Towards the Future Internet
G. Tselentis et al. (Eds.)
IOS Press, 2010

doi:10.3233/978-1-60750-539-6-51

Conceptual Design and Use Cases for a FIRE Resource Federation Framework

Sebastian WAHLE[a] and Thomas MAGEDANZ[b] and Anastasius GAVRAS[c]
[a] *Fraunhofer FOKUS, Berlin, Germany*
[b] *Technische Universität Berlin, Berlin, Germany*
[c] *Eurescom GmbH, Heidelberg, Germany*

Abstract. Projects of the Future Internet Research & Experimentation (FIRE) initiative are building an experimental facility that shall serve the needs of Future Internet research and development. The main design principles are virtualization of resources and federation. Federation is a means to meet requirements from Future Internet research that cannot be met by individual testbeds. In particular, to support large scale experiments utilizing heterogeneous resources, a federation of experimental facilities is needed. While several initiatives are currently establishing large scale testbeds, the mechanisms for federating such environments across the boundaries of administrative domains are unclear. This is due to the lack of established and agreed federation models, methods, and operational procedures. In this article we propose a federation model that defines high level conceptual entities for federating resources across administrative domains. A first prototype implementation of the functional components derived from the model has been realized and evaluated. This is demonstrated by the discussion of use cases that depict the flexibility of the proposed approach. The model can guide future testbed developments and harmonize the currently scattered efforts across several FIRE projects in order to establish an agreed resource federation framework. This framework shall be the basis for Future Internet research and experimentation in Europe and provide experimental facility services to academia and industry.

Keywords. FIRE, Federation, Experimental Facility, Model, Use Case, Panlab, Teagle, Future Internet, Experimentation, Testing

Introduction

The pace of network convergence and technology evolution has dramatically decreased infrastructure lifetime – the time an infrastructure remains at the technology's cutting edge – making investments in expensive isolated and specialized infrastructures more risky than they were already. This applies in particular to complex cross-layer and cross-technology infrastructures. For this reason existing and future test and experimental infrastructures increasingly endorse federation principles.

While the concept of *federation* can be applied to a number of fields such as identity management, networking, or trust, in this paper we focus on the federation of testbeds. A federation is understood to be an organization within which smaller divisions have some internal autonomy (Oxford definition). Merriam-Webster defines federal as: (1) formed by a compact between political units that surrender their individual sovereignty to a central authority but retain limited residuary powers of

government; (2) of or constituting a form of government in which power is distributed between a central authority and a number of constituent territorial units.

This concept, clearly stemming from a political background, enables combining infrastructural network resources and services of more than one administrative domain which enhances significantly the utility of the infrastructures. Federation enables access to additional resources increasing scale, or access to resources with unique properties to enrich experiments. Furthermore, combining resources from different communities promotes the collaboration between these and the related research groups [1].

Large research programs address both the development of new Internet architectures and suitable experimental platforms. Examples are the NSF programs GENI (Global Environment for Network Innovations) [2] and FIND (Future Internet Design) [3] as well as the European FIRE initiative [4], [5]. GENI focuses on the deployment of experimental platforms whereas FIND addresses foundational concepts and methods for the Future Internet. In the GENI programme five competing testbed control frameworks are under development (TIED [6],[1], PlanetLab [7], ProtoGENI [8], ORCA [9], ORBIT [10]). In FIRE, several projects are contributing to the experimental facility (e.g. Onelab2 [11], Federica [14], PII [12],[13]). In Asia similar programs have been launched such as AKARI [15] in Japan. Joint Asian activities are carried out under the APAN (Asia-Pacific Advanced Network) [16] initiative, the Asia Future Internet Forum (AsiaFI) [18] as well as PlanetLab CJK (China, Japan, Korea), a joint PlanetLab cooperation by China, Japan, and Korea [17]. An in-depth discussion and comparison between the different control framework approaches for experimental facilities has been published earlier by the authors [19].

1. FIRE Federation Model

In this section we will present the *Base Model* of our framework and will derive from it different levels of "surrender"; the *Central Scenario* and the *Distributed Scenario*.

The *Base Model* follows the definition of federation given in the previous section which uses the concept of surrendering individual sovereignty to a central authority. This understanding is extended for our field to support resource federations *on a par*.

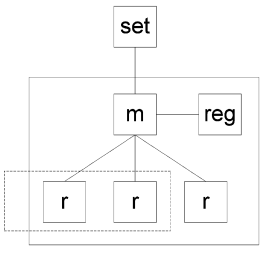

Figure 1. Federation model entities.

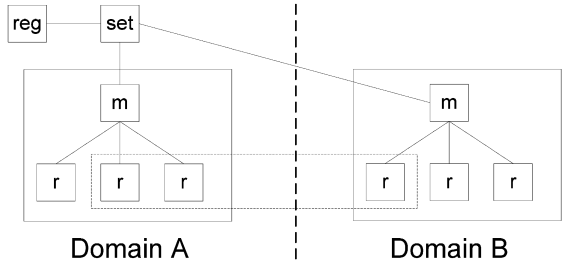

Figure 2. Full surrender scenario.

Independent of the level of surrender similar functional entities must be provided to enable cross-domain and cross-technology federation. The entities are shown in Figure 1 and are defined below. They constitute the proposed *FIRE federation model*.

Resources (r): The model abstracts from concrete resource types. A resource can be anything that can be controlled by software. Examples are: physical and virtual machines, software packages, dedicated hardware such as sensors and routers, as well as abstract constructs such as for example domains, accounts, databases, and identities. Resources may contain other (child) resources.

Domain manager (m): software that controls resources inside an administrative domain. It exposes resource management functionalities at the border of a domain and connects to a resource registry. Supported operations on resources are typically the CRUD (create, read, update, delete) commands for controlling resources via a common interface. Proper security mechanisms and policies need to be supported in order to protect administrative domains from resource misuse.

Registry (reg): holds data records for domain resources. Registries may or may not expose an interface to (external) setup utilities (set).

Creation / setup tool (set): resides within or outside of a domain and communicates with domain managers and registries. Set utilities provide a user interface for the configuration, deployment, and monitoring of virtual resource groupings.

Virtual grouping of resources (dotted rectangle): each administrative domain enables access to a number of resources. Jointly, all administrative domains provide a large pool of resources. Experiments usually require only a subset of the total resources that need to be provided in a certain configuration. This subset may or may not span the border of several domains and is here referred to as a virtual grouping.

Administrative domain (solid rectangle): is typically represented by an organization such as a research institute and provides a collection of resources.

The *Central Scenario* is what we also call the *full surrender* scenario in Figure 2 where the resources committed from domain B can be fully controlled via domain A.

An example of the full surrender scenario is the Panlab federation [12], [26] where all Panlab member domains allow Teagle (section 2.4), the central setup tool (set), to control resources in their domain. It relies on a central registry where resources from all member domains are registered. The advantage of this scenario is that resource representations backed by centrally administered resource models and operational procedures can be simplified. As all central solutions, this approach faces scalability, trust, and availability issues. An implementation of this scenario is described in section 3.1.

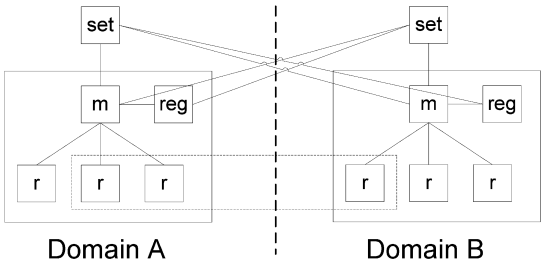

Figure 3. Federation on a par scenario.

The *Distributed Scenario* is what we also call the *federation on a par* scenario in Figure 3 where the participating domains allow the mutual control of resources across the borders of their domains.

Here, the set utilities are allowed access to each other's domain managers and registries. This enables the full scale of resource sharing across organizational boundaries. However, in order to achieve this, a number of agreements need to be in place such as common resource descriptions and management interfaces. Legal and operational procedures are more difficult to realize compared with the central scenario. This scenario has been implemented to federate Panlab resources and resources from a private PlanetLab installation and is described in section 3.2.

Other scenarios that implement something in between the two extreme scenarios explained above are possible and can be applied to meet other requirements and constraints in specific federation contexts. For example only the registries of domains might be shared allowing users to "roam" between domains and use other domain's resources. Such concepts that partly reflect the flexibility of our model are discussed in the literature under the term *federated identity management*. An application is allowing network access to visiting scholars among federated universities where home authentication credentials are used to obtain Internet access at peer institutions.

2. FIRE Resource Federation Prototype Implementation

Based on the presented model, a prototype implementation has been realized and is currently being extended. Here we discuss the model entities (r, m, reg, etc.), their corresponding prototype implementation, and design decisions. Figure 4 shows the components and interfaces of the prototype and their mapping to the model entities.

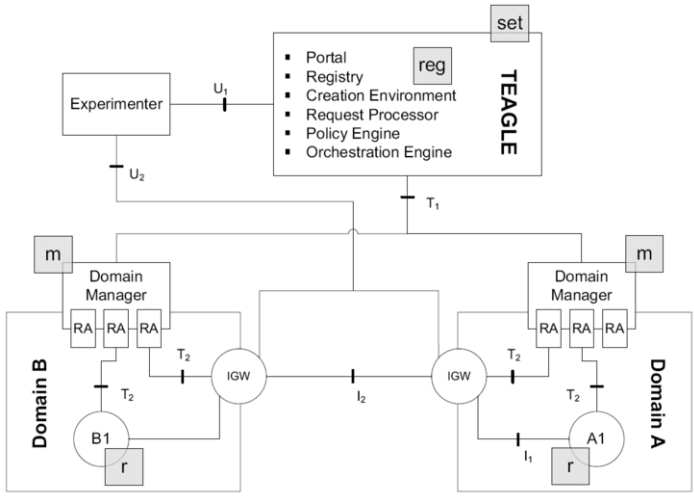

Figure 4. FIRE resource federation prototype framework and mapping to the model entities.

2.1. The Resources and How to Control Them

A resource can be anything that can be controlled by software. In our prototype implementation resources are controlled by resource adaptors (RA) that are plugged in to the *Domain Manager (DM)* framework (see next subsection). RAs are implemented following the DM framework guidelines for pluggable modules. An RA can be seen as

a device driver that supports resource specific communication on interface T2. Examples for T2 communication are Service Provisioning Markup Language (SPML) based messages, the Simple Network Management Protocol (SNMP), or command-line interface (CLI) commands. Any type of resource can be supported by the DM as long as an RA can be implemented and the configuration options can be described and modeled so that the *set* and *reg* entities can handle them. This approach allows us to manage heterogeneous resources that support a variety of different communication mechanisms, reside in different layers and belong to different administrative domains.

Example resources for which RAs have been implemented are *virtual machines* (network, storage, computation parameters can be controlled), *software packages* (Presence Server, MySQL, DNS server, IP Multimedia Subsystem (IMS) components such as Home Subscriber Server, Call Session Control Functions, etc.), the *interconnection component* (IGW) for interconnection of domains, and *abstract constructs* like system users, IMS domains, DNS domains, or database tables.

Further RAs are under development to control (i) sensors, (ii) cloud computing resources, (iii) more advanced interconnection components such as lightpath equipment, (iv) WiFi mesh network resources, (v) 3GPP Evolved Packet Core (EPC) for Next Generation Mobile Networks (NGMN), (vi) monitoring systems, as well as (vii) further software packages that are frequently needed such as mail server, messaging server, and multimedia media software.

2.2. The Domain Manager

This subsection describes our DM prototype implementation that is mapped to the *m* model entity and which has been implemented in Python and Java.

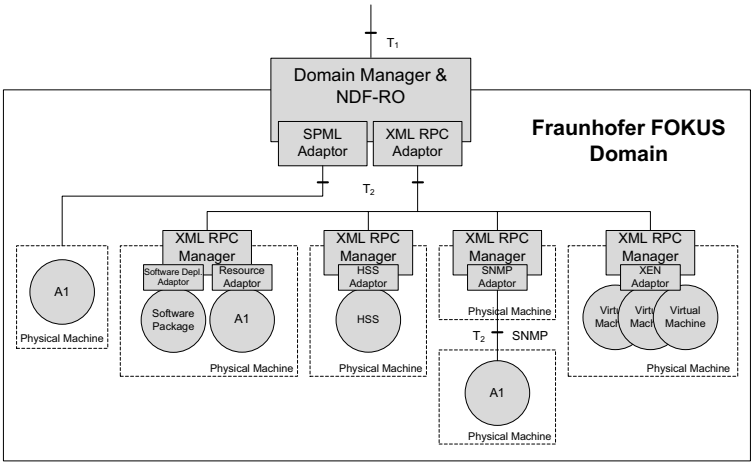

Figure 5. Overview of Fraunhofer FOKUS administrative domain and its domain manager framework [26].

As shown by Figure 5, the DM controls several resources in its domain. This is enabled by several RAs, e.g. XEN adaptor, SNMP adaptor, etc. Through its modular structure, the DM supports multiple resource provisioning schemata and languages, and enables the incorporation of various resources and their native communication mechanisms. For example it is possible to instantiate several virtual machines on a

physical machine and deploy software RAs on the virtual machines themselves in order to control both the container and the actual software resource residing inside the virtual node. Resources that already natively support a provisioning schema, such as SPML, can be directly controlled. We call this concept that supports both pluggable resource adaptors (SNMP, CLI, etc.) as well as pluggable provisioning schemata (SPML, XML-RPC, etc.) *Network Domain Federation Remote Objects (NDF-RO)*.

Generic management operations supported by the resources are exposed as REST (Representational State Transfer) services on interface T1. This allows for a flexible support of heterogeneous resources. Together with the management operation, an XML document is send via T1 to the DM, carrying configuration parameters to be applied to a specific resource.

Next steps in the DM and NDF-RO development will be the realization of a software repository to allow different types and versions of software images to be accessible via our control framework, as well as the incorporation of specialized resources such as more hardware devices, sensors, wireless nodes, etc.

2.3. The Registry

The registry holds data sets related to other federation entities and is designed as an information persistence store for Teagle (section 2.4). The structure of this information follows a generally accepted information model to promote interoperability.

All information in the registry is model based. In particular we have defined a DM model for storing information about registered DMs, a configuration model that underlies the structure of each resource configuration, a reservation model to support resource availability and scheduling, a virtual resource grouping model, as well as a person and organization models to cater for the definition of ownerships. An information model can also be used to manage the legal and operational aspects of federations, such as Service Level Agreements (SLAs) and enforcement of policies. Of particular interest is the resource information model that is necessary for managing the large number of heterogeneous resources and services in the federation. The information model provides the structure and the framework allowing Teagle to catalogue, search, reserve, connect, and configure resources that shall be provisioned within a custom experimental facility setup, which we call *virtual resource grouping*. A common information model facilitates the automatic orchestration of a testbed setup by the *set* entity. The DEN-ng information model [20] was re-used and is currently being adapted to suit the FIRE resource federation framework's requirements.

The necessary data models that are stored in the registry have been automatically derived from the information models following a model driven architecture approach.

2.4. The Extended Creation / Setup Tool Teagle

Our prototype implementation considerably extends the *set* entity of the proposed model. The functions provided by our prototype are collectively called *Teagle* [27]. Teagle is the central search and composition engine of our prototype implementation of the proposed resource federation framework. It provides a web-based interface that allows browsing through the federation's offerings, enables the definition of virtual resource groupings and executes the provisioning thereof. A virtual resource grouping is an isolated network where the experimenter has direct access to the resources and configurations provisioned by Teagle. Each experimenter operates inside its own

virtual resource grouping and has no access to other groupings. Currently, Teagle implements the following functions (see also Figure 4):

- Registry (users, resources, configurations, as described in the previous section)
- Creation Environment (setup and configuration of virtual resource groupings, this is the VCT tool)
- Request Processor (validates configurations and triggers setup execution)
- Orchestration Engine (generates an executable workflow that orchestrates services form different domains to actually provision resources for the experimenter)
- Web Portal (exposes search, configuration interfaces, and general information)

3. Use Cases

3.1. Phosporus & HPDMnet integration

This use case demonstrates the central scenario described in section 1 where a central *set* utility and a central resource registry are used to control the resources offered by participating domains. The involved domains are part of already existing, successful and very advanced networking testbeds: Phosporus [21] and HPDMnet [22].
Phosphorus addresses some of the key technical challenges to enable on-demand end-to-end network services across multiple domains over high speed optical networks. It uses the Harmony service interface. HPDMnet builds a high performance digital media network that provides end-to-end network services across multiple domains like Phosphorus but focuses on high bandwidth media streaming like uncompressed high-definition video streaming. It uses the Chronos interface which is a Harmony derivative.

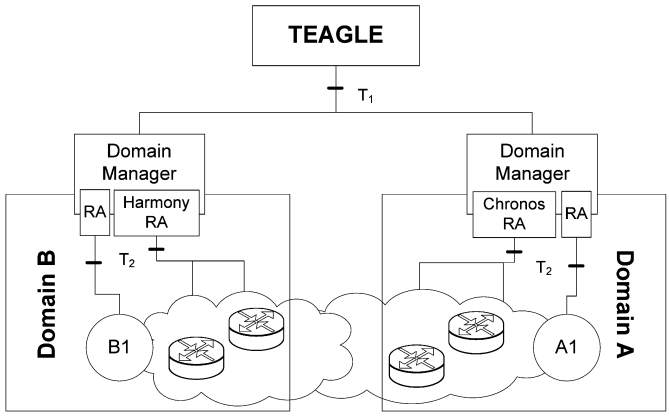

Figure 6. Phosporous (Harmony interface) and HPDMnet (Chronos interface) integration.

The prototype implementation allows establishing a high bandwidth network path across the Phosphorus and HPDMnet testbeds using the central *set* utility Teagle. Both testbeds can create dynamic network services modeled as Teagle resources. The use case has been successfully implemented within 4 weeks in July 2009, demonstrating

that integrating complex existing testbeds for central federation control can be done with reasonable efforts. We implemented RAs for the Harmony (Phosphorus) and the Chronos (HPDMnet) interfaces and demonstrated the setting up of a path with specific bandwidth from Canada to Spain using Teagle and its corresponding control and federation framework.

This use case was easy to implement as no resource description or model mappings were needed. All that was necessary were the implementation of the two RAs and the definition of configuration parameters (start/end time of the reservation, bandwidth, target/source IP address endpoints) as part of a virtual resource in Teagle.

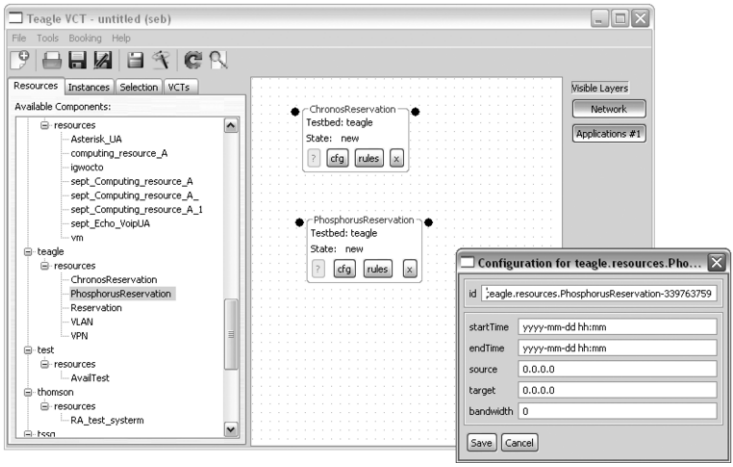

Figure 7. Phosporus and HPDMnet reservations and its configuration options in Teagle.

3.2. PlanetLab & SFA or FIRE-GENI Federation

In this section we discuss the proposed federation scenario *on a par.* Our prototype implements a full federation between a private PlanetLab (at Fraunhofer FOKUS premises, domain A) and a Panlab domain (also at Fraunhofer FOKUS, domain B). Both domains maintain their own registry services and provide a domain manager to allow the other party to provision resources in their domain. As this has been published earlier [23], [24] we only give a brief overview.

3.2.1. Federation Scenario

Users and administrators interact with their local creation tools (Teagle on the Panlab side and slice manager on the PlanetLab side) to create and manage *virtual resource groupings* (VCTs in Panlab terminology, slices in PlanetLab/SFA [25] terminology) which may span both federations. The slice managers delegate look-up requests to their local registry which will in turn query the foreign registry if necessary. Provisioning requests are issued directly to the respective domain managers which forward these requests to their components. The Panlab federation mechanisms have been adapted for this scenario through the implementation of an aggregate manager module and lightweight SFA registry to support the PlanetLab Central (PLC) and Slice-based Facility Architecture (SFA) mechanisms.

As a proof of concept, we have chosen to realize a two-way scenario that shows the implementation's capabilities to provide services towards both domains of the meta-federation:

- A researcher affiliated with PlanetLab uses his *PLC credentials* to access the PlanetLab slice manager. He creates a slice and subsequently requests slivers from a PlanetLab and a Panlab node to be part of his slice.
- A researcher uses her *Panlab credentials* to log into the Teagle portal and create a new VCT (Virtual Customer Testbed), a *virtual resource grouping* in terms of our model. Using the VCT tool she adds a PlanetLab node to her testbed and additionally chooses to deploy a software package onto it.

We implemented a module, called SFAAdapter, which acts as a bridge between the internal semantics and protocols of a Panlab domain manager (denoted as PTM in the following) and the SFA. From the viewpoint of the SFA, this module appears just as any other federation partner, exposing the interfaces of a registry and aggregate manager.

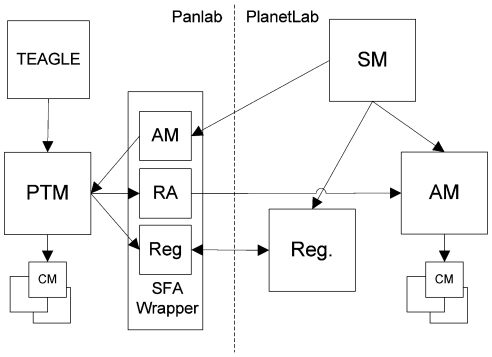

Figure 8. Implementation layout; the arrows indicate the directions in which requests are issued.

Internally, the SFAAdapter comprises three parts, shown in Figure 8 as squares inside the box labeled SFAWrapper. In detail, these elements and their duties are:

Aggregate manager (AM): This entity acts as an aggregate manager as specified by the SFA. It will advertise the resources the PTM can provide when queried by the slice managers of federation partners. Furthermore, it receives provisioning requests which it will translate towards the PTM core to acquire resources.

Resource Adapter (RA): This entity acts as the counterpart to the aggregate manager. It queries foreign aggregate managers about the resources they have to offer and relays the information to the PTM. Subsequently, it issues provisioning requests received from the PTM side towards other aggregate managers. Towards the PTM, it behaves as a native RA. It is responsible for deploying PTM RAs for the acquired resources so they can be managed by the PTM.

Registry (Reg): This module provides a registry service as specified by the SFA. It can be queried by the PTM as well as by remote registries and provides information about the SFA objects which this domain is responsible for. The information is obtained from a database shared between the different parts of SFAAdapter.

The modules providing SFA interfaces (aggregate manager and SFA registry) are based on the original SFA implementation by Princeton. This means that existing SFA front-end clients can interact with them without modifications.

3.2.2. Proof of Concept

This subsection shows a glimpse of how a researcher could access federated services from each side of the federation by walking through a number of steps to set up a simple testing environment. Note that in the example shown, the output is occasionally truncated for brevity and readability.

3.2.2.1. PlanetLab Perspective

A PlanetLab researcher wishes to use his PLC credentials to create a testbed environment via SFA. He has already been added to the SFA registry by a PlanetLab administrator and owns a slice (*plc.fokus.s1*) which, however, does not have any slivers associated yet:

```
#sfi.py resources plc.fokus.s1
<RSpec ...>
   <networks>
      <NetSpec name="plc" .../>
   </networks>
</RSpec>
```

Therefore, he first procures an overview over all available resources while at the same time saving the output for later use:

```
#sfi.py resources –o all.rspec
<Rspec ...>
   <networks>
      <NetSpec name="ptm" ...>
         <nodes>
            <NodeSpec name=”pnode-0.ptm”>
               <net_if>
                  <IfSpec addr="10.0.0.10" .../>
               </net_if>
            </NodeSpec>
         </nodes>
      </NetSpec>
   </networks>
   <networks>
      <NetSpec name="plc" ...>
         <nodes>
            <NodeSpec name=”pln0.plc”>
               <net_if>
                  <IfSpec addr="10.0.0.20" .../>
               </net_if>
            </NodeSpec>
         </nodes>
      </NetSpec>
   </networks>
</Rspec>
```

From this information, he learns that he has two nodes at his disposal, *pln0.plc* from the domain *plc* and *pnode-0.ptm* from the domain *ptm*. He adds them to his slice:

```
#sfi.py create plc.fokus.s1 all.rspec
```

The PlanetLab slice manager will now contact the PTM’s aggregate manager and request it to instantiate a sliver on pnode-0.ptm. The PTM in turn contacts the appropriate RA, asking it to set up a virtual node and to configure it to be PLC compatible. To give simple access to researchers, this means installing corresponding

user credentials and configuring the sliver's SSH server. The researcher can now access the sliver in the same way he would access a sliver acquired from PLC:

```
#ssh -i .ssh/id_rsa focus_s1@pnode-0.ptm sudo su -
```

3.2.2.2. Panlab Perspective

The Panlab researcher mentioned in our use case opens the VCT tool via the Teagle portal. After entering her Panlab credentials, she starts assembling a new testbed comprising an installation of the MySQL software package on a node committed by PlanetLab (see Figure 9). After carefully reviewing her setup, she issues the provisioning requests. Upon receiving these, the PTM contacts the SFA resource adapter which in turn will relay the request towards the aggregate manager of the PlanetLab side.

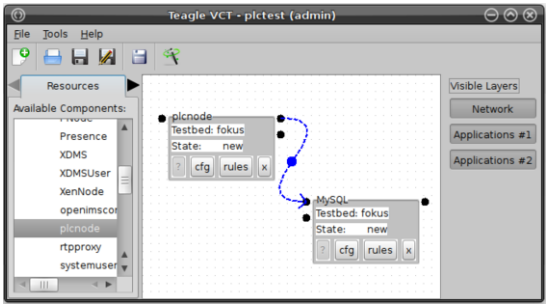

Figure 9. VCT tool view.

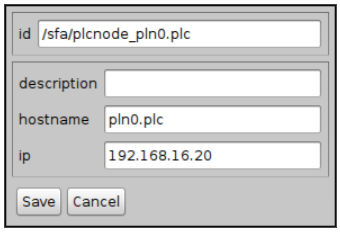

Figure 10. Configuration page for a PLC node in Teagle.

Since a sliver must always be part of a slice, the latter must obviously exist for the provisioning to take place. However, PTM dynamically creates those internally and hides this detail from the user. After the sliver is created by PLC, PTM accesses it and installs a number of resource adapters; among them the so called SoftwareAdapter which it will subsequently use to deploy the requested MySQL package.

The researcher can now bring up the configuration page of the newly provisioned resource to learn some of its details (Figure 10).

4. Outlook & Future Work

The power of federation and the proposed model lies in the generic way resources can be combined, controlled, and used. One part of our future efforts will be to apply it to more application domains such as Cloud Computing, the vertical integration of Service Delivery Platforms, and Smart Cities. While heterogeneity can be overcome with our approach, standardization efforts are needed. Live migration of virtual machines and seamless service migration across administrative domains, needs to the backed, in addition to the underlying technical concepts, by legal, business, and operational procedures and are difficult to realize today. However, many regional data and service providers like governmental authorities, insurance companies, power suppliers, etc. would benefit from federating their data, infrastructure, and services to enable the composition of new applications for the benefit of our society.

References

[1] Ted Faber, John Wroclawski, A federated experiment environment for emulab-based testbeds, Testbeds and Research Infrastructures for the Development of Networks & Communities, International Conference on, pp. 1-10, 2009
[2] National Science Foundation, GENI website: http://www.geni.net
[3] National Science Foundation, FIND website: http://www.nets-find.net
[4] European Commission, FIRE website: http://cordis.europa.eu/fp7/ict/fire
[5] Gavras, A., Karila, A., Fdida, S., May, M., and Potts, M. 2007. Future internet research and experimentation: the FIRE initiative. SIGCOMM Comput. Commun. Rev. 37, 3, 89-92 (2007)
[6] Faber, T., Wroclawski, J., Lahey, K.: A DETER Federation Architecture, DETER Community Workshop on Cyber-Security and Test (2007)
[7] Peterson, L. and Roscoe, T: The Design Principles of PlanetLab. SIGOPS Oper. Syst. Rev. 40, 1, 11-16 (2006)
[8] GENI Project Office, ProtoGENI Control Framework Overview, GENI-SE-CF-PGO-01.4 (2009)
[9] Chase, J et al.: Beyond Virtual Data Centers: Toward an Open Resource Control Architecture, Selected Papers from the International Conference on the Virtual Computing Initiative (ACM Digital Library) (2007)
[10] Ott, M et al.: ORBIT Testbed Software Architecture: Supporting Experiments as a Service, Proceedings of IEEE Tridentcom 2005 (2005)
[11] OneLab project website, http://www.onelab.eu/
[12] Anastasius Gavras, Halid Hrasnica, Sebastian Wahle, David Lozano, Denis Mischler, and Spyros Denazis. Towards the Future Internet - A European Research Perspective, chapter Control of Resources in Pan-European Testbed Federation, pages 67 - 78. IOS Press, ISBN 978-1-60750-007-0 (2009)
[13] Website of Panlab and PII European projects, supported by the European Commission in its both framework programmes FP6 (2001-2006) and FP7 (2007-2013): http://www.panlab.net
[14] Campanella, M.: The FEDERICA Project - A federated infrastructure for Future Internet research, EURESCOM mess@ge, issue 2/2008 (2008)
[15] AKARI project website: http://akari-project.nict.go.jp/eng/index2.htm
[16] Asia-Pacific Advanced Network initiative website: http://www.apan.net/
[17] Maoke Chen, Sue Moon, and Akihiro Nakao. Goals and Blueprint for PlanetLab CJK. Presentation at Conference for Future Internet 2008 PlanetLab BoF, June 19th, 2008, Seoul, Korea.
[18] Asia Future Internet Forum website: http://www.asiafi.net/
[19] Thomas Magedanz and Sebastian Wahle. Control Framework Design for Future Internet Testbeds. e & i Elektrotechnik und Informationstechnik, 126(07/08):274-279, July 2009.
[20] Strassner J.: Policy Based Network Management. Morgan Kaufman, ISBN 1-55860-859-1
[21] S. Figuerola, N. Ciulli, M. de Leenheer, Y. Demchenko, W. Ziegler, and A. Binczewski. PHOSPHORUS: single-step on-demand services across multi-domain networks for e-science. DOI:10.1117/12.746371
[22] HPDMnet website, http://www.hpdmnet.net
[23] Sebastian Wahle, Thomas Magedanz, and Konrad Campowsky. Interoperability in Heterogeneous Resource Federations. In International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TRIDENTCOM 2010). ICST/Springer, 2010. To appear.
[24] Konrad Campowsky, Thomas Magedanz, and Sebastian Wahle. Resource Management in Large Scale Experimental Facilities: Technical Approach to Federate Panlab and PlanetLab. In 12th IEEE/IFIP Network Operations and Management Symposium (NOMS 2010). IEEE/IFIP, 2010.To appear.
[25] Peterson, L., et al. Slice Based Facility Architecture. Princeton, 2007.
[26] Sebastian Wahle, Konrad Campowsky, Bogdan Harjoc, Thomas Magedanz, and Anastasius Gavras. "Pan-European Testbed and Experimental Facility Federation – Architecture Refinement and Implementation." Inderscience International Journal of Communication Networks and Distributed Systems(IJCNDS), Special Issue: Recent Advances in Test-bed Driven Networking Research. To appear.
[27] TEAGLE portal website: http://www.fire-teagle.org