
         <xs:restriction base="xs:integer">
            <xs:enumeration value="0"/>
            <xs:enumeration value="1"/>
         </xs:restriction>
      </xs:simpleType>
   </xs:element>
</xs:schema>
```

Listing F.9 – geometryInstance.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
<xs:element name="geometryInstance">
     <xs:complexType>
        <xs:all>
           <xs:element ref="w"/>
           <xs:element ref="y"/>
           <xs:element ref="h"/>
           <xs:element ref="x"/>
        </xs:all>
     </xs:complexType>
   </xs:element>
   <xs:element name="w" type="xs:integer"/>
   <xs:element name="y" type="xs:integer"/>
   <xs:element name="h" type="xs:integer"/>
   <xs:element name="x" type="xs:integer"/>
</xs:schema>
```

Listing F.10 – resourceSpecInstance.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
   <xs:element name="resourceSpecInstance">
      <xs:complexType>
         <xs:all>
            <xs:element ref="commonName"/>
            <xs:element ref="description"/>
            <xs:element ref="cost" minOccurs="0"/>
            <xs:element ref="configurationParameters"/>
         </xs:all>
      </xs:complexType>
   </xs:element>
   <xs:element name="commonName">
      <xs:simpleType>
         <xs:restriction base="xs:string">
            <xs:minLength value="1"/>
         </xs:restriction>
      </xs:simpleType>
  </xs:element>
   <xs:element name="description" type="xs:string"/>
   <xs:element name="cost" type="decimal-or-empty"/>
   <xs:element name="configurationParameters" type="xs:integer"/>
```

```

   <xs:simpleType name="decimal-or-empty">
     <xs:union memberTypes="xs:decimal empty-string" />
   </xs:simpleType>

    <xs:simpleType name="empty-string">
     <xs:restriction base="xs:string">
      <xs:enumeration value="" />
     </xs:restriction>
   </xs:simpleType>
</xs:schema>
```

Listing F.11 – resourceInstanceInstance.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
   <xs:element name="resourceInstanceInstance">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="commonName"/>
            <xs:element ref="geometry"/>
            <xs:element ref="shared"/>
            <xs:element ref="resourceSpec"/>
            <xs:element ref="description"/>
            <xs:element ref="state.id"/>
            <xs:element ref="configurationData" minOccurs="0" maxOccurs="unbounded"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
   <xs:element name="shared" type="xs:boolean"/>
   <xs:element name="resourceSpec" type="xs:integer"/>
   <xs:element name="commonName">
      <xs:simpleType>
         <xs:restriction base="xs:string">
            <xs:minLength value="1"/>
         </xs:restriction>
      </xs:simpleType>
  </xs:element>
   <xs:element name="configurationData" type="xs:integer"/>
   <xs:element name="description" type="xs:string"/>
   <xs:element name="state.id" type="xs:integer"/>
   <xs:element name="geometry" type="decimal-or-empty"/>

   <xs:simpleType name="decimal-or-empty">
     <xs:union memberTypes="xs:decimal empty-string" />
   </xs:simpleType>

    <xs:simpleType name="empty-string">
     <xs:restriction base="xs:string">
      <xs:enumeration value="" />
     </xs:restriction>
   </xs:simpleType>
</xs:schema>
```

Listing F.12 – srcInstance.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
   <xs:element name="srcInstance">
      <xs:complexType>
         <xs:all>
            <xs:element ref="side"/>
            <xs:element ref="resourceInstance"/>
            <xs:element ref="identifier"/>
            <xs:element ref="pos"/>
         </xs:all>
      </xs:complexType>
   </xs:element>
   <xs:element name="side">
      <xs:simpleType>
         <xs:restriction base="xs:integer">
            <xs:enumeration value="0"/>
            <xs:enumeration value="1"/>
         </xs:restriction>
```

```
      </xs:simpleType>
    </xs:element>
   <xs:element name="resourceInstance" type="xs:integer"/>
   <xs:element name="identifier" type="xs:string"/>
   <xs:element name="pos">
      <xs:simpleType>
         <xs:restriction base="xs:integer">
            <xs:enumeration value="0"/>
            <xs:enumeration value="1"/>
         </xs:restriction>
      </xs:simpleType>
   </xs:element>
</xs:schema>
```

Listing F.13 – vctInstance.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
   <xs:element name="vctInstance">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="commonName"/>
            <xs:element ref="description"/>
            <xs:element ref="shared"/>
            <xs:element ref="state.id"/>
            <xs:element ref="user"/>
            <xs:element ref="hasConnections" minOccurs="0" maxOccurs="unbounded"/>
            <xs:element ref="hasBookings" minOccurs="0" maxOccurs="unbounded"/>
            <xs:element ref="providesResources" minOccurs="0" maxOccurs="unbounded"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
   <xs:element name="commonName" type="xs:string"/>
   <xs:element name="description" type="xs:string"/>
   <xs:element name="hasConnections" type="xs:integer"/>
   <xs:element name="hasBookings" type="xs:integer" nillable="true"/>
   <xs:element name="user" type="xs:integer"/>
   <xs:element name="shared" type="xs:boolean"/>
   <xs:element name="state.id" type="xs:integer"/>
   <xs:element name="providesResources" type="xs:integer"/>
</xs:schema>
```

In the following, the format of the XML documents carried in the body of Teagle repository HTTP response messages are given.

Listing F.14 – booking.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
   <xs:element name="list">
      <xs:complexType>
         <xs:sequence maxOccurs="unbounded" minOccurs="0">
            <xs:element ref="booking"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>

   <xs:element name="booking">
      <xs:complexType>
         <xs:all>
            <xs:element ref="startDate"/>
            <xs:element ref="vct"/>
          <xs:element ref="endDate"/>
            <xs:element ref="rated"/>
         </xs:all>
         <xs:attribute name="id" type="xs:integer" use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="startDate">
     <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:pattern value="\d{4}-(0[1-9]|1[012])-(0[1-9]|[12][0-9]|3[01]) \d{2}:\d{2}:\d{2}.\d
```

```
            GMT"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:element>
   <xs:element name="vct">
     <xs:complexType>
       <xs:attribute name="id" type="xs:integer" use="required"/>
     </xs:complexType>
   </xs:element>
   <xs:element name="endDate">
     <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:pattern value="\d{4}-(0[1-9]|1[012])-(0[1-9]|[12][0-9]|3[01]) \d{2}:\d{2}:\d{2}.\d
            GMT"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:element>

   <xs:element name="rated" type="xs:boolean"/>
</xs:schema>
```

Listing F.15 – configParamAtomic.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
   <xs:element name="list">
      <xs:complexType>
         <xs:sequence maxOccurs="unbounded" minOccurs="0">
            <xs:element ref="configParamAtomic"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>

   <xs:element name="configParamAtomic">
      <xs:complexType>
         <xs:all>
            <xs:element ref="commonName"/>
            <xs:element ref="description"/>
            <xs:element ref="defaultParamValue"/>
            <xs:element ref="configParamType"/>
         </xs:all>
         <xs:attribute name="id" type="xs:integer" use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="commonName">
      <xs:simpleType>
         <xs:restriction base="xs:string">
            <xs:minLength value="1"/>
         </xs:restriction>
      </xs:simpleType>
   </xs:element>
   <xs:element name="description" type="xs:string"/>
   <xs:element name="defaultParamValue" type="xs:string"/>
   <xs:element name="configParamType" type="xs:string"/>
</xs:schema>
```

Listing F.16 – configParamComposite.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:cmn="http://repos.com/CommonTypes
    " elementFormDefault="qualified">
   <xs:import namespace="http://repos.com/CommonTypes" schemaLocation="CommonTypes.xsd"/>
   <xs:element name="list">
      <xs:complexType>
         <xs:sequence maxOccurs="unbounded" minOccurs="0">
            <xs:element ref="configParamComposite"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>

   <xs:element name="configParamComposite">
      <xs:complexType>
         <xs:all>
            <xs:element ref="description"/>
```

```
            <xs:element ref="commonName"/>
            <xs:element ref="configParams"/>
         </xs:all>
         <xs:attribute name="id" type="xs:integer" use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="description" type="xs:string"/>
   <xs:element name="commonName">
      <xs:simpleType>
         <xs:restriction base="xs:string">
            <xs:minLength value="1"/>
         </xs:restriction>
      </xs:simpleType>
   </xs:element>
   <xs:element name="configParams" nillable="true">
      <xs:complexType>
         <xs:sequence minOccurs="0" maxOccurs="unbounded">
            <xs:choice>
               <xs:element name="configParamComposite" type="cmn:configParamComp"/>
               <xs:element ref="configParamAtomic"/>
            </xs:choice>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
   <xs:element name="configParamAtomic">
      <xs:complexType>
         <xs:attribute name="id" type="xs:integer" use="required"/>
      </xs:complexType>
   </xs:element>
</xs:schema>
```

Listing F.17 – configlet.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
   <xs:element name="list">
      <xs:complexType>
         <xs:sequence maxOccurs="unbounded" minOccurs="0">
            <xs:element ref="configlet"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>

   <xs:element name="configlet">
      <xs:complexType>
         <xs:all>
            <xs:element ref="commonName"/>
            <xs:element ref="description"/>
            <xs:element ref="paramValue"/>
            <xs:element ref="configurationParametersAtomic"/>
         </xs:all>
         <xs:attribute name="id" type="xs:integer" use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="commonName">
      <xs:simpleType>
         <xs:restriction base="xs:string">
            <xs:minLength value="1"/>
         </xs:restriction>
      </xs:simpleType>
   </xs:element>
   <xs:element name="description" type="xs:string"/>
   <xs:element name="paramValue" type="xs:string"/>
   <xs:element name="configurationParametersAtomic">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="configParamAtomic" minOccurs="0" maxOccurs="unbounded"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
   <xs:element name="configParamAtomic">
      <xs:complexType>
         <xs:attribute name="id" type="xs:integer" use="required"/>
      </xs:complexType>
```

```
   </xs:element>
</xs:schema>
```

Listing F.18 – configuration.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
   <xs:element name="list">
      <xs:complexType>
         <xs:sequence maxOccurs="unbounded" minOccurs="0">
            <xs:element ref="configuration"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>

   <xs:element name="configuration">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="configurationParamComposite"/>
            <xs:element ref="commonName"/>
            <xs:element ref="description"/>
            <xs:element ref="configurationBases"/>
         </xs:sequence>
         <xs:attribute name="id" type="xs:integer" use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="configurationParamComposite">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="configParamComposite" minOccurs="0" maxOccurs="unbounded"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
   <xs:element name="configParamComposite">
      <xs:complexType>
         <xs:attribute name="id" type="xs:integer" use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="commonName">
      <xs:simpleType>
         <xs:restriction base="xs:string">
            <xs:minLength value="1"/>
         </xs:restriction>
      </xs:simpleType>
   </xs:element>
   <xs:element name="description" type="xs:string"/>
   <xs:element name="configurationBases">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="configlet" minOccurs="0" maxOccurs="unbounded"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
   <xs:element name="configlet">
      <xs:complexType>
         <xs:attribute name="id" type="xs:integer" use="required"/>
      </xs:complexType>
   </xs:element>
</xs:schema>
```

Listing F.19 – connection.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
   <xs:element name="list">
      <xs:complexType>
         <xs:sequence maxOccurs="unbounded" minOccurs="0">
            <xs:element ref="connection"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>

   <xs:element name="connection">
      <xs:complexType>
```

```
13          <xs:sequence>
14             <xs:element ref="source"/>
15             <xs:element ref="connectionId"/>
16             <xs:element ref="type"/>
17             <xs:element ref="rules"/>
18             <xs:element ref="destination"/>
19          </xs:sequence>
20          <xs:attribute name="id" type="xs:integer" use="required"/>
21       </xs:complexType>
22    </xs:element>
23    <xs:element name="source">
24       <xs:complexType>
25          <xs:attribute name="id" type="xs:integer" use="required"/>
26       </xs:complexType>
27    </xs:element>
28    <xs:element name="connectionId" type="xs:string"/>
29    <xs:element name="type">
30       <xs:complexType>
31          <xs:simpleContent>
32             <xs:extension base="xs:string">
33                <xs:attribute name="enumType" type="xs:string" use="required" fixed="
                       ConnectionType"/>
34             </xs:extension>
35          </xs:simpleContent>
36       </xs:complexType>
37    </xs:element>
38    <!-- This should define the values that are allowed in the type element -->
39    <xs:simpleType name="type_values">
40       <xs:restriction base="xs:string">
41          <xs:enumeration value="REFERENCES"/>
42          <xs:enumeration value="CONTAINS"/>
43       </xs:restriction>
44    </xs:simpleType>
45    <xs:element name="rules" type="xs:string"/>
46    <xs:element name="destination">
47       <xs:complexType>
48          <xs:attribute name="id" type="xs:integer" use="required"/>
49       </xs:complexType>
50    </xs:element>
51 </xs:schema>
```

Listing F.20 – cost.xsd

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
3     <xs:element name="list">
4        <xs:complexType>
5           <xs:sequence maxOccurs="unbounded" minOccurs="0">
6              <xs:element ref="cost"/>
7           </xs:sequence>
8        </xs:complexType>
9     </xs:element>
10
11    <xs:element name="cost">
12       <xs:complexType>
13          <xs:sequence>
14             <xs:element ref="costAmount"/>
15             <xs:element ref="costDenominator"/>
16          </xs:sequence>
17          <xs:attribute name="id" type="xs:integer" use="required"/>
18       </xs:complexType>
19    </xs:element>
20    <xs:element name="costAmount" type="xs:integer"/>
21    <xs:element name="costDenominator" type="xs:string"/>
22 </xs:schema>
```

Listing F.21 – dst.xsd

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
3     <xs:element name="list">
4        <xs:complexType>
5           <xs:sequence maxOccurs="unbounded" minOccurs="0">
6              <xs:element ref="dst"/>
```

```
         </xs:sequence>
      </xs:complexType>
   </xs:element>

   <xs:element name="dst">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="side"/>
            <xs:element ref="resourceInstance"/>
            <xs:element ref="identifier"/>
            <xs:element ref="pos"/>
         </xs:sequence>
         <xs:attribute name="id" type="xs:integer" use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="side" type="xs:integer"/>
   <xs:element name="resourceInstance">
      <xs:complexType>
         <xs:attribute name="id" type="xs:integer" use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="identifier" type="xs:string"/>
   <xs:element name="pos" type="xs:integer"/>
</xs:schema>
```

Listing F.22 – geometry.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
   <xs:element name="list">
      <xs:complexType>
         <xs:sequence maxOccurs="unbounded" minOccurs="0">
            <xs:element ref="geometry"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>

   <xs:element name="geometry">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="w"/>
            <xs:element ref="y"/>
            <xs:element ref="h"/>
            <xs:element ref="x"/>
         </xs:sequence>
         <xs:attribute name="id" type="xs:integer" use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="w" type="xs:integer"/>
   <xs:element name="y" type="xs:integer"/>
   <xs:element name="h" type="xs:integer"/>
   <xs:element name="x" type="xs:integer"/>
</xs:schema>
```

Listing F.23 – resourceInstance.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
   <xs:element name="list">
      <xs:complexType>
         <xs:sequence maxOccurs="unbounded" minOccurs="0">
            <xs:element ref="resourceInstance"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>

   <xs:element name="resourceInstance">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="shared"/>
            <xs:element ref="resourceSpec"/>
            <xs:element ref="commonName"/>
            <xs:element ref="configurationData"/>
            <xs:element ref="description"/>
```

```
            <xs:element ref="state"/>
            <xs:element ref="geometry"/>
         </xs:sequence>
         <xs:attribute name="id" type="xs:integer" use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="shared" type="xs:boolean"/>
   <xs:element name="resourceSpec">
      <xs:complexType>
         <xs:attribute name="id" type="xs:integer" use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="commonName">
      <xs:simpleType>
         <xs:restriction base="xs:string">
            <xs:minLength value="1"/>
         </xs:restriction>
      </xs:simpleType>
   </xs:element>
   <xs:element name="configurationData">
      <xs:complexType>
         <xs:sequence minOccurs="0" maxOccurs="unbounded">
            <xs:choice>
               <xs:element ref="configuration"/>
               <xs:element ref="configlet"/>
            </xs:choice>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
   <xs:element name="configuration">
      <xs:complexType>
         <xs:attribute name="id" type="xs:integer" use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="configlet">
      <xs:complexType>
         <xs:attribute name="id" type="xs:integer" use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="description" type="xs:string"/>
   <xs:element name="state">
      <xs:complexType>
         <xs:attribute name="id" type="xs:integer" use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="geometry">
      <xs:complexType>
         <xs:attribute name="id" type="xs:integer" use="required"/>
      </xs:complexType>
   </xs:element>
</xs:schema>
```

Listing F.24 – resourceSpec.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
   <xs:element name="list">
      <xs:complexType>
         <xs:sequence maxOccurs="unbounded" minOccurs="0">
            <xs:element ref="resourceSpec"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>

   <xs:element name="resourceSpec">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="commonName"/>
            <xs:element ref="description"/>
            <xs:element ref="cost"/>
            <xs:element ref="configurationParameters"/>
         </xs:sequence>
         <xs:attribute name="id" type="xs:integer" use="required"/>
      </xs:complexType>
```

```
   </xs:element>
   <xs:element name="commonName">
      <xs:simpleType>
         <xs:restriction base="xs:string">
            <xs:minLength value="1"/>
         </xs:restriction>
      </xs:simpleType>
   </xs:element>
   <xs:element name="description" type="xs:string"/>
   <xs:element name="cost">
      <xs:complexType>
         <xs:attribute name="id" type="xs:integer"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="configurationParameters">
      <xs:complexType>
         <xs:attribute name="id" type="xs:integer" use="required"/>
      </xs:complexType>
   </xs:element>
</xs:schema>
```

Listing F.25 – src.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
   <xs:element name="list">
      <xs:complexType>
         <xs:sequence maxOccurs="unbounded" minOccurs="0">
            <xs:element ref="src"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>

   <xs:element name="src">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="side"/>
            <xs:element ref="resourceInstance"/>
            <xs:element ref="identifier"/>
            <xs:element ref="pos"/>
         </xs:sequence>
         <xs:attribute name="id" type="xs:integer" use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="side" type="xs:integer"/>
   <xs:element name="resourceInstance">
      <xs:complexType>
         <xs:attribute name="id" type="xs:integer" use="required"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="identifier" type="xs:string" nillable="false"/>
   <xs:element name="pos" type="xs:integer"/>
</xs:schema>
```

Listing F.26 – vct.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
   <xs:element name="list">
      <xs:complexType>
         <xs:sequence maxOccurs="unbounded" minOccurs="0">
            <xs:element ref="vct"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>

   <xs:element name="vct">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="shared"/>
            <xs:element ref="hasConnections"/>
            <xs:element ref="hasBookings"/>
            <xs:element ref="commonName"/>
            <xs:element ref="description"/>
```

```
19            <xs:element ref="state"/>
20            <xs:element ref="providesResources"/>
21            <xs:element ref="user"/>
22         </xs:sequence>
23         <xs:attribute name="id" type="xs:integer" use="required"/>
24      </xs:complexType>
25   </xs:element>
26
27   <xs:element name="shared" type="xs:boolean"/>
28
29   <xs:element name="hasConnections">
30      <xs:complexType>
31         <xs:sequence>
32            <xs:element ref="connection" minOccurs="0" maxOccurs="unbounded"/>
33         </xs:sequence>
34      </xs:complexType>
35      </xs:element>
36      <xs:element name="hasBookings">
37      <xs:complexType>
38         <xs:sequence>
39            <xs:element ref="booking" minOccurs="0" maxOccurs="unbounded"/>
40         </xs:sequence>
41      </xs:complexType>
42   </xs:element>
43   <xs:element name="connection">
44      <xs:complexType>
45         <xs:attribute name="id" type="xs:integer" use="required"/>
46      </xs:complexType>
47   </xs:element>
48   <xs:element name="booking">
49      <xs:complexType>
50         <xs:attribute name="id" type="xs:integer" use="required"/>
51      </xs:complexType>
52   </xs:element>
53   <xs:element name="commonName">
54      <xs:simpleType>
55         <xs:restriction base="xs:string">
56            <xs:minLength value="1"/>
57         </xs:restriction>
58      </xs:simpleType>
59   </xs:element>
60   <xs:element name="description" type="xs:string"/>
61   <xs:element name="state">
62      <xs:complexType>
63         <xs:attribute name="id" type="xs:integer" use="required"/>
64      </xs:complexType>
65   </xs:element>
66   <xs:element name="providesResources">
67      <xs:complexType>
68         <xs:sequence>
69            <xs:element ref="resourceInstance" minOccurs="0" maxOccurs="unbounded"/>
70         </xs:sequence>
71      </xs:complexType>
72   </xs:element>
73   <xs:element name="resourceInstance">
74      <xs:complexType>
75         <xs:attribute name="id" type="xs:integer" use="required"/>
76      </xs:complexType>
77   </xs:element>
78   <xs:element name="user">
79      <xs:complexType>
80         <xs:attribute name="id" type="xs:integer" use="required"/>
81      </xs:complexType>
82   </xs:element>
83 </xs:schema>
```

Listing F.27 – CommonTypes.xsd

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <xs:schema targetNamespace="http://repos.com/CommonTypes" xmlns:xs="http://www.w3.org/2001/
       XMLSchema" elementFormDefault="qualified">
3     <xs:complexType name="configParamComp">
4        <xs:attribute name="id" type="xs:integer" use="required"/>
5     </xs:complexType>
```

```
6     <xs:complexType name="configlet">
7        <xs:attribute name="id" type="xs:integer" use="required"/>
8     </xs:complexType>
9     <xs:complexType name="configuration">
10       <xs:attribute name="id" type="xs:integer" use="required"/>
11    </xs:complexType>
12 </xs:schema>
```

G Typographical Conventions

G.1 Literature References

For citations, the Institute of Electrical and Electronics Engineers (IEEE) citation style is used. This style supports references of online documents, patents, and standards in addition to the basic BibTeX styles and builds on the IEEE citation-sequence system that lists and numbers references in the sequence in which they are first cited. Subsequent citations of the same document use the same number as that of its initial citation. In the Portable Document Format (PDF) version of this document the reference numbers are hyperlinks. Clicking upon them allows to navigate to the respective reference entry in the bibliography section.

Online resource are clearly marked using the following pattern proceeding the URL of the respective resource.

Example: [Online]. Available: http://example.url

In case of webpages the date on which the page has been accessed for the last time is proceeding all other reference details. This is in line with the IEEE citation style and is done to emphasize the short lifetime of information published on webpages.

Example: (2011, March) The Open Grid Forum website. [Online]. Available: http://www.ogf.org

If figures have been extracted from other sources, the source and page number are indicated in the text behind the figure reference (not in the figure caption). Reference number and page number are placed in round brackets as demonstrated by the following example.

Example: As shown in figure 1.1 ([1], p. 168), the federation concept ...

The same mechanism is applied to indicate the sources of tables and listings that originate from referenced literature.

G.2 Footnotes

Footnotes are used to provide extra information that is not important enough to be mentioned elsewhere. This is often the case with URLs. Important webpages and their respective URLs are listed in the bibliography section. However, sometimes some piece of information is contained on a subpage of the main website that is referenced in the bibliography. Such websites are given as footnotes to ease the reading and prevent an inflation of the bibliography section. Also, links to prototypes or other software components are usually given as footnotes.

G.3 Acronyms

There is a list of acronyms starting on page xxiii. In the PDF version of this document, acronyms are hyperlinks. Clicking upon them allows to navigate to the respective acronym entry in the acronyms section.

G.4 Source Code and Formal Syntax Notation

Source code snippets are given as separate listings in a typewriter-style format. Line numbers are printed in very small font on the left hand side of the listing in case it contains more than one line.

Formal syntax notations (e.g. for interfaces or message formats) are given as separate listings in EBNF. The EBNF is a formal metasyntax notation (or syntactic metalanguage). It is specified by an International Organization for Standardization (ISO) international standard [171].

Listing G.1 – Example code snippet

```
1 public static void main(String[] args) {
2    System.out.println("Hello World!");
3 }
```

Listing G.2 – Example message format specification in EBNF

```
1 <PROV_REQ> ::= {vct_id}
2 [resource_id] ;   null | static id
3 [config_data] ;   null | data
4 [call_back] ;     null | call back url
```

G.5 Quotations

Quotations extracted from referenced literature are handled according to the Chicago Manual of Style (CMOS) [174]. Quotations are either *run in* or *set off* from the text as block quotations.

In case of *run in* quotations, quotation marks indicate the beginning and the end of the quotation. The source is given at the end of the quotation using the IEEE citation style as described before. If the page is also given, the entire reference is placed in round brackets. An

example *run in* quotation is given in the following: “This is an example quotation!” ([reference number], p. pagenumer)

To emphasize some quotations, for example in case of important definitions, or because the quoted text spans multiple lines, a block quote layout is used. Block quotations are *set off* from the text by starting a new line and indenting the entire block. Additionally, the text is set in italic. Example:

> *A design artifact is complete and effective when it satisfies the requirements and constraints of the problem it was meant to solve.* [11], p. 85

Note that quotations are always given word for word. Changes to the original wording are placed in square brackets. Also, any omitted words or entire sentences that have been omitted are indicated by three dots placed in square brackets. If original work has been paraphrased, the source is indicated by giving the reference and page number. No quotation marks or specific type setting are used in this case.

G.6 Important Terms

Important terms specifically to be noted by the reader are typeset in *italic font*.

G.7 Gender Considerations

Unfortunately, the English language misses a genderless pronoun. Therefore, the pronoun *he*, which is the masculine pronoun, is also often used in sentence construction to refer to both sexes. This is generally done to avoid clumsy expressions like *he/she* or *s/he*.

For this work, it can be assumed that whenever the male pronoun *he* or the corresponding determiner *his* is used, that it refers to subjects of both genders. The consistent use of the male form is motivated by the ease of reading, simpleness, and clarity of writing rather than discrimination of any sort. Example: “The user expressed his need.” In this sentence, *the user* can be of any sex.

Acronyms

3G	3rd Generation
3GPP	3rd Generation Partnership Project
ABAC	attribute-based access control
AM	aggregate manager
ASCII	American Standard Code for Information Interchange
API	Application Programming Interface
AQCM	Automated Quality Certification Model
Arpanet	Advanced Research Projects Agency Network
ATM	asynchronous transfer mode
BEN	Breakable Experimental Network
BMBF	Federal Ministry of Education and Research
BPEL	Business Process Execution Language
CA	certification authority
CAPEX	capital expenditure
CCIF	Cloud Computing Interoperability Forum
CDMI	Cloud Data Management Interface
CEDL	Canonical Experiment Description Language
CERN	European Organization for Nuclear Research
CM	component manager
CMOS	Chicago Manual of Style
COD	Cluster-on-Demand
CoT	Circle of Trust
CPU	Central Processing Unit
CRUD	create, read, update, delete
CRL	certificate revocation list
CSI	Cloud Storage Initiative
CSA	Cloud Security Alliance
CSCF	Call Session Control Function
CSSP	Cloud Services Specification Project
DEN-ng	Directory Enabled Networks - next generation
DFA	DETER Federation Architecture
DM	domain manager
DMTF	Distributed Management Task Force

DNS	Domain Name System
DoW	Description of Work
DS	design science
EBNF	Extended Backus Naur Form
EC	European Commission
ECLC	Enterprise Cloud Leadership Council
EICT	European Center for Information and Communication Technologies
EMF	Eclipse Modeling Framework
EPC	Evolved Packet Core
ETSI	European Telecommunications Standards Institute
FCI	Federation Computing Interface
FIDM	federated identity management
FI	Future Internet
FIND	Future Internet Design
FIRE	Future Internet Research and Experimentation
FG Cloud	Focus Group on Cloud Computing
FM	Federation Manager
FOKUS	Short form for Fraunhofer Institute for Open Communication Systems
FP7	seventh framework programme for research and technological development
FP6	sixth framework programme
FPGA	Field Programmable Gate Arrays
GENI	Global Environment for Network Innovations
G-Lab	German-Lab
GPO	GENI Project Office
GRL	goal-oriented requirements language
GSN	Green Star Network
HPDMnet	High Performance Digital Media Network
HSS	Home Subscriber Server
HTTP	Hypertext Transfer Protocol
HTML	Hypertext Markup Language
IaaS	Infrastructure as a Service
I-CSCF	Interrogating-Call Session Control Function
ICT	Information and Communication Technology
IDM	identity management
IdP	Identity Provider
IDspec	Format of how federated identity information is presented and conveyed
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
i/f	Interface
IGW	interconnection gateway
IM	Information Model
IMS	IP Multimedia Subsystem
IoM/C	Internet of Media and Content
IoS	Internet of Services
IoT	Internet of Things
IP	Internet Protocol

iRODS	integrated Rule Oriented Data System
IRP	integration reference point
ISO	International Organization for Standardization
IS	information system
IT	information technology
ITU	International Telecommunication Union
ITU-T	Telecommunication Standardization Sector of ITU
JNLP	Java^{TM} Network Launching Protocol
JSP	Java Server Pages
LAN	local area network
L2TP	Layer Two Tunneling Protocol
LTE	3GPP Long Term Evolution
MA	management authority
MDA	Model Driven Architecture
MR_A	Set of federated resources provided by domain A
MOF	Meta Object Facility
NAT	network address translation
NDL	network description language
NFS	Network File System
NICTA	National ICT Australia
NIST	National Institute of Standards and Technology
NGG	Next Generation Grid
NGN	Next Generation Network
NoF	Network of the Future
NSF	National Science Foundation
ns	network simulator
OASIS	Organization for the Advancement of Structured Information Standards
OCC	Open Cloud Consortium
OCCI	Open Cloud Computing Interface
OCCI-WG	Open Cloud Computing Interface Working Group
OE	orchestration engine
OEDL	OMF Experiment Description Language
OF	Open Flow
OFA	Open Federation Alliance
OGF	Open Grid Forum
OMA	Open Mobile Alliance
OMFold	Orbit Management Framework
OMF	Control and Management Framework
OMG	Object Management Group
OPEX	operational expenditure
OGSA	Open Grid Services Architecture
ORBIT	Open Access Research Testbed for Next-Generation Wireless Networks
OS	Operating System
OSI model	Open Systems Interconnection model
OVF	Open Virtualization Format
OWL	Web Ontology Language

PaaS	Platform as a Service
P-CSCF	Proxy-Call Session Control Function
PDF	Portable Document Format
PE	policy engine
PEL	policy expression language
PI	principle investigator
PII	Pan-European laboratory infrastructure implementation
PKI	Public Key Infrastructure
PLC	PlanetLab Central
PLE	PlanetLab Europe
PLJ	PlanetLab Japan
POJO	plain old Java object
Pspec	Policy description format in a federated environment
PTM	Panlab Testbed Manager
PWI	Panlab web interface
QoE	Quality of Experience
QoS	Quality of Service
QRAI	Qualified resource adapter identifier
R_A	Resource provided by domain A
RA	resource adapter
RADL	resource adapter description language
RAM	Random-Access Memory
R&D	Research and Development
RAI	Resource adapter identifier
RII	Resource instance identifier
RDF	Resource Description Framework
RENCI	Renaissance Computing Institute
REST	Representational State Transfer
RFC	request for comments
ROI	Return on Investment
RP	request processor
RPC	Remote Procedure Call
Rspec	resource specification
$Rspec_A$	Format of the resource specification for domain A
SA	slice authority
SaaS	Software as a Service
SAP	service access point
S-CSCF	Serving-Call Session Control Function
SDO	standard development organization
SDP	service delivery platform
SFAv1	Slice-based Facility Architecture
SFAv2	Slice-based Federation Architecture
SGML	Standard Generalized Markup Language
SHARP	Secure Highly Available Resource Peering
SLA	service level agreement
SME	small-to-medium sized enterprise

SMS	Short Message Service
SNIA	Storage Networking Industry Association
SNMP	Simple Network Management Protocol
SOA	Service Oriented Architecture
SoaML	Service oriented architecture Modeling Language
SOAP	Simple Object Access Protocol
SPATEL	Spice Advanced language for TELecommunication services
SPML	Service Provisioning Markup Language
SSA	specific support action
SSH	Secure Shell
SSL	Secure Sockets Layer
SSO	Single Sign-On
SUT	system under test
SVN	Apache Subversion
TC	technical committee
Telco	telecommunications operator
TIED	Trial Integration Environment built on DETER
TGW	Teagle Gateway
TMF	TM Forum (formerly teleManagement Forum)
UML	Unified Modeling Language
UCM	use case map
URL	Uniform Resource Locator
URI	Uniform Resource Identifier
URN	user requirements notation
U.S.	United States of America
VCT	Virtual Customer Testbed
VG	Virtual resource grouping
VLAN	virtual local area network
VM	virtual machine
VMAN	Virtualization Management
VOIP	voice over IP
VPN	virtual private network
Vspec	Specification of a virtual resource grouping
W3C	World Wide Web Consortium
WS-Federation	Web Services Federation Language
WiMAX	Worldwide Interoperability for Microwave Access
WS	Web Service
WWW	World Wide Web
WSDL	Web Services Description Language
WSN	wireless sensor network
WSRF	Web Services Resource Framework
XaaS	Everything as a Service
XCAP	XML configuration access protocol
XEN	Name of the XEN virtual machine monitor/hypervisor
XML	Extensible Markup Language
XML-RPC	Extensible Markup Language Remote Procedure Call

XSD XML schema document

Bibliography

[1] S. Wahle, C. Tranoris, S. Denazis, A. Gavras, K. Koutsopoulos, T. Magedanz, and S. Tompros, "Emerging Testing Trends and the Panlab Enabling Infrastructure," *IEEE Communications Magazine*, vol. 49, no. 3, pp. 167–175, March 2011, ISSN: 0163-6804. [Online]. Available: http://dx.doi.org/10.1109/MCOM.2011.5723816

[2] A. Gavras, Ed., *Experimentally driven research white paper*, April 2010, version 1. [Online]. Available: http://www.ict-fireworks.eu/fileadmin/documents/Experimentally_driven_research_V1.pdf

[3] (2011, February) Merriam Webster dictionary entry for "generic". Encyclopedia Britannica. [Online]. Available: http://www.merriam-webster.com/dictionary/generic

[4] (2011, February) Merriam Webster dictionary entry for "framework". Encyclopedia Britannica. [Online]. Available: http://www.merriam-webster.com/dictionary/framework

[5] (2011, February) Merriam Webster dictionary entry for "resource". Encyclopedia Britannica. [Online]. Available: http://www.merriam-webster.com/dictionary/resource

[6] (2011, February) Wikipedia entry for "resource (computer science)". Wikimedia Foundation, Inc. [Online]. Available: http://en.wikipedia.org/wiki/Resource_(computer_science)

[7] (2011, February) Wikipedia entry for "resource (web)". Wikimedia Foundation, Inc. [Online]. Available: http://en.wikipedia.org/wiki/Resource_(Web)

[8] (2011, February) Merriam Webster dictionary entry for "federation". Encyclopedia Britannica. [Online]. Available: http://www.merriam-webster.com/dictionary/federation

[9] (2011, February) Merriam Webster dictionary entry for "federal". Encyclopedia Britannica. [Online]. Available: http://www.merriam-webster.com/dictionary/federal

[10] (2011, February) Wikipedia entry for "federation". Wikimedia Foundation, Inc. [Online]. Available: http://en.wikipedia.org/wiki/Federation_(information_technology)

[11] A. R. Hevner, S. T. March, J. Park, and S. Ram, "Design Science in Information Systems Research," *MIS Quarterly*, vol. 28, no. 1, pp. 75–105, 2004.

[12] J. C. Henderson and H. Venkatraman, "Strategic alignment: Leveraging information technology for transforming organizations," *IBM Systems Journal*, vol. 32, no. 1, pp. 472 –484, 1993. [Online]. Available: http://dx.doi.org/10.1147/sj.382.0472

[13] S. Paul, J. Pan, and R. Jain, "Architectures for the future networks and the next generation internet: A survey," *Comput. Commun.*, vol. 34, pp. 2–42, January 2011. [Online]. Available: http://dx.doi.org/10.1016/j.comcom.2010.08.001

[14] J. Crowcroft, P. Demeester, J. Magen, P. Tran-Gia, and J. Wilander, "Towards a collaboration and highlevel federation structure for the fire facility," Working Group on modular federation of FIRE Facilities, July 2009. [Online]. Available: http://www.ict-fireworks.eu/fileadmin/documents/Wise-men_final.pdf

[15] D. Witaszek, F. Gouveia, S. Wahle, and T. Magedanz, "IMS Playground in Pan-European Network of Testbeds - Benefits and Challenges," in *3rd International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities, TRIDENTCOM 2007*. IEEE, May 2007, ISBN: 1-4244-0739-7. [Online]. Available: http://dx.doi.org/10.1109/TRIDENTCOM.2007.4444738

[16] S. Wahle, N. Blum, and T. Magedanz, "Evolution of the Open IMS Playground - Open Next Generation Network Testbeds in Face of Service Oriented Architectures, Web2.0 and European Testbed Federations," in *Mobilfunk - Technologien und Anwendungen, ITG-Fachbericht 208*. VDE VERLAG GmbH, May 2008, pp. 49 – 54, ISBN: 978-3-8007-3104-6, ISSN: 0932-6022. [Online]. Available: http://www.vde-verlag.de/proceedings-de/453104008.html

[17] T. Magedanz, F. Schreiner, and S. Wahle, "From NGN to Future Internet Testbed Management - Collaborative Testbeds as Enabler for Cross-Technology, Cross-Layer, and Cross-Domain Communication and Network Research," *Tele Kommunikation Aktuell*, vol. 62, no. 5-6, pp. 20–40, 2008, ISSN: 1619-2036.

[18] *Reference Model for Service Oriented Architecture 1.0*, Organization for the Advancement of Structured Information Standards (OASIS) Std. 1.0, October 2006. [Online]. Available: http://docs.oasis-open.org/soa-rm/v1.0/soa-rm.pdf

[19] *Service oriented architecture Modeling Language (SoaML) - Specification for the UML Profile and Metamodel for Services (UPMS)*, Object Management Group (OMG) Std. ptc/2009-12-09, Rev. Beta 2 document, 2009. [Online]. Available: http://www.omg.org/spec/SoaML/20091101

[20] F. Schreiner, S. Wahle, N. Blum, and T. Magedanz, "Modular Exposure of Next Generation Network Services to Enterprises and Testbed Federations," in *Second International Conference on Communications and Electronics (HUT-ICCE 2008)*. Hoi An, Vietnam: IEEE, June 2008, pp. 98 – 103, ISBN: 978-1-4244-2425-2. [Online]. Available: http://dx.doi.org/10.1109/CCE.2008.4578940

[21] N. Blum, T. Magedanz, F. Schreiner, and S. Wahle, "From IMS Management to SOA based NGN Management," *Journal of Network and Systems Management*, vol. 17, no. 1-2, pp. 33–52, June 2009, ISSN: 1064-7570 (Print) 1573-7705 (Online). [Online]. Available: http://dx.doi.org/10.1007/s10922-009-9118-4

[22] N. Blum *et al.*, "A Research Infrastructure for SOA-based Service Delivery Frameworks," in *5th International Conference on Testbeds and Research Infrastructures for the Development of Networks & Communities and Workshops (TridentCom 2009)*. Washington DC, USA: IEEE, April 2009, ISBN: 978-1-4244-2846-5. [Online]. Available: http://dx.doi.org/10.1109/TRIDENTCOM.2009.4976202

[23] N. Blum, T. Magedanz, F. Schreiner, and S. Wahle, "Service Oriented Testbed Infrastructures: a Cross-Layer Approach for NGNs," *ACM/Springer Mobile Networks and Applications (MONET), Special Issue: Advances In Wireless Test beds and Research Infrastructures*, vol. 15, pp. 413–424, 2010, ISSN: 1383-469X (Print) 1572-8153 (Online). [Online]. Available: http://dx.doi.org/10.1007/s11036-009-0201-6

[24] T. Magedanz, F. Schreiner, and S. Wahle, "Service-Oriented Testbed Infrastructures and Cross-Domain Federation for Future Internet Research," in *Integrated Network Management-Workshops, 2009. IM '09. IFIP/IEEE International Symposium on*. New York, USA: IEEE, June 2009, pp. 101–106, ISBN: 978-1-4244-3923-2. [Online]. Available: http://dx.doi.org/10.1109/INMW.2009.5195944

[25] "Architecture for Managing Clouds," DMTF, June 2010, a White Paper from the Open Cloud Standards Incubator, Version: 1.0.0. [Online]. Available: http://dmtf.org/sites/default/files/standards/documents/DSP-IS0102_1.0.0.pdf

[26] "Interoperable Clouds," DMTF, November 2009, a White Paper from the Open Cloud Standards Incubator, Version: 1.0.0. [Online]. Available: http://www.dmtf.org/sites/default/files/standards/documents/DSP-IS0101_1.0.0.pdf

[27] P. Mell and T. Grance, "The NIST Definition of Cloud Computing," National Institute of Standards and Technology, Computer Security Division, January 2011, Special Publication 800-145 (Draft). [Online]. Available: http://csrc.nist.gov/publications/drafts/800-145/Draft-SP-800-145_cloud-definition.pdf

[28] X. Fan, W.-D. Weber, and L. A. Barroso, "Power provisioning for a warehouse-sized computer," in *Proceedings of the 34th annual international symposium on Computer architecture*, ser. ISCA '07. New York, NY, USA: ACM, 2007, pp. 13–23. [Online]. Available: http://doi.acm.org/10.1145/1250662.1250665

[29] (2011, February) The Amazon S3 website. [Online]. Available: http://aws.amazon.com/s3/

[30] (2011, June) The Amazon EC2 website. [Online]. Available: http://aws.amazon.com/ec2/

[31] "The Future of Cloud Computing," Commission of the European Communities, Information Society & Media Directorate-General, Software & Service Architectures, Infrastructures and Engineering Unit, Tech. Rep. Public Version 1.0, 2010. [Online]. Available: http://cordis.europa.eu/fp7/ict/ssai/docs/cloud-report-final.pdf

[32] F. Gouveia, S. Wahle, N. Blum, and T. Magedanz, "Cloud Computing and EPC / IMS Integration: New Value-added Services on Demand," in *Mobimedia '09: Proceedings of the 5th International ICST Mobile Multimedia Communications*

Conference. ICST/ACM, September 2009, pp. 1–5, ISBN: 978-963-9799-62-2. [Online]. Available: http://portal.acm.org/citation.cfm?id=1653604&dl=ACM&coll=DL&CFID=8258013&CFTOKEN=80920174

[33] (2011, March) The Open Grid Forum website. [Online]. Available: http://www.ogf.org/

[34] (2011, March) The Open Cloud Computing Interface Working Group website. [Online]. Available: http://occi-wg.org/

[35] R. Nyrén, A. Edmonds, A. Papaspyrou, and T. Metsch, *Open Cloud Computing Interface - Core*, Open Grid Forum (OGF) Std. GFD-P-R.183, April 2011. [Online]. Available: http://occi-wg.org/about/specification

[36] (2011, March) The Storage Networking Industry Association website. [Online]. Available: http://www.snia.org/

[37] *Cloud Data Management Interface*, Storage Networking Industry Association Std. 1.0, April 2010. [Online]. Available: http://www.snia.org/tech_activities/standards/curr_standards/cdmi/CDMI_SNIA_Architecture_v1.0.pdf

[38] (2011, March) The Cloud Storage Initiative website. [Online]. Available: http://www.snia.org/forums/csi

[39] *Cloud Storage Reference Model*, Storage Networking Industry Association Std. 0.3, Rev. 0, June 2009. [Online]. Available: http://www.snia.org/tech_activities/publicreview/CloudStorageReferenceModelV03.pdf

[40] *Cloud Storage Use Cases*, Storage Networking Industry Association Std. 0.5, Rev. 0, June 2009. [Online]. Available: http://www.snia.org/tech_activities/publicreview/CloudStorageUseCasesv0.5.pdf

[41] (2011, March) The Object Management Group website. [Online]. Available: http://omg.org/

[42] (2011, March) The Cloud Services Specification Project website. [Online]. Available: http://sites.google.com/site/omgcssp/

[43] R. W. Thrash, "Building A Cloud Computing Specification: Fundamental Engineering for Optimizing Cloud Computing Initiatives," April 2009. [Online]. Available: http://sites.google.com/site/omgcssp/file-cabinet/BuildingaCloudComputingSpecification090519_1402_ver3.0.pdf

[44] (2011, March) The Organization for the Advancement of Structured Information Standards (OASIS) website. [Online]. Available: http://www.oasis-open.org

[45] (2011, March) The OASIS IDCloud Technical Commitee website. [Online]. Available: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=id-cloud

[46] *Identity in the Cloud - Use Cases*, Organization for the Advancement of Structured Information Standards (OASIS) Committee Draft Draft Version 0.1n, January 2011. [Online]. Available: http://tools.oasis-open.org/version-control/browse/wsvn/id-cloud/doc/committee/interim/

[47] (2011, March) The Distributed Management Task Force website. [Online]. Available: http://www.dmtf.org/

[48] (2011, March) The Distributed Management Task Force Cloud Management Working Group website. [Online]. Available: http://www.dmtf.org/standards/cloud

[49] (2011, March) The Open Cloud Consortium website. [Online]. Available: http://opencloudconsortium.org/home/

[50] (2011, March) Malgen project website. [Online]. Available: http://code.google.com/p/malgen/

[51] (2011, March) Thriftstore project website. [Online]. Available: http://code.google.com/p/thriftstore/

[52] (2011, March) The Open Group website. [Online]. Available: https://www.opengroup.org/

[53] M. Behrendt *et al.*, "Introduction and Architecture Overview IBM Cloud Computing Reference Architecture 2.0," February 2011, Revision Number 1.0. [Online]. Available: https://www.opengroup.org/cloudcomputing/uploads/40/23840/CCRA.IBMSubmission.02282011.doc

[54] (2011, March) The Cloud Computing Interoperability Forum website. [Online]. Available: http://www.cloudforum.org/

[55] (2011, March) The TeleManagement Forum website. [Online]. Available: http://www.tmforum.org

[56] (2011, March) The Cloud Security Alliance website. [Online]. Available: http://www.cloudsecurityalliance.org/

[57] "Domain 12: Guidance for Identity & Access Management V2.1," Cloud Security Alliance whitepaper, April 2010. [Online]. Available: https://cloudsecurityalliance.org/guidance/csaguide-dom12-v2.10.pdf

[58] "Top Threats to Cloud Computing," Cloud Security Alliance, Tech. Rep. 1.0, March 2010. [Online]. Available: http://www.cloudsecurityalliance.org/topthreats/csathreats.v1.0.pdf

[59] (2011, March) The European Telecommunications Standards Institute website. [Online]. Available: http://www.etsi.org

[60] (2011, March) The International Telecommunication Union Telecommunication Standardization Sector website. [Online]. Available: http://www.itu.int/net/ITU-T

[61] I. Foster and C. Kesselman, "Globus: A Metacomputing Infrastructure Toolkit," *International Journal of Supercomputer Applications*, vol. 11, pp. 115–128, 1996.

[62] "Open Grid Services Architecture Glossary of Terms," Open Grid Forum, December 2007, Version 1.6. [Online]. Available: http://www.ogf.org/documents/GFD.120.pdf

[63] I. Foster *et al.*, "Open Grid Services Architecture," Open Grid Forum, Tech. Rep., July 2006, Version 1.5. [Online]. Available: http://www.ogf.org/documents/GFD.80.pdf

[64] S. Graham *et al.*, *Web Services Resource 1.2*, Organization for the Advancement of Structured Information Standards (OASIS) Std. 1.2, April 2006. [Online]. Available: http://docs.oasis-open.org/wsrf/wsrf-ws_resource-1.2-spec-os.pdf

[65] J. Alexander *et al.*, *Web Services Transfer (WS-Transfer)*, W3C Member Submission, World Wide Web Consortium (W3C) Std., September 2006. [Online]. Available: http://www.w3.org/Submission/WS-Transfer/

[66] "Study of ICT Grid interoperability gaps; Part 1: Inventory of ICT Stakeholders," European Telecommunications Standards Institute, Tech. Rep. ETSI TR 102 659-1 V1.2.1, October 2009. [Online]. Available: http://www.etsi.org/deliver/etsi_tr/102600_102699/10265901/01.02.01_60/tr_10265901v010201p.pdf

[67] "Liberty Alliance Contractual Framework Outline for Circles of Trust," whitepaper, Liberty Alliance. [Online]. Available: http://projectliberty.org/liberty/files/whitepapers/liberty_alliance_contractual_framework_outline_for_circles_of_trust

[68] (2011, March) Merriam Webster dictionary entry for "identity". Encyclopedia Britannica. [Online]. Available: http://www.merriam-webster.com/dictionary/identity

[69] (2011, March) Merriam Webster dictionary entry for "persona". Encyclopedia Britannica. [Online]. Available: http://www.merriam-webster.com/dictionary/persona

[70] Z. P. Jin, J. Xu, M. Xu, and N. Zheng, "An Attribute-Oriented Model for Identity Management," in *Proc. Int. Conf. e-Education, e-Business, e-Management, and e-Learning IC4E '10*, January 2010, pp. 440–444. [Online]. Available: http://dx.doi.org/10.1109/IC4E.2010.55

[71] P. Weik and S. Wahle, "Towards a Generic Identity Enabler for Telco Networks," in *Proceedings of the 12th International Conference on Intelligence in Service Delivery Networks (ICIN)*, Bordeaux, France, October 2008, ISBN: 978-1-60702-868-0. [Online]. Available: http://www.icin.biz/files/2008papers/Session5A-2.pdf

[72] H. Lockhart *et al.*, *Web Services Federation Language (WS-Federation)*, Std., December 2006. [Online]. Available: http://public.dhe.ibm.com/software/dw/specs/ws-fed/WS-Federation-V1-1B.pdf

[73] (2011, March) The Future Internet Design website. [Online]. Available: http://www.nets-find.net

[74] (2011, March) The Network of the Future. European Commission. [Online]. Available: http://cordis.europa.eu/fp7/ict/future-networks

[75] S. Fischer and P. Müller, "Experimentalforschung für das Future Internet – deutsche und europäische Initiativen," *Informatik-Spektrum*, vol. 33, no. 2, pp. 122–130, 2010, ISSN: 0170-6012. [Online]. Available: http://dx.doi.org/10.1007/s00287-010-0419-5

[76] (2011, March) The Global Environment for Network Innovations (GENI) website. National Science Foundation. [Online]. Available: http://www.geni.net

[77] "GENI System Overview," GENI Project Office, 10 Moulton Street Cambridge, MA 02138 USA, Tech. Rep. 2.0, September 2008. [Online]. Available: http://groups.geni.net/geni/wiki/GeniSysOvrvw

[78] "GENI System Requirements Document," GENI Project Office, 10 Moulton Street Cambridge, MA 02138 USA, Tech. Rep. 2.0, July 2009. [Online]. Available: http://groups.geni.net/geni/wiki/SysReqDoc

[79] T. Magedanz and S. Wahle, "Control Framework Design for Future Internet Testbeds," *e & i Elektrotechnik und Informationstechnik*, vol. 126, no. 07/08, pp. 274–279, July 2009, ISSN: 0932-383X (print) ISSN: 1613-7620 (online). [Online]. Available: http://dx.doi.org/10.1007/s00502-009-0655-z

[80] T. Faber and J. Wroclawski, "A Federated Experiment Environment for Emulab-based Testbeds," in *Testbeds and Research Infrastructures for the Development of Networks Communities and Workshops, 2009. TridentCom 2009. 5th International Conference on*, April 2009, pp. 1 –10. [Online]. Available: http://dx.doi.org/10.1109/TRIDENTCOM.2009.4976238

[81] J. Chase, L. Grit, D. Irwin, V. Marupadi, P. Shivam, and A. Yumerefendi, "Beyond Virtual Data Centers: Toward an Open Resource Control Architecture," in *Proccedings of the International Conference on the Virtual Computing Initiative (ICVCI 2007)*, May 2007. [Online]. Available: http://www.cs.duke.edu/nicl/pub/papers/vci07.pdf

[82] M. Ott, I. Seskar, R. Siraccusa, and M. Singh, "ORBIT Testbed Software Architecture: Supporting Experiments as a Service," in *Proc. First Int. Conf. Testbeds and Research Infrastructures for the Development of Networks and Communities Tridentcom 2005*, 2005, pp. 136–145. [Online]. Available: http://dx.doi.org/10.1109/TRIDNT.2005.27

[83] T. Rakotoarivelo, M. Ott, G. Jourjon, and I. Seskar, "OMF: A Control and Management Framework for Networking Testbeds," *SIGOPS Oper. Syst. Rev.*, vol. 43, pp. 54–59, January 2010. [Online]. Available: http://doi.acm.org/10.1145/1713254.1713267

[84] L. Peterson, R. Ricci, A. Falk, and J. Chase, "Slice-Based Federation Architecture," Tech. Rep. 2.0, July 2010. [Online]. Available: http://groups.geni.net/geni/wiki/SliceFedArch

[85] "Sprial 2 Overview," GENI Project Office, 10 Moulton Street Cambridge, MA 02138 USA, Tech. Rep., June 2010. [Online]. Available: http://groups.geni.net/geni/attachment/wiki/SpiralTwo/GENIS2Ovrvw060310.pdf

[86] (2011, March) The PlanetLab website. [Online]. Available: http://www.planet-lab.org/

[87] L. L. Peterson and T. Roscoe, "The Design Principles of PlanetLab," *Operating Systems Review*, vol. 40, no. 1, pp. 11–16, 2006. [Online]. Available: http://doi.acm.org/10.1145/1113361.1113367

[88] K. Campowsky, T. Magedanz, and S. Wahle, "Resource Management in Large Scale Experimental Facilities - Technical Approach to Federate Panlab and PlanetLab," in *12th IEEE Network Operations and Management Symposium (NOMS 2010)*. Osaka, Japan: IEEE, April 2010, pp. 930 – 933, ISSN: 1542-1201. [Online]. Available: http://dx.doi.org/10.1109/NOMS.2010.5488336

[89] L. Peterson *et al.*, "Slice-Based Facility Architecture," Tech. Rep. 1.01, August 2008. [Online]. Available: http://groups.geni.net/geni/attachment/wiki/GeniControlBr

[90] J. Chase, "ORCA Control Framework Architecture and Internals," Duke University Computer, Science Department, Tech. Rep., November 2009. [Online]. Available: https://geni-orca.renci.org/trac/attachment/wiki/WikiStart/

[91] Y. Fu, J. Chase, B. Chun, S. Schwab, and A. Vahdat, "SHARP: An Architecture for Secure Resource Peering," in *Proceedings of the nineteenth ACM symposium on Operating systems principles*, ser. SOSP '03. New York, NY, USA: ACM, 2003, pp. 133–148. [Online]. Available: http://doi.acm.org/10.1145/945445.945459

[92] D. Irwin, J. Chase, L. Grit, A. Yumerefendi, and D. Becker, "Sharing Networked Resources with Brokered Leases," in *Proceedings of the USENIX Technical Conference*, 2006, pp. 199–212. [Online]. Available: http://www.usenix.org/events/usenix06/tech/full_papers/irwin/irwin.pdf

[93] L. Ramakrishnan, D. Irwin, L. Grit, A. Yumerefendi, A. Iamnitchi, and J. Chase, "Toward a Doctrine of Containment: Grid Hosting with Adaptive Resource Control," in *Proceedings of the 2006 ACM/IEEE conference on Supercomputing*, ser. SC '06. New York, NY, USA: ACM, 2006. [Online]. Available: http://doi.acm.org/10.1145/1188455.1188561

[94] J. S. Chase, D. E. Irwin, L. E. Grit, J. D. Moore, and S. E. Sprenkle, "Dynamic Virtual Clusters in a Grid Site Manager," in *Proceedings of the 12th IEEE International Symposium on High Performance Distributed Computing*, ser. HPDC '03. Washington, DC, USA: IEEE Computer Society, 2003, pp. 90–. [Online]. Available: http://dx.doi.org/10.1109/HPDC.2003.1210019

[95] (2010, March) The Future Internet Research and Experimentation (FIRE) website. European Commission. [Online]. Available: http://cordis.europa.eu/fp7/ict/fire

[96] (2011, March) The Onelab2 project website. FP7 Onelab2. [Online]. Available: http://www.onelab.eu

[97] (2011, March) The Panlab SSA and Pan-European laboratory infrastructure implementation (PII) projects website. FP6 Panlab and FP7 PII. [Online]. Available: http://www.panlab.net

[98] S. Wahle, T. Magedanz, A. Gavras, H. Hrasnica, and S. Denazis, "Technical Infrastructure for a Pan-European Federation of Testbeds," in *Testbeds and Research Infrastructures for the Development of Networks & Communities and Workshops, 2009. TridentCom 2009. 5th International Conference on.* Washington DC, USA: IEEE, April 2009, pp. 1–8, ISBN: 978-1-4244-2846-5. [Online]. Available: http://dx.doi.org/10.1109/TRIDENTCOM.2009.4976205

[99] A. Gavras, H. Hrasnica, S. Wahle, D. Lozano, D. Mischler, and S. Denazis, *Towards the Future Internet - A European Research Perspective.* IOS Press, May 2009, ch. Control of Resources in Pan-European Testbed Federation, pp. 67 – 78, ISBN: 978-1-60750-007-0. [Online]. Available: http://dx.doi.org/doi:10.3233/978-1-60750-007-0-67

[100] S. Wahle, A. Gavras, F. Gouveia, H. Hrasnica, and T. Magedanz, "Network Domain Federation - Infrastructure for Federated Testbeds," in *2008 NEM Summit - Towards Future Media Internet*, Saint-Malo, France, October 2008, pp. 179 – 184, ISBN: 978-3-00-025978-4.

[101] S. Wahle, B. Harjoc, K. Campowsky, T. Magedanz, and A. Gavras, "Pan-European testbed and experimental facility federation - architecture refinement and implementation," *International Journal of Communication Networks and Distributed Systems (IJCNDS), Special Issue on Recent Advances in Testbed Driven Networking Research*, vol. 5, no. 1/2, pp. 67–87, June 2010, ISSN: 1754-3916. [Online]. Available: http://dx.doi.org/10.1504/IJCNDS.2010.033968

[102] A. Köpsel *et al.*, "Monitoring requirements and procedures for service level agreement compliance," Pan-European laboratory infrastructure implementation (PII) project consortium, project deliverable version 1.0, 2010. [Online]. Available: http://www.panlab.net/fileadmin/documents/PII-Deliverables/D4.2-Monitoring_requirements_and_procedures_for_SLA_compliance_v1.0.pdf

[103] H. Hrasnica *et al.*, "Iteration on legal requirements with stakeholders in the clusters," Pan-European laboratory infrastructure implementation (PII) project consortium, project deliverable version 1.1, 2009. [Online]. Available: http://www.panlab.net/fileadmin/documents/PII-Deliverables/D1.3-Iteration_on_legal_requirements_with_stakeholders_in_the_clusters_v1.1.pdf

[104] (2011, March) The Wisebed project website. The Wisebed project consortium. [Online]. Available: http://www.wisebed.eu

[105] "iWSN Federator Implementation," The Wisebed project consortium, Tech. Rep. 1.0, May 2010. [Online]. Available: http://www.wisebed.eu/images/stories/deliverables/TR/TR-2010-iWSN-Federator-v1.pdf

[106] "iWSN API 2.1.1," The Wisebed project consortium, Tech. Rep. 2.1.1, May 2010. [Online]. Available: http://www.wisebed.eu/images/stories/deliverables/TR/TR-iWSN-API-V2.1.1.pdf

[107] (2011, March) The G-Lab initiative website. The G-Lab project consortium. [Online]. Available: http://www.german-lab.de

[108] P. Müller and B. Reuther, "Future Internet Architecture – A Service Oriented Approach," *it - Information Technology*, vol. 50, no. 6, pp. 383–389, 2008. [Online]. Available: http://dx.doi.org/10.1524/itit.2008.0510

[109] D. Schwerdel, D. Günther, M. R. Khondoker, B. Reuther, and P. Müller, "A Building Block Interaction Model for Flexible Future Internet Architectures," in *7th EURO-NF Conference on Next Generation Internet*, Kaiserslautern, Germany, June 2011, pp. 1–8. [Online]. Available: http://hdl.handle.net/123456789/304

[110] D. Schwerdel, D. Günther, R. Henjes, B. Reuther, and P. Müller, "German-lab experimental facility," in *Future Internet - FIS 2010*, ser. Lecture Notes in Computer Science, A. Berre, A. Gómez-Pérez, K. Tutschku, and D. Fensel,

Eds. Springer Berlin / Heidelberg, 2010, vol. 6369, pp. 1–10. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-15877-3_1

[111] D. Schwerdel *et al.*, "Tomato - a network experimentation tool," in *Proceedings of TRIDENTCOM 2011 - 7th International ICST Conference on Testbeds and Research Infrastructures for the Development of Networks & Communities*, Shanghai, Republic of China, April 2011, pp. 1–10.

[112] S. Wahle, T. Magedanz, and A. Gavras, *Towards the Future Internet - Emerging Trends from European Research*. IOS Press, April 2010, ch. Conceptual Design and Use Cases for a FIRE Resource Federation Framework, pp. 51–62, ISBN: 978-1-60750-538-9 (print), 978-1-60750-539-6 (online). [Online]. Available: http://dx.doi.org/10.3233/978-1-60750-539-6-51

[113] Cooper *et al.*, *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*, Internet Engineering Task Force (IETF) Std. RFC 5280, May 2008. [Online]. Available: http://www.ietf.org/rfc/rfc5280.txt

[114] (2011, April) Merriam Webster dictionary entry for "recursion". Encyclopedia Britannica. [Online]. Available: http://www.merriam-webster.com/dictionary/recursion

[115] S. Wahle, A. Steinbach, T. Magedanz, and K. Campowsky, "Dynamic Virtual Overlay Networks for Large Scale Resource Federation Frameworks," in *Proceedings of TRIDENTCOM 2011 - 7th International ICST Conference on Testbeds and Research Infrastructures for the Development of Networks & Communities*, Shanghai, Republic of China, April 2011, pp. 1–14.

[116] S. Wahle, T. Magedanz, S. Fox, and E. Power, "Heterogeous Resource Description and Management in Generic Resource Federation Frameworks," in *Proceedings of the 12th IFIP/IEEE International Symposium on Integrated Network Management (IM 2011), Workshop on Managing Federations and Cooperative Management*, Dublin, Ireland, May 2011, pp. 1196–1199, ISBN: 978-1-4244-9219-0. [Online]. Available: http://dx.doi.org/10.1109/INM.2011.5990582

[117] *Vocabulary for 3GPP Specifications*, 3rd Generation Partnership Project (3GPP) Std. 3GPP TR 21.905 V10.3.0, March 2011. [Online]. Available: http://www.3gpp.org/ftp/Specs/html-info/21905.htm

[118] S. Shepler, M. Eisler, and D. Noveck, *Network File System (NFS) Version 4 Minor Version 1 Protocol*, Internet Engineering Task Force (IETF) Std. RFC 5661, January 2010. [Online]. Available: http://www.ietf.org/rfc/rfc5661.txt

[119] A. Alves *et al.*, *Web Services Business Process Execution Language Version 2.0*, Organization for the Advancement of Structured Information Standards (OASIS) Std., April 2007. [Online]. Available: http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.pdf

[120] L. Andersson and T. Madsen, *Provider Provisioned Virtual Private Network (VPN) Terminology*, Internet Engineering Task Force (IETF) Std. RFC 4026, March 2005. [Online]. Available: http://www.ietf.org/rfc/rfc4026.txt

[121] *IEEE Standard for Local and metropolitan area networks: Virtual Bridged Local Area Networks*, Institute of Electrical and Electronics Engineers (IEEE) Computer Society Std. 802.1Q, May 2006. [Online]. Available: http://standards.ieee.org/getieee802/download/802.1Q-2005.pdf

[122] *Information Technology - Open Systems Interconnection - Basic Reference Model: The Basic Model*, International Organization for Standardization Std. ISO/IEC 7498-1:1994(E), Rev. 2nd edition, June 1996.

[123] W. Townsley *et al.*, *Layer Two Tunneling Protocol "L2TP"*, Internet Engineering Task Force (IETF) Std. RFC 2661, August 1999. [Online]. Available: http://www.ietf.org/rfc/rfc2661.txt

[124] *User Requirements Notation (URN) - Language requirements and framework*, Telecommunication Standardization Sector of ITU (ITU-T) Std. ITU-T Rec. Z.150, February 2003.

[125] *User requirements notation (URN) - Language definition*, Telecommunication Standardization Sector of ITU (ITU-T) Std. Rec. ITU-T Z.151, November 2008.

[126] M. Gudgin *et al.*, *SOAP Version 1.2 Part 1: Messaging Framework (Second Edition)*, Recommendation, World Wide Web Consortium (W3C) Std., April 2007. [Online]. Available: http://www.w3.org/TR/soap12-part1/

[127] T. Dierks and E. Rescorla, *The Transport Layer Security (TLS) Protocol Version 1.2*, Internet Engineering Task Force (IETF) Std. RFC 5246, August 2008. [Online]. Available: http://www.ietf.org/rfc/rfc5246.txt

[128] G. Cole *et al.*, *OASIS Service Provisioning Markup Language (SPML) Version 2*, Organization for the Advancement of Structured Information Standards (OASIS) Std., April 2006. [Online]. Available: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=provision

[129] *Java™ Network Launching Protocol & API Specification (JSR-56)*, Sun Microsystems, Inc. Std. version 6.0.18, January 2010. [Online]. Available: http://jcp.org/en/jsr/detail?id=56

[130] P. Delisle, J. Luehe, M. Roth, and K. Chung, *JavaServer Pages™ Specification*, Sun Microsystems, Inc. Std. version 2.2, Rev. maintenance release 2, December 2009. [Online]. Available: http://jcp.org/aboutJava/communityprocess/mrel/jsr245/index.html

[131] J. Strassner, *Policy-Based Network Management: Solutions for the Next Generation (The Morgan Kaufmann Series in Networking)*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2003.

[132] A. Berglund *et al.*, *XML Path Language (XPath) 2.0*, World Wide Web Consortium (W3C) Recommendation, January 2007. [Online]. Available: http://www.w3.org/TR/2007/REC-xpath20-20070123/

[133] A. Laux and L. Martin, *XUpdate Working Draft*, Std., September 2000. [Online]. Available: http://xmldb-org.sourceforge.net/xupdate/xupdate-wd.html

[134] J. Robie *et al.*, *XQuery Update Facility 1.0*, World Wide Web Consortium (W3C) Recommendation, March 2011. [Online]. Available: http://www.w3.org/TR/xquery-update-10/

[135] R. Fielding *et al.*, *Hypertext Transfer Protocol – HTTP/1.1*, Internet Engineering Task Force (IETF) Std. RFC 2616, June 1999. [Online]. Available: http://www.ietf.org/rfc/rfc2616.txt

[136] J. Franks *et al.*, *HTTP Authentication: Basic and Digest Access Authentication*, Internet Engineering Task Force (IETF) Std. RFC 2617, June 1999. [Online]. Available: http://www.ietf.org/rfc/rfc2617.txt

[137] *Policy Evaluation, Enforcement and Management Management Interface (PEM-2) Technical Specification*, Open Mobile Alliance (OMA) Std. OMA-TS-PEEM_PEM2-V1_0-20 081 014-C, Rev. Candidate Version 1.0, October 2008. [Online]. Available: http://www.openmobilealliance.org/Technical/Release_program/peem_v1_0.aspx

[138] J. Rosenberg, *The Extensible Markup Language (XML) Configuration Access Protocol (XCAP)*, Internet Engineering Task Force (IETF) Std. RFC 4825, May 2007. [Online]. Available: http://www.ietf.org/rfc/rfc4825.txt

[139] S. Wahle *et al.*, "Implementation Report," Pan-European laboratory infrastructure implementation (PII) project, deliverable D3.7, April 2011, version 2.0. [Online]. Available: http://www.panlab.net/fileadmin/documents/PII-Deliverables/D3.7-Implementation_Report_v2.0.pdf

[140] *PEEM Policy Expression Language Technical Specification*, Open Mobile Alliance (OMA) Std. OMA-TS-PEEM_PEL-V1_0-20 080 805-C, Rev. Candidate Version 1.0, August 2008. [Online]. Available: http://www.openmobilealliance.org/Technical/Release_program/peem_v1_0.aspx

[141] H. Schulzrinne *et al.*, *Common Policy: A Document Format for Expressing Privacy Preferences*, Internet Engineering Task Force (IETF) Std. RFC 4745, February 2007. [Online]. Available: http://www.ietf.org/rfc/rfc4745.txt

[142] *Policy Evaluation, Enforcement and Management Callable Interface (PEM-1) Technical Specification*, Open Mobile Alliance (OMA) Std. OMA-TS-PEEM_PEM1-V1_0-20 090 715-C, Rev. Candidate Version 1.0, July 2009. [Online]. Available: http://www.openmobilealliance.org/Technical/Release_program/peem_v1_0.aspx

[143] R. Mordani, *Java^{TM} Servlet Specification*, Sun Microsystems, Inc. Std. Version 3.0, Rev. a, February 2011. [Online]. Available: http://www.jcp.org/en/jsr/detail?id=315

[144] M. Belaunde *et al.*, "Testbed Orchestration System Specification," Pan-European laboratory infrastructure implementation (PII) project consortium, project deliverable version 1.0, March 2009.

[145] M. Belaunde and P. Falcarin, "Realizing an MDA and SOA Marriage for the Development of Mobile Services," in *Model Driven Architecture - Foundations and Applications*, ser. Lecture Notes in Computer Science, I. Schieferdecker and A. Hartman, Eds. Springer Berlin / Heidelberg, 2008, vol. 5095, pp. 393–405. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-69100-6_28

[146] K. Campowsky, "Design and Implementation of a Generic Resource Management Framework for Network Domain Federations," Diploma Thesis, Technische Universität Berlin, October 2009.

[147] E. Gamma, R. Helm, R. E. Johnson, and J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*. Amsterdam: Addison-Wesley Longman, 1995.

[148] T. Berners-Lee, R. Fielding, and L. Masinter, *Uniform Resource Identifier (URI): Generic Syntax*, Internet Engineering Task Force (IETF) Std. RFC 3986, January 2005. [Online]. Available: http://www.ietf.org/rfc/rfc3986.txt

[149] S. Wahle, C. Tranoris, S. Fox, and T. Magedanz, "Resource Description in Large Scale Heterogeneous Resource Federations," in *Proceedings of TRIDENTCOM 2011 - 7th International ICST Conference on Testbeds and Research Infrastructures for the Development of Networks & Communities*, Shanghai, Republic of China, April 2011, pp. 1–16.

[150] *Meta Object Facility (MOF) Core Specification*, Object Management Group (OMG) Std., Rev. Version 2.4 Convenience, December 2010. [Online]. Available: http://www.omg.org/technology/documents/modeling_spec_catalog.htm

[151] (2011, June) Eclipse Modeling Framework Project (EMF) website. [Online]. Available: http://www.eclipse.org/modeling/emf/?project=emf

[152] (2011, June) Xpand project website. [Online]. Available: http://www.eclipse.org/modeling/m2t/?project=xpand

[153] C. Tranoris, "Adopting the DSM paradigm: defining federation scenarios through resource brokers for experimentally driven research," in *Proceedings of the 12th IFIP/IEEE International Symposium on Integrated Network Management (IM 2011), Workshop on Managing Federations and Cooperative Management*, Dublin, Ireland, May 2011, pp. 1136–1143, ISBN: 978-1-4244-9220-6.

[154] D. Harrington, R. Presuhn, and B. Wijnen, *An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks*, Internet Engineering Task Force (IETF) Std. RFC 3411, December 2002. [Online]. Available: http://www.ietf.org/rfc/rfc3411.txt

[155] J. Leguay *et al.*, "Global architecture and overall design," TEFIS project consortium, project deliverable D2.1.1 version 1.0, November 2010.

[156] I. Rosenberg *et al.*, "D2.2.1 - Architecture definition," BonFIRE project consortium, project deliverable version 1.0, September 2010.

[157] (2011, July) FP7 NOVI project website. [Online]. Available: http://www.fp7-novi.eu

[158] L. Lymberopoulos *et al.*, "Managing Federations of Virtualized Infrastructures: A Semantic-Aware Policy Based Approach," in *Proceedings of the third IEEE/IFIP International Workshop on Management of the Future Internet (ManFI 2011)*, Dublin, Ireland, May 2011, pp. 1231–1238, ISBN: 978-1-4244-9220-6.

[159] C. Henke, R. Wuttke, T. Zseby, and K. Campowsky, "On Creating Overlay Routing Topologies between Heterogeneous Experimental Facilities," in *Proceedings of TRIDENTCOM 2011 - 7th International ICST Conference on Testbeds and Research Infrastructures for the Development of Networks & Communities*, Shanghai, Republic of China, April 2011, pp. 1–10.

[160] S. Harder, "Design and Implementation of a Federation Interoperability Framework," yet unpublished master thesis, to appear in 2011.

[161] S. Fdida *et al.*, "OpenLab: extending FIRE testbeds and tools," 2011, large-scale integrating project (IP) proposal, ICT Call 7, unpublished description of work, version 3.5.

[162] M. P. Fernandez, S. Wahle, and T. Magedanz, "Teagle-OF: Management Plane for an OpenFlow Network," in *13th IEEE/IFIP Network Operations and Management Symposium (NOMS 2012)*, Hawaii, USA, April 2012, currently under review.

[163] S. Wahle, T. Magedanz, and K. Campowsky, *Testbeds and Research Infrastructures*, ser. Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering. Springer Berlin Heidelberg NewYork, January 2011, vol. 46, ch. Interoperability in Heterogeneous Resource Federations, pp. 35–51, ISBN: 978-3-642-17850-4. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-17851-1

[164] S. Wahle, "Federation Interoperability - Dealing With Heterogeneity," NSF/FIRE Workshop on Federating Computing Resources, Princeton University, Princeton, NJ, USA, May 2010, position paper. [Online]. Available: http://svn.planet-lab.org/attachment/wiki/FederationWorkshop/PapersM-Z.tgz

[165] S. Wahle, T. Magedanz, and K. Campowsky, "Federation Interoperability - Dealing With Heterogeneity," 5. Fachgespräch der GI/ITG-Fachgruppe 'Kommunikation und Verteilte Systeme' (KuVS), Stuttgart, Germany, June 2010. [Online]. Available: http://www.future-internet.org/archiv_2010_KuVS5.shtml

[166] S. Wahle and K. Campowsky, "Federation of Future Internet Research & Experimentation (FIRE) Facilities - "Teagle" as a Tool for Generic Resource Federation," Berlin, Germany, September 2010, half day tutorial at FIS 2010. [Online]. Available: http://www.fis2010.org/program/tutorials-menu

[167] Wahle and Campowsky, "Getting started with Teagle - A FIRE testbed federation tool," Berlin, Germany, May 2010, half day tutorial at Tridentcom 2010. [Online]. Available: http://www.tridentcom.org/tridentcom2010/tutorials/index.php

[168] T. Magedanz, S. Wahle, and K. Campowsky, "Heterogeneous Resource Federation - The "Teagle" Framework as a Means to Federate Future Internet Research & Experimentation Facilities," Shanghai, Republic of China, April 2011, half day tutorial at Tridentcom 2011. [Online]. Available: http://www.tridentcom.org/programtutorial.shtml

[169] T. Zseby *et al.*, "Multipath Routing Slice Experiments in Federated Testbeds," in *The Future Internet*, ser. Lecture Notes in Computer Science, J. Domingue *et al.*,

Eds. Springer Berlin / Heidelberg, 2011, vol. 6656, pp. 247–258. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-20898-0_18

[170] T. Zinner, D. Klein, K. Tutschku, T. Zseby, P. Tran-Gia, and Y. Shavitt, "Performance of Concurrent Multipath Transmissions – Measurements and Model Validation," in *Proceedings of the 7th EURO-NGI conference on Next Generation Internet (NGI)*, June 2011, pp. 1 –8. [Online]. Available: http://dx.doi.org/10.1109/NGI.2011.5985862

[171] *Syntactic metalanguage - Extended BNF*, International Organization for Standardization Std. ISO/IEC 14 977:1996(E), December 1996.

[172] *XML Schema Part 1: Structures Second Edition*, Recommendation, World Wide Web Consortium (W3C) Std., October 2004. [Online]. Available: http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/structures.html

[173] *XML Schema Part 2: Datatypes Second Edition*, Recommendation, World Wide Web Consortium (W3C) Std., October 2004. [Online]. Available: http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/datatypes.html

[174] *The Chicago Manual of Style*, 15th ed. The University of Chicago Press, 2003, ISBN: 0226104036.