Semantic-Based Management of Federated Infrastructures for Future Internet Experimentation

vorgelegt von
Alexander Willner, M.Sc.
geb. in Mülheim a.d. Ruhr

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. Rafael Schaefer (Technische Universität Berlin)
Gutachter:	Prof. Dr.-Ing. Thomas Magedanz (Technische Universität Berlin)
Gutachter:	Prof. Dr. Serge Fdida (University Pierre et Marie Curie)
Gutachter:	Prof. dr. ir. Piet Demeester (Ghent University)

Tag der wissenschaftlichen Aussprache: 11. Februar 2016

Berlin 2016

Technische Universität Berlin

Acknowledgments

The work for this thesis was conducted mainly during my time as a research assistant at the chair for Next Generation Networks at the Technische Universität Berlin. Therefore, first and foremost, I would like to thank Prof. Dr.-Ing. Thomas Magedanz for his support. Without him and the team at the Fraunhofer Institute for Open Communication Systems (FOKUS) and at the university, this work would have not been possible. I would also like to thank my second assessors Prof. Dr. Serge Fdida and Prof. dr. ir. Piet Demeester for the possibility to write this thesis and their guidance.

Next, I would like to express special thanks to Florian Schreiner, Daniel Nehls, Yahya Al-Hazmi, Björn Riemer, Alaa Alloush, Mitja Nikolaus, Alexander Ortlieb and Robyn Loughnane for their support. Moreover, I highly appreciate the time with my former colleagues at the University of Bonn, in which I learned many valuable lessons required to finish my research. Further, although rather uncommon, I would like to thank the Berliner Verkehrsbetriebe (BVG), which allowed me to have regular uninterrupted periods of time while commuting to focus on writing this thesis.

Finally, and of great importance, I thank my beloved family and friends for their unconditional emotional support in the long process of finishing this chapter of my life. In particular, I thank my wife for her continuing patience, her love and for going on this journey with me; my kids, for bringing so much joy to my life and giving me the best reason to finish this thesis; and my parents, for making all this possible in the first place.

Berlin, February 12, 2016

ABSTRACT

Sharing and granting access to geographically dispersed resources is the underlying concept in the field of Distributed Computing. Associated with this, considerable efforts have been spent on architectures to manage heterogeneous resources across multiple administrative domains. Such an environment is referred to as a resource federation.

Research Area: Distributed Computing

A specific field of application is the execution of experiments in the context of Future Internet research. Given the scale, complexity and heterogeneity of the Internet, experimental validation is carried out within large-scale, distributed test environments. As a result, several independent testbeds have been established, with accompanying protocols to support the experiment life cycle across sites.

Application Area: Distributed Experimentation

An important challenge that arises in this context is the exchange of information about the provided resources with their types and characteristics. Existing work rests upon certain interfaces and syntactic data models with arbitrary extensions and identifiers, which aggravate the management of heterogeneous resources across autonomous testbeds.

Research Issue: Resource Description

This thesis introduces an approach that is founded on well-defined semantic information models and architectural abstraction to address these issues. Based on mechanisms that have their origins in Semantic Web research, the use of semantically annotated graphs allows for automatic reasoning, linking, querying and validation of heterogeneous data.

Own Approach: Semantics

The main contributions of the research conducted for this thesis are the definition of an ontology for the life cycle management of resources in federated infrastructures, a corresponding semantic- and microservice-based architecture for interface abstraction, and a proof-of-concept implementation.

Scientific Contributions

The work has been validated within several European Future Internet research projects and testbeds. Further, a performance evaluation has been carried out and contributions to relevant standardization activities have been made. In general, the approach forms a basis for further work in the context of distributed resource management in federated environments, such as Intercloud and Edge Computing approaches, multidomain Software Defined Networks (SDNs), and the Internet of Things (IoT).

Validation & Outlook

ZUSAMMENFASSUNG

Die gemeinsame Nutzung und Bereitstellung geografisch verteilter Betriebsmittel ist das zugrundeliegende Konzept im Forschungsbereich Verteiltes Rechnen. Hierfür sind zahlreiche Architekturen entworfen worden, um heterogene Ressourcen über Verwaltungsdomänen hinweg administrieren zu können, was als Föderation bezeichnet wird.

Forschungsbereich: Verteiltes Rechnen

Ein besonderes Anwendungsgebiet ist die Durchführung von Experimenten zur Erforschung des Internets der Zukunft. Angesichts der Größe, der Komplexität und Heterogenität des Internets, wird eine experimentelle Validierung meist im großen Maßstab in verteilten Infrastrukturen durchgeführt. Infolgedessen sind mehrere unabhängige Testumgebungen aufgebaut sowie entsprechende Protokolle entworfen worden, um den gesamten Lebenszyklus eines Experiments standortübergreifend durchführen zu können.

Eingrenzung: Verteilte Experimente

Eine wichtige Herausforderung in dem Zusammenhang ist der Austausch von Informationen über die vorhandenen Ressourcentypen und -eigenschaften. Existierende Arbeiten beruhen auf einer Vielzahl von Schnittstellen und syntaktischen Datenmodellen mit beliebigen Erweiterungen und Bezeichnern, die eine Verwaltung von heterogenen Ressourcen zwischen autonomen Testumgebungen nur eingeschränkt ermöglichen.

Problemstellung: Ressourcenbeschreibung

Die vorliegende Arbeit stellt einen Ansatz vor, der auf wohldefinierten semantischen Informationsmodellen und architektonischer Abstraktion beruht. Basierend auf Grundlagen die ihren Ursprung im Forschungsbereich Semantic Web haben, werden semantisch annotierte Graphen genutzt, um automatisierte Schlussfolgerungen sowie die Verknüpfung, Abfrage und Validierung von heterogenen Daten zu ermöglichen.

Eigener Ansatz: Semantik

Die wichtigsten Beiträge der Arbeit sind die Definition einer Ontologie für die Verwaltung des Lebenszyklusses von Ressourcen in föderierten Infrastrukturen, der Entwurf einer Semantik-basierten Systemarchitektur zur Abstraktion von Schnittstellen und ein Machbarkeitsnachweis in Form einer Implementation.

Wissenschaftlicher Beitrag

Das Ergebnis wurde in mehreren europäischen Forschungsprojekten und Testumgebungen validiert. Ferner sind eine Leistungsbeurteilung durchgeführt und Beiträge zu entsprechenden Standards erbracht worden. Der Ansatz bietet eine Grundlage für weitere Forschung zur Verwaltung von verteilten Ressourcen in föderierten Umgebungen, wie Intercloud- und Edge-Computing-Ansätzen, Software-basiereten Multidomain-Netzen (SDNs) und dem Internet der Dinge (IoT).

Validierung & Ausblick

TABLE OF CONTENTS

LIST OF FIGURES

List of Listings

List of Tables

Chapter 1

Introduction

1.1 Background and Motivation

Research Context: The Internet

The idea of a network of networks first proposed at the 1960th in the Interuniversity Communications Council (EDUCOM) [p197] workshops became the current Internet. As explained in detail in [p135, p148], its foundations were laid by Kleinrock in 1961 by publishing the theory of packet switching [p137] and by Licklider who published his vision of a "Galactic Network" [p150] in 1962. The term itself was first coined in 1974 in the Request for Comments (RFC) number 675 "Specification of Internet Transmission Control Program" [t14].

Research Area: Distributed Computing

These developments paved the way for concepts allowing computing tasks to be distributed between different geographically distributed areas. However, most of the current approaches are rooted in high speed network experiments that did not take place not until 1992, like the gigabit test environment at the University of Illinois [p37]. In particular in the Information Wide Area Year (I-WAY) [p56] experiment in 1996, where 17 facilities in the United States (US) and Canada were interconnected, made such an approach public to the wider research community. These trials built up the idea for forming the Metacomputing [p206] concept.

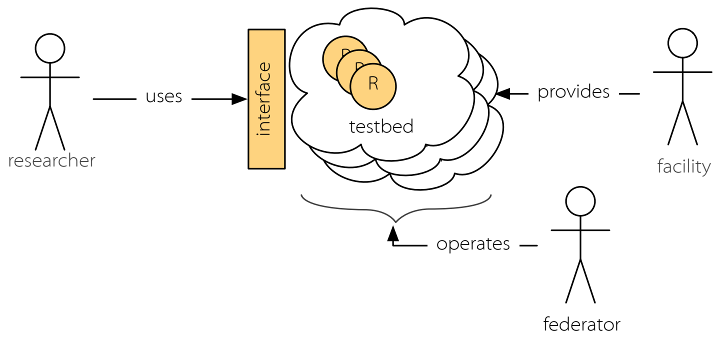


Figure 1.1: Simplified overview of the two-sided market (based on [p52])

Application Area: Distributed Experimentation

A specific field of application is the distribution of experiments between geographically distributed test environments in the context of Future Internet (FI) research. While the extraordinary growth and socioeconomic influence of the Internet is omnipresent, it remains an evolving and unfinished work [p38]. As a result, many research activities around the globe are focusing on defining and developing architectures for the FI in order to overcome the limitations of the current Internet [p175]. Under the umbrella of experimentally driven FI research, the original design and protocols [p45] are gradually extended to meet new requirements, while conserving compatibility with existing mechanisms. Despite some progress in this manner, it has become apparent that this evolutionary strategy can't proceed forever [p46, p73, p193, m15]. Clean-slate approaches, which tend to break compatibility with established designs, moved into the focus of research. They investigate fundamentally new communication protocols and blueprints to solve issues the current Internet design cannot cope with.

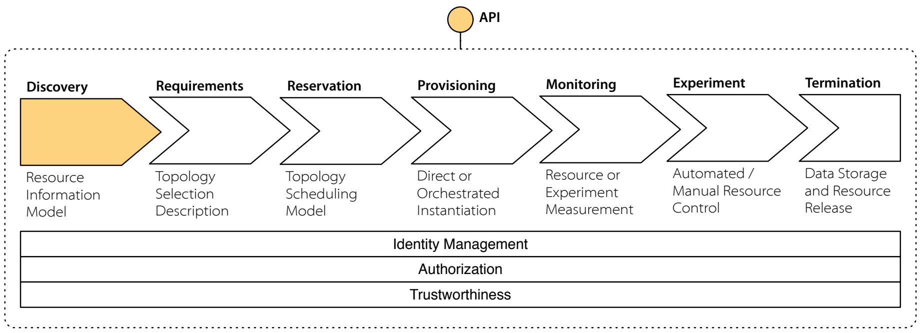


Figure 1.2: The experiment life cycle (based on [a16])

Research Focus: Federated Testbeds

In order to evaluate, whether a developed solution meets the given requirements and solves the targeted issue, analytic modeling, simulations, case or field studies or the measurement of real environments can be conducted [p124, p233]. Given the scale, complexity and heterogeneity of the Internet, developments need to be evaluated involving diversified physical systems. Therefore, several environments for experimentally driven research have been established [t25], in particular within the Future Internet Research and Experimentation (FIRE) [p94] and Global Environment for Network Innovations (GENI) [p17] initiatives. These environments, termed testbeds, are usually built specifically to cater to the needs of particular use case scenarios and therefore contain a set of specialized resources needed for the analysis in question. In order to allow access to resources from different testbeds for large-scale experiments, current research concentrates on mechanisms to federate testbeds from different facilities with each other. Figure 1.1 sketches the relevant two-sided market that is build on the platform economics [p67] concept. This approach increases on the one hand the reasonability and scalability of experiments, and on the other hand the visibility and usefulness of single testbeds.

Taxonomy

As a result, several competitive approaches are under development to interconnect testbeds. Since reproducibility and automation are needed to gain scientific knowledge from experiments, all areas of the relevant life cycle (Figure 1.2) have to be covered. This includes federated Authorization (AuthZ) and Authentication (AuthN), resource description, discovery, reservation, orchestration, provisioning, monitoring, and release, as well as experiment control and measurement [a16]. For this, a volunteer peer-to-peer infrastructure federation approach is followed within the GENI and FIRE context, as shown in Figure 1.3.

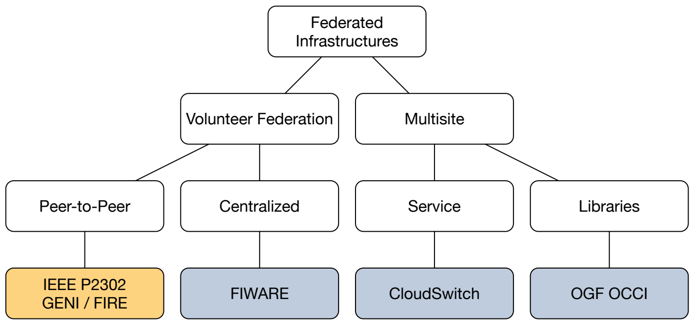


Figure 1.3: Taxonomy of federation approaches (based on [p104])

1.2 Problem Statement

State of the Art

The work laid out above leads to a wide range of interesting research questions and, in recent years, a variety of frameworks, protocols, and architectures have been designed for the above described purposes. Currently, particular attention is paid to the Slice-based Federation Architecture (SFA) [t56] for resource discovery and provisioning; the cOntrol and Management Framework (OMF) [p192], with its Federated Resource Control Protocol (FRCP) [p191] for experiment control; and the ORBIT Measurement Library (OML) [t45, p204], with its OML Measurement Stream Protocol (OMSP) for experiment measurement and resource monitoring. To support the whole experiment life cycle, a combination each of these approaches is required, which introduces a number of challenges. This thesis focuses on the two most fundamental of these challenges.

Issue: API Incompatibility

First, interoperability between these systems is problematic as they have chosen different communication protocols. While SFA uses Transport Layer Security (TLS) enabled Extensible Markup Language (XML) Remote Procedure Calls (RPCs), FRCP exchanges signed messages over the Extensible Messaging and Presence Protocol (XMPP) or Advanced Message Queuing Protocol (AMQP), and OMSP transports data via plain Transmission Control Protocol (TCP) sockets without safety precautions. Therefore, exchanging information between these protocols requires significant development efforts.

Issue: Resource Description

Second, interoperability between these approaches is further aggravated by the use of incompatible data models. Within SFA, testbed-specific XML-based Resource Specifications (RSpecs) are used to describe resources within an infrastructure. FRCP uses either XML or JavaScript Object Notation (JSON) depending on the transport protocol. Within OMSP, arbitrary tuples can be defined. Based on these data formats, each testbed uses its own independent definitions for resource types, resource control capabilities, resource monitoring information and further management data, such as reservation information.

Synopsis

Both issues prevent mutual understanding and minimum interoperation, despite the fact that this was the major objective of the above mentioned protocols. With n testbeds participating in a federation, this leads to a combinatorial problem of n^2 required conversions, not including the needed handover implementations between different protocols. Further, given the use of semistructured data models within the above mentioned contexts with their implied semantics, these transformations further need complex functional code. This also restricts meaningful operations on the data, such as the retrieval of information about equality, symmetry, transitivity or dependencies between resources. A single, canonical reference model would reduce this complexity to $2n$ or even n. In addition, the utilization of a formal information model (cf. Figure 1.4) would allow logical conclusions to be automatically inferred using descriptive languages, which helps to extract and query the information about resources.

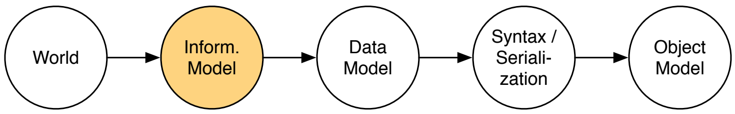


Figure 1.4: Relationship between models and syntax (based on [t57, p187])

1.3 Assumptions and Scope

Research Assumptions

Based on experience gained from current research and a survey presented in [a16], several assumptions underlie the work in this thesis. First, experimentation and federation mechanisms already in place are unlikely to be replaced in the medium term. Therefore, existing work and implementations have to be incorporated into any envisioned solution. Second, a majority of the functionality of these technologies focus on the implementation of the experiment life cycle. Consequently, it forms a common foundation for each mechanism. Third, everything is a resource that can be federated, including a service, a testbed itself or personnel. Hence, they all share some information on a higher level of abstraction. Fourth, existing testbed features, such as user databases or billing mechanisms, must be incorporated. That means a potential architecture has to take external services into account. Fifth, the concept of federated resource management will be adopted by more fields of application in the future. While the requirements are derived from a specific area of application, the solution should not be limited to this area. Sixth, the description of resources is the most important underlying foundation. A sophisticated model allows interoperability between different Application Programming Interfaces (APIs) throughout the whole experiment life cycle.

Research Scope

To limit the scope of this thesis, research is restricted to the topics highlighted in Figures 1.1 to 1.4. That is to say, this work treats as its main subject matter the different APIs and most importantly, the description and discovery of resources. Inversely, this implies that other parts of the experiment life cycle are not the focus of this research. Notably, the main emphasis does not lie in developing new user tools (e.g. for experimentation) or protocols, although these may be supported or enhanced in future work. The emphasis of the work is reflected by the title of this thesis, whose main terminology is as follows:

Semantic-Based Management Resources are managed based on their semantics, i.e. their underlying meaning and relations, while specific descriptions, data models and necessary API interactions are abstracted. In other words, heterogeneous resources are described in a formalized manner to build a basis for their management.

of Federated Infrastructures This work is generally applicable for infrastructures that are grouped under a central administration but maintain their internal autonomy. While this includes Information and Communication Technology (ICT) infrastructures at large, it includes in particular Intercloud [p104] sites, Internet of Things (IoT) [p3] islands or distributed Software Defined Networking (SDN) [p54] topologies.

for Future Internet Experimentation The specific area of application from which the requirements are derived and against which the applicability is evaluated, is the field of experiment driven research for the FI. In this context, so-called testbeds are accessible for researchers and are federated with each other to allow large-scale experimentation.

1.4 Objectives and Contributions

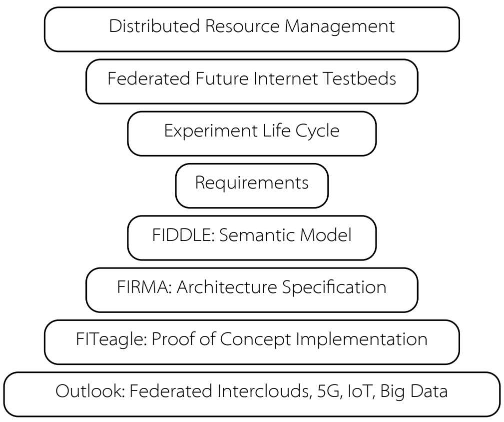


Figure 1.5: Structure of research

Research Objectives & Contributions

Based on this scope, the objectives are to answer two main research questions. First, how to model heterogeneous resources in federated testbeds to support the whole FI experiment life cycle. Second, how to design an architecture that supports this approach and is extensible enough to allow further fields of application to use it. To answer these questions, the major contributions of the present work are the three different contributions and the related structure of research is shown in Figure 1.5:

Ontology

Resource Information Model (FIDDLE) In order to describe heterogeneous resources in federated testbeds for FI experimentation, concepts of the Semantic Web [p18] have been adopted. As a result, semantically labeled, directed multigraphs have been used to design a canonical information model named Federated Infrastructure Description and Discovery Language (FIDDLE) [a20]. This includes the definition of federations, infrastructures, abstract resources and services, life cycle phases, and their relationships. The model went on to act as a seeding document for the Open-Multinet (OMN) [a24] ontology, which has been developed within an international consortium and is further extended within the World Wide Web Consortium (W3C) Federated Infrastructures Community Group[1], of which the author of this thesis is the chair of. Further, an open-source translation mechanism has been developed to convert the graph from and to GENI RSpecs and other data models, such as the Topology and Orchestration Specification for Cloud Applications (TOSCA) [t55] defined by the Organization for the Advancement of Structured Information Standards (OASIS).

Architecture

Resource Management Architecture (FIRMA) By building upon a semantic-driven Microservice [m5, p166, p216] design pattern, an architecture called Federated Infrastructure Resource Management Architecture (FIRMA) [a21] was designed. Following the terminology defined by the Institute of Electrical and Electronics Engineers (IEEE) in the technical report 42010 [t73], an architecture is described as “fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution”. The design allows both the requirements in the given context to be met, and its use in further federated infrastructure environments. It supports, on the one hand, the abstraction away from delivery mechanisms like SFA, FRCP or OMSP and from resources such as Virtual Infrastructure Managers (VIMs), Network Functions (NFs) or specific hardware. On the other hand, it allows the integration of internal services, like monitoring or billing systems, and of shared modules, such as user management or data storage.

Implementation

Resource Management Framework (FITeagle) In order to evaluate these approaches, an extensible, open-source proof-of-concept framework called FITeagle[2] [a23] was developed and used as a reference implementation in several European projects and FI testbeds. Here the definition of [p75] is followed, describing a framework as “a semicomplete application [. . . that. . .] provides a reusable, common structure to share among applications.” For evaluation purposes and to show its applicability to current research projects, particular attention has been spent on the development of semantic-enabled SFA, Representational State Transfer (REST) and OMSP interfaces. To show its applicability to further fields of application, interfaces for the IEEE Standard for Intercloud Interoperability and Federation (P2302) [t9, a10] efforts have been implemented as well.

[1] `https://w3.org/community/omn`
[2] `http://fiteagle.org`

1.5 Methodology and Outline

The research methodology followed is depicted in Figure 1.6 and is reflected in the structure of this thesis:

State of the Art

Chapter 2: After the initial motivation and objectives have been described, the state of the art is highlighted in order to bring the thesis contribution into context.

Requirement Analysis

Chapter 3: Given the overview of the state of the art and the focus on federated FI experimentation, the requirements of the related stakeholders are analyzed, listed and discussed.

Ontology

Chapter 4: The related work in the field of semantic resource description is analyzed. This analysis is the result of detailed studies and examination of existing work in the context of resource description in related environments to identify the main open research issues and reusable approaches. As a result, the design and specification of a new ontology is presented.

Architecture

Chapter 5: The design and specification of an ontology-based architecture is given that adopts well-known design patterns for reusable and extensible systems.

Implementation

Chapter 6: As a result of the research conducted and the specifications identified, a reference implementation was created.

Evaluation

Chapter 7: The implementations of the information model and the architecture were used to evaluate and refine the research conducted for this thesis. Details about the evaluation carried out and project-based validation are presented.

Summary

Chapter 8: Finally, a synopsis of the work done in the thesis is presented, highlighting open research questions.

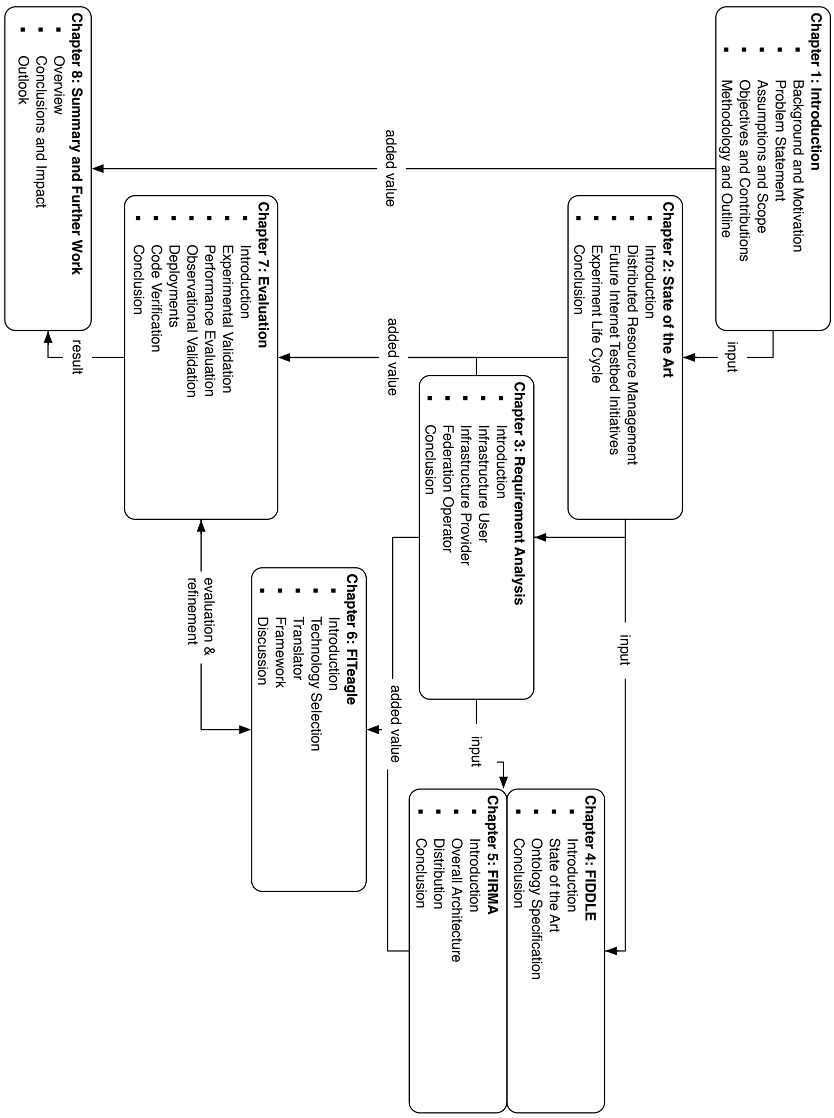


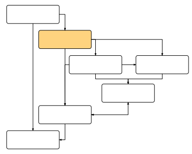
Figure 1.6: Workflow of the research and structure of the thesis

CHAPTER 2

STATE OF THE ART

2.1 Introduction

Overview



In order to place the contribution of the thesis in context, this chapter provides an overview of the state of the art in both the area of research in question and related research areas. First, the essential fundamentals and historical context regarding distributed resource management are provided. The focus is subsequently narrowed by presenting an overview of the specific field of application of federated FI experimentation. After presenting relevant initiatives and approaches, details about relevant life cycle phases, challenges and technologies in this context are provided.

2.2 Distributed Resource Management

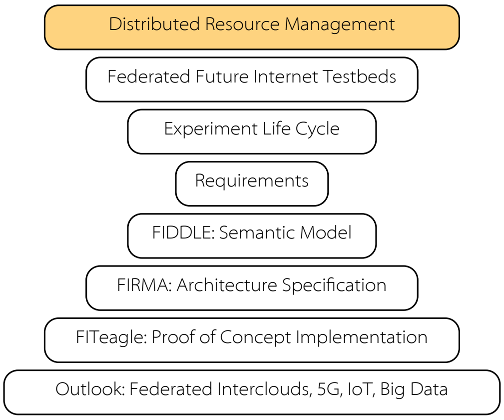


Figure 2.1: Distributed resource management within the structure of research

Distributed Computing

It was Kleinrock who in 1969 already predicted [p136] that: “We will probably see the spread of ‘computer utilities’, which, like present electric and telephone utilities, will service individual homes and offices across the country.” Following this vision, the overall context of the work conducted in this thesis is contextualized within the field of distributed computing, the study of distributed interconnected systems (cf. Figure 2.1). More specifically, the main priority is the distributed management of resources. According to Gartner[1], Distributed Resource Management (DRM) is “an evolving discipline [...] for enabling distributed enterprise systems to operate effectively in production [...and ...] embraces solutions [...] needed to maintain effective productivity in a distributed networked computing environment.”

Focus

To limit the scope of this thesis, the focus is set on the management of federated infrastructures. The *New Oxford Dictionary of English* [p178] defines a federation as “an organization or group within which smaller divisions have some degree of internal autonomy”. A federation in the ICT sector is an agreement between independent autonomous administrative domains that are grouped under a central administration to interoperate with each other for economic or organizational reasons. Therefore,

[1] http://gartner.com/it-glossary/drm-distributed-resource-management

noteworthy approaches and milestones with respect to this definition are briefly outlined in the following subsections.

2.2.1 Metacomputing

Metacomputing Definition

Metacomputing characterizes a specific form of distributed computing in which computing centers are interconnected by high performance networks. While concepts for the distribution of tasks had already been developed in the 1960s, the term was first coined by Larry Smarr in 1992 [p206]. Around this time, early experiments within gigabit networks at the University of Illinois [p37] and, later, the I-WAY project in particular increased awareness of this field of research.

Issues

Within the I-WAY experiment, a distributed Asynchronous Transfer Mode (ATM) based testbed was established to demonstrate the concept of distributed supercomputing sites. One major use case was executing large-scale scientific simulations across multiples facilities. For these simulations, the Message Passing Interface (MPI) implementation MPI Chameleon (MPICH) was used to execute code in parallel and it became clear that more sophisticated protocols and architectures are needed in order to distribute tasks at that scale. As a result, the communication library Nexus [p86] was developed beneath MPICH to support efficient wide-area computations by introducing global pointers and remote service requests, with AuthN mechanisms and environment monitoring information.

2.2.2 Grid Computing

Grid Computing Definition

Based on experiences from the Metacomputing context in general and the I-WAY experiments in particular, the concept of Grid Computing [p83] was born and gained large commercial and scientific attention in 1998. Following the definition of [p84], “a computational grid is a hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities.” This definition has further been refined by [p79, p85] and the following three point check-list was defined [p81]: A Grid is a system that

- coordinates resources that are not subject to centralized control…
- …using standard, open, general-purpose protocols and interfaces
- …to deliver nontrivial qualities of service.

Implementations

An exhaustive history of the Grid is given in [p78] and, over time, a number of architectures, such as the Distributed Resource Management Application API (DRMAA) [t13], as well as standards and implementations were defined. Three distinct major lines of developments can be identified: First, the Globus Toolkit (GT) [p77, p80] that specifies Resource Specification Language (RSL), Globus Resource Allocation Manager (GRAM), Monitoring and Discovery System (MDS) and Grid Security Infrastructure (GSI); Second, gLite [p145] using the Job Definition Language (JDL) and Workload Management Service (WMS); Third, the Uniform Interface to Computing Resources (UNICORE) [p119] with Abstract Job Objects (AJOs) and a Network Job Supervisor (NJS).

Research

Along with these implementations, a number of related research questions have been addressed in the literature. For example, a taxonomy of Grid workflow colocation and scheduling problems have been described in [p241]. Similarly, a number of approaches

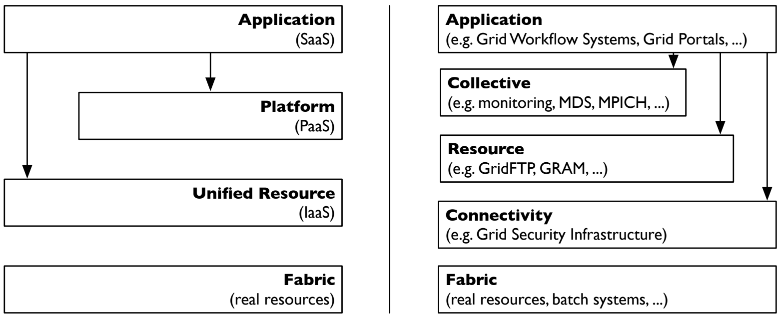


Figure 2.2: Comparison of Cloud and Grid Computing layers (based on [p87])

for Quality of Service (QoS) and dynamic Service Level Agreement (SLA) [p213] negotiation approaches are introduced [p64, p82, p103, p144, p208].

Grid Standards

As a result, the Open Grid Forum (OGF) specified a set of standards within the Open Grid Service Architecture (OGSA) [t24] to allow interoperability between heterogeneous Grid systems. This specification covers, for example, job management using the Job Submission Description Language (JSDL), AuthN via a Public Key Infrastructure (PKI), AuthZ using the Extensible Access Control Markup Language (XACML) [t61] and Security Assertion Markup Language (SAML) assertions, or resource state manipulation via the Open Grid Services Infrastructure (OGSI).

Adoption

These standards had partly been adopted by the different Grid implementations and further approaches such as WS-Agreement (WSAG) [t27] for SLA negotiations were candidates for inclusion. However, instead WSAG as included in the Web Services Resource Framework (WSRF) and Web Services Notification (WSN) specifications that have been described by OASIS as an alternative for implementing the OGSA capabilities using Web services.

2.2.3 Intercloud Computing

From Grid to Cloud

Although Grid Computing was claimed to be the new infrastructure for the 21st century [p76], it is the Cloud Computing paradigm that has instead been attracting the interest of the Information Technology (IT) industry [p122] and has been viewed as an economical model for renting technical resources [t36, p232]. In [p87] both approaches are compared with each other in detail, in [p125] the transition from Grid to Cloud Computing is analyzed, and in [p30, p95, m8, p160, p230] over twenty definitions of Cloud Computing are given. As depicted in Figure 2.2 a Cloud can offer three different service models: Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS).

Cloud Standards

Alongside to this Cloud Computing paradigm shift, a set of standards have been proposed [t76] and in [p172] the challenges are discussed that arise by the over involvement of vendors and standard bodies. One of the first standards was the Open Cloud Computing Interface (OCCI) [t47], which was defined within the OGF. Within the same forum, the Infrastructure On-Demand (ISOD) research group published a report on best practices for on-demand infrastructure service provisioning [a1]. Similar to the OCCI, the Distributed Management Task Force (DMTF) worked on the Cloud Infrastructure Management Interface (CIMI) [t75] in conjunction with the Open Virtualization Format (OVF). To manage data in the Cloud, the Storage Networking Industry Association

(SNIA) and the International Organization for Standardization (ISO) / International Electrotechnical Commission (IEC) group 17826 defined the Cloud Data Management Interface (CDMI) [t74]. Further, OASIS has defined the Cloud Application Management for Platforms (CAMP) specification that focuses on PaaS deployment models and TOSCA, which allows the definition of complex, platform-independent service topologies and their orchestration.

Intercloud However, these standards do not cover the federation and interoperation of administratively independent Cloud sites [p162]. Unlike with the Grid or telephone system or the Internet, this results in a vendor-lock-in situation, aggravated by the introduction of many provider-specific APIs. As a result, the term Intercloud was coined in 2007 and well over 20 designs for interoperability architectures have since been proposed [p176] and a taxonomy and survey of existing architectures was published in 2012 [p104] and 2014 [p218].

Intercloud Standards Due to architectural similarities, approaches from the Grid community, such as GridARS [p214], were adopted to the Intercloud context. As highlighted in [p57], simultaneous with the first academic publications also several Standards Developing Organizations (SDOs) have formed working groups to define Intercloud Computing architectures.

EGI In Europe, the European Grid Infrastructure (EGI) Federated Cloud Task Force is a federation of national and domain specific resource infrastructure providers comprised of individual resource centers. One of the objectives of EGI is to deploy a testbed to evaluate the integration of resources within the existing production infrastructure for monitoring, accounting and information services.

GICTF In Japan, the Global Inter-Cloud Technology Forum (GICTF) was formed to define Intercloud architecture requirements [t83], promote standardization of network protocols and Cloud interfaces [t78] and enhance the reliability of Cloud services.

NIST In the US, the National Institute of Standards and Technology (NIST) provided a definition of Cloud Computing [t37, t46, t70] and formed the Federated Community Cloud working group to support the seamless implementation of federated community Cloud environments.

ITU-T Likewise, the International Telecommunication Union – Telecommunication Standardization Sector (ITU-T) Focus Group on Cloud Computing published a series of technical documents containing functional requirements and a reference architecture [t82] to support different aspects of the Cloud domain, specifically addressing Intercloud capabilities.

IEEE The IEEE formed the working group P2302, which intent is to define required functionalities, protocols and topologies to support Cloud-to-Cloud interoperability within the Standard for Intercloud Interoperability and Federation (SIIF) [t9] chapter of the IEEE. The concept is based on the "Blueprint for the Intercloud" [p19] that was published to define protocols and formats for Cloud Computing interoperability, and a number of subsequent publications including security considerations [p20]. The realization of the envisioned IEEE Intercloud architecture is modeled analog to the design of the current public Internet. In contrast to most other approaches, this work specifically includes a semantic layer for the definition of resources and Cloud models. Mainly based on results produced within the Open-Source API and Platform for Multiple Clouds (mOSAIC) [p164] project, this allows to develop intelligent and autonomous applications exploiting data semantics, such as meaning-based search engines and information brokering.

Others Finally, a number of organized researched groups focuses on narrow parts of the problem, including the Open Data Center Alliance (ODCA); the TeleManagement

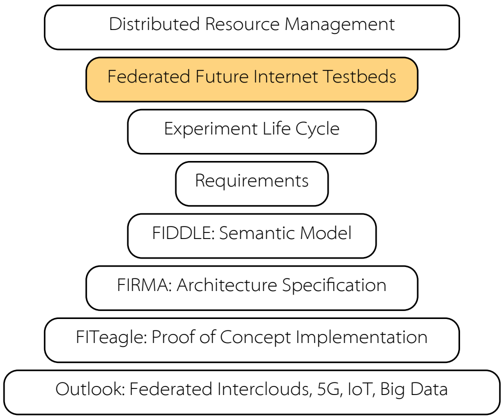


Figure 2.3: Future Internet testbed initiatives within the structure of research

Forum (TMF) [t86]; the Internet Engineering Task Force (IETF), with their Cloud Reference Framework [t35]; and the European Telecommunications Standards Institute (ETSI), with their Cloud Standards Coordination section.

2.3 Future Internet Testbed Initiatives

Federated Testbeds

As indicated in Figure 2.3, federated FI testbeds are a specific area of application for the distributed ICT infrastructure management mechanisms described above. Under the umbrella of FI research, the design of the current Internet has been gradually extended to meet new requirements since the 1960s. In other words, “the Internet is broken” [m15] and several international initiatives have been established to fix it. In this context, a number of FI approaches are evaluated experimentally within distributed test environments. Testbeds are federated with each other in order to both make as many testbeds as possible available to experimenters and, therefore, allow large-scale experimentation; and to increase the visibility and usefulness of single testbeds. Conceptually, the approaches in this context chosen for this thesis are based on Metacomputing, Grid Computing and Intercloud paradigms.

Initiatives

During federations a two-sided market is spanned which generates an added value for both researchers and facility providers (cf. Figure 1.1). Based on this, various initiatives have been established worldwide, as sketched in Figure 2.4. The major FI programs are the GENI and the Future Internet Design (FIND) programs in the US and the FIRE and Future Internet Public Private Partnership (FI-PPP) [p110] initiatives and the European Institute of Innovation and Technology (EIT) in Europe. The programs with mechanisms for federating testbeds will be briefly described in the following sections. To provide some broader context, further noteworthy programs are highlighted below.

Asia

In Asia, the Asia-Pacific Advanced Network (APAN) and the joint activities Asia Future Internet Forum (AsiaFI) and PlanetLab China, Japan, Korea (PlanetLab CJK) [m3] have been established. In Japan, the AKARI[2] project with the Japan Gigabit Network 2+ (JGN2Plus) and the Collaborative Overlay Research Environment (CORE) testbed networks are operated and administered by the National Institute of Information and

[2] http://akari-project.nict.go.jp

Figure 2.4: Overview of FI-related initiatives and noteworthy projects

Communications Technology (NICT). Further, the China Next Generation Internet (CNGI) project focuses on enhancing the scalability of IPv6. In South Korea, various projects in the Korea Institute of Science and Technology Information (KISTI), the Future Internet Forum (FIF), and the National Information Agency (NIA) are focusing on improving the structure of the existing Internet. These projects are either based on the Korea Advanced Reseach Network (KOREN), operated by the Electronics and Telecommunications Research Institute (ETRI), the Kyungpook National University (KNU) and the Chungnam National University (CNU); or the Korea Research Environment Open NETwork (KREONET), supported by the Ministry of Education, Science and Technology (MEST) and operated by KISTI. Further examples for FI-projects include the Future Network 2020 (FN2020), established by the government institutes NIA, ETRI and the Korea Internet and Security Agency (KISA) and the Mobile Oriented Future Internet (MOFI) identity network and architecture that has been developed within ETRI.

Australia

In Australia, the ICT research center National Information and Communication Technology Australia (NICTA) works in close cooperation with the GENI and FIRE initiatives. Together with the GENI Open Access Research Testbed for Next-Generation Wireless Networks (ORBIT) [p173, p194], NICTA developed OMF and is currently the main developer of the framework.

Others

Worldwide smaller projects and initiatives have also been established. In Canada, the Smart Applications on Virtual Infrastructures (SAVI) partnership is composed of industry, academia, Research & Education (R&E) networks, and High Performance Computing (HPC) centers. The research goal of SAVI is to understand elements of future application platforms and to design a flexible infrastructure for them where large-scale experiments can be deployed. In Germany, the German Lab (G-Lab) [t68] project formed a test infrastructure and developed the Topology Management Tool (ToMaTo) [p201], which is also used in the GENI context for educational purposes. Some projects that build cross-continental testing facilities are Testbeds for Reliable Smart City Machine-to-Machine Communication (TRESCIMO) [a12, a14], between Europe and South Africa; Intelligent Knowledge as a Service (iKaaS), between Europe and Japan; and Future Internet Testbeds Experimentation Between Brazil and Europe (FIBRE).

2.3.1 Global Environment for Network Innovations

Overview In the US, three major initiatives can be identified that focus on FI research. All of them build upon the US-wide National Research and Education Network (NREN) Internet2. Funded by the National Science Foundation (NSF), the FIND program addresses mainly foundational concepts and methods for the FI. In contrast, the GENI, established in 2007, focuses on the deployment of experimental platforms and is the initiative with the biggest influence in the context of this thesis. Since 2012, the US Ignite[3] initiative is leveraging these NSF investments to provide platforms and environments for application development with stronger focus on innovation and business. One example is the Global City Teams Challenge (GCTC), which is designed to advance the deployment of IoT technologies within a smart city / smart community environment. The GCTC was established as a joint effort by US Ignite, the Department of Transportation (DoT), the NSF, the International Trade Administration (ITA), the Department of Health and Human Services (HHS) and the Department of Energy (DoE).

GENI Frameworks Within GENI, the GENI Project Office (GPO) is the management and execution body, which coordinates the architecture, system engineering, costs and schedule of the projects. The GENI infrastructure is composed of multiple federated testbeds, which are controlled by five different competing control frameworks. A comparison between these control frameworks was published in [p152] (cf. Figure 2.5): the Open Resource Control Architecture (ORCA) [p10, t16, t17, p40] framework Shirako; PlanetLab Central (PLC), used for the PlanetLab[4] [p43, p181] infrastructure; the DETER Federation Architecture (DFA) [p68, p69], used in the Trial Integration Environment Based on DETER (TIED) [p68]; the OMF developed within ORBIT; and ProtoGENI, used in Emulab.

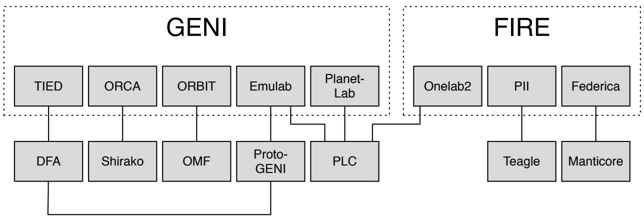


Figure 2.5: Overview of the main control frameworks (based on [p152])

Harmonization The need to federate testbeds controlled by different, competing control frameworks was identified in 2010 [t33]. As a result, in order to address the inherent complexity of heterogeneous testbed infrastructure management, common architectures, protocols and APIs have been developed.

SFA In short, the XML-RPC-based SFA allows federation across facilities using TLS-based AuthN. SFA-aligned testbeds can be controlled by various SFA-compliant user tools, such as the SFA Command-Line Interface (SFI), MySlice[5], Flack[6], Omni[7] or

[3] http://us-ignite.org
[4] http://planet-lab.org
[5] http://myslice.info
[6] http://protogeni.net/wiki/Flack
[7] http://trac.gpolab.bbn.com/gcf/wiki/Omni

jFed[8]. Resources are described using RSpecs and are grouped within Slices, i.e. requested virtual topologies. These RSpecs are arbitrary XML-based documents that define the offered (Advertisement RSpec), requested (Request RSpec), and allocated (Manifest RSpec) resources. A common denominator is the GENI RSpec v3, which includes definitions of Nodes and Links and can be extended by further XML Schema Definitions (XSDs). The formalization of resource descriptions are still the subject of current research [p219].

OMF

While SFA mainly addresses issues regarding the description, discovery and provisioning of resources, mechanisms are also needed to describe experiments and to control and monitor resources accordingly. As a result, OMF was developed and has been widely adopted to execute reproducible experiments. Workflows are described using the OMF Experiment Description Language (OEDL), a Domain Specific Language (DSL) based on Ruby. An experimenter using an Experiment Controller (EC) to send the related messages and commands to a Resource Controller (RC). Its underlying communication architecture is currently transitioning into a federation-enabled version initially called OMF-Federated (OMF-F) [t59], which uses FRCP to exchange signed messages over XMPP or AMQP. Another EC, called Network Experimentation Programming Interface (NEPI) [p89, p143, p189], allows complex experiments to be described and executed, has been enhanced to be compatible with FRCP. NEPI uses its own DSL to define workflows and supports combining resources from three different types of experimentation platforms: simulators, emulators, and real testbeds. However, in a similar fashion to RSpecs, either XML- or JSON-based arbitrary data structures are used to specify resources, which aggravates interoperability. A secure handover between SFA and FRCP is also the subject of current research [p209].

OML

Finally, to collect and transport monitoring information from experiments, OMSP was defined and, along with OML, a common implementation is provided to experimenters. OML is a measurement framework that enables experimenters to instrument their application by defining measurement points inside their application source code. These data are transported as streams from the measurement points with the help of the OML client libraries and stored in an OML server. The OMSP protocol communicates via plain TCP sockets and sends arbitrarily defined tuples.

2.3.2 Future Internet Research and Experimentation

Overview

The situation in the European Union (EU) is even more diverse than in the US [p175, p238]. Initiatives include the Future Internet Assembly (FIA), including the FI-PPP and FIRE initiatives; and the European Technology Platforms (ETPs), such as the IoT focused ETP on Smart Systems Integration (EPoSS) and NetWorld 2020, a fusion of the ETPs Integral SatCom Initiative (ISI) and Net!Works.

FIRE

Comparable to its US counterpart GENI, the FIRE initiative focuses on exploratory FI research. It enables experimentation by providing ICT facilities in targeted research communities. These facilities are connected via the NREN GÉANT, and represent the main application area of this thesis. Under the umbrella of FIRE, some notable projects, testbeds and support activities have been established. Beginning with the Pan-European Laboratory (Panlab) [p92] project, an analysis was conducted [p35, p238] of possible heterogeneous resource federation scenarios in large-scale experimental facilities. Subsequently, along with the relevant test facilities, experimentation frameworks were developed within the OneLab2 [p72], PII [p224, p239] and Federica [p34] projects

[8] http://jfed.iminds.be

(cf. Figure 2.5). Specifically, PLC was adopted to establish PlanetLab Europe (PLE), Manticore [p101] provided users an IaaS framework for logical Internet Protocol (IP) networks, and the Teagle [p93, p223, p236, p238, p240] framework was developed to cover the complete experiment life cycle.

FIRE Harmonization Teagle was the outcome of the PII project and served to fulfill the original proposal for a FIRE-wide federation architecture. Each testbed in the federation configures a Resource Adapter (RA) [p237] or describes it using the Resource Adapter Description Language (RADL) [p240] for each of their heterogeneous resources and exposes them using the Directory Enabled Networks New Generation (DEN-ng) [p210] data model by running a Panlab Testbed Manager (PTM) [p237]. The experimenters create their own Virtual Customer Testbeds (VCTs) using the Virtual Customer Testbed Tool (VCT-Tool) [p236] and requests are authorized using a Open Mobile Alliance (OMA) Policy Engine (PE). Based on this VCT, an experiment can be described and controlled using the Java API Federation Computing Interface (FCI) [p223].

OpenLab However, with the objective to harmonize the existing FIRE frameworks, in particular within the OpenLab [p153] project, SFA and OMF were identified as common determinants to be used for FI experimentation. Based on this, one approach followed in the context of OpenLab was to provide a wrapper to support testbeds adopting SFA to federate their infrastructures. The resulting SFAWrap[9] framework was extracted from PLC and requires the infrastructure owner to implement a set of drivers to integrate existing web services for discovery, reservation, provisioning and release of resources.

Fed4FIRE Finally, as a successor of the OneLab2 project, the Federation for FIRE (Fed4FIRE) [a16] project was started in 2012. Fed4FIRE focuses on federating European facilities by unifying existing experimentation and management tools and procedures. After gathering requirements within the FIRE community, a generic architecture for the heterogeneous federation of FI experimentation facilities on a larger scale was defined (cf. Figure 2.6). These requirements included, in particular, compatibility between FIRE and GENI facilities. It was identified that federated testbeds should support all the functions of the experiment life cycle: resource description, resource discovery, resource requirements, resource reservation, resource provisioning (direct or orchestrated), experiment control, facility monitoring, infrastructure monitoring, experiment measuring, permanent storage, and resource release. They should additionally support federated identity management, AuthZ, and SLA management [a16]. Analog to GENI, SFA has been adopted for resource discovery and provisioning; the FRCP-based tools OMF and NEPI, for experiment control; and OMSP, for transporting monitoring information about experiments, infrastructure and facilities. Furthermore, different possible federation architectures were evaluated. Based on identified characteristics, the heterogeneous federation approach was recognized to be the most suitable for the FI experimentation facilities under evaluation. In this architecture, all testbeds run their own native testbed management software.

Sustainability Within most FIRE projects it was discussed how infrastructures could be operated in a self-sustainable fashion. Continuity of facilities beyond the duration of the funding of particular projects, was challenging in many cases. Numerous facilities disappeared after the funding ended. A notable exception was noncommercial approaches, such as PlanetLab in the US or PLE in Europe, which provide access to a large-scale network of (mostly academic) computing resources through in-kind contributions models. Therefore, several further approaches and initiatives aimed to solve these sustainability problems.

[9] http://sfawrap.info

Figure 2.6: Fed4FIRE view of the cross-testbed federation ecosystem [a16]

Panlab

In 2006, the objective within Panlab was to work on business models and strategic development guidelines to establish a solid base for a future commercial and long-term self-sustainable operation. Panlab produced several relevant deliverables, including a Legal Framework [t26], a Vision for Pan-European Laboratories [t32], and a business model for the so called Panlab Office. However, the Panlab Office never saw the light of day, for several reasons, of which only some were technical ones.

FIRE Office

In 2009, within the FIRE Coordination and Support Action (CSA) FIREWORKS, the working group on modular federation of FIRE facilities, also called the "wise men", specified a manifest targeting collaboration and high-level federation for FIRE facilities. As a result, a basic structure aimed at federating testbeds was identified: the FIRE Office [p52]. Strong emphasis was initially put on technically realizing the federation of testbeds and to make functionalities of federated testbed available for experimenters. This gave birth to the broad spectrum of federation portals and research tools for experimenters mentioned above. Besides technical solutions, a focal point was set on identifying requirements and, at that particular time, also on possibly sustainable models.

CI-FIRE

Following this line of thought, the Coordination and Integration of FIRE Activities in Europe (CI-FIRE) project was working on an action plan for making EU-wide and national FI activities sustainable. CI-FIRE developed a template of an innovation business framework for sharing best practices and new services, performed a gap analysis across FIRE and national initiatives, and created benchmarks for assessing available solutions.

FIRE CSAs

Overall, several CSAs were established within FIRE to coordinate efforts to build a sustainable vision and business models, and to coordinate the role of FIRE facilities. The CSAs coordinated the different projects in a more sustainable fashion, deliberately aiming for unification and harmonization of FIRE testbeds by setting up the FIRE Portal[10] for sharing testbed information, participation, and collaboration. Succeeding the activities PARADISO, FIREWORK, FIREBALL, FIRESTATION, and MyFire, the latest programs FUSION and AmpliFIRE were established. AmpliFIRE started in January 2013 with the intention of supporting the community to prepare FIRE for

[10] http://ict-fire.eu

Horizon 2020[11]. One goal by AmpliFIRE was to develop a sustainable vision and business models, and to strengthen the role of FIRE facilities.

FedSM

One project that stands out in this context is the European Federated IT Service Management (FedSM) [t4] project. Techniques and approaches from commercial IT Service Management (ITSM) processes were analyzed and adopted in order to define and implement a lightweight framework for federated e-infrastructures. In Figure 2.7, both the experiment life cycle and the FedSM business models are depicted in relation to each other.

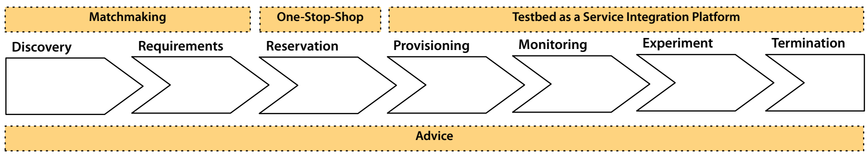


Figure 2.7: The experiment life cycle [a16] (solid) and the FedSM models [t4] (dotted)

2.3.3 Future Internet Public Private Partnership

Overview

Similar objectives of sustainability are targeted by the more market-oriented European Research & Development (R&D) line FI-PPP. Analog to the US Ignite initiative, the European FI-PPP provides platforms and environments for FI application development with stronger focus on innovation and business. A significant contribution is the Future Internet Core Platform (FIWARE) [m7]. Its goal is to deliver a service infrastructure that offers specifications of commonly used functions, so called Generic Enablers (GEs). These specifications are used to build a sustainable foundation for the FI based on their implementations, so called Generic Enabler implementations (GEis).

FIWARE Federation

In the FI-PPP context the federation of different administratively independent infrastructures has also been identified as a requirement for establishing a sustainable pan-European FI developer environment. Within the Experimental Infrastructures for the Future Internet (XIFI) [p4] project, a volunteered centralized federation (cf. Figure 1.3) was initially designed. As sketched in Figure 2.8 and detailed in Figure 2.9, its development was based solely on FIWARE GEis. The goal of the FIWARE federation was to establish a unique marketplace, by provisioning GEis, which are listed in the FIWARE Catalog[12], as a service for developers. Such a federation was considered to be crucial to encounter the current fragmentation of European infrastructures into isolated testbeds, which are individually unable to support large-scale trials.

FIWARE Lab

As a result, the FIWARE Lab[13] [p246] was established. This Open Innovation [p42] lab represents a running instance of such a federation architecture, to provide developers access to related technologies for free experimentation. In order to join the federation and operate various nodes, the FIWARE Ops[14] tools are used to manage deployment, configuration and maintenance.

FIWARE Accelerators

To increase long-term return on investments, the European Commision (EC) made 80 million Euros available for entrepreneurs, Small and Medium Enterprises (SMEs) and startups to use FIWARE-related infrastructures, developments and APIs through

[11] `http://ec.europa.eu/research/horizon2020`
[12] `http://catalogue.fi-ware.org`
[13] `http://lab.fi-ware.org`
[14] `http://fiware.org/fiware-operations/`

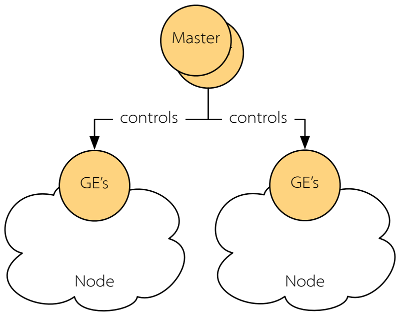


Figure 2.8: Overview of the FIWARE federation approach

open calls under the umbrella of the FIWARE Accelerators[15] program. To build a new self-sustainable ecosystem for the FI, innovative Internet applications in relevant business domains, such as smart cities, eHealth, transportation, energy, manufacturing or logistic, are developed.

2.3.4 European Institute of Innovation and Technology

Overview

Besides FIRE and the FI-PPP, the EIT promotes experimentally-driven research; is striving to create a dynamic, sustainable, large-scale European experimental facility; and is laying the foundation for commercial offers. The EIT Information and Communication Technology Labs (EIT ICT Labs) [p113] is one of the first Knowledge and Innovation Communities (KICs) and supports testbeds at the corresponding collocation nodes to enable them to demonstrate best practices and methodologies. Federation is also perceived as a catalyst to increase the utility of a testbed.

Beta-Plattform

In 2008, the German Beta-Plattform [p29] was established with close ties to Panlab. The goal was to develop business models for continued operation in order to keep research results available after the end of research projects. Although commercial use was also targeted in the long run, the Beta-Plattform initially served as a sustainability plan for German nationally funded ICT project results, such as the Multi-Access Modular Services Framework (MAMSplus) [p207].

Fantaastic

In 2013, the Fanning out Testbeds-as-a-Service for the EIT ICT (FanTaaStic) project again explored new ways of creating a self-sustainable business models for collaborative testbed facilities. In joint collaboration with the CI-FIRE project, the business model was defined and it envisioned third parties engaging with brokering product testing and hardening services. The underlying operational concept was defined based on the Enhanced Telecom Operations Map (eTOM) [p130] and a gap analysis between eTOM and the existing FIRE offerings. In 2014, the brokering service was implemented and served its first SMEs.

2.4 Experiment Life Cycle

Introduction

The overview given in the previous section shows manifold approaches and initiatives that support FI research within federated infrastructures. To limit the scope of this thesis even further, it focuses on scientific FI experimental evaluations (as highlighted

[15] http://fiware.org/accelerators/

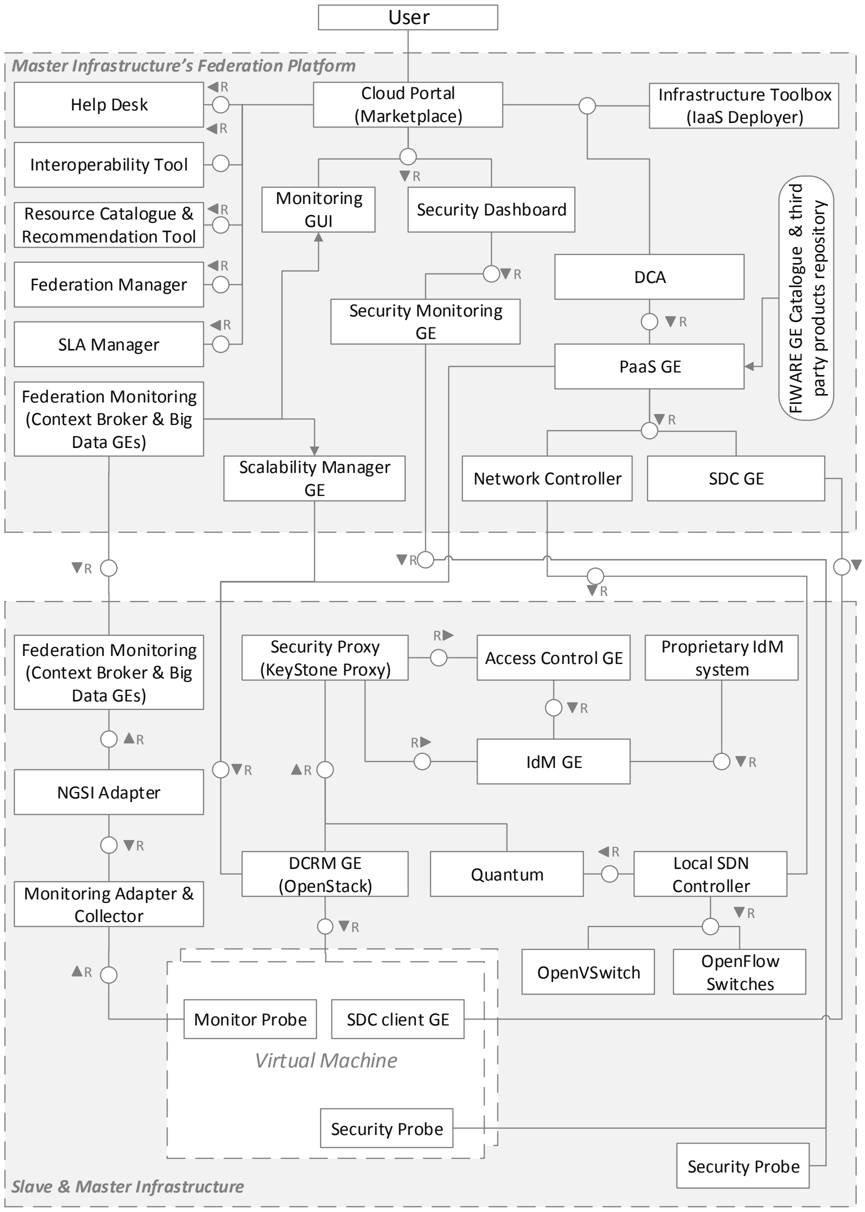


Figure 2.9: The FIWARE Lab federation architecture [a7]

in Figure 2.10), namely the GENI and FIRE initiatives. The workflow of such experiments includes federated AuthN and AuthZ; resource description, discovery, selection, reservation, orchestration, provisioning, monitoring, and release; and experiment control and measurement (cf. Figure 1.2). The following subsections provide further details about each phase.

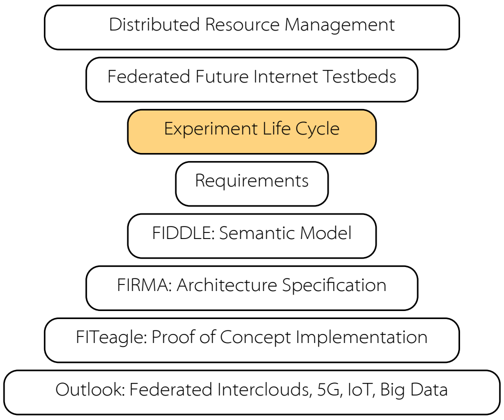


Figure 2.10: Experiment life cycle within the structure of research

2.4.1 Resource Discovery

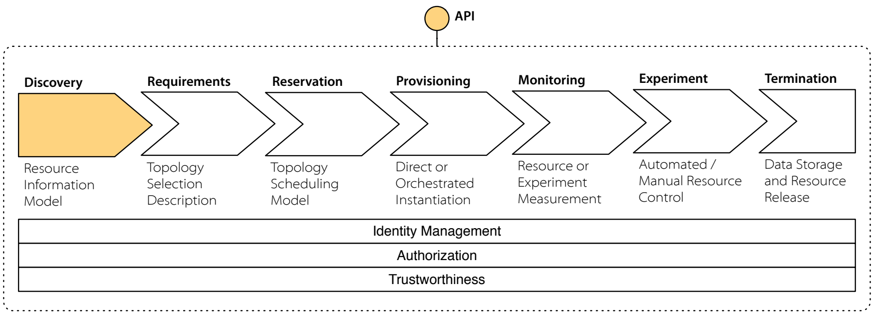


Figure 2.11: Discovery as the first step in the experiment life cycle

Overview

The first step for conducting an experiment is to discover the available resources, as highlighted in Figure 2.11. Given the stakeholder overview from Figure 1.1, this procedure involves at least two parties. While the infrastructure owner has to describe and publish the testbed capabilities, users need a way to discover the available offerings.

Federation Challenges

In federations between heterogeneous autonomous infrastructures, it is important for the federator to facilitate to lower barriers for users. This includes two major aspects. First, the relevant APIs used to publish and discover resources have to be the same across the whole federation. This ensures that a single user tool can be used to access the complete offerings of a federation. Second, the information published by each testbed and the queries submitted by users have to be coherent. This includes a way to combine different offerings, to specify relations between them, and to use this information to provide the user with a reasonable answer to his requests. A major challenge in this regard is that each testbed autonomously describes its resources and services, and as a result, produces disparate markups while sometimes identifying the same resource type.

Technologies

Within the GENI and FIRE initiatives described above, a number of APIs have been implemented for the purpose of federating infrastructures. The SFA Aggregate Manager (AM) API call *ListResources ()* is mainly used to provide a list of available resources for a given testbed. The returned data structure is an XSD-based tree, the GENI RSpec v3, which has a limited set of predefined resources (namely Nodes and Links) and arbitrary extensions that are applicable in any part of the structure. Queries are limited to filtering resources that are currently unavailable.

2.4.2 Resource Requirements

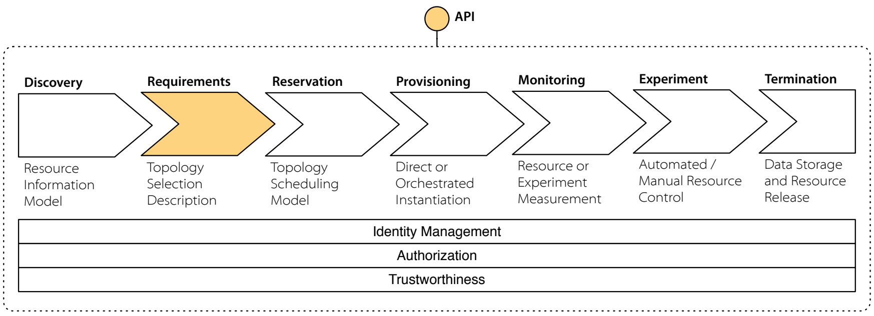


Figure 2.12: Selection of a subtopology in the experiment life cycle

Overview

Next, after the discovery of available resources, the user has to specify the resource requirements for a given experiment, as shown in Figure 2.12. Either concrete resources can be selected or abstract properties can be defined that the requested resources have to fulfill. This also includes the interconnection and dependencies between the resources, as well as further configuration information, as sketched in Figure 2.13. Based on the subgraph of available resources selected, the testbed management system creates an isolated topology, called a *Slice*, on top of an existing substrate.

Federation Challenges

Within a federated environment, the selected resources might not belong to a single administrative domain. Instead, an experiment could span across multiple testbeds and therefore may involve interdomain interconnectivity and require the setup of a network environment, including the configuration of firewalls, Virtual Private Networks (VPNs), dedicated layer 2 connections or Secure Shell (SSH) tunnels.

Technologies

The slice description is based on the XML data structure returned and further contains a reference to the relevant SFA AM of the testbed where each resource is located. Based on this information, each responsible AM API can be called and connectivity between the resources can be configured. If the relevant SFA stitching mechanisms are supported by the interdomain NREN, direct links can also be established.

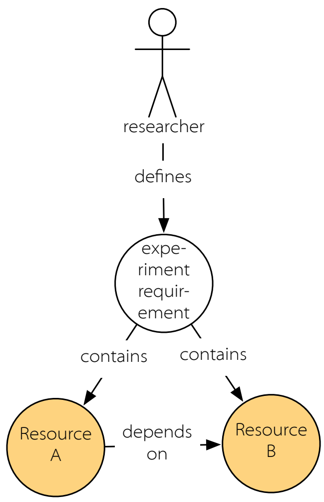


Figure 2.13: Resources and their dependencies within an experiment

2.4.3 Resource Reservation

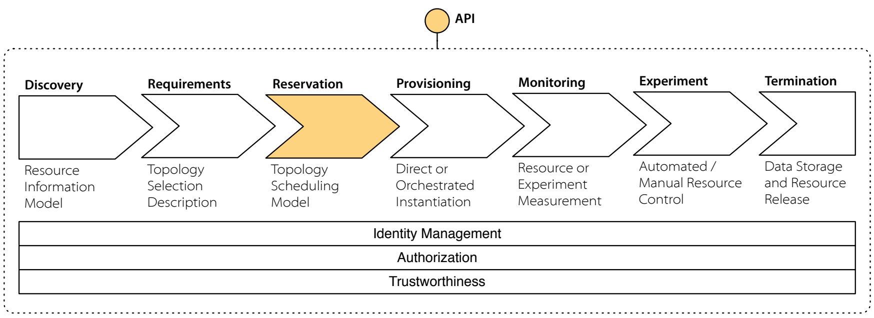


Figure 2.14: Reservation of the selected topology in the experiment life cycle

Overview

Given that reservation information is embedded in the slice description, resources can be reserved by an AM in the next phase, as shown in Figure 2.14. A reservation might include time or capacity related metrics and, in its simplest variant, a resource is reserved immediately for an unspecified duration. If the user has not selected a specific resource in the requirement phase (unbound request), mapping algorithms automatically select resources that meet the user's requirements [p243].

Federation Challenges

The reservation and scheduling of resources is a rather large research area on its own. As shown in Figure 2.15 for example, the capacity of a resource can be scheduled in a preemptive, malleable, or deferrable or in advance, while taking into account the relevant processing times. Since resources in a federated environment are not under centralized management, reservation-related information and information regarding availability of resources are distributed and might also be presented in different ways. As a result, mechanisms are needed to both support the selection of appropriate resources, and to reserve resources in all involved domains. While some testbeds allow basic, time-

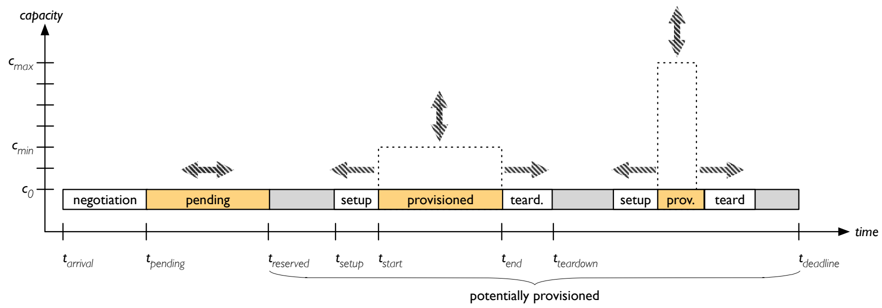


Figure 2.15: Resource reservation types (based on [a15])

oriented reservations within their facility, the FI context introduces higher complexity by including heterogeneous metrics that depend on the type of resources requested.

Technologies Although reservation and scheduling mechanisms in distributed systems have already been researched to a great extent, in particular in the fields of Meta and Grid Computing [p140], due to the heterogeneity of the available resources in the FI, they are still under active research in the GENI and FIRE context. One example is the efficient spectrum slicing in wireless testbeds as presented in [p6]. Here, the Network Implementation Testbed using Open Source code (NITOS) [p5] for wireless experiments operates a scheduler that is aware of the location of all available nodes and enables resource sharing based on this information in order to allow multiple users to conduct experiments simultaneously without interference. To allow analog federation-wide scheduling mechanisms, the relevant information has to be specified and exchanged using SFA and scheduling mechanisms have to take these metrics into account.

2.4.4 Resource Provisioning

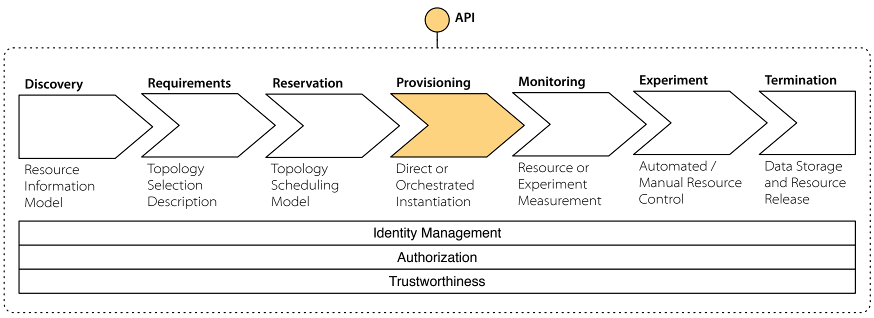


Figure 2.16: Provisioning of the selected topology in the experiment life cycle

Overview As shown in Figure 2.16, resources that have been discovered, selected and reserved within a slice are provisioned in the next phase at a given time. These instances of resources are called *slivers*, as indicated with the letter *S* in Figure 2.17. Depending on the type of resource, the provisioning phase could involve a high number of different procedures, including the configuration, e.g. allowing access by the requested user, and creating a possible instantiation. Information on how to access and use the resource has to be returned to the enquirer, and could include information about dynamically selected

IP addresses, SSH login credentials, and API endpoint information and other utilization related details.

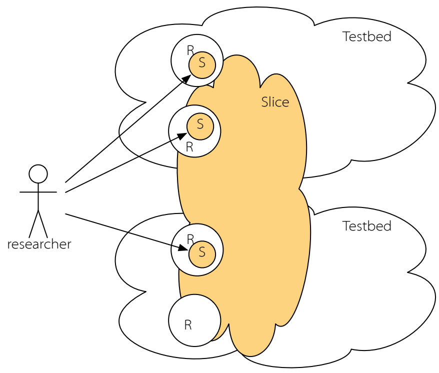


Figure 2.17: The concept of slices, slivers (S), resources (R) and testbeds

Federation Challenges

Depending on the resources requested, potential dependencies between the heterogeneous resources may have to be resolved and provisioning must be orchestrated between the different administrative domains. The IETF calls this Service Function Chaining (SFC), which involves not only the order of instantiation, but also forwarding arbitrary configuration parameters and the configuration of the network between the services. Further, depending on the use case, this might also include the dynamic modification of the number of instantiated resources within the runtime of an experiment.

Technologies

This phase is part of the SFA workflow and protocol specification and takes information from the requirement and reservation phases into account. For experiments that use resources from multiple testbeds, the relevant orchestration mechanism has to contact each infrastructure separately.

2.4.5 Resource Monitoring

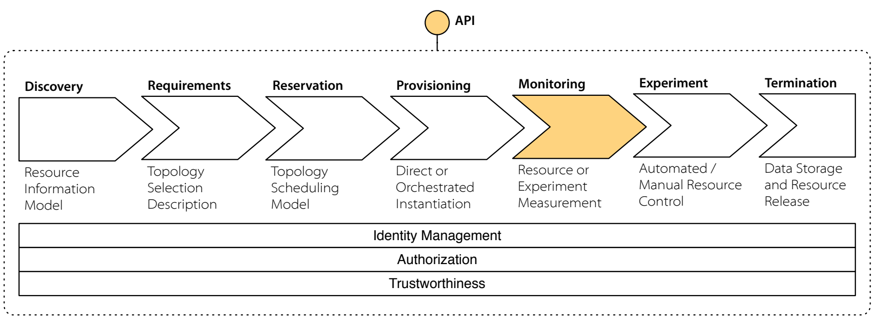


Figure 2.18: Monitoring of the selected topology in the experiment life cycle

Overview

Given that the selected resources have been provisioned (cf. Figure 2.18), monitoring data about these slivers, the potential resources to be selected and the infrastructure where the resources are located are of interest for several use cases. Therefore, three different levels of monitoring capabilities can be identified. First, *experiment measurements* relate

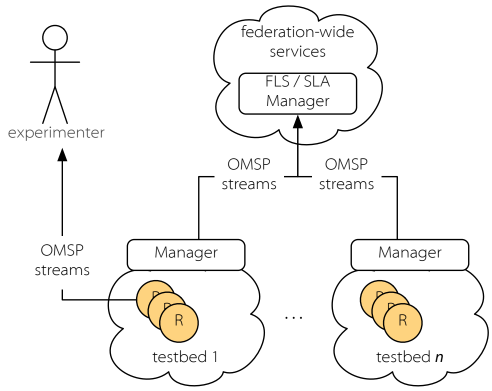


Figure 2.19: OMSP streams for experiments and FLS/SLA monitoring

to current resource utilization and present sliver-specific monitoring information that can be exported directly to the user. This might include measurements regarding network characteristics of a Virtual Machine (VM) that the experimenter is using. Second, *infrastructure monitoring* exports information about the resources that the provisioned sliver is associated with. One example is the load of the host on which a VIM is running. Finally, *facility monitoring* distills the overall status of resources within a testbed. Both, facility and infrastructure monitoring can be used to map unbound requests to eligible resources and infrastructures, to elastically orchestrate resources, and to study SLA compliancy.

Federation Challenges

Analog to the description of resources in the discovery and selection phase, monitoring information about resources can be highly heterogeneous in nature. Each facility and each resource might describe various monitoring metrics differently and the level of detail needed differs from use case to use case. Further, monitoring information is are subject to change frequently and the amount of data collected within a federation can introduce issues with respect to manageability of data.

Technologies

While basic status information about available resources and provisioned slivers can be accessed using SFA, OMSP was introduced to export real-time measurement streams from a testbed to a sink, such as the Network Measurement Virtual Observatory (nmVO) [p159]. The TCP-socket-based protocol supports text and binary serializations and allows the definition of arbitrary schema as Comma Separated Values (CSV). After the TCP session has been initiated and the schema definition has been pushed, a stream of monitoring updates are send from the client to one or multiple servers. As shown in Figure 2.19, the streams can be transported directly to the experimenter or to federation-wide services for First Level Support (FLS) and SLA monitoring. The sources from which information is exported can be general purpose monitoring systems, such as Zabbix, as, for example, used within the Building Service Testbeds on FIRE (BonFIRE) [p121] [p111] project; or context-specific systems, such as the TopHat Dedicated Measurement Infrastructure (TDMI) [p28], used within PLE.

2.4.6 Resource Control

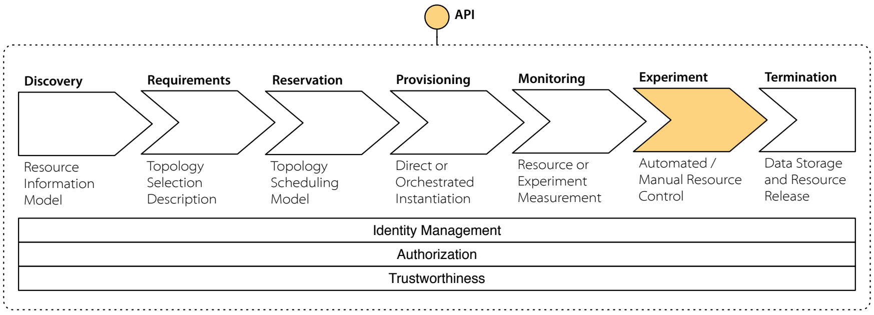


Figure 2.20: Control of the selected topology in the experiment life cycle

Overview

After the potential monitoring measures have been setup, resource control is the next step within the experiment life cycle, as depicted in Figure 2.20. A common procedure is to use SSH to login into a node and to execute the commands needed for the experiment. However, in order to gain scientific knowledge, mechanisms for reproducibility and automation are needed.

Federation Challenges

Again, within a federated environment for FI experimentation it is a challenge to support control of heterogeneous resources that in turn offer different APIs and functionalities. For example, not every resource offers an SSH login and configuration of parameters might differ between resources and testbeds.

Technologies

For these purposes, a number of ECs have been implemented, such as OMF, NEPI or FCI. Within GENI and FIRE FRCP was adopted as a common protocol for controlling resources within experiments. The tools used for control mask the heterogeneity and the complexity within resource control by enabling users to create their experiments using a single description language, such as OEDL. This, however, introduces further challenges as the provision and control phases are implemented by different APIs using disparate data models.

2.4.7 Resource Release

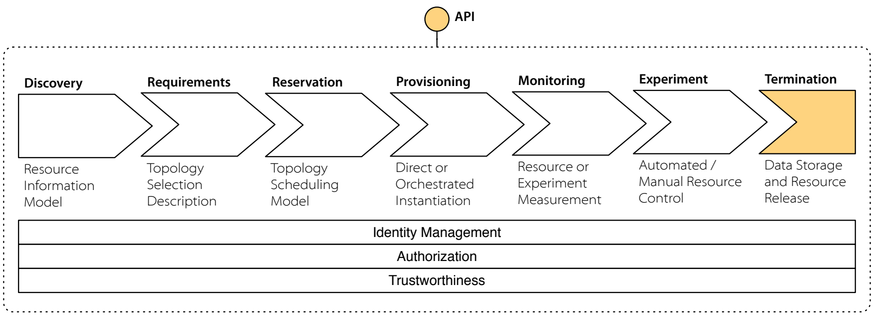


Figure 2.21: Termination as the last step in the experiment life cycle

Overview

As the final step in the experiment life cycle (cf. Figure 2.21), resources have to be released after the lifetime of an experiment is over. This can additionally include ensuring that data collected within the experiment remains available for further scientific evaluation, such as measurements, disk images or other metainformation.

Federation Challenges As information in a federated environment can be distributed across multiple sites and large amounts of data can be produced, appropriate strategies are needed to assign version information, describe the data and make sure it is available in the long term. Further, after use of resources from different sites, depending on the context, accounting information might need to be shared between the infrastructures based on the aforementioned monitoring data.

Technologies Within GENI and FIRE, the release of resources and information about the duration of an experiment are part of the SFA APIs and RSpec definitions. Measurements are transported via OMSP to a data sink at the experimenter's premises or other available databases. Ensuring availability of other types of information, such as modified disk images or the exchange of accounting information, has not been investigated to a large extent within the field of FI experimentation.

2.4.8 Authentication and Authorization

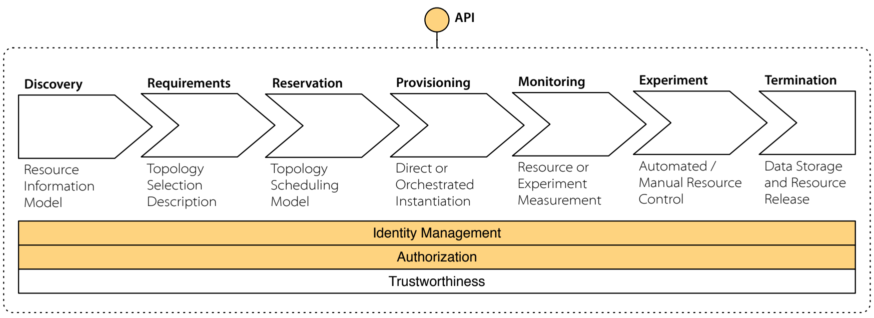


Figure 2.22: AuthN and AuthZ in the experiment life cycle

Overview Most of the phases of the experiment life cycle described above have to support AuthN and AuthZ, as depicted in Figure 2.22. Based on the identification provided not only might a method call be approved or denied, but the information exchanged, such as the list of available resources, can also be influenced. AuthZ is of importance in order to enforce access rights to resources, e.g., who is allowed to create or access which resource.

Federation Challenges Usually each testbed uses a specific approach for user AuthN and AuthZ. In federated independent experimentation facilities, identity AuthN is required in order to confirm the identity of users or their claims. These identification claims might not even be attached to a particular user but rather to a requested experimentation topology (*slice*) instead, on which an AuthZ decision can be made. Further, policies that assign roles or attributes within one domain, might have no meaning or an other meaning in another domain. Also, to allow handovers between protocols, each API has to support the same type of credentials.

Technologies Within GENI and FIRE, each testbed can act as an Identity Provider (IdP), e.g. using a local Lightweight Directory Access Protocol (LDAP) server; or federation-wide IdPs can be provided for convenience. Following public-key cryptography mechanisms, messages are signed and encrypted on the transport level (cf. Figure 2.23). AuthZ decisions are made at each testbed locally and are based on the attributes assigned to

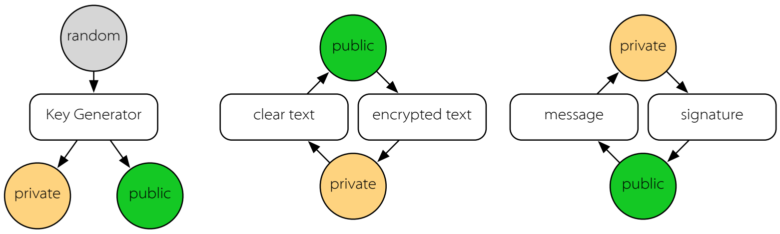


Figure 2.23: Public key mechanism overview

credentials by an IdP. The concept basically follows the XACML reference architecture (cf. Figure 2.24) and extends Attributed Based Access Control (ABAC) [p168, p245] mechanisms by introducing, for example, Speaksfor [t15] delegations. Standard AuthZ mechanisms with built-in federated AuthZ delegation support, however, such as SAML assertions, OAuth [t28], or JSON Web Tokens have not yet been adopted. The same holds true with federated AuthN protocols, such as Shibboleth [t62], OpenID [p195] or Persona [p221].

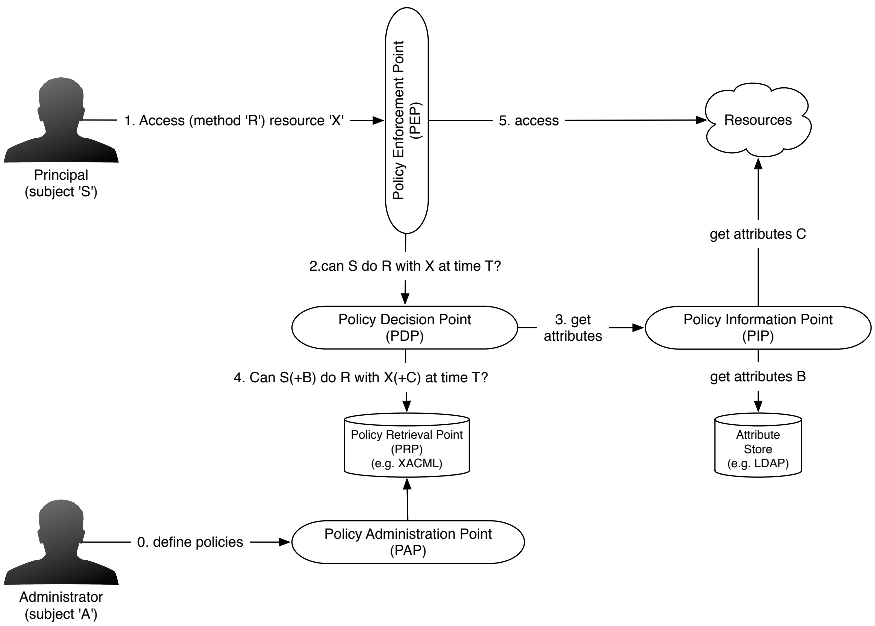


Figure 2.24: XCAML reference architecture (based on [t61])

2.4.9 Trustworthiness

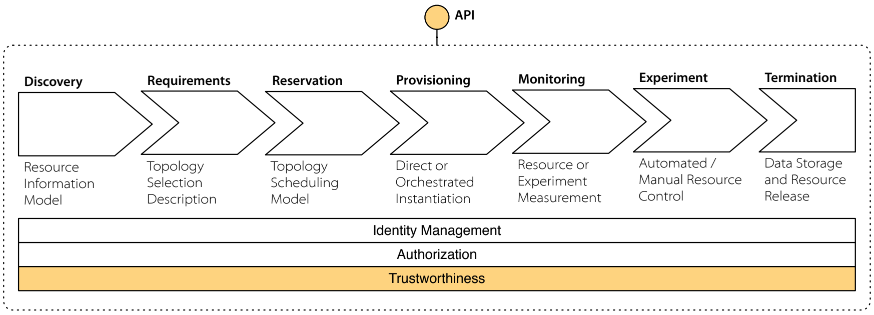


Figure 2.25: Trustworthiness in the experiment life cycle

Overview As indicated in Figure 2.25, trust in the reliability, truthfulness, and ability of both an infrastructure and its user is a foundational element of remote access to resources in general and experimentation in particular. Relevant directives are described in policies, which are "rules governing the choices in behavior of a system" [t20]. These include access and pricing policies, resource placement policies, or reservation policies, as well as SLAs, Operational Level Agreements (OLAs), or Life Cycle Agreements (LCAs). Policies are highly interweaved with the information and processes described above. For example, facility monitoring data can be used to assess the reputation of an infrastructure.

Federation Challenges The trust of the overall federation relies on the trust of its members and is subject of change while members join, leave and change their behavior. Manual negotiations may be required to sign off on agreements on policies between facilities and the federation, and to renegotiate terms of existing agreements. By contrast, compliance verification during operation needs automated mechanisms and involves information originating from different phases of the life cycle.

Technologies While approaches such as the Web Service Level Agreement (WSLA), the Service Level Agreement Language (SLAng) [p144] and WSAG have already been developed in the Grid and Web service contexts, technologies to support automated trustworthiness management in federated experimentation facilities had not been investigated in detail in the past. However, in current work basic trust is established by operating federation-wide PKIs using X.509 certificates, as shown in Figure 2.26. Further, within the Fed4FIRE project, additional measures have been implemented involving mainly Quality of Experience (QoE) rating mechanisms based on user feedback and facility monitoring information about the availability of offered services transported via OMSP for a FLS.

2.5 Conclusion

DRM This chapter provided a brief overview of the field of distributed resource management as an introduction to the research field of management of infrastructures in federated environments. In particular, the evolution from Metacomputing, via Grid Computing to Cloud Computing and Intercloud Computing paradigms has been presented. This further included a description of the underlying concepts and the related standardization bodies.

Testbeds To further narrow down the research focus, the specific field of application of testbed federation was outlined. The most important initiatives, GENI, FIRE, FI-PPP and EIT,

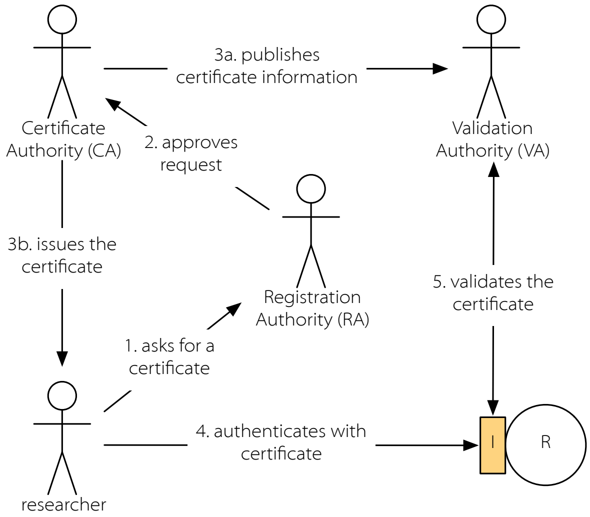


Figure 2.26: Public key infrastructure overview (based on [p2])

along with their approaches, technologies and projects were characterized. They were further put into context within the relevant worldwide community.

Life Cycle

Based on this, more precise details about the required workflow for reproducible, scientific experimental evaluations were provided. The life cycle includes steps for resource description, discovery, reservation, orchestration, provisioning, monitoring, and release, as well as experiment control and measurement. Additionally, the relevant AuthN, AuthZ and trustworthiness measures have to be taken into account.

Next

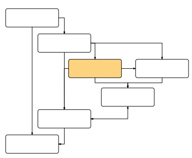
It was highlighted that, in a federated environment, each phase of this cycle embodies a number of research questions on its own. A common issue throughout the life cycle is the use of multiple APIs and heterogeneous data models that impede interoperability and protocol handovers. Given the information provided in this section, the next chapter elaborates on the relevant requirements from different stakeholder perspectives, in order to build a basis for the proposed approaches designed within the thesis to overcome these issues.

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 Introduction

Overview



The previous chapter provided an overview of the state of the art in the field of distributed resource management with a focus on federated FI testbeds and the experimentation life cycle, including relevant standards, initiatives, projects and research activities. Based on this overview and in order to elaborate on the research issues in the thesis, this chapter describes the requirements from the perspective of the three main stakeholders depicted in Figure 3.1: the user, the resource provider and the federation operator. The structure of the chapter reflects these actors and further follows the life cycle as previously described. While mainly functional requirements are highlighted, further nonfunctional aspects, such as legal arrangements, economic assessments, sustainability aspects or performance-related characteristics, may apply as well.

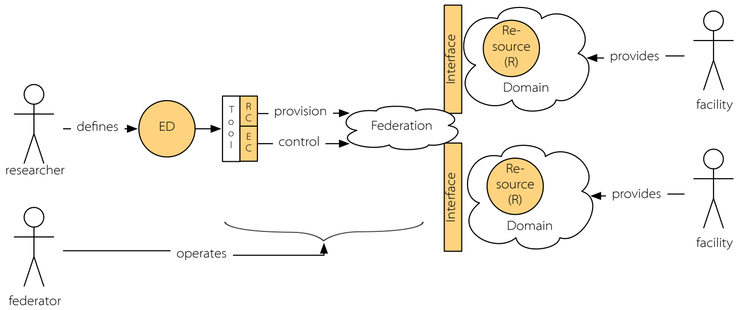


Figure 3.1: More detailed overview of the two-sided market

Structure of Research

As shown in Figure 3.2, this chapter provides the requirements for the design decisions of the subsequent work. Parts of this work have been published before in [a16, a18]. The requirements identified here and open issues are derived from both the analysis in Chapter 2 and experience gained within several related research projects, including OpenLab, Fed4FIRE, CI-FIRE and TRESCIMO.

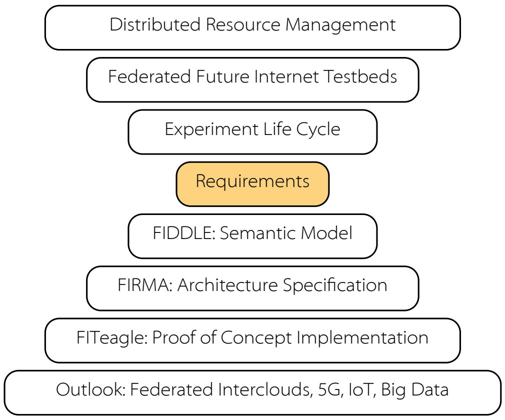


Figure 3.2: Placement of the requirement section in the structure of research

3.2 Infrastructure User

Table 3.1

The most important stakeholder and use-case provider is the client of the federation. The main objective of the user is to control and observe resources in a reproducible, managed manner in each phase of the experiment life cycle. Easy migration and portability from one testbed to another should be supported to validate reproducibility and to avoid a vendor / testbed lock-in effect. Around this, a number of requirements appear that are driving aspects of managing a federation. In Table 3.1 a short description these prerequisites are given.

Table 3.1: Requirements from a user perspective

#	Description
U1: Discovery	Locating available resources within a federation as a whole that meet given requirements is the first and most crucial requirement from a user perspective. Given the context of FI experimentation, the relevant description and discovery mechanism should support a wide range of heterogeneous resources and be extensible enough to integrate new ones. Information about the resource should include details about its unique type, how this type relates to others (e.g., equivalence, disjointedness and inheritance), what dependencies exist, where it is located, which parts it is composed of, information about its availability, or whether it is a virtual or physical resource.
U2: Selection	Based on the available information on existing resources within a federation, a user has to be able to specify a topology of required resources during an experiment. In its simplest form, this includes required nodes and links between them, and can further comprise configuration parameters, dependencies, storage or software libraries. The selection may only include abstract resource types and required attributes, which can later be mapped to concrete resource instances, which can further take information for service chaining into account.
U3: Reservation	Depending on their type, resources should be allocable with respect to capacity (e.g. bandwidth), amount of utilization (e.g. transferred data) or for a given time period (e.g. future reservations). Here, the reservation start or end time can be fixed or flexible and the resource exclusiveness may vary from single entities to potentially unlimited virtualized instances.
U4: Provisioning	The actual implementation of the instantiation of the resources should be supported in order to allow users access to the designated topology. This might occur via directed API calls at the relevant facility or through an orchestration mechanism that involves a number of testbeds. In this phase, a mapping from the requested requirements to the actual resources may occur.
U5: Monitoring	The observation phase can be divided into three different areas that might be supported depending on the use case in question. First, measurements that result from the experiment conducted should be transferred to or stored for the user. Second, information about the underlying infrastructure on which the experiment is executed should available. Third, the overall status of the facility that hosts the experiment, or a specific part of it, might be of interest, e.g. for FLS.
U6: Control	Users need means to control the provisioned topology within the lifetime of the experiment. This may range from simple SSH access to automated event- and time-based workflow management.

#	Description
U7: Termination	After conducting the experiment, the provisioned resource should be released. This might include the permanent storage of experiment results or setup parameters.
U8: AuthN	Users and testbeds within a federation should be authenticated. Either each testbed acts as an IdP, or the federation itself or a trusted third party carries out this functionality. The role of the IdP is both to account for the identity of the requesting user and to codify security assertions for the subject. This allows access to global resources without additional subscriptions.
U9: AuthZ	Based on the trusted authentication of the requester and its related assertions, communication should be authorized following the rules and policies in place. This holds true for each part of the experiment life cycle and may occur at the testbed or federation level.
U10: Trust	Along with the trusted authentication of users and testbeds, bilateral confidence in the federation services might be built upon two aspects. First, SLAs might be negotiated between the users and the federation, ensuring offered services meet specific KPIs. Second, OLAs could be defined in an analog fashion between the federation and the resource providers.
U11: API	All key functionalities should be supported by one or multiple common interfaces, agreed upon before federation. If more than one interface is involved, appropriate handover, AuthN and AuthZ mechanisms should be supported.

3.3 Infrastructure Provider

Table 3.2 The main objective of an infrastructure operator is to provide its resources to the user. Depending on the used business or operational mode, this might include participation in one or multiple federations, specific SLAs and pricing models. The resulting requirements are given in Table 3.2.

Table 3.2: Requirements from a provider perspective

#	Description
P1: Discovery	The resources available within a testbed should be described and published, includes their types, properties and dependencies. Full internal information may not be disclosed, but instead an abstracted representation or user-tailored filtering might be needed. To avoid overheads and inconsistencies, resources should be described only once, independently of the number of federations the testbed is involved in and the description should cover information needed for each phase of the experiment life cycle.

#	Description
P2: Selection	The published resource description should allow users of the testbed to describe their requirements in an abstract manner to allow mapping to concrete resource instances in later phases. Further, federation-wide experiment topologies should be supported, i.e., stitching network connections between different facilities.
P3: Reservation	Booking information about resources should be stored and handled within the testbed management system. While this information can also be stored centrally in a federation-wide service, this would impede participation in multiple federations, and local resource reservation. Further, this data should be exported to schedulers to allow federation-wide planning.
P4: Provisioning	The information provided during reservation should allow the testbed management framework to instantiate the allocated resources for the user requesting them. This includes providing access to physical resources and starting virtual ones, as well as configurations, such as network or monitoring setups.
P5: Monitoring	The testbed should support the export of the aforementioned types of monitoring information. While experiment measurement itself is the responsibility of the user, the resource provider should provide necessary tools for this purpose that are common to the given federation. Depending on the resource, the export of infrastructure monitoring information should be supported. Further, for a federation-wide FLS, generic information about the testbed status has to be exported.
P6: Control	Based on the description exported for the discovery phase, and the information provided in the reservation and provisioning phases, resources within a testbed might be controllable after they have been provisioned. This includes access for external users by offering resource-specific APIs, or integration for federation-wide control protocols.
P7: Termination	It should be possible to release resources, in order to make them available to other experimenters in the federation. Based on information about utilization and duration from the provisioning phase, invoices or other reports could be generated.
P8: AuthN	Within a federation, users might authenticate themselves using external IdPs, since they are unknown to the local testbed. These IdPs have to be trusted and a facility might also act as an IdP. Further, the infrastructure has to prove its identity to the user.
P9: AuthZ	Access to resources within a testbed should not be provided to unauthenticated users. Based on the trusted identity of a user and other attributes that have been provided, specific authorization policies take effect and may be synchronized with federation-wide rules.

#	Description
P10: Trust	To attract experimenters and to establish a solid ground for commercial offerings, SLAs should be negotiated between the user and the facility. This can be implemented in a static or dynamic manner and might range from soft best-effort services to hard KPIs with specified penalties in case of a violation. Further, arrangements within the whole federation might occur. These OLAs are valid between the testbed, and the federation including their members; and have to be synchronized with the SLA.
P11: API	An infrastructure communicates with users and the federation via APIs and each of the required functionalities have to be supported. Further, in the context of federated FI experimentation, multiple APIs have been established that each cover parts of the required functionalities. Therefore, a testbed has to support each API that is needed for a user in the federation to conduct an experiment.

3.4 Federation Operator

Table 3.3 The main objective of the federation is to facilitate the participation of its infrastructures, e.g. in large-scale experiments, and therefore contribute to their increased sustainability and usefulness. This can be shown by greater rate if utilization in the public-funded sector or by increased Return of Investment (RoI) in the commercial context. Requirements for the setup and operation of such a federation are given in Table 3.3.

Table 3.3: Requirements from a federation operator perspective

Area	Description
O1: Discovery	Users of a testbed federation are challenged by a large set of highly heterogeneous resources. In order to facilitate the discovery of required resources, descriptions published by each testbed should be expressive and as harmonized as possible. Given the autonomy of testbeds in such an alliance, linking and merging of information should be supported.
O2: Selection	The use of single testbeds by experimenters involves only one administrative domain. In a federated environment, however, the selection of resources from any administrative domain and the combination of resources from different domains at the same time have to be supported. This might also include exchange of resource descriptions between different facilities. Each infrastructure involved has to support and extract relevant information from the provided data structure.

#	**Description**
O3: Reservation	Resource reservation, in its simplest form an immediate reservation with no specified end time, should be supported across multiple testbeds. As each testbed is independent from the federation and each resource might have specific scheduling information, the local scheduling information for different resources has to be aligned. This concept could further be extended by providing federation-wide scheduling information.
O4: Provisioning	Based on the reservations conducted, it should be possible within the federation to allocate the relevant resources. Determined by their interdependencies, this might be performed in parallel or sequentially by the user client tool or by a federation-wide orchestrator.
O5: Monitoring	Similar to the discovery and reservation phase, monitoring information about the heterogeneous resources within a federation should follow compatible, linkable formats. Additionally, for FLS services, it might be necessary to collect and merge status information from each testbed.
O6: Control	For federation-wide control of the provisioned resources, the same APIs and compatible models have to be exposed for each resource involved. The federation should facilitate this by recommending interfaces, models and appropriate tools.
O7: Termination	A coordinated release of resources within the federation and merging information about the utilization of resources within several testbeds should be supported.
O8: AuthN	Mechanisms for distributed authentication are one of the most commonly used concepts within federations. The establishment of and the trust between IdPs should be facilitated in order to allow access to the service offerings within a federation.
O9: AuthZ	Based on the aforementioned federated identity management, authorization information about users and resources should be harmonized and streamlined within the federation to allow the construction and evaluation of appropriate access rules.
O10: Trust	Trust in a federation is based on the trust in each of its testbeds. Therefore, the federation should enable the establishment of global OLAs between participating infrastructures, closely monitor the status of the resources provided and facilitate feedback mechanisms for users.
O11: API	Since a user communicates with a number of administratively independent infrastructures in a federated environment, the set of APIs used within the federation to cover the previously described experiment life cycle functionalities should be enforced and limited.

3.5 Conclusion

Summary This chapter enumerated a number of requirements and challenges with respect to the distinct phases of the experiment life cycle that are imposed by conducting experimentation across multiple administrative domains for FI research as seen from the perspective of different stakeholders. Based on these requirements and the state of the art described in Chapter 2, a number of open research questions can be identified. Two major underlying issues arise that are fundamental for enabling interoperable resource utilization between federated infrastructures in general and federated testbeds in particular.

Issue: Protocols First, different phases of the experiment life cycle are covered by various APIs that further differ from federation to federation. Therefore, each infrastructure has to support a number of protocols and proper handovers in parallel (P11), users rely on different tools (U11), and federations have to ensure compatibility across the alliance (O11).

Issue: Models Second, while a selected resource of an experiment used within each of the APIs is always the same, each API and federation context uses different information and data models. As a result, the use of heterogeneous data structures impedes interoperability within a federation, increases complexity for infrastructures and developers, and complicates the use of multiple tools from a user perspective. The selection of proper models is relevant for publishing information about infrastructure resources (P1), the discovery of resources from the users view (U1), and the harmonization of resource descriptions on the federation level (O1). Proper models consequently influence each phase of the life cycle, such as linking monitoring information to resources allocated by a specific user.

Approach Both issues not only increase the administration and maintenance costs within a testbed, but also aggravates handover and interoperation between infrastructures, tools and federations. This is contrary to the major goal to conduct the experiment life cycle in an automated and reproducible manner across multiple administrative domains. Further, given that these interfaces and models are established in their relevant communities and new ones are continuously developed in related contexts, the situation tends to get more complex over time.

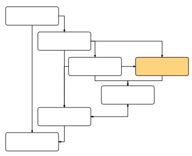
Next These insights build the argumentative foundation for the design of the two subsequent contributions: a canonical formal information model is developed in Chapter 4, and a protocol-agnostic architecture, using this model is specified in Chapter 5.

Chapter 4

Design and Specification of the FIDDLE Ontology

4.1 Introduction

Overview



As described in Chapters 2 and 3, each phase of the aforementioned experiment life cycle yields a range of interesting research issues in a federated environment. Of importance to a federation is the exchange of information about resources between participating testbeds and across protocols. As indicated, it directly influences all life cycle phases, since information that refers to the same resource has to be encoded and transported using different serializations and protocols. As a result, the main objective of this chapter is to introduce a formal Information Model (IM) called Federated Infrastructure Description and Discovery Language

(FIDDLE) [a20]. FIDDLE is a model, intended to form a foundation for a coherent life cycle management across federated infrastructures for FI experimentation and beyond.

Structure of Research As shown in Figure 4.1, the model developed in this chapter depicts the first and underlying foundation for the design and implementation introduced later in the thesis. Parts of the work presented here have been published before in [a11, a20, a24] and the ontology specification acted as a starting point for broader standardization for the semantic management of federated ICT infrastructures, developed within the W3C Federated Infrastructures Community Group.

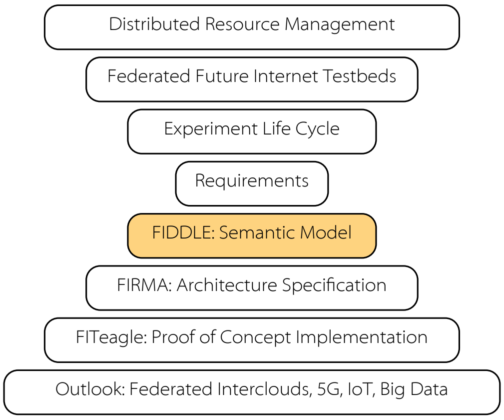


Figure 4.1: Placement of the information model in the structure of research

4.2 State of the Art

Overview To be able to assess possible solutions and to understand the aforementioned challenges, it is important to elaborate on information modeling and processing in general and on the difference between an Object Model (OM), a Syntax or Serialization, a Data Model (DM), an IM and a Semantic Model (SM) in particular. This distinction is often not well understood in the ICT sector, terminology and concepts are often misused. As a result, theses concepts are not properly considered when defining information-exchange architectures between independent domains. The general difference between Data, Information, Knowledge and Wisdom has been a topic of information science for many years [p1, p108] and this thesis the terminology and argumentation from RFC 3444 [t57] is used. The corresponding relationships between the different models are highlighted in Figure 4.2. Starting from the bottom, several models are needed to map real-world concepts in a form that allows them to be processed by machines. These layers are further described and distinguished in the following sections together with related technologies currently used in the ICT sector.

4.2.1 Object and Data Models

Object Models In order to access, manipulate or store information within a computer program, data is often interpreted using functional code. Depending on the expected type of input, two main approaches can be distinguished. The first type of approach involves event-driving parsing mechanisms, such as the Simple API for XML (SAX), which analyze streams of data for the occurrence of specific elements, allowing very large data sets

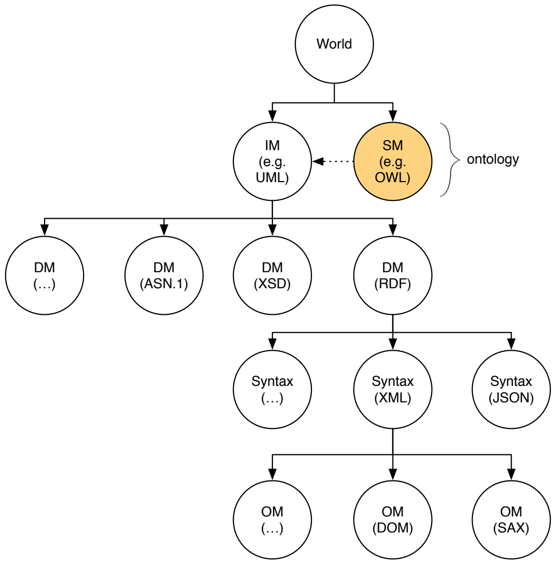


Figure 4.2: Relationship between models and syntax (based on [t57, p187])

to be processed. In the second type of approach, the complete data structure can be mapped onto a Document Object Model (DOM) [t49], using implementations such as the Java Architecture for XML Binding (JAXB) or Apache XML Binding (XMLBeans), to enable directed navigation within the data structure.

Serializations

Parsers that interpret data to build such OMs are specialized for a specific serialization, also called syntax or language. Common examples are XML, JSON, the Abstract Syntax Notation One (ASN.1), Notation 3 (N3) [t8], Turtle (TTL) [t6], CSV, the YAML Ain't Markup Language (YAML), and the Yet Another Next Generation (YANG) [t11] format used in the Network Configuration Protocol (NETCONF) [t23]. Often a specific syntax is bound to a given DM and it is therefore sometimes problematic to distinguish between the two.

Data Models

A DM is a specification, often serialization dependent, how to organize information in a machine-processable way. Lists, trees or graphs can be used to define relationships between elements. In [t89], DMs are further described as "a mapping of the contents of an information model into a form that is specific to a particular type of data store or repository. A 'data model' is basically the rendering of an information model according to a specific set of mechanisms for representing, organizing, storing and handling data." Examples are CSV, XSD, the Resource Description Framework (RDF) [t19], Management Information Base (MIB) modules using the Structure of Management Information (SMI) [t44] or Guidelines for the Definition of Managed Objects (GDMO) [t87] syntax, and the Policy Information Base (PIB) following the Structure of Policy Provisioning Information (SPPI) [t43] syntax.

General Issues

In many cases middleware APIs concentrate on the exchange of semistructured tree DMs, often serialized as XML or JSON, which are then transformed to programming language dependent OMs. Such an approach is introducing interoperability issues between federated domains, as choosing a certain data model imposes limitations on how information can be expressed. For example, as briefly described in Section 2.4, resources are currently described using XML-based RSpecs in the discovery, selection, reservation, provisioning and termination phases using SFA; for the control phase either a JSON or XML serialization is used within FRCP; and, for monitoring information, user-defined triplets are communicated through OMSP. Each approach has the means to define markups that provide a uniform framework for the exchange of data and metadata between applications. However, the chosen list- and tree-based data structures do not define explicit semantics. In particular, there is no explicit meaning associated with the nesting of tags and the structures have limited support for expressing relationships (dependency graphs). Furthermore, the vocabulary of the tags and their allowed combinations is not fixed and, for example, each GENI RSpec extension follows its own approach. Listing 4.1 illustrates the issue of structure-implied semantics. Although both examples contain the same information, a different nesting has been chosen in each. The correct interpretation of the information serialized in these examples has to be implemented in the functional code that interprets the fragment.

Federation Issues Within a federated environment, these challenges become even more aggravated as every testbed publishes its resources in slightly different schema documents. As stated in [p120], the problem of bringing together heterogeneous and distributed information systems is known as the interoperability problem [p235]. In general, XML-encoded documents cannot be arbitrarily combined with other XML trees in a flexible manner [p190]. In particular, it is noted in [p116] that the simple union of two tree structures is not a tree anymore, so that combining n resource catalogs would require $O(n^2)$ translators. Even if two XML files refer to the same resource, it is likely the relevant information is in different locations in the tree and additional choices must be made to obtain a new well-formed XML document (cf. Listing 4.1 again). The composition of such data in a federated environment is rather complex and involves a significant amount of functional code. One example is the characterization of relationships between different resource descriptions, such as same-as relations, that cannot be encoded in plain XML. Further, the discovery of resources, which is one of the main use cases, needs implicit knowledge to formulate adequate XML Path Language (XPath) [t7] queries.

Listing 4.1: Same semantics but different syntax

```
<resource name="PC-133">
  <administrator>Alexander Willner</administrator>
</resource>
<administrator name="Alexander␣Willner">
  <resource>PC-133</resource>
</administrator>
```

4.2.2 Semantic Models

Intro Related schema-based integration issues were analyzed as early as 1986 [p13] and problems of heterogeneous data integration within distributed databases were the subject of research starting in 1991 [p133]. Therefore, the goal is to identify better ways to describe heterogeneous resources and to discover, link and merge related information within federated facilities.

IMs One approach to achieve disambiguation is to specify IMs. An IM itself is defined in RFC 3198 [t89] as “an abstraction and representation of the entities in a managed environment, their properties, attributes and operations, and the way that they relate to each other. It is independent of any specific repository, software usage, protocol, or platform." Following this definition, an IM can be represented by an unstructured, semistructured or highly structured document. Plain text can describe an IM and, as a matter of fact, in particular, in SDOs, is the most common way to describe world concepts and standards. Examples are the RFC 3945 [t39] that defines the Generalized MPLS (GMPLS) architecture, and the ITU-T Generic Functional Architecture of Transport Networks (G.805) recommendation. To facilitate the implementation of standard-compliant code, some specifications, such as OASIS TOSCA, elaborate on semistructured DMs with an authoritative text version of the related IM.

Nonsemantic Approaches

Other standards, e.g. in NIST, include more formal specifications in the form of Unified Modelling Language (UML) [p27] diagrams, which allow, for example, the automatic generation of functional code. The DMTF has mapped this approach to information modeling by specifying the Common Information Model (CIM) [t77]. CIM uses a metamodel, similar to the Object Management Group (OMG) UML, to specify real world, partly technology-specific concepts, classes and relationships to allow the distributed management of IT systems. One application of CIM, the Autonomic Communications Forum (ACF) DEN-ng model, based on CIM with an LDAP syntax, is a possible model for FI network management [p211]. Derived from version 3.5 of DEN-ng, the TMF Shared Information & Data Model (SID) uses UML to model management information in the telecommunications domain and is used in New Generation Operations Systems and Software (NGOSS) standards, such as the eTOM framework.

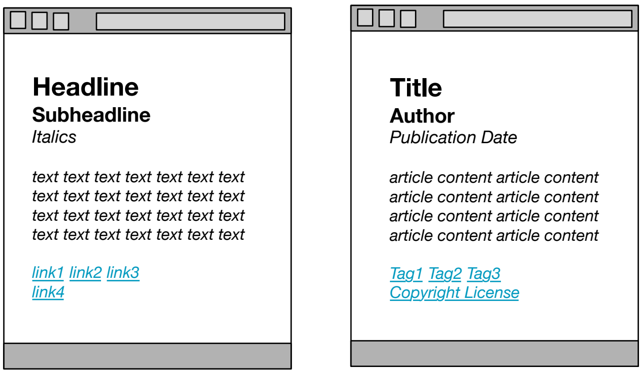


Figure 4.3: Syntactic and semantic interpretation of markup [t1]

Issue

UML and CIM models seem to be appropriate candidates to exchange information about heterogeneous resources in an implementation-agnostic manner, and are therefore applicable within the context of this thesis. However, as noted in 1999 by A. Sheth et al, interoperability challenges in federated environments should be approached by elevating abstraction to the semantic level and exchanging SMs [p174, p203]. In Figure 4.4 relevant categories of interoperability are depicted, whereas the abstraction increases from the middle to the outside ring. Semantic interoperability is positioned on a higher level, while organizational interoperability are ensured within a federation. The OMG noted the lack of a reliable set of semantics and a model theory in [t52] and it was further shown in [p16] and [p190] that CIM can't be converted into such a SM.

Semantics

The *New Oxford Dictionary of English* [p178] defines semantics as "the branch of linguistics and logic concerned with meaning. The two main areas are logical semantics, concerned with matters such as sense and reference and presupposition and implication, and lexical semantics, concerned with the analysis of word meanings and relations

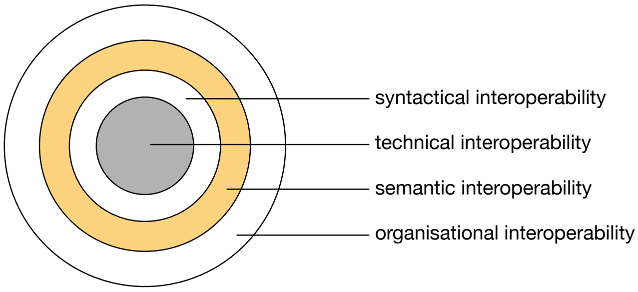


Figure 4.4: Different levels of interoperability (based on [t72])

between them." Mapped to the ICT domain, a semantic model is an "explicit meaning of concepts and relationships between models. This meaning is defined in a machine-readable format, thus, making it accessible to both software management components and humans." [p187] Adopting semantics enables software tools to understand the meaning of the resource description. Semantics allow the use of automated reasoners to autonomously infer knowledge, such as the relationships of different resources. Reasoning further allows the evaluation of axioms to identify the compatibility of two models, which is a crucial requirement in interoperable information exchange between federated testbeds. Semantics allows information to be linked, combined, validated and queried using only formal methods instead of functional code. It enables compatibility problems to be solved without having to change existing models by mapping information to a common model. The foundation is a formal representation of information that computers are capable of processing. Figure 4.3 shows how a markup language can not only highlight the structure of data, but also the intended unique meaning, i.e. the semantics.

Ontologies Such a formal definition of knowledge within a specific domain is called an ontology. An ontology is defined as "the branch of metaphysics dealing with the nature of being." [p178] Generally in the ICT domain the word has a specialized meaning, it is a "formal, explicit specification of a shared conceptualization" [p105]. In other words, in ICT an ontology is a shared, formalized vocabulary regarding individuals (instances), classes (concepts), attributes, and relations for a given use case, eliminating conceptual and terminological mismatches [p70, p71, p149, p198, p200]. To distinguish ontologies from related terms, they are defined as follows: a vocabulary is a list of unambiguous explicit terms, a taxonomy organizes such a list into a hierarchical structure, a thesaurus then associates relationships to build a networked vocabulary, and, finally, a metamodel contains rules on how to construct models within a domain. Taxonomical information about concepts and their structural dependencies are denoted as the Terminology-Box (T-Box). Assertional information about concrete instances of concepts are referred as the Assertion-Box (A-Box). Generally, ontologies follow an open-world assumption (OWA) [p129], i.e. statements which cannot be proven are not necessarily false as they can be extended by other ontologies. An example for a language that follows a closed world assumption (CWA) [p128] is the Structured Query Language (SQL) for relational databases.

Disadvantages While ontologies, i.e. semantic IMs, allow a machine-understandable abstraction from concrete data models and serializations and introduce the aforementioned benefits, their use is accompanied by disadvantages. First, the underlying concepts are often difficult to learn, and sometimes do not address an existing problem [m16]. Second, as

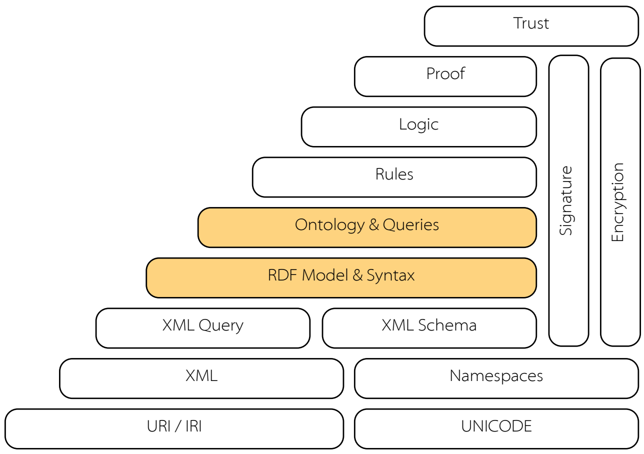


Figure 4.5: The Semantic Web layer cake (based on [m2])

noted in [p134, p169, p170, p235], merging, combining and linking ontologies can be an expensive process. While the former holds true for most new technologies that are introduced in a specific field, initial approaches for the latter are presented, for example, in [t51, p212]. Further, this thesis follows the scientific consensus that enabling interoperability across multiple administrative domains involving heterogeneous resources should not be addressed by focusing on the data-model level.

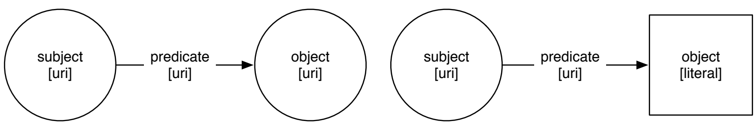
4.2.3 Semantic Web

Intro

Approaches that promise to successfully implement and communicate SMs are rooted in Semantic Web research (see Figure 4.5), whose development was motivated by Tim Berners-Lee. Its growth and the adoption of related technologies has seen an expansion of accompanying languages, protocols and standards.

RDF

At its core, the Resource Description Framework (RDF) [t19] represents triples (subject, predicate and object) in a graph-based data model, independent of a specific serialization. As shown in Figure 4.6, each subject and predicate is represented as a unique Uniform Resource Identifier (URI), whereas the object can either be a literal or a URI. Based on this, a labeled, directed and potentially cyclic graph can be created, which can be serialized in a number of formats. Common variants are RDF/XML, N-Triplets, N3, TTL, RDF in Attributes (RDFa) [t2] or JSON for Linked Data (JSON-LD) [t66]. The RDF specification further contains common concepts, such as blank nodes, types, containers and collections.



(a) With a resource as an object (b) With a literal as an object

Figure 4.6: Types of RDF triples

RDFS In order to describe simple ontologies, however, important vocabularies missing in the RDF specification have been defined with Resource Description Framework Schema (RDFS) [t21]. By introducing classes, ranges, and domains, along with subclassing properties and the accompanying transitive entailment rules, basic ontologies, such as taxonomies, can be defined. RDFS in particular, allows resources to be grouped by introducing the concepts *rdfs:Class* and *rdfs:Resource*, while each resource is specified to be a member of the latter. Along with the new transitive properties *rdfs:subClassOf* and *rdfs:subPropertyOf* it provides means to specify simple ontologies by assigning memberships to instances and specifying valid source and target concepts for properties using *rdfs:Domain* and *rdfs:Range*.

OWL In order to construct a larger set of more complex ontologies that require specifications like equivalency, symmetry or inverse relations, further rules and vocabularies are needed. With the Web Ontology Language (OWL) [t29], in its three variants OWL Lite, OWL DL and OWL Full (cf. Figure 4.7), a rich set of semantic annotations and rules have been introduced. While OWL Lite provides only a limited set of features to express semantic information, OWL Full allows complex expressions that make the language undecidable and therefore unsuitable for automated reasoning. OWL DL provides tangible expressiveness while still being computational. It allows the knowledge representation formalism Description Logics (DL) to be used. DL is a decidable subset of the First-Order Predicate Logic (FOL), used for defining general ontologies with all its constraints to concepts. The latest version of OWL (version 2) allows further modeling of semantic information by adding, for example, qualified number restrictions and increasing datatype expressiveness.

Reasoning Engines such as Pellet [p205] or HermiT [p202] can be used to reason over such semantic models and automatically extend the knowledge within the graph. For example, given that *rdfs:subClassOf* is defined as a transitive property and assuming that class *A* is *rdfs:subClassOf* class *B* which in turn is *rdfs:subClassOf* class *C*, then a reasoner would add *A rdfs:subClassOf C* to the graph. The rules that these engines evaluate can be defined by using e.g. the Semantic Web Rule Language (SWRL) [t31], a combination of OWL DL and OWL Lite with Datalog RuleML [p23].

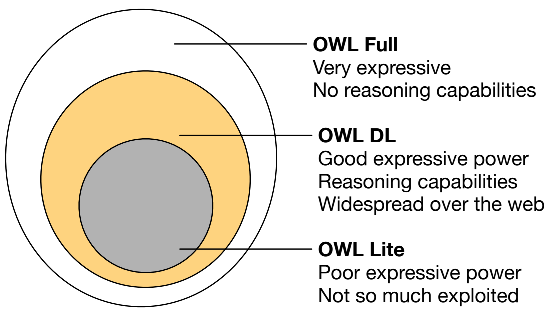


Figure 4.7: OWL sublanguages (based on [t40])

SPARQL Finally, once the information has been encoded into an annotated, directed multigraph, queries can be conducted over it. While in [p90] a number of different languages are compared, the SPARQL Protocol And RDF Query Language (SPARQL) [t5] has been established as the de facto standard. Following a syntax similar to SQL, its underlying concept is to match TTL-serialized graph patterns to expressions containing variables. As a result, SPARQL is more expressive and flexible than its counterpart

XML Query Language (XQuery) [t22] for XML trees. In its current version, SPARQL can further be used to create, update and delete subgraphs.

4.2.4 Linked Data

Linked Open Data

Based on these standardized and wildly adopted mechanisms for describing and accessing information, a number of RDF-encoded ontologies and data repositories have been defined. This expansion has enabled the implementation of a variety of tools and the spread of the concept of Linked Data [m1, p112]; or Linked Open Data (LOD), if the information is publicly available. These concepts describe how to link to nodes within semantically annotated graphs based on their Hyper Text Transfer Protocol (HTTP) accessible URIs to construct a globally distributed network of information. In particular, governmental institutions and the public sector are opening data, to be linked and allowing information facilitating the establishment of services, such as Intelligent Transport Systems (ITS) within Smart Cities. One example is the European Open Data set Joinup[1] [p88], which developed the RDF-based Data Catalog Vocabulary (DCAT) [t38] ontology.

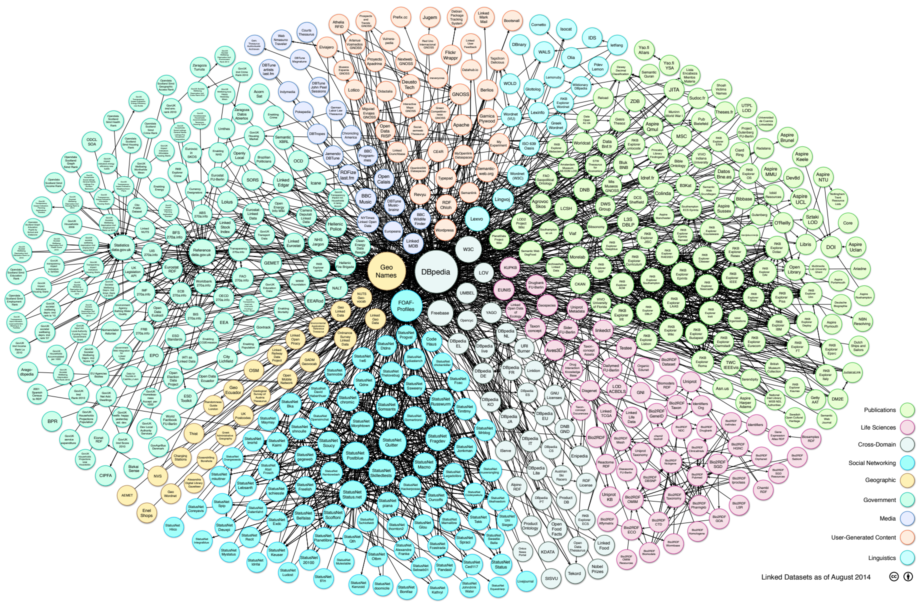


Figure 4.8: The Linking Open Data cloud diagram [t63]

Knowledge Graphs

A well known visualization of such a global linked graph is the Linking Open Data cloud diagram as shown in Figure 4.8. The most commonly used data sets shown here are DBpedia[2] [p8], with over one billion triples of structured information from Wikipedia; GeoNames[3], with 10 million geographical names; and several distributed social Web sources that use the Friend of a Friend (FOAF) vocabulary to describe relationships between people. Another large source is Freebase[4] [p24], a community-

[1]https://joinup.ec.europa.eu
[2]http://dbpedia.org
[3]http://geonames.org
[4]http://freebase.com

driven, collaborative knowledge base, that is part of Google's Knowledge Graph[5] [m13]. This graph not only enhances search results by providing structured information, but further enables services such as Google Now.

Big Data Given the size of these big data sets, research is being undertaken on the optimization of queries over graphs. As a result, a number of approaches have been published that try to efficiently optimize the process by partitioning data and decomposing and rewriting multiple SPARQL queries [p99, p118, p146]. Further, as most RDF databases rely internally on rational databases to store, index and query triples, approaches such as Microsoft's Graph Engine Trinity[6] [t65], used in Microsoft's Satori Knowledge Base, can achieve a speed-up of several orders of magnitude by using distributed in-memory key-value stores to natively save and query web-scale RDF data [p247]. While these concepts mainly consider single or multiple SPARQL queries, optimizations for continuous stream processing are considered by language extensions such as C-SPARQL [p12] or SPARQLstream [p25], which are further substantially optimized by approaches such as Continuous Query Evaluation over Linked Stream (CQELS) [p183].

Web Besides performance, research is ongoing on semantically annotate publicly available information on Web sites. While RDF can simply be embedded using the RDFa serialization and existing vocabularies such as FOAF, other ontologies are needed to express further information. For example, search engines companies Bing, Google, and Yahoo! have collaborated to provide a vocabulary called Schema.org[7]. It provides a shared collection of thematic schemas used to annotate websites in a common way to allow search engines to recognize, evaluate and display their semantics. By embedding such information as JSON-LD-serialized RDF graphs into emails, services like Google Mail can provide more detailed search results and context-related actions. Additionally, by applying Facebook's Open Graph Protocol (OGP) [], websites can be integrated into a global social graph and therefore be searched and used with the relevant social network. A commercial example that uses OGP- and Schema.org-encoded information embedded on sites is Apple's Siri personal assistant, which officially enriches its search results based on these graphs.

4.2.5 Semantic Management

Intro The mechanisms described in the previous sections can be adopted to manage sets of federated domains. As discussed in Section 2.2, the research field of Distributed Resource Management (DRM) covers, besides FI testbeds, a number of other concepts such as Grid Computing, Web services and Cloud Computing. In these related areas and other ICT domains such as IoT, a transition from syntactic to semantic interoperability for managing information and resources can already be observed.

Grid Computing For example, in the Grid Computing context the main purpose of the Grid Laboratory for a Uniform Environment (GLUE) [t3] schema, started 14 years ago, was to allow interoperability between Grid projects in the US and the EU by defining a schematic vocabulary with serializations in XML, LDAP and SQL [p63]. It specified a uniform description of Grid resources used for resource discovery and brokering. The lack of formalism, and, as a consequence, the missing means to reason over the information motivated the transition to a Semantic Open Grid Service Architecture (S-OGSA) [p49].

Web Services A similar movement can be observed in the context of Web services. First, syntactic service description languages were defined, such as the Web Services Descrip-

[5] http://google.com/insidesearch/features/search/knowledge.html
[6] http://research.microsoft.com/en-us/projects/trinity/
[7] http://schema.org

tion Language (WSDL) and Web Service Business Process Execution Language (WS-BPEL) [t34]. As described in [t12], semantic-enabled languages such as Web Service Modeling Language (WSML), Web Service Definition Language Semantics (WSDL-S), DARPA Agent Markup Language for Services (DAML-S) and Semantic Markup for Web Services (OWL-S) [t41] were developed. Semantic Web services are described using ontologies and, based on them, discovery systems can match requests using vocabularies and the introduced Semantic Web mechanisms. A survey of discovery systems used for Semantic Web services is given in [p196]. Further, work was conducted to merge existing Web services, like the Universal Description, Discovery and Integration (UDDI) concepts for service discovery, with semantic approaches [p139] using, for example, Semantic Annotations for WSDL and XML Schema (SAWSDL) [p138] or by specifying concepts around Linked Open Services (LOS) [p141]. To semantically enrich service descriptions based on Unified Service Description Language (USDL), Linked USDL[8] [p179] uses linked RDF graphs that are further planned to be integrated into TOSCA for automated cloud life cycle management [p36].

Cloud Computing

In the Cloud Computing domain and all its application sectors, it is inevitable that interoperability on a semantic level will be required sooner or later. Since 2008, research is conducted towards ontologies and tools for semantic cloud computing [p109, p151, p244]. For example, the current charter of the OASIS TOSCA technical committee[9] states that the definition of semantic models for cloud services are not within the scope of the current work. Further, only one of the 20 Intercloud proposals presented in [p104] is currently focusing on semantic resource information exchange. The proposal in question is the Intercloud architecture developed within the IEEE P2302 Working Group, which uses RDF-encoded graphs to describe and discover cloud resources based on the existing mOSAIC ontology. This ontology is a cloud-specific model for federated clouds, which focuses on leveraging SWRL rules to extend the knowledge base. It uses the nomenclature and categorization of resources developed within the OGF OCCI working group. Recently, this group announced that they will specify semantic RDF renderings of their data models.

Internet of Things

In a similar vein, the need for formal IMs in the IoT domain has already been identified, for example, for interoperability between Smart Cities and Smart Factories [p91]. For the latter, the working group Ontologies for Robotics and Automation (ORA) of the IEEE Robotics and Automation Society (RAS) is working on semantic IMs and mechanisms for the automation area. The European Research Cluster on the Internet of Things (IERC) has established Activity Chain 4 – Service Openness and Interoperability Issues/Semantic Interoperability (AC4) [t64] and a number of semantic models have been developed. For example, the W3C Semantic Sensor Network (SSN) [p47] ontology is used in frameworks like OpenIoT[10] [p132] to semantically annotate information from sensors and other public sources using Extended Global Sensor Network (X-GSN) [p32] gateways to aggregate and to continuously query them using CQELS. Additionally, support for semantics in Machine-To-Machine Communication (M2M) has also received attention [p31]. Accordingly, the main standardization body in this context, the ETSI M2M Working Group, has identified the need for semantic resource descriptions [t79] and the successor of the group, OneM2M[11], has already established the OneM2M Working Group 5 Management, Abstraction and Semantics (MAS).

[8] http://linked-usdl.org
[9] https://www.oasis-open.org/committees/tosca/charter.php
[10] http://openiot.eu
[11] http://onem2m.org

4.2.6 Related Work

Intro Applying semantic models in order to address interoperability issues within the network management context, has been a topic of research since 2005 [p242]. These concepts were further adopted in the Global Lambda Integrated Facility (GLIF) and GENI communities and an overview of current work in this area is given in [p219] and [t90].

NDL/NML One notable thesis that continued the initial work was the definition of the Network Description Language (NDL) [p226], a formalization of the G.805 IM using RDFS. The motivation for NDL was the emergence of hybrid network models in NRENs. Besides visualization, it has been used, for example, for hierarchical routing across heterogeneous multilayer multidomain networks within GLIF and for topology aggregation and abstraction. Based on this work, the subsequent Network Mark-Up Language (NML) [t71] was designed to describe and define computer networks in general. The model underwent a thorough review and definition process to finally become an OGF standard and is under consideration to be used within the OGF Network Service Interface (NSI) [a8]. NML was developed to be as general as possible, with the possibility of extension in order to customize it for emerging network architectures and novel use cases.

NOVI/INDL While NML mainly focused on the underlying network, consequent work analyzed challenges for semantically describing federated virtualized infrastructures [p228] within the Networking innovations Over Virtualized Infrastructures (NOVI) [p229] project. As a result, three SMs for resources, policies and monitoring were created to take these additional aspects into account. They support, in particular, resource discovery and provisioning with a focus on intradomain topology embedding and path finding [p185]. This work lead to the development of its successor Infrastructure and Network Description Language (INDL) [p96, p97] in the GEYSERS [p66] project. INDL describes computing infrastructures in a technology-independent manner by adding concepts and relations that are specific to the computing, processing and storage parts of an infrastructure. INDL addresses in particular the modeling of resource and service virtualization and generally supports the description, discovery, modeling, composition, and monitoring of resources. Further, NML is imported to include the networking part of a computing infrastructure.

NDL-OWL In parallel, NDL has been studied in the GENI initiative to manage and orchestrate resources using the ORCA framework in federated infrastructures within the ExoGENI [p11] testbed since 2010 [p9]. As a result, the Network Description Language based on the Web Ontology Language (NDL-OWL) [p9, t90, t91, t93] was defined, which specifies capabilities for controlling and managing complex networked testbed infrastructures. It extends NDL with OWL concepts to allow complex reasoning and descriptive query-based programming to implement resource allocation, path computation, and topology embedding. Another long-term goal for the definition of NDL-OWL was to provide a potential candidate to replace XML-based RSpecs within SFA.

TaaSOR Based on NDL-OWL, the Testbed as a Service Ontology Repository (TaaSOR) [p219, p220] was developed as a proof of concept to semantically annotate XML RSpecs and publish them using a SPARQL endpoint. As shown in Figure 4.9, testbeds expose their resources using nonsemantic descriptions and, based on a set of domain- and testbed-specific T-Boxes, the information is converted to into RDF triplets and a repository of A-Boxes is created. Further, source code to access the annotated services is generated and the IMs are exposed and visualized within a Web interface. The objective of this work was to move towards the definition of semantic domain-specific description languages to allow resource management and experimentation within federated testbeds. As TaaSOR mainly targets OMF controlled testbeds, the Semantic OMF Experiment Description Language (SOEDL) was specified to generate OEDL application definitions by executing SPARQL queries over semantic knowledge and transforming the results using RDFa template mechanisms.

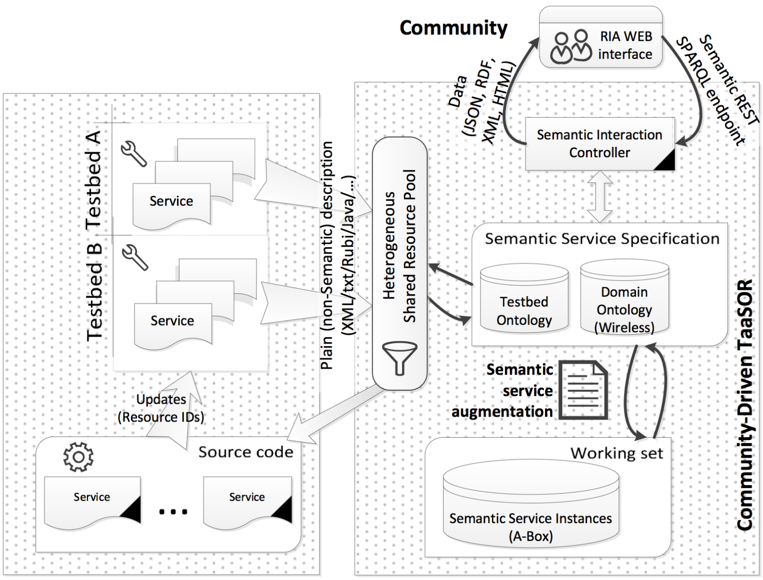


Figure 4.9: Overview of the TaaSOR Architecture [p220]

Finally, besides the definition of semantic models to describe federated infrastructures in general, ontologies have also been defined for the execution of specific experiments. One example of the use of formal IMs in experimental research is the Accessible Wayfinding Testbed (AWT) [p126]. Within this infrastructure, pedestrian wayfinding challenges for people living with disabilities are investigated by modeling indoor and outdoor environments, including sidewalks, hallways, obstacles and connectors, by reusing a number of existing ontologies from this context. Using the constructed graph, it is possible to calculate an accessibility index and feasible paths for navigation purposes, which bears some resemblance to the path-finding approach using NDL. *AWT*

4.3 Ontology Specification

4.3.1 Overview

As shown in the previous section, considerable effort has been spent applying semantic information models within the ICT sector. Existing work in this context serves dedicated purposes in different domains of application and represents valuable foundations. The focus, however, lies mainly on topology embedding [t93], i.e. finding mappings and paths between the available resources. In order to be usable in federated experimentation including the whole life cycle of the resources offered and services within distributed infrastructures, some important concepts must be added, and others are defined slightly differently in related ontologies. This includes modeling the whole federation with its members and infrastructures, interrelation of each phase of the life cycle with it specific *Intro* requirements, and dynamic modification of the model based on the state of requested services.

Goal In order to fill this gap, this section provides the first major contribution of this thesis in the form of an upper ontology called Federated Infrastructure Description and Discovery Language (FIDDLE) [a20]. This OWL-based language can be used throughout all phases of a federated experiment and acts as a common information model. It uses the RDF data model as a common basis, to allow the abstract management of distributed objects within dynamic semantic graphs independent from specific APIs, resource types or management architectures. A single semantic reference structure allows a reduction in the number of potential data converters between testbeds and allows the establishment of higher-value services based on the mechanisms described above, such as automated reasoning, information linking and complex queries. A simplified comparison between an XML RSpec and RDF model is given in Figure 4.10. While semantic information within the RSpec on the left-hand side is implied in the structure of the XML tree and the wording of the names, the partial RDF graph on the right side uses multiple specific namespaced concepts to encode meaning.

```
<rspec>
  <node component_name="device1">
    <hardware_type name="HTC Evo 4G" />
    <sliver_type name="mobile phone" />
  </node>
</rspec>
```

```
:device1 rdf:type owl:NamedIndividual;
         rdf:type fed4fire:MobilePhone;
         rdf:type geni:CellPhone;
         rdfs:label "HTC Evo 4G"@en ;
         omn:instantiates fed4fire:MobilePhone;
         rdfs:comment "This is a mobile phone";
         foaf:based_near :location .
```

Figure 4.10: Simplified comparison of a GENI RSpec and a TTL serialization

Approach The development of a formal model for a domain is not deterministic [t50]. However, as described in [p215], an ontology for a specific domain can be built from scratch [p51] or by modifying an existing ontology [p100]. The starting point for the design of the model at hand is the existing work NML, NOVI, INDL, and NDL-OWL, as well as a number of RSpecs extensions. A number of high-level concepts and properties have been defined by these SMs in similar but not equal ways. Therefore, the present work is intended to act as a seeding document for a joint upper ontology for the generic life cycle management of federated ICT infrastructures in general by building upon, importing and referencing related ontologies. This approach can be compared to the Suggested Upper Merged Ontology (SUMO) [p167], which served as a starter document for the IEEE Standard Upper Ontology (P1600.1).

Overview An overview of the proposed concepts covered by the FIDDLE ontology is given in Figure 4.11. The main levels of semantic abstraction that have been identified are Federation, Infrastructure, Life Cycle, as well as Generic and Specific Concepts. Assigned to each layer are particular objects with their properties to ontologically describe these layers and the relation between them. Additionally, on the right-hand side, potential methods of integration in existing implementations for each layer are identified. In addition to recommending protocols used within the Semantic Web, suggestions are given for utilization and extension of SFA AM and Clearinghouse (CH) APIs for resource discovery and provisioning, FRCP for experiment control, and OMSP for resource and experiment monitoring. Further details will be provided within the subsequent sections that are based on [a20].

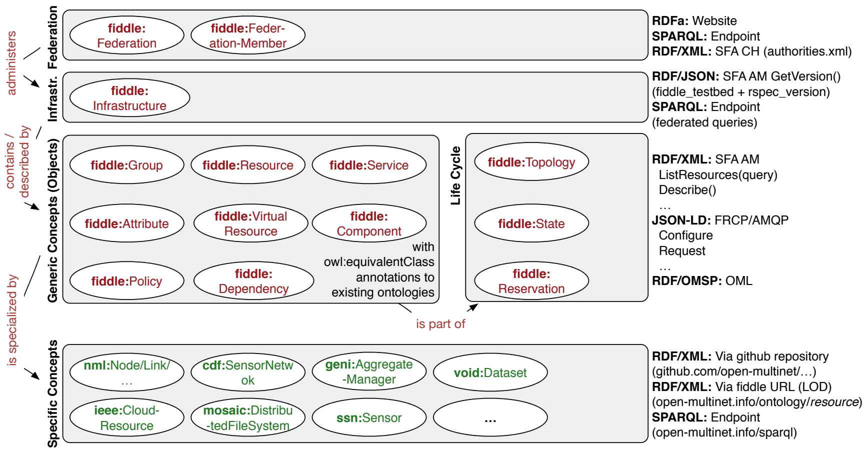


Figure 4.11: FIDDLE architecture and integration concepts (based on [a20])

4.3.2 Federation

The overarching concept that links all available resources with each other is the notion of a *Federation*. In order to describe the federation itself, the concepts *fiddle:Federation* and *fiddle:FederationMember* are introduced. These are both subclasses of the Schema.org *schema:Organization* class, thus allowing them to be described by properties of the Schema vocabulary that apply to this class. The federation level of abstraction is illustrated in Figure 4.12, along with the relevant object properties. *Overview*

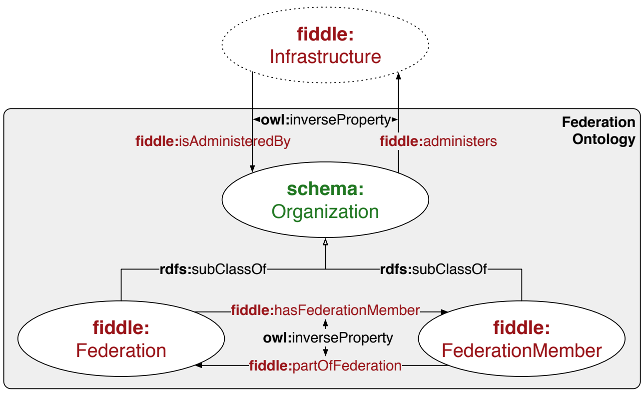


Figure 4.12: FIDDLE federation level (based on [a20])

This approach was chosen because these concepts are not purely technical and may have properties like an address, an administrative structure, a funding body and so forth. If these were grouped with the other entities in the ontology, then it would not be appropriate to attribute such properties to them, from either an abstract semantic or technical perspective. Both a *fiddle:Federation* and a *fiddle:FederationMember* may also administer one or more *fiddle:Infrastructures*, denoted by the object property *fiddle:administers* and its inverse, *fiddle:isAdministeredBy*. *Approach*

Currently, this kind of information is often described on a Web site to provide users information about the federation and the participating infrastructures. By annotating *Integration* existing data on pages using RDFa, it would be possible to reuse existing information, maintain the information at a single location and take it as a starting point to create a semantic graph of the overall federation. Further, in the GENI context the possible establishment of an SFA CH API would provide metainformation about the federation. For logical reasons, this information should be encoded as semantic graphs. Finally, a SPARQL endpoint could be offered, operated by the federation, in order to allow complex federation-wide queries. For the last case, further information about participating members, testbeds and their resources should be available as well and each federation member could offer its own endpoint, thereby exploiting the built-in federated query capabilities of SPARQL.

4.3.3 Infrastructure

Overview While the object property *fiddle:administers* denotes an administrative relationship to an organization, an infrastructure itself may exist whether or not it is part of a *fiddle:Federation* and its description focuses on the technological aspects. The related concept is illustrated in Figure 4.13, along with the relevant object properties.

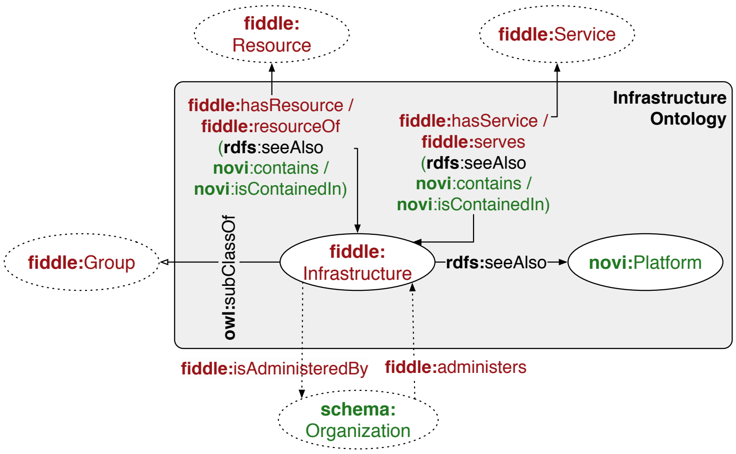


Figure 4.13: FIDDLE infrastructure level (based on [a20])

Approach The term *Infrastructure* is introduced here with a similar sense to *novi:Platform*, but with broader scope. In particular, it is not restricted to testbeds, but is expanded to general ICT infrastructures that can be federated, such as in the Intercloud context. Further, the class *fiddle:Infrastructure* is a subclass of a *fiddle:Group* and can therefore be seen as a general collection of *fiddle:Thing* (analog to *schema:Thing*), expressed by the relations *fiddle:hasResource* and *fiddle:hasService* (analog to the properties *nml:hasService* and *nml:hasNode*).

Integration In order to make this information available in a semantic manner within SFA, the response of the existing *GetVersion* method could be extended. Its definition allows arbitrary information to be attached to the returned JSON-based data model, e.g. by adding JSON-LD-encoded RDF graphs. Further, this information could be exposed by a potential SPARQL endpoint at each facility or by the federation's central endpoint.

4.3.4 Generic and Specific Concepts

Overview

The generic FIDDLE concepts are shown in Figure 4.14, together with all subclasses, relations and selected object properties. Note that most of the object properties have inverse relations that are not shown in the figure. Equivalent classes and *rdfs:seeAlso* references are also not shown to improve readability. The main purpose of these concepts is to allow the description of resources and services available within an infrastructure. As an infrastructure is a specific, static type of the general *fiddle:Group* concept, these relationships can be expressed by using the *fiddle:hasResource* and *fiddle:hasService* properties. In particular, it is possible to define that a given resource is composed of multiple components and can implement (i.e. instantiate) virtual resources and services. Further, an object can require or depend on the existence of another object to support service chains.

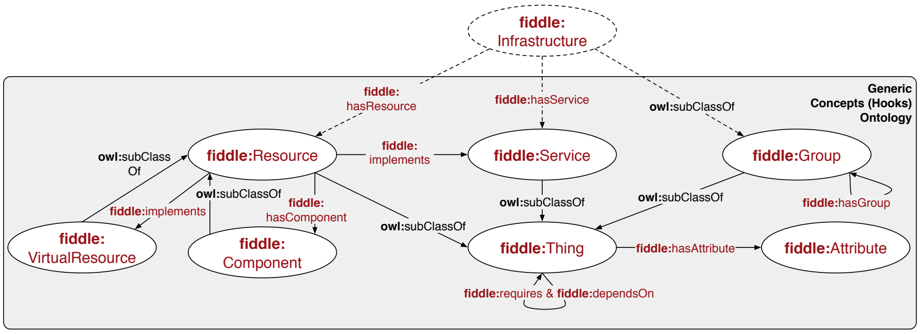


Figure 4.14: FIDDLE generic concepts (based on [a20])

Approach

A number of related common concepts have been previously identified in the NML, NOVI and INDL vocabularies. These are linked to here with *owl:equivalentClass* or *rdfs:seeAlso* annotations, as depicted in Figure 4.15. The class *fiddle:Thing* is comparable to *nml:NetworkObject*, schema:Thing and *novi:Resource* and acts as the superclass of the basic concepts *fiddle:Resource*, *fiddle:Group* and *fiddle:Service*. These are again based on the NOVI classes of the same name and are comparable with the NML concepts *nml:Node*, *nml:Group* and *nml:Service*. As per NOVI and INDL, a *fiddle:Resource* has the subclasses *fiddle:VirtualResource* and *fiddle:Component*. These can be related to a *fiddle:Resource* by the object properties *fiddle:implements* and *fiddle:hasComponent* respectively.

Others and Outlook

Each of these basic concepts can also take on a *fiddle:Attribute* via the *fiddle:hasAttribute* object property, allowing extension for, for example, read-only monitoring attributes or read-write resource control attributes. This approach is comparable with the *indl:Capability* concept and allows detailed description of the kind of capabilities a resource or service offers to a user. Further work is also planned to design the *fiddle:Dependency* concept, analog to the *ndl-owl:Color* idea, in order to specify generic dependencies between individuals based on output or input values for service chaining. The envisioned *fiddle:Policy* concept could specify authorization-related information, analog to the NOVI Policy IM.

Specific Objects

In order to describe the heterogeneous services and resources in a federated environment, domain-specific ontologies have to be defined that re-use and link existing work. While the definition of these ontologies is purposely out of scope, they should be linked to the aforementioned generic concepts to support their discovery and management.

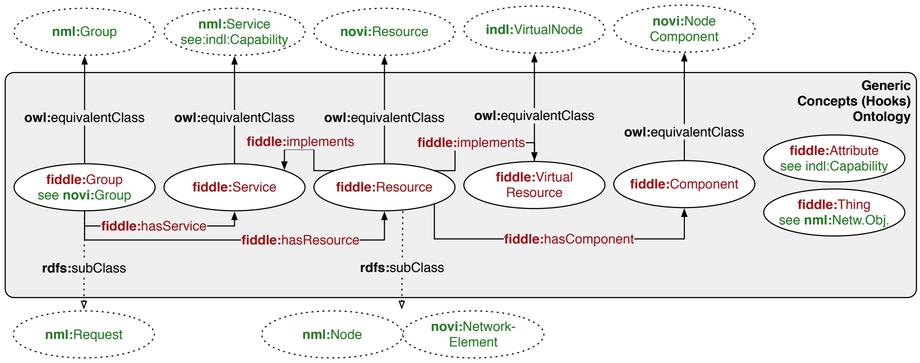


Figure 4.15: FIDDLE relation of generic concepts to existing ones (based on [a20])

Examples are ontologies to describe wireless access technologies or Evolved Packet Core (EPC) components for testbeds that offer these type of resources and services.

Integration Integration-wise the SFA AM method *ListResources* and related methods for resources description allow the user to specify arbitrary data models. Therefore, instead of the default GENI RSpecs, RDF/XML-encoded graphs can be returned. As the method calls further allow any kind of parameters to be specified, SPARQL queries could be included in the request to filter resources on the server side. Next, while FRCP allows two different serializations for the control and monitoring phase, it does not specify a concrete resource model. As a result, RDF/XML- or JSON-LD-serialized graphs could be embedded in the relevant method calls. Finally, as OMSP for monitoring only requires the exchanged data structure to be a list, an appropriate serialization could be specified and used within this protocol.

4.3.5 Life Cycle

Overview A major goal of FIDDLE is to allow the life cycle of heterogeneous resources in federated environments to be modeled within a dynamic graph structure. Therefore, the concepts depicted in Figure 4.16 together with the properties shown in Figure 4.15, such as *fiddle:implementedBy*, allow a user to request a subtopology that can be provisioned, monitored and controlled within an experiment.

Approach The key definition in this area is the concept of a *Topology*. Analog to the *nml:Topology* and *novi:Topology* it is a subclass of *Group* and acts as a container for a dynamic topology, also called testbed or infrastructure, or *Slice* in GENI terminology. It is a collection of resources and/or services and the relationships between them, as requested by an experimenter, with a reservation and state attached to it. As resources, services and the requested topology as a whole can be in various states, such as unallocated, pending or provisioned, this information has to be encoded as well. Finally, as each resource may be reserved within multiple time slots, the reservation of the topology is modeled using the *fiddle:Reservation* class to specify the lifetime of a requested topology. Specifically, for resources that may be exclusively reserved by experimenters, availability should be explicitly advertised. The object property *omn:hasReservation* expresses the relation between a topology and its reservation.

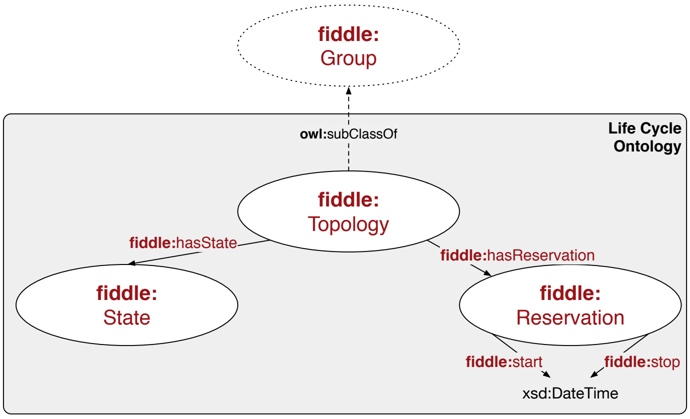


Figure 4.16: FIDDLE life cycle concepts

4.3.6 Open-Multinet

Overview

The purpose of the FIDDLE ontology presented above was to facilitate the development and potential standardization of a formal model for life cycle management in federated infrastructures. The definition of an upper ontology for the broad application context of federated infrastructures, however, requires the involvement of many stakeholders, such as tool developers, facility owners and federation operators. As a result, the OWL-encoded Open-Multinet ontology suite has been defined as a refinement and extension of FIDDLE, in close collaboration with a community of relevant experts within the FIRE and GENI community and authors of related ontologies. The models are still under revision and the initial work conducted in this context has been published in [a24], forming the basis for this subsection.

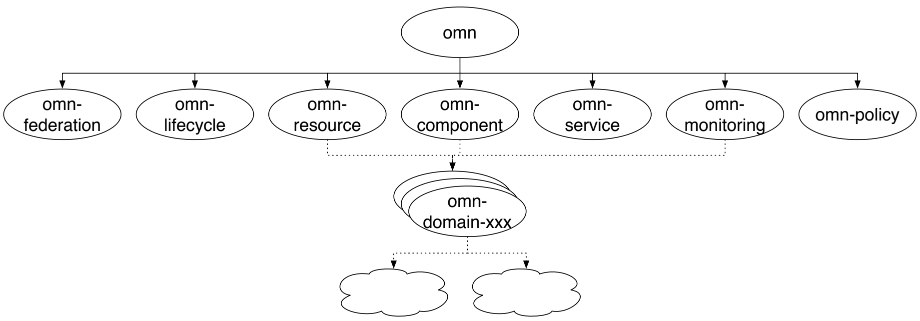


Figure 4.17: OMN upper ontologies (based on [a24])

Approach

The ontology bundle is split into a hierarchy of a number of different ontologies as shown in Figure 4.17. The OMN ontology on the highest level defines basic concepts and properties, analog to the generic FIDDLE concepts, which are then re-used and specialized in the subjacent ontologies. Included at every level are (i) axioms, such as the disjointedness of each class to allow proper reasoning; (ii) links to concepts in existing ontologies, such as NML, INDL and NOVI (cf. Figure 4.18); and (iii) properties that have been shown to be needed in related ontologies.

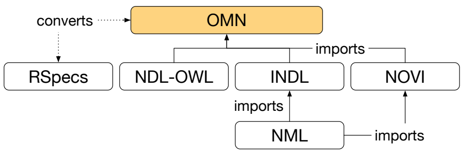


Figure 4.18: Relation between OMN and other work (based on [a24])

Main Concepts

The OMN upper ontology defines the abstract terms required for describing federated infrastructures in general. Figure 4.19 illustrates an overview of the key concepts and properties. The main concepts are as follows based on [a24]:

- *Resource*: A stand-alone component of the infrastructure such as a network node, which can be provisioned, i.e., granted to an experimenter.
- *Service*: A manageable entity, which can be controlled and/or used via APIs or capabilities it supports, e.g. an SSH login.
- *Component*: A part of a *Resource* or a *Service*, e.g. a port of a network node.
- *Attribute*: Description of the characteristics and properties of a specific *Resource*, *Group*, or *Component*, e.g. QoS.
- *Group*: A collection of resources and services, e.g. a testbed or a requested network topology.
- *Dependency*: A unidirectional relationship between *Resource*, *Service*, *Component* or *Group*, which opens up the possibility to add more properties to a dependency via annotation.
- *Layer*: A place within a hierarchy that a specific *Group*, *Resource*, *Service* or *Component* can adapt to.
- *Environment*: The conditions under which a *Resource*, *Group*, or *Service* is operating, for example, concurrent virtual machines.
- *Reservation*: A specification of a guarantee for a certain duration, which is a subclass of the *Interval* class of the W3C Time ontology [t30], used to encode, for example, start and end times.

Main Properties

Besides the main concepts, 21 properties have currently been defined, including inverse counterparts in order to support rich querying and inferencing.

- *adaptableTo* relates a *Layer* to another *Layer* to which it can adapt (e.g. from Ethernet to IP). The inverse is *adaptableFrom*.
- *adaptsFrom* determines the *Group*, *Resource*, *Service* or *Component* from which another *Group*, *Resource*, *Service* or *Component* adapts. The inverse is *adaptsTo*.
- *fromDependency* relates a *Group*, *Resource*, *Service* or *Component* to the *Dependency* to which it belongs. The inverse is *toDependency*.
- *relatesTo* generally relates a *Group*, *Resource*, *Service* or *Component* to another *Group*, *Resource*, *Service* or *Component* to which it belongs. The subproperty *dependsOn* claims a direct dependency.
- *hasAttribute* the Attribute associated with a *Component*, *Resource*, *Service* or *Group*. The inverse is *isAttributeOf*.
- *hasComponent* links a *Component*, *Resource*, or *Service* to its subcomponent. The inverse is *isComponentOf*.
- *hasGroup* connects a *Group* to its subgroup. The inverse is *isGroupOf*.
- *hasReservation* relates *Group*, *Resource* or *Service* to its *Reservation*. The inverse is *isReservationOf*.

- *hasResource* declares that a specific *Group* has a *Resource*. The inverse is *isResourceOf*.
- *hasService* declares that a *Group*, *Resource* or *Service* provides a *Service*. The inverse is *isServiceOf*.
- *withinEnvironment* defines the *Environment* in which a *Group*, *Resource*, *Service* or *Component* operates.

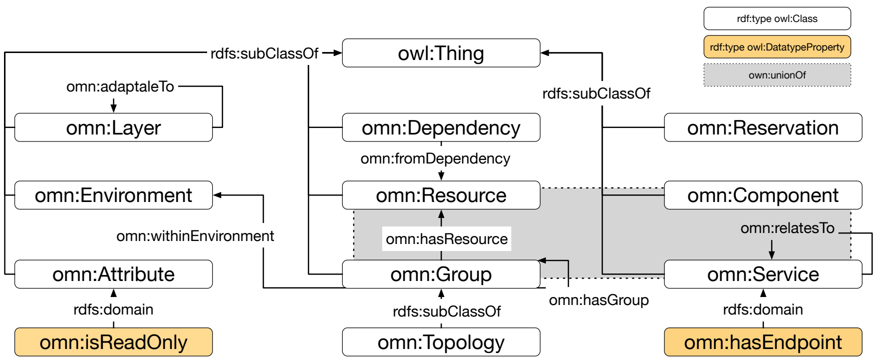


Figure 4.19: Key concepts and properties of the OMN upper ontology (based on [a24])

Life Cycle

Due to its novelty, importance and extensive utilization within implementations, the life cycle ontology in particular was revised to a large extent. The initial *Topology* class has been extended by four new concepts, in order to map all relevant phases of the life cycle: (i) *Offering*, which provides all resources and services that can be allocated to a user's request; (ii) *Request*, which expresses a collection of resources or services and the relationships between them, as requested by an experimenter in a bound or unbound manner; (iii) *Confirmation*, which provides a collection of resources or services that are reserved and scheduled to be allocated at a future point in time; and (iv) *Manifest*, which describes a collection of resources or services currently provisioned by the experimenter. The relevant mapping to the SFA methods, including the potential serializations for FRCP and OMSP, is depicted in Figure 4.20, which illustrates that the work presented covers the whole life cycle of an experiment.

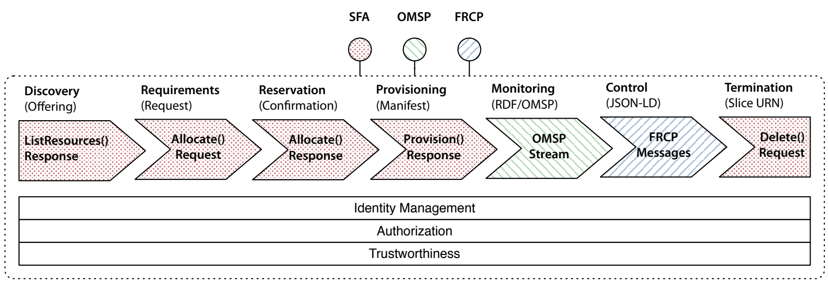


Figure 4.20: Mapping of the ontology to the life cycle phases

4.4 Conclusion

Summary This chapter gave an overview of the state of the art in information modeling and motivated the need for formal information models in the information exchange between federated infrastructures. Based on best practices and related work, the semantic model FIDDLE and its successor OMN were introduced. These ontologies are capable of describing federations of infrastructures and the whole life cycle of an experiment on a semantic level, while being integrable into existing protocols. They support the management of distributed resources and services by dynamically creating and modifying RDF graphs and are therefore decoupled from specific APIs. The proposed work acts as a canonical model to solve compatibility issues between federated testbeds and enable software tools to understand the meaning of resource descriptions. This allows the use of reasoners and formal concepts to query, link, chain, map, combine and validate information abstracted from concrete implementations details.

Standardization To broaden the scope of this work beyond experimentation in federated testbeds and with the goal of defining a standard, the W3C Federated Infrastructures Community Group was established.Using the OMN ontologies as the starting point, a larger community is validating their adaptability for further fields of application, such as Intercloud computing, in order to further refine the models and to incorporate additional requirements. As a result, the work is discussed within the IEEE P2302 working group.

Outlook As highlighted in Figure 4.20, the AuthN, AuthZ and trustworthiness layers in particular have not yet been covered to a great extent. Notably, adding authorization hooks, i.e. properties relating resources to authorization roles, would allow federation-wide and fine granular access rules to be defined. The definition of pricing and availability of resources could enable the negotiation and implementation of SLAs. Adopting existing work on semantically describing policies, such as [p217], would allow behavior governing the managed environment to be defined.

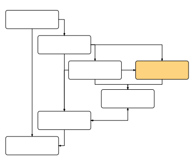
Next To show the applicability of the information model developed for the management of the entire life cycle of arbitrary resources, a reference implementation is presented in Chapter 6, based on the management architecture defined in Chapter 5. The framework internally uses OMN to discover, provision, control and monitor resources based on a dynamic, semantically labeled graph, by adding, removing and updating edges and graph nodes.

CHAPTER 5

DESIGN AND SPECIFICATION OF THE FIRMA ARCHITECTURE

5.1 Introduction

Overview



The ontology designed for this thesis and presented in Chapter 4 can be used to describe each phase of the experiment life cycle and, therefore, provides the means to manage federated resources via abstracted semantic graphs. Following such an approach allows two of the main challenges described in Chapter 2 to be addressed. First, within the GENI and FIRE context, multiple incompatible APIs with different DMs exist in parallel for conducting experiments using federated testbeds, leading to a fragmented and complex environment for users, testbed owners, and developers. Second, resources used within FI experiments are heterogeneous in nature and defining arbitrary resources upfront or allowing any kind of extensions impedes the possibility of complex discovery mechanisms. Therefore, the extensible Federated Infrastructure Resource Management Architecture (FIRMA) [a21] with a protocol- and resource-agnostic, semantic core is presented in

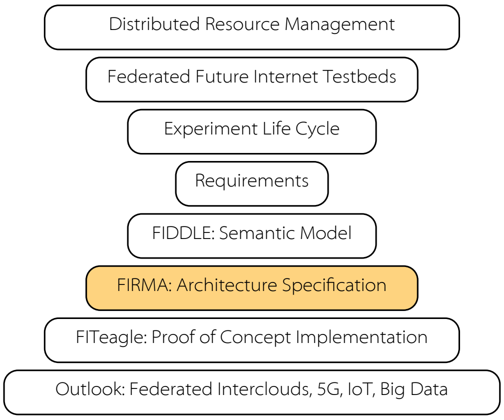


Figure 5.1: Placement of the architecture specification in the structure of research

this section. FIRMA offers interchangeable federation and experimentation interfaces concurrently, supports heterogeneous resources, and integrates existing infrastructure services. As the basic principles of mechanisms for federating infrastructures, such as in the FI-PPP or the IEEE group P2302, are mostly equivalent on a semantic level, it supports adding further APIs without modifying core components.

Structure of Research

As indicated in Figure 5.1, the design of FIRMA follows the requirements presented in Chapter 3 and incorporates the semantic model developed in Chapter 4. Together with the ontology, the architecture builds the main basis for the reference framework in Chapter 6. Parts of the work presented in this chapter have been published before in [a21] and have been revised during the course of its implementation.

5.2 Overall Architecture

Overview

Based on the knowledge gained from the contexts described in Chapter 2, the proposed architecture should to be decoupled from both the user-facing APIs and the underlying resources. The generalized overview of the FIRMA architecture in Figure 5.2 highlights two further independent areas that communicate with each other using semantic models. Management of the resource life cycle is handled in an abstracted graph-based manner in the *Core Modules*, the concrete user-facing APIs are implemented as *Delivery Mechanisms*, specific resources are integrated using *Resource Adapters*, and other information is exported and imported in the *Service Integration* context. The different modules are decoupled from each other by allowing internal communication only through the exchange of semantic information models over a message bus. While the notion of a semantic service bus is not new [p41, p249], no such architecture is currently known to exist.

5.2.1 Design Principles

Details

The architecture presented here encourages reusability by Separation of Concerns (SoC) [p156, m9], a term coined by Edsger W. Dijkstra in 1974 [p59]. FIRMA combines established design patterns, such as Entity, Boundary, Interactor (EBI) [p123] and Data, Context and Interaction (DCI) [p48], with modern approaches, such as Microservices,

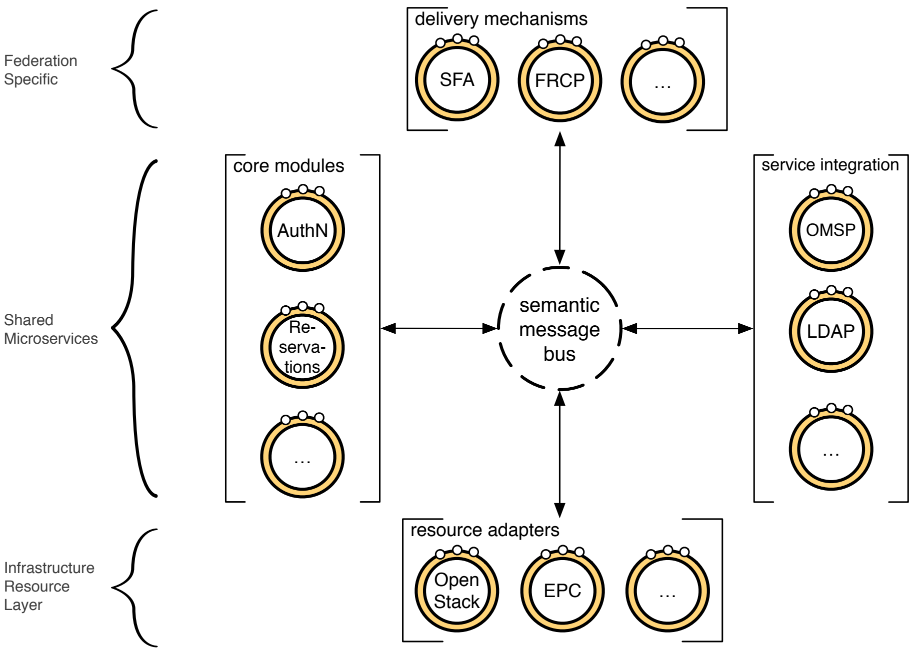


Figure 5.2: FIRMA generalized architecture

and Semantic Web based technologies. These patterns share some common objectives and requirements for the design of a system that can be summarized as follows:

1. Framework independent: The architecture should not depend on too many external libraries or frameworks. They can be used as tools rather than inseparable parts of the architecture.
2. Interface independent: The User Interfaces (UIs) and APIs can be changed without affecting the architecture because they are not bound to it.
3. Database independent: The persistence technology can be changed without affecting the architecture because the core logic is not bound to it.
4. Resource independent: The testbed resources can be changed without affecting the architecture because the core logic is not bound to them.
5. Testable: The core logic can be tested without any external elements, following a Test Driven Development (TDD) [p14, p156] approach.

MVP

Another well-known architectural pattern for SoC is the Model-View-Presenter (MVP) [m12] design. The architecture proposed here (cf. Figure 5.4) maps to MVP as follows. The *Model*, i.e. the actual business objects and data, is represented by the *Adapters*, *Entities*, *Premises*, and *Resources*. The *View*, i.e. the interface to the model, is represented by the *Delivery Mechanisms*, which are used by the *Clients*. Finally, the *Presenter*, i.e. the component that contains the actual business logic, is represented by the *Interactors*.

Details

Based on these principles, FIRMA follows the Microservice architectural design pattern, with semantically enriched messages. In contrary to complex Enterprise Service Bus (ESB) [p39] approaches, its underlying concept is based on “smart endpoints and dump pipes” [m5]. Each module, as sketched in Figure 5.3, consists of a reusable core functionality that is accessible from the outside via one or more decoupled *delivery mechanisms* with their corresponding API *endpoints*. Communication to external ser-

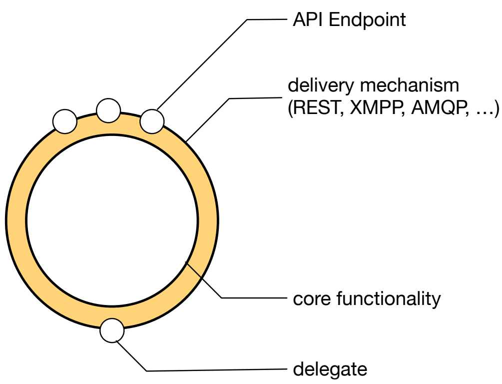


Figure 5.3: FIRMA module design

vices are handled by so called *delegates*, e.g. code that contains logic for communicating with a specific resource, or mocked functionality for testing purposes.

Layered Architecture

A more detailed view of the overall reusable and extensible architecture is given in Figure 5.4. FIRMA takes the aforementioned architectural design patterns into account by dividing it into five functional areas: existing (i) *User Tools* are used to communicate with a number of (ii) *Delivery Mechanisms*, from which events are forwarded to a (iii) *Protocol and Resource Agnostic Core*, which in turn delegates concrete actions to *Resources* via (iv) *Adapters* and services at the *Premises* via (v) *Integrators*. Following the Dependency Inversion Principle (DIP) [p157] makes the architecture reusable and is important for construction of code that is resilient to change. Empty arrows depict the implementation and filled arrows represent the use of an interface by a module. The concrete areas are described in the following sections.

5.2.2 User Tools

Tools

As described in Chapter 2, a variety of user tools and developer toolkits are well established in their respective user communities and are under active development. Depending on the working environment of the user, specific tools will be used and it is unlikely that this would change in the medium term. Therefore, as mentioned in Chapter 1, the main target is not to offer new user tools but to support existing clients, use them for acceptance testing and ensure compatibility with other implementations. Existing tools could include SFA-, FRCP-, and OMSP-compliant user tools, such as SFI, OMF, NEPI and OML, including support for Graphical User Interfaces (GUIs) such as jFed or MySlice. Other possible candidates include OCCI-compliant clients within the Intercloud and Generic Enabler (GE) compliant clients within the FI-PPP / FIWARE contexts.

Links

The links from various user tools shown in Figure 5.4 are (**1**) communication with the *Delivery Mechanisms*, (**2**) communication with other federation partners, and (**3**) communication with the *Delivery Mechanisms* from other clients originating from within the federation.

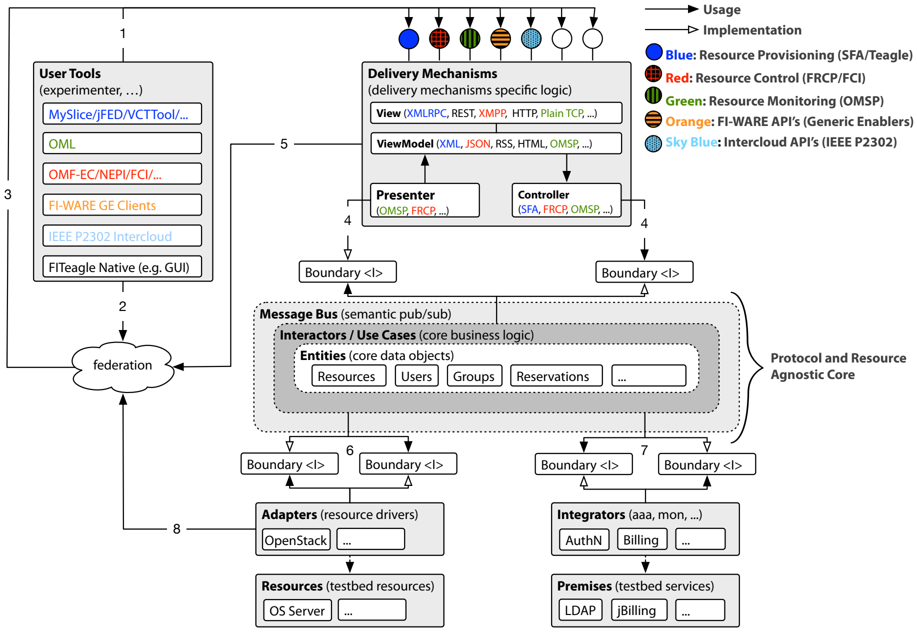


Figure 5.4: FIRMA layered architecture overview (based on [a21])

5.2.3 North: Delivery Mechanisms

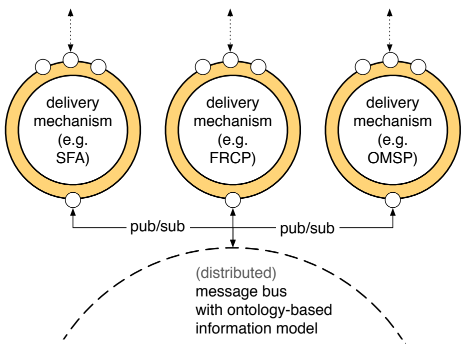


Figure 5.5: FIRMA northbound modules

Overview

Analogous to the number of clients, there are also countably many standardized interfaces that are used by federation and experiment control protocols. Therefore, the *Delivery Mechanisms* block is the single point of entry and is responsible for all incoming communication from external components. In particular, as depicted in Figure 5.5, it simultaneously offers multiple APIs for the different clients and further handles transport layer security. Due to its independence from the core modules, further delivery mechanisms, like legacy interfaces such as Common Object Request Broker Architecture (CORBA), Simple Object Access Protocol (SOAP) or other proprietary message protocols, can be added without interfering with exiting protocol implementations.

Modules To achieve flexibility within this block, the specific implementation of each protocol is designed to be separated into four independent modules, as shown in Figure 5.4:

- **View**: The view abstracts the transport protocol from the actual message. Examples are the XML-RPC interface over Secure Sockets Layer (SSL) for SFA clients; the Extensible Messaging and Presence Protocol (XMPP) or the AMQP interfaces for FRCP clients; raw TCP sockets for OMSP clients; REST interfaces for Teagle clients or to provide a SPARQL-compliant endpoint; or a HTTP interface for Web clients.
- **ViewModel**: The *ViewModel* presents the data exchanged. Independent of the protocol used, the message itself is structured following a given syntax. Examples are XML, Hyper Text Markup Language (HTML), JSON, N3, and the DEN-ng-based SID in the eTOM/NGOSS context. This abstraction allows the syntax to be replaced without modifying the data model of the actual messages that are used in the underlying modules.
- **Controller**: The *Controller* contains protocol-dependent business logic for incoming messages, e.g. for SFA AM *ListResources* calls. The request is processed and functionalities available in the core are delegated to the required *Interactors*.
- **Presenter**: Analog to the *Controller*, protocol-dependent business logic for outgoing messages,e.g. for OMSP streams, is located in the *Presenter*. The *Presenter* is called by the *Interactors* in case a request is sent out to an external component.

Links The links shown in Figure 5.4 for the northbound area are (**4**) bilateral publish-subscribe communication between the core system and the delivery mechanism and (**5**) push communication to other federation partners, e.g. for pushing monitoring data.

5.2.4 West: Core Modules

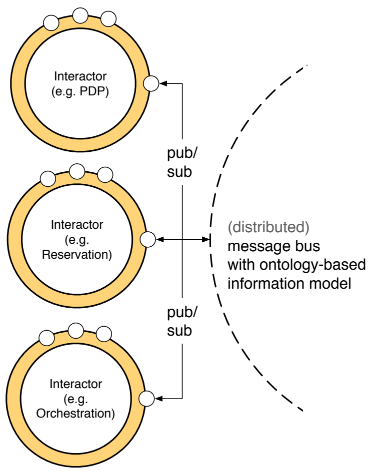


Figure 5.6: FIRMA westbound modules

Overview In the westbound modules, the core business logic and use cases that are needed across several northbound protocols are implemented. This area is composed of cross-cutting concerns that are needed by a number of services within the system. As indicated in Figure 5.6, examples include a Policy Decision Point (PDP) and a resource reservation and orchestration module. Other westbound modules include management of users, groups, resources, policies, persistence and reservations; schedules or monitoring data; logging; configuration; elasticity; and analytics. As a result of the varied functions of the modules, the *Delivery Mechanisms* contain a limited set of business logic that is specific to the relevant protocol, and sends messages to these modules for further functionalities.

Details As detailed in Figure 5.4, these *Interactors* use semantic business objects called *Entities* for internal data representation and information exchange. This abstraction allows the implementation to be agnostic to the selected federation and be reusable in other contexts. The overall concept of loosely interconnected components assures reusability, interchangeability and testability. It further allows components to be distributed and, therefore, enable the architecture to scale as demands raise or to add and remove components without interfering with the running system.

Links The links shown in Figure 5.4 for the core area are (**6**) bilateral publish-subscribe communication between the core system and the adapters to provide resources from local or federated infrastructures and (**7**) bilateral publish-subscribe communication between the core system and services to integrate infrastructure-internal aids.

5.2.5 South: Resource Adapters

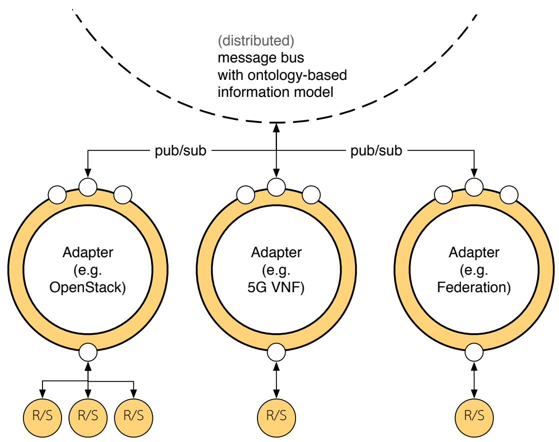


Figure 5.7: FIRMA southbound modules

Overview To unify the management of heterogeneous resources, the common *Adapter* architectural design pattern is used, comparable to the concept of device drivers. The concept of adapters has been adopted from the Teagle architecture, where RAs only handle resource provisioning and release. As depicted in Figure 5.7, each adapter encapsulates one or more resource instances of a single resource type by offering a semantic interface for bidirectional message exchange related to resource description, provisioning, control, monitoring, and release.

Details Translation from the incoming function call to the actual outgoing call is resource specific and it can be mapped, for example, to Telnet or SSH sessions, Simple Mail Transfer Protocol (SMTP) commands, REST calls or Web Socket pipes. Generally, physical devices, such as servers, network routers or even a coffee machine, are resources in this context, as are virtualized software components, such as EPC, IP Multimedia Subsystem (IMS) or M2M-related Virtualized Network Functions (VNFs). On this level, another infrastructure or a whole federation of infrastructures could likewise be handled as a group of available resources that can recursively be abstracted and controlled. In this case, a specific FIRMA instance would act as a broker.

Links The link shown in Figure 5.4 for resource adapters is (**8**) providing resources from external infrastructures.

5.2.6 East: Service Integrators

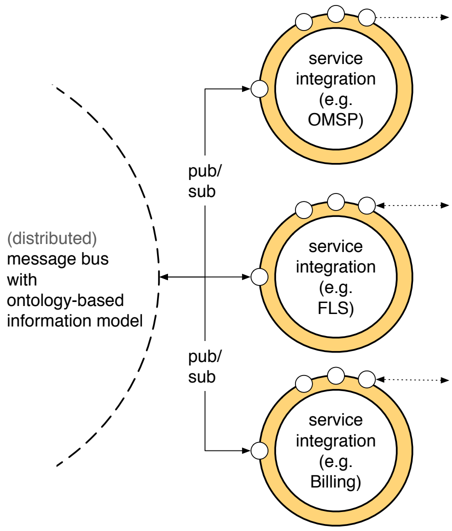


Figure 5.8: FIRMA eastbound modules

Overview Although the main functionalities are implemented within the core, several services are often already in place or must be integrated. Analogous to *Resource Adapters*, which encapsulate manageable resources within an infrastructure or federation, the *Service Integrators* act as the interface to infrastructure or federation-wide utilities. As shown in Figure 5.8, the architecture is designed to delegate parts of the functionalities to external services located at the infrastructure *Premises*.

Details Business frameworks, such as the IT Infrastructure Library (ITIL) or eTOM, which is published by TMF and is part of NGOSS, define technical functionalities that are required for the operation of commercial infrastructures. Examples include FLS, billing, Customer Relationship Management (CRM), SLA monitoring and OMSP export, and local Identity Managements (IdMs) for delegating AuthN requests to an existing LDAP database.

Links The relevant link shown in Figure 5.4 for this area is (**7**) using services located within the local infrastructures.

5.3 Distribution

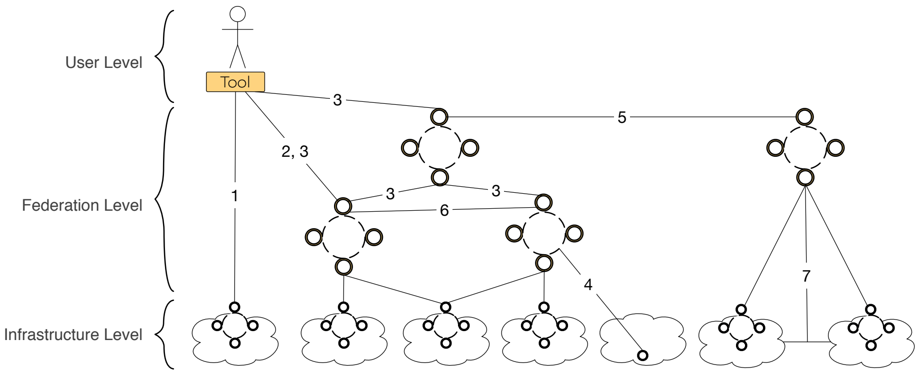


Figure 5.9: FIRMA distribution across administrative domains

Overview

The preceding sections described the overall FIRMA design and its components with a focus on its deployment within a single infrastructure. As shown in Figure 5.9, however, for administrative, scalability or security reasons, the architecture could be operated in a range of potential distribution patterns involving several domains.

5.3.1 Centralized and Hierarchical

(a) centralized architecture (b) hierarchical architecture

Figure 5.10: FIRMA centralized and hierarchical distributions

Single Domain

The single domain setup (cf. **1** in Figure 5.9) involves the installation of FIRMA within one infrastructure that the user is connected to. While users can reuse their authentication credentials to access multiple testbeds within the GENI and FIRE federations, this architectural type is currently the most commonly used in these contexts. Each infrastructure operates its own AM and users connect to without intermediate components.

Centralized

Within the centralized domain setup (cf. **2** in Figure 5.9) the user is communicating with a broker that in turn orchestrates the request to the infrastructures. Figure 5.10a shows such an architecture, where the white circle represents the broker, the gray circles the AMs and the gray boxes the infrastructures. To facilitate the discovery and reservation of resources and to enable the orchestration of unbound requests, appropriate centralized brokering mechanisms have been developed, e.g. [p222], and further advanced in research projects such as OpenLab and Fed4FIRE.

Hierarchical Domain

As an extension to centralized brokering, the hierarchical architecture (cf. **3** in Figure 5.9) further divides the reach of each broker into a tree of subbrokers. As sketched in Figure 5.10b the client connects to one of the existing brokers and the messages and resource information are routed between the interconnected administrative domains. This setup distributes the load between different brokers and, further, can be required due to reasons of governance (e.g. one broker for the EU and one for the US). An example for such a distribution has been presented in [a2].

Unmanaged Domain

As the setup and maintenance of an AM at an infrastructure involves significant effort, unmanaged domains could also be part of a federation (cf. **4** in Figure 5.9). In this setup one or multiple infrastructures are integrated into a federation by a single FIRMA instance that is operated by a third party. Given a certain level of trust, infrastructures can outsource the management of their own resources in this manner. Such a design was chosen by the TRESCIMO project and, partly, by the Teagle framework.

5.3.2 Peered and Hybrid

(a) peered architecture (b) hybrid architecture

Figure 5.11: FIRMA peered and hybrid distributions

Peered

Besides the hierarchical broker setup, the FIRMA instances could also communicate on a peer-to-peer level (cf. **5** in Figure 5.9). In Figure 5.11a, a variant is depicted, which allows administrative broker domains to exchange information on an equally privileged level without the use of a superior entity. In addition to administrative concerns, such an architecture can reduce the length of the daisy-chained communication needed to reach the corresponding broker within a hierarchy and reduce the overall response time.

Hybrid

The hybrid mode (cf. **6** in Figure 5.9) combines the hierarchical and peer-to-peer architectures. As highlighted in Figure 5.11b, brokers within a hierarchy could additionally establish peered trust and communication channels to reflect more complex governance structures. However, depending on the message exchange algorithms, corresponding routing loops, for example, have to be considered and avoided.

Network

Finally, the interdomain network itself can be considered a manageable infrastructure (cf. **7** in Figure 5.9). Research on related Dynamic Circuit Networks (DCNs) was conducted in recent years for provisioning and reserving of hybrid multilayer networks. Based on, besides others, the work within AutoBahn [p234], UCLP [p102], ARGON [a15], Harmony [a19], OSCARS [p107], and DRAC [t69], the OGF NSI group is proposing a related standard interface. Further, with the growing interest in OpenFlow [p161] based SDNs, the network and Software Defined Wide Area Networks (SD-WANs) [t54] are gaining attention as first class manageable resources.

ETSI NFV

This has been further reflected by the functional blocks that are defined in the ETSI Management and Orchestration (MANO) [t81] specification to manage VNFs across multiple domains. As shown in Figure 5.12, the Wide Area Network (WAN) is managed by an Element Management System (EMS) / Network Management System (NMS) and interconnects two Network Function Virtualization Infrastructures (NFVIs).

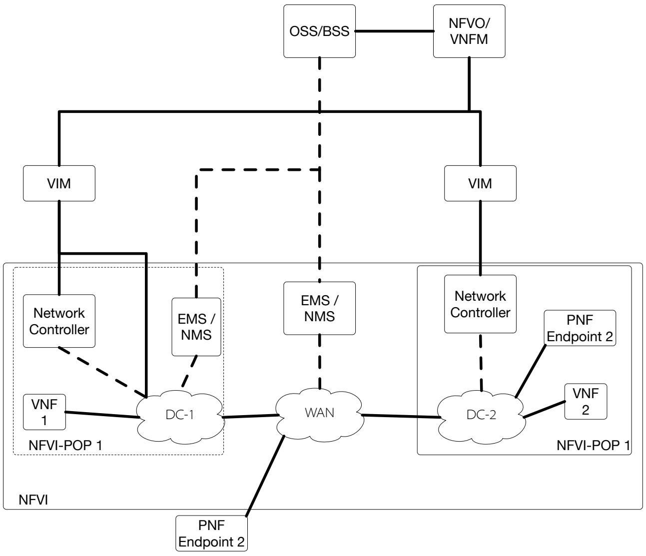


Figure 5.12: Interconnected NFVI multidomain network (based on [t81])

5.4 Conclusion

Summary

The previous chapters described existing approaches to federate and control resources across multiple administrative domains. It become clear that a single mechanism can't be used in every context. By contrast, most of these mechanisms share many similar concepts and information about the same resources. Furthermore, within the FI research area, different independent tools must interact with each other to cover the whole experiment life cycle. Therefore, in this chapter the extensible multiprotocol FIRMA architecture, its underlying design principles and its components were introduced and described. Based on the exchange of semantic events between decoupled services, the approach supports multiple, independent interfaces in parallel, allowing them to discover, reserve, provision, monitor, control, and release heterogeneous resources and integrate external services. The management architecture can be used within a single testbed, on top of multiple infrastructures involving interdomain connectivity, or as a federation-wide service, and can further be operated in a hierarchical or peered manner.

Improvements

As a result of this design, developers can use this architecture as a foundation for implementing and linking federation mechanisms within a single architecture using existing common denominators (e.g. for authorization handovers between protocols), testbed owners can provide resources simultaneously using different interfaces, and users have a single point of entry for their experiments. The use of formal information models further allows existing work from the Semantic Web context to be exploited, such as supporting complex queries, reasoning and information linking.

Next

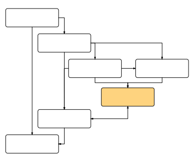
As proof of concept of the presented architecture and ontology, Chapter 6 describes a reference implementation. The implementation demonstrates the life cycle management of resources in the FI experimentation context based on semantic graphs, and abstraction from context-specific northbound and resource-specific southbound APIs.

CHAPTER 6

IMPLEMENTATION OF THE FITEAGLE FRAMEWORK

6.1 Introduction

Overview



In Chapter 4, the FIDDLE and OMN ontologies were introduced, allowing infrastructure federations and the experiment life cycle to be modeled formally based on semantic graphs. In Chapter 5, the FIRMA architecture was described. FIRMA uses OMN in order to be agnostic to the federation protocol and the managed resources, facilitate interoperability on a semantic level and separate use-case-specific implementations from a common core. To provide a proof-of-concept and establish a basis for the evaluation, a concrete implementation was developed. By following a requirement- and test-driven approach, the fundamental functionalities were implemented as an extensible, open-source framework that additionally lays the foundation for other developers to plugin further extensions.

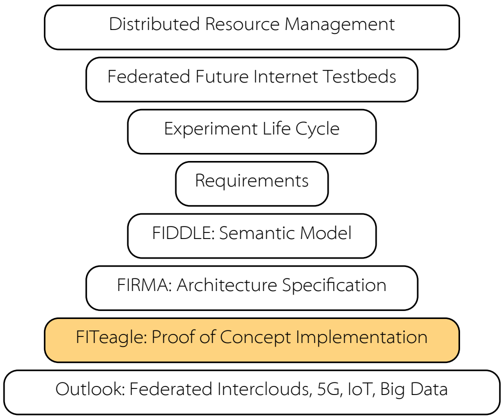


Figure 6.1: Placement of the reference implementation in the structure of research

Implementations

Two major implementations were carried out during in the course of this thesis, with a focus on GENI- and FIRE-related APIs and semantic models. First, a translation mechanism called *omnlib* was developed to convert between the OMN ontologies and different data models, in particular GENI RSpecs, TOSCA, and the YANG-based IETF proposal for modeling VNFs [t92]. Second, a FIRMA-compliant, semantic-driven resource management framework called *FITeagle* was developed that uses this translation mechanism within some delivery mechanisms and resource adapters.

Structure of Research

As indicated in Figure 6.1, these implementations are based on the theoretical constructs presented in Chapters 4 and 5. Parts of this chapter have been published before in [a10, a22–a24].

6.2 Technology Selection

Overview

The implementation presented here has three main goals: (i) increasing ease of use by reducing the number of technologies involved, (ii) enabling reusability by abstracting from specific APIs, and (iii) focusing on managing semantic graphs rather than specific resources. While the implemented framework isn't intended to offer an exhaustive solution to all issues related to federated experimentation, it does offer an extensible foundation to further implement and study semantic resource management mechanisms.

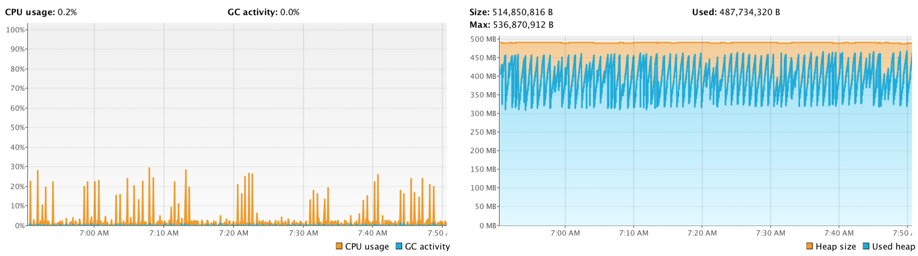


Figure 6.2: Hardware requirements based on the Oracle Java VisualVM Profiler

Software Requirements

Although each Microservice could have been implemented using different programming languages, Java 8 was chosen for the sake of consistency and to allow code to be executed on most operating systems capable of running a Java Virtual Machine (JVM), including Windows, UNIX, and Linux. The overall framework is a Message Oriented Middleware (MOM) and is divided into several Apache Maven 3.1[1] modules. Artifacts are published using Sonatype Nexus 2.11[2] and sources are hosted at GitHub[3], and tested after each commit using JUnit 4.12 and the Travis[4] Continous Integration (CI) environment. For the technical segmentation of the code Java Platform Enterprise Edition (J2EE) modules were developed and to deploy them using the WildFly 8.2[5] application server. An alternative would have been the Java Specification Request (JSR) 277 Java module system, initiated in 2005 and based on existing work conducted by the Open Service Gateway Initiative (OSGi) [p155]. The JSR release, however, was deferred and still is not available. For describing messages, the OWL-based OMN information model is serialized using TTL, transported within FITeagle using the Java Message Service (JMS) provider HornetQ 2.4[6], parsed by Apache Jena 2.12[7], persisted with Sesame 2.8[8], and converted from/into GENI RSpecs using the latest version of *omnlib*.

Hardware Requirements

As a result of the chosen software environment and implemented modules, the hardware used must provide at least 500 MB hard disk space for the WildFly environment and 300 MB for the FITeagle core packages and a 500 MHz CPU. In Figure 6.2 CPU and RAM utilization is depicted, based on the repetitive life cycle management of a virtual resource. In addition to about 500 MB static memory for the JVM, a heap space memory of 300 to 500 MB is required. Further, when tested on a MacBook Pro (Retina, 13-inch, Late 2012) with a 2,5 GHz Intel Core i5, OS X 10.11.1 and Java 1.8.0_45, the framework requires about 20% of the CPU.

Modules

Depending on the number of deployed modules and requests, these requirements increase. A number of reusable Microservices have been implemented, based on requirements of different research contexts. As noted in Chapter 5, each module can offer its service via a number of APIs and, within FITeagle, the JMS interface is compulsory. Communication to potential external resources is abstracted to allow their functionalities to be emulated for testing purposes. Figure 6.3 depicts an overview of selected modules and their relationships. The most relevant implementations are described in more detail in the following sections.

6.3 Translator

Intro

The *omnlib* translation mechanism was implemented in order to facilitate the transition of nonsemantic management systems towards those using graph-based information models, the adoption and advancement of the developed ontology, and to integrate semantic management systems into the GENI context. The translator provides support for translating locally used structured, semistructured and unstructured data models, such as GENI RSpecs, into RDF-based graphs and back. This approach has several advantages: (i) it automates and speeds up the process of converting data that is not using RDF; (ii) it encourages users and developers to migrate their systems to using

[1] http://maven.apache.org
[2] http://fiteagle.org/maven
[3] https://github.com/fiteagle and http://github.com/w3c/omn
[4] https://travis-ci.org/fiteagle and https://travis-ci.org/w3c/omn
[5] http://wildfly.org
[6] http://hornetq.jboss.org
[7] http://jena.apache.org
[8] http://rdf4j.org

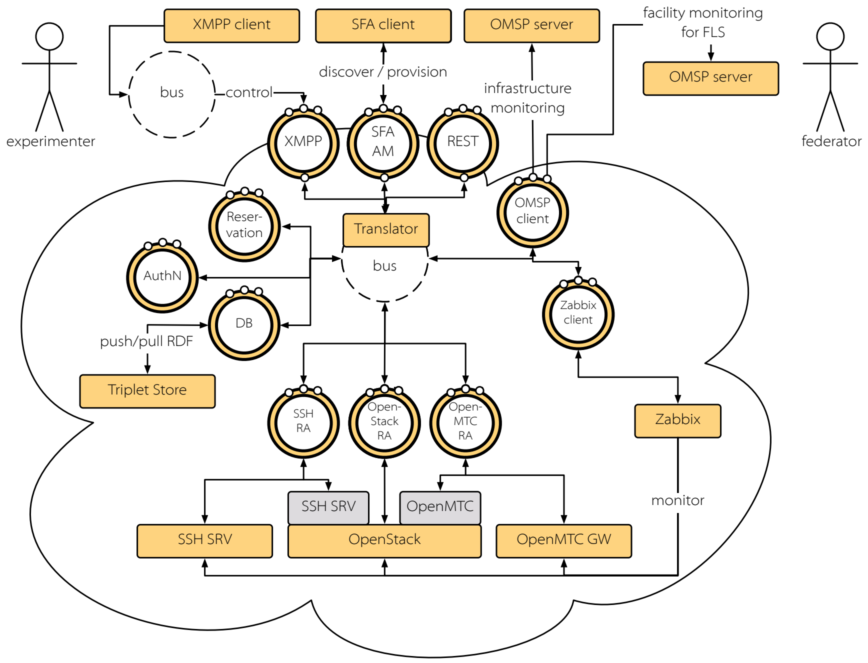


Figure 6.3: Selected FITeagle modules

Semantic Web technologies; and (iii) it ensures that the quality of generated RDF data corresponds to its counterpart data in the original system. With the assistance of the translator, legacy data formats used in interfaces such as SFA can be converted to and from OMN-based graphs and therefore integrated into the system. As the requirements of the developed ontology, architecture and framework are mainly based on research conducted in the FIRE context, a number of mappings between GENI RSpecs and the semantic model will be presented in this section. Besides for RSpecs, extensions have been implemented to support the translation of TOSCA and YANG models for the management of VNFs.

Related Work

Similar to this approach is the translation mechanism used within the ExoGENI deployment. The control framework ORCA used in ExoGENI employs the NDL-OWL semantic model for describing resource allocation policies, path computation, and topology embedding. The framework offers an SFA AM v2 API with a limited set of RSpec expressions and extensions. A stateless proxy mechanism was developed to translate between RSpecs and NDL-OWL and to validate user requests based on semantic constraints. While this work focused only on the translation between an internal ontology and SFA AM messages to reserve, provision and release resources, the corresponding research findings [t17] provided valuable starting points for OMN and the translator.

Approach

Analogous to this approach, the developed library statelessly converts, among others, GENI RSpec XML documents into RDF-based models and back using the OMN ontology by parsing the XML tree and converting the elements and attributes into their corresponding classes and properties. To give a better understanding of this translation process, some illustrative examples for conversions of Advertisements, Requests and Manifests are provided.

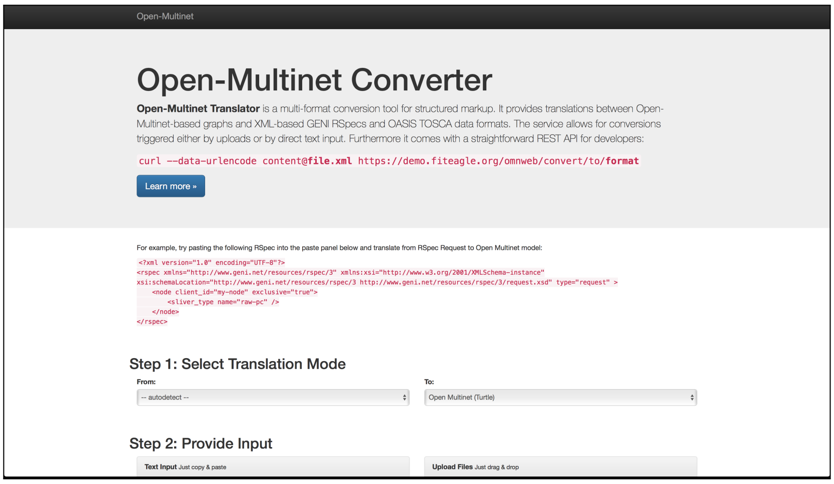


Figure 6.4: Translator Web GUI

Implementation

The implementation of the translation tool followed a TDD approach, is included in a CI environment with test coverage analytics, and is offered as a Java-based open-source library (*omnlib*) in a public repository under the W3C umbrella. It uses JAXB (generated Java classes based on given schemas) and Apache Jena (using imported ontologies) to map between XML, RDF and Java objects. The translator further supports a number of different APIs (cf. Figure 6.5): (i) a native API to be included in other Java projects; (ii) a CLI to be used within other applications; and (iii) a REST-based API to run as a Web service and Web GUI (cf. Figure 6.4) to return RDF/XML-, TTL- and JSON-LD-serialized versions, similar to the *RDF translator* [m14].

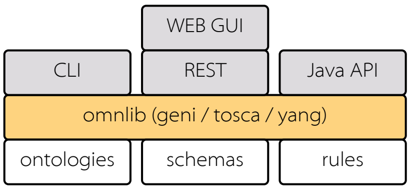


Figure 6.5: Translator architecture

Reasoning

In addition to the functional implementation of mappings between different models, the translator further allows rules to be defined to reason over an RDF graph. A rule is an appropriately defined set of axioms from which additional implicit information can be derived. In Listing 6.1 an exemplary forward-chaining rule is presented that is divided into two parts. In Line 1 an arbitrary identifier for the rule is given, and in Lines 2 to 4 a matching query is defined in this case that *omn-resource:Node*s have a label and are managed by the Netmode AM. In Lines 6 to 7 knowledge about the geographic positioning of these nodes is injected by extending the graph accordingly. This concept can be used to add background knowledge about resources within the federation and, in conjunction with further backward-chaining [p225] rules, can allow for more complex queries and information harmonization without changing the translator or the way infrastructures expose their data.

Listing 6.1: Reasoning rule (Apache Jena RuleSet)

```
[ruleToAddGeoData:
(?node rdf:type <http://open-multinet.info/ontology/omn-resource#
    Node>)
(?node <http://open-multinet.info/ontology/omn-lifecycle#managedBy>
     <urn:publicid:IDN+omf:netmode+authority+cm>)
(?node rdfs:label ?name)
->
(?node <http://www.w3.org/2003/01/geo/wgs84_pos#lat> "37.9813"^^xsd
    :float)
(?node <http://www.w3.org/2003/01/geo/wgs84_pos#long> "23.7827"^^
    xsd:float)
]
```

Motor The listings in the subsequent subsections, published in [a24], serve two main purposes. First, they present detailed and understandable examples how GENI RSpecs are translated into OMN RDF graphs. Second, they demonstrate how to uniquely specify any kind of resource. For this an artificial example, also used in the documentation of the experiment control system OMF, has been adopted: a resource *Garage* manages resources called *Motor*.

6.3.1 Advertisement

Input Listing 6.2 shows a simple GENI Advertisement RSpec (Line 1) used to publish available resources within a GENI federation. The example contains a single node (Line 2) of type *MotorGarage* (Line 6) that can provision the sliver type *Motor* (Line 7). While *hardware_type* and *sliver_type* are typically simple strings (such as “rawpc”) within GENI projects, unique URIs are used here to provide machine-interpretable information.

Listing 6.2: RSpec Advertisement (in)

```
1  <rspec xmlns="http://www.geni.net/resources/rspec/3" type="
       advertisement">
2    <node
3      component_manager_id="urn:publicid:IDN+testbed.example.org+
           authority+cm"
4      component_id="http://testbed.example.org/resources/motorgarage
           -1"
5      exclusive="false">
6      <hardware_type name="http://open-multinet.info/ontology/
           resources/motorgarage#MotorGarage" />
7      <sliver_type name="http://open-multinet.info/ontology/resources
           /motor#Motor" />
8      <available now="true" />
9      <location longitude="-7.491667" latitude="110.004444" country="
           ID" />
10   </node>
11 </rspec>
```

Output Based on this input, Listing 6.3 shows the converted graph, serialized in TTL. The overall approach is to define an *omn:Topology*, here its subclass *omn-lifecycle:Offering*

(Line 1), that links to the offered resources (Line 2). Each resource is an individual of a specific type that can provision (*omn-lifecycle:canImplement*, Line 7) one or more specific types. Other information, such as the location, is translated by reusing well-known existing ontologies (Lines 12 to 17).

Listing 6.3: OMN Offering

```
1  <urn:uuid:49fa0240> a omn-lifecycle:Offering ;
2    omn:hasResource <http://testbed.example.org/resources/motorgarage
         -1> .
3
4  <http://testbed.example.org/resources/motorgarage-1>
5    a motorgarage:MotorGarage, omn-resource:Node ;
6    omn:isResourceOf <urn:uuid:49fa0240> ;
7    omn-lifecycle:canImplement motor:Motor ;
8    omn-lifecycle:managedBy <urn:publicid:IDN+testbed.example.org+
         authority+cm> ;
9    omn-resource:hasHardwareType motorgarage:MotorGarage ;
10   omn-resource:isAvailable true ;
11   omn-resource:isExclusive false ;
12   omn-resource:hasLocation <urn:uuid:1aa53e5d> .
13
14   <urn:uuid:1aa53e5d> a omn-resource:Location ;
15        geonames:countryCode "ID" ;
16        wgs84_pos:lat "110.004444" ;
17        wgs84_pos:long "-7.491667" .
```

Round-Trip

In order to demonstrate a complete round-trip translation, i.e. from RSpec XML to OMN RDF and back to RSpec XML, the Advertisement RSpec is shown once more in Listing 6.4. Note that this example has been statelessly generated based solely on the graph from Listing 6.3. All information has been converted, making Listing 6.2 and Listing 6.4 semantically equivalent.

Listing 6.4: RSpec Advertisement (out)

```
<rspec
   generated="2015-02-12T09:46:59.480+01:00"
   generated_by="omnlib"
   expires="2015-02-12T09:46:59.480+01:00"
   type="advertisement"
   xmlns="http://www.geni.net/resources/rspec/3">
   <node
       component_manager_id="urn:publicid:IDN+testbed.example.org+
           authority+cm"
       component_id="http://testbed.example.org/resources/
           motorgarage-1"
       component_name="motorgarage-1"
       exclusive="false">
       <hardware_type name="http://open-multinet.info/ontology/
           resources/motorgarage#MotorGarage"/>
       <sliver_type name="http://open-multinet.info/ontology/
           resources/motor#Motor"/>
```

```
        <location longitude="-7.491667" latitude="110.004444"
            country="ID"/>
        <available now="true"/>
    </node>
</rspec>
```

6.3.2 Request

Input After receiving an Advertisement RSpec, the next step is to request a specific subtopology. In Listing 6.5, such a simple Request RSpec is shown (Line 1). Again URIs are used to specify the type of resource requested from a specific node (Line 6). In order to be able to map the requested resource to the provisioned resource at a later stage, the *client_id* string is set (Line 5).

Listing 6.5: RSpec Request (in)

```
1 <rspec xmlns="http://www.geni.net/resources/rspec/3" type="request"
    >
2     <node
3         component_manager_id="urn:publicid:IDN+testbed.example.org+
              authority+cm"
4         component_id="urn:publicid:IDN+testbed.example.org+node+
              testbed.example.org%2Fresource%2Fmotorgarage-1"
5         client_id="myMotor">
6         <sliver_type name="http://open-multinet.info/ontology/
              resources/motor#Motor" />
7     </node>
8 </rspec>
```

Output The conversion process is again shown in Listing 6.6. An *omn:Topology*, here an *omn-lifecycle:Request* (Line 1), has been created with pointers to the requested resources (Line 2). The resource is an individual of the requested type (Line 4) that is implemented by a specific resource (Lines 7 to 8) and has the above mentioned identifier (Line 6). Note that the property *omn:implementedBy* is only set if the request contains information about where a sliver should be created, i.e. in case of a *bound* request. Otherwise an *unbound* request is sent that has to be processed further and enhanced by a resource mapping mechanism.

Listing 6.6: OMN Request

```
1 example:request a omn-lifecycle:Request ;
2   omn:hasResource example:myMotor .
3
4 example:myMotor a motor:Motor ;
5   omn:isResourceOf example:request ;
6   omn-lifecycle:hasID "myMotor" ;
7   omn-lifecycle:implementedBy
8     <http://testbed.example.org/resources/motorgarage-1> .
```

Round-Trip As shown in Listing 6.7, the generated Request RSpec, based on Listing 6.6, maps to the incoming RSpec shown in Listing 6.5.

Listing 6.7: RSpec Request (out)

```
<rspec
   generated="2015-02-12T09:54:18.484+01:00"
   generated_by="omnlib"
   type="request"
   xmlns="http://www.geni.net/resources/rspec/3">
   <node
      client_id="myMotor"
      component_id="http://testbed.example.org/resources/
          motorgarage-1">
      <sliver_type name="http://open-multinet.info/ontology/
          resources/motor#Motor"/>
   </node>
</rspec>
```

6.3.3 Manifest

Input

In Listing 6.8 a Manifest RSpec is shown. A Manifest RSpec is usually returned after the requested resource has been successfully allocated or provisioned.

Listing 6.8: RSpec Manifest (in)

```
<rspec xmlns="http://www.geni.net/resources/rspec/3" type="manifest
    ">
  <node component_manager_id="urn:publicid:IDN+testbed.example.org+
      authority+cm"
      component_id="http://testbed.example.org/resources/
          motorgarage-1"
      client_id="myMotor"
      sliver_id="urn:publicid:IDN+testbed.example.org+sliver+http
          %3A%2F%2Ftestbed.example.org%2Fmotorgarage-1%2Fmotor-1"
          >
      <sliver_type name="http://open-multinet.info/ontology/
          resources/motor#Motor" />
  </node>
</rspec>
```

Output

As shown in Listing 6.9 an *omn:Topology*, here an *omn-lifecycle:Manifest* (Line 1), has once again been defined to identify the provisioned resources (Line 2). The relevant individual is of the requested type (Line 5), is further identified with the client identifier and is implemented/provisioned by a specific resource.

Listing 6.9: OMN Manifest

```
1 example:manifest a omn-lifecycle:Manifest ;
2   omn:hasResource <http://testbed.example.org/motorgarage-1/motor
      -1> .
3
4 <http://testbed.example.org/motorgarage-1/motor-1>
5   a motor:Motor ;
6   omn-lifecycle:hasID "myMotor" ;
7   omn-lifecycle:implementedBy
8     <http://testbed.example.org/resources/motorgarage-1> .
```

Round-Trip The translation into a GENI Manifest RSpec is shown in Listing 6.10. Note that the *client_id* identifies the requested resource and the *sliver_id*, the unique identifier of the newly created resource instance within the testbed, follows the GENI standard with implied semantics within a simple string. However, a URI is again used after the last "+" sign (Line 6) to allow internal semantic management of the resource without relying on GENI-related notions.

Listing 6.10: RSpec Manifest (out)

```
1 <rspec generated="2015-02-12T09:41:25.230+01:00" generated_by="
      omnlib" type="manifest" xmlns="http://www.geni.net/resources/
      rspec/3">
2   <node
3     client_id="myMotor"
4     component_id="http://testbed.example.org/resources/motorgarage
        -1"
5     component_name="motorgarage-1"
6     sliver_id="urn:publicid:IDN+testbed.example.org+sliver+http%3A
        %2F%2Ftestbed.example.org%2Fmotorgarage-1%2Fmotor-1">
7     <sliver_type name="http://open-multinet.info/ontology/resources
        /motor#Motor"/>
8   </node>
9 </rspec>
```

6.4 Framework

Intro As depicted in Figure 6.3, the translator described above can be used to support the implementation of different APIs that use nonsemantic models for incoming and outgoing messages. The actual management of the resulting semantic graph and its connected resources is implemented within several FITeagle modules that communicate RDF-based information with each other over a message bus. The implemented functionalities, required by several FIRE projects, are the subject of this section. An exemplary component messaging workflow is shown in Figure 6.6 to provide an overview of the internal communication flow.

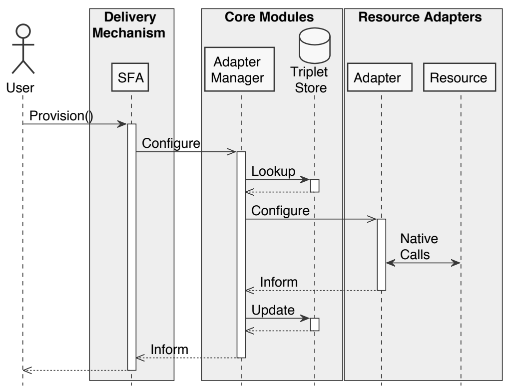


Figure 6.6: Exemplary component messaging workflow (provision request)

6.4.1 Message Bus

Message Types

The message bus is of central importance to the developed system, as each module is loosely decoupled from each other only exchanges notifications over the bus. The content of the messages follows the OMN information model and is serialized in TTL. A number of message types have been defined to distinguish different notifications. In Table 6.1 a list of message types is briefly described and mapped to message types used in other protocols. In total, 11 different operations are specified that are mapped to five message types: *Get*, *Create*, *Configure*, *Release*, and *Inform*. Most of these types have corresponding method calls or messages, depending on the delivery mechanism API used.

Get

The *Get* message type is used, along with the relevant OMN concept, in order to describe a whole infrastructure, get a list of resources, and describe a provisioned topology and the status of its resources, This maps directly to REST *GET*, FRCP *Request* and SPARQL *Describe* messages. In SFA the *GetVersion*, *ListResources*, *Describe*, and *Status* method calls provide equivalent functionalities.

Create

The SFA calls *Register* and *Allocate* are both mapped to the *Create* message type with either an *omn:Topology* or *omn:Reservation* concept. In REST, FRCP, and SPARQL the corresponding messages are *POST*, *Create*, and *Insert* respectively. These messages do not involve communication with any resource adapter, as a topology is only created and reserved, not provisioned.

Configure

For the resource provisioning and configuration phase, the *Configure* messages are forwarded to the resource adapters. *Configure* is equivalent to *PUT* messages in REST, *Configure* in FRCP, and *Delete* and *Insert* operators to update the graph using SPARQL. As a *Configure* message is also used to extend the reservation of a topology, the relevant method calls in SFA are *Renew*, *Provision*, and *PerformOperationalAction*.

Release

To release the reserved and provisioned resources, a *Release* message is sent to the adapters involved. This corresponds to *Delete* messages and method calls in SFA, REST and SPARQL, while FRCP also uses *Release*.

Inform

Finally, to push updates such as monitoring information about a resource from an adapter, the *Inform* type is introduced, analogous to the FRCP message. As SFA, REST and SPARQL are unidirectional protocols, they do not support notifications natively.

6.4.2 North: Delivery Mechanisms

FIRE API's

To allow the prototype to carry out all life cycle phases within the GENI and FIRE federations, the translator translates the different data models and the related interfaces have been implemented. Besides a GENI SFA AM API, the proof-of-concept implementation provides a ProtoGENI [t67] SFA Slice Authority (SA) API to act as an IdP for AuthN and AuthZ, an OMSP interface to receive monitoring information about resources, and a native REST interface.

Further API's

Besides these FI APIs, an initial XMPP-based IEEE P2302 interface has been implemented for Intercloud federations. The implementation of this and the native REST interface show the independence of the delivery mechanisms and the applicability of the implementation to other fields of application.

6.4.2.1 SFA

Overview

As described above, the focus of the development process was the implementation of SFA-compliant interfaces, in particular the methods in the GENI AM API version 3

Table 6.1: FITeagle notification types in relation to other protocols [a23]

FITeagle	**SFA**	**REST**	**FRCP**	**SPARQL**	**OMN**	**Description**
Get	GetVersion	GET	Request	Describe	*omn:Infrastructure*	Describe infrastructure
Get	ListResources	GET	Request	Describe	*omn:Resource/Service*	List resources
Create	Register	POST	Create	Insert	*omn:Topology*	Create a topology
Create	Allocate	POST	Create	Insert	*omn:Reserveration*	Reserve an instance
Get	Describe	GET	Request	Describe	*omn:Topology*	Describe a topology
Configure	Renew	PUT	Configure	Del/Ins	*omn:Reserveration*	Extend a reservation
Configure	Provision	PUT	Configure	Del/Ins	*omn:Resource/Service*	Create/provision instance
Get	Status	GET	Request	Describe	*omn:Status*	Get instance status
Configure	Perf.Op.Act.	PUT	Configure	Del/Ins	*omn:Attribute*	Change instance
Release	Delete	DELETE	Release	Delete	*omn:Topology*	Delete instance
Inform	—	—	Inform	—	*omn:Resource/Service*	Push update

specification for resource discovery, provisioning, and release and the methods used in the ProtoGENI SA API version 1 specification for user and slice management, which are invoked by clients such as jFed, MySlice and Omni. In Figure 6.7 the overall architecture of the implemented SFA modules is shown.

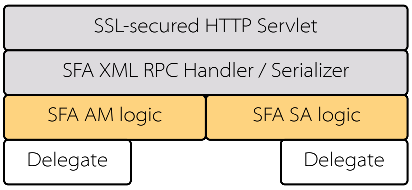


Figure 6.7: SFA delivery mechanism architecture

Both modules share the same SSL-secured HTTP servlet to communicate with external clients. The server authenticates itself by providing an X.509 certificate and only accepts incoming messages that are signed by a certificate that has been issued by a trusted Certificate Authority (CA), using the mechanisms provided by J2EE for client and server AuthN. The embedded privileges are then evaluated to approve or deny an incoming request. Next, an SFA message-specific layer handles XML-RPC messages and parses the relevant data structures. Depending on the API, the client communicates with, the object models are then forwarded to the AM or SA implementation. *Servlet*

The SFA AM implementation contains the logic needed to handle *GetVersion*, *ListResources*, *Allocate*, *Describe*, *Renew*, *Provision*, *Status*, *PerformOperationalAction*, *Delete*, and *Shutdown* method calls. This includes proper error handling and the construction of SFA-specific data models. The actual implementation of the related action is passed to a specific delegate. For testing purposes, such a delegate would be a stub that returns static values. In production the delegate is responsible for the construction of relevant RDF models and sending and receiving messages to and from the message bus. *AM*

In an analog manner, the SFA SA implementation contains the logic needed to handle *GetVersion*, *GetCredential*, *Register*, *RenewSlice*, *Resolve*, and *GetKeys* method calls. The core functionality is forwarded to the relevant delegates using the message bus, including the reservation module for slice creation, and a module for AuthN and AuthZ decisions. *SA*

6.4.2.2 Native

With the SFA interfaces, the discovery, selection, reservation, provisioning and termination phases of the life cycle have been implemented, including AuthN and AuthZ. Although the method call *PerformOperationalAction* could be used to communicate with the instantiated resources, in practice other protocols such as FRCP or REST-based APIs are exposed in order to control the experiment. *Overview*

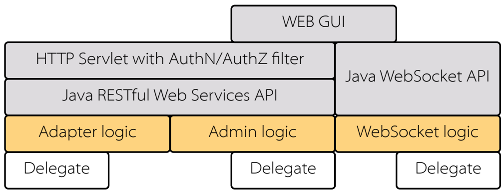


Figure 6.8: Native delivery mechanism architecture

WebSocket As depicted in Figure 6.8, FITeagle additionally offers two native APIs to access and modify information. Both are used in GUIs, such as the administrative one shown in Figure 6.9. The WebSocket API is mainly responsible for notifications and for logging and visualizing the information exchanged within JMS, such as monitoring data pushed by adapters. However, it can also forward RDF data to the message bus to control an experiment.

REST As WebSockets are mainly used as an administrative bidirectional interface, a Java API for RESTful Web Services (JAX-RS) interface has additionally been implemented. Depending on the incoming request, a user first needs to be identified to authorize the action. The API is used to access administrative functionalities; to control attributes of resources within an experiment; and to provide dereferenceable URIs for available and provisioned resources, following the LOD principles, including JSON with Padding (JSONP) filters for clients such as LodLive[9] [p33].

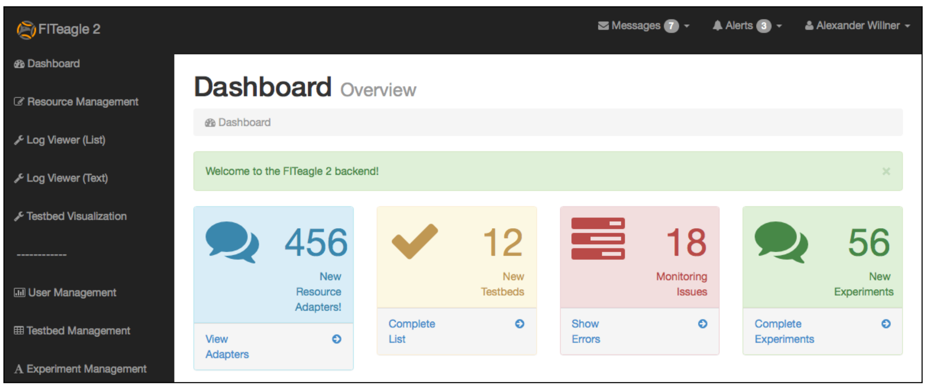


Figure 6.9: FITeagle administrative GUI

6.4.2.3 OMSP

Overview Finally, as in the testbed community the OMSP protocol is used to transfer monitoring data in the FI community, an OMSP-compliant delivery mechanism was implemented. This adds two distinct features to an instance of FITeagle. First, monitoring information about resources within an infrastructure can be provided by an external service. Although an adapter is responsible for monitoring its own resources, in practice monitoring systems are often already in place. Therefore, an external wrapper mechanism can be used to translate local monitoring information and push OMSP streams about the given resource to FITeagle. Second, besides providing access to resources offered by

[9] http://lodlive.it

an infrastructure, FITeagle can further act as an information broker to collect and link information about resources within a federation.

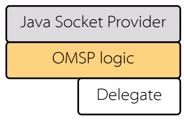


Figure 6.10: OMSP delivery mechanism architecture

Details

The OMSP module listens to a TCP socket to retrieve OMSP streams and its architecture is depicted in Figure 6.10. CSVs sent to this module are expected to follow a predefined RDF/OMSP serialization to encode semantic monitoring information as shown in Listing 6.11. In the header (Lines 1 to 3) metainformation is defined, such as the protocol version, message type and schema (triplets in this case). The data is then transferred continuously beginning with Line 4. In this way, the module can parse the information and create an RDF-based information model to send an *Inform* message through a delegate to the message bus.

Listing 6.11: RDF/OMSP serialization (excerpt)

```
1 protocol: 4
2 schema: 1 mystream subject:string predicate:string object:string
3 content: text
4 1.27909302711 1 0 <subjectURI> <predicateURI> <objectURI>
5 1.27919507027 1 1 <subjectURI> <predicateURI> "literal"
```

6.4.2.4 IEEE Intercloud

Overview

Finally, to demonstrate the applicability of FITeagle in other contexts, an initial XMPP-based delivery mechanism was implemented and presented in [a10]. This initial prototype for the IEEE P2302 working group exchanges RDF-encoded information about resources that are added and removed on demand.

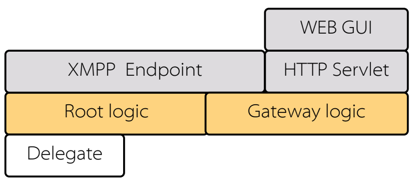


Figure 6.11: Intercloud delivery mechanism

Details

Following the P2302 definition, functionality has been divided into two components which communicate with each other via XMPP (cf. Figure 6.11). The root logic listens for requests and accordingly updates information about available resources within the cloud in the SPARQL database by dispatching *Create* and *Delete* messages to the FITeagle bus. The gateway provides a GUI to create, update and delete resources locally. Each gateway then sends the updates to the root, which in turn can be queried using SPARQL to identify the existing resources within the Intercloud federation.

6.4.3 West: Core Modules

Overview Besides shared libraries for common tasks, such as SPARQL-based communication and certificate validation, a number of Microservices are included as part of the *Core Modules* area to provide the required functionalities for the above described *Delivery Mechanisms*. These services includes a user management module to issue client certificates, a resource adapter manager for handling available resources, a reservation module to manage allocation requests and an orchestrator to handle resource provisioning and configuration requests.

6.4.3.1 User Management

User Management While external access decisions using federation-related interfaces, such as the SFA AM API, rely on PKI mechanisms and trusted CAs, local user management functionalities were implemented for two reasons. First, these enabled fine-grained, XACML-based authorization rules for given attributes assigned to the incoming request certificate. Depending on the CA issuer and the user's roles, access to specific resources could be granted or denied. Second, the capability to get, add, delete, and update users, and to assign, remove, and rename keys, certificates and roles allows FITeagle to act as an infrastructure or federation IdP itself. Therefore, each FITeagle installation is a CA and SA itself, which may or may not be trusted by other federation members. As described in Section 2.4.8, user management can further be extended by implementing federation-wide AuthZ delegation mechanisms and alternative AuthN protocols.

6.4.3.2 Reservation

Reservation The availability of resources within an infrastructure is limited and therefore mechanisms to manage the number of allocation requests are needed. As a result, a reservation module has been implemented that parses incoming *Create omn:Reservation* messages to map the maximal available number of resources of a specific type with the already reserved instances and the newly requested ones within a given period of time. This includes functionality to persist the schedules, to modify the status of a requested *omn:Topology* (e.g. to *omn-lifecycle:Allocated*), and to release a reservation after a given expiration time (i.e. setting its status to *omn-lifecycle:Unallocated*). However, as highlighted in Section 2.4.3, reservations could include more complex in advance schedules that are not only limited to time properties but might further include resource-specific capabilities, such as bandwidth within a network.

6.4.3.3 Orchestration

Orchestration Incoming messages to create, describe, configure, or delete resources are handled by an orchestration module. As incoming messages are RDF-encoded graphs with potentially arbitrary extensions, resource information is only extracted that have already been assigned to an *omn-lifecycle:Allocated omn:Topology* and is of type *omn:Resource*. As sketched in Section 2.4.4, this could include the evaluation of dependencies between resources that imply a specific instantiation order and message forwarding, e.g. for SFC. Further, as not every kind of *omn:Resource* can be offered by an infrastructure, the concrete type of resource must be *omn-lifecycle:implementedBy* a corresponding adapter instance to which the separate requests are then forwarded.

6.4.3.4 Adapter Management

Adapters As described in Section 5.2.5, communication with specific resources is abstracted by introducing resource adapters. These adapters in turn have to register at the adapter manager by sending their metadata as an RDF graph to distribute information about resources on offer. The manager extracts information about the resources an adapter can implement (*omn-lifecycle:canImplement*) and attaches corresponding *omn:hasResource* properties to the *omn:Infrastructure* graph, which, for example, is evaluated by the SFA AM *ListResources* method call. Within this process, the adapter manager can add further metainformation about the offered resources, besides their relationship to a specific infrastructure and, transitively, to a specific federation, such as geographic information based on the FITeagle default configuration.

6.4.4 South: Resource Adapters

Overview A number of *Resource Adapters* were implemented within the context of this thesis and these are described in this section. Others have been implemented by testbed owners or in the course of student projects. Their functionality can range from simply advertising and provisioning a specific resource type, to the management of various types, including their control and monitoring. An abstract adapter class from which new adapters are derived, offers basic functionality for implementing multiple APIs, error handling, and general communication to register, deregister or update adapter information.

6.4.4.1 Motor

Testing As described in Section 6.3, the *Motor Adapter* is used for testing and demonstration purposes. It offers support for describing, publishing, provisioning, controlling, monitoring and releasing virtual motors and acts as a reference implementation and template for other adapters. In this way, new concepts and design changes can first be evaluated at scale before integrating them into the code base.

6.4.4.2 SSH Adapter

Login Although the motor adapter is useful within the development phase, the most common functionality provided by many testbeds within the GENI and FIRE federations is SSH access to physical or virtual machines. Therefore, an *SSH Adapter* was implemented that extracts a list of usernames with public keys from the graph to setup and delete logins on a Linux machine. Information in the graph also includes the path to a script to be executed after login, environment variables such as the type of shell, and a pointer to a document identified by an Uniform Resource Locator (URL) to be downloaded to a given target folder. To setup the user, the adapter needs the IP or Domain Name System (DNS) of the target machine, a username with root permissions and either a preconfigured password or private key.

6.4.4.3 OpenStack Adapter

VMs Closely related to the SSH adapter, the *OpenStack Adapter* makes it possible for users to instantiate VMs on demand. To offer users a list of available images, default flavors are configured that include properties such as image names, number of Central Processing Units (CPUs) and the size of available memory. The adapter itself communicates via REST to the OpenStack Keystone, Nova, and Glance APIs for provisioning and

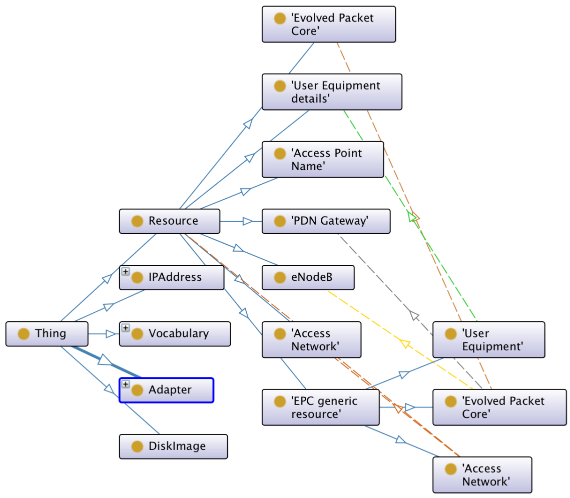


Figure 6.12: Model to describe an EPC topology

configuration. Depending on the requested topology, endpoint IPs are returned and SSH logins are setup. However, as plain SSH access to machines is often not possible due to security reasons or is not the type of service a testbed wants to offer, other services contained in the selected images can be used remotely.

OpenMTC

One example of a service that can be provisioned within an OpenStack image is M2M-as-a-Service. This includes the instantiation and configuration of an M2M communication substrate composed of Open Machine Type Communication (OpenMTC) [p50] server and gateway combinations, which can be complemented by additional images using these services, such as Smart-City-as-a-Service platform images. Connection to physical sensor and actuation devices is handled by sensor-/actuator-specific adapters. Depending on the device type, the adapter could send data to configured endpoints or change device parameters.

6.4.4.4 EPC Adapters

Functionality

Similar to M2M resources described above, adapters related to setting up an EPC topology have been implemented. As indicated in Figure 6.12, a typical experiment would include access to a User Equipment (UE), the provisioning and control of an Open Evolved Packet Core (OpenEPC) instance, and the configuration of the Access Network (AN), composed of a wireless Access Point (AP) and an eNodeB. Access to the AP is then handled by an *SSH Adapter* and an *EPC Adapter* is responsible for starting, stopping and configuring an OpenEPC setup. The management of the AN is done via an *Automatic Configuration Server (ACS) Adapter*, which communicates with the eNodeB using the bidirectional, SOAP-based TR-069 communication protocol, or via an *Attenuator Adapter* that communicates via TCP with a signal-strength attenuator.

6.4.4.5 TOSCA Adapter

NFV/VNF

While Physical Network Functions (PNFs) can be provisioned to build an OpenEPC topology and to conduct fundamental EPC-related experiments, such a setup does not reflect the infrastructure that mobile network operators will deploy for the 5th Generation Mobile Network (5G) [t48]. As the bandwidths, configurability and connectivity demands on network operators increase, the infrastructure must be frequently extended by adding new components and removing outdated ones. To address issues with the cost-intensive and error-prone process of rolling out new network services, the concept of Network Function Virtualization (NFV) [t18, t80] was introduced. By virtualizing network functions, software is decoupled from dedicated hardware and all VNFs should be able to run on industry-standard, high-volume servers, switches and storage. Among other benefits, new network functions can then be installed remotely and automatically. As a result, numerous interfaces for and implementations of management systems for NFV have been developed. Examples are the IETF proposal for a VNF [t92] data model, the OASIS TOSCA, the OpenStack Heat Orchestration Templates (HOTs), or the Amazon Web Services (AWS) CloudFormation model.

OpenSDNCore

Therefore, in order to offer 5G- and M2M-related VNFs to experimenters, a *TOSCA Adapter* has been implemented to communicate with a TOSCA-compatible orchestration framework. The adapter was tested with the ETSI MANO-compliant Network Function Virtualization Orchestrator (NFVO) OpenSDNCore[10] Orchestrator, which uses OpenStack as the underlying Virtual Network Function Manager (VNFM) and has recently been published as open source under the name OpenBaton[11]. It contains the required components to operate an NFVI and the TOSCA Virtualised Network Function Descriptor (VNFD) model can be used to specify the requested Network Services (NSs), based on Open5GCore[12] components.

6.4.5 East: Service Integrators

Overview

As described in Section 5.2.6, *Service Integrators* can be used to share information related to the execution of the experiment life cycle with services internal or external to the infrastructure. Within FITeagle one integrator related to monitoring services was developed.

Monitoring

As monitoring information about resources can be as heterogeneous as the resources themselves, gathering and modeling monitoring data is the responsibility of the relevant adapter. This can either be implemented directly within the adapter or the adapter could communicate with external monitoring services, as resources in managed infrastructures are often already observed by existing systems. As indicated in Figure 6.3, the implemented *Monitoring Service Integration* listens to the message bus for life-cycle related messages and exports OMSP streams based on information extracted from the monitoring system Zabbix, which examines a number of metrics related to OpenStack-based resources. These data are pushed to a federation-wide facility monitoring system for FLS and, if the experimenter requested measurement data, the information is sent to other OMSP servers as well.

6.5 Conclusion

Summary

In this chapter, the semantic-based open-source framework FITeagle was introduced. FITeagle is agnostic with regards to context-specific northbound and resource-specific southbound APIs. A breakdown of the employed technologies and the implemented

[10] http://opensdncore.org
[11] http://openbaton.org
[12] http://open5gcore.org

modules was given. The available delivery mechanisms, core modules, adapters and service integrators were described, along with a closer characterization of the translator mechanism. They were used to provide arbitrary resources to the GENI and FIRE federations by SFA and OMSP interfaces, and examples ranged from artificial *Motors* to interconnected VNFs. FITeagle represents a reference implementation of the FIRMA design and, as a result, shows the feasibility of this architecture and provides an extensible framework for further work.

Outlook Potential extensions related to the GENI and FIRE context include the implementation of an FRCP-compliant interface and the implementation of the next version of the SFA AM API, called Federation Aggregate Manager API. In this context, further research could be conducted with respect to extension and more efficient serialization of the OMN information model. An example for the latter is the Header, Dictionary, Triples (HDT) [p74] serialization, which could improve the overall communication performance. The ontology set will be extended within the W3C Federated Infrastructure Community Group to extend the concept to Software Defined Infrastructures (SDIs) in general. Here a focus can be set on the definition of 5G-specific NFV and SDN ontologies for complex SFC and orchestration. Related to this and due to its estimated market value of 2.75 billion USD in 2019 [t60], the definition and implementation of Metro Ethernet Forum (MEF) Lifecycle Service Orchestration (LSO) [t85] related delivery mechanisms for service orchestration on top of Open Networking Foundation (ONF) SDN and ETSI NFV MANO interfaces are of further interest.

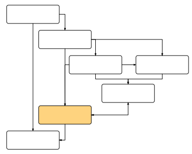
Next In the next chapter, the implemented software and models described above will be evaluated. With the objective to analyze the applicability of the implementation for the given context, an experimental validation, a performance evaluation, an observational validation, and a code verification will be carried out.

Chapter 7

Evaluation

7.1 Introduction

Overview In Chapters 4 to 6, the three main contributions of this thesis were described. Based on this, the applicability of the work presented for the FI experimentation use case will be evaluated in this chapter. As described in [p124, p233], information systems in general can be judged by observational, analytical, experimental, and descriptive evaluation, using modeling, simulation, or measurement approaches. Software implementations, in particular, can be verified or validated [p154, p158] (cf. Figure 7.1). Briefly, software verification ensures that a component behaves as expected using static code analyzers or dynamic unit tests; software validation assesses whether a certain design fits its purpose and meets the relevant requirements. Further, quantitative characteristics of an implementation can be analyzed to evaluate its applicability for current and future utilization. Finally, since the core of the system developed in this thesis is based on a semantic information model, an ontology can be validated either by evolution, logic, or metrics [p215].



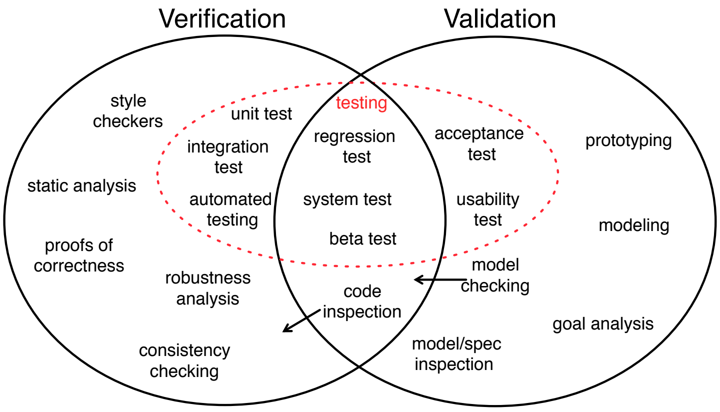


Figure 7.1: Choice of verification and validation techniques [m4] (based on [p65])

Approaches As a result, in this chapter a number of different techniques will be used. First, experimental validation ensures that the whole FI experiment life cycle can be modeled and executed by the tools designed here and that the implementation is compliant with the relevant standards. Second, a performance evaluation of the translator and the framework confirms their applicability for input sizes that are typical for the given use cases. Third, within an observational validation, the operative readiness of the implementations are demonstrated within research projects and testbeds. Fourth, results of code verification mechanisms are presented to assess the quality of the written code.

Structure of Research As this chapter represents the evaluation of the work presented in the three preceding sections, all of the corresponding parts are highlighted in Figure 7.2. Some results have been published before in [a11, a22–a24], allowing ratification of the approach by the scientific community.

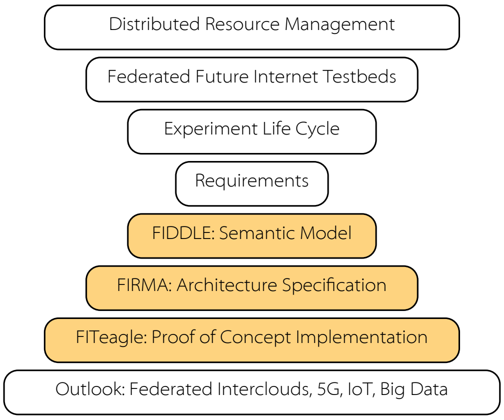


Figure 7.2: Placement of the evaluation in the structure of research

7.2 Experimental Validation

Overview

In this section, to demonstrate the functionality of the work designed in this thesis within the context of federated FI experimentation, a whole experiment life cycle is executed using the relevant user tools within a controlled experiment [p233]. Further, the standard compliancy of the reference implementation is validated by running an appropriate test suite. This procedure includes not only the discovery, selection, reservation, provisioning, control, monitoring, and release of a resource using FITeagle, but additionally validates the completeness of the ontology developed here. While “there is no single correct ontology-design methodology” [t50] and no single correct methodology for its evaluation, the approach follows the suggestion of [p171] in which an ontology can be evaluated by “its coverage of a particular domain and the richness, complexity and granularity of that coverage; the specific use cases, scenarios, requirements, applications, and data sources it was developed to address [. . .].”

7.2.1 Complete Life Cycle

Overview

The graph excerpt used for the execution of the exemplary life cycle is shown in Figure 7.3, using the RDF visualization tool LodLive. Within this section, this model will be constructed step by step and used for managing relevant information. It represents a *Localhost Federation* of type *omn-federation:Federation* with the member (*omn-federation:hasFederationMember*) *Localhost Organization* of type *omn-federation:FederationMember*. The *Localhost Organization* in turn administers (*omn-federation:administers*) a *Localhost Testbed* of type *omn-federation:Infrastructure* that offers a service (*omn:hasService*) *GENI AM v3 API*. The infrastructure further offers (*omn:hasResource*) a resource of type *motor:MotorGarage* that can instantiate (*omn-lifecycle:canImplement*) a resource of type *motor:Motor*, which is a subclass of *omn:Resource*. Based on this information a new resource, *motor1*, which is managed by the AM, will be allocated, provisioned, monitored, controlled, and deleted, using the integration concepts highlighted in Figure 4.11 and implemented by FITeagle.

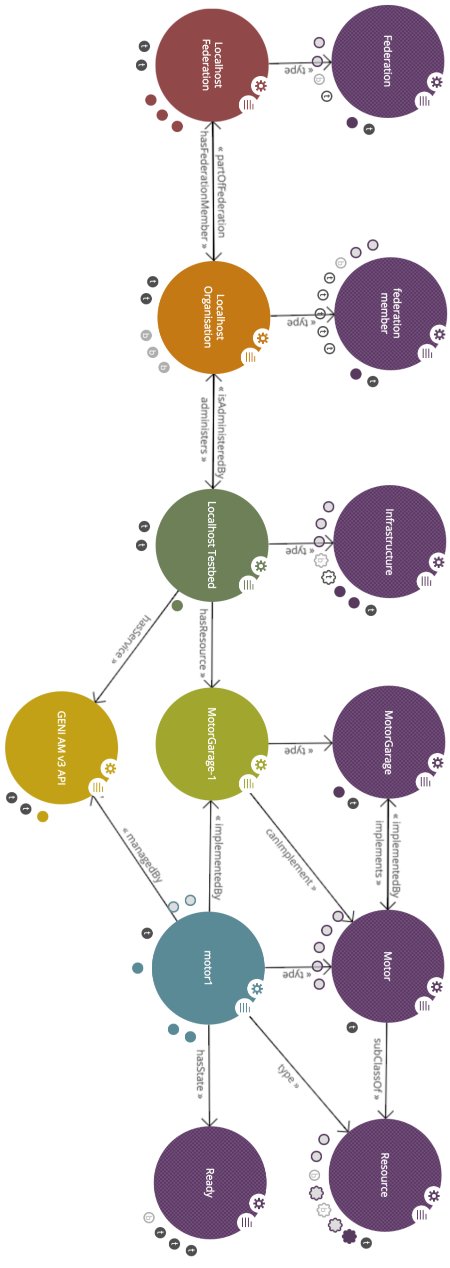


Figure 7.3: Graphical representation of the experiment under consideration

7.2.1.1 Federation

Overview

While not required for executing the Localhost life cycle example in this section, preliminary information about the federation and its members and infrastructures is needed. This data is usually encoded on Web sites that describe the federation in a human readable way and include further tutorials, descriptions and contact details.

Integration

As this information is not machine processable in its current form, work is ongoing on developing a GENI CH API that exports these metadata through an XML-RPC API. Further, the jFed tool currently retrieves this information from a manually maintained *authorities.xml* file. Following the approach of the Semantic Web, however, it would further be possible to embed a semantic graph using RDFa or to offer a SPARQL endpoint [p177].

Listing 7.1: OMN federation example

```
localhost:federation a omnfed:Federation, owl:NamedIndividual ;
    rdfs:label "Localhost␣Federation" ;
    schema:URL <http://localhost/> ;
        schema:logo <http://localhost/logo.jpg> ;
        schema:email <mailto:mail@localhost> ;
    omnfed:hasFederationMember localhost:organisation .

localhost:organisation a omnfed:FederationMember, owl:
     NamedIndividual ;
    rdfs:label "Localhost␣Organisation" ;
    schema:location [ a schema:Place ; wgs:lat "52.526"; wgs:long "
         13.314"] ;
    omnfed:partOfFederation localhost:federation ;
    omnfed:administers localhost:testbed .

localhost:testbed a omnfed:Infrastructure, owl:NamedIndividual;
    rdfs:label "Localhost␣Testbed" ;
    rdfs:seeAlso "https://localhost/" ;
    wgs:lat "-7.5508303" ;
    wgs:long "110.9850367" ;
    omnfed:isAdministeredBy localhost:organisation ;
    omn:hasService <urn:publicid:IDN+localhost+authority+cm> ;
    omn:hasService <urn:publicid:IDN+localhost+authority+sa> .

<urn:publicid:IDN+localhost+authority+cm> rdf:type omngeni:
     AMService, owl:NamedIndividual ;
                                          rdfs:label "GENI␣AM␣v3␣API"
                                               ;
                                          omn:hasEndpoint <https://
                                              localhost:8443/sfa/api/
                                              am/v3> .

<urn:publicid:IDN+localhost+authority+sa> rdf:type omngeni:
     SAService, owl:NamedIndividual ;
                                          rdfs:label "ProtoGENI␣SA␣v1
                                              ␣API" ;
```

```
29                                           omn:hasEndpoint <https://
                                                 localhost:8443/sfa/api/
                                                 sa/v1> .
```

Example The related A-Box is provided in Listing 7.1, for the topology used within this section and shown in Figure 7.3. The A-Box describes a dummy federation (Lines 1 to 5) with a member organization (Lines 6 to 11) from a high-level point of view using the OMN, RDFS, WGS84 and Schema.org ontologies. The organization administers an infrastructure (Lines 12 to 19) that could offer its resources via a number of APIs, which might be needed for different federation contexts. In this example (Lines 20 to 29), SFA AM and SA APIs are offered, in order to allow compliant tools to establish a connection to each of these endpoints for resource discovery and AuthZ management. A SPARQL query to retrieve all relevant URLs is given in Listing 7.2 and the result is shown in Listing 7.3.

Listing 7.2: SPARQL query to get information about the federation

```
SELECT ?federation ?organization ?infrastructure ?AMEndpoint WHERE
    {
   ?fed rdf:type omnfed:Federation ;
       rdfs:label ?federation ;
     omnfed:hasFederationMember ?member .
   ?member rdf:type omnfed:FederationMember ;
          rdfs:label ?organization ;
          omnfed:administers ?infra .
   ?infra rdf:type omnfed:Infrastructure ;
                 rdfs:label ?infrastructure ;
                 omn:hasService ?AMService .
   ?AMService rdf:type omngeni:AMService ;
             omn:hasEndpoint ?AMEndpoint .
}
```

Listing 7.3: SPARQL federation query result

```
$ sparql --result JSON --data localhost-federation.ttl --query
    query-getAMEndpoints.sparql
{
  "head": {
    "vars": [ "federation" , "organization" , "infrastructure" , "
        AMEndpoint" ]
  } ,
  "results": {
    "bindings": [
      {
        "federation": { "type": "literal" , "value": "Localhost␣
            Federation" } ,
        "organization": { "type": "literal" , "value": "Localhost␣
            Organisation" } ,
        "infrastructure": { "type": "literal" , "value": "Localhost
            ␣Testbed" } ,
        "AMEndpoint": { "type": "uri" , "value": "https://localhost
            :8443/sfa/api/am/v3" }
```

```
      }
    ]
  }
}
```

7.2.1.2 Discovery

Bootstrapping

Now that the endpoint is known, a client has to communicate with, in this case, `https://localhost:8443/sfa/api/am/v3`, and an according implementation has to listen on this port. As shown in Listing 7.4, a single command can be issued on a Linux-/Unix-compliant machine to bootstrap the environment and setup an SFA-compliant testbed using FITeagle in under four minutes. First, it tests the environment for required packages and then downloads, configures, compiles, installs, starts and tests all needed software components. After the installation is completed, a J2EE environment should be running on the designated machine and be ready to accept SFA method calls.

Listing 7.4: FITeagle bootstrapping

```
$ time (bash <(curl -fsSkL fiteagle.org/sfa))
Checking environment...
 * Checking for 'java'...OK
 * Checking for 'javac'...OK
 * Checking for 'mvn'...OK
 * Checking for 'git'...OK
 * Checking for 'curl'...OK
 * Checking for 'unzip'...OK
 * Checking for 'screen'...OK
 * Checking for 'svn'...OK
Getting FITeagle bootstrap sources...OK
Downloading container...
Installing container...
Configuring container...
...
real 2m46.679s
user 3m56.167s
sys 0m27.237s
```

Metadata

To provide metainformation about the infrastructure, a SPARQL endpoint should be available following the Semantic Web model. Within the GENI and FIRE context, however, the SFA AM API *GetVersion* method can be called to retrieve a JSON-serialized tree with basic API information and arbitrary extensions. As shown in Listing 7.1 (Lines 12 to 24), these extensions can be used to embed OMN-based infrastructure information into the data structure by applying one of the available JSON-based serializations. Figure 7.4 shows the result of such a call using the jFed Probe GUI.

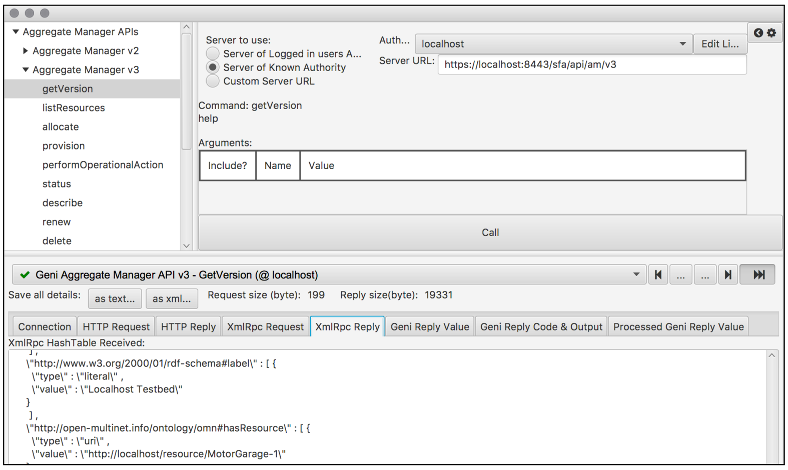


Figure 7.4: Integration into the SFA AM GetVersion method call

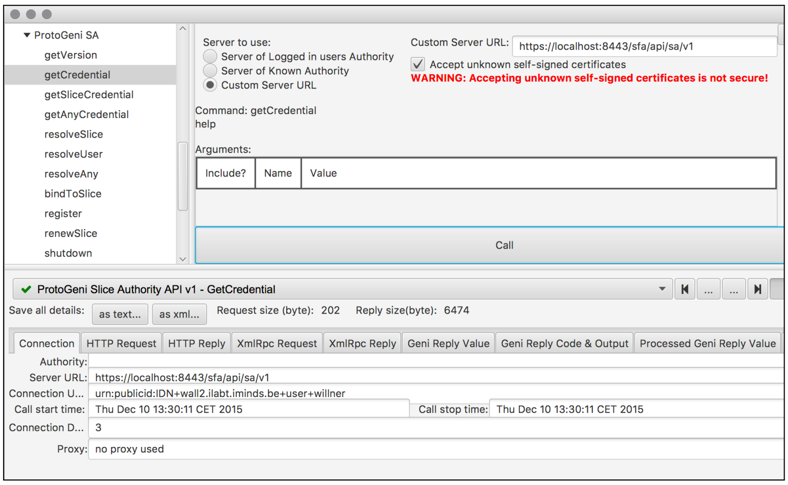


Figure 7.5: Execution of the SFA SA GetCredential method call

Resources

To discover resources in SFA, the AM method call *ListResources* is invoked and returns all resources on offer back to the requester. First, however, a user has to authenticate themselves and request credentials for this method, as shown in Figure 7.5. As shown in Figure 7.6, the aforementioned *MotorGarage* resource is returned. The result is serialized as a GENI RSpec v3 XML structure using *omnlib* and can therefore be used by any unmodified SFA AM client.

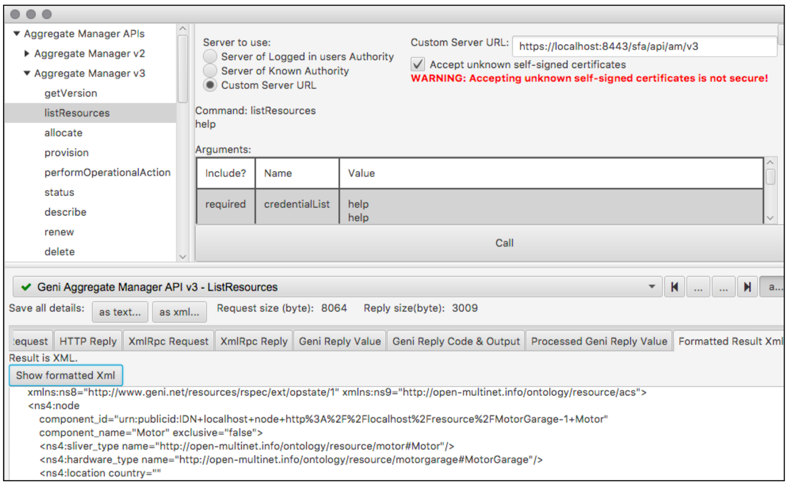


Figure 7.6: Integration into the SFA AM ListResources method call

Extension

Depending on the size of the managed infrastructure, the list of resources on offer can be comparatively large. While the option exists to return only resources that are currently available, this is the only filtering option. As the method call allows arbitrary options to be added, the *geni_query* field was introduced to the AM implementation of FITeagle to send a SPARQL query. This allows complex filtering mechanisms to be executed on the server side before a result is sent back to the user. Additionally, as the *ListResources* method call allows arbitrary RSpec return types to be specified, an RDF/XML-serialized graph can be returned instead. Together with the RDF data embedded in the *GetVersion* call (cf. Figure 7.4) and, potentially, on the Web site or a CH API, detailed information can be collected about a whole GENI-compliant federation, including its resources, as a semantically annotated graph. In Figure 7.7 this integration is shown within the result conforming to the models described Section 6.3.1.

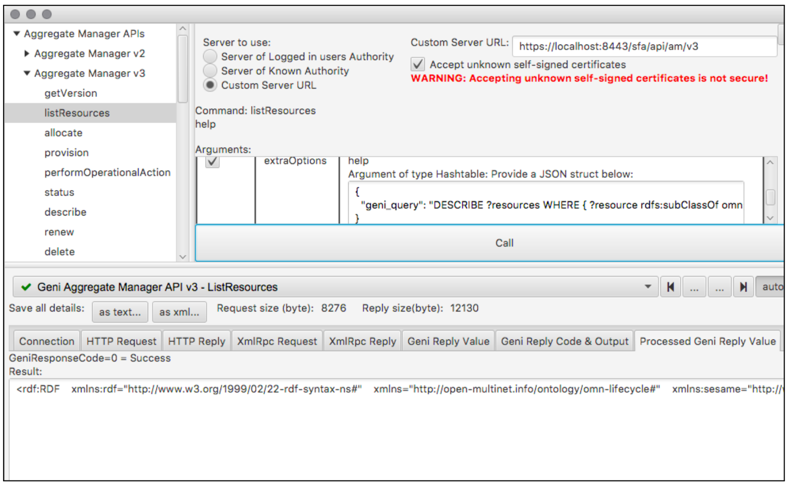
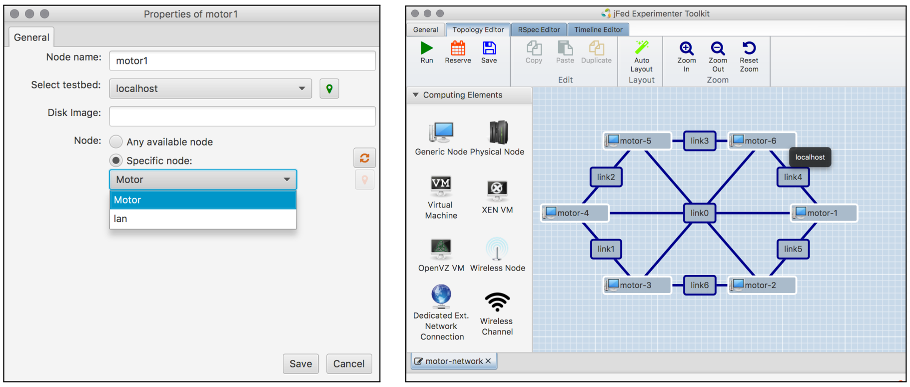


Figure 7.7: Extension of the SFA AM ListResources method call



(a) Selection of the resource

(b) Overview of the topology

Figure 7.8: Selection phase in the jFed experimenter GUI

7.2.1.3 Selection

Overview Based on the Advertisement shown in the previous section, the user can now construct their own topology with a subset of the available resources. For more complex slices that include heterogeneous resources, this involves the manual creation of GENI Request XML documents (cf. Section 6.3.2). Some GUI tools can be used, such as the jFed experimenter GUI, whose communication with FITeagle in the selection phase is shown in Figure 7.8.

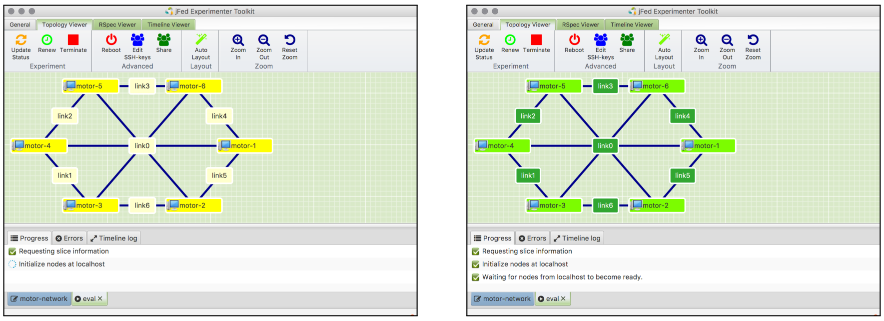
Topology The drop box in Figure 7.8a contains the list of available resources, in this case a motor and a dummy network resource that can virtually connect two motors. Based on these resources, the topology depicted in Figure 7.8b was created, composed of six virtual motors interconnected with seven virtual links.

7.2.1.4 Allocation and Provisioning

Overview As shown in Figure 7.9, the selected topology is now allocated (Figure 7.9a) and provisioned (Figure 7.9b) in the next step. This step dynamically extends the semantic graph by adding another resource of a specific type that is implemented by a specific *MotorGarage* and has a given state as visualized in Figure 7.3. While this topology can be deleted in a final step, the GENI SFA APIs only cover the life cycle phases up to deletion. In order to monitor and control the provisioned resources, handover to other protocols is needed and was implemented as shown in the next subsections.

7.2.1.5 Control

Overview To control provisioned resources, the FITeagle native REST API can be invoked, which takes RDF graphs as input. In Listing 7.5 a possible configuration file is presented, which refers to a specific motor resource (Line 6) within a provisioned slice (Lines 1 to 4), and sets the Revolutions per Minute (RPM) attribute to 111 (Lines 12 to 13). As shown in Listing 7.6, information about the resource can be queried (Lines 1 to 6) and modified (Line 8). The changes are presented in the subsequent query in Lines 10 to 15.



(a) Selection of the resource (b) Overview of the topology

Figure 7.9: Provisioning phase in the jFed experimenter GUI

Listing 7.5: Configuration graph for a resource in TTL

```
<http://localhost/topology/1449786502>
        a <http://open-multinet.info/ontology/omn#Topology> ;
        <http://open-multinet.info/ontology/omn#hasResource>
            <http://localhost/resource/MotorGarage-1/23bcb079-2f0d-4a39-81a8-06918fb690bb> .
<http://localhost/resource/MotorGarage-1/23bcb079-2f0d-4a39-81a8-06918fb690bb>
        a <http://open-multinet.info/ontology/resource/motor#Motor> ;
        <http://open-multinet.info/ontology/omn-lifecycle#implementedBy>
            <http://localhost/resource/MotorGarage-1> ;
        <http://open-multinet.info/ontology/resource/motor#rpm>
            "111"^^<http://www.w3.org/2001/XMLSchema#int> .
```

Listing 7.6: Configuration of a resource

```
$ curl -sH "Accept:␣text/turtle" http://localhost:8080/native/api/resources/MotorGarage-1/instances|grep -A3 23bcb079-2f0d-4a39-81a8-06918fb690bb
<http://localhost/resource/MotorGarage-1/c900aa36-30d8-48e1-bf73-356bb48d01cf>
        a <http://open-multinet.info/ontology/resource/motor#Motor> ;
        :implementedBy <http://localhost/resource/MotorGarage-1> ;
        <http://open-multinet.info/ontology/resource/motor#rpm>
                "0"^^<http://www.w3.org/2001/XMLSchema#int> .
$ curl --request POST --data @config.ttl http://localhost:8080/native/api/resources/
$ curl -sH "Accept:␣text/turtle" http://localhost:8080/native/api/resources/MotorGarage-1/instances|grep -A3 23bcb079-2f0d-4a39-81a8-06918fb690bb
<http://localhost/resource/MotorGarage-1/c900aa36-30d8-48e1-bf73-356bb48d01cf>
```

```
       a <http://open-multinet.info/ontology/resource/motor#Motor>
            ;
       :implementedBy <http://localhost/resource/MotorGarage-1> ;
       <http://open-multinet.info/ontology/resource/motor#rpm>
              "111"^^<http://www.w3.org/2001/XMLSchema#int> .
```

7.2.1.6 Monitoring

Overview Finally, the experimenter may want to get measurement information about the resource. As described in Section 6.4.5, *Service Integration Modules* can be used to export monitoring information about provisioned resources. For this purpose, *INFORM* messages were introduced (cf. Table 6.1), which allow an adapter to notify other modules in FITeagle about resource status changes. As the motors provisioned in the example topology change their RPMs every five seconds by a random number, corresponding *INFORM* messages are sent by the managing adapter. This procedure is shown in Listing 7.7, where a WebSocket connection to the message bus is established and the content of the messages are printed.

Listing 7.7: Logging of messages

```
$ ws-client ws://localhost:8080/bus/api/logger
# Connecting to ws://localhost:8080/bus/api/logger...
# Connected in session 71e986f2-5add-48e0-bdb5-2b225fe8b6dd
# text-message: {"body":"<http://localhost/resource/MotorGarage
    -1/23bcb079-2f0d-4a39-81a8-06918fb690bb>...<http://open-
    multinet.info/ontology/resource/motor#rpm>...\"545\"...",
              "METHOD_TARGET":"N.A.",
              "JMSCorrelationID":"aa292a41-bafc-4503-8d4d-6145
                  e362598f",
              "serialization":"TURTLE",
              "JMSXDeliveryCount":"1",
              "METHOD_TYPE":"INFORM"}
# text-message: {"body":"<http://localhost/resource/MotorGarage
    -1/23bcb079-2f0d-4a39-81a8-06918fb690bb>...<http://open-
    multinet.info/ontology/resource/motor#rpm>...\"2054\"...",
              "METHOD_TARGET":"N.A.",
              "JMSCorrelationID":"3eb8256f-a20a-4e2e-85af-159
                  d47c3c461",
              "serialization":"TURTLE",
              "JMSXDeliveryCount":"1",
              "METHOD_TYPE":"INFORM"}
session 71e9...b6dd>
```

7.2.1.7 Delete

Overview The last step is the release of resources after the experiment has ended. Figure 7.10 shows the topology that was created, provisioned, controlled and monitored within the preceding sections. With the invocation of the GENI AM *Delete* method, each adapter releases its associated resources and related information is removed from the database.

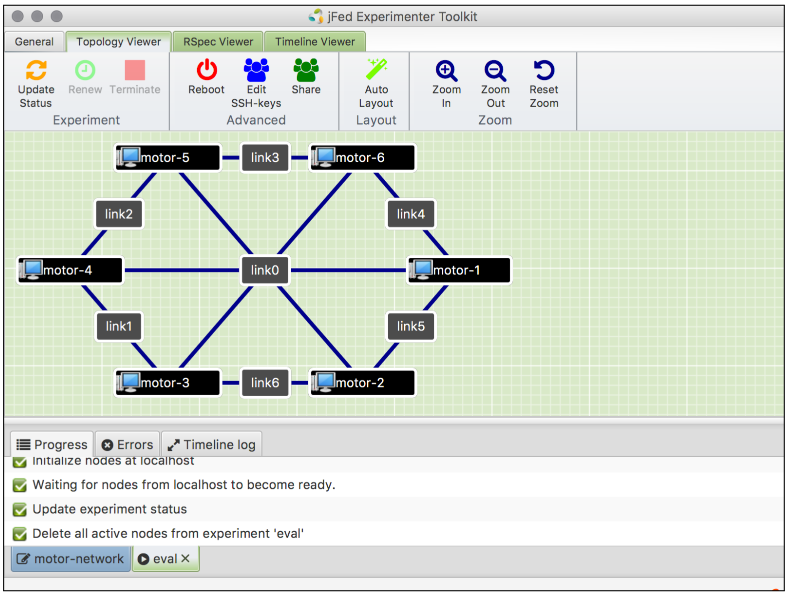


Figure 7.10: Ended experiment in the jFed experimenter GUI

7.2.2 Conformity

Overview In the preceding section, a complete experiment life cycle workflow was executed step by step. This shows that the fundamental capabilities have been implemented and are compatible with the jFed Probe and Experimenter GUIs. As stated in Requirements U11, P11 and O11, the implementation needs to be compliant with version 3 of the SFA AM specification. In this section, in order to show that the SFA AM and SA modules in FITeagle fully comply with the specification, a standard conformity tests is executed.

Result Such a compliance test suite is integrated in the jFed Automated Tester. A script to automatically execute the test suite after each change in the code for any FITeagle module is included in the code repository. The jFed Automated Tester also tests the conformity by executing the jFed low level test TestAggregateManager3. As shown in Listing 7.8, the integration test is executed from the command line (Line 1) and downloads all required binaries and configuration files (Lines 2 and 6) and finishes after about a minute (Line 82). The test is configured to use both APIs (Lines 13 to 14) and executes 20 consecutive tests against the installation on localhost (Lines 17 to 36) involving a single *Motor* resource. Additionally, a network of *Motors* is tested (Lines 38 to 46), as shown in Figure 7.8b. Further results are given in Appendix B.

Semantics Besides these default conformity tests, RDF/XML-serialized inputs are tested (Lines 48 to 80). Instead of standard GENI RSpec v3 Request XML documents, OMN-based RDF graphs are used to exchange information. Warnings appear in Lines 72, 76 and 78 since the test suite expects specific RSpec XML tags that are now represented by analogous OMN concepts.

Listing 7.8: jFed test results

```
$ time ./integration-test/runJfed_local.sh
Downloading latest library...
...
TEST: TestAggregateManager3
...
Read context properties from file "conf/cli.properties":
   Tested Authority:localhost
      URN (connect):urn:publicid:IDN+localhost+authority+cm
      URN (rspec):urn:publicid:IDN+localhost+authority+cm
      Hrn:localhost
      Server certificates:[...]
      Allowed server certificate hostname aliases:[localhost]
      URL for ServerType{"PROTOGENI_SA" "1"}: https://localhost:8443/sfa/api/sa/v1
      URL for ServerType{"AM" "3"}: https://localhost:8443/sfa/api/am/v3
   User:urn:publicid:IDN+localhost+user+testing
      Authority URN:urn:publicid:IDN+localhost+authority+cm
Running testGetVersionXmlRpcCorrectness...SUCCESS
Running testGetVersionResultCorrectness...SUCCESS
Running testGetVersionResultApiVersionsCorrectness...SUCCESS
Running testGetVersionResultNoDuplicates...SUCCESS
Running testListAvailableResources...SUCCESS
Running testStatusBadSlice...SUCCESS
Running testListResourcesBadCredential...SUCCESS
Running createTestSlices...SUCCESS
Running testStatusNoSliverSlice...SUCCESS
Running testDescribeNoSliverSlice...SUCCESS
Running testAllocate...SUCCESS
Running testProvision...SUCCESS
Running testSliverBecomesProvisioned...SUCCESS
Running testPerformOperationalAction...SUCCESS
Running testStatusExistingSliver...SUCCESS
Running testDescribeProvisionedSliver...SUCCESS
Running testSliverBecomesStarted...SUCCESS
Running testDescribeReadySliver...SUCCESS
Running testRenewSliver...SUCCESS
Running testDeleteSliver...SUCCESS
...
TEST: NetworkedMotorTopology
...
slice urn:publicid:IDN+localhost+slice+1446300518 does not yet exist
Contacting urn:publicid:IDN+localhost+authority+cm...
Sliver at urn:publicid:IDN+localhost+authority+cm is created and initializing...
Will now wait until the sliver is ready...
Contacting urn:publicid:IDN+localhost+authority+cm to check status ...
Status of sliver at urn:publicid:IDN+localhost+authority+cm is READY
The sliver is ready.
...
```

```
TEST: RDF
...
Read context properties from file "conf/cli.rdfxml.properties":
   Tested Authority:localhost
      URN (connect):urn:publicid:IDN+localhost+authority+cm
      URN (rspec):urn:publicid:IDN+localhost+authority+cm
      Hrn:localhost
      Server certificates:[...]
      Allowed server certificate hostname aliases:[localhost]
      URL for ServerType{"PROTOGENI_SA" "1"}: https://localhost
          :8443/sfa/api/sa/v1
      URL for ServerType{"AM" "3"}: https://localhost:8443/sfa/api/
          am/v3
   User:urn:publicid:IDN+localhost+user+testing
      Authority URN:urn:publicid:IDN+localhost+authority+cm
Running testGetVersionXmlRpcCorrectness...SUCCESS
Running testGetVersionResultCorrectness...SUCCESS
Running testGetVersionResultApiVersionsCorrectness...SUCCESS
Running testGetVersionResultNoDuplicates...SUCCESS
Running testListAvailableResources...SUCCESS
Running testStatusBadSlice...SUCCESS
Running testListResourcesBadCredential...SUCCESS
Running createTestSlices...SUCCESS
Running testStatusNoSliverSlice...SUCCESS
Running testDescribeNoSliverSlice...SUCCESS
Running testAllocate...SUCCESS
Running testProvision...WARN
Running testSliverBecomesProvisioned...SUCCESS
Running testPerformOperationalAction...SUCCESS
Running testStatusExistingSliver...SUCCESS
Running testDescribeProvisionedSliver...WARN
Running testSliverBecomesStarted...SUCCESS
Running testDescribeReadySliver...WARN
Running testRenewSliver...SUCCESS
Running testDeleteSliver...SUCCESS
...
real 1m7.958s
user 0m35.573s
sys 0m1.895s
```

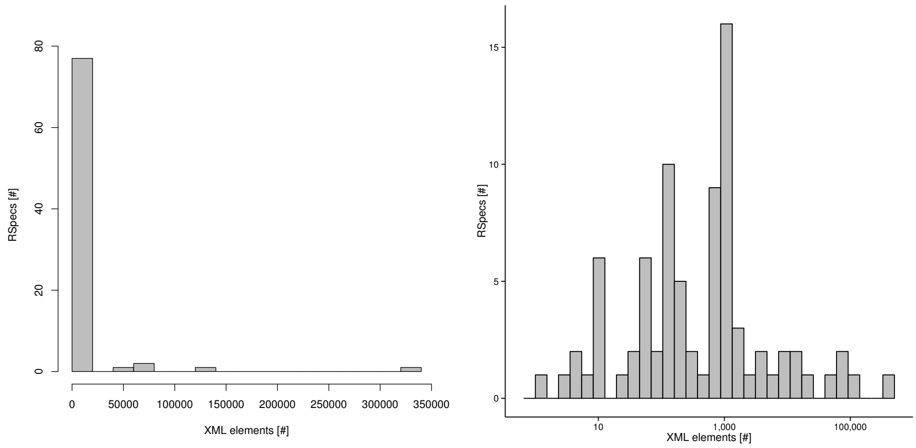
7.3 Performance Evaluation

Overview

Besides demonstrating that the FI resource experimentation life cycle can be modeled using OMN and executed using FITeagle, the applicability of the approach is validated by assessing performance to ensure practicability within existing experiment environments. If not stated otherwise, (i) measurements were executed on the software and hardware environment described in Section 6.2 and (ii) measurements were repeated 100 times with 1 second breaks in between and 10 repetitions were executed before filtering out possible start up, initialization and compilation outliers. Average values are expressed based on a 95% Confidence Interval (CI).

7.3.1 Translator

Size In order to show the applicability of the work, this section illustrates that the time required to translate resource information using the translator is in a practicable span for the given context. The results of the *ListResources* method calls of the 99 SFA AMs that are monitored[1] within the Fed4FIRE project were analyzed. This list contains 82 valid XML-based GENI RSpec replies with 762,634 XML elements in total, of which only 3,043 are Nodes, 31,155 are Links and 25,493 Interfaces. As shown in Figure 7.11, most testbeds expose less than 20,000 XML elements.



(a) linear (b) logarithmic

Figure 7.11: Size distribution of RSpec Advertisements [a11]

Costs To compare the translation time for an RSpec Advertisement with the duration of the underlying function call in the FI experimentation context, the query and translation time was measured for a single testbed. Based on the analysis above, the CloudLab Wisconsin testbed[2] was used, which exposes 19,371 XML elements. The results in Figure 7.12 show that the average translation time of 583 ms ± 9 ms would add about 10% to the average response time of 5,453 ms ± 131 ms. This effect, however, could be mitigated by translating in advance, distributing the work load or optimizing the translation code.

Optimizing The time required for translating models was further investigated. The input is based on the RSpec Advertisement published by the Virtual Wall testbed, whose XML serialization is about 2.4 MB in size and contains 87,638 XML tags. In total 212 nodes, including their 619 sliver and 1,297 hardware types, were translated. To highlight the most expensive operation, the evaluation is divided into two parts. The first part includes the conversion of the XML Advertisement document into a JAXB OM. This conversion takes, with a 95% confidence interval, 5,263 ms ± 15 ms, as shown in Figure 7.14. The second part includes the conversion process from the JAXB OM to the RDF graph and the serialization of the graph to XML again. As shown in Figure 7.13, although the most expensive operation is the XML serialization, the bottleneck is by and large caused by the JAXB OM creation.

[1] https://flsmonitor.fed4fire.eu/api/index.php/result
[2] https://www.cloudlab.us

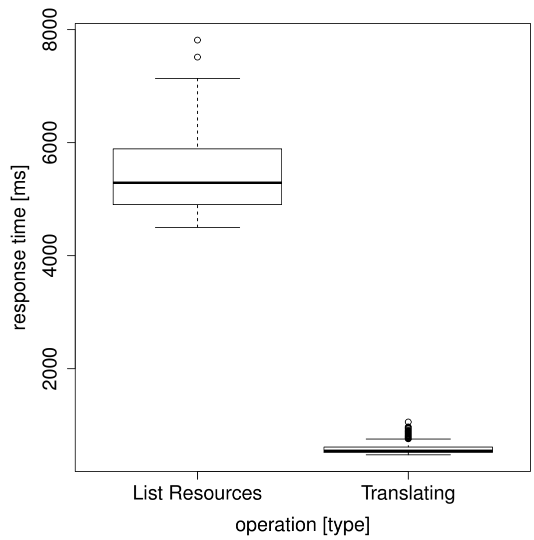


Figure 7.12: Performance comparison of listing and translating resource information

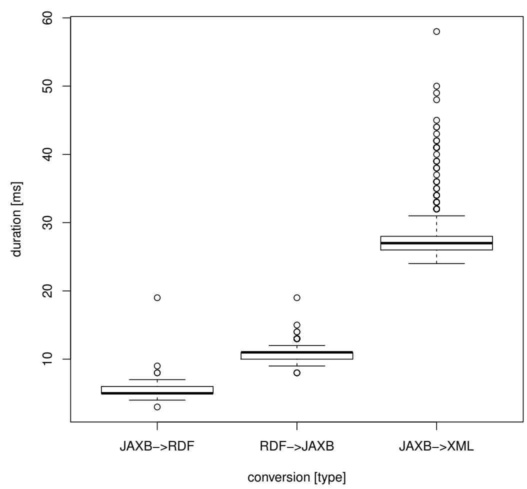


Figure 7.13: Performance of the object model translation process [a24]

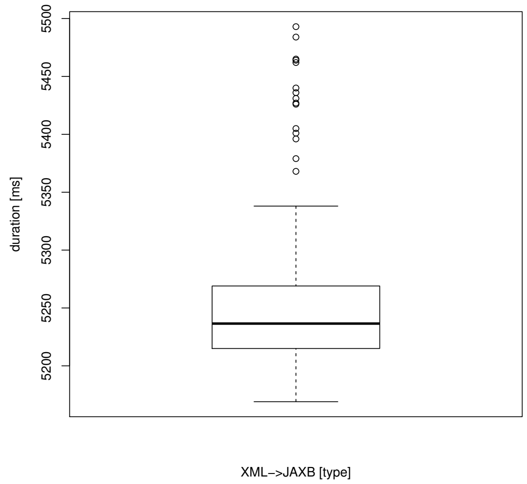


Figure 7.14: Performance of the object model creation process [a24]

Resource Matching

Assuming a testbed accepts the potentially higher response time in favor of the added value of merging its information into a global linked data set, its resources can be found by applying the aforementioned resource matching queries. The translation of all available tree data structures into an RDF-based graph, using the OMN vocabulary and rules, resulted in a set of 3,345,439 statements. This knowledge graph has, by adding further rules, infrastructures and other data sources, the potential to grow by a multiple thereof. This knowledge graph can now be used to retrieve information about the federation. As an example, a user might require a topology, which consists of two wireless Linux nodes with specific hardware requirements, located within a distance of 50 Km from the Acropolis in Athens, Greece. Such a query is shown in Listing 7.9 and the corresponding result in Listing 7.10.

Listing 7.9: Resource matching query example [a11]

```
SELECT ?node1 ?node2 WHERE { ?node1 rdf:type omnres:Node.
?node2 rdf:type omnres:Node.
?node1 omn:hasComponent ?memComp1. ?node2 omn:hasComponent ?
    memComp2.
?memComp1 rdf:type omncomp:MemoryComponent.
?memComp2 rdf:type omncomp:MemoryComponent.
?memComp1 dbp:memory ?mvalue1.FILTER (?mvalue1 = "256"^^xsd:
    integer)
?memComp2 dbp:memory ?mvalue. FILTER (?mvalue = "256"^^xsd:
    integer)
?node1 omnres:hasSliverType/omndpc:hasDiskImage/omndpc:
    hasDiskimageOS ?os1.
FILTER (xsd:string(?os1) = "VoyageLinux"^^xsd:string ||xsd:
    string(?os1) = "Fedora"^^xsd:string || xsd:string(?os1) = "
    FreeBSD"^^xsd:string || xsd:string(?os1) = "Linux"^^xsd:
    string)
?node2 omnres:hasSliverType/omndpc:hasDiskImage/omndpc:
    hasDiskimageOS ?os2.
FILTER (xsd:string(?os2) = "VoyageLinux"^^xsd:string ||xsd:
    string(?os2) = "Fedora"^^xsd:string || xsd:string(?os2) = "
    FreeBSD"^^xsd:string || xsd:string(?os2) = "Linux"^^xsd:
    string)
?node1 geo:lat ?lat1. ?node1 geo:long ?lon1.
?node2 geo:lat ?lat2. ?node2 geo:long ?lon2.
FILTER( (37.971472-xsd:float(?lat1))*( 37.971472-xsd:float(?
    lat1))+( 23.726633-xsd:float(?lon1))*( 23.726633-xsd:float
    (?lon1))*( 0.942964-(0.0084674*xsd:float(?lat1))) <
    0.00808779738472242*250/100).FILTER( (37.971472-xsd:float(?
    lat2))*( 37.971472-xsd:float(?lat2))+( 23.726633-xsd:float
    (?lon2))*( 23.726633-xsd:float(?lon2))*(
    0.942964-(0.0084674*xsd:float(?lat2))) <
    0.00808779738472242*250/100).
FILTER (?lat1=?lat2)
FILTER (?lon1=?lon2)
FILTER (?node1 != ?node2) } LIMIT 1
```

Listing 7.10: Resource matching query result [a11]

```
:node1=> <urn:publicid:IDN+omf:netmode+node+node11>,
:node2=> <urn:publicid:IDN+omf:netmode+node+node14>
TIME EXECUTION: 0.1687551689147949
```

Performance Comparison

To assess the performance impact of the complexity of the query, it has been compared with another simpler one, which is shown in Listing 7.11 together with its result in Listing 7.12. In Figure 7.15, the duration for executing both the queries are shown. While finding the three largest aggregates took on average 129 ms ± 3 ms, the matching query took on average 168 ms ± 1 ms and about 30% longer, however, much less than a single *ListResources* call in a single testbed.

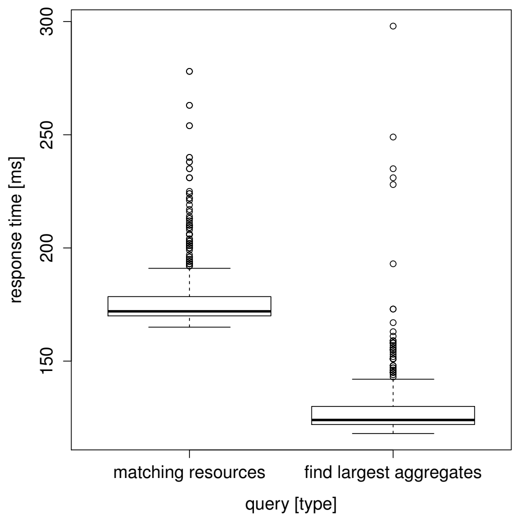


Figure 7.15: Performance comparison of queries [a11]

Listing 7.11: Find largest aggregate via query [a11]

```
SELECT (COUNT(?am) as ?fre) ?am WHERE {
  ?node omn-lifecycle:managedBy ?am .
} GROUP BY (?am) ORDER BY DESC (?fre) LIMIT 3
```

Listing 7.12: Largest aggregates [a11]

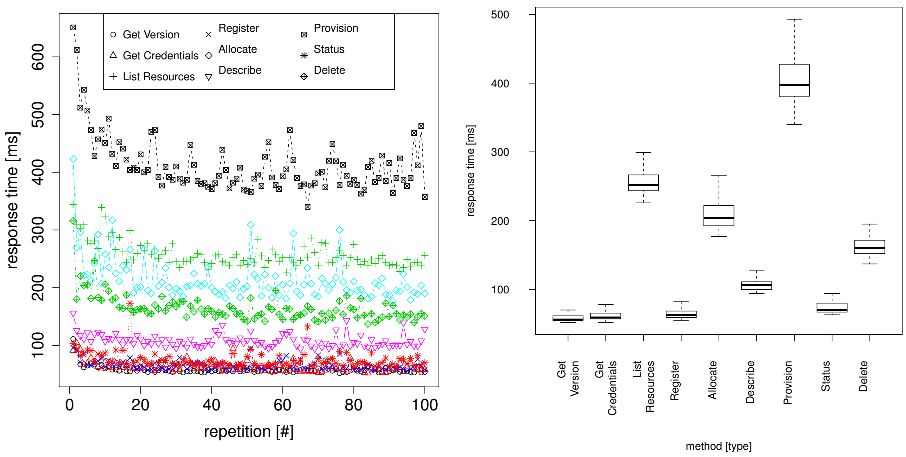
```
?fre ?am
719 <urn:publicid:IDN+emulab.net+authority+cm>
326 <urn:publicid:IDN+utah.cloudlab.us+authority+cm>
255 <urn:publicid:IDN+ple+authority+cm>
```

7.3.2 FITeagle

Response Times

The translation mechanism takes only a part of the time needed to implement the whole life cycle. Based on the workflow and conformance tests described in Section 7.2.1, the overall performance of the complete FITeagle framework is evaluated within this section. For this evaluation eight different AM and SA methods were invoked to allocate, provision and delete a single motor instance. In Figure 7.16a, the relevant end-to-end response times of the SFA AM method calls *GetVersion*, *ListResources*, *Allocate*, *Provision*, *Status* and *Delete*, as well as SFA SA method calls *GetCredentials* and *Register* using the jFed library, are visualized and compared to each other. While Figure 7.16a shows that the performance of each operation does not degrade over time, the following response times as a function of the request type are compared in more detail in Figure 7.16b:

- *GetVersion*: 59 ms ± 2 ms (95% CI)
- *GetCredential*: 62 ms ± 2 ms (95% CI)
- *ListResources*: 258 ms ± 4 ms (95% CI)
- *Register*: 65 ms ± 2 ms (95% CI)
- *Allocate*: 214 ms ± 7 ms (95% CI)
- *Describe*: 108 ms ± 2 ms (95% CI)
- *Provision*: 412 ms ± 10 ms (95% CI)
- *Status*: 75 ms ± 3 ms (95% CI)
- *Delete*: 165 ms ± 5 ms (95% CI)



(a) As a function of the request type over time (b) As a function of the request type

Figure 7.16: FITeagle response times

Serialization

A short analysis of the delay introduced by the communication and serialization overhead within FITeagle, is given in Figure 7.17. Due to their design, modules in FITeagle can communicate with each other using different communication protocols and serializations. Where messages are serialized using TTL, differences between the use of direct API calls, Enterprise Java Bean (EJB) mechanisms and Message Driven Beans (MDBs) are negligible. However, increasing the size and complexity of the messages, here due to the use of the RDF/XML serialization, increases the response time by a factor of three and, compared to direct API calls, an overhead of about 50% is introduced by applying message-driven communication patterns. As a result, TTL was chosen as the internal serialization and, by applying even more efficient formats such as HDT, the communication overhead could be reduced even further, making the choice of communication protocol insignificant.

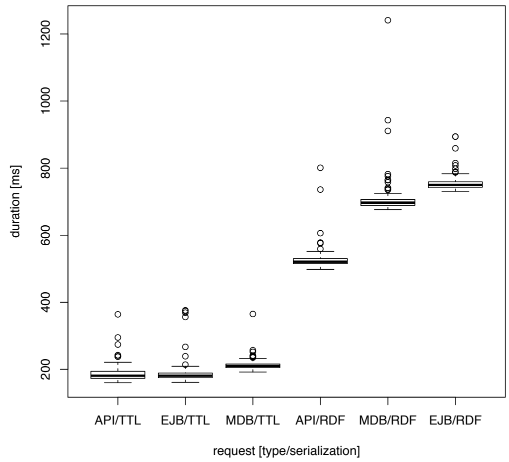
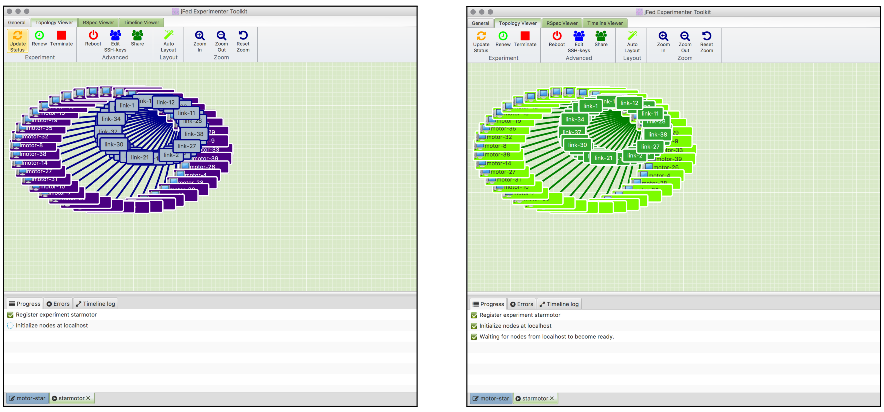


Figure 7.17: Influence of the communication and serialization types [a21]

Scalability

Analyzing the response times of different aspects of the FITeagle framework, including translation, serialization, and communication, provides a valuable insight into its overall performance and stability. Based on this analysis, the scalability of a running instance was examined to asses the scale the system can operate within. To achieve this, the response time for each method call was measured while increasing the number of nodes requested in a topology. An example of such as star topology of interconnected motors is given in Figure 7.18.



(a) Constructed topology (b) Provisioned topology

Figure 7.18: Provisioning of a topology using the jFed experimenter GUI

Figure 7.19

First, a new topology is created *(Register)*, then the requested numbers of nodes are reserved *(Allocate)* before being instantiated *(Provision)*. Finally, the status of each node is queried *(Status)* and the topology deleted *(Delete)*. In order to reduce side effects and outliers, the workflow was executed twice before measurements started and a 1 second pause was introduced between each function call. The result is depicted in Figure 7.19 and shows a constant time for registering, provisioning, querying and deleting a slice. The time for allocation first grows linearly but with more that 200 requested resources, the behavior of the response time changes to start growing exponentially and the response times of other method calls increase as well. As no specific limits are encoded within the FITeagle implementation, this observation leads to the conclusion that the system scales linearly up to a point where the resource limits of the hosting machine, such as Random Access Memory (RAM), are exceeded.

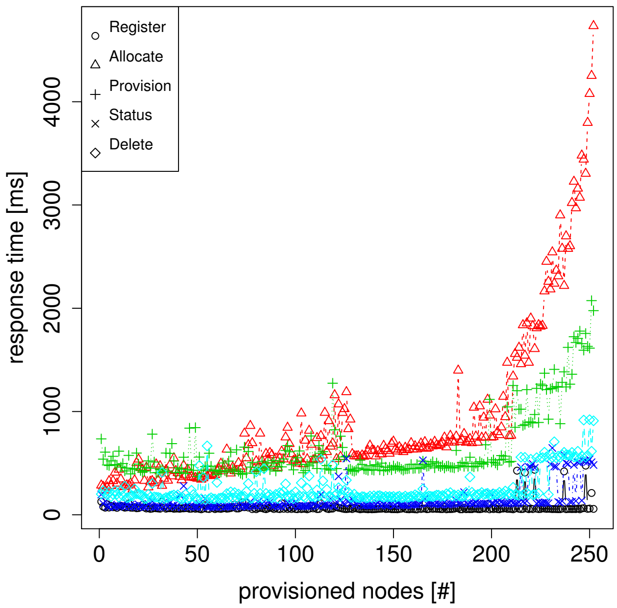


Figure 7.19: Response time as a function of nodes per request

7.4 Observational Validation

Overview

In order to evaluate the applicability of the proposed architecture and resource description, the design must be shown to comply with requirements established in Chapter 3. While there are several options for analyzing this conformance, observing real environments is the most suitable one. Based on the utilization of the developed software and models, this section shows their applicability in the context of European research projects. Besides the two exemplary projects described below, parts of the research conducted and used in the projects OpenLab, FIRE LTE Testbeds for Open Experimentation (FLEX), INfrastructures for the Future INternet CommunITY (INFINITY), and Universities for Future Internet (UNIFI).

7.4.1 FP7 FIRE Fed4FIRE Project

Overview

The Fed4FIRE project was described in Section 2.3.2 and the work conducted within this thesis has been integrated into the project in two ways. First, the OMN ontology used to model the Fed4FIRE federation with its members and infrastructures and was together with an automated translation of the available Advertisement RSpecs, a public SPARQL endpoint was established. Second, the linking of information related to provisioning and monitoring of resources was demonstrated based on the SFA AM implementation of FITeagle.

7.4.1.1 DBcloud

Overview As described above, information about GENI- and FIRE-compatible federations is encoded in different formats. In particular, information about resources on offer published as XML trees with a number of arbitrary extensions at distributed, secured SFA AM endpoints. Adopting formal information models and semantically annotated graphs would allow users to link, relate, enhance, query and reason over the heterogeneous data sets, which would not be possible otherwise. Therefore, a centralized knowledge base was implemented using the OMN ontology.

DBcloud Following the approach of DBpedia, the DBcloud[3] extracts information from testbeds involved in GENI, Fed4FIRE and related federations and makes this information accessible on the Web. The resulting knowledge base, as described and used in Section 7.3.1, currently contains information about more than 100 aggregates, 2,500 nodes, 6,700 links, and about 11,000 interfaces. This results in 3.3 million statements and has the potential to grow by many times this amount.

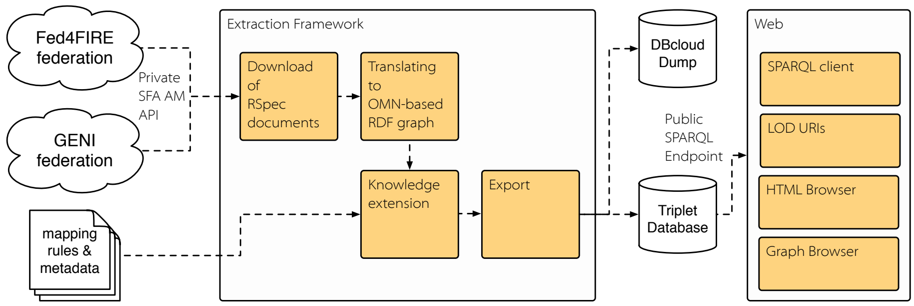


Figure 7.20: Overview of the DBcloud extraction framework (analogous to [p147])

Extraction In Figure 7.20, an overview of the DBcloud extraction framework is given. Its design follows the DBpedia extraction framework [p147] and is available at http://lod.fed4fire.eu (cf. Figure 7.21). To gather information about published resources, the SFA AM APIs of the known testbeds are called, using X.509 certificates that are trusted by each infrastructure involved. The downloaded XML documents are then translated into an RDF graph using the OMN ontology. To extend the knowledge encoded in this graph, the Apache Jena inference engine is used by applying infrastructure-specific rules. Finally, after adding more static information about the federations, the resulting knowledge graph is written in a Sesame triplet database and a TTL-serialized file. Information stored in the database is then available via a SPARQL endpoint. Most URIs are dereferenceable, as suggested in the LOD principles[4], and both an HTML rendering, using LodView[5] [], and a graph browser, using LodLive, are available.

[3] http://lod.fed4fire.eu
[4] http://www.w3.org/DesignIssues/LinkedData.html
[5] http://lodview.it

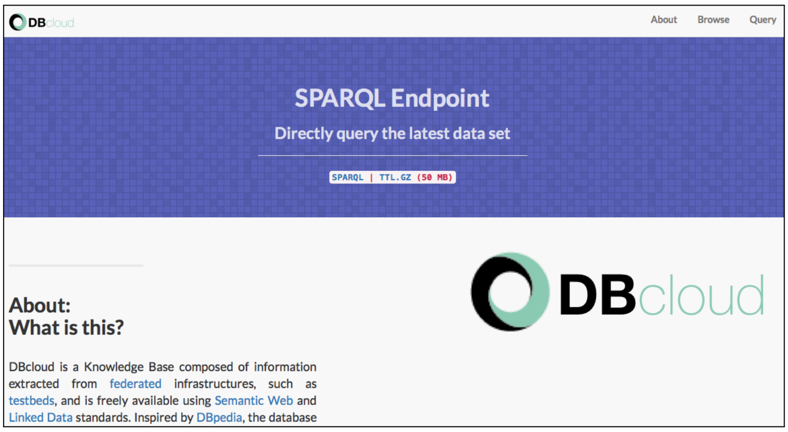


Figure 7.21: DBcloud Web site

Example

As mentioned in Section 7.2.1.1, the static information about a federation is usually encoded on Web sites. In Listing 7.13 an excerpt of the extracted RDF information from an annotated version of the Web site is shown. The federation consists of multiple partners and, as shown in the HTML renderings in Figures 7.22 and 7.23, the member `http://lod.fed4fire.eu/fokus.fraunhofer.de` exposes basic information about the Fraunhofer FOKUS using well-known vocabularies. By applying the *rdfs:isDefinedBy* property, Fraunhofer FOKUS is further linked to the corresponding DBpedia entry which provides more detailed information.

Listing 7.13: RDFa embedded information about the federation

```
$ rapper -i rdfa testbeds.html > testbeds.n3 && sparql --query
    testbeds.sparql --data testbeds.n3
rapper: Parsing URI file:///testbeds.html with parser rdfa
rapper: Serializing with serializer ntriples
rapper: Parsing returned 50 triples
================================================
| testbed |
================================================
| <http://lod.fed4fire.eu/id/atos.net> |
| <http://lod.fed4fire.eu/id/av.tu-berlin.de |
| <http://lod.fed4fire.eu/id/bris.ac.uk> |
| <http://lod.fed4fire.eu/id/bt.com> |
| <http://lod.fed4fire.eu/id/dante.net> |
| <http://lod.fed4fire.eu/id/deimos-space.com> |
| <http://lod.fed4fire.eu/id/ed.ac.uk> |
| <http://lod.fed4fire.eu/id/eng.nia.or.kr> |
| <http://lod.fed4fire.eu/id/eurescom.eu> |
| <http://lod.fed4fire.eu/id/i2cat.net> |
...
================================================
```

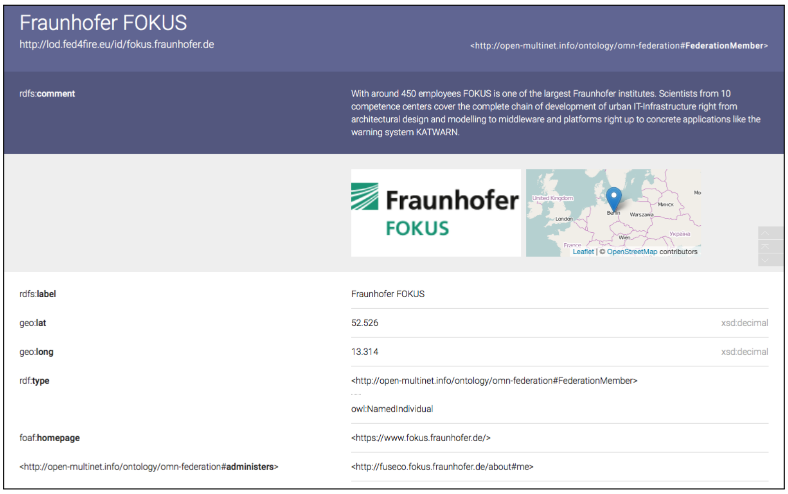


Figure 7.22: Visualization of the DBcloud Fraunhofer FOKUS entry within LodView

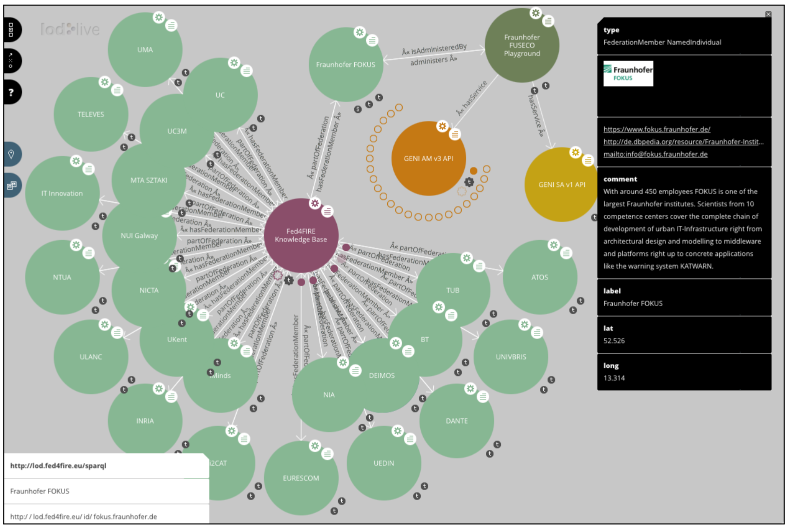


Figure 7.23: Visualization of the DBcloud Fraunhofer FOKUS entry within LodLive

7.4.1.2 Integration into the Architecture

Overview The goal of Fed4FIRE as a large integration project is to federate a potentially high number of heterogeneous testbeds within Europe. Therefore, its overall architecture includes the exchange of more types of information than just the ones described above. These include information about services offered, monitoring data, trust relationships, availability and reservations. Potential points of interaction between DBcloud and the Fed4FIRE architecture are shown in Figures 7.25 to 7.27. The resource *(R)* in the center of Figure 7.24 can be provisioned *(prov)* using SFA with XML-based RSpecs, monitored *(mon)* using OMSP with CSV-based lists, and controlled *(ctrl)* using FRCP via an XMPP or AMQP interface with XML- or JSON-based models. Finally, these interfaces should be interconnected using a PDP to allow secure handovers between these protocols.

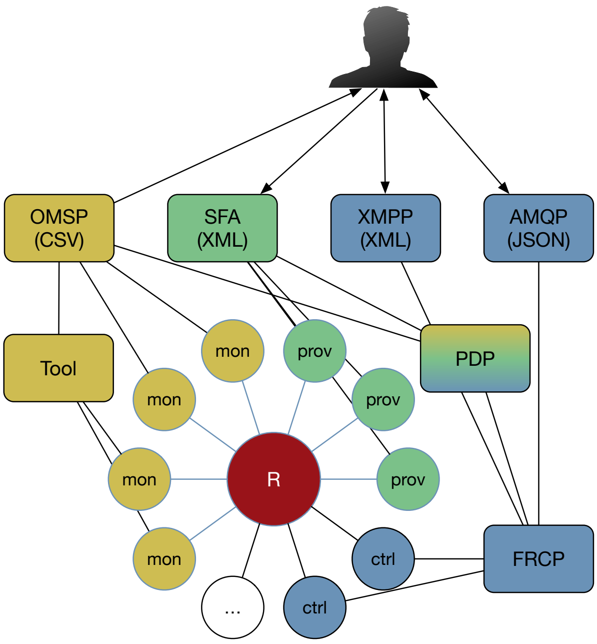


Figure 7.24: Resource- and information-centric view of the Fed4FIRE architecture

Provisioning

Figure 7.25 depicts the potential integration of information related to the discovery, provisioning, and reservation of resources based mainly on the SFA AM API. The methods for importing information from *Testbed A* and the *Testbed Directory* was detailed in the previous subsection. The imported information could further be used by a *Future Reservation Broker* to use SPARQL queries to identify available time slots that match specific user requirements. Additionally, testbeds that natively export RDF-based information, e.g. like those using in FITeagle, can further expose information that is not supported by the translation mechanism.

Monitoring

In Figure 7.26, the potential integration of dynamic monitoring data is depicted. Currently, experiment measurements, facility status and infrastructure monitoring information is exported using OMSP to one or multiple OML server instances. In particular, information about resources within an infrastructure and the facility itself is used for FLS and could be integrated into the DBcloud. The same holds true for monitoring information, which could be exported within the SFA AM API.

Control

Finally, Figure 7.27 shows that in Fed4FIRE the use of FRCP for resource control is envisioned. As data related to the control of resources might not have been exported through the SFA AM API, this information could potentially be imported directly by the appropriate FRCP calls.

Demonstration

To show how to link and combine information in this context using OMN and FITeagle, an example workflow is depicted in Figure 7.28. First, an experimenter uses an RSpec with monitoring extensions and URI-based sliver information that can be used by FITeagle and *omnlib* to provision a VM within OpenStack that is to be monitored. Next, the measured information is sent as RDF/OMSP-serialized data to a Semantic OML (SOML) server. In this way, two separate graphs about resource allocation and

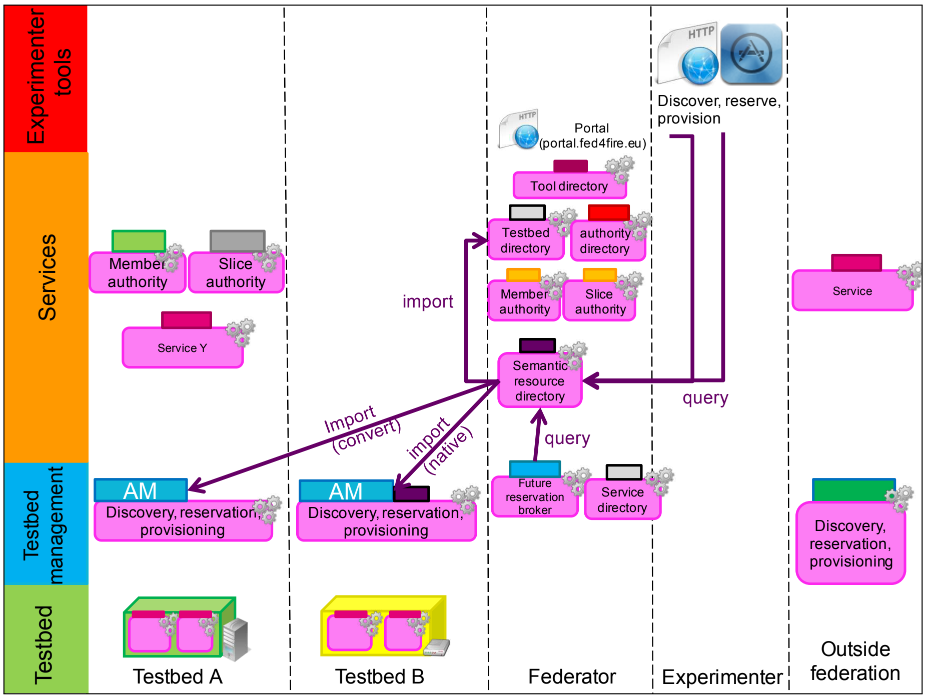


Figure 7.25: Resource provisioning information within Fed4FIRE (based on [t88])

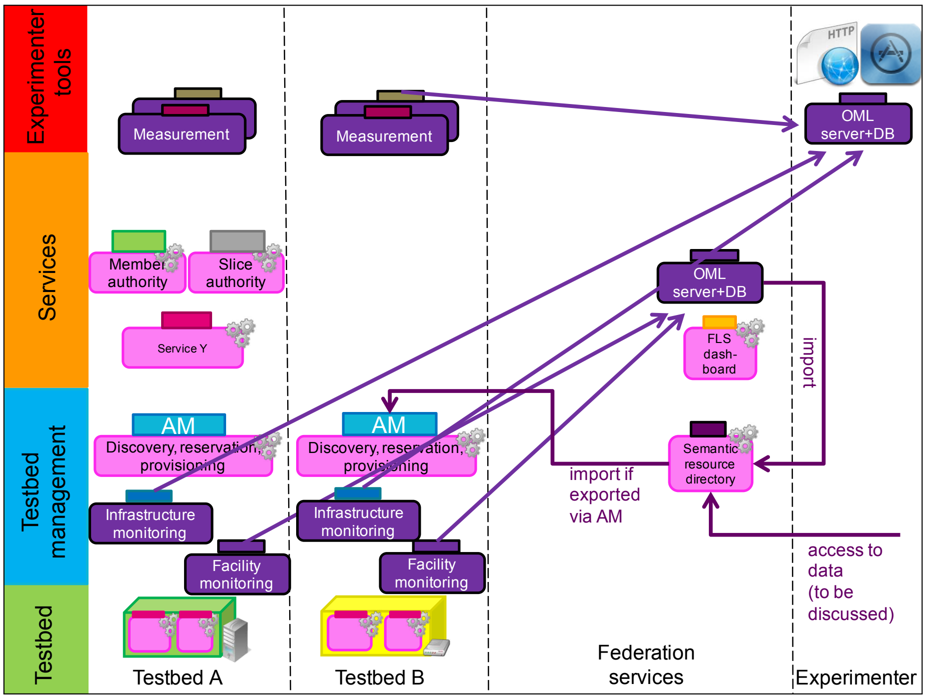


Figure 7.26: Resource monitoring information within Fed4FIRE (based on [t88])

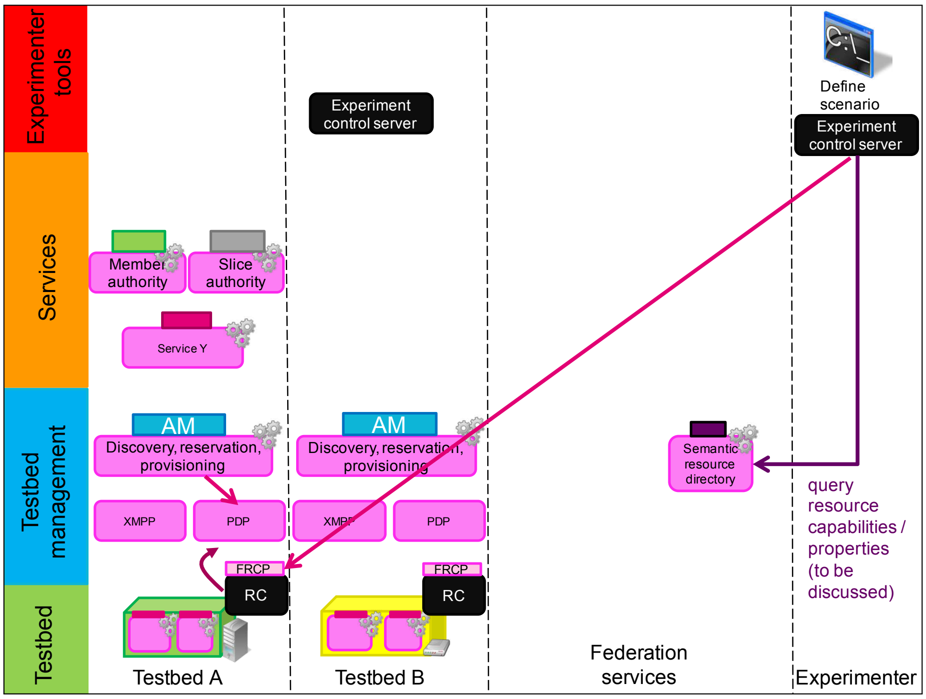


Figure 7.27: Resource control information within Fed4FIRE (based on [t88])

status information are stored in two triplet stores using the same identifier for the VM. In the third step both sinks are queried using SPARQL and the two graphs are joined into a single connected graph. The result is visualized using LodLive, as shown in Figure 7.29.

7.4.2 FP7 FIRE TRESCIMO Project

Overview

TRESCIMO is a FIRE-funded project that focuses on collaboration between Europe and South Africa to address environmental challenges due to the increase in urbanization in underdeveloped countries. Besides conducting field studies, FIRE experimenters can provision an SDI that can be used to evaluate related technologies for a variety of different scenarios. The TRESCIMO testbed is composed of a Smart City (SC) platform, an ETSI-/OneM2M-compliant M2M framework and a number of sensors and actuators.

FITeagle

FITeagle is used to dynamically instantiate the virtualized environment and to offer it to the GENI and FIRE community. As a result, the testbed has been added to the list of trusted authorities and is shown in the jFed Experimenter GUI (cf. Figure 7.30) and can be used to provision an experiment.

Topology

Such a topology is shown in Figure 7.31 and was loaded by an external URL. It contains a Smart City Platform (SCP) as a Platform (SCPaaP) that is connected to an M2M Server as a Service (M2MSRVaaS) instance. This server in turn communicates with an M2M Gateway as a Service (M2MGWaaS) that exposes information about an M2M Device as a Service (M2MDEVaaS), which includes virtual and physical sensors and actuators.

OpenBaton

As depicted in the overall architecture in Figure 7.32, while FITeagle is used to export FIRE-related APIs to experimenters and to manage physical devices, OpenBaton

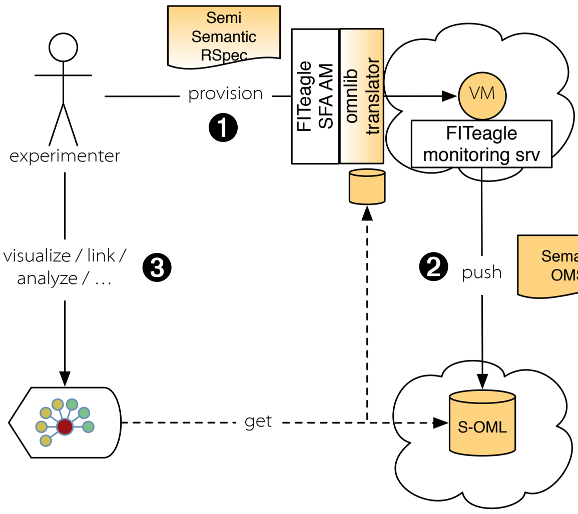


Figure 7.28: Example Fed4FIRE semantic resource description workflow

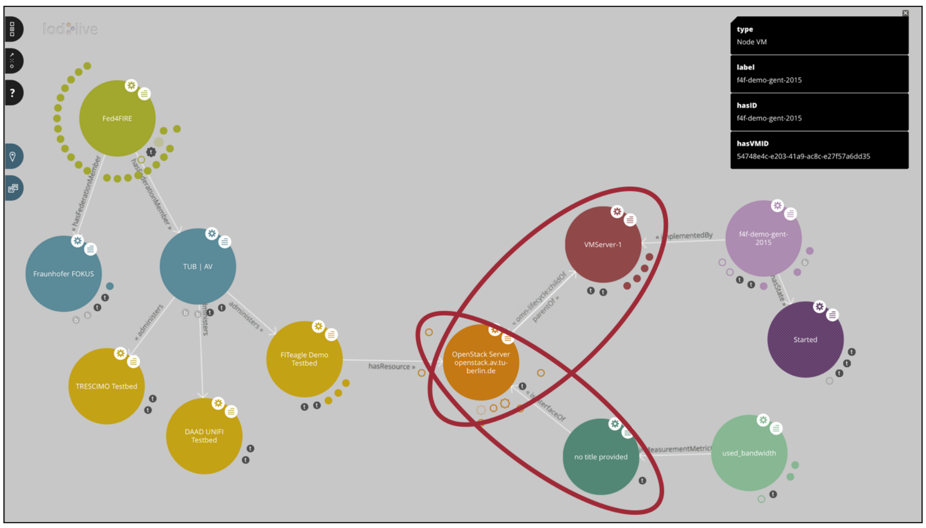


Figure 7.29: Example Fed4FIRE semantic resource description graph

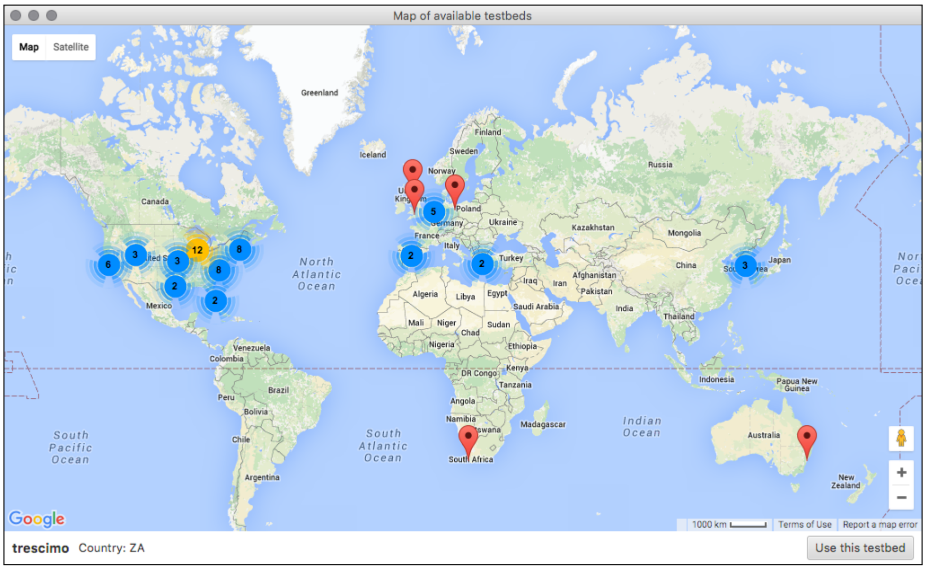


Figure 7.30: Geographic jFed testbed overview

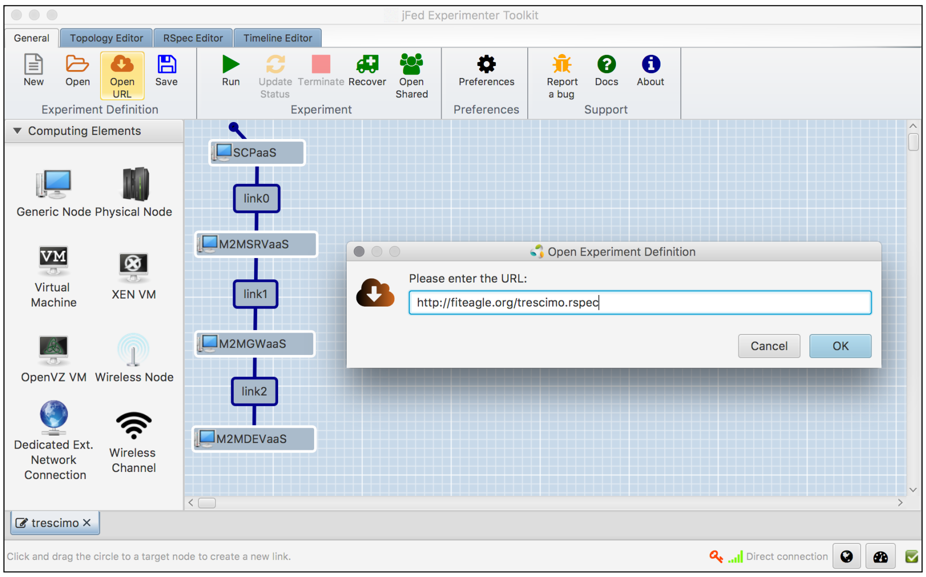


Figure 7.31: Requested Smart City software stack in the jFed experimenter GUI

is used to provision virtualized services within the distributed OpenStack-based testbed. The FITeagle framework handles all aspects of the experiment life cycle, including AuthN and AuthZ, based on X.509 certificates signed by trusted CAs, such as used in Fed4FIRE or PLE. OpenBaton is an open-source ETSI NFV MANO-compliant NFVO to provision and control VNFs. Depending on the selected location, the relevant OpenStack sites are contacted to instantiate the requested services. Both systems communicate with each other using a native JSON/REST interface and, as the *omnlib* translator supports the relevant data models, information is additionally exchanged using a TOSCA-compliant API.

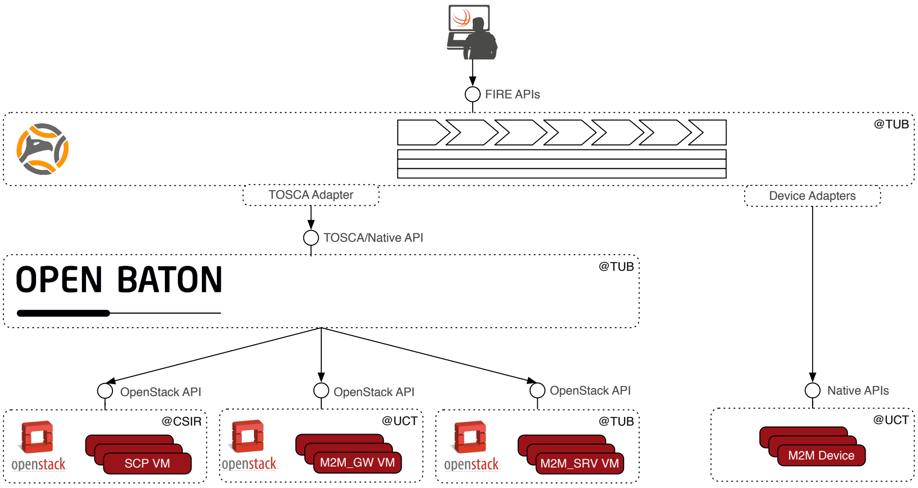


Figure 7.32: Overall TRESCIMO architecture

Experiment

The underlying workflow with references to the related SFA AM method calls is depicted in Figure 7.33. In the example experiment shown in Figure 7.34, a developer of an SME is implementing devices to be used in a SC context. For this experiment it is assumed that, in order to overcome interoperability issues, the devices are compliant with the OneM2M standard. However, due to the lack of an appropriate SC test infrastructure to evaluate the devices, the developer can't proceed to the next stage of the development phase. As both SME and TRESCIMO are part of the FIRE initiative, the developer can use resources from a testbed for this purpose. As a result, the experimenter uses the jFed client to gain access to the required services offered by the FITeagle server.

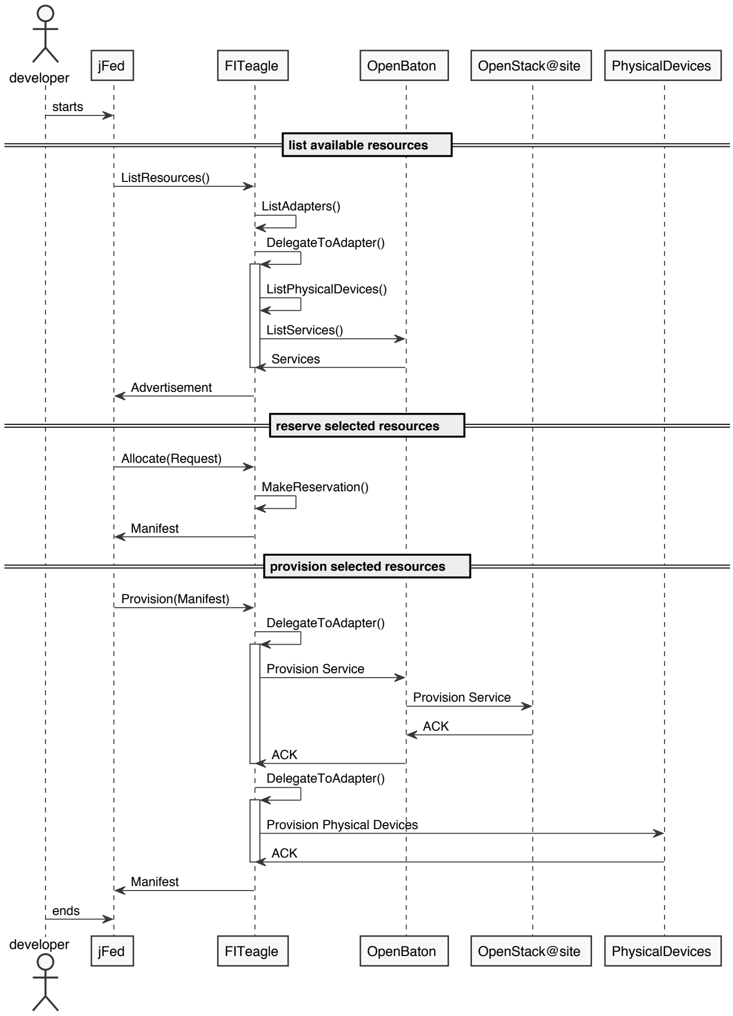


Figure 7.33: Message flow between FITeagle, OpenBaton and devices

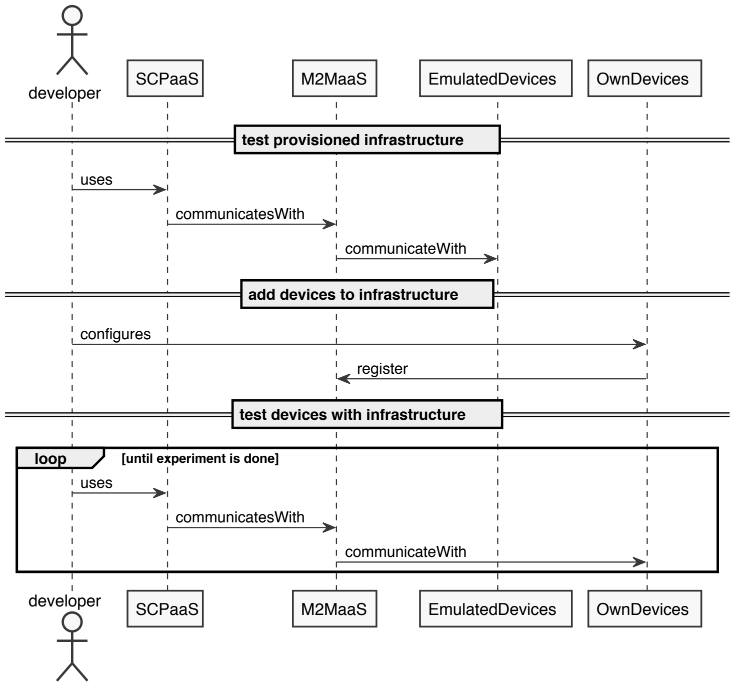


Figure 7.34: Developer testing devices within the TRESCIMO testbed

7.5 Deployments

Overview Besides its use in numerous research projects, the prototype has been deployed in different testbed setups. Within this section, a few of theses installations will be described to validate the functionalities of the reference implementation in different environments.

7.5.1 Fraunhofer FUSECO Playground Testbeds

Overview Operated by the Fraunhofer Institute for Open Communication Systems the FUSECO Playground[6] offers a set of testbeds covering multiple technologies related to Next Generation Mobile Networks (NGMNs) for research and prototype development. The playground allows Proof-of-Concept (PoC) validation in multiaccess network environments, M2M sensor networks, and SDN and VNF cloud setups in a number of use case scenarios, based on the toolkits Open5GCore, OpenSDNCore, Open5GMTC[7], and OpenBaton.

Integration In Figure 7.35 an overview of the FUSECO Playground is given, where the *Remote Access* and *Federation* functionalities are handled by FITeagle. The central *Testbed Control Center* is implemented by both, the OpenBaton and FITeagle frameworks. This allows, on the one hand, the management of 5G-related resources, focused on SDNs, VNFs, and M2M communication in an OpenStack cloud environment. The wireless lab, on the other hand, provides access to physical devices interconnected over multiple technologies.

[6]http://fuseco-playground.org
[7]http://open5gmtc.net

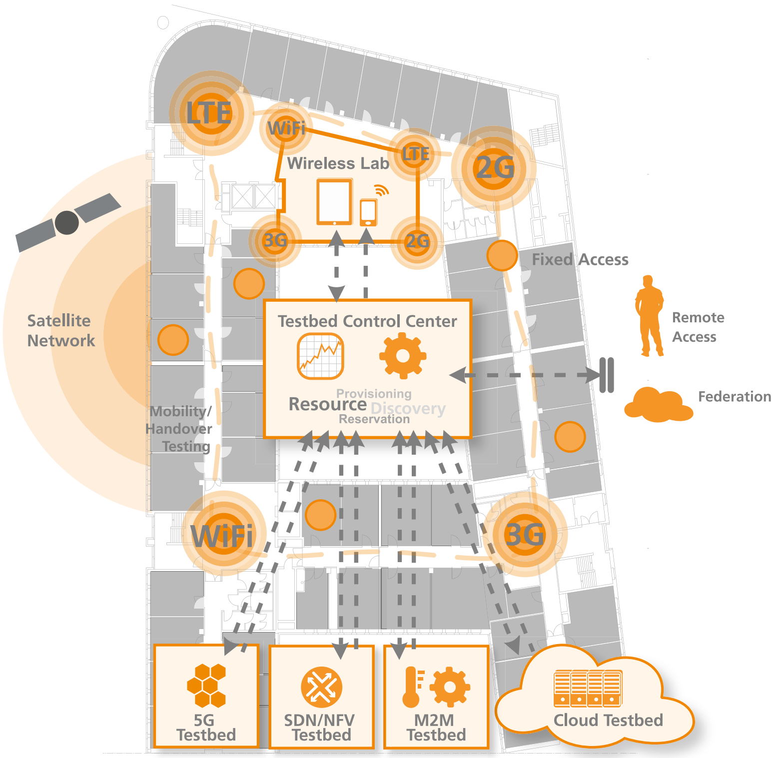


Figure 7.35: Fraunhofer FUSECO Playground testbed [m6]

7.5.2 Poznan Supercomputing and Networking Center Testbed

The PL-LAB[8] [p142] testbed was built within the Future Internet Engineering (FIE) project and consists of eight distributed laboratories hosted by research institutions in Poland, interconnected by the PIONIER NREN. PL-LAB offers numerous hardware resources aimed to support experiments related to low-level network FI research, with a focus on Content Aware Networks, IPv6 deployments, high-bandwidth video streaming, and infrastructure virtualization. The hardware includes general purpose servers, programmable switching and measurement devices, traffic generators, routers and application-specific equipment. *Overview*

The PL-LAB testbed, depicted in Figure 7.36, uses FITeagle as its SFA AM implementation. Resources that can be provisioned include physical servers, Juniper MX80/240 routers and Virtex2/Virtex5 NetFPGA cards. Their internal testbed management service, PL-LAB Access System, has been extended to offer a REST-based API to FITeagle adapters, which have been developed during the course of the FITeagle integration. *Integration*

[8] http://pllab.pl

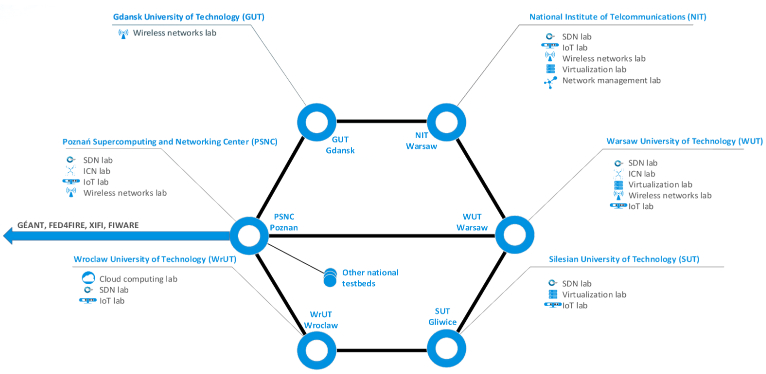


Figure 7.36: PL-LAB testbed [m10]

7.5.3 IEEE Intercloud Testbed

Overview The IEEE Intercloud approach was described briefly in Sec. 2.2.3 and has been used to validate the applicability for the Intercloud context of the resource ontology developed in this thesis and the FITeagle management architecture. The work has been discussed and deployed in the Intercloud Testbed Project[9], which is the reference implementation of the IEEE P2302 working group.

Architecture Figure 7.37 depicts the reference network topology and elements of the foreseen Intercloud architecture. The public clouds play the role of Internet Service Providers (ISPs). Private clouds are infrastructures that are used within a single administrative domain. Accordingly, the Intercloud exchanges match the Internet exchanges and peering points where these clouds can interoperate, and the Intercloud root is a distributed entity that contains services for trust, naming and discovery functionalities. Finally, the gateway implements the required Intercloud protocols and acts analogous to an Internet router.

Integration Figure 7.38 gives an overview of the proof-of concept setup executed using FITeagle as the underlying framework. It consists of a central XMPP server as the main communication channel, one Intercloud root that stores information in an RDF triplet store, and two Intercloud gateways (named Alice and Bob) from which information about available resources is pushed. While the architecture needed to cover the whole service life cycle is depicted in Figure 2.7, the focus in the initial evaluation was on the establishment of an initial API and a serialized formal information model, which was based on the P2302 ontology and was extended using OMN concepts. Based on this model and exemplary data sets, it was possible to semantically describe the aforementioned setup in a distributed manner and to forward this information from the gateways to the root.

7.6 Code Verification

Overview A number of metrics has been used to evaluate the quality of the source code written during the course of this thesis. Therefore, this section provides the results of relevant static analyzers and dynamic tests.

[9] http://intercloudtestbed.org

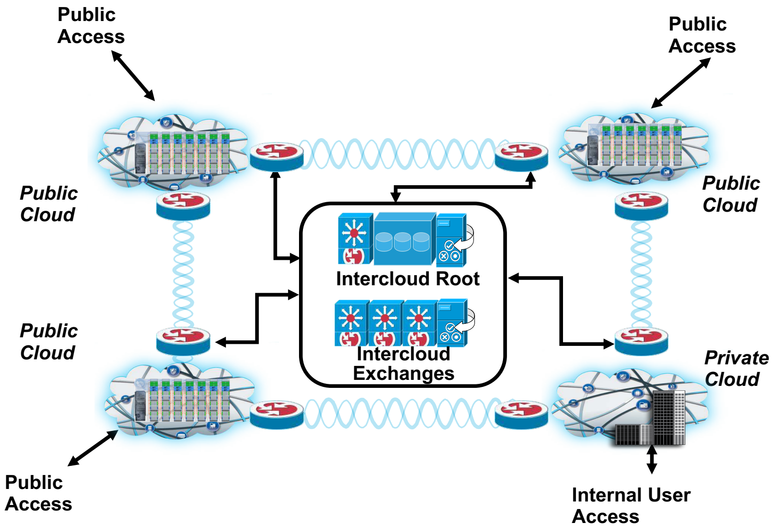


Figure 7.37: IEEE Intercloud reference network topology and elements [t9]

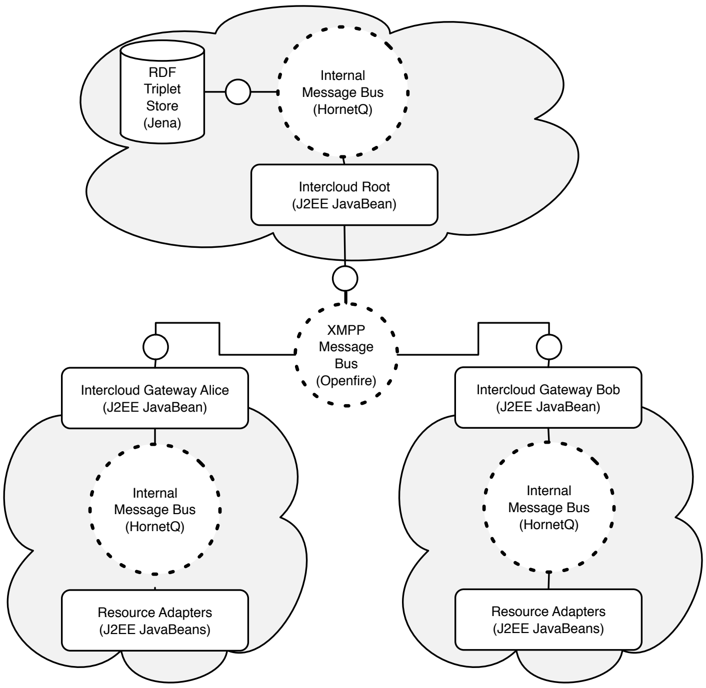


Figure 7.38: IEEE Intercloud testbed demonstration topology [a10]

7.6.1 Ontology

Static Tests To ensure quality and reusability of the information model, the ontology was encoded using OWL2, due to its expressive power and broad adoption. The principles described in the AMOR Manifesto[10] were been followed, which are an adoption of the 5 star LOD requirements[11] for ontologies. Further, any change to the ontologies is automatically checked using Apache Jena Eyeball inspectors (cf. Listing 7.14). Other validators, such as the OntOlogy Pitfall Scanner (OOPS) [p186], are executed manually.

Listing 7.14: Verification of the ontology set

```
$ time ./bin/rdflint.sh -assume ontologies/*.ttl import/*.ttl -check ontologies/*.ttl; echo $?
Running Apache Eyeball...
real 0m8.366s
user 0m18.429s
sys 0m1.003s
0
$
```

Metadata As part of the design process, steps were taken to ensure the broadest possible dissemination of the ontology. As shown in Listing 7.15, the Dublin Core (DC), Vocabulary for Annotating Vocabulary Descriptions (VANN), Vocabulary of a Friend (VOAF) and Creative Commons (CC) vocabularies are used to describe metainformation about the ontologies. The specified concepts and properties are linked (at least using *rdfs:seeAlso*) to existing counterparts in the NML, INDL, NOVI, NDL-OWL, W3C Geo, W3C Time, Good Relations (GR) [p115] and OWL-S ontologies.

Listing 7.15: OMN metainformation (excerpt)

```
<http://open-multinet.info/ontology/omn> rdf:type owl:Ontology, voaf:Vocabulary ;
    dc:title "Open-Multinet␣Upper␣Ontology"@en ;
    dc:description "This␣ontology␣defines..."@en ;
    rdfs:label "omn"@en ;
    vann:preferredNamespacePrefix "omn" ;
    vann:preferredNamespaceUri <http://open-multinet.info/ontology/omn#> ;
    owl:versionInfo "2015-04-22"^^xsd:string ;
    dc:publisher <http://open-multinet.org/> ;
    dcterms:license <http://creativecommons.org/licenses/by/4.0/> ;
    dc:creator <http://alex.willner.ws/about#me> ;
    dc:contributor <http://www.commit-nl.nl/people/morsey>, ...
```

Linked Open Vocabulary Further, the guidelines from [t10] and best practices[12] have been followed for publishing the files. The URL `http://open-multinet.info/ontology/omn` provides human-readable documentation using Live OWL Documentation Environment (LODE) [p180] and machine-readable serializations using an RDF translator [m14].

[10] http://knowledgecraver.blogspot.de/2013/04/the-amor-manifesto.html
[11] http://www.w3.org/DesignIssues/LinkedData.html
[12] http://www.w3.org/TR/swbp-vocab-pub/

Additionally, the permanent identifier https://w3id.org/omn has been registered with the W3C Permanent Identifier Community Group, which acts as a more stable alternative to purl.org.

Visibility

To expand its visibility, the root ontology was published to repositories operated by the Open Knowledge Foundation (OKFN), such as Linked Open Vocabulary (LOV). Next, the *omn* namespace http://prefix.cc/omn was registered. The ontology was also submitted to Swoogle[p61] and Watson[p53]. Finally, the upper ontology is embedded, together with other related metadata, in the PDF version of this thesis using ISO standard Extensible Metadata Platform (XMP).

7.6.2 FITeagle

Continuous Integration

Following the TDD paradigm, every source change within any FITeagle module is tested automatically after each commit to the code repository by executing the relevant JUnit tests. As shown in Figure 7.39, tests are executed within eight repositories, including an integration test in which all components are deployed, tested by executing the acceptance tests described in Section 7.2.2, and, finally, uploaded to a Maven artifact repository.

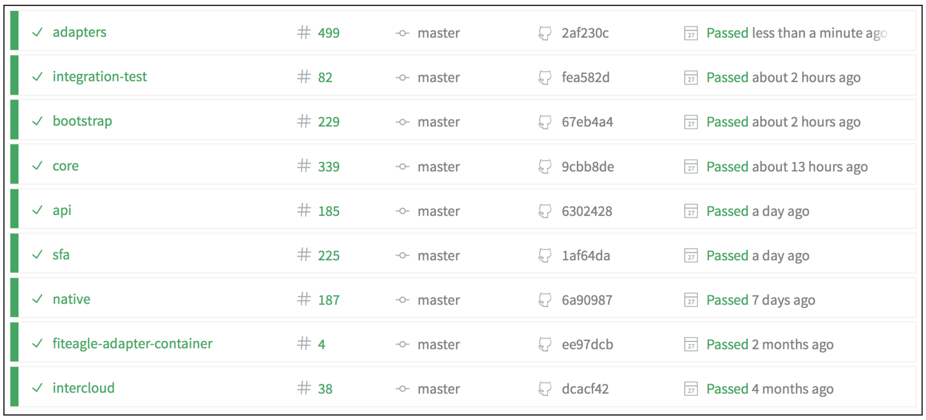


Figure 7.39: Test status overview of all FITeagle modules

Test Coverage

Verifying the functionality of the code by executing dynamic and acceptance tests after each commit ensures that the expected functionality was implemented without changing existing functionalities. The value of these tests, however, depends on the overall test coverage of code segments. While an increase of code coverage generally increases the reliability of the developed software, a reasonable coverage rate depends on the specific code base, as only parts of it might contain logic that can and should be tested. To find untested parts of the code base, Coveralls[13] was used after each commit. Figure 7.40 shows the test coverage within the *omnlib* translation module.

[13] https://coveralls.io

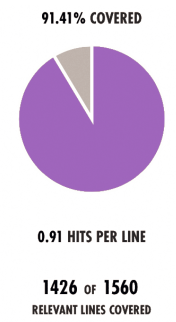


Figure 7.40: Line coverage within the translator module using Coveralls

Static Tests

Additionally, static code tests were executed against the code base. Based on the comparison of code analyzers in [p199], two different frameworks were chosen. In Listings 7.16 and 7.17, the result of analyzing the motor adapter code is shown using FindBugs[14] [p117] and PMD[15] respectively.

Listing 7.16: Analyzing the motor code base using FindBugs

```
$ mvn findbugs:check
[INFO] Scanning for projects...
[...]
[INFO] --- findbugs-maven-plugin:3.0.2:check (default-cli) @
    motor ---
[INFO] BugInstance size is 0
[INFO] Error size is 0
[INFO] No errors/warnings found
[INFO] -------------...
[INFO] BUILD SUCCESS
[INFO] -------------...
[INFO] Total time: 14.949 s
[INFO] Finished at: 2015-12-16T10:48:57+01:00
[INFO] Final Memory: 31M/210M
[INFO] -------------...
$
```

Listing 7.17: Analyzing the motor code base using PMD

```
$ mvn -X pmd:check
[INFO] Scanning for projects...
[...]
[INFO] --- maven-pmd-plugin:3.5:check (default-cli) @ motor ---
[...]
[DEBUG] PMD failureCount: 0, warningCount: 0
[INFO]
[INFO] -------------...
[INFO] BUILD SUCCESS
[INFO] -------------...
```

[14] http://findbugs.sf.net
[15] https://pmd.github.io

```
[INFO] Total time: 5.963 s
[INFO] Finished at: 2015-12-16T10:57:24+01:00
[INFO] Final Memory: 25M/214M
[INFO] -------------...
$
```

7.7 Comparative Analysis

Overview

After the experimental validation, performance evaluation, observational validation, code verification, and the description of selected deployments, in this section the work conducted within this thesis is compared against the requirements identified before and against approaches similar to the developed framework.

7.7.1 Requirement Evaluation

Table 7.1

In Chapter 3 the requirements from the perspective of the three main stakeholders were listed. They were further divided based on the seven phases of the experiment life cycle, the three underlying trustworthiness-related functionalities, and the overarching API. As a result, 33 requirements were identified against which the work conducted in this thesis is compared against in Table 7.1. It was further highlighted in Figure 1.2, that the thesis focused on two main areas. First, the API level with support for multiple protocols (P11), multiple tools (U11), and compatibility across the alliance (O11). These requirements were the main driving factors for the design of the decoupled FIRMA architecture presented in Chapter 5. Second, the resource description phase to publish information (P1), discover resources (U1), and harmonize the descriptions (O1). These requirements were the starting point for the design of the formal information models FIDDLE and OMN presented in Chapter 4.

Table 7.1: Mapping requirements against own approach

Requirements	Approach
U11 P11 O11 (API)	By decoupling the delivery mechanisms in the FIRMA design, all key functionalities are supported by multiple interfaces. This includes the handover between protocols using the FITeagle framework and the means to add further APIs that might be required for additional areas of application of the developed framework.
U1 P1 O1 (Discovery)	The level of abstraction for the description of resources was raised from the data model to the information model layer by designing the FIDDLE/OMN ontologies. This allows to a user to locate a wide range of heterogeneous resources within a federation, e.g. by invoking SPARQL queries in the DBcloud. The queries can include details about the unique resource types, relationships, dependencies, and arbitrary properties. Further, infrastructure providers are able to publish resource information with the required level of detail and within the federation the available information can be linked and merged.

Requirements	Approach
U2 P2 O2 (Selection)	The formal information models support the construction of federation-wide topologies that can comprise configuration parameters, dependencies, attributes and potential interdomain connectivity. Further, the mapping of abstract resource requirements to concrete resource instances is supported by the models and could be implemented by resource schedulers.
U3 P3 O3 (Reservation)	Support for time- and quantity-based reservation information has been added to the information models and are supported by the FITeagle implementation. The models and implementation can be extended to add further resource-specific properties. Further, scheduling information can be exported and be used by resource schedulers.
U4 P4 O4 (Provisioning)	The ontologies support the modeling of allocation and instantiation information and FITeagle offers an SFA interface to provision virtual and physical resources based on such a graph. FITeagle also provides remote access to these resources and can configure additional services, such as monitoring. Further, the information models support the definition of dependencies, which could be used by an extended orchestration module for service chaining and federation-wide orchestration.
U5 P5 O5 (Monitoring)	Monitoring information about the underlying infrastructure, on which the experiment is executed, can be modeled using the ontologies, queried from adapters and exported using the framework. Additionally, the overall status of a facility can be imported and exported using OMSP. As the monitoring information is RDF-encoded as well, it could be linked to other information in the graph on a federation and infrastructure level.
U6 P6 O6 (Control)	Users can control the provisioned topology using a REST-based API, based on the same information model used for the provisioning and monitoring phases. Adding an FRCP-based API could further allow a federation-wide event- and time-based workflow management.
U7 P7 O7 (Termination)	Releasing the provisioned topology is supported by both, the information models and the framework. The user, however, is responsible for the permanent storage of experiment results and configurations. Further, due to its message-driven architecture, FITeagle conceptually supports the deployment of service integrators to generate utilization reports or invoke billing systems to create usage-based invoices.
U8 P8 O8 (AuthN)	The implemented framework supports federation-wide AuthN by accepting only X.509 credentials in the SFA interface that are signed or delegated by trusted third parties. While FITeagle can further act as an IdP on its own, the AuthN mechanisms have not been implemented for the native REST API and the OMSP API, as the specification for the latter does not yet cover AuthN aspects.

Requirements	Approach
U9 P9 O9 (AuthZ)	Based on the AuthN support described above, basic trust relationships can be established and a list of privileges, predefined by GENI, are evaluated per request. However, defining federation-wide policy rules and ensuring their compliance based on specific security assertions, requires further work and harmonization, potentially on a semantic level.
U10 P10 O10 (Trust)	Support for exporting infrastructure monitoring information via OMSP for SLA compliance evaluation has been implemented. Further, FITeagle can consume these measurement streams and an additional module could evaluate the information against specific SLAs. However, the specification of SLAs and OLAs, in particular based on semantic information models, were out of scope.

7.7.2 Comparison with Other Approaches

Overview

In Chapter 6 the framework FITeagle was presented, which main purpose is to allow infrastructure owners to join a federation for FI experimentation, while supporting multiple APIs and formal information models. Following the same structure used in the subsection before, related work with similar objectives are compared with FITeagle in Table 7.2.

Table 7.2

As described in Sections 4.2.6 and 6.3, in the FI context only the ORCA framework Shirako [t17] focuses on exploiting Semantic Web approaches to address the Requirements P1, U1,O1, together with related requirements such as resource mapping and query validation. However, during the preparation of the thesis, the OMF system, presented in Section 2.3.1, was extended recently [p209] to cover the Requirements P11, U11, O11 (called OMF++ in Table 7.2). It can be summarized that at the time this thesis was written, no other approach besides FITeagle was identified that supports both multiple APIs and semantic information models to cover most of the requirements.

Table 7.2: Comparison of related work with own approach

Requirements	SFAWrap	OMF6++	OML	Teagle	Shirako	FITeagle
U11 P11 O11 (API)	**(+)** SFA AM v3, SFA SA	**(++)** SFA AM v2, FRCP, native REST	**(o)** OMSP	**(-)** native REST	**(o)** SFA AM v2	**(+++)** SFA AM v3, SFA SA, OMSP, native REST, native Web-Socket
U1 P1 O1	(+)	(++)		(+)	(++)	(+++)

Requirements	SFAWrap	OMF6++	OML	Teagle	Shirako	FITeagle
(Discovery) and U2 P2 O2 (Selection)	GENI RSpec v3	GENI RSpec v2, native JSON	—	VCT / DEN-ng	GENI RSpec v2, NDL-OWL internally	GENI RSpec v3, OMN
U3 P3 O3 (Reservation)	(o) via drivers	(++) quota- / role-based	—	—	(++) two-step ticket / lease based	(+) quota-based
U4 P4 O4 (Provisioning)	(+) via drivers	(+) via controllers	—	(+) via adapters	(+) via agents	(+) via adapters
U5 P5 O5 (Monitoring)	—	(+) via OML	(++) OMSP streams	—	—	(++) via adapters and OMSP streams
U6 P6 O6 (Control)	—	(+++) via OEDL, native REST	—	(+) via FCI	—	(+) via native REST, native Web-Socket
U7 P7 O7 (Termination)	(+) via drivers	(+) via controllers	—	(+) via adapters	(+) via handlers	(+) via adapters
U8 P8 O8 (AuthN)	(+) PKI	(+) PKI	—	(-) username / password	(+) PKI	(+) PKI, username / password
U9 P9 O9 (AuthZ)	(o) via driver	(+++) formally specified	—	(+) OMA PE	(++) ABAC	(+) trust- and attribute-based
U10 P10 O10						(+)

Requirements	SFAWrap	OMF6++	OML	Teagle	Shirako	FITeagle
(Trust)	—	—	—	—	—	OMSP export to FLS

7.8 Conclusion

Summary

In this chapter, the tools and models developed in this thesis were evaluated by executing a manual experiment and automated conformity tests, by analyzing the performance of the OMN translation mechanism and the FITeagle management system, by demonstrating their utilization within research projects and testbed deployments and, finally, by analyzing the source code using static and dynamic tests.

Results

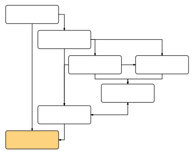
It was shown that the approach enables the management of resources based on a dynamic semantic graph, including the creation and termination of instances. The approach is further independent of the involved deliver mechanisms or resources, can be applied to different fields of application, and repetitive execution does not influence the overall stability. Information about the same resource, originating from different sources within the experiment life cycle, can be linked to form a coherent graph that represents a knowledge base that can be queried to discover and reuse facts.

Chapter 8

Summary and Further Work

8.1 Overview

Contributions



This chapter presents a synopsis of the work presented in this thesis (cf. Figure 8.1), which addressed the semantic-based life cycle management of resources within federated infrastructures [a22]. The research area under consideration was narrowed down from Distributed Resource Management to Future Internet experimentation in federated infrastructures with a focus on resource description and information exchange. The key contributions were the definition of a formal information model [a20] and its international standardization [a11, a24], the specification of a microservices-based architecture to dynamically modify a semantic graph representing the managed resources [a21], and the implementation and validation of the concept [a23]. Further, as resource sharing within federated testing facilities is a specific form of distributed computing, the presented work is applicable to further fields of application, such as SDN-based Intercloud Computing environments [a10, a17], mobile network operators [a13], including the management of VNFs.

Dissemination

Throughout the course of the research for this thesis, a number of dissemination activities carried out: six conference and workshop papers (one still under review) reflecting the core contributions [a11, a20–a24], nine publications about related fields of application [a6, a7, a10, a12–a14, a16–a18], and eight papers and presentations and one book chapter about preliminary work on Bandwidth on Demand (BoD) and infrastructure provisioning [a1–a5, a8, a15, a19, a25]. In addition, two international consortia were established, namely the Open-Multinet Forum and the W3C Federated Infrastructures Community Group, and one workshop on Semantic Web for Federated Software Defined Infrastructures (SWSDI) at the Extended Semantic Web Conference (ESWC) was organized. Finally, the management software developed has been used in

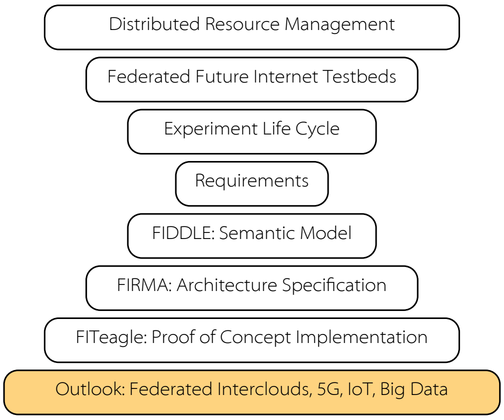


Figure 8.1: Placement of the outlook in the structure of research

four different testbeds, in five international research projects, and the implementation of the translation service has been adopted within one external toolkit.

8.2 Conclusions and Impact

Context

The experiment life cycle in the context of FI research provides a number of interesting research questions as highlighted in Chapter 2. Together with the requirements presented in Chapter 3, it became evident that the key question on how to describe resources and harmonize and link information is an important and long-standing open issue in this area of research. It strongly impacts each step of the experiment life cycle and, within this thesis, three main contributions were made to this end.

FIDDLE and Impact

First, in Chapter 4 the formal information model FIDDLE was introduced. Related work in the field of resource description in federated environments and, in particular, the Semantic Web have been incorporated. This approach allows information about highly heterogeneous resources from different administrative domains to be linked and combined, using a homogeneous data model. It further enables model validation and descriptive error detection, inference knowledge, such as equality, availability or connectivity of resources, by autonomous reasoning, and, finally, allows for complex declarative resource discovery. This work presents a foundation of how to describe and exchange information between interconnected testbeds and federated infrastructures in general. In order to sustain the conducted work on an international level and to extend it independently of the scope of research projects and this thesis, the Open-Multinet Forum[1] and subsequently the W3C Federated Infrastructures Community Group were created and are chaired by the author of this thesis.

FIRMA and Impact

Second, in Chapter 5 a reusable and extensible architecture was defined to manage heterogeneous resources on a semantic level independent of their type or specific APIs. Based on a semantic, message-bus-driven microservices design pattern, FIRMA takes related work in the field of reusable and distributed software architectures into account and allows each step of the experiment life cycle to be incorporated. Depending on their functionality and the given context, several delivery mechanisms (i.e. protocol implementations) can be offered at once to different federations, information can be

[1] `http://open-multinet.info`

integrated and handed over between life cycle phases and external services. Due to its generic design and the emphasis on formal information models, the architecture can be applied to further fields of application and has exemplarily been evaluated within the IEEE working group P2302.

FITeagle and Impact

Third, in Chapter 6 a prototype of the presented architecture has been implemented, using the introduced ontology. It served as a proof-of-concept and as a basis for the continuous refinement of the design principles. In Chapter 7 the applicability of the implementation was evaluated and its compliancy with related protocol standards shown. As a result, a GENI-/FIRE-compliant testbed can be bootstrapped in under four minutes by executing a single command. The framework has been used by a number of testbeds, such as the Fraunhofer FUSECO Playground, PL-LAB, and the University of Cape Town (UCT) and Technische Universität Berlin (TUB) infrastructures in the context of research projects such as Fed4FIRE, TRESCIMO, FLEX, UNIFI, and INFINITY.

8.3 Outlook

Intro

Wherever interoperability between different systems is required, processing of the information exchanged is a point of interest. The semantic description of resources and related data in order to overcome interoperability issues is currently playing an important role in a number of different research areas. With increasing interconnectivity and distribution of infrastructures and their resources, the topic is gaining even more importance. Following Metcalfe's law, the value of having semantically enriched information is proportional to the square of the number of corresponding sources [p114]. Below an outlook is given that reflects a number of foreseeable fields of application for the ontology, architecture and framework developed in this thesis.

Federated Testbeds

The context of experimentation within federated testbeds provided the main requirements and evaluation scenarios for this thesis. With regards to the experiment life cycle, the focus was set on the description of testbeds that participate in one or multiple federations. This description can build a basis that can be exploited by each other phase of the life cycle. For the selection of resources, the mapping between unbound subtopology requests and the available resources can be solved descriptively on a semantic level; and the same holds true for the integration of complex reservation information in this decision process [p98]. Within the provisioning phase, semantic annotations, such as monitoring data, can continuously be evaluated to elastically orchestrate and optimize resource allocation. Introducing the same formal information models the control of resources would allow seamless handovers and integration between each phase and enable fully automated reproducible large-scale experiments. Implementing an FRCP-compliant interface would further provide an added value to the framework. Further, the security and trustworthiness layers would benefit from a homogeneous information model and architecture. One example is the definition of federated authorization attributed and rules, based on work that was published 2006 in [p188] by defining an extended XACML architecture. Finally, elevating the focus from the infrastructure to the federation level would increase the complexity of each of the challenges described above. The work in this thesis is envisioned to be extended and applied as appropriate in future research projects and to provide the implementations to further testbeds.

Intercloud Computing

The work presented in this thesis is aligned with the current Experimentation as a Service (EaaS) trend, in which resources needed for an experiment are virtualized on demand. This paradigm shift will require further alignment of the existing APIs and RSpecs with concepts developed within the cloud computing context. Given its

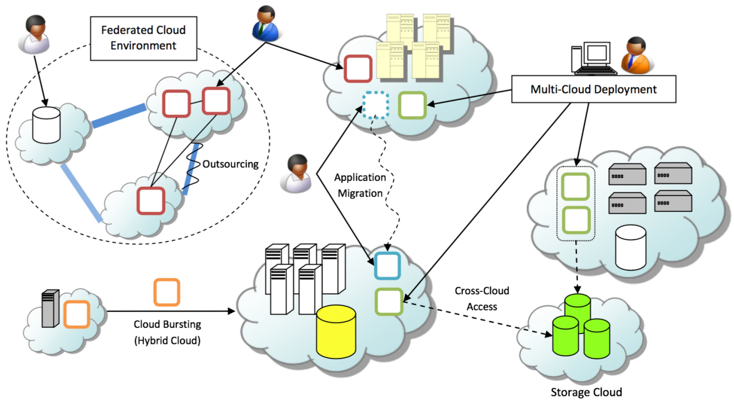


Figure 8.2: Intercloud taxonomy overview [p218]

architectural similarity and commercial relevance, the federated cloud context is of interest (cf. Figure 8.2). Initial work in this area was conducted in the IEEE P2302 Intercloud testbed and the SIIF working group. While the standard provides interesting foundations for realizing a volunteer peer-to-peer Intercloud approach for trading services based on constraint optimization algorithms, many research questions arise. The specified architectural components (roots, exchanges, gateways) have not yet been implemented and the FITeagle framework could be extended to act as a prototype [a10]. Besides an initial ontology based on the outcomes of the mOSAIC project, a number of aspects of federation have not yet been covered, but been already specified in the FIDDLE model [a10]. Additionally, some functional elements (name spaces, presence, messaging, trust) and governance elements (registration, geo-independency, trust anchors) have not yet been defined in detail and parts of the FIRMA architecture could act as a foundation for these. Further, other SDOs such as GICTF, NIST or ITU-T are working on additional approaches and it is not yet clear how the internationally accepted Intercloud standard will look like.

Federated Networks

Another possible field of application is the management of network connectivity over multiple administrative domains. To establish virtual links between these infrastructures, certain QoS parameters, such as high bandwidth demands, have to be ensured. SLA-aware connection management and bandwidth brokering for VPNs have been a topic more than a decade [p55, p248]. The emphasis subsequently shifted to the management of optical exchanges and multidomain management approaches [p60, p231], based on the availability of Cross Border Dark Fibers (CBDFs), To promote this paradigm, the international GLIF was established, the GLIF Interdomain Resource Reservation Architecture (GIRRA) [p127] was proposed and frameworks such as Harmony [a19] were implemented. In addition, programs such as the Defense Advanced Research Projects Agency (DARPA) Core Optical Networks: Architecture, Protocols, Control and Management (CORONET) [p44] were started to offer wavelength services to dynamically setup dedicated end-to-end paths. For example, CORONET supports the setup of 100 Tbps multidomain paths within 100 ms. Along with this work, the standardization of the relevant APIs and information models has begun. Not surprisingly, the adoption of semantic information models in this context has been suggested, e.g. by the introduction of the NDL, which enables sophisticated pathfinding algorithms and lead to the OGF

NML standard. The NML is currently a candidate for the underlying information model for the OGF multidomain network provisioning standard NSI. Further, the Optical Internetworking Forum (OIF), for example, defined requirements for an intelligent multidomain optical carrier network control plane [t53]. Given these developments and based on the present work, semantic-based software defined bearer Intercloud networks have been proposed in [a17] and could be further aligned in the context of Open Cloud Exchanges (OCXs) [p58].

Internet of Things

A complementary field of application is the exchange of highly heterogeneous information between distributed devices, generally known as IoT, Internet of Everything (IoE), or Web of Things (WoT) [p106, t58] with a focus on W3C technologies such as RDF. Devices in this context need to be discovered, their types and capabilities have to be described, measurement information have to be distributed, and actuation on devices invoked. To address data integration and interoperability issues, semantic models again have been proposed by a number of working groups, e.g. in the IERC AC4. Another current example that ratifies the importance of semantic interoperability in the IoT context, is the establishment of the *IoT Semantic Interoperability Workshops*[2] organized by the IETF Internet Architecture Board (IAB). The concept of managing semantic information about Things is known as Graph of Things (GoT) [p184] and mainly involves the discovery of information and invoking actuation on devices only partly. The dynamic manipulation of such a graph, based on provisioned resources within virtualized environments, seem to be an open research issue. Further, by focusing on the network, a semantic integration with the M2M layer would allow sophisticated optimizations. Devices that are part of the IoT and that are operated in a managed network, e.g. within a Smart City environment [a9], could autonomously change their network behavior based on the semantic context. SDOs such as ETSI M2M [t79] started to identify semantic interoperability issues and have been taken up by the OneM2M MAS group [t84].

Big Data

"Data is the 21st century's new raw material." [t42] and information produced within the IoT and the trend to Open Data will tremendously increase the amount of available data. As reported by the International Data Corporation (IDC), 90% of all data worldwide was generated in the last two years. It is further projected that in 2020 the total amount of data will reach around 40 Zettabytes (ZBs), which is over 5.000 Gigabytes (GBs) of data for every person on earth. Enabling the analysis of such diverse data sources to turn them into meaningful information and actions, will create new capabilities and account for unprecedented economic opportunities for businesses, individuals, and countries. Currently the cloud computing paradigm is being used to store and process very large, centralized data sets for data mining and real-time analytics. However, besides the increasing volume of data, other peculiarities such as variety, velocity, veracity, variability, vigilancy, virality and viscosity have to be taken into account. To address these issues and to further support privacy and allow information integration, reasoning and autonomous local processing, the work presented in this thesis could be used and extended to enable a semantic-based Fog Computing [p26] paradigm. In this paradigm, the analytic logic can be moved dynamically closer to the data, e.g. to the administrative domains in a federated context, to overcome the barriers of centralized processing. Approaches including e.g. CQELS or Yahoo Glimmer[3] [p21, p163] show that real-time and efficient distributed processing over large RDF data sets is possible. Such distributed processing allows queries in near real-time over the Web Data

[2] `https://www.iab.org/activities/workshops/iotsi/`
[3] `https://github.com/yahoo/Glimmer`

Commons (WDC) [p165] data set, which contains 160 Terabytes (TBs) of compressed structured data with over 20 billion triplets.

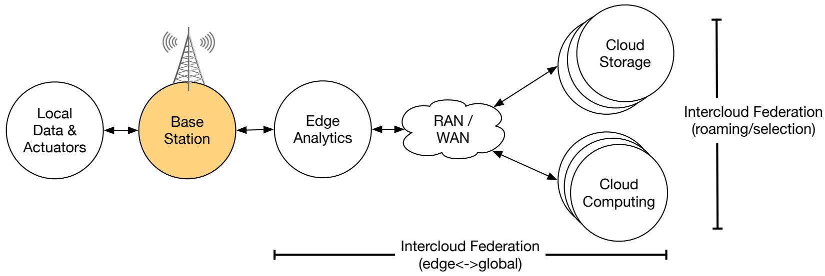


Figure 8.3: Horizontal and vertical federation for the mobile edge computing paradigm

5G

A specific use case where functionality is dynamically distributed closer to the edge of a network is the 5G. 5G will support, among other specifications, data rates up to 1 GB/s, End-to-End (E2E) latency of 1 ms, and ubiquitous connectivity. To address some of these metrics, intelligent (autonomous) NF placement and service chaining in particular will play an important role. Based on, for example, the ETSI-driven NFV standardization initiative, VNFs will be provisioned on demand topologically closer to the end device and will be interconnected with dedicated SDNs to reduce latency and increase security, e.g. on the edge of the Radio Access Network (RAN) in base stations. This mechanism is also denoted as “network slicing” and Mobile Edge Computing (MEC) [p15] (cf. Figure 8.3) and a number of data models have been proposed for VNFMs to form a corresponding dependency graph and describing resource types, relationships and properties. As mentioned in Chapter 6, examples include the IETF proposal for a VNF data model [t92], OASIS TOSCA, the OpenStack HOTs, and the AWS CloudFormation model. However, these approaches consider mainly schema-based tree data models with arbitrary extensions (e.g. “anyxml” tags) serialized in JSON, XML, YAML or YANG. Analogous to the challenges addressed in this thesis, this approach could introduce similar interoperability issues. Not only are SDN implementations such as OpenFlow evolving into distributed Self Organising Network (SON) architectures [p131], also 5G networks “will operate in a highly heterogeneous environment characterized by the existence of multiple types of access technologies, multilayer networks, multiple types of devices, multiple types of user interactions” [t48]. As suggested in [a13], adopting the approaches of this thesis may allow a reduction in potential interoperability issues by exploiting Semantic Web capabilities. Besides native support for linking different services with each other based on formal property specifications, it would be possible to reason over such a dependency graph to enable sophisticated SFC management, similar to the pathfinding approach presented in [p227].

Third Network

Based on the idea of managing multiple NFV domains using SDNs, architectures, such as the DIstributed SDN COntrol plane (DISCO) [p182], and the vision of a new “Third Network” paradigm were established. As shown in Figure 8.4 the MEF LSO defines an architecture for service orchestration on top of ONF SDN and ETSI NFV MANO interfaces. This enables agile networks that deliver assured connectivity services orchestrated across network domains (SD-WANs) between physical or virtual service endpoints. Due to its estimated market value of 2.75 billion USD in 2019 [t60], the industry driven TMF Zero-touch Orchestration, Operations and Management (ZOOM) program develops relevant standards for service provider support systems. In this context it is again of particular importance to ensure interoperability, potentially on the semantic level, as the concept involves multiple administrative domains and heterogeneous resources.

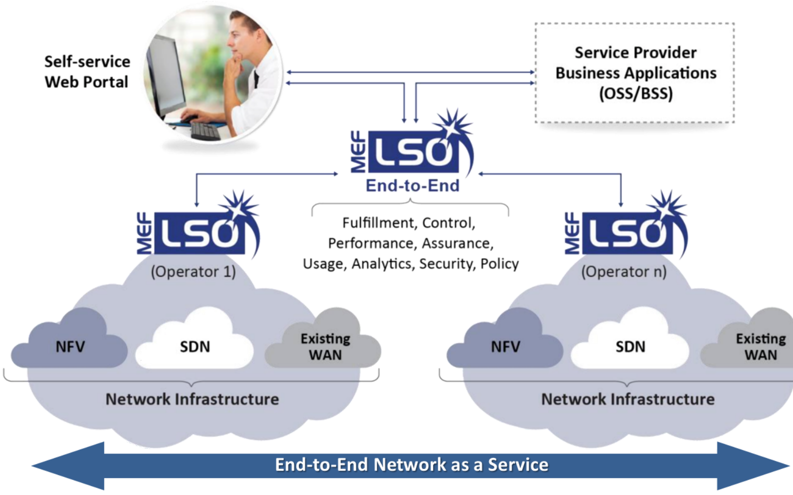


Figure 8.4: Conceptional MEF LSO model [t85]

Education

Finally, another interesting field of application is the description and management of Massive Open Online Courses (MOOCs) [m11]. A paradigm shift from small lectures at universities to large-scale online education is eminent [m11]. The user base of systems such as Moodle[4] [p62], Edx[5], or CourseSites[6] is continuously growing. In addition, research projects such as Forging Online Education through FIRE (FORGE) or UNIFI integrate existing infrastructures into lectures to allow students to remotely conduct experiments. The need to semantically describe and discover the heterogeneous educational materials in such a federated environment has already been identified. While the XML-based standard Sharable Content Object Reference Model (SCORM) [p22] was proposed in 2000, subsequent proposals such as [p7] suggests RDF models instead. The Learning Resource Metadata Initiative (LRMI), for example, strives to extend Schema.org and DC ontologies to fill this gap. As a next step, courses could also be reserved, provisioned and monitored along with the required resources based on a Graph of Education (GoE).

[4] https://moodle.org
[5] https://edx.org
[6] https://coursesites.com

APPENDIX A

ONTOLOGY SPECIFICATIONS

A.1 FIDDLE

The formal FIDDLE information model that has been described in Chapter 4 and presented the starting point for the formal OMN information model, is depicted in Listing A.1 as TTL-serialized RDF graph.

Listing A.1: FIDDLE ontology

```
@prefix : <urn:fiddle:ontology#> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix nml: <http://schemas.ogf.org/nml/2013/05/base#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix xml: <http://www.w3.org/XML/1998/namespace#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix indl: <http://www.science.uva.nl/research/sne/indl#> .
@prefix novi: <http://fp7-novi.eu/im.owl#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix void: <http://rdfs.org/ns/void#> .
@prefix dcterms: <http://purl.org/dc/terms/> .
@base <urn:fiddle:ontology#> .
<urn:fiddle:ontology> rdf:type owl:Ontology ;
 dc:date "2014-12-18"^^xsd:date;
 owl:versionInfo "2014-12-18"^^xsd:string;
 dc:title "Federated␣Infrastructure␣Description␣and␣Discovery␣Language"@en;
 dc:creator <http://alex.willner.ws/about#me>;
 dcterms:license <http://creativecommons.org/licenses/by/4.0/> .
#################################################################
```

```
#
# Object Properties
#
################################################################
### urn:fiddle:ontology#hasFederationMember
:hasFederationMember rdf:type owl:ObjectProperty ;
                    rdfs:label "has federation member"@en ;
                    rdfs:comment "a federation can have an organization as a member"@en ;
                    rdfs:domain :Federation ;
                    rdfs:range :FederationMember .
### urn:fiddle:ontology#partOfFederation
:partOfFederation rdf:type owl:ObjectProperty ;
                 rdfs:label "is part of federation"@en ;
                 rdfs:comment "an organization can be part of a federation"@en ;
                 rdfs:range :Federation ;
                 rdfs:domain :FederationMember ;
                 owl:inverseOf :hasFederationMember .
### urn:fiddle:ontology#administers
:administers rdf:type owl:ObjectProperty ;
            rdfs:label "administers"@en ;
            rdfs:comment "an organization (e.g. a federation member) administers its own infrastructure"@en ;
            rdfs:range :Infrastructure ;
            owl:inverseOf :isAdministeredBy ;
            rdfs:domain foaf:Organization .
### urn:fiddle:ontology#isAdministeredBy
:isAdministeredBy rdf:type owl:ObjectProperty ;
                 rdfs:label "is administered by"@en ;
                 rdfs:comment "an infrastructure can be administered by an organization (e.g. a federation member)"@en ;
                 rdfs:domain :Infrastructure ;
                 rdfs:range foaf:Organization .
### urn:fiddle:ontology#hasGroup
:hasGroup rdf:type owl:ObjectProperty ;
             rdfs:label "has group"@en ;
             rdfs:seeAlso nml:hasTypology ;
             rdfs:range :Group ;
             rdfs:domain :Group .
### urn:fiddle:ontology#hasComponent
:hasComponent rdf:type owl:ObjectProperty ;
             rdfs:label "has resource component"@en ;
             rdfs:seeAlso novi:hasComponent ;
             rdfs:range :Component ;
             rdfs:domain :Resource .
### urn:fiddle:ontology#componentOf
:componentOf rdf:type owl:ObjectProperty ;
            rdfs:label "is component of a resource"@en ;
            rdfs:domain :Component ;
            rdfs:range :Resource ;
            owl:inverseOf :hasComponent .
### urn:fiddle:ontology#requires
```

```
:requires rdf:type owl:ObjectProperty ;
            rdfs:label "requires␣an␣ICT␣object"@en ;
            rdfs:range :Object ;
            rdfs:domain :Object ;
            owl:inverseOf :requiredBy .
### urn:fiddle:ontology#requiredBy
:requiredBy rdf:type owl:ObjectProperty ;
            rdfs:label "is␣required␣by␣an␣ICT␣object"@en ;
            rdfs:range :Object ;
            rdfs:domain :Object ;
            owl:inverseOf :requires .
### urn:fiddle:ontology#hasAttribute
:hasAttribute rdf:type owl:ObjectProperty ;
            rdfs:label "has␣attribute"@en ;
            rdfs:domain :Object ;
            rdfs:range :Attribute .
### urn:fiddle:ontology#isAttributef
:isAttributeOf rdf:type owl:ObjectProperty ;
             rdfs:label "is␣attribute␣of"@en ;
             rdfs:range :Object ;
             rdfs:domain :Attribute ;
             owl:inverseOf :hasAttribute .
### urn:fiddle:ontology#hasResource
:hasResource rdf:type owl:ObjectProperty ;
            rdfs:label "has␣resource"@en ;
            rdfs:seeAlso novi:contains ;
            rdfs:domain :Infrastructure ;
            rdfs:range :Resource ;
            owl:inverseOf :resourceOf .
### urn:fiddle:ontology#resourceOf
:resourceOf rdf:type owl:ObjectProperty ;
           rdfs:label "is␣resource␣of␣an␣infrastructure"@en ;
           rdfs:range :Infrastructure ;
           rdfs:domain :Resource .
### urn:fiddle:ontology#hasService
:hasService rdf:type owl:ObjectProperty ;
           rdfs:label "has␣service"@en ;
           rdfs:domain :Infrastructure ;
           rdfs:range :Service .
### urn:fiddle:ontology#isServiceOf
:isServiceOf rdf:type owl:ObjectProperty ;
            rdfs:label "serves␣an␣infrastructure"@en ;
            rdfs:range :Infrastructure ;
            rdfs:domain :Service ;
            owl:inverseOf :hasService .
### urn:fiddle:ontology#implements
:implements rdf:type owl:ObjectProperty ;
           rdfs:label "implements"@en ;
           rdfs:seeAlso novi:implements ;
           rdfs:domain :Resource ;
           owl:inverseOf :implementedBy ;
           rdfs:seeAlso indl:implements ;
           rdfs:range [ rdf:type owl:Class ;
                        owl:unionOf ( :Service
```

```
                                    :VirtualResource
                                  )
                     ] .
### urn:fiddle:ontology#implementedBy
:implementedBy rdf:type owl:ObjectProperty ;
              rdfs:label "is␣implemented␣by"@en ;
              rdfs:seeAlso novi:implementedBy ;
              rdfs:range :Resource ;
              rdfs:domain [ rdf:type owl:Class ;
                           owl:unionOf ( :Service
         :VirtualResource
       )
                         ] .
#################################################################
#
# Data properties
#
#################################################################
### urn:fiddle:ontology#hasEndpoint
:hasEndpoint rdf:type owl:DatatypeProperty ;
            rdfs:domain novi:Service ;
            rdfs:range xsd:anyURI .
#################################################################
#
# Classes
#
#################################################################
### urn:fiddle:ontology#Component
:Component rdf:type owl:Class ;
           rdfs:subClassOf :Resource ;
          rdfs:label "Resource␣Component" .
### urn:fiddle:ontology#Federation
:Federation rdf:type owl:Class ;
           rdfs:label "Federation"@en ;
           rdfs:subClassOf foaf:Organization .
### urn:fiddle:ontology#FederationMember
:FederationMember rdf:type owl:Class ;
                 rdfs:label "member␣of␣a␣federation"@en ;
                 rdfs:subClassOf foaf:Organization .
### urn:fiddle:ontology#Group
:Group rdf:type owl:Class ;
      rdfs:label "Group"@en ;
      rdfs:subClassOf :Object ;
      owl:equivalentClass nml:Group ;
      rdfs:seeAlso novi:Group .
### urn:fiddle:ontology#Infrastructure
:Infrastructure rdf:type owl:Class ;
               rdfs:label "Infrastructure"@en ;
               rdfs:subClassOf :Group ;
               rdfs:comment "an␣infrastructure␣such␣as␣a␣testbed␣
                    or␣cloud␣facility"@en ;
               rdfs:seeAlso novi:Platform .
### urn:fiddle:ontology#Object
:Object rdf:type owl:Class ;
```

```
        rdfs:label "ICT␣Object"@en ;
        owl:equivalentClass nml:NetworkObject .
### urn:fiddle:ontology#Attribute
:Attribute rdf:type owl:Class ;
         rdfs:label "Attribute" .
### urn:fiddle:ontology#Resource
:Resource rdf:type owl:Class ;
         rdfs:subClassOf :Object ;
         rdfs:label "Resource"@en ;
         rdfs:seeAlso nml:NetworkObject .
### urn:fiddle:ontology#Service
:Service rdf:type owl:Class ;
         rdfs:subClassOf :Object ;
        rdfs:label "Service"@en ;
        rdfs:seeAlso indl:Capability .
### urn:fiddle:ontology#VirtualResource
:VirtualResource rdf:type owl:Class ;
               rdfs:subClassOf :Resource ;
                rdfs:label "Virtual␣Resource"@en ;
                owl:equivalentClass indl:VirtualNode .
### Generated by the OWL API (version 3.5.0) http://owlapi.sourceforge.net
```

A.2 OMN

Overview

The upper part of the formal OMN information model that has been developed based on the FIDDLE ontology, is depicted in Listing A.2. Note that the OMN ontology consists of multiple subontologies of which, due to space limitations, only the Life Cycle ontology is being presented in Listing A.3, as it presents the underlying concepts and properties to semantically manage the resource life cycle. Further, the FIDDLE, OMN and OMN-Lifecycle ontologies have been embedded, along with other related metadata, into the PDF version of this thesis as XML-serialized annotations using the International Organization for Standardization (ISO) standard Extensible Metadata Platform (XMP).

Listing A.2: OMN upper ontology

```
@prefix : <http://open-multinet.info/ontology/omn#> .
@prefix voaf: <http://purl.org/vocommons/voaf#> .
@prefix vann: <http://purl.org/vocab/vann/> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix nml: <http://schemas.ogf.org/nml/2013/05/base#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix xml: <http://www.w3.org/XML/1998/namespace#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix indl: <http://www.science.uva.nl/research/sne/indl#> .
@prefix novi: <http://fp7-novi.eu/im.owl#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix color: <http://geni-orca.renci.org/owl/app-color.owl#> .
@prefix schema: <http://schema.org/> .
```

```
@prefix dcterms: <http://purl.org/dc/terms/> .
@prefix collections: <http://geni-orca.renci.org/owl/collections.owl#> .
@prefix move: <http://www.ontologydesignpatterns.org/cp/owl/move.owl#> .
@prefix time: <http://www.w3.org/2006/time#> .
@prefix gr: <http://purl.org/goodrelations/v1#> .
@prefix dctype: <http://purl.org/dc/dcmitype/> .
@prefix service: <http://purl.org/ontology/service#> .
@prefix cc: <http://creativecommons.org/ns#> .
@prefix owl-s: <http://www.daml.org/services/owl-s/1.0DL/Service.owl#> .
@base <http://open-multinet.info/ontology/omn#> .
<http://open-multinet.info/ontology/omn> rdfs:comment "This ontology defines the most abstract concepts and properties that are needed to semantically manage resource within federated infrastructures."@en ;
                        rdf:type owl:Ontology,
                        voaf:Vocabulary ;
                        rdfs:label "omn"@en ;
                        vann:preferredNamespacePrefix "omn"^^xsd:string ;
                        vann:preferredNamespaceUri <http://open-multinet.info/ontology/omn#> ;
                        dc:date "2014-11-11"^^xsd:date ;
                        dcterms:modified "2015-04-18"^^xsd:date ;
                        owl:versionInfo "2015-04-18"^^xsd:string ;
                        dc:title "Open-Multinet Upper Ontology"@en ;
                        dc:description "This ontology defines the most abstract concepts and properties that are needed to semantically manage resource within federated infrastructures."@en ;
                        dc:description <http://raw.githubusercontent.com/open-multinet/playground-rspecs-ontology/master/ontologies/pics/omn.png> ;
                        dc:creator <mailto:alexander.willner@tu-berlin.de> ;
                        dc:publisher <http://open-multinet.info/> ;
                        foaf:homepage <http://open-multinet.info/> ;
                        dcterms:license <http://creativecommons.org/licenses/by/4.0/> ;
                        cc:license <http://creativecommons.org/licenses/by/4.0/> ;
                        dc:rights <http://creativecommons.org/licenses/by/4.0/> ;
                        dc:contributor <mailto:brecht.vermeulen@iminds.be> ;
```

```
                        dc:contributor <mailto:thijs.walcarius@intec.ugent.be> ;
                        dc:contributor <mailto:jorge.lopez_vergara@uam.es> ;
                        dc:contributor <mailto:chrisap@noc.ntua.gr> ;
                        dc:contributor <https://www.linkedin.com/in/yahyaalhazmi> ;
                        dc:contributor <mailto:loughnane@campus.tu-berlin.de> ;
                        dc:contributor <https://staff.fnwi.uva.nl/p.grosso> ;
                        dc:contributor <http://www.commit-nl.nl/people/morsey> ;
                        dc:contributor <mailto:ibaldin@renci.org> ;
                        dc:contributor <mailto:yxin@renci.org> .
#################################################################
#
# Object Properties
#
#################################################################
### http://open-multinet.info/ontology/omn#adaptableFrom
:adaptableFrom rdf:type owl:ObjectProperty ;
       rdfs:label "adaptable␣from"@en ;
       rdfs:comment "determines␣the␣resource␣from␣which␣this␣resource␣can␣be␣adapted␣from␣-␣e.g.␣from␣an␣Ethernet␣to␣a␣FDDI␣port."@en ;
       rdfs:range :Layer ;
       rdfs:domain :Layer ;
       owl:inverseOf :adaptableTo .
### http://open-multinet.info/ontology/omn#adaptableTo
:adaptableTo rdf:type owl:ObjectProperty ;
           rdfs:label "adaptable␣to"@en ;
           rdfs:comment "determines␣to␣which␣resource␣this␣resource␣can␣adapts␣to␣-␣e.g.␣from␣an␣Ethernet␣to␣a␣FDDI␣port."@en ;
           rdfs:range :Layer ;
           owl:inverseOf :adaptableFrom ;
           rdfs:domain :Layer .
### http://open-multinet.info/ontology/omn#adaptsFrom
:adaptsFrom rdf:type owl:IrreflexiveProperty ,
             owl:ObjectProperty ;
          rdfs:label "adapts␣from"@en ;
          owl:inverseOf :adaptsTo ;
          rdfs:comment "determines␣from␣which␣resource␣this␣resource␣adapts␣-␣e.g.␣from␣an␣Ethernet␣to␣a␣FDDI␣port."@en ;
          #todo: domain and range must not be of the same type
          rdfs:range [ rdf:type owl:Class ;
               owl:unionOf ( :Component
                     :Group
                     :Resource
                     :Service
```

```
                    )
                ] ;
            rdfs:domain [ rdf:type owl:Class ;
                   owl:unionOf ( :Component
                          :Group
                          :Resource
                          :Service
                        )
                 ] .
### http://open-multinet.info/ontology/omn#adaptsTo
:adaptsTo rdf:type owl:IrreflexiveProperty ,
           owl:ObjectProperty ;
         rdfs:label "adapts␣to"@en ;
         rdfs:comment "determines␣to␣which␣resource␣this␣resource
             ␣adapts␣-␣e.g.␣from␣an␣Ethernet␣to␣a␣FDDI␣port."@en
             ;
         owl:inverseOf :adaptsFrom ;
         rdfs:domain [ rdf:type owl:Class ;
                owl:unionOf ( :Component
                       :Group
                       :Resource
                       :Service
                     )
              ] ;
         rdfs:range [ rdf:type owl:Class ;
               owl:unionOf ( :Component
                      :Group
                      :Resource
                      :Service
                    )
             ] .
### http://open-multinet.info/ontology/omn#dependsOn
:dependsOn rdf:type owl:ObjectProperty ;
          rdfs:label "depends␣on"@en ;
          rdfs:subPropertyOf :relatesTo ;
          rdfs:comment "claims␣dependency"@en ;
          rdfs:domain [ rdf:type owl:Class ;
                 owl:unionOf ( :Component
                        :Group
                        :Resource
                        :Service
                      )
               ] ;
          rdfs:range [ rdf:type owl:Class ;
                owl:unionOf ( :Component
                       :Group
                       :Resource
                       :Service
                     )
              ] .
### http://open-multinet.info/ontology/omn#relatesTo
:relatesTo rdf:type owl:ObjectProperty ;
          rdfs:label "relates␣to"@en ;
          rdfs:comment "claims␣a␣general␣dependency"@en ;
```

```
         rdfs:domain [ rdf:type owl:Class ;
               owl:unionOf ( :Component
                     :Group
                     :Resource
                     :Service
                   )
             ] ;
         rdfs:range [ rdf:type owl:Class ;
              owl:unionOf ( :Component
                    :Group
                    :Resource
                    :Service
                  )
            ] .
### http://open-multinet.info/ontology/omn#fromDependency
:fromDependency rdf:type owl:ObjectProperty ;
        rdfs:label "from␣dependency"@en ;
        rdfs:comment "claims␣dependency"@en ;
        rdfs:domain :Dependency ;
        owl:inverseOf :toDependency ;
        rdfs:range [ rdf:type owl:Class ;
                   owl:unionOf ( :Component
                         :Group
                         :Resource
                         :Service
                       )
                 ] .
### http://open-multinet.info/ontology/omn#hasAttribute
:hasAttribute rdf:type owl:ObjectProperty ;
             rdfs:label "has␣attribute"@en ;
             rdfs:comment "link␣to␣a␣general␣attribute␣of␣the␣
                 resource␣-␣e.g.␣to␣a␣ReadOnly␣class"@en ;
             rdfs:range :Attribute ;
             owl:inverseOf :isAttributeOf ;
             rdfs:domain [ rdf:type owl:Class ;
                   owl:unionOf ( :Component
                         :Group
                         :Resource
                         :Service
                       )
                 ] .
### http://open-multinet.info/ontology/omn#hasComponent
:hasComponent rdf:type owl:ObjectProperty ;
             rdfs:label "has␣component"@en ;
             rdfs:comment "component␣of␣the␣resource␣-␣e.g.␣a␣CPU
                 "@en ;
             owl:inverseOf :isComponentOf ;
             rdfs:seeAlso novi:hasComponent ;
             rdfs:range :Component ;
             rdfs:domain [ rdf:type owl:Class ;
                   owl:unionOf ( :Component
                         :Resource
                         :Service
                       )
```

```
                ] .
### http://open-multinet.info/ontology/omn#hasGroup
:hasGroup rdf:type owl:ObjectProperty ;
         rdfs:label "has group"@en ;
         rdfs:comment "a group that is related to this resource - e.g. a reserved topology within an infrastructure"@en ;
         rdfs:domain :Group ;
         rdfs:range :Group ;
         owl:inverseOf :isGroupOf .
### http://open-multinet.info/ontology/omn#hasResource
:hasResource rdf:type owl:ObjectProperty ;
            rdfs:label "has resource"@en ;
            rdfs:comment "a resource that this resource contains - e.g. a node within a reserved topology"@en ;
            rdfs:seeAlso novi:contains ;
            rdfs:domain :Group ;
            rdfs:range :Resource ;
            owl:inverseOf :isResourceOf .
### http://open-multinet.info/ontology/omn#hasService
:hasService rdf:type owl:ObjectProperty ;
           rdfs:label "has service"@en ;
           rdfs:comment "a service that this resource contains - e.g. a Hadoop instance within a reserved topology"@en ;
           rdfs:range :Service ;
           rdfs:domain [ rdf:type owl:Class ;
                  owl:unionOf ( :Group
                         :Resource
                         :Service
                       )
                ] .
:hasReservation rdf:type owl:ObjectProperty ;
           rdfs:label "has reservation"@en ;
           rdfs:comment "the reservation details of a resource - e.g. an immediate reservation for 3 hours"@en ;
           rdfs:range :Reservation ;
           rdfs:domain [ rdf:type owl:Class ;
                  owl:unionOf ( :Group
                         :Resource
                         :Service
                       )
                ] .
### http://open-multinet.info/ontology/omn#isAttributeOf
:isAttributeOf rdf:type owl:ObjectProperty ;
       rdfs:comment "a general attribute of a resource - e.g. to a ReadOnly class"@en ;
       rdfs:label "is attribute of"@en ;
       rdfs:domain :Attribute ;
       owl:inverseOf :hasAttribute ;
       rdfs:range [ rdf:type owl:Class ;
                owl:unionOf ( :Component
                       :Group
                       :Resource
```

```
                        :Service
                      )
               ] .
### http://open-multinet.info/ontology/omn#isComponentOf
:isComponentOf rdf:type owl:ObjectProperty ;
       rdfs:comment "is␣component␣of␣a␣resource␣-␣e.g.␣a␣CPU␣in␣a␣PC"@en ;
       rdfs:label "is␣component␣of"@en ;
       rdfs:domain :Component ;
       owl:inverseOf :hasComponent ;
       rdfs:range [ rdf:type owl:Class ;
                  owl:unionOf ( :Component
                        :Resource
                        :Service
                      )
               ] .
### http://open-multinet.info/ontology/omn#isGroupOf
:isGroupOf rdf:type owl:ObjectProperty ;
         rdfs:comment "a␣group␣that␣is␣related␣to␣a␣resource␣-␣e.g.␣a␣reserved␣topology␣within␣an␣infrastructure"@en ;
         rdfs:label "is␣group␣of"@en ;
         rdfs:range :Group ;
         rdfs:domain :Group .
### http://open-multinet.info/ontology/omn#isResourceOf
:isResourceOf rdf:type owl:ObjectProperty ;
            rdfs:label "is␣resource␣of"@en ;
            rdfs:comment "a␣resource␣that␣another␣resource␣contains␣-␣e.g.␣a␣node␣within␣a␣reserved␣topology"@en ;
            rdfs:seeAlso novi:contains ;
            rdfs:range :Group ;
            rdfs:domain :Resource .
### http://open-multinet.info/ontology/omn#isServiceOf
:isServiceOf rdf:type owl:ObjectProperty ;
           rdfs:label "is␣service␣of"@en ;
           rdfs:comment "a␣service␣of␣a␣resource␣-␣e.g.␣a␣Hadoop␣instance␣within␣a␣reserved␣topology"@en ;
           rdfs:domain :Service ;
           owl:inverseOf :hasService ;
           rdfs:range [ rdf:type owl:Class ;
                owl:unionOf ( :Group
                      :Resource
                      :Service
                    )
             ] ;
          rdfs:seeAlso service:providedBy .
:isReservationOf rdf:type owl:ObjectProperty ;
           rdfs:label "is␣reservation␣of"@en ;
           rdfs:comment "the␣reservation␣details␣of␣a␣resource␣-␣e.g.␣an␣immediate␣reservation␣for␣3␣hours"@en ;
           rdfs:domain :Reservation ;
           owl:inverseOf :hasReservation ;
           rdfs:range [ rdf:type owl:Class ;
```

```
                owl:unionOf ( :Group
                      :Resource
                      :Service
                    )
              ] .
### http://open-multinet.info/ontology/omn#toDependency
:toDependency rdf:type owl:ObjectProperty ;
             rdfs:label "to␣dependency"@en ;
             rdfs:comment "claims␣dependency"@en ;
             rdfs:range :Dependency ;
             rdfs:domain [ rdf:type owl:Class ;
                   owl:unionOf ( :Component
                         :Group
                         :Resource
                         :Service
                       )
                ] .
### http://open-multinet.info/ontology/omn#withinEnvironment
:withinEnvironment rdf:type owl:ObjectProperty ;
           rdfs:label "within␣environment"@en ;
           rdfs:comment "within␣environment"@en ;
           rdfs:range :Environment ;
           rdfs:domain [ rdf:type owl:Class ;
                 owl:unionOf ( :Component
                       :Group
                       :Resource
                       :Service
                     )
               ] .
################################################################
#
# Data properties
#
################################################################
### http://open-multinet.info/ontology/omn#hasEndpoint
:hasEndpoint rdf:type owl:DatatypeProperty ,
              owl:FunctionalProperty ;
            rdfs:label "has␣endpoint"@en ;
            rdfs:comment "The␣URL␣of␣the␣API␣of␣a␣service"@en ;
            rdfs:domain :Service ;
            rdfs:range xsd:anyURI .
### http://open-multinet.info/ontology/omn#isReadonly
:isReadonly rdf:type owl:DatatypeProperty ,
             owl:FunctionalProperty ;
           rdfs:label "is␣read␣only"@en ;
           rdfs:comment "information/attribute␣that␣is␣not␣writable"@en ;
           rdfs:domain :Attribute ;
           rdfs:range xsd:boolean .
################################################################
#
# Classes
#
################################################################
```

```
### http://open-multinet.info/ontology/omn#Attribute
:Attribute rdf:type owl:Class ;
          rdfs:comment "Describes␣the␣attributes␣of␣an␣omn:Group,
              ␣omn:Resource,␣omn:Service␣or␣omn:Component␣in␣more
              ␣detail"@en ,
               "Examples:␣Monitoring␣information,␣Color␣
                   attributes,␣Reservation␣information,␣QoS,␣SLAs
                   ,␣Location,␣Configuration,␣..."@en ;
          rdfs:seeAlso color:ColorAttribute ,
               nml:Location .
### http://open-multinet.info/ontology/omn#Component
:Component rdf:type owl:Class ;
          rdfs:comment "Examples:␣CPU,␣Sensor,␣Core,␣Port,␣Image"
              @en ,
               "An␣Entity␣that␣is␣part␣of␣an␣omn:Resource␣or␣omn:
                   Service.␣It␣does␣not␣need␣to␣be␣an␣omn:
                   Resource␣or␣an␣omn:Service␣itself."@en ;
          rdfs:seeAlso dcterms:isPartOf ;
          rdfs:seeAlso move:formsPartOf ;
          rdfs:seeAlso schema:partOfSystem .
### http://open-multinet.info/ontology/omn#Dependency
:Dependency rdf:type owl:Class ;
           rdfs:comment "Examples:␣application␣coloring␣(in␣GENI␣
               context),␣orchestration␣needs␣dependencies"@en ,
                "Helps␣to␣defines␣a␣directional␣relationship␣
                    between␣omn:Resource,␣omn:Group,␣omn:
                    Component␣or␣omn:Service.␣It␣makes␣it␣
                    possible␣to␣annotate␣the␣dependencies␣with␣
                    additional␣properties."@en ;
           rdfs:seeAlso novi:implements ,
                color:ColorAttribute ,
                indl:implements .
### http://open-multinet.info/ontology/omn#Environment
:Environment rdf:type owl:Class ;
            rdfs:comment "Examples:␣interference,␣concurrent␣
                virtual␣machines,␣concurrent␣traffic,␣temperature,
                ␣heat,␣..."@en ,
                 "The␣operating␣conditions␣under␣which␣a␣omn:
                     Resource,␣omn:Group,␣omn:Service␣is␣
                     operating."@en ;
            rdfs:seeAlso schema:Place .
### http://open-multinet.info/ontology/omn#Group
:Group rdf:type owl:Class ;
      rdfs:comment "Examples:␣Bi-directional␣Link,␣..."@en ,
            "A␣collection␣of␣omn:Resource,␣omn:Service␣or␣omn:
                Group"@en ;
      rdfs:seeAlso novi:Group ,
            collections:Collection ,
            nml:Group .
### http://open-multinet.info/ontology/omn#Topology
:Topology rdf:type owl:Class ;
      rdfs:comment "Examples:␣Infrastructure,␣Reservation,␣Slice,
          ␣..."@en ,
```

```
            "A␣collection␣of␣omn:Resource,␣omn:Service␣or␣omn:Group"@en ;
      rdfs:subClassOf :Group .
### http://open-multinet.info/ontology/omn#Layer
:Layer rdf:type owl:Class ;
      rdfs:comment "Describes␣a␣place␣within␣a␣hierarchy␣a␣specific␣omn:Group,␣omn:Resource,␣omn:Service␣or␣omn:Component␣can␣adapt␣to."@en ,
            "Examples:␣In␣networking,␣an␣end-to-end␣connectivity␣has␣to␣be␣on␣the␣same␣layer␣(path␣finding).␣For␣resources,␣it␣can␣describe␣the␣capability␣to␣adapt␣to␣a␣virtualized␣version"@en ;
      rdfs:seeAlso nml:AdaptationService ,
            indl:VirtualNode .
### http://open-multinet.info/ontology/omn#Resource
:Resource rdf:type owl:Class ;
         rdfs:comment "Examples:␣Node,␣Link,␣People,␣..."@en ,
               "An␣Entity␣that␣can␣be␣provisioned/controlled/measured␣by␣APIs"@en ;
         rdfs:seeAlso novi:Node ,
               nml:Node .
### http://open-multinet.info/ontology/omn#Service
:Service rdf:type owl:Class ;
        rdfs:comment "An␣Entity␣that␣has␣an␣API/capability␣to␣use␣it,␣it␣may␣depend␣on␣an␣omn:Resource"@en ,
              "Examples:␣Aggregate␣Manager,␣Portal,␣Measurement␣Service,␣Hadoop,␣Broker,␣..."@en ;
        rdfs:seeAlso novi:Service ,
              nml:Service ,
              gr:ProductOrService,
              dctype:Service,
              schema:Service,
              service:Service,
              owl-s:Service .
### http://open-multinet.info/ontology/omn#Reservation
:Reservation rdf:type owl:Class ;
        rdfs:comment "A␣specification␣of␣a␣guarantee"@en ,
              "Examples:␣(Earliest)␣Start␣and␣(lates)␣end␣time,␣data␣volume,␣..."@en ;
        rdfs:seeAlso time:Interval .
time:Interval rdfs:subClassOf :Reservation .
#################################################################
#
# General axioms
#
#################################################################
[ rdf:type owl:AllDisjointClasses ;
  owl:members ( :Attribute
        :Component
        :Dependency
        :Environment
        :Group
        :Layer
        :Resource
```

```
        :Service
            )
] .
### Fixes for validation
owl:NamedIndividual rdf:type owl:Class .
owl:IrreflexiveProperty rdf:type owl:Class .
owl:AllDisjointClasses rdf:type owl:Class .
### Generated by the OWL API (version 3.5.0) http://owlapi.sourceforge.net
```

Listing A.3: OMN life cycle ontology

```
@prefix : <http://open-multinet.info/ontology/omn-lifecycle#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xml: <http://www.w3.org/XML/1998/namespace#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix dcterms: <http://purl.org/dc/terms/> .
@prefix vann: <http://purl.org/vocab/vann/> .
@prefix voaf: <http://purl.org/vocommons/voaf#> .
@prefix cc: <http://creativecommons.org/ns#> .
@prefix omn: <http://open-multinet.info/ontology/omn#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix gr: <http://purl.org/goodrelations/v1#> .
@base <http://open-multinet.info/ontology/omn-lifecycle#> .
<http://open-multinet.info/ontology/omn-lifecycle> rdf:type owl:Ontology, voaf:Vocabulary ;
                         rdfs:label "omn-lifecycle"@en ;
                         dcterms:created "2014-11-11"^^xsd:date ;
                         owl:versionInfo "2015-04-08"^^xsd:string ;
                         dc:title "Open-Multinet␣Upper␣Lifecycle␣Ontology"@en ;
                         dc:description "This␣ontology␣defines␣generic␣concepts␣related␣to␣the␣life␣cycle␣of␣resource␣or␣service."@en ;
                         dcterms:modified "2015-04-08"^^xsd:date ;
                         vann:preferredNamespaceUri <http://open-multinet.info/ontology/omn-lifecycle#> ;
                         vann:preferredNamespacePrefix "omn-lifecycle"^^xsd:string ;
                         dc:publisher <http://open-multinet.info/> ;
                         dc:creator <http://alex.willner.ws/about#me> ;
                         dc:description <https://raw.githubusercontent.com/open-multinet/playground-rspecs-ontology/master/ontologies/pics/omn-lifecycle.png> ;
                         dc:creator <http://alex.willner.ws/about#me> ;
```

```
                         dcterms:license <http://creativecommons.org/licenses/by/4.0/> ;
                         cc:license <http://creativecommons.org/licenses/by/4.0/> ;
                         dc:rights "This␣ontology␣is␣distributed␣under␣a␣Creative␣Commons␣Attribute␣License␣-␣http://creativecommons.org/licenses/by/4.0/"@en ;
                         dc:contributor <mailto:brecht.vermeulen@iminds.be> ,
                          <mailto:thijs.walcarius@intec.ugent.be> ,
                          <mailto:jorge.lopez_vergara@uam.es> ,
                          <mailto:chrisap@noc.ntua.gr> ,
                          <mailto:yahya.al-hazmi@tu-berlin.de> ,
                          <mailto:loughnane@campus.tu-berlin.de> ,
                          <https://staff.fnwi.uva.nl/p.grosso> ,
                          <http://www.commit-nl.nl/people/morsey> ,
                          <mailto:ibaldin@renci.org> ,
                          <mailto:yxin@renci.org> .
#################################################################
#
# Object Properties
#
#################################################################
### http://open-multinet.info/ontology/omn-lifecycle#hasReservationState
:hasReservationState rdf:type owl:FunctionalProperty ,
                owl:IrreflexiveProperty ,
                owl:ObjectProperty ;
           rdfs:domain omn:Reservation ;
           rdfs:range :ReservationState ;
           rdfs:subPropertyOf :hasState .
### http://open-multinet.info/ontology/omn-lifecycle#hasState
:hasState rdf:type owl:FunctionalProperty ,
          owl:IrreflexiveProperty ,
          owl:ObjectProperty ;
      rdfs:range :State ;
      rdfs:domain [ rdf:type owl:Class ;
           owl:unionOf ( omn:Component
                    omn:Group
                    omn:Resource
                    omn:Service
                  )
            ] .
### http://open-multinet.info/ontology/omn-lifecycle#usesService
:usesService rdf:type
          owl:IrreflexiveProperty ,
          owl:ObjectProperty ;
      rdfs:range omn:Service ;
      rdfs:domain [ rdf:type owl:Class ;
           owl:unionOf ( omn:Component
                    omn:Group
```

```
                    omn:Resource
                    omn:Service
                  )
              ] .
### http://open-multinet.info/ontology/omn-lifecycle#canImplement
:canImplement rdf:type owl:ObjectProperty ;
       rdfs:comment "That which does not currently implement, but can potentially implement a resource, service, component or group."@en ;
       owl:inverseOf :canBeImplementedBy ;
       rdfs:domain [ rdf:type owl:Class ;
            owl:unionOf ( omn:Component
                  omn:Group
                  omn:Resource
                  omn:Service
                    )
            ] ;
       rdfs:range [ rdf:type owl:Class ;
           owl:unionOf ( omn:Component
                     omn:Group
                     omn:Resource
                     omn:Service
                   )
              ] .
### http://open-multinet.info/ontology/omn-lifecycle#canBeImplementedBy
:canBeImplementedBy rdf:type owl:ObjectProperty ;
          rdfs:comment "That which is not currently implemented, but can potentially be implemented by a resource, service, component or group."@en ;
          rdfs:range [ rdf:type owl:Class ;
               owl:unionOf ( omn:Component
                    omn:Group
                    omn:Resource
                    omn:Service
                  )
              ] ;
          rdfs:domain [ rdf:type owl:Class ;
                owl:unionOf ( omn:Component
                     omn:Group
                     omn:Resource
                     omn:Service
                   )
              ] .
### http://open-multinet.info/ontology/omn-lifecycle#implementedBy
:implementedBy rdf:type owl:ObjectProperty ;
          rdfs:range [ rdf:type owl:Class ;
               owl:unionOf ( omn:Component
                    omn:Group
                    omn:Resource
                    omn:Service
                  )
              ] ;
```

```
          rdfs:domain [ rdf:type owl:Class ;
               owl:unionOf ( omn:Component
                    omn:Group
                    omn:Resource
                    omn:Service
                  )
             ] .
### http://open-multinet.info/ontology/omn-lifecycle#implements
:implements rdf:type owl:ObjectProperty ;
       owl:inverseOf :implementedBy ;
       rdfs:domain [ rdf:type owl:Class ;
            owl:unionOf ( omn:Component
                 omn:Group
                 omn:Resource
                 omn:Service
                   )
          ] ;
       rdfs:range [ rdf:type owl:Class ;
           owl:unionOf ( omn:Component
                    omn:Group
                    omn:Resource
                    omn:Service
                  )
             ] .
### http://open-multinet.info/ontology/omn-lifecycle#
    isReservationStateOf
:isReservationStateOf rdf:type owl:ObjectProperty ;
            rdfs:range omn:Reservation ;
            rdfs:domain :ReservationState ;
            rdfs:subPropertyOf :isStateOf .
### http://open-multinet.info/ontology/omn-lifecycle#isStateOf
:isStateOf rdf:type owl:ObjectProperty ;
      rdfs:domain :State ;
      owl:inverseOf :hasState ;
      rdfs:range [ rdf:type owl:Class ;
          owl:unionOf ( omn:Component
                   omn:Group
                   omn:Resource
                   omn:Service
                 )
            ] .
### http://open-multinet.info/ontology/omn-lifecycle#parentOf
:parentOf rdf:type owl:ObjectProperty ;
     rdfs:range [ rdf:type owl:Class ;
             owl:unionOf ( omn:Component
                  omn:Group
                  omn:Resource
                  omn:Service
                )
           ] ;
     rdfs:domain [ rdf:type owl:Class ;
          owl:unionOf ( omn:Component
                   omn:Group
                   omn:Resource
```

```
                    omn:Service
                  )
             ] .
### http://open-multinet.info/ontology/omn-lifecycle#childOf
:childOf rdf:type owl:ObjectProperty ;
     owl:inverseOf :parentOf ;
     rdfs:domain [ rdf:type owl:Class ;
          owl:unionOf ( omn:Component
                    omn:Group
                    omn:Resource
                    omn:Service
                  )
             ];
     rdfs:range [ rdf:type owl:Class ;
             owl:unionOf ( omn:Component
                   omn:Group
                   omn:Resource
                   omn:Service
                 )
            ] .
#################################################################
#
# Data properties
#
#################################################################
### http://open-multinet.info/ontology/omn-lifecycle#hasAuthenticationInformation
:hasAuthenticationInformation rdf:type owl:DatatypeProperty ;
                rdfs:comment "A specific authentification information for the management system"@en ;
                rdfs:seeAlso "GENI Slice X.509 certificates"@en ;
                rdfs:domain omn:Resource ,
                 omn:Service ;
                rdfs:range xsd:string .
### http://open-multinet.info/ontology/omn-lifecycle#hasID
:hasID rdf:type owl:DatatypeProperty ;
      rdfs:comment "A unique identifier set by the management system"@en ;
      rdfs:seeAlso "GENI Manifest RSpec v3: component_id"@en ;
      rdfs:domain omn:Resource ,
          omn:Service ;
      rdfs:range xsd:string .
#################################################################
#
# Classes
#
#################################################################
### http://open-multinet.info/ontology/omn#Attribute
omn:Attribute rdf:type owl:Class .
### http://open-multinet.info/ontology/omn#Component
omn:Component rdf:type owl:Class .
### http://open-multinet.info/ontology/omn#Group
omn:Group rdf:type owl:Class .
### http://open-multinet.info/ontology/omn#Reservation
```

```
omn:Reservation rdf:type owl:Class .
### http://open-multinet.info/ontology/omn#Resource
omn:Resource rdf:type owl:Class .
### http://open-multinet.info/ontology/omn#Service
omn:Service rdf:type owl:Class .
### http://open-multinet.info/ontology/omn#Topology
omn:Topology rdf:type owl:Class .
### http://open-multinet.info/ontology/omn-lifecycle#Active
:Active rdf:type owl:Class ;
    rdfs:label "Active"@en ;
    rdfs:subClassOf :State ;
    rdfs:seeAlso "GENI␣geni_ready_busy␣operational␣state"@en ;
    rdfs:comment "The␣related␣resource/service␣is␣actively␣performing␣an␣action"@en .
### http://open-multinet.info/ontology/omn-lifecycle#Allocated
:Allocated rdf:type owl:Class ;
      rdfs:label "Allocated"@en ;
      rdfs:subClassOf :ReservationState ;
      rdfs:seeAlso "GENI␣geni_allocated␣allocation␣state"@en ;
      rdfs:comment "The␣related␣resources/services␣are␣reserved"@en .
### http://open-multinet.info/ontology/omn-lifecycle#Cleaned
:Cleaned rdf:type owl:Class ;
     rdfs:label "Cleaned"@en ;
     rdfs:subClassOf :State ;
     rdfs:comment "The␣related␣resource/service␣has␣been␣cleaned"@en .
### http://open-multinet.info/ontology/omn-lifecycle#Confirmation
:Confirmation rdf:type owl:Class ;
         rdfs:subClassOf omn:Topology ;
         rdfs:comment "A␣collection␣(group)␣of␣resources/services/groups␣confirmed␣to␣be␣allocated␣for␣the␣user."@en .

### http://open-multinet.info/ontology/omn-lifecycle#Error
:Error rdf:type owl:Class ;
      rdfs:label "Error"@en ;
      rdfs:subClassOf :State ;
      rdfs:seeAlso "GENI␣geni_failed␣operational␣state"@en ;
      rdfs:comment "The␣related␣resource/service␣is␣in␣an␣error␣state"@en .
### http://open-multinet.info/ontology/omn-lifecycle#Initialized
:Initialized rdf:type owl:Class ;
        rdfs:label "Initialized"@en ;
        rdfs:subClassOf :State ;
        rdfs:comment "The␣related␣resource/service␣has␣been␣initialized"@en .
### http://open-multinet.info/ontology/omn-lifecycle#Installed
:Installed rdf:type owl:Class ;
      rdfs:label "Installed"@en ;
      rdfs:subClassOf :State ;
      rdfs:comment "The␣related␣resource/service␣has␣been␣installed"@en .
### http://open-multinet.info/ontology/omn-lifecycle#Manifest
:Manifest rdf:type owl:Class ;
```

```
      rdfs:subClassOf omn:Topology ;
      rdfs:comment "A␣collection␣(group)␣of␣resources/services/groups␣allocated␣for␣the␣user."@en .
### http://open-multinet.info/ontology/omn-lifecycle#NotYetInitialized
:NotYetInitialized rdf:type owl:Class ;
           rdfs:label "NotYetInitialized"@en ;
           rdfs:subClassOf :State ;
           rdfs:seeAlso "GENI␣geni_instantiating␣operational␣state"@en ;
           rdfs:comment "The␣related␣resource/service␣are␣not␣yet␣active/ready"@en .
### http://open-multinet.info/ontology/omn-lifecycle#Offering
:Offering rdf:type owl:Class ;
      rdfs:subClassOf omn:Topology ;
      rdfs:comment "A␣collection␣(group)␣of␣services␣and␣resources␣provided␣by␣an␣Infrastructure.␣The␣collection␣is␣the␣result␣of␣the␣application␣of␣Policies."@en ;
      rdfs:seeAlso gr:Offering .
### http://open-multinet.info/ontology/omn-lifecycle#Pending
:Pending rdf:type owl:Class ;
     rdfs:label "Pending"@en ;
     rdfs:subClassOf :State ;
     rdfs:seeAlso "GENI␣geni_pending_allocation␣operational␣state"@en ;
     rdfs:comment "The␣related␣resource/service␣is␣not␣yet␣provisioned"@en .
### http://open-multinet.info/ontology/omn-lifecycle#Preinit
:Preinit rdf:type owl:Class ;
     rdfs:label "Preinit"@en ;
     rdfs:subClassOf :State ;
     rdfs:seeAlso "GENI␣geni_configuring␣operational␣state"@en ;
     rdfs:comment "The␣related␣resource/service␣is␣currently␣configuring"@en .
### http://open-multinet.info/ontology/omn-lifecycle#Provisioned
:Provisioned rdf:type owl:Class ;
        rdfs:label "Provisioned"@en ;
        rdfs:subClassOf :ReservationState ;
        rdfs:seeAlso "GENI␣geni_provisioned␣allocation␣state"@en ;
        rdfs:comment "The␣related␣resources/services␣are␣provisioned"@en .
### http://open-multinet.info/ontology/omn-lifecycle#Ready
:Ready rdf:type owl:Class ;
       rdfs:label "Ready"@en ;
       rdfs:subClassOf :State ;
       rdfs:seeAlso "GENI␣geni_ready␣operational␣state"@en ;
       rdfs:comment "The␣related␣resource/service␣is␣ready"@en .
### http://open-multinet.info/ontology/omn-lifecycle#Removing
:Removing rdf:type owl:Class ;
      rdfs:label "Removing"@en ;
      rdfs:subClassOf :State ;
      rdfs:comment "The␣related␣resource/service␣gets␣removed"@en .
```

```
### http://open-multinet.info/ontology/omn-lifecycle#Request
:Request rdf:type owl:Class ;
    rdfs:subClassOf omn:Topology ;
    rdfs:comment "A␣collection␣(group)␣of␣resources/services/groups␣requested␣by␣the␣user"@en .
### http://open-multinet.info/ontology/omn-lifecycle#ReservationState
:ReservationState rdf:type owl:Class ;
         rdfs:label "Reservation␣State"@en ;
         rdfs:subClassOf omn:Attribute ;
         rdfs:comment "The␣current␣state␣of␣a␣reservation"@en .
### http://open-multinet.info/ontology/omn-lifecycle#Started
:Started rdf:type owl:Class ;
    rdfs:label "Started"@en ;
    rdfs:subClassOf :State ;
    rdfs:comment "The␣related␣resource/service␣has␣been␣started"@en .
### http://open-multinet.info/ontology/omn-lifecycle#State
:State rdf:type owl:Class ;
      rdfs:label "State"@en ;
      rdfs:subClassOf omn:Attribute ;
      rdfs:comment "The␣current␣state␣of␣the␣resource,␣service␣or␣group"@en .
### http://open-multinet.info/ontology/omn-lifecycle#Stopped
:Stopped rdf:type owl:Class ;
    rdfs:label "Stopped"@en ;
    rdfs:subClassOf :State ;
    rdfs:comment "The␣related␣resource/service␣is␣stopped"@en .
### http://open-multinet.info/ontology/omn-lifecycle#Stopping
:Stopping rdf:type owl:Class ;
     rdfs:label "Stopping"@en ;
     rdfs:subClassOf :State ;
     rdfs:seeAlso "GENI␣geni_stopping␣operational␣state"@en ;
     rdfs:comment "The␣related␣resource/service␣is␣stopping"@en .
### http://open-multinet.info/ontology/omn-lifecycle#Unallocated
:Unallocated rdf:type owl:Class ;
        rdfs:label "Unallocated"@en ;
        rdfs:subClassOf :ReservationState ;
        rdfs:seeAlso "GENI␣geni_unallocated␣allocation␣state"@en ;
        rdfs:comment "The␣related␣resources/services␣are␣not␣reserved"@en .
### http://open-multinet.info/ontology/omn-lifecycle#Uncompleted
:Uncompleted rdf:type owl:Class ;
        rdfs:label "Uncompleted"@en ;
        rdfs:subClassOf :State ;
        rdfs:comment "The␣related␣resource/service␣is␣not␣complete"@en .
### http://open-multinet.info/ontology/omn-lifecycle#Updating
:Updating rdf:type owl:Class ;
     rdfs:label "Updating"@en ;
     rdfs:subClassOf :State ;
     rdfs:comment "The␣related␣resource/service␣is␣getting␣updated"@en .
```

```
#################################################################
#
# Individuals
#
#################################################################
### http://open-multinet.info/ontology/omn-lifecycle#Active
:Active rdf:type :State ,
        owl:NamedIndividual .
### http://open-multinet.info/ontology/omn-lifecycle#Allocated
:Allocated rdf:type :ReservationState ,
           owl:NamedIndividual .
### http://open-multinet.info/ontology/omn-lifecycle#Cleaned
:Cleaned rdf:type :State ,
         owl:NamedIndividual .
### http://open-multinet.info/ontology/omn-lifecycle#Error
:Error rdf:type :State ,
       owl:NamedIndividual .
### http://open-multinet.info/ontology/omn-lifecycle#Initialized
:Initialized rdf:type :State ,
             owl:NamedIndividual .
### http://open-multinet.info/ontology/omn-lifecycle#Installed
:Installed rdf:type :State ,
           owl:NamedIndividual .
### http://open-multinet.info/ontology/omn-lifecycle#NotYetInitialized
:NotYetInitialized rdf:type :State ,
                   owl:NamedIndividual .
### http://open-multinet.info/ontology/omn-lifecycle#Pending
:Pending rdf:type :State ,
         owl:NamedIndividual .
### http://open-multinet.info/ontology/omn-lifecycle#Preinit
:Preinit rdf:type :State ,
         owl:NamedIndividual .
### http://open-multinet.info/ontology/omn-lifecycle#Provisioned
:Provisioned rdf:type :ReservationState ,
             owl:NamedIndividual .
### http://open-multinet.info/ontology/omn-lifecycle#Ready
:Ready rdf:type :State ,
       owl:NamedIndividual .
### http://open-multinet.info/ontology/omn-lifecycle#Removing
:Removing rdf:type :State ,
          owl:NamedIndividual .
### http://open-multinet.info/ontology/omn-lifecycle#Started
:Started rdf:type :State ,
         owl:NamedIndividual .
### http://open-multinet.info/ontology/omn-lifecycle#Stopped
:Stopped rdf:type :State ,
         owl:NamedIndividual .
### http://open-multinet.info/ontology/omn-lifecycle#Stopping
:Stopping rdf:type :State ,
          owl:NamedIndividual .
### http://open-multinet.info/ontology/omn-lifecycle#Unallocated
:Unallocated rdf:type :ReservationState ,
             owl:NamedIndividual .
```

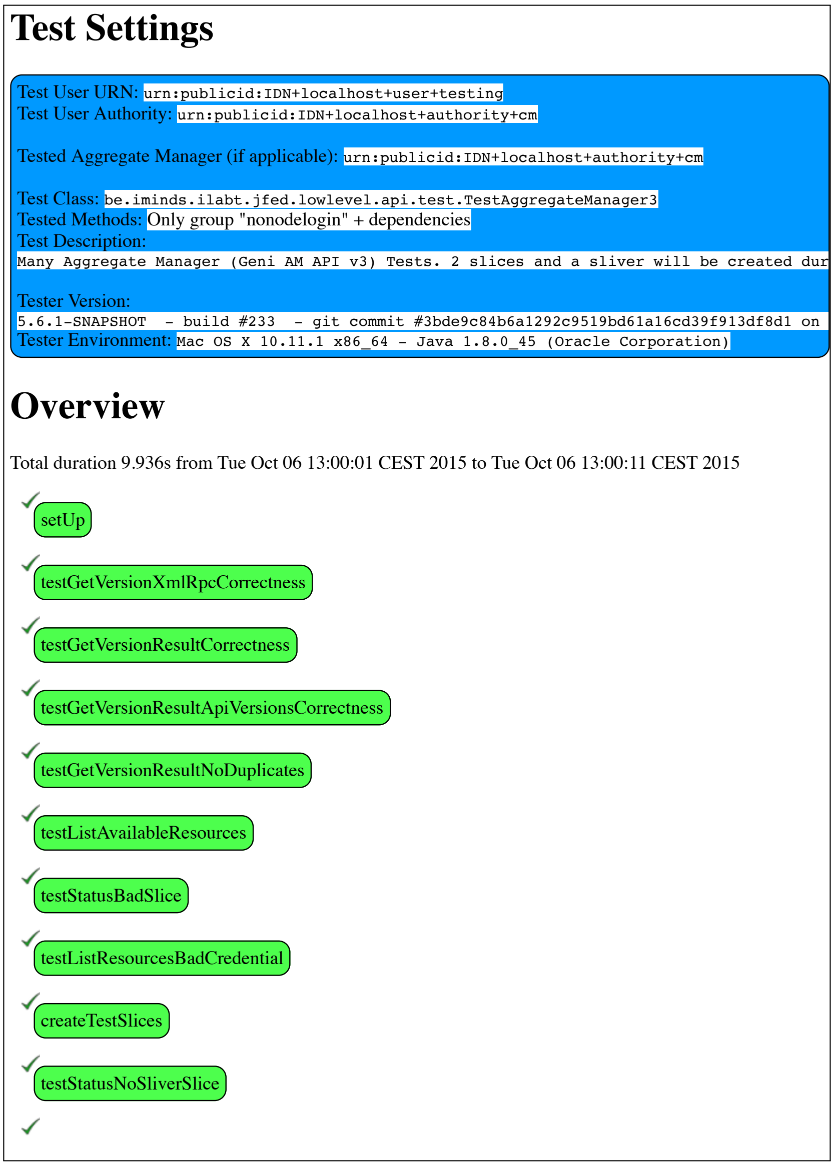
```
### http://open-multinet.info/ontology/omn-lifecycle#Uncompleted
:Uncompleted rdf:type :State ,
             owl:NamedIndividual .
### http://open-multinet.info/ontology/omn-lifecycle#Updating
:Updating rdf:type :State ,
          owl:NamedIndividual .
### Fixes for validation
owl:NamedIndividual rdf:type owl:Class .
owl:IrreflexiveProperty rdf:type owl:Class .
owl:AllDisjointClasses rdf:type owl:Class .
<http://alex.willner.ws/about#me> a foaf:Person .
### Generated by the OWL API (version 3.5.0) http://owlapi.
    sourceforge.net
```

APPENDIX B

TEST RESULTS

B.1 Conformance Results

This annex provides additional Figures B.1 to B.5 and Listings B.1 to B.3 related to the conformance tests described in Section 7.2.2. *Overview*



Test Settings

Test User URN: `urn:publicid:IDN+localhost+user+testing`
Test User Authority: `urn:publicid:IDN+localhost+authority+cm`

Tested Aggregate Manager (if applicable): `urn:publicid:IDN+localhost+authority+cm`

Test Class: `be.iminds.ilabt.jfed.lowlevel.api.test.TestAggregateManager3`
Tested Methods: Only group "nonodelogin" + dependencies
Test Description:
`Many Aggregate Manager (Geni AM API v3) Tests. 2 slices and a sliver will be created dur`

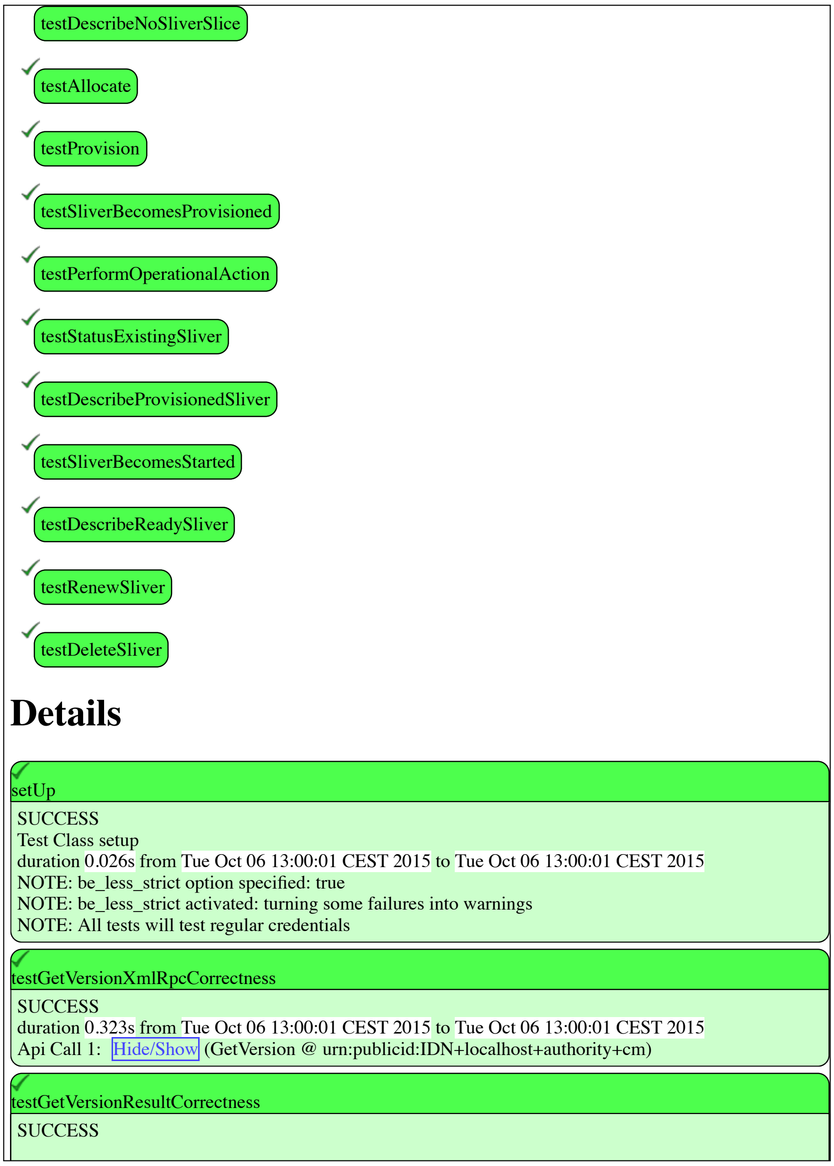
Tester Version:
`5.6.1-SNAPSHOT  - build #233  - git commit #3bde9c84b6a1292c9519bd61a16cd39f913df8d1 on`
Tester Environment: `Mac OS X 10.11.1 x86_64 - Java 1.8.0_45 (Oracle Corporation)`

Overview

Total duration 9.936s from Tue Oct 06 13:00:01 CEST 2015 to Tue Oct 06 13:00:11 CEST 2015

- ✓ setUp
- ✓ testGetVersionXmlRpcCorrectness
- ✓ testGetVersionResultCorrectness
- ✓ testGetVersionResultApiVersionsCorrectness
- ✓ testGetVersionResultNoDuplicates
- ✓ testListAvailableResources
- ✓ testStatusBadSlice
- ✓ testListResourcesBadCredential
- ✓ createTestSlices
- ✓ testStatusNoSliverSlice
- ✓

Figure B.1: jFed SFA AMv3 compliance test result (page 1)


testDescribeNoSliverSlice
✓ testAllocate
✓ testProvision
✓ testSliverBecomesProvisioned
✓ testPerformOperationalAction
✓ testStatusExistingSliver
✓ testDescribeProvisionedSliver
✓ testSliverBecomesStarted
✓ testDescribeReadySliver
✓ testRenewSliver
✓ testDeleteSliver

Details

✓ setUp

SUCCESS
Test Class setup
duration 0.026s from Tue Oct 06 13:00:01 CEST 2015 to Tue Oct 06 13:00:01 CEST 2015
NOTE: be_less_strict option specified: true
NOTE: be_less_strict activated: turning some failures into warnings
NOTE: All tests will test regular credentials

✓ testGetVersionXmlRpcCorrectness

SUCCESS
duration 0.323s from Tue Oct 06 13:00:01 CEST 2015 to Tue Oct 06 13:00:01 CEST 2015
Api Call 1: Hide/Show (GetVersion @ urn:publicid:IDN+localhost+authority+cm)

✓ testGetVersionResultCorrectness

SUCCESS

Figure B.2: jFed SFA AMv3 compliance test result (page 2)

from Tue Oct 06 13:00:01 CEST 2015 to Tue Oct 06 13:00:01 CEST 2015
NOTE: This test does not call GetVersion again, it uses the result received by "testGetVersionXmlRpcCorrectness"
NOTE: GetVersion does not specify "urn" (which contains the component manager urn). This is not mandatory, but it is useful to include.

✓ testGetVersionResultApiVersionsCorrectness

SUCCESS
from Tue Oct 06 13:00:01 CEST 2015 to Tue Oct 06 13:00:01 CEST 2015
NOTE: This test does not call GetVersion again, it uses the result received by "testGetVersionXmlRpcCorrectness"
NOTE: The URL of the server has localhost. This test will assume that this is a test server, and will therefor NOT warn or fail if the URL's in GetVersion have localhost in them.

✓ testGetVersionResultNoDuplicates

SUCCESS
duration 0.001s from Tue Oct 06 13:00:01 CEST 2015 to Tue Oct 06 13:00:01 CEST 2015
NOTE: This test does not call GetVersion again, it uses the result received by "testGetVersionXmlRpcCorrectness"

✓ testListAvailableResources

SUCCESS
duration 0.995s from Tue Oct 06 13:00:01 CEST 2015 to Tue Oct 06 13:00:02 CEST 2015
Api Call 1: Hide/Show (GetCredential @ urn:publicid:IDN+localhost+authority+cm)
Api Call 2: Hide/Show (ListResources @ urn:publicid:IDN+localhost+authority+cm)

✓ testStatusBadSlice

SUCCESS
This test calls Status with user credentials, and the urn of a non exisiting slice. That should fail.
duration 0.796s from Tue Oct 06 13:00:02 CEST 2015 to Tue Oct 06 13:00:03 CEST 2015
Api Call 1: Hide/Show (Status @ urn:publicid:IDN+localhost+authority+cm)

✓ testListResourcesBadCredential

SUCCESS
This test calls ListResources without any credentials. That should fail.
duration 0.076s from Tue Oct 06 13:00:03 CEST 2015 to Tue Oct 06 13:00:03 CEST 2015
Api Call 1: Hide/Show (ListResources @ urn:publicid:IDN+localhost+authority+cm)

✓ createTestSlices

SUCCESS
Create the slices used in the next tests
duration 3.35s from Tue Oct 06 13:00:03 CEST 2015 to Tue Oct 06 13:00:07 CEST 2015
Api Call 1: Hide/Show (Resolve @ urn:publicid:IDN+localhost+authority+cm)
Api Call 2: Hide/Show (Register @ urn:publicid:IDN+localhost+authority+cm)
Api Call 3: Hide/Show (Resolve @ urn:publicid:IDN+localhost+authority+cm)
Api Call 4: Hide/Show (Register @ urn:publicid:IDN+localhost+authority+cm)
NOTE: testCredentialType = REGULAR
NOTE: The longest CM urn top level authority is: "localhost" (9 chars)
NOTE: The user authority of max length is: "localhost" (9 chars)

Figure B.3: jFed SFA AMv3 compliance test result (page 3)

NOTE: The maximum node name length is: 12
NOTE: Slice name length is 19
DEBUG: RAW_INFO BEGIN_SLICE_NAME n3EnEgR6NyPmsrlJ2Kf END_SLICE_NAME
DEBUG: RAW_INFO BEGIN_SLICE_URN urn:publicid:IDN+localhost+slice+n3EnEgR6NyPmsrlJ2Kf END_SLICE_URN
NOTE: testCredentialType = REGULAR
NOTE: The longest CM urn top level authority is: "localhost" (9 chars)
NOTE: The user authority of max length is: "localhost" (9 chars)
NOTE: The maximum node name length is: 12
NOTE: Slice name length is 19
DEBUG: RAW_INFO BEGIN_SLICE_NAME sz3fxHzsgt5nQEwH5h1 END_SLICE_NAME
DEBUG: RAW_INFO BEGIN_SLICE_URN urn:publicid:IDN+localhost+slice+sz3fxHzsgt5nQEwH5h1 END_SLICE_URN

testStatusNoSliverSlice

SUCCESS
Call Status on a slice without slivers. It is expected an error such as 12 "Search Failed (eg for slice)" is returned.
duration 0.112s from Tue Oct 06 13:00:07 CEST 2015 to Tue Oct 06 13:00:07 CEST 2015
Api Call 1: Hide/Show (Status @ urn:publicid:IDN+localhost+authority+cm)
NOTE: AMV3 spec says: "Attempting to get Status() for a slice (no slivers identified) with no current slivers at this aggregate
may return an empty list for geni_slivers, may return a list of previous slivers that have since been deleted, or may even return an error (e.g. SEARCHFAILED or EXPIRED).
Note therefore that geni_slivers may be an empty list."

testDescribeNoSliverSlice

SUCCESS
duration 0.445s from Tue Oct 06 13:00:07 CEST 2015 to Tue Oct 06 13:00:07 CEST 2015
Api Call 1: Hide/Show (Describe @ urn:publicid:IDN+localhost+authority+cm)
Api Call 2: Hide/Show (Describe @ urn:publicid:IDN+localhost+authority+cm)

testAllocate

SUCCESS
duration 0.4s from Tue Oct 06 13:00:07 CEST 2015 to Tue Oct 06 13:00:08 CEST 2015
Api Call 1: Hide/Show (Allocate @ urn:publicid:IDN+localhost+authority+cm)

testProvision

SUCCESS
duration 1.303s from Tue Oct 06 13:00:08 CEST 2015 to Tue Oct 06 13:00:09 CEST 2015
Api Call 1: Hide/Show (Provision @ urn:publicid:IDN+localhost+authority+cm)
NOTE: Successfully found the following client_id's in both request and manifest: "demo-motor-1"
NOTE: Found no node service login in manifest RSpec
NOTE: Did not find node login info in Provision reply

testSliverBecomesProvisioned

SUCCESS
duration 0.159s from Tue Oct 06 13:00:09 CEST 2015 to Tue Oct 06 13:00:09 CEST 2015
Api Call 1: Hide/Show (Status @ urn:publicid:IDN+localhost+authority+cm)

Figure B.4: jFed SFA AMv3 compliance test result (page 4)

testPerformOperationalAction
SUCCESS
duration 0.248s from Tue Oct 06 13:00:09 CEST 2015 to Tue Oct 06 13:00:09 CEST 2015
Api Call 1: Hide/Show (PerformOperationalAction @ urn:publicid:IDN+localhost+authority+cm)
NOTE: ERROR: Invalid reply to PerformOperationalAction. The "value" of a successful call should be a Vector (API specifies: "list of struct"), but it was: class java.util.Hashtable.
Note: This ERROR has been converted to just a note because of the be_less_strict option

testStatusExistingSliver
SUCCESS
duration 0.148s from Tue Oct 06 13:00:09 CEST 2015 to Tue Oct 06 13:00:09 CEST 2015
Api Call 1: Hide/Show (Status @ urn:publicid:IDN+localhost+authority+cm)

testDescribeProvisionedSliver
SUCCESS
duration 0.269s from Tue Oct 06 13:00:09 CEST 2015 to Tue Oct 06 13:00:10 CEST 2015
Api Call 1: Hide/Show (Describe @ urn:publicid:IDN+localhost+authority+cm)
NOTE: Successfully found the following client_id's in both request and manifest: "demo-motor-1"
NOTE: Found no node service login in manifest RSpec
NOTE: Did not find node login info in Describe reply

testSliverBecomesStarted
SUCCESS
duration 0.116s from Tue Oct 06 13:00:10 CEST 2015 to Tue Oct 06 13:00:10 CEST 2015
Api Call 1: Hide/Show (Status @ urn:publicid:IDN+localhost+authority+cm)

testDescribeReadySliver
SUCCESS
duration 0.202s from Tue Oct 06 13:00:10 CEST 2015 to Tue Oct 06 13:00:10 CEST 2015
Api Call 1: Hide/Show (Describe @ urn:publicid:IDN+localhost+authority+cm)
NOTE: Successfully found the following client_id's in both request and manifest: "demo-motor-1"
NOTE: Found no node service login in manifest RSpec
NOTE: Did not find node login info in Describe reply

testRenewSliver
SUCCESS
duration 0.6s from Tue Oct 06 13:00:10 CEST 2015 to Tue Oct 06 13:00:11 CEST 2015
Api Call 1: Hide/Show (Status @ urn:publicid:IDN+localhost+authority+cm)
Api Call 2: Hide/Show (Renew @ urn:publicid:IDN+localhost+authority+cm)
Api Call 3: Hide/Show (Status @ urn:publicid:IDN+localhost+authority+cm)

testDeleteSliver
SUCCESS
duration 0.36s from Tue Oct 06 13:00:11 CEST 2015 to Tue Oct 06 13:00:11 CEST 2015
Api Call 1: Hide/Show (Delete @ urn:publicid:IDN+localhost+authority+cm)
Api Call 2: Hide/Show (Status @ urn:publicid:IDN+localhost+authority+cm)

Figure B.5: jFed SFA AMv3 compliance test result (page 5)

Listing B.1: jFed AM conformance tests executed under 15 seconds

```
$ time testAM3rspec.sh
...
Read context properties from file "conf/cli.properties":
   Tested Authority:localhost
      URN (connect):urn:publicid:IDN+localhost+authority+cm
      URN (rspec):urn:publicid:IDN+localhost+authority+cm
      Hrn:localhost
      Server certificates:[...]
```

```
      Allowed server certificate hostname aliases:[localhost]
      URL for ServerType{"PROTOGENI_SA" "1"}: https://localhost:8443/sfa/api/sa/v1
      URL for ServerType{"AM" "3"}: https://localhost:8443/sfa/api/am/v3
   User:urn:publicid:IDN+localhost+user+testing
      Authority URN:urn:publicid:IDN+localhost+authority+cm
Running testGetVersionXmlRpcCorrectness...SUCCESS
Running testGetVersionResultCorrectness...SUCCESS
Running testGetVersionResultApiVersionsCorrectness...SUCCESS
Running testGetVersionResultNoDuplicates...SUCCESS
Running testListAvailableResources...SUCCESS
Running testStatusBadSlice...SUCCESS
Running testListResourcesBadCredential...SUCCESS
Running createTestSlices...SUCCESS
Running testStatusNoSliverSlice...SUCCESS
Running testDescribeNoSliverSlice...SUCCESS
Running testAllocate...SUCCESS
Running testProvision...SUCCESS
Running testSliverBecomesProvisioned...SUCCESS
Running testPerformOperationalAction...SUCCESS
Running testStatusExistingSliver...SUCCESS
Running testDescribeProvisionedSliver...SUCCESS
Running testSliverBecomesStarted...SUCCESS
Running testDescribeReadySliver...SUCCESS
Running testRenewSliver...SUCCESS
Running testDeleteSliver...SUCCESS
...
real 0m8.415s
user 0m13.003s
sys 0m0.546s
```

Listing B.2: jFed experimenter GUI provisioning a motor network

```
Showing version because debug is enabled:
jFed Experimenter GUI -- http://jfed.iminds.be/
Send bugreports to: jfed-bugreports@atlantis.ugent.be
Version: 5.6.1-SNAPSHOT - build #233 - git commit #3 bde9c84b6a1292c9519bd61a16cd39f913df8d1 on origin/develop
Environment: Mac OS X 10.11.2 x86_64 - Java 1.8.0_45 (Oracle Corporation)
ARG: c context-file conf/cli.properties [conf/cli.properties] 1
ARG: null authorities-file conf/cli.authorities [conf/cli.authorities] 1
ARG: r rspec conf/motor-network.rspec [conf/motor-network.rspec] 1
ARG: s slice urn:publicid:IDN+localhost+slice+1446302793 [urn:publicid:IDN+localhost+slice+1446302793] 1
ARG: null create-slice null [] 0
ARG: d debug null [] 0
Note: This RSpec has 1 component managers: [urn:publicid:IDN+localhost+authority+cm]
User urn:publicid:IDN+localhost+user+testing will be used.
```

```
User Authority: localhost
User Authority URN: urn:publicid:IDN+localhost+authority+cm
1 user credentials retrieved.
User urn:publicid:IDN+localhost+user+testing has the following slices: []
slice urn:publicid:IDN+localhost+slice+1446302793 does not yet exist
Adding userspec from user certificate.
Adding 0 userspecs from RSpec
The following user specification will be added to all create sliver calls:
  User urn:publicid:IDN+localhost+user+testing with 1 keys
     ssh-rsa AAAAB3NzaC1yc2...0b7m01z
Contacting urn:publicid:IDN+localhost+authority+cm...
Sliver at urn:publicid:IDN+localhost+authority+cm is created and initializing...
Writing combined manifest to file "manifest-1446302793.rspec"
Will now wait until the sliver is ready...
Waiting 5 seconds before checking status...
Contacting urn:publicid:IDN+localhost+authority+cm to check status...
Status of sliver at urn:publicid:IDN+localhost+authority+cm is READY
The sliver is ready.
All done. Exiting.
```

Listing B.3: jFed RDF-based AM conformance tests executed in 10 seconds

```
$ time testAM3rdf.sh
...
Read context properties from file "conf/cli.rdfxml.properties":
   Tested Authority:localhost
      URN (connect):urn:publicid:IDN+localhost+authority+cm
      URN (rspec):urn:publicid:IDN+localhost+authority+cm
      Hrn:localhost
      Server certificates:[...]
      Allowed server certificate hostname aliases:[localhost]
      URL for ServerType{"PROTOGENI_SA" "1"}: https://localhost:8443/sfa/api/sa/v1
      URL for ServerType{"AM" "3"}: https://localhost:8443/sfa/api/am/v3
   User:urn:publicid:IDN+localhost+user+testing
      Authority URN:urn:publicid:IDN+localhost+authority+cm
Running testGetVersionXmlRpcCorrectness...SUCCESS
Running testGetVersionResultCorrectness...SUCCESS
Running testGetVersionResultApiVersionsCorrectness...SUCCESS
Running testGetVersionResultNoDuplicates...SUCCESS
Running testListAvailableResources...SUCCESS
Running testStatusBadSlice...SUCCESS
Running testListResourcesBadCredential...SUCCESS
Running createTestSlices...SUCCESS
Running testStatusNoSliverSlice...SUCCESS
Running testDescribeNoSliverSlice...SUCCESS
```

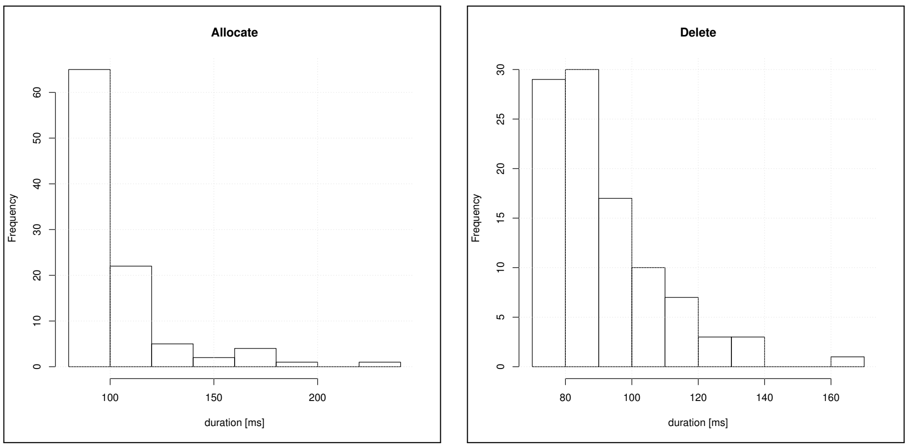
```
Running testAllocate...SUCCESS
Running testProvision...WARN
Running testSliverBecomesProvisioned...SUCCESS
Running testPerformOperationalAction...SUCCESS
Running testStatusExistingSliver...SUCCESS
Running testDescribeProvisionedSliver...WARN
Running testSliverBecomesStarted...SUCCESS
Running testDescribeReadySliver...WARN
Running testRenewSliver...SUCCESS
Running testDeleteSliver...SUCCESS
...
real 0m7.731s
user 0m10.719s
sys 0m0.466s
```

B.2 Performance Results

B.2.1 Histograms

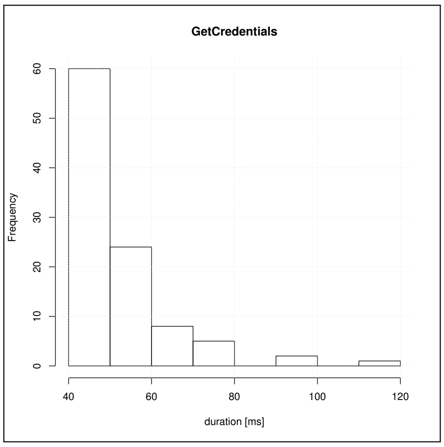
Overview

This subsection provides additional histogram data in Figures B.6 to B.9 for the performance evaluation described in Section 7.3.

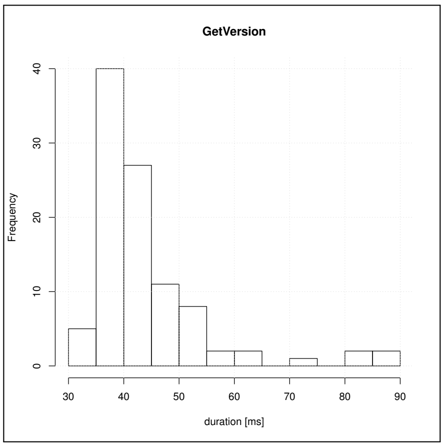


(a) Allocate (b) Delete

Figure B.6: Histogram of the Allocate and Delete method calls

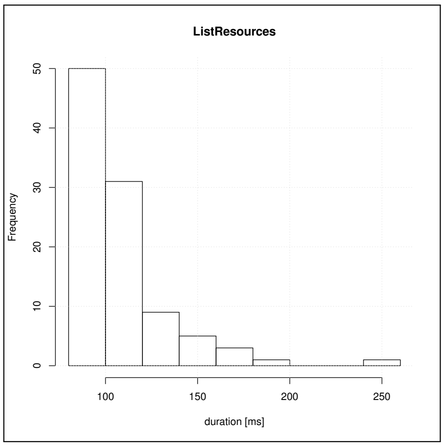


(a) GetCredentials

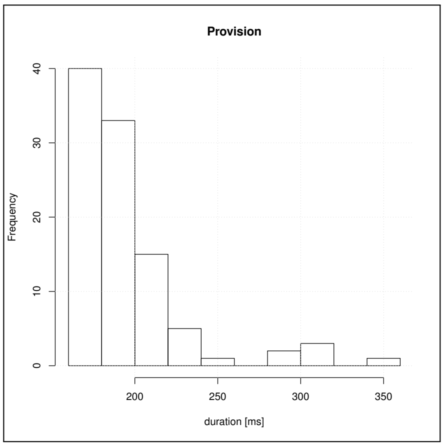


(b) GetVersion

Figure B.7: Histogram of the GetCredential and GetVersion method calls

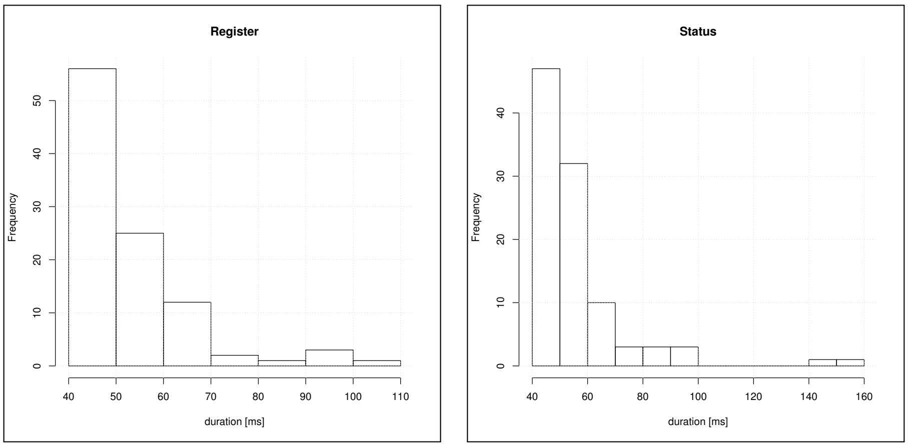


(a) ListResources



(b) Provision

Figure B.8: Histogram of the ListResources and Provision method calls

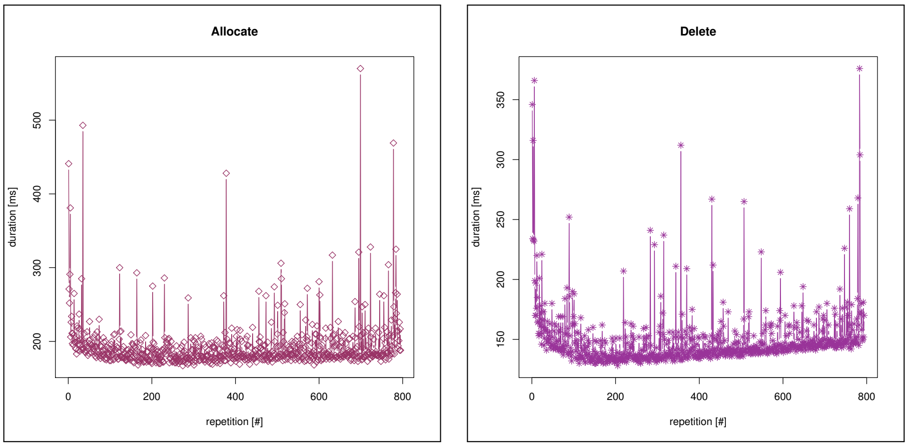


(a) Register (b) Status

Figure B.9: Histogram of the Register and Status method calls

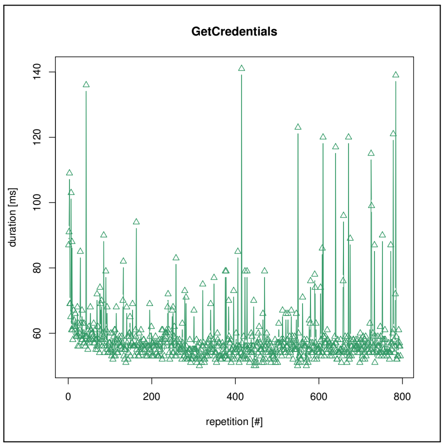
B.2.2 Lineplots

This subsection provides additional lineplots in Figures B.10 to B.13 for the performance evaluation described in Section 7.3. *Overview*

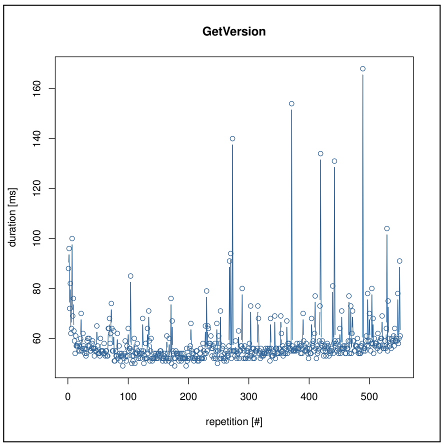


(a) Allocate (b) Delete

Figure B.10: Lineplot of the Allocate and Delete method calls

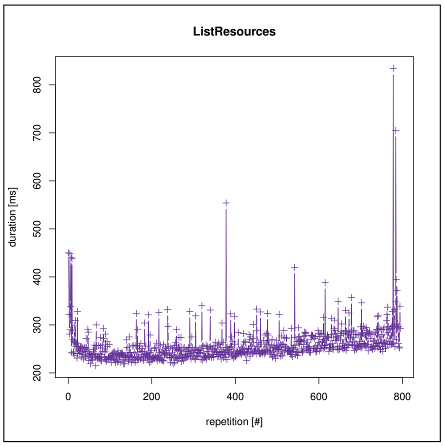


(a) GetCredentials

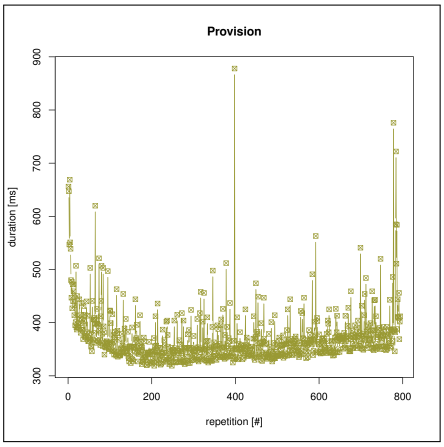


(b) GetVersion

Figure B.11: Lineplot of the GetCredential and GetVersion method calls

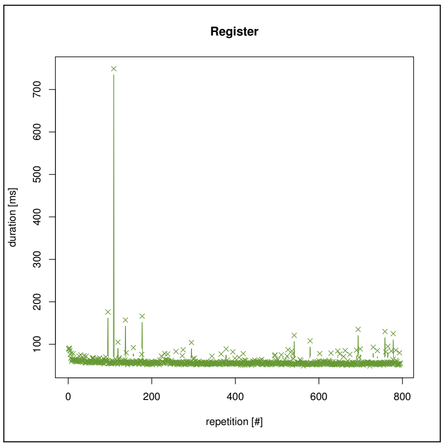


(a) ListResources

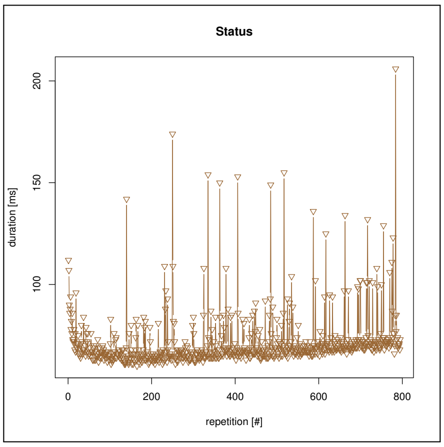


(b) Provision

Figure B.12: Lineplot of the ListResources and Provision method calls



(a) Register



(b) Status

Figure B.13: Lineplot of the Register and Status method calls

Acronyms

5G	5th Generation Mobile Network	95
A-Box	Assertion-Box	48
ABAC	Attributed Based Access Control	31
AC4	Activity Chain 4 – Service Openness and Interoperability Issues/Semantic Interoperability	53
ACF	Autonomic Communications Forum	47
ACS	Automatic Configuration Server	94
AJO	Abstract Job Object	11
AM	Aggregate Manager	24
AMQP	Advanced Message Queuing Protocol	3
AN	Access Network	94
AP	Access Point	94
APAN	Asia-Pacific Advanced Network	14
API	Application Programming Interface	4
AsiaFI	Asia Future Internet Forum	14
ASN.1	Abstract Syntax Notation One	45
ATM	Asynchronous Transfer Mode	11
AuthN	Authentication	3
AuthZ	Authorization	3
AWS	Amazon Web Services	95
AWT	Accessible Wayfinding Testbed	55
BoD	Bandwidth on Demand	143
BonFIRE	Building Service Testbeds on FIRE	28
CA	Certificate Authority	89
CAMP	Cloud Application Management for Platforms	13
CBDF	Cross Border Dark Fiber	146
CC	Creative Commons	134
CDMI	Cloud Data Management Interface	13
CH	Clearinghouse	56
CI	Continous Integration	79
CI-FIRE	Coordination and Integration of FIRE Activities in Europe	19
CI	Confidence Interval	111
CIM	Common Information Model	47
CIMI	Cloud Infrastructure Management Interface	12

CLI	Command Line Interface	81
CNGI	China Next Generation Internet	15
CNU	Chungnam National University	15
CORBA	Common Object Request Broker Architecture	69
CORE	Collaborative Overlay Research Environment	14
CORONET	Core Optical Networks: Architecture, Protocols, Control and Management	146
CPU	Central Processing Unit	93
CQELS	Continuous Query Evaluation over Linked Stream	52
CRM	Customer Relationship Management	72
CSA	Coordination and Support Action	19
CSV	Comma Separated Values	28
CWA	closed world assumption	48
DAML-S	DARPA Agent Markup Language for Services	53
DARPA	Defense Advanced Research Projects Agency	146
DC	Dublin Core	134
DCAT	Data Catalog Vocabulary	51
DCI	Data, Context and Interaction	66
DCN	Dynamic Circuit Network	74
DEN-ng	Directory Enabled Networks New Generation	18
DFA	DETER Federation Architecture	16
DIP	Dependency Inversion Principle	68
DISCO	DIstributed SDN COntrol plane	148
DL	Description Logics	50
DM	Data Model	44
DMTF	Distributed Management Task Force	12
DNS	Domain Name System	93
DoE	Department of Energy	16
DOM	Document Object Model	45
DoT	Department of Transportation	16
DRM	Distributed Resource Management	10
DRMAA	Distributed Resource Management Application API	11
DSL	Domain Specific Language	17
E2E	End-to-End	148
EaaS	Experimentation as a Service	145
EBI	Entity, Boundary, Interactor	66
EC	Experiment Controller	17
EDUCOM	Interuniversity Communications Council	1
EGI	European Grid Infrastructure	13
EIT	European Institute of Innovation and Technology	14
EIT ICT Labs	EIT Information and Communication Technology Labs	21
EJB	Enterprise Java Bean	117
EMS	Element Management System	75
EPC	Evolved Packet Core	60
EPoSS	ETP on Smart Systems Integration	17
ESB	Enterprise Service Bus	67
ESWC	Extended Semantic Web Conference	143
eTOM	Enhanced Telecom Operations Map	21
ETP	European Technology Platform	17
ETRI	Electronics and Telecommunications Research Institute	15

ETSI	European Telecommunications Standards Institute	14
EU	European Union	17
EC	European Commision	20
FanTaaStic	Fanning out Testbeds-as-a-Service for the EIT ICT	21
FCI	Federation Computing Interface	18
Fed4FIRE	Federation for FIRE	18
FedSM	Federated IT Service Management	20
FI	Future Internet	2
FI-PPP	Future Internet Public Private Partnership	14
FIA	Future Internet Assembly	17
FIBRE	Future Internet Testbeds Experimentation Between Brazil and Europe	16
FIDDLE	Federated Infrastructure Description and Discovery Language	6
FIE	Future Internet Engineering	131
FIF	Future Internet Forum	15
FIND	Future Internet Design	14
FIRE	Future Internet Research and Experimentation	2
FIRMA	Federated Infrastructure Resource Management Architecture	6
FLS	First Level Support	28
FN2020	Future Network 2020	15
FOAF	Friend of a Friend	51
FOKUS	Fraunhofer Institute for Open Communication Systems	i
FOL	First-Order Predicate Logic	50
FORGE	Forging Online Education through FIRE	149
FRCP	Federated Resource Control Protocol	3
G-Lab	German Lab	15
G.805	ITU-T Generic Functional Architecture of Transport Networks	47
GB	Gigabyte	147
GCTC	Global City Teams Challenge	16
GDMO	Guidelines for the Definition of Managed Objects	45
GE	Generic Enabler	20
GEi	Generic Enabler implementation	20
GENI	Global Environment for Network Innovations	2
GICTF	Global Inter-Cloud Technology Forum	13
GIRRA	GLIF Interdomain Resource Reservation Architecture	146
GLIF	Global Lambda Integrated Facility	54
GLUE	Grid Laboratory for a Uniform Environment	52
GMPLS	Generalized MPLS	47
GoE	Graph of Education	149
GoT	Graph of Things	147
GPO	GENI Project Office	16
GR	Good Relations	134
GRAM	Globus Resource Allocation Manager	11
GSI	Grid Security Infrastructure	11
GT	Globus Toolkit	11
GUI	Graphical User Interface	68
HHS	Department of Health and Human Services	16

Acronyms

HOT	Heat Orchestration Template	95
HPC	High Performance Computing	15
HTML	Hyper Text Markup Language	70
HTTP	Hyper Text Transfer Protocol	51
I-WAY	Information Wide Area Year	1
IaaS	Infrastructure as a Service	12
IAB	Internet Architecture Board	147
ICT	Information and Communication Technology	5
IDC	International Data Corporation	147
IdM	Identity Management	72
IdP	Identity Provider	30
IEC	International Electrotechnical Commission	13
IEEE	Institute of Electrical and Electronics Engineers	6
IERC	European Research Cluster on the Internet of Things	53
IETF	Internet Engineering Task Force	14
iKaaS	Intelligent Knowledge as a Service	15
IM	Information Model	43
IMS	IP Multimedia Subsystem	71
INDL	Infrastructure and Network Description Language	54
INFINITY	INfrastructures for the Future INternet CommunITY	119
IoE	Internet of Everything	147
IoT	Internet of Things	5
IP	Internet Protocol	18
ISI	Integral SatCom Initiative	17
ISO	International Organization for Standardization	13
ISOD	Infrastructure On-Demand	12
ISP	Internet Service Provider	132
IT	Information Technology	12
ITA	International Trade Administration	16
ITIL	IT Infrastructure Library	72
ITS	Intelligent Transport Systems	51
ITSM	IT Service Management	20
ITU-T	International Telecommunication Union – Telecommunication Standardization Sector	13
J2EE	Java Platform Enterprise Edition	79
JAX-RS	Java API for RESTful Web Services	90
JAXB	Java Architecture for XML Binding	45
JDL	Job Definition Language	11
JGN2Plus	Japan Gigabit Network 2+	14
JMS	Java Message Service	79
JSDL	Job Submission Description Language	12
JSON	JavaScript Object Notation	4
JSON-LD	JSON for Linked Data	49
JSONP	JSON with Padding	90
JSR	Java Specification Request	79
JVM	Java Virtual Machine	79
KIC	Knowledge and Innovation Community	21
KISA	Korea Internet and Security Agency	15
KISTI	Korea Institute of Science and Technology Information	15
KNU	Kyungpook National University	15

KOREN	Korea Advanced Reseach Network	15
KREONET	Korea Research Environment Open NETwork	15
LCA	Life Cycle Agreement	32
LDAP	Lightweight Directory Access Protocol	30
LOD	Linked Open Data	51
LODE	Live OWL Documentation Environment	134
LOS	Linked Open Services	53
LOV	Linked Open Vocabulary	135
LRMI	Learning Resource Metadata Initiative	149
LSO	Lifecycle Service Orchestration	96
M2M	Machine-To-Machine Communication	53
MAMSplus	Multi-Access Modular Services Framework	21
MANO	Management and Orchestration	74
MAS	OneM2M Working Group 5 Management, Abstraction and Semantics	53
MDB	Message Driven Bean	117
MDS	Monitoring and Discovery System	11
MEC	Mobile Edge Computing	148
MEF	Metro Ethernet Forum	96
MEST	Ministry of Education, Science and Technology	15
MIB	Management Information Base	45
MOFI	Mobile Oriented Future Internet	15
MOM	Message Oriented Middleware	79
MOOC	Massive Open Online Course	149
mOSAIC	Open-Source API and Platform for Multiple Clouds	13
MPI	Message Passing Interface	11
MPICH	MPI Chameleon	11
MVP	Model-View-Presenter	67
N3	Notation 3	45
NDL	Network Description Language	54
NDL-OWL	Network Description Language based on the Web Ontology Language	54
NEPI	Network Experimentation Programming Interface	17
NETCONF	Network Configuration Protocol	45
NF	Network Function	6
NFV	Network Function Virtualization	95
NFVI	Network Function Virtualization Infrastructure	75
NFVO	Network Function Virtualization Orchestrator	95
NGMN	Next Generation Mobile Network	130
NGOSS	New Generation Operations Systems and Software	47
NIA	National Information Agency	15
NICT	National Institute of Information and Communications Technology	14
NICTA	National Information and Communication Technology Australia	15
NIST	National Institute of Standards and Technology	13
NITOS	Network Implementation Testbed using Open Source code	26
NJS	Network Job Supervisor	11
NML	Network Mark-Up Language	54

NMS	Network Management System	75
nmVO	Network Measurement Virtual Observatory	28
NOVI	Networking innovations Over Virtualized Infrastructures	54
NREN	National Research and Education Network	16
NS	Network Service	95
NSF	National Science Foundation	16
NSI	Network Service Interface	54
OASIS	Organization for the Advancement of Structured Information Standards	6
OCCI	Open Cloud Computing Interface	12
OCX	Open Cloud Exchange	147
ODCA	Open Data Center Alliance	13
OEDL	OMF Experiment Description Language	17
OGF	Open Grid Forum	12
OGP	Open Graph Protocol	52
OGSA	Open Grid Service Architecture	12
OGSI	Open Grid Services Infrastructure	12
OIF	Optical Internetworking Forum	147
OKFN	Open Knowledge Foundation	135
OLA	Operational Level Agreement	32
OM	Object Model	44
OMA	Open Mobile Alliance	18
OMF	cOntrol and Management Framework	3
OMF-F	OMF-Federated	17
OMG	Object Management Group	47
OML	ORBIT Measurement Library	3
OMN	Open-Multinet	6
OMSP	OML Measurement Stream Protocol	3
ONF	Open Networking Foundation	96
OOPS	OntOlogy Pitfall Scanner	134
OpenEPC	Open Evolved Packet Core	94
OpenMTC	Open Machine Type Communication	94
ORA	Ontologies for Robotics and Automation	53
ORBIT	Open Access Research Testbed for Next-Generation Wireless Networks	15
ORCA	Open Resource Control Architecture	16
OSGi	Open Service Gateway Initiative	79
OVF	Open Virtualization Format	12
OWA	open-world assumption	48
OWL	Web Ontology Language	50
OWL-S	Semantic Markup for Web Services	53
P1600.1	Standard Upper Ontology	56
P2302	Standard for Intercloud Interoperability and Federation	6
PaaS	Platform as a Service	12
Panlab	Pan-European Laboratory	17
PDF	Portable Document Format	135
PDP	Policy Decision Point	70
PE	Policy Engine	18
PIB	Policy Information Base	45
PKI	Public Key Infrastructure	12

PlanetLab CJK	PlanetLab China, Japan, Korea	14
PLC	PlanetLab Central	16
PLE	PlanetLab Europe	18
PNF	Physical Network Function	94
PoC	Proof-of-Concept	130
PTM	Panlab Testbed Manager	18
QoE	Quality of Experience	32
QoS	Quality of Service	12
RA	Resource Adapter	18
RADL	Resource Adapter Description Language	18
RAM	Random Access Memory	119
RAN	Radio Access Network	148
RAS	Robotics and Automation Society	53
RC	Resource Controller	17
RDF	Resource Description Framework	45
RDFa	RDF in Attributes	49
RDFS	Resource Description Framework Schema	50
REST	Representational State Transfer	6
RFC	Request for Comments	1
R&D	Research & Development	20
R&E	Research & Education	15
RoI	Return of Investment	40
RPC	Remote Procedure Call	3
RPM	Revolutions per Minute	106
RSL	Resource Specification Language	11
RSpec	Resource Specification	4
S-OGSA	Semantic Open Grid Service Architecture	52
SA	Slice Authority	87
SaaS	Software as a Service	12
SAML	Security Assertion Markup Language	12
SAVI	Smart Applications on Virtual Infrastructures	15
SAWSDL	Semantic Annotations for WSDL and XML Schema	53
SAX	Simple API for XML	44
SC	Smart City	125
SCORM	Sharable Content Object Reference Model	149
SCP	Smart City Platform	125
SD-WAN	Software Defined Wide Area Network	74
SDI	Software Defined Infrastructure	96
SDN	Software Defined Networking	5
SDO	Standards Developing Organization	13
SFA	Slice-based Federation Architecture	3
SFC	Service Function Chaining	27
SFI	SFA Command-Line Interface	16
SID	Shared Information & Data Model	47
SIIF	Standard for Intercloud Interoperability and Federation	13
SLA	Service Level Agreement	12
SLAng	Service Level Agreement Language	32
SM	Semantic Model	44
SME	Small and Medium Enterprise	20
SMI	Structure of Management Information	45

VNFD	Virtualised Network Function Descriptor	95
VNFM	Virtual Network Function Manager	95
VOAF	Vocabulary of a Friend	134
VPN	Virtual Private Network	24
W3C	World Wide Web Consortium	6
WAN	Wide Area Network	74
WDC	Web Data Commons	147
WMS	Workload Management Service	11
WoT	Web of Things	147
WS-BPEL	Web Service Business Process Execution Language	53
WSAG	WS-Agreement	12
WSDL	Web Services Description Language	52
WSDL-S	Web Service Definition Language Semantics	53
WSLA	Web Service Level Agreement	32
WSML	Web Service Modeling Language	53
WSN	Web Services Notification	12
WSRF	Web Services Resource Framework	12
X-GSN	Extended Global Sensor Network	53
XACML	Extensible Access Control Markup Language	12
XIFI	Experimental Infrastructures for the Future Internet	20
XML	Extensible Markup Language	3
XMP	Extensible Metadata Platform	135
XMPP	Extensible Messaging and Presence Protocol	3
XPath	XML Path Language	46
XQuery	XML Query Language	51
XSD	XML Schema Definition	17
YAML	YAML Ain't Markup Language	45
YANG	Yet Another Next Generation	45
ZB	Zettabyte	147
ZOOM	Zero-touch Orchestration, Operations and Management	148

Bibliography

List of Author's Publications Covered in this Thesis

[a1] P. Brzozwski, S. Campbell, T. Coulouarn, Y. Demchenko, F. Dijkstra, M. Giertych, J. A. Garcıa Espin, E. Grasa, C. Guok, J. van der Ham, R. Krzywania, T. Kudoh, M. Lemay, A. Takefusa, A. Willner, and Y. Xin. *On-Demand Infrastructure Services Provisioning Best Practices*. Tech. rep. Open Grid Forum (OGF), Oct. 2012 (cit. on pp. 12, 143).

[a2] S. Figuerola, J. A. Garcia, A. Sanchez, C. de Waal, and A. Willner. "The network service plane: An approach for inter-domain network reservations". In: *10th Anniversary International Conference on Transparent Optical Networks (ICTON)*. Vol. 1. Athens, Greece: IEEE, June 2008, pp. 13–15. ISBN: 978-1-4244-2625-6. DOI: 10.1109/ICTON.2008.4598358 (cit. on pp. 74, 143).

[a3] S. Figuerola, J. A. Garcia-Espin, J. Ferrer, and A. Willner. "Performance analysis of harmony: An optical, multi-domain network resource broker". In: *11th International Conference on Transparent Optical Networks (ICTON)*. Azores: IEEE, June 2009, pp. 1–5. ISBN: 978-1-4244-4825-8. DOI: 10.1109/ICTON.2009.5185032 (cit. on p. 143).

[a4] S. Figuerola, J. A. Garcia-Espin, J. Ferrer, and A. Willner. *Scalability Analysis and Evaluation of the Multi-domain, Optical Network Service Plane in Harmony*. Poster at the 35th European Conference and Exhibition on Optical Communication (ECOC). Vienna, Sept. 2009 (cit. on p. 143).

[a5] S. Figuerola, E. Grasa, J. A. Garcıa-Espın, J. F. Riera, V. Reijs, E. Kenny, M. Lemay, M. Savoie, S. Campbell, M. Ruffini, D. O'Mahony, A. Willner, and B. S. Arnaud. "Bringing Optical Network Control to the User Facilities: Evolution of the User-Controlled LightPath Provisioning Paradigm". In: *Cross-Layer Design in Optical Networks*. Springer, 2013, pp. 291–323. DOI: 10.1007/978-1-4614-5671-1_14 (cit. on p. 143).

[a6] Y. Al-Hazmi, A. Willner, O. O. Ozpehlivan, D. Nehls, S. Covaci, and T. Magedanz. "An automated health monitoring solution for future Internet infrastructure marketplaces". In: *26th International Teletraffic Congress (ITC)*. Karlskrona, Sweden: IEEE, Sept. 2014, pp. 1–6. ISBN: 978-0-9883045-0-5. DOI: 10.1109/ITC.2014.6932979 (cit. on p. 143).

[a7] Y. Al-hazmi, A. Willner, B. Pickering, A. Alloush, T. Magedanz, and S. Cretti. “Creating a Sustainable Federation of Cloud-Based Infrastructures for the Future Internet - The FIWARE Approach”. In: *10th EAI International Conference on Testbeds and Research Infrastructures for the Development of Networks & Communities (TRIDENTCOM)*. Vancouver, Canada: IEEE, June 2015, pp. 1–10. DOI: 10.4108/icst.tridentcom.2015.259747 (cit. on pp. 22, 143).

[a8] R. Krzywania, J. A. Garcia-Espin, C. Guok, J. Van Der Ham, T. Kudoh, J. MacAuley, J. Mambretti, I. Monga, G. Roberts, J. Sobieski, and A. Willner. “Network Service Interface–gateway for future network services”. In: *Terena Networking Conference (TNC)*. Prague, Czech Republic: Open Access TCN, 2012, pp. 1–15. ISBN: 978-90-77559-00-0 (cit. on pp. 54, 143).

[a9] T. Magedanz, R. Steinke, A. Weber, and A. Willner. “Das Internet der Dinge und Maschine-zu- Maschine-Kommunikation als Rückgrat für Smart Citys”. In: *OBJEKTspektrum* 6.Online Themenspecal: IT-Trends - BigData/Hadoop und Internet der Dinge (2014), pp. 13–17. ISSN: 0945-0491 (cit. on p. 147).

[a10] B. D. Martino, G. Cretella, A. Esposito, A. Willner, A. Alloush, D. Bernstein, D. Vij, and J. Weinman. “Towards an Ontology-Based Intercloud Resource Catalogue – The IEEE P2302 Intercloud Approach for a Semantic Resource Exchange”. In: *International Conference on Cloud Engineering*. Tempe, Arizona: IEEE, Mar. 2015, pp. 458–464. ISBN: 978-1-4799-8218-9. DOI: 10.1109/IC2E.2015.76 (cit. on pp. 6, 78, 91, 133, 143, 146).

[a11] M. Morsey, Y. Al-Hazmi, M. Giatili, C. Papagianni, P. Grosso, I. Baldine, and A. Willner. “Open-Multinet: Supporting Resource Management in Federated Infrastructures with Semantic Web Ontologies (to be accepted)”. In: *13th European Semantic Web Conference (ESWC)*. Crete: Springer, Apr. 2016, pp. 1–15 (cit. on pp. xv, 44, 98, 112, 115, 116, 143).

[a12] J. Mwangama, J. Orimolade, N. Ventura, A. Elmangoush, R. Steinke, A. Willner, A. Corici, and T. Magedanz. “Prototyping Machine-to-Machine Applications for Emerging Smart Cities in Developing Countries”. In: *Southern Africa Telecommunication Networks and Applications Conference (SATNAC)*. Stellenbosch, South Africa: Roy Volkwyn (Telkom), 2014, pp. 383–388. ISBN: 9780620619653 (cit. on pp. 15, 143).

[a13] J. Mwangama, N. Ventura, A. Willner, Y. Al-Hazmi, G. Carella, and T. Magedanz. “Towards Mobile Federated Network Operators”. In: *1st Conference on Network Softwarization (NetSoft)*. London, United Kingdom: IEEE, Apr. 2015, pp. 1–6. ISBN: 978-1-4799-7899-1. DOI: 10.1109/NETSOFT.2015.7116187 (cit. on pp. 143, 148).

[a14] J. Mwangama, A. Willner, N. Ventura, A. Elmangosh, T. Pfeifer, and T. Magedanz. “Testbeds for Reliable Smart City Machine-to-Machine Communication”. In: *South African Telecommunication Networks and Applications Conference (SATNAC)*. Stellenbosch, South Africa: Open Access, 2013, pp. 339–344. ISBN: 978-0-620-57883-7 (cit. on pp. 15, 143).

[a15] M. Pilz, C. Barz, U. Bornhauser, P. Martini, C. de Waal, and A. Willner. “ARGON: Reservation in Grid-enabled Networks”. In: *1. DFN-Forum on Communication Technologies*. 130. Kaiserslautern, Germany: Gesellschaft für Informatik (GI), May 2008, pp. 75–84 (cit. on pp. 26, 74, 143).

[a16] W. Vandenberghe, B. Vermeulen, P. Demeester, A. Willner, S. Papavassiliou, A. Gavras, A. Quereilhac, Y. Al-Hazmi, F. Lobillo, C. Velayos, A. Vico-oton, and G. Androulidakis. "Architecture for the Heterogeneous Federation of Future Internet Experimentation Facilities". In: *Future Network and Mobile Summit (FNMS)*. Lisboa, Portugal: IEEE, 2013, pp. 1–11 (cit. on pp. 2–4, 18–20, 36, 143).

[a17] D. Vij, D. Bernstein, M. Morsey, P. Grosso, T. Magedanz, and A. Willner. "Software defined Bearer Intercloud networks semantic-based network exchange for the IEEE P2302 Intercloud approach". In: *12th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*. Hua Hin, Thailand: IEEE, June 2015, pp. 1–6. ISBN: 978-1-4799-7961-5. DOI: 10.1109/ECTICon.2015.7207136 (cit. on pp. 143, 147).

[a18] T. Wauters, B. Vermeulen, W. Vandenberghe, P. Demeester, S. Taylor, L. Baron, M. Smirnov, Y. Al-Hazmi, A. Willner, and M. Sawyer. "Federation of Internet experimentation facilities: architecture and implementation". In: *European Conference on Networks and Communications (EuCNC)*. Bologna: Ghent University Academic Bibliography, 2014, pp. 1–5. ISBN: 9781479952809 (cit. on pp. 36, 143).

[a19] A. Willner, C. Barz, J. A. Garcia Espin, J. Ferrer Riera, S. Figuerola, and P. Martini. "Harmony - Advance Reservations in Heterogeneous Multi-domain Environments". In: *Lecture Notes in Computer Science (LNCS)*. Vol. 5550. Aachen: Springer, May 2009, pp. 871–882. ISBN: 9783642013980. DOI: 10.1007/978-3-642-01399-7_68 (cit. on pp. 74, 143, 146).

[a20] A. Willner, R. Loughnane, and T. Magedanz. "FIDDLE: Federated Infrastructure Discovery and Description Language". In: *International Conference on Cloud Engineering*. Ed. by D. Bernstein, Y. Demchenko, and C. Rong. Tempe, Arizona: IEEE, Mar. 2015, pp. 465–471. ISBN: 978-1-4799-8218-9. DOI: 10.1109/IC2E.2015.77 (cit. on pp. 6, 44, 56–60, 143).

[a21] A. Willner and T. Magedanz. "FIRMA: A Future Internet Resource Management Architecture". In: *Workshop on Federated Future Internet and Distributed Cloud Testbeds (FIDC)*. Karlskrona, Sweden: IEEE Xplore Digital Library, Sept. 2014, pp. 1–4. DOI: 10.1109/ITC.2014.6932981 (cit. on pp. 6, 65, 66, 69, 118, 143).

[a22] A. Willner and D. Nehls. "Semantic-based Management of Federated Infrastructures for Future Internet Experimentation". In: *PhD Forum at the Conference on Networked Systems (NetSys)*. Cottbus: IEEE, Mar. 2015, p. 2 (cit. on pp. 78, 98, 143).

[a23] A. Willner, D. Nehls, and T. Magedanz. "FITeagle: A Semantic Testbed Management Framework". In: *Proceedings of the 10th EAI International Conference on Testbeds and Research Infrastructures for the Development of Networks & Communities*. Vancouver: ACM, Aug. 2015, pp. 1–6. ISBN: 978-1-63190-070-9. DOI: 10.4108/icst.tridentcom.2015.259748 (cit. on pp. 6, 78, 88, 98, 143).

Bibliography

[a24] A. Willner, C. Papagianni, M. Giatili, P. Grosso, M. Morsey, Y. Al-Hazmi, and I. Baldin. "The Open-Multinet Upper Ontology Towards the Semantic-based Management of Federated Infrastructures". In: *10th EAI International Conference on Testbeds and Research Infrastructures for the Development of Networks & Communities (TRIDENTCOM)*. Vancouver, Canada: ACM, Aug. 2015, p. 10. ISBN: 978-1-63190-070-9. DOI: 10.4108/icst.tridentcom.2015.259750 (cit. on pp. 6, 44, 61–63, 78, 82, 98, 113, 114, 143).

[a25] A. Willner, J. F. Riera, J. Garcia-Espin, S. Figuerola, M. De Leenheer, and C. Develder. "An analytical model of the service provisioning time within the harmony network service plane". In: *Globecom Workshops*. Miami: IEEE, Dec. 2010, pp. 514–518. ISBN: 978-1-4244-8863-6. DOI: 10.1109/GLOCOMW.2010.5700373 (cit. on p. 143).

References to Scientific Publications

[p1] R. L. Ackoff. "From data to wisdom". In: *Journal of applied systems analysis* 16.1 (1989), pp. 3–9 (cit. on p. 44).

[p2] C. Adams and S. Lloyd. *Understanding PKI: concepts, standards, and deployment considerations*. Addison-Wesley Professional, 2003 (cit. on p. 33).

[p3] A. M. Alberti and D. Singh. "Internet of Things: Perspectives, Challenges and Opportunities". In: *Int. WS on Telecommunications (IWT)*. IEEE, 2013, pp. 1–6 (cit. on p. 5).

[p4] F. Alvarez, J. Gonzalez, F. M. Facca, and S. Cretti. "Technical and functional solutions to build a community cloud for future Internet services from an Infrastructure Owner Perspective". In: *Workshop on Test beds for the Networks & Communications community (EUCNC)*. Open Access, 2014, pp. 1–5 (cit. on p. 20).

[p5] A.-C. Anadiotis, A. Apostolaras, D. Syrivelis, T. Korakis, L. Tassiulas, L. Rodriguez, and M. Ott. "A new slicing scheme for efficient use of wireless testbeds". In: *4th ACM international workshop on Experimental evaluation and characterization (WINTECH)*. New York, New York, USA: ACM Press, 2009, p. 83. ISBN: 9781605587400. DOI: 10.1145/1614293.1614311 (cit. on p. 26).

[p6] A.-C. Anadiotis, A. Apostolaras, D. Syrivelis, T. Korakis, L. Tassiulas, L. Rodriguez, I. Seskar, and M. Ott. "Towards Maximizing Wireless Testbed Utilization Using Spectrum Slicing". In: *Testbeds and Research Infrastructures - Development of Networks and Communities*. Springer, 2011, pp. 299–314. DOI: 10.1007/978-3-642-17851-1_25 (cit. on p. 26).

[p7] L. Aroyo, S. Pokraev, and R. Brussee. "Preparing SCORM for the Semantic Web". In: *On The Move to Meaningful Internet Systems: CoopIS, DOA, and ODBASE*. Springer, 2003, pp. 621–634. DOI: 10.1007/978-3-540-39964-3_40 (cit. on p. 149).

[p8] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. "DBpedia: A Nucleus for a Web of Open Data". In: *The Semantic Web (ISWC)*. Springer, 2007, pp. 722–735. DOI: 10.1007/978-3-540-76298-0_52 (cit. on p. 51).

[p9] I. Baldine, Y. Xin, A. Mandal, C. H. Renci, U.-C. J. Chase, V. Marupadi, A. Yumerefendi, and D. Irwin. "Networked cloud orchestration: A GENI perspective". In: *Globecom Workshops*. IEEE, Dec. 2010, pp. 573–578. ISBN: 978-1-4244-8863-6. DOI: 10.1109/GLOCOMW.2010.5700385 (cit. on p. 54).

[p10] I. Baldine, Y. Xin, A. Mandal, C. H. Renci, U.-C. J. Chase, V. Marupadi, A. Yumerefendi, and D. Irwin. "Networked cloud orchestration: A GENI perspective". In: *Globecom Workshops*. IEEE, Dec. 2010, pp. 573–578. ISBN: 978-1-4244-8863-6. DOI: 10.1109/GLOCOMW.2010.5700385 (cit. on p. 16).

[p11] I. Baldine, Y. Xin, A. Mandal, P. Ruth, C. Heerman, and J. Chase. "ExoGENI: A Multi-domain Infrastructure-as-a-Service Testbed". In: *Testbeds and Research Infrastructure. Development of Networks and Communities*. Ed. by T. Korakis, M. Zink, and M. Ott. Vol. 44. June. Springer, 2012, pp. 97–113. ISBN: 978-3-642-35575-2. DOI: 10.1007/978-3-642-35576-9_12 (cit. on p. 54).

[p12] D. F. Barbieri, D. Braga, S. Ceri, E. Della Valle, and M. Grossniklaus. "C-SPARQL". In: *18th international conference on World wide web - (WWW)*. New York, New York, USA: ACM Press, 2009, p. 1061. ISBN: 9781605584874. DOI: 10.1145/1526709.1526856 (cit. on p. 52).

[p13] C. Batini, M. Lenzerini, and S. B. Navathe. "A comparative analysis of methodologies for database schema integration". In: *ACM Computing Surveys* 18.4 (Dec. 1986), pp. 323–364. ISSN: 03600300. DOI: 10.1145/27633.27634 (cit. on p. 46).

[p14] K. Beck. *Test Driven Development: By Example*. Addison-Wesley Professional, 2002, p. 240. ISBN: 0321146530 (cit. on p. 67).

[p15] M. T. Beck, M. Werner, S. Feld, and S. Schimper. "Mobile Edge Computing: A Taxonomy". In: *6th International Conference on Advances in Future Internet*. IARIA, 2014, pp. 48–54 (cit. on p. 148).

[p16] A. Belghiat and M. Bourahla. "From UML class diagrams to OWL ontologies : a Graph transformation based Approach". In: *4th International Conference on Web and Information Technologies (ICWIT)*. Open Access CEUR, Apr. 2012, pp. 330–335 (cit. on p. 47).

[p17] M. Berman, J. S. Chase, L. Landweber, A. Nakao, M. Ott, D. Raychaudhuri, R. Ricci, and I. Seskar. "GENI: A federated testbed for innovative network experiments". In: *Computer Networks* 61.3 (Mar. 2014), pp. 5–23. ISSN: 13891286. DOI: 10.1016/j.bjp.2013.12.037 (cit. on p. 2).

[p18] T. Berners-Lee, J. Hendler, and O. Lassila. "The Semantic Web". In: *Scientific American* 284.5 (May 2001), pp. 34–43. ISSN: 0036-8733. DOI: 10.1038/scientificamerican0501-34 (cit. on p. 6).

[p19] D. Bernstein, E. Ludvigson, K. Sankar, S. Diamond, and M. Morrow. "Blueprint for the Intercloud - Protocols and Formats for Cloud Computing Interoperability". In: *4th International Conference on Internet and Web Applications and Services*. IEEE, May 2009, pp. 328–336. ISBN: 978-1-4244-3851-8. DOI: 10.1109/ICIW.2009.55 (cit. on p. 13).

[p20] D. Bernstein and D. Vij. “Intercloud Security Considerations”. In: *2nd International Conference on Cloud Computing Technology and Science*. IEEE. IEEE, Nov. 2010, pp. 537–544. ISBN: 978-1-4244-9405-7. DOI: 10.1109/CloudCom.2010.82 (cit. on p. 13).

[p21] R. Blanco, P. Mika, and S. Vigna. “Effective and Efficient Entity Search in RDF Data”. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 7031 LNCS. PART 1. Springer, 2011, pp. 83–97. ISBN: 9783642250729. DOI: 10.1007/978-3-642-25073-6_6 (cit. on p. 147).

[p22] O. Bohl, J. Scheuhase, R. Sengler, and U. Winand. “The sharable content object reference model (SCORM) - a critical review”. In: *International Conference on Computers in Education*. Vol. 1. IEEE. IEEE Comput. Soc, 2002, pp. 950–951. ISBN: 0-7695-1509-6. DOI: 10.1109/CIE.2002.1186122 (cit. on p. 149).

[p23] H. Boley, S. Tabet, and G. Wagner. “Design Rationale for RuleML: A Markup Language for Semantic Web Rules.” In: *International Semantic Web Working Symposium (SWWS)*. Vol. 1. Open Access SWWS, 2001, pp. 381–401 (cit. on p. 50).

[p24] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. “Freebase”. In: *SIGMOD international conference on Management of data*. New York, New York, USA: ACM Press, 2008, p. 1247. ISBN: 9781605581026. DOI: 10.1145/1376616.1376746 (cit. on p. 51).

[p25] A. Bolles, M. Grawunder, and J. Jacobi. “Streaming SPARQL - Extending SPARQL to Process Data Streams”. In: *The Semantic Web: Research and Applications*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 448–462. DOI: 10.1007/978-3-540-68234-9_34 (cit. on p. 52).

[p26] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli. “Fog computing and its role in the internet of things”. In: *1st edition of the workshop on Mobile cloud computing (MCC)*. New York, New York, USA: ACM Press, 2012, p. 13. ISBN: 9781450315197. DOI: 10.1145/2342509.2342513 (cit. on p. 147).

[p27] G. Booch. *The Unified Modeling Language User Guide, 2/E*. Pearson Education India, 2005 (cit. on p. 47).

[p28] T. Bourgeau, J. Augé, and T. Friedman. “TopHat: Supporting Experiments through Measurement Infrastructure Federation”. In: *Testbeds and Research Infrastructures - Development of Networks and Communities*. Springer, 2011, pp. 542–557. DOI: 10.1007/978-3-642-17851-1_41 (cit. on p. 28).

[p29] U. Bub and H. Woesner. “The National Beta Platform, Repository and Infrastructure for Sustainable Research Environments”. In: *Kommunikation in Verteilten Systemen (KiVS)*. Kassel: Gesellschaft für Informatik (GI), 2009, pp. 1–8 (cit. on p. 21).

[p30] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic. “Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility”. In: *Future Generation computer systems* 25.6 (2009), pp. 599–616 (cit. on p. 12).

[p31] V. Čačković and Ž. Popović. "Abstraction and Semantics support in M2M communications". English. In: *36th Int. Convention on Information Communication Technology Electronics Microelectronics (MIPRO)*. IEEE, 2013, pp. 404–408. ISBN: 978-953-233-076-2 (cit. on p. 53).

[p32] J.-P. Calbimonte, S. Sarni, J. Eberle, and K. Aberer. "XGSN: An Open-source Semantic Sensing Middleware for the Web of Things". In: *7th International Workshop on Semantic Sensor Networks*. Elsevier, 2014, pp. 1–16 (cit. on p. 53).

[p33] D. V. Camarda, S. Mazzini, and A. Antonuccio. "LodLive, exploring the web of data". In: *8th International Conference on Semantic Systems (I-SEMANTICS)*. New York, New York, USA: ACM Press, Sept. 2012, p. 197. ISBN: 9781450311120. DOI: 10.1145/2362499.2362532 (cit. on p. 90).

[p34] M. Campanella. "The FEDERICA Project: A federated infrastructure for Future Internet research". In: *EURESCOM mess@ge* 2.1 (2008), p. 11 (cit. on p. 17).

[p35] K. Campowsky, T. Magedanz, and S. Wahle. "Resource management in large scale experimental facilities". English. In: *Network Operations and Management Symposium (NOMS)*. IEEE, 2010, pp. 930–933. ISBN: 978-1-4244-5366-5. DOI: 10.1109/NOMS.2010.5488336 (cit. on p. 17).

[p36] J. Cardoso, T. Binz, U. Breitenbücher, O. Kopp, and F. Leymann. "Cloud Computing Automation: Integrating USDL and TOSCA". In: *CAiSE*. Vol. 7908. Lecture Notes in Computer Science (LNCS). Springer-Verlag, 2013, pp. 1–16. DOI: 10.1007/978-3-642-38709-8_1 (cit. on p. 53).

[p37] C. Catlett. "In search of gigabit applications". In: *IEEE Communications Magazine* 30.4 (Apr. 1992), pp. 42–51. ISSN: 0163-6804. DOI: 10.1109/35.135788 (cit. on pp. 1, 11).

[p38] V. G. Cerf. "Unfinished Business". In: *IEEE Internet Computing* 18.1 (Jan. 2014), pp. 88–88. ISSN: 1089-7801. DOI: 10.1109/MIC.2014.18 (cit. on p. 2).

[p39] D. Chappell. *Enterprise service bus*. " O'Reilly Media, Inc.", 2004 (cit. on p. 67).

[p40] J. Chase, L. Grit, and D. Irwin. "Beyond virtual data centers: Toward an open resource control architecture". In: *Selected Papers from the International Conference on the Virtual Computing Initiative*. ACM, 2007, pp. 1–10 (cit. on p. 16).

[p41] H. Chen, T. Finin, and A. Joshi. "Semantic Web in the context broker architecture". In: *2nd Annual Conference on Pervasive Computing and Communications*. DTIC Document. IEEE, 2004, pp. 277–286. ISBN: 0-7695-2090-1. DOI: 10.1109/PERCOM.2004.1276865 (cit. on p. 66).

[p42] H. W. Chesbrough. *Open innovation: The new imperative for creating and profiting from technology*. Harvard Business Press, 2003 (cit. on p. 20).

[p43] B. Chun, D. Culler, T. Roscoe, A. Bavier, L. Peterson, M. Wawrzoniak, and M. Bowman. "PlanetLab". In: *SIGCOMM Computer Communication Review* 33.3 (July 2003), p. 3. ISSN: 01464833. DOI: 10.1145/956993.956995 (cit. on p. 16).

[p44] G. Clapp, R. A. Skoog, A. C. Von Lehmen, and B. Wilson. "Management of switched systems at 100 Tbps: The DARPA CORONET program". In: *International Conference on Photonics in Switching*. IEEE, Sept. 2009, pp. 1–4. ISBN: 978-1-4244-3857-0. DOI: 10.1109/PS.2009.5307846 (cit. on p. 146).

[p45] D. Clark. "The design philosophy of the DARPA internet protocols". In: *Symposium proceedings on Communications architectures and protocols (SIGCOMM)*. Vol. 18. 4. New York, New York, USA: ACM Press, Aug. 1988, pp. 106–114. ISBN: 0897912799. DOI: 10.1145/52324.52336 (cit. on p. 2).

[p46] D. Clark, J. Wroclawski, K. Sollins, and R. Braden. "Tussle in cyberspace: defining tomorrow's Internet". In: *Transactions on Networking*. Vol. 13. 3. ACM, June 2005, pp. 462–475. DOI: 10.1109/TNET.2005.850224 (cit. on p. 2).

[p47] M. Compton, P. Barnaghi, L. Bermudez, R. Garcıa-Castro, O. Corcho, S. Cox, J. Graybeal, M. Hauswirth, C. Henson, A. Herzog, V. Huang, K. Janowicz, W. D. Kelsey, D. Le Phuoc, L. Lefort, M. Leggieri, H. Neuhaus, A. Nikolov, K. Page, A. Passant, A. Sheth, and K. Taylor. "The SSN ontology of the W3C semantic sensor network incubator group". In: *Web Semantics: Science, Services and Agents on the World Wide Web* 17.1 (Dec. 2012), pp. 25–32. ISSN: 15708268. DOI: 10.1016/j.websem.2012.05.003 (cit. on p. 53).

[p48] J. O. Coplien and G. Bjørnvig. *Lean Architecture: for Agile Software Development*. Wiley, 2010, p. 376. ISBN: 0470684208 (cit. on p. 66).

[p49] O. Corcho, P. Alper, I. Kotsiopoulos, P. Missier, S. Bechhofer, and C. Goble. "An overview of S-OGSA: A reference semantic grid architecture". In: *Web Semantics: Science, Services and Agents on the World Wide Web* 4.2 (2006), pp. 102–115 (cit. on p. 52).

[p50] M. Corici, H. Coskun, A. Elmangoush, A. Kurniawan, T. Mao, T. Magedanz, and S. Wahle. "OpenMTC: Prototyping Machine Type communication in carrier grade operator networks". In: *2012 IEEE Globecom Workshops*. IEEE, Dec. 2012, pp. 1735–1740. ISBN: 978-1-4673-4941-3. DOI: 10.1109/GLOCOMW.2012.6477847 (cit. on p. 94).

[p51] M. Cristani and R. Cuel. "A survey on ontology creation methodologies". In: *International Journal on Semantic Web and Information Systems (IJSWIS)* 1.2 (2005), pp. 49–69 (cit. on p. 56).

[p52] J. Crowcroft, P. Demeester, J. Magen, P. Tran-gia, and J. Wilander. "Towards a collaboration and high-level federation structure for the FIRE Facility". In: *Working Group on modular federation of FIRE Facilities* 1.1 (2009), pp. 1–18 (cit. on pp. 2, 19).

[p53] M. D'Aquin and E. Motta. "Watson, more than a semantic web search engine". In: *Semantic Web* 2.1 (2011), pp. 55–63 (cit. on p. 135).

[p54] S. Das, G. Parulkar, and N. McKeown. "SDN based unified control architecture". In: *Photonics Conference*. IEEE, Sept. 2012, pp. 778–779. ISBN: 978-1-4577-0733-9. DOI: 10.1109/IPCon.2012.6358854 (cit. on p. 5).

[p55] F. De Turck, S. Vanhastel, F. Vandermeulen, and P. Demeester. "Design and implementation of a generic connection management and service level agreement monitoring platform supporting the virtual private network service". In: *International Symposium on Integrated Network Management*. IEEE, 2001, pp. 153–166. ISBN: 0-7803-6719-7. DOI: 10.1109/INM.2001.918020 (cit. on p. 146).

[p56] T. DeFanti, I. Foster, M. Papka, R. Stevens, and T. Kuhfuss. "Overview of the I-WAY: Widearea visual supercomputing". In: *International Journal of Supercomputer Applications*. Vol. 10. 2-3. SAGE Publications, 1996, pp. 123–130 (cit. on p. 1).

[p57] Y. Demchenko, C. Ngo, C. de Laat, J. Rodriguez, L. M. Contreras, J. A. Garcia-Espin, S. Figuerola, G. Landi, and N. Ciulli. "Intercloud Architecture Framework for Heterogeneous Cloud Based Infrastructure Services Provisioning On-Demand". In: *27th International Conference on Advanced Information Networking and Applications Workshops*. IEEE, Mar. 2013, pp. 777–784. ISBN: 978-1-4673-6239-9. DOI: 10.1109/WAINA.2013.237 (cit. on p. 13).

[p58] Y. Demchenko, J. van der Ham, C. Ngo, T. Matselyukh, S. Filiposka, C. de Laat, and E. Escalona. "Open Cloud eXchange (OCX): Architecture and Functional Components". In: *5th International Conference on Cloud Computing Technology and Science*. Vol. 2. IEEE, Dec. 2013, pp. 81–87. ISBN: 978-0-7695-5095-4. DOI: 10.1109/CloudCom.2013.108 (cit. on p. 147).

[p59] E. W. Dijkstra. "On the Role of Scientific Thought". In: *Selected Writings on Computing: A personal Perspective*. New York, NY: Springer New York, 1982. Chap. On the rol, pp. 60–66. DOI: 10.1007/978-1-4612-5695-3_12 (cit. on p. 66).

[p60] F. Dijkstra and C. de Laat. "Optical Exchanges". In: *1st Workshop on Networks for Grid Applications (GridNets)*. IEEE, 2004, pp. 1–6 (cit. on p. 146).

[p61] L. Ding, T. Finin, A. Joshi, R. Pan, R. S. Cost, Y. Peng, P. Reddivari, V. Doshi, and J. Sachs. "Swoogle". In: *13th ACM conference on Information and knowledge management (CIKM)*. New York, New York, USA: ACM Press, 2004, p. 652. ISBN: 1581138741. DOI: 10.1145/1031171.1031289 (cit. on p. 135).

[p62] M. Dougiamas and P. Taylor. "Moodle: Using learning communities to create an open source course management system". In: *World conference on educational multimedia, hypermedia and telecommunications*. Vol. 2003. 1. Honolulu, Hawaii: Association for the Advancement of Computing in Education (AACE), 2003, pp. 171–178. ISBN: 978-1-880094-48-8 (cit. on p. 149).

[p63] M. Drozdowicz, M. Ganzha, M. Paprzycki, R. Olejnik, I. Lirkov, P. Telegin, M. Senobari, et al. "Ontologies, agents and the grid: An overview". In: *Parallel, Distributed and Grid Computing for Engineering*. Stirlingshire: Saxe-Coburg Publications, 2009, pp. 117–140. DOI: 10.4203/csets.21.7 (cit. on p. 52).

[p64] C. L. Dumitrescu and I. Foster. "GRUBER: A Grid Resource Usage SLA Broker". In: *Euro-Par Parallel Processing*. Springer, 2005, pp. 465–474. DOI: 10.1007/11549468_53 (cit. on p. 12).

Bibliography

[p65] S. M. Easterbrook. "The Role of Independent V & V in Upstream Software Development Processes". In: *2nd World Conference on Integrated Design and Process Technology (IDPT)*. Austin, Texas: Society for Design and Process Science, Dec. 1996. Chap. The Proces, p. 488 (cit. on p. 98).

[p66] E. Escalona, S. Peng, R. Nejabati, D. Simeonidou, J. A. Garcia-Espin, J. F. Riera, S. Figuerola, and C. de Laat. "GEYSERS: A Novel Architecture for Virtualization and Co-Provisioning of Dynamic Optical Networks and IT Services". In: *Future Network and Mobile Summit*. IEEE, 2011, pp. 1–8 (cit. on p. 54).

[p67] D. S. Evans, R. Schmalensee, M. D. Noel, H. H. Chang, and D. D. Garcia-Swartz. *Platform economics: Essays on multi-sided businesses*. Competition Policy International (CPI), 2011, p. 459 (cit. on p. 3).

[p68] T. Faber and J. Wroclawski. "A federated experiment environment for emulab-based testbeds". In: *5th International Conference on Testbeds and Research Infrastructures for the Development of Networks & Communities and Workshops (TRIDENTCOM)*. IEEE, 2009, pp. 1–10. ISBN: 978-1-4244-2846-5. DOI: 10.1109/TRIDENTCOM.2009.4976238 (cit. on p. 16).

[p69] T. Faber, J. Wroclawski, and K. Lahey. "A DETER federation architecture". In: *DETER Community Workshop on Cyber Security Experimentation and Test*. Open Access DETER, 2007, pp. 1–8 (cit. on p. 16).

[p70] R. d. A. Falbo, C. S. de Menezes, and A. R. C. da Rocha. "A Systematic Approach for Building Ontologies". In: *6th Ibero-American Conference on AI: Progress in Artificial Intelligence (IBERAMIA)*. London, UK: Springer, 1998, pp. 349–360. DOI: 10.1007/3-540-49795-1_31 (cit. on p. 48).

[p71] R. d. A. Falbo, G. Guizzardi, and K. C. Duarte. "An ontological approach to domain engineering". In: *14th international conference on Software engineering and knowledge engineering (SEKE)*. New York, New York, USA: ACM Press, 2002, p. 351. ISBN: 1581135564. DOI: 10.1145/568760.568822 (cit. on p. 48).

[p72] S. Fdida, T. Friedman, and T. Parmentelat. "OneLab: An Open Federated Facility for Experimentally Driven Future Internet Research". In: *New Network Architectures*. Springer, 2010, pp. 141–152. ISBN: 978-3-642-13246-9. DOI: 10.1007/978-3-642-13247-6_7 (cit. on p. 17).

[p73] A. Feldmann. "Internet clean-slate design". In: *ACM SIGCOMM Computer Communication Review* 37.3 (July 2007), p. 59. ISSN: 01464833. DOI: 10.1145/1273445.1273453 (cit. on p. 2).

[p74] J. D. Fernández, M. A. Martınez-Prieto, M. Arias, C. Gutierrez, S. Álvarez-Garcıa, and N. R. Brisaboa. "Lightweighting the Web of Data through Compact RDF/HDT". In: *14th Conference of the Spanish Association for Artificial Intelligence (CAEPIA)*. Vol. 7023. Lecture Notes in Computer Science. Springer, 2011, pp. 483–493. DOI: 10.1007/978-3-642-25274-7_49 (cit. on p. 96).

[p75] B. Foote and R. E. Johnson. "Designing reusable classes". In: *Journal of Object-Oriented Programming* 2.1 (1988), pp. 22–35 (cit. on p. 6).

[p76] I. Foster. "The Grid: A New Infrastructure for 21st Century Science". In: *Physics Today* 55.2 (2002), pp. 42–47 (cit. on p. 12).

[p77] I. Foster and C. Kesselman. "Globus: a Metacomputing Infrastructure Toolkit". In: *International Journal of High Performance Computing Applications* 11.2 (1997), p. 115 (cit. on p. 11).

[p78] I. Foster and C. Kesselman. "The History of the Grid". In: *Computing* 20.21 (2010), p. 22 (cit. on p. 11).

[p79] I. Foster, C. Kesselman, J. M. Nick, and S. Tuecke. "The Physiology of the Grid". In: *Grid computing: making the global infrastructure a reality*. John Wiley & Sons, Ltd., 2003, pp. 217–249 (cit. on p. 11).

[p80] I. Foster. "Globus Toolkit Version 4: Software for Service-Oriented Systems". In: *Journal of Computer Science and Technology*. Vol. 21. 4. Springer, July 2006, pp. 513–520. DOI: 10.1007/s11390-006-0513-y (cit. on p. 11).

[p81] I. Foster. "What is the Grid? A Three Point Checklist". In: *Grid Today* 1.6 (2002), pp. 32–36 (cit. on p. 11).

[p82] I. Foster, M. Fidler, A. Roy, V. Sander, and L. Winkler. "End-to-end quality of service for high-end applications". In: *Computer Communications* 27.14 (Sept. 2004), pp. 1375–1388. ISSN: 01403664. DOI: 10.1016/j.comcom.2004.02.014 (cit. on p. 12).

[p83] I. Foster and C. Kesselman. *The Grid 2: Blueprint for a New Computing Infrastructure*. 2nd. Morgan Kaufmann, 2003, p. 748. ISBN: 978-1558609334 (cit. on p. 11).

[p84] I. Foster and C. Kesselman. *The Grid: Blueprint for a New Computing Infrastructure: Blueprint for the New Computing Infrastructure*. Morgan Kaufmann Publishers, 1998, p. 675. ISBN: 1558604758 (cit. on p. 11).

[p85] I. Foster, C. Kesselman, and S. Tuecke. "The Anatomy of the Grid: Enabling Scalable Virtual Organizations". In: *International Journal of High Performance Computing Applications* 15.3 (2001), p. 200 (cit. on p. 11).

[p86] I. Foster, C. Kesselman, and S. Tuecke. "The Nexus task-parallel runtime system". In: *1st International Workshop on Parallel Processing*. IEEE, 1994, pp. 457–462 (cit. on p. 11).

[p87] I. Foster, Y. Zhao, I. Raicu, and S. Lu. "Cloud Computing and Grid Computing 360-Degree Compared". In: *Grid Computing Environments Workshop*. IEEE, Nov. 2008, pp. 1–10. ISBN: 978-1-4244-2860-1. DOI: 10.1109/GCE.2008.4738445 (cit. on p. 12).

[p88] J. R. Frade, D. Di Giacomo, S. Goedertier, N. Loutas, and V. Peristeras. "Building semantic interoperability through the federation of semantic asset repositories". In: *8th International Conference on Semantic Systems (I-SEMANTICS)*. New York, New York, USA: ACM Press, Sept. 2012, p. 185. ISBN: 9781450311120. DOI: 10.1145/2362499.2362528 (cit. on p. 51).

[p89] C. D. Freire, A. Quereilhac, T. Turletti, and W. Dabbous. "Automated Deployment and Customization of Routing Overlays on Planetlab". In: *Testbeds and Research Infrastructure - Development of Networks and Communities (TRIDENTCOM)*. Springer, 2012, pp. 240–255. DOI: 10.1007/978-3-642-35576-9_21 (cit. on p. 17).

[p90] T. Furche, B. Linse, F. Bry, D. Plexousakis, and G. Gottlob. "RDF Querying: Language Constructs and Evaluation Methods Compared". In: *Reasoning Web*. Springer, 2006, pp. 1–52. DOI: 10.1007/11837787_1 (cit. on p. 50).

[p91] S. Gaglio and G. Lo Re. *Advances onto the Internet of Things*. Ed. by S. Gaglio and G. Lo Re. Vol. 260. Advances in Intelligent Systems and Computing. Cham: Springer International Publishing, 2014. ISBN: 978-3-319-03991-6. DOI: 10.1007/978-3-319-03992-3 (cit. on p. 53).

[p92] A. Gavras, H. Bruggemann, D. Witaszek, K. Sunell, and J. Jimenez. "Pan European Laboratory for Next Generation Networks and Services". In: *2nd International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TRIDENTCOM)*. IEEE, 2006, pp. 436–441. ISBN: 1-4244-0106-2. DOI: 10.1109/TRIDNT.2006.1649180 (cit. on p. 17).

[p93] A. Gavras, H. Hrasnica, S. Wahle, D. Lozano, D. Mischler, and S. Denazis. "Control of Resources in Pan-European Testbed Federation". In: *Towards the Future Internet: A European Research Perspective*. IOS Press, 2009, pp. 67–78 (cit. on p. 18).

[p94] A. Gavras, A. Karila, S. Fdida, M. May, and M. Potts. "Future internet research and experimentation". In: *ACM SIGCOMM Computer Communication Review* 37.3 (July 2007), p. 89. ISSN: 01464833. DOI: 10.1145/1273445.1273460 (cit. on p. 2).

[p95] J. Geelan. "Twenty-one experts define cloud computing". In: *Cloud Computing Journal* 4.1 (2009), pp. 1–5 (cit. on p. 12).

[p96] M. Ghijsen, J. van der Ham, P. Grosso, and C. de Laat. "Towards an Infrastructure Description Language for Modeling Computing Infrastructures". In: *10th International Symposium on Parallel and Distributed Processing with Applications*. IEEE, July 2012, pp. 207–214. ISBN: 978-1-4673-1631-6. DOI: 10.1109/ISPA.2012.35 (cit. on p. 54).

[p97] M. Ghijsen, J. van der Ham, P. Grosso, C. Dumitru, H. Zhu, Z. Zhao, and C. de Laat. "A Semantic-Web Approach for Modeling Computing Infrastructures". In: *Computers and Electrical Engineering* 39.8 (2013), pp. 2553–2565 (cit. on p. 54).

[p98] M. Giatili, C. Papagianni, and S. Papavassiliou. "Semantic aware resource mapping for future internet experimental facilities". In: *19th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*. IEEE, Dec. 2014, pp. 61–65. ISBN: 978-1-4799-5725-5. DOI: 10.1109/CAMAD.2014.7033206 (cit. on p. 145).

[p99] R. Gomathi, C. Sathya, and D. Sharmila. "Efficient Optimization of Multiple SPARQL Queries". In: *IOSR Journal of Computer Engineering (IOSR-JCE)* 8.6 (2013), pp. 97–101 (cit. on p. 52).

[p100] A. Gómez-Pérez and M. D. Rojas-Amaya. "Ontological Reengineering for Reuse". In: *Knowledge Acquisition, Modeling and Management*. Springer, 1999, pp. 139–156. DOI: 10.1007/3-540-48775-1_9 (cit. on p. 56).

[p101] E. Grasa, X. Hesselbach, S. Figuerola, V. Reijs, D. Wilson, J.-M. Uzé, L. Fischer, and T. de Miguel. "The MANTICORE project: providing users with a logical IP network service". In: *TERENA Networking Conference*. Vol. 5. Open Access, 2008, pp. 1–11 (cit. on p. 18).

[p102] E. Grasa, G. Junyent, S. Figuerola, A. Lopez, and M. Savoie. "UCLPv2: a network virtualization framework built on web services". In: *IEEE Communications Magazine* 46.3 (Mar. 2008), pp. 126–134. ISSN: 0163-6804. DOI: 10.1109/MCOM.2008.4463783 (cit. on p. 74).

[p103] L. Green, V. Mirchandani, I. Cergol, and D. Verchere. "Design of a dynamic SLA negotiation protocol for grids". In: *1st international conference on Networks for grid applications*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering). Open Access ICST, 2007, p. 13 (cit. on p. 12).

[p104] N. Grozev and R. Buyya. "Inter-Cloud architectures and application brokering: taxonomy and survey". In: *Software: Practice and Experience* 44.3 (Mar. 2014), pp. 369–390. ISSN: 00380644. DOI: 10.1002/spe.2168 (cit. on pp. 3, 5, 13, 53).

[p105] T. Gruber. "A translation approach to portable ontology specifications". In: *Knowledge acquisition* 5.2 (1993), pp. 199–220 (cit. on p. 48).

[p106] D. Guinard, V. Trifa, F. Mattern, and E. Wilde. "From the Internet of Things to the Web of Things: Resource-oriented Architecture and Best Practices". In: *Architecting the Internet of Things*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 97–129. DOI: 10.1007/978-3-642-19157-2_5 (cit. on p. 147).

[p107] C. Guok, D. Robertson, M. Thompson, J. Lee, B. Tierney, and W. Johnston. "Intra and Interdomain Circuit Provisioning Using the OSCARS Reservation System". In: *Broadband Communications, Networks and Systems, 2006. BROADNETS 2006. 3rd International Conference on*. 2006, pp. 1–8 (cit. on p. 74).

[p108] M. Hammer and D. McLeod. "The semantic data model: a modelling mechanism for data base applications". In: *International Conference on Management of Data (SIGMOD)*. ACM, 1978, pp. 26–36 (cit. on p. 44).

[p109] T. Han and K. M. Sim. "An ontology-enhanced cloud service discovery system". In: *International MultiConference of Engineers and Computer Scientists*. Vol. 1. Springer, 2010, pp. 17–19 (cit. on p. 53).

[p110] D. Havlik, S. Schade, Z. Sabeur, P. Mazzetti, K. Watson, A. J. Berre, and J. L. Mon. "From Sensor to Observation Web with environmental enablers in the Future Internet". In: *Sensors* 11.4 (Jan. 2011), pp. 3874–907. ISSN: 1424-8220. DOI: 10.3390/s110403874 (cit. on p. 14).

[p111] Y. Al-Hazmi, K. Campowsky, and T. Magedanz. "A monitoring system for federated clouds". In: *1st International Conference on Cloud Networking (CLOUDNET)*. IEEE, Nov. 2012, pp. 68–74. ISBN: 978-1-4673-2798-5. DOI: 10.1109/CloudNet.2012.6483657 (cit. on p. 28).

[p112] T. Heath and C. Bizer. "Linked data: Evolving the web into a global data space". In: *Synthesis lectures on the semantic web: theory and technology* 1.1 (2011), pp. 1–136. DOI: 10.2200/S00334ED1V01Y201102WBE001 (cit. on p. 51).

[p113] T. Heger and U. Bub. "The EIT ICT Labs — Towards a Leading European Innovation Initiative". In: *it - Information Technology* 54.6 (Nov. 2012), pp. 288–295. ISSN: 1611-2776. DOI: 10.1524/itit.2012.0691 (cit. on p. 21).

[p114] J. Hendler and J. Golbeck. "Metcalfe's law, Web 2.0, and the Semantic Web". In: *Web Semantics: Science, Services and Agents on the World Wide Web* 6.1 (Feb. 2008), pp. 14–20. ISSN: 15708268. DOI: 10.1016/j.websem.2007.11.008 (cit. on p. 145).

[p115] M. Hepp. "GoodRelations: An Ontology for Describing Products and Services Offers on the Web". In: *Knowledge Engineering: Practice and Patterns*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 329–346. DOI: 10.1007/978-3-540-87696-0_29 (cit. on p. 134).

[p116] P. Hitzler, M. Krotzsch, and S. Rudolph. *Foundations of semantic web technologies*. CRC Press, 2011 (cit. on p. 46).

[p117] D. Hovemeyer and W. Pugh. "Finding bugs is easy". In: *ACM SIGPLAN Notices* 39.12 (Dec. 2004), p. 92. ISSN: 03621340. DOI: 10.1145/1052883.1052895 (cit. on p. 136).

[p118] J. Huang, D. J. Abadi, and K. Ren. "Scalable SPARQL querying of large RDF graphs". In: *VLDB Endowment* 4.11 (2011), pp. 1123–1134 (cit. on p. 52).

[p119] V. Huber. "UNICORE: A Grid Computing Environment for Distributed and Parallel Computing". In: *Euro-Par 2001 Parallel Processing*. Springer, 2001, pp. 258–265. DOI: 10.1007/3-540-44743-1_25 (cit. on p. 11).

[p120] J. S. Hughes, D. J. Crichton, and C. a. Mattmann. "Ontology-based information model development for science information reuse and integration". In: *International Conference on Information Reuse & Integration*. IEEE, Aug. 2009, pp. 79–84. ISBN: 978-1-4244-4114-3. DOI: 10.1109/IRI.2009.5211603 (cit. on p. 46).

[p121] A. C. Hume, Y. Al-Hazmi, B. Belter, K. Campowsky, L. M. Carril, G. Carrozzo, V. Engen, D. Garcıa-Pérez, J. Jofre Ponsatı, R. Kűbert, Y. Liang, C. Rohr, and G. Seghbroeck. *Testbeds and Research Infrastructure. Development of Networks and Communities*. Ed. by T. Korakis, M. Zink, and M. Ott. Vol. 44. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 81–96. ISBN: 978-3-642-35575-2. DOI: 10.1007/978-3-642-35576-9 (cit. on p. 28).

[p122] K. Hwang. "Massively Distributed Systems : From Grids and P2P to Clouds". In: *Advances in Grid and Pervasive Computing*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 1–1. DOI: 10.1007/978-3-540-68083-3_1 (cit. on p. 12).

[p123] I. Jacobson, M. Christerson, and P. Jonsson. *Object-oriented software engineering - a use case driven approach*. 4. A. Reno: ACM Press, 1992. ISBN: 978-0-201-54435-0 (cit. on p. 66).

[p124] R. Jain. *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*. John Wiley and Sons Ltd. Chichester, UK, 1991. ISBN: 978-0471503361 (cit. on pp. 2, 98).

[p125] M. L. Jayalal, R. Jehadeesan, S. Rajeswari, and S. A. V. S. Murty. "Moving From Grid to Cloud Computing: The Challenges in an Existing Computational Grid Setup". In: *International Journal of Computer Science & Communication (IJCSC)* 1.2 (2010), pp. 415–418 (cit. on p. 12).

[p126] H. Karimi and M. Hashemi. “Accessible Wayfinding Testbed: Infrastructure and Components”. In: *10th EAI International Conference on Testbeds and Research Infrastructures for the Development of Networks & Communities*. Vol. 15. 2. Canada: ACM, June 2015, pp. 1–8. ISBN: 978-1-63190-070-9. DOI: 10.4108/icst.tridentcom.2015.259709 (cit. on p. 55).

[p127] G. Karmous-Edwards, S. G. Polito, A. Jukan, and G. Rouskas. “A new framework for GLIF Interdomain Resource Reservation Architecture (GIRRA)”. In: *annals of telecommunications - annales des télécommunications* 65.11-12 (Dec. 2010), pp. 723–737. ISSN: 0003-4347. DOI: 10.1007/s12243-010-0186-y (cit. on p. 146).

[p128] C. M. Keet. “Closed World Assumption”. In: *Encyclopedia of Systems Biology*. Ed. by W. Dubitzky, O. Wolkenhauer, K.-H. Cho, and H. Yokota. New York, NY: Springer New York, 2013, pp. 415–415. DOI: 10.1007/978-1-4419-9863-7_731 (cit. on p. 48).

[p129] C. M. Keet. “Open World Assumption”. In: *Encyclopedia of Systems Biology*. Ed. by W. Dubitzky, O. Wolkenhauer, K.-H. Cho, and H. Yokota. New York, NY: Springer New York, 2013, pp. 1567–1567. ISBN: 978-1-4419-9862-0. DOI: 10.1007/978-1-4419-9863-7_734 (cit. on p. 48).

[p130] M. B. Kelly. “TeleManagement Forum (TMF) Report: The TeleManagement Forum’s Enhanced Telecom Operations Map (eTOM)”. In: *Journal of Network and Systems Management* 11.1 (2003), pp. 109–119. ISSN: 1064-7570. DOI: 10.1023/A:1022449209526 (cit. on p. 21).

[p131] R. Khondoker, A. Zaalouk, R. Marx, and K. Bayarou. “Feature-based comparison and selection of Software Defined Networking (SDN) controllers”. In: *World Congress on Computer Applications and Information Systems (WCCAIS)*. IEEE, Jan. 2014, pp. 1–7. DOI: 10.1109/WCCAIS.2014.6916572 (cit. on p. 148).

[p132] J. Kim and J.-W. Lee. “OpenIoT: An open service framework for the Internet of Things”. In: *World Forum on Internet of Things (WF-IoT)*. IEEE, Mar. 2014, pp. 89–93. ISBN: 978-1-4799-3459-1. DOI: 10.1109/WF-IoT.2014.6803126 (cit. on p. 53).

[p133] W. Kim and J. Seo. “Classifying schematic and data heterogeneity in multidatabase systems”. In: *Computer* 24.12 (Dec. 1991), pp. 12–18. ISSN: 0018-9162. DOI: 10.1109/2.116884 (cit. on p. 46).

[p134] M. Klein. “Combining and relating ontologies: an analysis of problems and solutions”. In: *Workshop on ontologies and information sharing (IJCAI)*. Open Access CEUR, 2001, pp. 53–62 (cit. on p. 49).

[p135] L. Kleinrock. “An early history of the internet [History of Communications”. In: *IEEE Communications Magazine* 48.8 (Aug. 2010), pp. 26–36. ISSN: 0163-6804. DOI: 10.1109/MCOM.2010.5534584 (cit. on p. 1).

[p136] L. Kleinrock. “An Internet vision: the invisible global infrastructure”. In: *Ad Hoc Networks* 1.1 (July 2003), pp. 3–11. ISSN: 15708705. DOI: 10.1016/S1570-8705(03)00012-X (cit. on p. 10).

[p137] L. Kleinrock. “Information Flow in Large Communication Nets”. Ph.D. Thesis Proposal. Massachusetts Institute of Technology, 1961 (cit. on p. 1).

[p138] J. Kopecký, T. Vitvar, C. Bournez, and J. Farrell. "SAWSDL: Semantic Annotations for WSDL and XML Schema". In: *IEEE Internet Computing* 11.6 (Nov. 2007), pp. 60–67. ISSN: 1089-7801. DOI: 10.1109/MIC.2007.134 (cit. on p. 53).

[p139] D. Kourtesis and I. Paraskakis. "Combining SAWSDL, OWL-DL and UDDI for Semantically Enhanced Web Service Discovery". In: *The Semantic Web: Research and Applications*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 614–628. DOI: 10.1007/978-3-540-68234-9_45 (cit. on p. 53).

[p140] K. Krauter and R. Buyya. "A taxonomy and survey of grid resource management systems for distributed computing". In: *Software-Practice and Experience* 32.2 (2002), pp. 135–64. DOI: 10.1002/spe.432 (cit. on p. 26).

[p141] R. Krummenacher, B. Norton, and A. Marte. "Towards Linked Open Services and Processes". In: *Future Internet (FIS)*. Springer, 2010, pp. 68–77. DOI: 10.1007/978-3-642-15877-3_8 (cit. on p. 53).

[p142] R. Krzywania, L. Dolata, P. Krawiec, W. Latoszek, A. Szymanski, and J. WszoLek. "PL-LAB: Polish Future Internet Distributed Laboratory". In: *13th International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (ACIS)*. IEEE, Aug. 2012, pp. 666–671. ISBN: 978-1-4673-2120-4. DOI: 10.1109/SNPD.2012.73 (cit. on p. 131).

[p143] M. Lacage, M. Ferrari, M. Hansen, T. Turletti, and W. Dabbous. "NEPI". In: *ACM SIGOPS Operating Systems Review* 43.4 (Jan. 2010), p. 60. ISSN: 01635980. DOI: 10.1145/1713254.1713268 (cit. on p. 17).

[p144] D. Lamanna, J. Skene, and W. Emmerich. "SLAng: A language for defining service level agreements". In: *9th Workshop on Future Trends of Distributed Computing Systems (FTDCS)*. IEEE, 2003, pp. 100–106. ISBN: 0-7695-1910-5. DOI: 10.1109/FTDCS.2003.1204317 (cit. on pp. 12, 32).

[p145] E. Laure, A. Edlund, F. Pacini, P. Buncic, M. Barroso, A. Di Meglio, F. Prelz, A. Frohner, O. Mulmo, A. Krenek, et al. "Programming the Grid with gLite". In: *Computational Methods in Science and Technology*. Open Access CERN, 2006, pp. 1–17 (cit. on p. 11).

[p146] W. Le, A. Kementsietsidis, S. Duan, and F. Li. "Scalable Multi-query Optimization for SPARQL". In: *28th International Conference on Data Engineering*. IEEE, Apr. 2012, pp. 666–677. ISBN: 978-0-7695-4747-3. DOI: 10.1109/ICDE.2012.37 (cit. on p. 52).

[p147] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, et al. "DBpedia-a large-scale, multilingual knowledge base extracted from wikipedia". In: *Semantic Web Journal (SWJ)* 5.1 (2014), pp. 1–29 (cit. on p. 120).

[p148] B. M. Leiner, V. G. Cerf, D. D. Clark, R. E. Kahn, L. Kleinrock, D. C. Lynch, J. Postel, L. G. Roberts, and S. Wolff. "A brief history of the internet". In: *ACM SIGCOMM Computer Communication Review* 39.5 (Oct. 2009), p. 22. ISSN: 01464833. DOI: 10.1145/1629607.1629613 (cit. on p. 1).

[p149] L. Liao, Y. Qu, and H. Leung. "A software process ontology and its application". In: *WS on Semantic Web Enabled Software Engineering (SWESE)*. Galway, Ireland: IOS Press, 2005, pp. 6–10. DOI: 10.3233/978-1-61499-370-4-207 (cit. on p. 48).

[p150] J. C. R. Licklider and W. E. Clark. "On-line man-computer communication". In: *Joint computer conference (AIEE-IRE)*. New York, New York, USA: ACM Press, May 1962, p. 113. DOI: 10.1145/1460833.1460847 (cit. on p. 1).

[p151] Y. B. Ma, S. H. Jang, and J. S. Lee. "Ontology-Based Resource Management for Cloud Computing". In: *Intelligent Information and Database Systems*. Springer, 2011, pp. 343–352. DOI: 10.1007/978-3-642-20042-7_35 (cit. on p. 53).

[p152] T. Magedanz and S. Wahle. "Control framework design for Future Internet testbeds". In: *e & i Elektrotechnik und Informationstechnik* 126.7-8 (July 2009), pp. 274–279. ISSN: 0932-383X. DOI: 10.1007/s00502-009-0655-z (cit. on p. 16).

[p153] P. Mähönen, T. Priol, and K. Pohl. *Towards a Service-Based Internet*. Ed. by W. Abramowicz, I. M. Llorente, M. Surridge, A. Zisman, and J. Vayssière. Vol. 6994. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. ISBN: 978-3-642-24754-5. DOI: 10.1007/978-3-642-24755-2 (cit. on p. 18).

[p154] R. Mall. *Fundamentals of Software Engineering*. Prentice Hall India Pvt., Limited, 2004. ISBN: 9788120324459 (cit. on p. 98).

[p155] D. Marples and P. Kriens. "The Open Services Gateway Initiative: an introductory overview". In: *IEEE Communications Magazine* 39.12 (2001), pp. 110–114. ISSN: 01636804. DOI: 10.1109/35.968820 (cit. on p. 79).

[p156] R. C. Martin. *Agile Software Development: Principles, Patterns, and Practices*. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2003. ISBN: 0135974445 (cit. on pp. 66, 67).

[p157] R. C. Martin. "The dependency inversion principle". In: *C++ Report* 8.6 (1996), pp. 61–66. URL: http://www.objectmentor.com/resources/articles/dip.pdf (cit. on p. 68).

[p158] P. Marwedel. "Evaluation and Validation". In: *Embedded System Design*. Dordrecht: Springer Netherlands, 2011, pp. 203–234. ISBN: 978-94-007-0256-1. DOI: 10.1007/978-94-007-0257-8_5 (cit. on p. 98).

[p159] P. Matray, I. Csabai, P. Haga, J. Steger, L. Dobos, and G. Vattay. "Building a prototype for network measurement virtual observatory". In: *3rd annual ACM workshop on Mining network data (MineNet)*. New York, New York, USA: ACM Press, 2007, p. 23. ISBN: 9781595937926. DOI: 10.1145/1269880.1269887 (cit. on p. 28).

[p160] P. Mcfredries. "Technically speaking: The cloud is the computer". In: *IEEE Spectrum* 45.8 (Aug. 2008), pp. 20–20. ISSN: 0018-9235. DOI: 10.1109/MSPEC.2008.4586277 (cit. on p. 12).

[p161] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner. "OpenFlow". In: *ACM SIGCOMM Computer Communication Review* 38.2 (Mar. 2008), p. 69. ISSN: 01464833. DOI: 10.1145/1355734.1355746 (cit. on p. 74).

[p162] L. Mei, W. Chan, and T. Tse. "A Tale of Clouds: Paradigm Comparisons and Some Thoughts on Research Issues". In: *Asia-Pacific Services Computing Conference*. IEEE, Dec. 2008, pp. 464–469. DOI: 10.1109/APSCC.2008.168 (cit. on p. 13).

[p163] P. Mika. “Distributed indexing for semantic search”. In: *3rd International Workshop on Semantic Search (SEMSEARCH)*. New York, New York, USA: ACM Press, 2010, pp. 1–4. ISBN: 9781450301305. DOI: 10.1145/1863879.1863882 (cit. on p. 147).

[p164] F. Moscato, R. Aversa, B. Di Martino, T. Fortis, and V. Munteanu. “An analysis of mOSAIC ontology for Cloud resources annotation”. In: *Federated Conference on Computer Science and Information Systems (FedCSIS)*. IEEE, 2011, pp. 973–980. ISBN: 978-1-4577-0041-5 (cit. on p. 13).

[p165] H. Mühleisen and C. Bizer. “Web data commons - Extracting structured data from two large web corpora”. In: *Sun SITE Central Europe Workshop (CEUR)*. Vol. 937. Open Access CEUR, 2012, pp. 133–145 (cit. on p. 148).

[p166] S. Newman. *Building Microservices*. O’Reilly Media, 2015, pp. 1–250. ISBN: 978-1491950357 (cit. on p. 6).

[p167] I. Niles and A. Pease. “Towards a standard upper ontology”. In: *International conference on Formal Ontology in Information Systems (FOIS)*. Vol. 2001. New York, New York, USA: ACM Press, 2001, pp. 2–9. ISBN: 1581133774. DOI: 10.1145/505168.505170 (cit. on p. 56).

[p168] Ninghui Li, J. Mitchell, and W. Winsborough. “Design of a role-based trust-management framework”. In: *Symposium on Security and Privacy*. IEEE Comput. Soc, 2002, pp. 114–130. ISBN: 0-7695-1543-6. DOI: 10.1109/SECPRI.2002.1004366 (cit. on p. 31).

[p169] N. F. Noy. “Semantic integration: a survey of ontology-based approaches”. In: *SIGMOD Record* 33.4 (Dec. 2004), p. 65. ISSN: 01635808. DOI: 10.1145/1041410.1041421 (cit. on p. 49).

[p170] N. F. Noy and M. Klein. “Ontology Evolution: Not the Same as Schema Evolution”. In: *Knowledge and Information Systems* 6.4 (July 2004), pp. 428–440. ISSN: 0219-1377. DOI: 10.1007/s10115-003-0137-2 (cit. on p. 49).

[p171] L. Obrst, W. Ceusters, I. Mani, S. Ray, and B. Smith. “The evaluation of ontologies”. In: *Semantic Web*. Springer, 2007, pp. 139–158. DOI: 10.1007/978-0-387-48438-9_8 (cit. on p. 99).

[p172] S. Ortiz. “The Problem with Cloud-Computing Standardization”. In: *Computer* 44.7 (July 2011), pp. 13–16. ISSN: 0018-9162. DOI: 10.1109/MC.2011.220 (cit. on p. 12).

[p173] M. Ott, I. Seskar, R. Siraccusa, and M. Singh. “ORBIT Testbed Software Architecture: Supporting Experiments as a Service”. In: *First International Conference on Testbeds and Research Infrastructures for the DEvelopment of NeTworks and COMmunities*. IEEE, 2005, pp. 136–145. ISBN: 0-7695-2219-X. DOI: 10.1109/TRIDNT.2005.27 (cit. on p. 15).

[p174] A. Ouskel and A. Sheth. “Semantic Interoperability in Global Information Systems. A brief Introduction to the Research Area and the Special Section”. In: *SIGMOD Record* 28.1 (1999), pp. 5–12 (cit. on p. 47).

[p175] J. Pan, S. Paul, and R. Jain. “A survey of the research on future internet architectures”. In: *IEEE Communications Magazine* 49.7 (July 2011), pp. 26–36. ISSN: 0163-6804. DOI: 10.1109/MCOM.2011.5936152 (cit. on pp. 2, 17).

[p176] M. P. Papazoglou. "Cloud Blueprints for Integrating and Managing Cloud Federations". In: *Software Service and Application Engineering*. Springer, 2012, pp. 102–119. DOI: 10.1007/978-3-642-30835-2_8 (cit. on p. 13).

[p177] H. Paulheim and S. Hertling. "Discoverability of SPARQL Endpoints in Linked Open Data." In: *International semantic web conference (Posters & Demos)*. Open Access CEUR, 2013, pp. 245–248 (cit. on p. 101).

[p178] J. Pearsall and P. Hanks. *The new Oxford dictionary of English*. Clarendon Press, 1998 (cit. on pp. 10, 47, 48).

[p179] C. Pedrinaci, J. Cardoso, and T. Leidig. "Linked USDL: A Vocabulary for Web-Scale Service Trading". In: *The Semantic Web: Trends and Challenges*. Springer, 2014, pp. 68–82. DOI: 10.1007/978-3-319-07443-6_6 (cit. on p. 53).

[p180] S. Peroni, D. Shotton, and F. Vitali. "The Live OWL Documentation Environment: A Tool for the Automatic Generation of Ontology Documentation". In: *Knowledge Engineering and Knowledge Management*. Springer, 2012, pp. 398–412. DOI: 10.1007/978-3-642-33876-2_35 (cit. on p. 134).

[p181] L. Peterson and T. Roscoe. "The design principles of PlanetLab". In: *SIGOPS Operating Systems Review* 40.1 (Jan. 2006), p. 11. ISSN: 01635980. DOI: 10.1145/1113361.1113367 (cit. on p. 16).

[p182] K. Phemius, M. Bouet, and J. Leguay. "DISCO: Distributed multi-domain SDN controllers". In: *Network Operations and Management Symposium (NOMS)*. IEEE, May 2014, pp. 1–4. ISBN: 978-1-4799-0913-1. DOI: 10.1109/NOMS.2014.6838330 (cit. on p. 148).

[p183] D. Le-Phuoc, M. Dao-Tran, J. Xavier Parreira, and M. Hauswirth. "A Native and Adaptive Approach for Unified Processing of Linked Streams and Linked Data". In: *The Semantic Web (ISWC)*. Springer, 2011, pp. 370–388. DOI: 10.1007/978-3-642-25073-6_24 (cit. on p. 52).

[p184] D. Le-phuoc, H. Nguyen, M. Quoc, Q. H. Ngo, T. T. Nhat, and M. Hauswirth. "Enabling Live Exploration on The Graph of Things". In: *Semantic Web Challenge*. Vol. 287305. Elsevier, 2014, pp. 1–8 (cit. on p. 147).

[p185] C. Pittaras, C. Papagianni, A. Leivadeas, P. Grosso, J. van der Ham, and S. Papavassiliou. "Resource discovery and allocation for federated virtualized infrastructures". In: *Future Generation Computer Systems* 42 (2015), pp. 55–63. ISSN: 0167-739X. DOI: 10.1016/j.future.2014.01.003 (cit. on p. 54).

[p186] M. Poveda-Villalón, M. C. Suárez-Figueroa, and A. Gómez-Pérez. "Validating Ontologies with OOPS!" In: *Knowledge Engineering and Knowledge Management*. Springer, 2012, pp. 267–281. DOI: 10.1007/978-3-642-33876-2_24 (cit. on p. 134).

[p187] A. Pras, J. Schonwalder, M. Burgess, O. Festor, G. Perez, R. Stadler, and B. Stiller. "Key research challenges in network management". In: *IEEE Communications Magazine* 45.10 (Oct. 2007), pp. 104–110. ISSN: 0163-6804. DOI: 10.1109/MCOM.2007.4342832 (cit. on pp. 4, 45, 48).

[p188] T. Priebe, W. Dobmeier, and N. Kamprath. "Supporting attribute-based access control with ontologies". In: *First International Conference on Availability, Reliability and Security (ARES)*. IEEE, 2006, 8 pp.–472. ISBN: 0-7695-2567-9. DOI: 10.1109/ARES.2006.127 (cit. on p. 145).

[p189] A. Quereilhac, M. Lacage, C. D. Freire, T. Turletti, and W. Dabbous. “NEPI: An integration framework for Network Experimentation”. In: *19th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*. IEEE, 2011, pp. 1–5. ISBN: 978-1-4577-1439-9 (cit. on p. 17).

[p190] S. Quirolgico, P. Assis, A. Westerinen, M. Baskey, and E. Stokes. “Toward a Formal Common Information Model Ontology”. In: *Web Information Systems (WISE)*. Springer, 2004, pp. 11–21. DOI: 10.1007/978-3-540-30481-4_2 (cit. on pp. 46, 47).

[p191] T. Rakotoarivelo, G. Jourjon, and M. Ott. “Designing and Orchestrating Reproducible Experiments on Federated Networking Testbeds”. In: *Computer Networks, Special Issue on Future Internet Testbeds* 63.1 (2014), pp. 173–187. DOI: 10.1016/j.bjp.2013.12.033 (cit. on p. 3).

[p192] T. Rakotoarivelo, M. Ott, G. Jourjon, and I. Seskar. “OMF”. In: *ACM SIGOPS Operating Systems Review* 43.4 (Jan. 2010), p. 54. ISSN: 01635980. DOI: 10.1145/1713254.1713267 (cit. on p. 3).

[p193] S. Ratnasamy, S. Shenker, and S. McCanne. “Towards an evolvable Internet architecture”. In: *SIGCOMM Computer Communication Review*. Vol. 35. 4. ACM, 2005, pp. 313–324 (cit. on p. 2).

[p194] D. Raychaudhuri, I. Seskar, M. Ott, S. Ganu, K. Ramachandran, H. Kremo, R. Siracusa, H. Liu, and M. Singh. “Overview of the ORBIT radio grid testbed for evaluation of next-generation wireless network protocols”. In: *Wireless Communications and Networking Conference*. Vol. 3. IEEE, 2005, pp. 1664–1669. ISBN: 0-7803-8966-2. DOI: 10.1109/WCNC.2005.1424763 (cit. on p. 15).

[p195] D. Recordon and D. Reed. “OpenID 2.0”. In: *2nd ACM workshop on Digital identity management (DIM)*. New York, New York, USA: ACM Press, Nov. 2006, p. 11. ISBN: 1595935479. DOI: 10.1145/1179529.1179532 (cit. on p. 31).

[p196] A. Reddy and S. Kamath. “Research on Potential Semantic Web Service Discovery Mechanisms”. In: *International Conference on Recent Trends in Computer Science and Engineering (ICRTCSE)*. arXiv, Apr. 2012, p. 6. arXiv: 1304.1676 (cit. on p. 53).

[p197] J. D. Rogers. “Internetworking and the Politics of Science: NSFNET in Internet History”. In: *The Information Society* 14.3 (Aug. 1998), pp. 213–228. ISSN: 0197-2243. DOI: 10.1080/019722498128836 (cit. on p. 1).

[p198] F. Ruiz and J. R. Hilera. “Using Ontologies in Software Engineering and Technology”. In: *Ontologies for Software Engineering and Software Technology*. Heidelberg: Springer Berlin Heidelberg, 2006, pp. 49–102. DOI: 10.1007/3-540-34518-3_2 (cit. on p. 48).

[p199] N. Rutar, C. Almazan, and J. Foster. “A Comparison of Bug Finding Tools for Java”. In: *15th International Symposium on Software Reliability Engineering (ISSRE)*. ISSRE ’04. Washington, DC, USA: IEEE, 2004, pp. 245–256. ISBN: 0-7695-2215-7. DOI: 10.1109/ISSRE.2004.1 (cit. on p. 136).

[p200] F. B. Ruy, G. Bertollo, and R. Falbo. “Knowledge-based support to process integration in ODE”. In: *Clei Electronic Journal* 7.1 (2004), pp. 1–18 (cit. on p. 48).

[p201] D. Schwerdel, D. Hock, D. Günther, B. Reuther, P. Müller, and P. Tran-Gia. "ToMaTo - A Network Experimentation Tool". In: *Testbeds and Research Infrastructure - Development of Networks and Communities*. Vol. 90. 1. Springer, 2012, pp. 1–10. ISBN: 978-3-642-29272-9. DOI: 10.1007/978-3-642-29273-6_1 (cit. on p. 15).

[p202] R. Shearer, B. Motik, and I. Horrocks. "HermiT: A Highly-Efficient OWL Reasoner." In: *5th {OWLED} Workshop on {OWL:} Experiences and Directions*. Vol. 432. Karlsruhe: Open Access CEUR, Oct. 2008, p. 91 (cit. on p. 50).

[p203] A. P. Sheth. "Changing Focus on Interoperability in Information Systems:From System, Syntax, Structure to Semantics". In: *Interoperating Geographic Information Systems*. Boston, MA: Springer US, 1999, pp. 5–29. DOI: 10.1007/978-1-4615-5189-8_2 (cit. on p. 47).

[p204] M. Singh, M. Ott, I. Seskar, and P. Kamat. "ORBIT Measurements Framework and Library (OML): Motivations, Design, Implementation, and Features". In: *First International Conference on Testbeds and Research Infrastructures for the DEvelopment of NeTworks and COMmunities*. IEEE, 2005, pp. 146–152. ISBN: 0-7695-2219-X. DOI: 10.1109/TRIDNT.2005.25 (cit. on p. 3).

[p205] E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, and Y. Katz. "Pellet: A practical owl-dl reasoner". In: *Web Semantics: Science, Services and Agents on the World Wide Web* 5.2 (2007), pp. 51–53 (cit. on p. 50).

[p206] L. Smarr and C. E. Catlett. "Metacomputing". In: *Communications of the ACM* 35.6 (June 1992), pp. 44–52. ISSN: 00010782. DOI: 10.1145/129888.129890 (cit. on pp. 1, 11).

[p207] U. Staiger. "MAMSplus-Individuelle Dienste für jedermann". In: *VDE-Kongress 2008*. VDE VERLAG GmbH. München: VDE, 2008, p. 6 (cit. on p. 21).

[p208] V. Stantchev and C. Schröpfer. "Negotiating and Enforcing QoS and SLAs in Grid and Cloud Computing". In: *Advances in grid and pervasive computing*. Springer, 2009, pp. 25–35. DOI: 10.1007/978-3-642-01671-4_3 (cit. on p. 12).

[p209] D. Stavropoulos, A. Dadoukis, T. Rakotoarivelo, M. Ott, T. Korakis, and L. Tassiulas. "Design, architecture and implementation of a resource discovery, reservation and provisioning framework for testbeds". In: *13th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*. Bombay, India: IEEE, May 2015, pp. 48–53. ISBN: 978-3-9018-8274-6. DOI: 10.1109/WIOPT.2015.7151032 (cit. on pp. 17, 139).

[p210] J. Strassner. "DEN-ng: achieving business-driven network management". English. In: *Network Operations and Management Symposium (NOMS)*. IEEE, 2002, pp. 753–766. ISBN: 0-7803-7382-0. DOI: 10.1109/NOMS.2002.1015622 (cit. on p. 18).

[p211] J. Strassner, S. van der Meer, and J. W.-K. Hong. "The Applicability of Self-Awareness for Network Management Operations". In: *Modelling Autonomic Communications Environments*. Springer, 2009, pp. 15–28. DOI: 10.1007/978-3-642-05006-0_2 (cit. on p. 47).

[p212] G. Stumme and A. Maedche. "Ontology merging for federated ontologies on the semantic web". In: *International Workshop for Foundations of Models for Information Integration (FMII)*. Springer, 2001, pp. 413–418 (cit. on p. 49).

[p213] R. Sturm, W. Morris, and M. Jander. *Foundations of service level management*. Vol. 13. Sams Indianapolis, IN, 2000 (cit. on p. 12).

[p214] A. Takefusa, H. Nakada, R. Takano, T. Kudoh, and Y. Tanaka. "GridARS: A Grid Advanced Resource Management System Framework for Intercloud". In: *3rd International Conference on Cloud Computing Technology and Science*. IEEE, Nov. 2011, pp. 705–710. ISBN: 978-1-4673-0090-2. DOI: 10.1109/CloudCom.2011.109 (cit. on p. 13).

[p215] S. Tartir, I. B. Arpinar, and A. P. Sheth. "Ontological Evaluation and Validation". In: *Theory and Applications of Ontology: Computer Applications*. Dordrecht: Springer Netherlands, 2010, pp. 115–130. DOI: 10.1007/978-90-481-8847-5_5 (cit. on pp. 56, 98).

[p216] J. Thones. "Microservices". In: *IEEE Software* 32.1 (Jan. 2015), pp. 116–116. ISSN: 0740-7459. DOI: 10.1109/MS.2015.11 (cit. on p. 6).

[p217] G. Tonti, J. M. Bradshaw, R. Jeffers, R. Montanari, N. Suri, and A. Uszok. "Semantic Web Languages for Policy Representation and Reasoning: A Comparison of KAoS, Rei, and Ponder". In: *The Semantic Web- (SWC)*. Springer, 2003, pp. 419–437. DOI: 10.1007/978-3-540-39718-2_27 (cit. on p. 64).

[p218] A. N. Toosi, R. N. Calheiros, and R. Buyya. "Interconnected Cloud Computing Environments". In: *ACM Computing Surveys* 47.1 (May 2014), pp. 1–47. ISSN: 03600300. DOI: 10.1145/2593512 (cit. on pp. 13, 146).

[p219] M. Tosic and I. Seskar. "Resource specification and intelligent user interaction for federated testbeds using Semantic Web technologies". In: *Computer Networks* 63.1 (Jan. 2014), pp. 84–100. ISSN: 13891286. DOI: 10.1016/j.comnet.2014.01.005 (cit. on pp. 17, 54).

[p220] M. Tosic, I. Seskar, and F. Jelenkovic. "TaaSOR – Testbed-as-a-Service Ontology Repository". In: *Testbeds and Research Infrastructure Development of Networks and Communities (TRIDENTCOM)*. Vol. 44. 1. Springer, 2012, pp. 419–420. DOI: 10.1007/978-3-642-35576-9_49 (cit. on pp. 54, 55).

[p221] K. Toth and M. Subramanium. "The persona concept: a consumer-centered identity model". In: *3rd International Workshop on Emerging Applications for Wireless and Mobile Access (MobEA)*. Budapest: American Academy of Financial Management, 2003, pp. 1–5 (cit. on p. 31).

[p222] C. Tranoris. "Adopting the DSM paradigm: Defining federation scenarios through resource brokers for experimentally driven research". In: *12th IFIP/IEEE International Symposium on Integrated Network Management (IM) and Workshops*. IEEE, May 2011, pp. 1140–1147. ISBN: 978-1-4244-9219-0. DOI: 10.1109/INM.2011.5990574 (cit. on p. 73).

[p223] C. Tranoris and S. Denazis. "Federation Computing: A pragmatic approach for the Future Internet". In: *International Conference on Network and Service Management*. IEEE, Oct. 2010, pp. 190–197. ISBN: 978-1-4244-8910-7. DOI: 10.1109/CNSM.2010.5691310 (cit. on p. 18).

[p224] G. Tselentis, J. Domingue, A. Galis, A. Gavras, D. Hausheer, S. Krco, V. Lotz, and T. Zahariadis. *Towards the Future Internet: A European Research Perspective*. IOS Press, 2009. ISBN: 978-1607500070 (cit. on p. 17).

[p225] J. Urbani, F. van Harmelen, S. Schlobach, and H. Bal. "QueryPIE: Backward Reasoning for OWL Horst over Very Large Knowledge Bases". In: *10th International Semantic Web Conference (ISWC)*. Springer, 2011, pp. 730–745. DOI: 10.1007/978-3-642-25073-6_46 (cit. on p. 81).

[p226] J. Van Der Ham. "A Semantic Model for Complex Computer Networks: The Network Description Language". PhD thesis. Universiteit van Amsterdam, 2010, p. 164 (cit. on p. 54).

[p227] J. van der Ham, P. Grosso, R. van der Pol, A. Toonk, and C. de Laat. "Using the Network Description Language in Optical Networks". In: *International Symposium on Integrated Network Management*. IEEE, May 2007, pp. 199–205. ISBN: 1-4244-0798-2. DOI: 10.1109/INM.2007.374784 (cit. on p. 148).

[p228] J. van der Ham, C. Papagianni, J. Steger, P. Matray, Y. Kryftis, P. Grosso, and L. Lymberopoulos. "Challenges of an information model for federating virtualized infrastructures". In: *5th International DMTF Academic Alliance Workshop on Systems and Virtualization Management: Standards and the Cloud (SVM)*. IEEE, Oct. 2011, pp. 1–6. ISBN: 978-1-4577-1811-3. DOI: 10.1109/SVM.2011.6096468 (cit. on p. 54).

[p229] J. van der Ham, J. Stéger, S. Laki, Y. Kryftis, V. Maglaris, and C. de Laat. "The NOVI information models". In: *Future Generation Computer Systems* 42 (2015), pp. 64–73. ISSN: 0167-739X. DOI: 10.1016/j.future.2013.12.017 (cit. on p. 54).

[p230] L. M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner. "A break in the clouds". In: *ACM SIGCOMM Computer Communication Review* 39.1 (Dec. 2008), p. 50. ISSN: 01464833. DOI: 10.1145/1496091.1496100 (cit. on p. 12).

[p231] M. Veeraraghavan, M. Karol, and G. Clapp. "Optical dynamic circuit services". In: *IEEE Communications Magazine* 48.11 (Nov. 2010), pp. 109–117. ISSN: 0163-6804. DOI: 10.1109/MCOM.2010.5621976 (cit. on p. 146).

[p232] D. Villegas, N. Bobroff, I. Rodero, J. Delgado, Y. Liu, A. Devarakonda, L. Fong, S. Masoud Sadjadi, and M. Parashar. "Cloud federation in a layered service model". In: *Journal of Computer and System Sciences* 78.5 (2012), pp. 1330–1344 (cit. on p. 12).

[p233] R. H. von Alan, S. T. March, J. Park, and S. Ram. "Design science in information systems research". In: *MIS quarterly* 28.1 (2004), pp. 75–105 (cit. on pp. 2, 98, 99).

[p234] P. Vuletić and A. Sevasti. "A Network Management Architecture proposal for the GÉANT-NREN environment". In: *The 26th TERENA Conference, Selected Papers*. Open Access GÉANT, 2010, pp. 1–11. ISBN: 978-90-77559-20-8 (cit. on p. 74).

[p235] H. Wache, T. Voegele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann, and S. Hübner. "Ontology-based integration of information-a survey of existing approaches". In: *Workshop on ontologies and information sharing (IJCAI)*. Vol. 2001. 1. IJCAII, 2001, pp. 108–117 (cit. on pp. 46, 49).

[p236] S. Wahle. "A Generic Framework for Heterogeneous Resource Federation". PhD Thesis. Technische Universität Berlin, 2011 (cit. on p. 18).

[p237] S. Wahle, B. Harjoc, K. Campowsky, and T. Magedanz. "Pan-European testbed and experimental facility federation–architecture refinement and implementation". In: *International Journal of Communication Networks and Distributed Systems* 5.1 (2010), pp. 67–87. DOI: 10.1504/IJCNDS.2010.033968 (cit. on p. 18).

[p238] S. Wahle, T. Magedanz, and K. Campowsky. "Interoperability in Heterogeneous Resource Federations". In: *Testbeds and Research Infrastructures. Development of Networks and Communities*. Springer, 2011, pp. 35–50. DOI: 10.1007/978-3-642-17851-1_3 (cit. on pp. 17, 18).

[p239] S. Wahle, C. Tranoris, S. Denazis, A. Gavras, K. Koutsopoulos, T. Magedanz, and S. Tompros. "Emerging testing trends and the Panlab enabling infrastructure". In: *IEEE Communications Magazine* 49.3 (Mar. 2011), pp. 167–175. ISSN: 0163-6804. DOI: 10.1109/MCOM.2011.5723816 (cit. on p. 17).

[p240] S. Wahle, C. Tranoris, S. Fox, and T. Magedanz. "Resource Description in Large Scale Heterogeneous Resource Federations". In: *Testbeds and Research Infrastructure - Development of Networks and Communities*. Springer, 2012, pp. 100–115. ISBN: 978-3-642-29272-9. DOI: 10.1007/978-3-642-29273-6_8 (cit. on p. 18).

[p241] M. Wieczorek, A. Hoheisel, and R. Prodan. "Taxonomies of the Multi-Criteria Grid Workflow Scheduling Problem". In: *Grid Middleware and Services*. Boston, MA: Springer US, 2008, pp. 237–264. DOI: 10.1007/978-0-387-78446-5_16 (cit. on p. 11).

[p242] A. Wong, P. Ray, N. Parameswaran, and J. Strassner. "Ontology mapping for the interoperability problem in network management". In: *IEEE Journal on Selected Areas in Communications* 23.10 (Oct. 2005), pp. 2058–2068. ISSN: 0733-8716. DOI: 10.1109/JSAC.2005.854130 (cit. on p. 54).

[p243] Q. Yin and T. Roscoe. "Arosa: testbed resource allocation using late-binding and constraints". In: *Poster on the 23rd Symposium on Operating Systems Principles (SOSP)*. Cascais: ACM, Oct. 2011, pp. 1–2 (cit. on p. 25).

[p244] L. Youseff, M. Butrico, and D. Da Silva. "Toward a Unified Ontology of Cloud Computing". In: *Grid Computing Environments Workshop*. IEEE, Nov. 2008, pp. 1–10. ISBN: 978-1-4244-2860-1. DOI: 10.1109/GCE.2008.4738443 (cit. on p. 53).

[p245] E. Yuan and J. Tong. "Attributed based access control (ABAC) for Web services". English. In: *International Conference on Web Services (ICWS)*. IEEE, 2005, p. 569. ISBN: 0-7695-2409-5. DOI: 10.1109/ICWS.2005.25 (cit. on p. 31).

[p246] T. Zahariadis, A. Papadakis, F. Alvarez, J. Gonzalez, F. Lopez, F. Facca, and Y. Al-Hazmi. "FIWARE Lab: Managing Resources and Services in a Cloud Federation Supporting Future Internet Applications". In: *7th International Conference on Utility and Cloud Computing*. 604590. London, UK: IEEE, Dec. 2014, pp. 792–799. ISBN: 978-1-4799-7881-6. DOI: 10.1109/UCC.2014.129 (cit. on p. 20).

[p247] K. Zeng, J. Yang, H. Wang, B. Shao, and Z. Wang. “A Distributed Graph Engine for Web Scale RDF Data”. In: *Proceeding of the Very Large Database Endowment (PVLDB)*. VLDB Endowment, Aug. 2013, pp. 265–276. URL: http://research.microsoft.com/apps/pubs/default.aspx?id=183717 (cit. on p. 52).

[p248] Zhi-Li Zhang, Zhenhai Duan, and Y. Hou. “On scalable network resource management using bandwidth brokers”. In: *Network Operations and Management Symposium (NOMS)*. Vol. 84. 8. IEEE, 2001, pp. 169–183. ISBN: 0-7803-7382-0. DOI: 10.1109/NOMS.2002.1015561 (cit. on p. 146).

[p249] W. Zhu. “Semantic Mediation Bus: An Ontology-based Runtime Infrastructure for Service Interoperability”. In: *16th International Enterprise Distributed Object Computing Conference Workshops*. IEEE, Sept. 2012, pp. 140–145. ISBN: 978-1-4673-5005-1. DOI: 10.1109/EDOCW.2012.27 (cit. on p. 66).

Technical References

[t1] B. Adida and M. Birbeck. *RDFa Primer - Bridging the Human and Data Webs*. Tech. rep. World Wide Web Consortium (W3C), 2008 (cit. on p. 47).

[t2] B. Adida, M. Birbeck, S. McCarron, and I. Herman. *RDFa Core 1.1 - Second Edition*. Working Draft. World Wide Web Consortium (W3C), 2013. URL: http://www.w3.org/TR/rdfa-syntax/ (cit. on p. 49).

[t3] S. Andreozzi, S. Burke, L. Field, S. Fisher, B. Konya, M. Mambelli, J. M. Schopf, M. Viljoen, and A. Wilson. *GFD 147: Glue Schema Specification*. GFD. Open Grid Forum (OGF), 2007 (cit. on p. 52).

[t4] O. Appleton. *Deliverable D3.1: Business models for Federated e-Infrastructures*. Tech. rep. FedSM, 2012 (cit. on p. 20).

[t5] C. B. Aranda, O. Corby, S. Das, L. Feigenbaum, P. Gearon, B. Glimm, S. Harris, S. Hawke, I. Herman, N. Humfrey, N. Michaelis, C. Ogbuji, M. Perry, A. Passant, A. Polleres, E. Prud’hommeaux, A. Seaborne, and G. T. Williams. *SPARQL 1.1 Overview*. Recommendation. World Wide Web Consortium (W3C), Mar. 2013. URL: http://www.w3.org/TR/sparql11-overview (cit. on p. 50).

[t6] D. Beckett, T. Berners-Lee, and E. Prud’hommeaux. *Turtle-terse RDF triple language*. Team Submission. World Wide Web Consortium (W3C), 2008 (cit. on p. 45).

[t7] A. Berglund, S. Boag, D. Chamberlin, M. F. Fernandez, M. Kay, J. Robie, and J. Siméon. *XML Path Language*. Tech. rep. World Wide Web Consortium (W3C), 2003 (cit. on p. 46).

[t8] T. Berners-Lee and D. Connolly. *Notation3 (N3): A readable RDF syntax*. Team Submission. World Wide Web Consortium (W3C), Jan. 1998 (cit. on p. 45).

[t9] D. Bernstein, V. Deepak, and R. Chang. *Draft Standard for Intercloud Interoperability and Federation (SIIF)*. Tech. rep. IEEE P2303, Sept. 2015 (cit. on pp. 6, 13, 133).

[t10] D. Berrueta, J. Phipps, A. Miles, T. Baker, and R. Swick. *Best practice recipes for publishing RDF vocabularies*. Working Draft. World Wide Web Consortium (W3C), 2008 (cit. on p. 134).

[t11] M. Bjorklund. *RFC 6020: YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)*. RFC 6020 (Proposed Standard). Internet Engineering Task Force (IETF), 2010. URL: http://www.ietf.org/rfc/rfc6020.txt (cit. on p. 45).

[t12] B. Blau, C. van Dinther, and M. Behrendt. *State of the Art in Service Modeling Languages*. Tech. rep. Universität Karlsruhe (TH), 2007 (cit. on p. 53).

[t13] R. Brobst, W. Chan, F. Ferstl, A. Haas, D. Templeton, and J. Tollefsrud. *GWD-R: Distributed Resource Management Application API (DRMAA) Specification 1.0*. Tech. rep. Global Grid Forum (GGF), Apr. 2006 (cit. on p. 11).

[t14] V. G. Cerf, Y. Dalal, and C. Sunshine. *RFC 675: Specification of Internet Transmission Control Program*. RFC 675. Internet Engineering Task Force (IETF), 1974. URL: http://www.ietf.org/rfc/rfc675.txt (cit. on p. 1).

[t15] J. Chase. *Authorization and Trust Structure in GENI: A Perspective on the Role of ABAC*. Tech. rep. GENI, July 2011 (cit. on p. 31).

[t16] J. Chase. *ORCA Control Framework Architecture and Internals*. Tech. rep. Duke University, 2009 (cit. on p. 16).

[t17] J. Chase and I. Baldin. *A Retrospective on ORCA: Open Resource Control Architecture*. Tech. rep. Duke University, Dec. 2014, pp. 1–16 (cit. on pp. 16, 80, 139).

[t18] M. Chiosi, D. Clarke, P. Willis, A. Reid, J. Feger, M. Bugenhagen, W. Khan, M. Fargano, C. Cui, H. Deng, et al. *Network Functions Virtualisation–Introductory*. White Paper. European Telecommunications Standards Institute (ETSI), 2012 (cit. on p. 95).

[t19] R. Cyganiak, D. Wood, and M. Lanthaler. *Resource Description Framework (RDF) 1.1 Concepts and Abstract Syntax*. Recommendation. World Wide Web Consortium (W3C), Feb. 2014. URL: http://www.w3.org/TR/rdf11-concepts/ (cit. on pp. 45, 49).

[t20] N. Damianou, A. Bandara, M. Sloman, and E. Lupu. *A survey of policy specification approaches*. Tech. rep. London: Department of Computing, Imperial College of Science Technology and Medicine, 2002, pp. 1–37 (cit. on p. 32).

[t21] B. Dan and R. V. Guha. *Resource Description Framework (RDF) Schema Specification 1.1*. Recommendation. World Wide Web Consortium (W3C), Feb. 2014. URL: http://www.w3.org/TR/1998/WD-rdf-schema/ (cit. on p. 50).

[t22] M. Dyck, J. Snelson, D. Chamberlin, and J. Robie. *XQuery 3.0: An XML Query Language*. Tech. rep. World Wide Web Consortium (W3C), Apr. 2014 (cit. on p. 51).

[t23] R. Enns, M. Bjorklund, J. Schoenwaelder, and A. Bierman. *RFC 6241: Network Configuration Protocol (NETCONF)*. RFC 6241 (Proposed Standard). Internet Engineering Task Force (IETF), 2011. URL: http://www.ietf.org/rfc/rfc6241.txt (cit. on p. 45).

[t24] I. Foster, H. Kishimoto, A. Savva, D. Berry, A. Djaoui, A. Grimshaw, B. Horn, F. Maciel, F. Siebenlist, and R. Subramaniam. *The Open Grid Services Architecture, Version 1.0*. Tech. rep. Global Grid Forum (GGF), 2005 (cit. on p. 12).

[t25] A. Gavras. *Experimentally driven research white paper*. Tech. rep. ICT Fireworks, Apr. 2010 (cit. on p. 2).

[t26] A. Gavras, T. Magedanz, S. Wahle, H. Hrasnica, S. Avéssta, and J.-C. Imbeaux. *Deliverable D2.1: Legal Framework (Version 2)*. Tech. rep. Panlab, 2008, pp. 1–34 (cit. on p. 19).

[t27] GRAAP-WG. *WS-Agreement Negotiation*. Tech. rep. Open Grid Forum (OGF), 2010, pp. 1–53 (cit. on p. 12).

[t28] D. Hardt. *RFC 6749: The OAuth 2.0 Authorization Framework*. RFC 6749 (Proposed Standard). Internet Engineering Task Force (IETF), 2012. URL: http://www.ietf.org/rfc/rfc6749.txt (cit. on p. 31).

[t29] I. Herman, I. Horrocks, and P. F. Patel-Schneider. *OWL 2 Web Ontology Language Document Overview (Second Edition)*. Recommendation. World Wide Web Consortium (W3C), Dec. 2012. URL: http://www.w3.org/TR/owl-overview (cit. on p. 50).

[t30] J. R. Hobbs and F. Pan. *Time Ontology in OWL*. Working Draft. World Wide Web Consortium (W3C), Sept. 2006 (cit. on p. 62).

[t31] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, M. Dean, et al. *SWRL: A semantic web rule language combining OWL and RuleML*. Member Submission. World Wide Web Consortium (W3C), 2004, p. 79 (cit. on p. 50).

[t32] J.-C. Imbeaux, A. Gavras, H. Hrasnica, S. Wahle, O. Martinot, and S. Avéssta. *Deliverable D2.1: Vision for a Pan-European Laboratory (Version 2)*. Tech. rep. Panlab, 2007, pp. 1–52 (cit. on p. 19).

[t33] S. Jeong and A. Bavier. *GENI Federation Scenarios and Requirements*. Tech. rep. GENI: Global Environment for Network Innovations, 2010, pp. 1–16 (cit. on p. 16).

[t34] D. Jordan, J. Evdemon, A. Alves, A. Arkin, S. Askary, C. Barreto, B. Bloch, F. Curbera, M. Ford, Y. Goland, et al. *Web services business process execution language version 2.0*. Tech. rep. OASIS, 2007 (cit. on p. 53).

[t35] B. Khasnabish, C. JunSheng, S. Ma, Y. Meng, N. So, P. Unbehagen, M. Morrow, and M. Hasan. *Cloud Reference Framework*. Internet-Draft draft-khasnabish-cloud-reference-framework-02. Internet Engineering Task Force (IETF), 2014. URL: http://www.ietf.org/internet-drafts/draft-khasnabish-cloud-reference-framework-02.txt (cit. on p. 14).

[t36] G. Lewis. *Basics about cloud computing*. Tech. rep. Pittsburgh, Pennsylvania: Software Engineering Institute Carniege Mellon University, Sept. 2010 (cit. on p. 12).

[t37] F. Liu, J. Tong, J. Mao, R. Bohn, J. Messina, L. Badger, and D. Leaf. *NIST Cloud Computing Reference Architecture*. Tech. rep. NIST, 2011, p. 35 (cit. on p. 13).

[t38] F. Maali, J. Erickson, and P. Archer. *Data catalog vocabulary (DCAT)*. W3C Recommendation. World Wide Web Consortium (W3C), 2014 (cit. on p. 51).

[t39] E. Mannie. *RFC 3945: Generalized Multi-Protocol Label Switching (GMPLS) Architecture*. Standard. Internet Engineering Task Force (IETF), 2004. URL: http://www.ietf.org/rfc/rfc3945.txt (cit. on p. 47).

[t40] A. Marchetti, F. Ronzano, M. Tesconi, and M. Minutoli. *Formalizing Knowledge by Ontologies: OWL and KIF*. Tech. rep. Pisa: rfc6241Institute of Informatics and Telematics of the National Research Council (CNR), 2008 (cit. on p. 50).

[t41] D. Martin, M. Burstein, J. Hobbs, O. Lassila, D. McDermott, S. McIlraith, S. Narayanan, M. Paolucci, B. Parsia, T. Payne, et al. *OWL-S: Semantic Markup for Web Services*. Member Submission. Word Wide Web Consortium (W3C), 2004 (cit. on p. 53).

[t42] F. Maude. *Open Data White Paper-Unleashing the potential*. Tech. rep. London, United Kingdom: HM Government, Cabinet Office, The Stationary Office Limited, 2012 (cit. on p. 147).

[t43] K. McCloghrie, M. Fine, J. Seligson, K. Chan, S. Hahn, R. Sahita, A. Smith, and F. Reichmeyer. *RFC 3159: Structure of Policy Provisioning Information (SPPI)*. RFC 3159 (Proposed Standard). Internet Engineering Task Force (IETF), Aug. 2001. URL: http://www.ietf.org/rfc/rfc3159.txt (cit. on p. 45).

[t44] K. McCloghrie, D. Perkins, and J. Schoenwaelder. *RFC 2578: Structure of Management Information Version 2 (SMIv2)*. RFC 2578 (INTERNET STANDARD). Internet Engineering Task Force (IETF), Apr. 1999. URL: http://www.ietf.org/rfc/rfc2578.txt (cit. on p. 45).

[t45] O. Mehani, G. Jourjon, J. White, T. Rakotoarivelo, R. Boreli, and T. Ernst. *Characterisation of the Effect of a Measurement Library on the Performance of Instrumented Tools*. Report 4879. NICTA, 2011 (cit. on p. 3).

[t46] P. Mell and T. Grance. *The NIST definition of cloud computing*. Tech. rep. NIST, 2011 (cit. on p. 13).

[t47] T. Metsch and A. Edmonds. *Open Cloud Computing Interface–Infrastructure (OCCI)*. Tech. rep. Open Grid Forum (OGF), 2010 (cit. on p. 12).

[t48] NGMN Alliance. *5G White Paper - Executive Version*. Tech. rep. NGMN Alliance, 2014 (cit. on pp. 95, 148).

[t49] G. Nicol, L. Wood, M. Champion, and S. Byrne. *Document Object Model (DOM) level 3 core specification*. Tech. rep. World Wide Web Consortium (W3C), 2001 (cit. on p. 45).

[t50] N. F. Noy and D. L. Mcguinness. *Ontology Development 101: A Guide to Creating Your First Ontology*. Tech. rep. Stanford, 2001 (cit. on pp. 56, 99).

[t51] N. F. Noy and M. A. Musen. *Algorithm and tool for automated ontology merging and alignment*. Presented at the 17th National Conference on Artificial Intelligence (AAAI) and available as Technical Report SMI-2000-0831. Stanford, 2000 (cit. on p. 49).

[t52] Object Management Group. *Ontology Definition Metamodel*. Tech. rep. Object Management Group (OMG), May 2009, pp. 1–334 (cit. on p. 47).

[t53] OIF Carrier WG Guideline Document. *Control Plane Requirements for Multi-Domain Optical Transport Networks*. Tech. rep. OIF, 2010, pp. 1–58 (cit. on p. 147).

[t54] Open Networking User Group (ONUG). *Software-Defined WAN Use Case*. Tech. rep. ONUG, Oct. 2014, pp. 1–10 (cit. on p. 74).

[t55] D. Palma and T. Spatzier. *Topology and Orchestration Specification for Cloud Applications (TOSCA) Version 1*. Standard November. OASIS, Nov. 2013, pp. 1–114 (cit. on p. 6).

[t56] L. Peterson, S. Sevinc, J. Lepreau, and R. Ricci. *Slice-based Federation architecture*. Draft version. GENI, 2009. URL: `http://groups.geni.net/geni/wiki/SliceFedArch` (cit. on p. 3).

[t57] A. Pras and J. Schoenwaelder. *RFC 3444: On the Difference between Information Models and Data Models*. RFC 3444 (Informational). Internet Engineering Task Force (IETF), 2003. URL: `http://www.ietf.org/rfc/rfc3444.txt` (cit. on pp. 4, 44, 45).

[t58] D. Raggett. *Web of Things Framework*. Draft Charter. World Wide Web Consortium (W3C), 2015. URL: `http://www.w3.org/WoT/IG/` (cit. on p. 147).

[t59] T. Rakotoarivelo, G. Jourjon, and M. Ott. *Designing and Orchestrating Reproducible Experiments on Federated Networking Testbeds*. Tech. rep. Sydney, Australia: Technical Report 6111, NICTA, 2012 (cit. on p. 17).

[t60] Rayno Report. *Service Provider Lifecycle Service Orchestration (LSO) Overview and Market Forecast*. Tech. rep. Rayno, 2015 (cit. on pp. 96, 148).

[t61] E. Rissanen. *Extensible Access Control Markup Language (XACML) Version 3.0*. Tech. rep. OASIS, Jan. 2013 (cit. on pp. 12, 31).

[t62] T. Scavo, S. Cantor, and N. Dors. *Shibboleth architecture: Technical overview*. Working draft. Internet2, 2005 (cit. on p. 31).

[t63] M. Schmachtenberg, C. Bizer, A. Jentzsch, and R. Cyganiak. *Linking Open Data cloud diagram*. Tech. rep. LOD Community, 2014. URL: `http://lod-cloud.net` (cit. on p. 51).

[t64] M. Serrano, P. Barnaghi, and P. Cousin. *IoT Semantic Interoperability: Research Challenges, Best Practices, Solutions and Next Steps*. Tech. rep. IERC, 2013 (cit. on p. 53).

[t65] B. Shao, H. Wang, and Y. Li. *The Trinity Graph Engine*. Tech. rep. MSR-TR-2012-30. Microsoft Research (MSR-TR-2012-30), 2012. URL: `http://research.microsoft.com/apps/pubs/default.aspx?id=161291` (cit. on p. 52).

[t66] M. Sporny, G. Kellogg, and M. Lanthaler. *JSON-LD 1.0 - A JSON-based Serialization for Linked Data*. Working Draft. World Wide Web Consortium (W3C), 2013 (cit. on p. 49).

[t67] The GENI Project Office. *ProtoGENI Control Framework Overview*. Tech. rep. GENI, 2009, pp. 1–41 (cit. on p. 87).

[t68] P. Tran-Gia, A. Feldmann, R. Steinmetz, J. Eberspächer, M. Zitterbart, P. Müller, and H. Schotten. *G-Lab White Paper Phase 1-Studien und Experimentalplattform für das Internet der Zukunft*. Tech. rep. German Lab, 2009 (cit. on p. 15).

[t69] F. Travostino, R. Keates, T. Lavian, I. Monga, and B. Schofield. *Project DRAC: creating an applications-aware network*. Tech. rep. Pittsburgh, Pennsylvania: Nortel, Nov. 2004 (cit. on p. 74).

[t70] US Government Cloud Computing Technology Roadmap Volume III. *First Working Draft - Technical Considerations for USG Cloud Computing Deployment Decisions*. Tech. rep. NIST, Oct. 2011 (cit. on p. 13).

[t71] J. van der Ham, F. Dijkstra, R. Lapacz, and J. Zurawski. *GFD.206: Network Markup Language Base Schema*. Tech. rep. Open Grid Forum (OGF), 2013 (cit. on p. 54).

[t72] H. van der Veer and A. Wiles. *Achieving technical interoperability*. White Paper. European Telecommunications Standards Institute (ETSI), 2008 (cit. on p. 48).

[t73] Various. *42010: Systems and Software Engineering, Architecture Description*. Technical Report. ISO/IEC/IEEE, 2011. DOI: 10.1109/IEEESTD.2011.6129467 (cit. on p. 6).

[t74] Various. *Cloud Data Management Interface (CDMI) Std. 1.0*. Tech. rep. Storage Networking Industry Association (SNIA), 2010 (cit. on p. 13).

[t75] Various. *Cloud Infrastructure Management Interface (CIMI) Model and RESTful HTTP-based Protocol*. Tech. rep. Distributed Management Task Force (DMTF), Sept. 2012 (cit. on p. 12).

[t76] Various. *Cloud Standards Coordination; Final Report*. Group Specification GS NFV 002. European Telecommunications Standards Institute (ETSI), 2013, pp. 1–21 (cit. on p. 12).

[t77] Various. *Common Information Model (CIM) Schema*. Tech. rep. Distributed Management Task Force (DMTF), June 2013 (cit. on p. 47).

[t78] Various. *Intercloud Interface Specification Draft (InterCloud Protocol)*. White Paper. Global Inter-Cloud Technology Forum (GICTF), 2012 (cit. on p. 13).

[t79] Various. *Machine-to-Machine Communications (M2M); Functional Architecture*. Technical Specification 102 690 V2.1.1. European Telecommunications Standards Institute (ETSI), 2013, pp. 1–332 (cit. on pp. 53, 147).

[t80] Various. *Network Functions Virtualisation (NFV); Architectural Framework*. Group Specification GS NFV 002. European Telecommunications Standards Institute (ETSI), 2013, pp. 1–21 (cit. on p. 95).

[t81] Various. *Network Functions Virtualisation (NFV); Management and Orchestration V1.1.1*. Group Specification GS NFV-MAN 001. European Telecommunications Standards Institute (ETSI), 2014 (cit. on pp. 74, 75).

[t82] Various. *Part 2: Functional Requirements and Reference Architecture*. Tech. rep. ITU-T Focus Group on Cloud Computing, Feb. 2012 (cit. on p. 13).

[t83] Various. *Technical Requirements for Supporting the Intercloud Networking*. White Paper. Global Inter-Cloud Technology Forum (GICTF), Apr. 2012 (cit. on p. 13).

[t84] Various. *Technical Specification TS-0001-V-2014-08: Functional Architecture Baseline Draft*. Tech. rep. OneM2M, Aug. 2014, pp. 1–297 (cit. on p. 147).

[t85] Various. *The Third Network: Lifecycle Service Orchestration Vision*. Whitepaper February. Metro-Ethernet-Forum (MEF), 2015, pp. 1–18 (cit. on pp. 96, 149).

[t86] Various. *TR139, Service Delivery Framework (SDF) Overview, Release 2.0*. Tech. rep. TeleManagement Forum (TMF), 2008 (cit. on p. 14).

[t87] Various. *X.722: Structure of management information: Guidelines for the definition of managed objects*. Tech. rep. International Telecommunication Union (ITU-T), Jan. 1992 (cit. on p. 45).

[t88] B. Vermeulen and E. al. *Deliverable D2.4: Second Federation Architecture*. Tech. rep. Fed4FIRE, Mar. 2014 (cit. on pp. 124, 125).

[t89] A. Westerinen, J. Schnizlein, J. Strassner, M. Scherling, B. Quinn, S. Herzog, A. Huynh, M. Carlson, J. Perry, and S. Waldbusser. *RFC 3198: Terminology for Policy-Based Management*. RFC 3198 (Informational). Internet Engineering Task Force (IETF), Nov. 2001. URL: http://www.ietf.org/rfc/rfc3198.txt (cit. on pp. 45, 46).

[t90] Y. Xin, I. Baldin, J. Chase, K. Ogan, and K. Anyanwu. *Leveraging Semantic Web Technologies for Managing Resources in a Multi-Domain Infrastructure-as-a-Service Environment*. Tech. rep. Renci, 2014, pp. 1–20. arXiv: 1403.0949 (cit. on p. 54).

[t91] Y. Xin, C. Hill, I. Baldine, A. Mandal, C. Heermann, and J. Chase. *Semantic Plane : Life Cycle of Resource Representation and Reservations in a Network Operating System*. Tech. rep. RENCI, 2013 (cit. on p. 54).

[t92] W. Xu, Y. Jiang, and C. Zhou. *Data Models for Network Functions Virtualization*. Draft. Internet Engineering Task Force (IETF), 2014. URL: https://tools.ietf.org/id/draft-xjz-nfv-model-datamodel-01.txt (cit. on pp. 78, 95, 148).

[t93] X. Yufeng, I. Baldine, J. Chase, and K. Anyanwu. *TR-13-02: Using Semantic Web Description Techniques for Managing Resources in a Multi-Domain Infrastructure-as-a-Service Environment*. Tech. rep. April. RENCI Technical Report Series, 2013 (cit. on pp. 54, 55).

Miscellaneous References

[m1] T. Berners-Lee. *Linked Data*. World Wide Web Design Issues. July 2006. URL: http://www.w3.org/DesignIssues/LinkedData.html (visited on 03/17/2014) (cit. on p. 51).

[m2] T. Berners-Lee. *WWW Past and Future*. Presentation at the Royal Society, London. World Wide Web Consortium (W3C), Sept. 2003. URL: http://www.w3.org/2003/Talks/0922-rsoc-tbl/ (cit. on p. 49).

[m3] M. Chen, S. Moon, and A. Nakao. *Goals and Blueprint for PlanetLab CJK*. Presentation at Conference for Future Internet. Presentation at Conference for Future Internet. Seoul, Korea, 2008 (cit. on p. 14).

[m4] S. Easterbrook. *The difference between Verification and Validation*. 2010. URL: http://www.easterbrook.ca/steve/2010/11/the-difference-between-verification-and-validation/ (visited on 10/23/2015) (cit. on p. 98).

[m5] M. Fowler and J. Lewis. *Microservices*. 2014. URL: http://martinfowler.com/articles/microservices.html (visited on 07/28/2014) (cit. on pp. 6, 67).

[m6] Fraunhofer FOKUS. *Welcome to the FUSECO Playground*. 2015. URL: http://fuseco-playground.org (visited on 12/29/2015) (cit. on p. 131).

[m7] A. Glikson. *FI-WARE: Core Platform for Future Internet Applications*. Poster on the 4th Annual International Conference on Systems and Storage (SYSTOR). Haifa, 2011 (cit. on p. 20).

[m8] E. Knorr and G. Gruman. "What cloud computing really means". In: *InfoWorld* 7.1 (2008), pp. 1–2. URL: http://www.infoworld.com/article/2683784/ (cit. on p. 12).

[m9] R. C. Martin. *The Clean Architecture*. 2012. URL: http://blog.8thlight.com/uncle-bob/2012/08 (visited on 08/19/2013) (cit. on p. 66).

[m10] National Institute of Telecommunications Warsaw. *PL-LAB2020*. 2015. URL: http://pllab.pl (visited on 12/29/2015) (cit. on p. 132).

[m11] L. Pappano. "The Year of the MOOC". In: *The New York Times* 2.12 (2012), p. 2012 (cit. on p. 149).

[m12] M. Potel. *MVP: Model-View-Presenter the Taligent Programming Model for C++ and Java*. Taligent Inc. 1996. URL: http://www.wildcrest.com/Potel/Portfolio/mvp.pdf (visited on 04/23/2013) (cit. on p. 67).

[m13] A. Singhal. *Introducing the Knowledge Graph: Things, not Strings*. Google. 2012. URL: http://googleblog.blogspot.de/2012/05/introducing-knowledge-graph-things-not (visited on 06/17/2014) (cit. on p. 52).

[m14] A. Stolz, B. Rodriguez-Castro, and M. Hepp. *RDF Translator: A RESTful Multi-Format Data Converter for the Semantic Web*. 2013. arXiv: 1312.4704 (cit. on pp. 81, 134).

[m15] D. Talbot. "The Internet is broken". In: *Technology Review* 108.1 (Feb. 2006), p. 11. URL: http://www.technologyreview.com/news/405318/the-internet-is-broken/ (cit. on pp. 2, 14).

[m16] D. Zambonini. *The 7 (f) laws of the Semantic Web*. 2006. URL: http://www.oreillynet.com/xml/blog/2006/06/the_7_flaws_of_the_semantic_we.html (visited on 02/11/2015) (cit. on p. 48).