Future Network & MobileSummit 2013 Conference Proceedings
Paul Cunningham and Miriam Cunningham (Eds)
IIMC International Information Management Corporation, 2013
ISBN: 978-1-905824-37-3

Architecture for the Heterogeneous Federation of Future Internet Experimentation Facilities

Wim VANDENBERGHE[1], Brecht VERMEULEN[1], Piet DEMEESTER[1], Alexander WILLNER[2], Symeon PAPAVASSILIOU[3], Anastasius GAVRAS[4], Michael SIOUTIS[5], Alina QUEREILHAC[6], Yahya AL-HAZMI[2], Felicia LOBILLO[7], Florian SCHREINER[9], Celia VELAYOS[8], Albert VICO-OTON[8], Georgios ANDROULIDAKIS[3], Chrysa PAPAGIANNI[3], Okung NTOFON[10], Michael BONIFACE[11]

[1]*Ghent University – iMinds, Gaston Crommenlaan 8 Bus 201, Gent, 9050, Belgium*
Email: wim.vandenberghe@intec.ugent.be
[2]*Technische Universität Berlin, Berlin, Germany*
[3]*National Technical University of Athens, Athens, Greece*
[4]*Eurescom GmbH, Heidelberg, Germany*
[5] *Université Pierre et Marie CURIE, Paris, France*
[6]*INRIA, Sophia Antipolis, France*
[7]*Atos, Madrid, Spain*
[8]*i2CAT, Barcelona, Spain*
[9]*Fraunhofer FOKUS, Berlin, Germany*
[10] *University of Bristol, Bristol, UK*
[11] *IT innovation centre, Southampton, UK*

Abstract: Internet systems are currently too complex to be entirely designed in advance and therefore must be thoroughly evaluated in realistic environments. Experimentally driven research is at the heart of Future Internet Research and Experiment (FIRE) facilities, which target various experimenter profiles, ranging from core Internet communities and sensor networks to clouds and web services. Such facilities exist in relative isolation to the detriment of innovative research ideas that could arise from the mixture of their diverse technologies and resources, and their combined power. Internet research communities can benefit from gaining access to a larger number and variety of resources through a federation of these facilities. To this end, we present an architecture to support such a federation of Future Internet experimentation facilities, based on use cases and requirements from infrastructure owners, as well as services and first line support communities.

Keywords: Future Internet Experimentation Facilities, Federation Architecture, FIRE

1. Introduction

The Future Internet (FI) ecosystem is highly complex and diverse. This diversity lies at the heart of both the challenges and the benefits related to experimentation federation. This ecosystem can be represented as a layered structure including the following three main categories (cf. *Figure 1*):

- **Infrastructure**: the core Internet industry is grounded in the infrastructure and consists of players aiming to provide a range of communications, processing, and storage infrastructures.

- **Services**: building on the core Internet infrastructure, the Services domain aims at providing platforms offering utility functions for composing, controlling, managing, securing, and billing for distributed service-based systems.
- **Applications**: on top of the services, the applications combine and extend available services to deliver functionality to the Internet users themselves.

Each of these layers can be further divided into several research domains. An important challenge is to understand the needs of the different involved research communities. Each community developed its own research challenges, methodologies, and best practices which led to the construction of many dedicated experimentation facilities and tools [1][2][3].

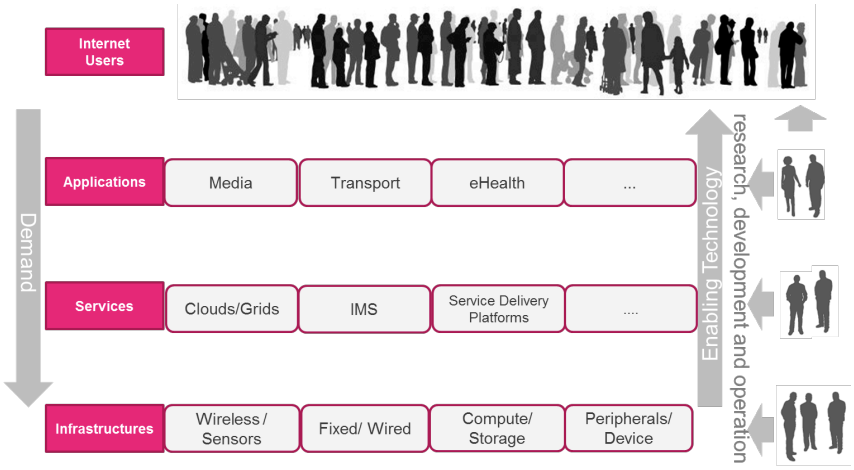

Figure 1: The Future Internet Ecosystem

Aligning all these different visions in a single federation platform would offer many advantages, one of them being the plethora of new technological combinations that can be experimented with. This paves the way for the research and development of new applications and services, which are driven by a clever combination of many innovations across the entire ecosystem. But even in cases where this cross-domain fertilization is of less importance, the participation in a federation can still prove to be beneficial. One example is the reduced cost for experimentation when resources are shared between the federation partners. Another one is the possibility to repeat a same experiment on different facilities, increasing the confidence in the experimental results. A third example is the fact that due to federation, the widened resource offers of individual facilities can more easily attract industrial experimenters.

We present a federation architecture designed to support experimentation in the FI ecosystem. The architecture has been designed taking into account use cases and requirements from infrastructure, services, and first line support communities, prioritizing those aspects considered critical. The architecture tries to fully take advantage of existing tools and mechanisms in the facilities to be federated. It also allows current experimenters of these facilities to keep on using their own tools with a broader range of resources, i.e., no specific tools are imposed for experimentation.

The remainder of this paper is structured as follows: in Section 2 we describe the different aspects of the experiment lifecycle, and enunciate the collected requirements in that regard in Section 3. Based on this analysis, we outline and evaluate possible architectures in Section 4. In Section 5 the chosen architecture is illustrated. Finally, we conclude with a summary and outlook of future work.

2. The experiment lifecycle

To be able to understand the implications of federating FI experimentation facilities, one must have a good understanding of all related functionalities. For this purpose, this section introduces the different aspects of the experiment lifecycle, shown in Table 1.

Table 1: The experiment lifecycle

Function		Description
Resource discovery		Finding available resources across all facilities, and acquiring the necessary information to match required specifications.
Resource requirements		Specification of the resources required during the experiment, including compute, network, storage and software libraries.
Resource reservation		Allocation of a time slot in which exclusive access and control of particular resources is granted.
Resource provisioning	Direct (API)	Instantiation of specific resources directly through the facility API, being the responsibility of the experimenter to select individual resources.
	Orchestrated	Instantiation of resources through a functional component, which automatically chooses resources that best fit the experimenter's requirements.
Experiment control		Control of resource behavior during experiment execution, involving actions to query and modify resource state and their correct sequencing.
Monitoring	Facility monitoring	Instrumentation of resources to supervise the behavior and performance of facilities, allow system administrators or first level support operators to verify that facilities are performing correctly.
	Infrastructure monitoring	Instrumentation of resources to collect data on the behavior and performance of services, technologies, and protocols to obtain measurements in the context of a concrete experiment.
	Experiment measuring	Collection of experimental data generated by frameworks or services that the experimenter can deploy on its own.
Permanent storage		Storage of experiment related information beyond the experiment lifetime, such as experiment description, disk images and measurements.
Resource release		Release of experiment resources after deletion or expiration the experiment.

3. Collected requirements

The objective of this paper is to design an architecture for the federation of FI experimentation facilities that spans diverse research communities. This federation should support all aspects of the experiment lifecycle in a trustworthy manner. In order to design this architecture, more insights are needed into the corresponding requirements. For this purpose; we defined a list of generic federation requirements. We have also queried the infrastructures, services, applications and first level support communities, which participate in the EU FP7 Fed4FIRE project[1], which has motivated this work. The most significant requirements are summarized in *Figure 2*.

The surveys and interactions with the Fed4FIRE partners have revealed some valuable insights into the current trends within facility deployments, such as the current wide spread adoption of the Slice based Federation Architecture (SFA) [4] for resource discovery and provisioning. Other commonalities could also be identified for tools supporting other parts

[1] http://fed4fire.eu

of the experiment lifecycle. Examples in terms of monitoring are the Zabbix[2], Nagios[3], Zenoss, Ganglia, and OML [5] monitoring frameworks. In terms of experiment control, OMF [6], VCTTool [7] / FCI [11], and NEPI [8] are commonly used. However, no dominating technologies could be distinguished for any of these categories.

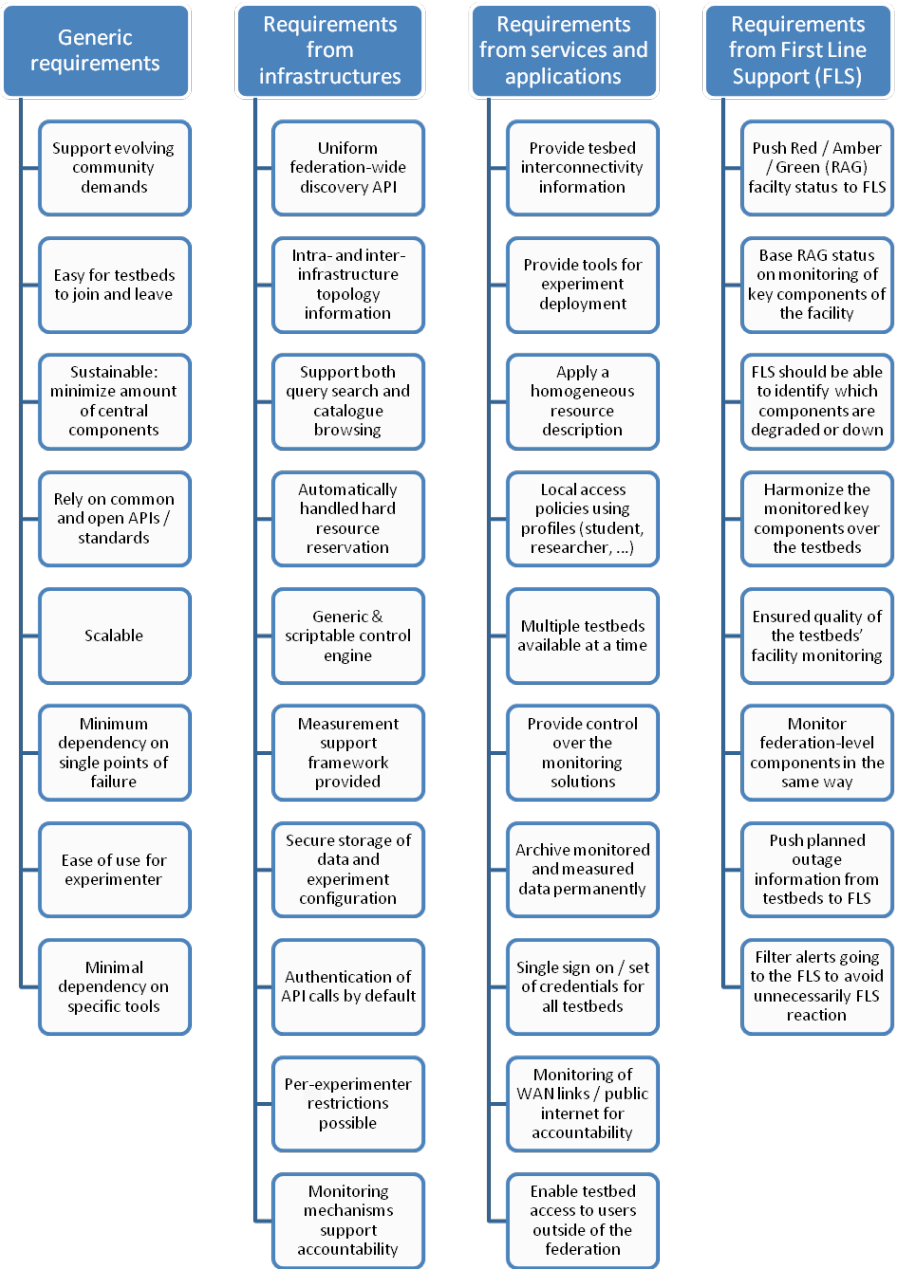

Figure 2: Overview of most significant identified architectural requirements

4. Introduction of possible architectural approaches

Several technical approaches are possible for a FIRE federation architecture [1]. Numerous functionalities need to be in place in order to operate the federation. The adopted architecture could prefer a centralized approach, deploying them as much as possible in a

2 http://www.zabbix.com

3 http://www.nagios.org

few centralized federation services. It could also choose for the opposite approach, distributing these functionalities across the federation as much as possible. Between these two ends of the spectrum, any other approach is also possible. Each of them will be characterized by its own specific advantages and disadvantages. Identifying the most appropriate approach is not trivial, but of high importance for the successful implementation of the envisaged FIRE federation. Therefore, the remainder of this section will focus on the comparison of the four different architectural approaches that are considered to be the most relevant architectural candidates.

4.1. Federated testbed resources under the control of a central management framework

All management software is run by one federation facilitator, which manages all resources of all facilities. In this architecture, the testbeds themselves do not run a testbed management system locally. The PlanetLab testbed [9] is such an example: resources (servers) are distributed on multiple locations without any separate local testbed software framework. The local testbed administrator only boots the nodes once with a specific operating system image, provided by PlanetLab, and then the central PlanetLab software contacts and manages the nodes.

Advantages: As illustrated by the success of PlanetLab, this architecture is easy to deploy and to maintain.

Disadvantages: Several disadvantages make it ineligible for reaching the Fed4FIRE goals in a federation of pre-existing heterogeneous facilities. First of all, this architecture would require all existing testbed management frameworks to be substituted by a single framework, which should cope with all different kinds of resources. To the best of our knowledge, no candidate framework can be identified which can handle all available resource types in all facilities at the moment. Further, a considerable effort would be necessary to customize any central software to do so, while the current testbed management frameworks are in place and can perfectly cope with all existing resources. Thus, the cost of bringing such central software framework in place would be very high. Besides the technical implementation cost, other operational problems should be considered. The imposition on testbed providers to delegate full control to a central management organization is not feasible for facilities that participate and share resources in multiple federations. Agreeing centralized policies for common access and usage across a large and diverse range of organizations and resources is expensive and difficult to establish. The dependency on a central management component is a risk to the entire federation should this component fail due to error or malicious attack.

4.2. Central frontend in the form of a website listing all facilities

All facilities and associated tools remain unchanged, and keep their own user registration procedures, user database, experiment management tools, etc. Nothing is shared among facilities, and the only commonality is a central location, hosting a list of all facilities and their tools.

Advantages: It is the most lightweight and cheap form of federation, requiring practically no effort on behalf of the facilities. An example is the FI-PPP XIPI Repository developed by the EC Infinity Project.

Disadvantages: This is not a real federation, since no simplification of cross facility resource access and control would be achieved (e.g., if an experimenter wants to use

resources on three different facilities, they have to acquire three different accounts and use three different tools/interfaces, which contradicts with the federation requirements). Alternatively, the cost of federation will fall on the backs of experimenters, who would have to implement their own tools to do cross-facility experimentation. This implies a major endeavor, which would have to be repeated for every new combination of federated facilities.

4.3. Homogenous federation running the same testbed management software on all testbeds

The same testbed management software is deployed locally on all facilities, allowing the use of common tools in all facilities. Each facility runs its own instance of the management software and, as such, is independent of central components or organizations. Moreover, each facility has full administration rights over its resources and users, and can deploy and manage images on nodes locally, rendering facility administration fast and efficient.

Advantages: Problems related to the fully centralized management are not present in this architecture.

Disadvantages: This approach still suffers from previously mentioned disadvantages. It requires all existing testbed management software to be substituted with a single framework, which is capable of supporting all different kinds of resources. As already stated, no single framework was identified to be capable of doing so. Additionally, experiment communities that have knowledge of a set of tools and interfaces to access their facilities, would face the challenge of re-training for adopting to new technologies.

4.4. Heterogeneous federation where all testbeds run their native testbed management software

Each facility keeps its current management software, but common interfaces on top of the testbed management software are specified, standardized and made available within the federation. This means that a tool that supports such a common interface would be able to work on any facility, rendering tool standardization possible and, thus, allowing different communities to capitalize from the development efforts of others. It is within the constraints of this approach to define different complementary interfaces for different aspects of the experiment lifecycle (e.g., provisioning and resource control).

Advantages: The proposed approach makes it possible to address federated functionality in a flexible and incremental way. This is essential for a facility of heterogeneous resources where experiment usage patterns will vary significantly depending upon the communities of practice and the research questions being asked. A federation interface can start with limited functionality and extend it later on. This means that the federated functionality can easily grow over time. Additionally, facilities can choose when they implement certain interfaces, and more types of federation interfaces can be specified for the different steps in the experiment lifecycle, eliminating the need of a clean slate approach to the testbed software.

Disadvantages: Specifying federation interfaces and testing interoperability is heavy and costly. Defining common models for naming and resource descriptions that are semantically understandable by client tools and testbed management systems is non-trivial when considering diverse resource characteristics. Common protocols need to be agreed for interacting with resources throughout all activities in the experiment lifecycle, whilst coherence and consistent security models are required that are either agreed or (more likely)

supported through federated identity schemes and access policies. Nevertheless, we believe greater flexibility can be achieved through common models and, in the end, it will reduce the cost against adapting a testbed management framework to all types of resources. The work of standardizing common interfaces can be costly and no guarantees of finding a unique global solution exist. Further, each testbed management software must be extended to implement the specified interfaces, which requires effort.

Based on these characteristics, the heterogeneous federation approach turns out to be the *most suitable* for the characteristics and diversity of FI experimentation facilities under evaluation.

5. Architecture

In the previous section the desired architectural approach was identified. As a result, it is now possible to define the actual architecture. In order not to overload the architectural figures, we split up the detailed architecture discussion in multiple parts grouping main aspects of the experiment lifecycle (cf. Table 1).

5.1. Resource discovery, resource requirement, resource reservation and resource provisioning

The architectural components, which play a role in resource discovery, requirement, reservation, and provisioning, are depicted in *Figure 3*. The architecture considers four layers:

- **Testbed resources:** servers, virtual machines, switches, sensors, software, services, etc.
- **Testbed management:** manages resources, but also the users and experiments of a facility.
- **Broker:** contains services run by 3rd parties or the federation that mediate between the facilities and the experimenters. For example, a broker reservation service that tries to find a match between the resources requested by an experimenter and those offered by the facilities.
- **Experimenter:** tools and interfaces that are used by this experimenter to communicate with the testbed management frameworks, testbed resources, and brokers.

Each software component depicted in *Figure 3* has an interface describing how other components can communicate with it. Identical interfaces are annotated with the same color. Four administrative domains are envisioned: testbed "A", testbed "B", the federation facilitator, and the experimenter. These four domains refer to logical locations, not physical ones. So testbed "A" resources can be distributed over multiple locations (e.g., PlanetLab), but the management of that testbed is under a single administration. The same holds for the federation facilitator: components can be distributed over multiple datacenters, but they are under a single administration entity. This, however, does not exclude the possibility that third parties will arise with additional facilitation functionalities.

Figure 3 shows the adopted distributed architectural design. Components can be identified at the testbed location, the federation facilitator, and the experimenter level (in this case different experimenter client's tools). One of the main design principles of the architecture is that components belonging to the federation facilitator are only intended to make operation and usage of the federation more convenient, but should never be mandatory. Brokers provide 'brokered' access between experimenter tools and the testbeds.

The following components will be provided by a federation facilitator for resource discovery, requirement, reservation, and provisioning:

- **Portal**: A central starting place for experimenters. The portal, using MySlice[4], provides user registration and allows viewing all the available resources in the federation, supporting resource discovery, requirements definition, reservation, and provision operations. Other experimenter standalone tools can provide access to the federation, but MySlice allows experimenters to access it through a single site.
- **Identity provider**: An Identity Provider (IdP) is run by an organization to provide experimenters and services working on their behalf with the means to authenticate themselves within a security domain. Organizations register Experimenters and services with IdPs through trusted registration processes. Testbeds can deploy their own IdP (testbed A) or rely on 3rd party IdPs, for example those operated by the Federation Facilitator. IdPs issue security tokens to subjects that codify signed assertions about them, which can be presented to services at the point of use. The services then enforce policies related to level of trust in the IdP and the subject by associating rights (or declining rights) to each security token.

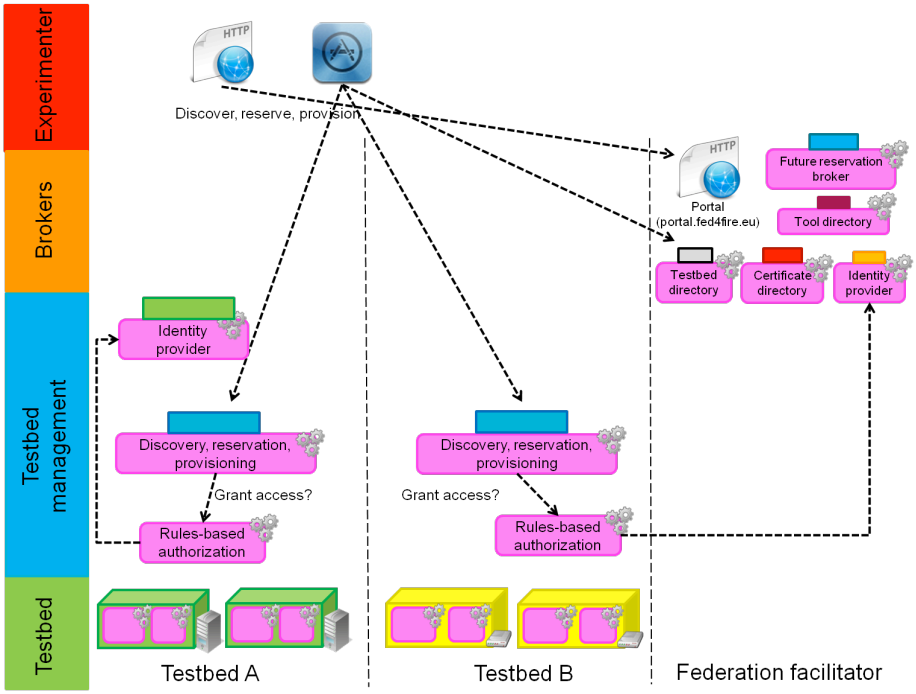

Figure 3: Proposed architecture for discovery, requirements, reservation and provisioning

- **Certificate directory:** Each IdP provides a root of trust for subjects within a security domain. Each testbed must decide which IdP they trust to make assertions about subjects in authentication and authorization decisions. The current architecture relies on a public key infrastructure where the root of trust in each security domain is a Certificate Authority. The Certificate directory provides a mechanism to distribute root certificates for IdPs and avoid the need to manually exchange certificates between testbed providers and experimenters.
- **Testbed directory:** A directory readable by humans and by computers that has an overview of all testbeds in the federation. Two interfaces are provided, a computer readable one and a human readable one. The former relies on the principle of self-describing facilities, where each facility exposes common interfaces for interface for

4 http://www.myslice.info/

discovery, reservation, and provisioning, and advertises information using a same metadata schema. The latter takes the form of a webpage, displaying introductory information about all facilities in the federation.

- **Tool directory:** It gives an overview of available tools for the experimenter, as a webpage were more information regarding FIRE tools is gathered. Users will gain access to the human readable testbed directory through the portal.
- **Reservation broker**: The resource brokering service facilitates reservation of resources, by matching and optimizing requirements imposed by the users. The reservation may span multiple testbeds within the Fed4FIRE federation.

At the testbed side, the following components have been identified:

- **Identity provider:** a testbed either operates an IdP or relies on the IdP operated by the Federation Facilitator.
- **Rules-based authorization**: a facility implements policy enforcement and decision points for authentication and authorization decisions. Authentication is based on security tokens issued and signed by trusted IdPs. Authorization is based on associating rights to subject attributes using rules. Attribute certificates offer one implementation for associating attributes to subject identifies. The benefits of attributes are that subjects can be distinguished based on the affiliation and the experimenter's profile.
- **Common interface**: The existing component(s) responsible for discovery, reservation, and provisioning should expose this functionality through a common interface. SFA is considered to be a suitable choice for such a common interface since this is already a widely adopted standard among considered facilities.

5.2. Monitoring and measurement

The following types of monitoring and measurement are identified (*Figure 4*):

- **Facility monitoring**: Provides monitoring information used for operational performance management (e.g., tracking and analyzing key performance indicators such as utilization targets or experiment costs) and the first level support to see if the testbed facilities are still up and running.
- **Infrastructure monitoring:** Provides monitoring information about the infrastructure resources for experimenters. For instance, providing measurement data about resources such as switch traffic, wireless spectrum or physical host performance if the experimenter uses virtual machines.
- **Experiment measuring:** Measurements generated by services or measurement points deployed directly by the experimenter on the facility on the context of an experiment.

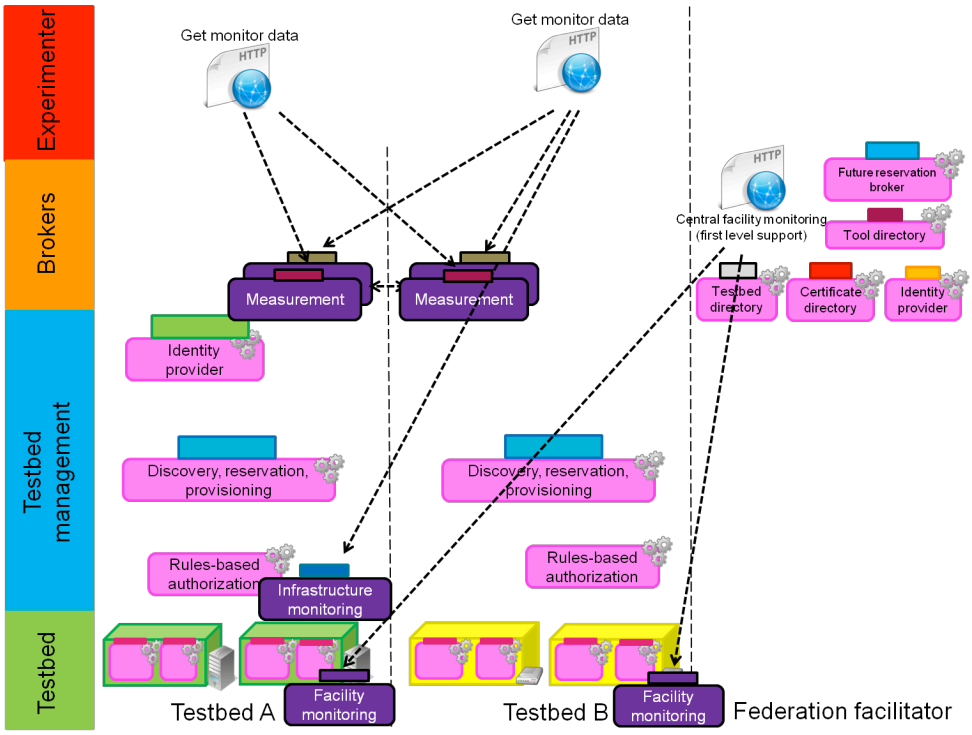

Figure 4: Monitoring and measurement architecture

5.3. Experiment control

A common resource control protocol supported by the management frameworks of all federated facilities, as an additional interface, will allow generic experiment control tools to manage all resources in a uniform manner. An example of such a protocol is the federated resource control protocol (FRCP) [10].

6. Conclusions

We have outlined the challenges and benefits of federated Future Internet (FI) experiment facilities. Based on the principals of the experiment lifecycle the most important requirements in this context have been collected, analyzed, and thoroughly presented. After comparing suitable approaches, a federation architecture designed to support experimentation in the FI ecosystem was proposed as the main result of this paper.
This architecture is conceived to support federation of facilities targeting at different niches of the FI ecosystem, on a global scale, without imposing a large impact on currently deployed tools. At this stage many stakeholders were involved in the definition of the architecture and a first complete implementation is expected in July this year. Based on this practical experience we will be able to assess how well the proposed architecture complies with the imposed requirements in real life, and which more fine-grained technical choices had to be made in order to actually implement it. These aspects will be presented in a follow-up paper.

Acknowledgements

This work was carried out with the support of the Fed4FIRE project ("Federation for FIRE"), an integrated project funded by the European Commission through the 7th ICT-Framework Programme (318389). It does not necessarily reflect the views of the European Commission. The European Commission is not liable for any use that may be made of the information contained herein.

References

[1] A. Gavras et al., “Future Internet Research and Experimentation: The FIRE Initiative”, in ACM SIGCOMM Comp. Comm. Review, Jul. 2007

[2] S. Wahle et al., "Emerging testing trends and the Panlab enabling infrastructure," in Communications Magazine, IEEE, vol.49, no.3, pp.167-175, Mar. 2011

[3] J. Crowcroft et al., “Towards a collaboration and highlevel federation structure for the FIRE Facility”, Jul. 2009

[4] L. Peterson et al., “Slice-Based Federation Architecture”, Tech. rep., Geni, 2010

[5] Manpreet Singh et al., “ORBIT Measurements Framework and Library (OML): Motivations, Design, Implementation, and Features”, in Proceedings of IEEE TRIDENTCOM 2005, Feb. 2005

[6] T. Rakotoariveloet et al., "OMF: A Control and Management Framework for Networking Testbeds", in ACM SIGOPS Operating Systems Review, Jan. 2010

[7] FITeagle, “Future Internet Testbed Experimentation and Management Framework”, fiteagle.org, last accessed on Feb. 2013

[8] C. Freire et al., “Automated deployment and customization of routing overlays on PlanetLab”, in proceedings of TRIDENTCOM, 2012

[9] Brent Chun et al., “PlanetLab: an overlay testbed for broad-coverage services”, in *SIGCOMM Comput. Commun. Rev.* 33, 3, Jul. 2003

[10] C. Dwertmann et al., “Architectural Foundation for Federated Experimental Facilities” (OMF6 Design)”, http://mytestbed.net/projects/omf/wiki/ArchitecturalFoundation2ProtocolInteractions

[11] C. Tranoris, “Adopting the DSM paradigm: defining federation scenarios through resource brokers for experimentally driven research”, in IFIP/IEEE Workshop 2011