Project Acronym	Fed4FIRE
Project Title	Federation for FIRE
Instrument	Large scale integrating project (IP)
Call identifier	FP7-ICT-2011-8
Project number	318389
Project website	www.fed4fire.eu

D4.6 - Report on third cycle developments of the services and applications community

Work package	WP4
Tasks	T4.2, T4.3, T4.4, T4.5
Due date	31/01/2016
Submission date	09/02/2016
Deliverable lead	Mark Sawyer (UEDIN)
Version	2.0
Authors	Mark Sawyer, Gareth Francis, Iakovos Panourgias (UEDIN) Julien Lefeuvre, David Margery (Inria) Thomas Günther (FOKUS) Pablo Sotres (UC) Celestino Alvarez, Lucía Fernández (Adele) Javier Garcia (Atos)
Reviewers	Steve Taylor (IT Innovation)

Abstract	This document describes the work done in WP4 to support cycle 3 of Fed4FIRE.
Keywords	Services, applications, experiment control, monitoring

Nature of the deliverable	R	Report	X
	P	Prototype	
	D	Demonstrator	
	O	Other	
Dissemination level	PU	Public	X
	PP	Restricted to other programme participants (including the Commission)	
	RE	Restricted to a group specified by the consortium (including the Commission)	
	CO	Confidential, only for members of the consortium (including the Commission)	

Disclaimer

The information, documentation and figures available in this deliverable, is written by the Fed4FIRE (Federation for FIRE) – project consortium under EC co-financing contract FP7-ICT-318389 and 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.

Executive Summary

The Fed4FIRE architecture addresses the needs of the experimenter communities in three key areas: experiment lifecycle, measurement and monitoring, and trustworthiness. The architecture is evolving as time progresses and the requirements of the federation (including its sustainability) become better understood. The architecture is being implemented in a series of cycles. Cycle 3 of the architecture is an evolutionary progression from cycles 1 and 2, meaning that the basic ideas have not changed. Cycle 3 (defined in deliverable D2.7) consist of improvements, clarifications and additions to the previous cycles and is intended to ready the test facilities for the sustainability phase of the project.

There are four service and application-focused test facilities: BonFIRE, SmartSantander, FUSECO and FIONA, the last of which joined Fed4FIRE through the Open Calls in M27. Good progress has been made in all the test facilities towards implementing the architectural features. BonFIRE, SmartSantander and FuSeCo offer Advanced Federation level, while FIONA (which joined after Open Call 2 for test facilities) offers Light Federation level. Many of the test facilities have implemented optional features as shown in the tables below.

In addition to progress within individual facilities, work on higher-level tools that will enable federated use of the test facilities has been carried out. A tool for service orchestration across facilities (YourEPM) has been developed, led by Atos. Work has been done, led by Inria, to define a framework for allowing service testbeds to be accessed using Fed4FIRE credentials over http to create service level federation, with an implementation of this for BonFIRE. Together, these two developments make federated use of the facilities easier for experimenters.

Fed4FIRE is an evolutionary project and the process of development is ongoing, driven by the principle of offering benefits and value to experimenters. In some cases this has involved temporary work-arounds to ensure that critical functionality is available. WP4 continues to M48 of the project, with the focus being on support for experiments.

The tables on the following pages give the status of each test facility at the end of cycle 3. This is a snapshot of status, and is likely to change during the final period of Fed4FIRE.

BASIC FUNCTIONALITY	BonFIRE	SmartSantander	FuSeCo	Fiona
Documentation				
Testbed description	OK	OK	OK	In Progress
Basic Tutorial	OK	OK	OK	In Progress
Policies				
Basic experiment w/o extra approval	OK	OK	OK	In Progress
Basic support via google group	OK	OK	OK	In Progress
Facility Monitoring				
First Level Support	OK	OK	OK	OK
Testbed interface for experimenters				
SFA AM approach				
GetVersion support	OK	OK	OK	In Progress
ListResources support	OK	Post Cycle 3	OK	In Progress
Cover full cycle	OK	Post Cycle 3	OK	In Progress
Provide AM URLs to testbed directory	OK	OK	OK	In Progress
Docs: RSpec models description	OK	Post Cycle 3	OK	In Progress
Speaks-For credential support (AM)	Not Planned	Post Cycle 3	Not Planned	Not Planned
Own Testbed interface approach				
Deploy own testbed service	OK	OK	Not Planned	OK
PKCS#12 F4F credentials support	OK	OK	Not Planned	OK
Include API in service directory	OK	OK	Not Planned	OK
Docs: API usage	OK	OK	Not Planned	OK
YourEPM integration (API RAML desc.)	OK	OK	Not Planned	N/A
Speaks-For credential support (service)	OK	OK	Not Planned	N/A
IPv4/IPv6 Connectivity				
SFA AM	OK	OK	OK	OK
Testbed SSL service endpoint	OK	OK	Not Planned	OK
Nodes / Resources	OK	N/A	OK	N/A

ADDITIONAL ENHANCEMENTS	BonFIRE	SmartSantander	FuSeCo	Fiona
Infra. mon. (federation services)	OK	Not Planned	OK	Not Planned
Infra. mon. (per experiment, AM/OML)	In progress	OK	in progress	Not Planned
Advanced reservation	In progress	N/A	Not Planned	Not Planned
SLA	Not Planned	Not Planned	cycle 2 SLA	Not Planned
Experiment control				
images with OMF	OK	Not Planned	OK	Not Planned
AMQP server	Not Planned	Not Planned	Not Planned	Not Planned
PDP	OK	Not Planned	OK	Not Planned
Layer 2 connectivity				
VLAN stitching	OK	Not Planned	Not Planned	Not Planned
Tunnels (in RSpec)	Not Planned	Not Planned	Not Planned	Not Planned

Acronyms and Abbreviations

AM	Aggregate Manager
API	Application Programming Interface
CA	Certificate Authority
EC	Experiment Controller
FLS	First Level Support
FRCP	Federated Resource Control Protocol
GENI	Global Environment for Network Innovations
NEL	SmartSantander Native Experiment Layer
NEPI	Network Experiment Programming Interface
OCCI	Open Cloud Computing Interface
OMF	Control and Management Framework
OML	OML Measurement Library
OMSP	OML Measurement Stream Protocol
QoS	Quality of Service
PDP	Policy Decision Point
RSpec	Resource Specification
SEL	SmartSantander Service Experiment Layer
SFA	Slice Federation Architecture
SNAA	Sensor Network Authentication and Authorization
VM	Virtual Machine
WSN	Wireless Sensor Network
XMPP	Extensible Messaging and Presence Protocol

Table of Contents

1 Introduction

This document is the report on the implementation of cycle 3 of the Fed4FIRE architecture as defined in previous deliverables in the service and application-focused test facilities BonFIRE, SmartSantander, FUSECO and FIONA, the last of which joined Fed4FIRE through the Open Calls.

Cycle 3 of the architecture is an evolutionary progression from cycles 1 and 2, meaning that the basic ideas have not changed. The implementation and deployment with the test facilities has therefore also followed an evolutionary path. As with cycles 1 and 2, test facilities have had some freedom to prioritize of which features to deploy based on their usefulness to the end-users (experimenters) and the cost and technical difficulty of implementation. In some cases certain features of the architecture make no sense to deploy because of the nature of the test facilities. This is to be expected in a heterogeneous collection of test facilities, and particularly in WP4 which involves service and application oriented test facilities which tend to be offer diverse features.

Although the third development cycle is the last one planned in Fed4FIRE, WP4 continues to the end of the project, with the test facilities supporting the work of experimenters where feasible with the effort remaining.

1.1 Experiment lifecycle management

Fed4FIRE chose Slice Federation Architecture (SFA) as a means to specify, discover and provision resources, and the OMF Control and Management Framework for controlling resources. Work continued in cycle 3 to bring the WP4 facilities in line with these standards with the objective of seamless interworking of test facilities.

In addition to this work, a higher level tool, YourEPM has been introduced. YourEPM (Your Experiment Process Manager) is a tool introduced in the third cycle of Fed4FIRE designed to allow the orchestration of high level Application Services offered by testbed providers. This tool is based on the Activiti Business Process Management (BPM) [19] engine for process design and extends it to y

YourEPM offers experimenters to access and combine different Application Services of the federation to create more complex experiments (experiment processes in the context of the tool) in a simple way by using BPM workflow diagrams. Once defined, experiment processes can be deployed in YourEPM to be started at any instant and as many times as desired. Experiment processes can also y

A REST proxy and RAML support are also integrated in YourEPM. This provides compatibility with the REST services of the federation. For Application Services to be used by YourEPM, they need to be included in the Service Directory and described using RAML [20] specification. Moreover, YourEPM supports speaks-for credentials to access some of the available services that require that y

Multitenancy is also supported by YourEPM, which allows experimenters from different organizations to have a complete isolated work environment within the tool. Members of an organization (i.e. tenant) will not be able to view nor execute the experiment processes defined by y

Further documentation about YourEPM can be found in deliverable D5.6 - Report on third cycle developments regarding experiment workflow tools and lifecycle management.

1.2 Measurement and monitoring

Work has continued on measurement and monitoring mainly to support federation services (currently SLA management and the reputation service).

1.3 Trust and security

Following a plenary meeting in September 2015, different partners have agreed on a high level view of how Fed4FIRE credentials could be used to access service testbeds over http. This creates a service level federation.

1.3.1 Context

We want Fed4FIRE users and tools to be able to access, in an authenticated way, services in Fed4FIRE that are exposed over http[s] using a service API. This would be used in particular for the work on Service orchestration (YourEPM). The key points when considering a solution here is that these services can be REST services, using the GET, POST, PUT, DELETE verbs of http. As DELETE and GET do not usually carry a payload, credentials can only be carried using:

- query parameters (base64 encoded, using urlsafe variant) Around 4800 bytes longs
- cookies
- http headers
- https client authentication

As the payload to carry is too big for cookies or http headers, and that https client authentication is already used as a trust anchor to identify the requester, the decision was made to use query parameters, despite the risk that long query parameters be flagged as suspect by firewalls, proxies or client tools. Indeed, the original RFC for query parameters (RFC 2616) has a note in section 3.2.1 on query limits saying “servers ought to be cautious about depending on URI lengths above 255 bytes”. RFC 7230, finalized in 2014, in section 3.1.1 says “Various ad hoc limitations on request-line length are found in practice. It is RECOMMENDED that all HTTP senders and recipients support, at a minimum, request-line lengths of 8000 octets”. It can therefore be expected that modern appliances and servers on the path between the client and the server would allow query with URI including one speaks-for parameter. If it is not the case, the query will be rejected, and the service not available to some clients. This analysis gives us enough confidence that a working implementation can be made for the simple use-cases at the very least.

Access to a service API would therefore be made using https client authentication to convey the certificate of the caller (user, or service), and the speaksfor query parameter (as defined in http://groups.geni.net/geni/wiki/TIEDCredentials as part of SFA) for the array of credentials to point to the final user for which the request is made. Documentation on how to build a speaks-for credentials for a service is available online (http://doc.fed4fire.eu/speaksfor.html). A speaks-for credential is a signed document using the XML Signature specification where a user A signs a document specifying user B (for example a federation tool instance) can act on behalf of user A.

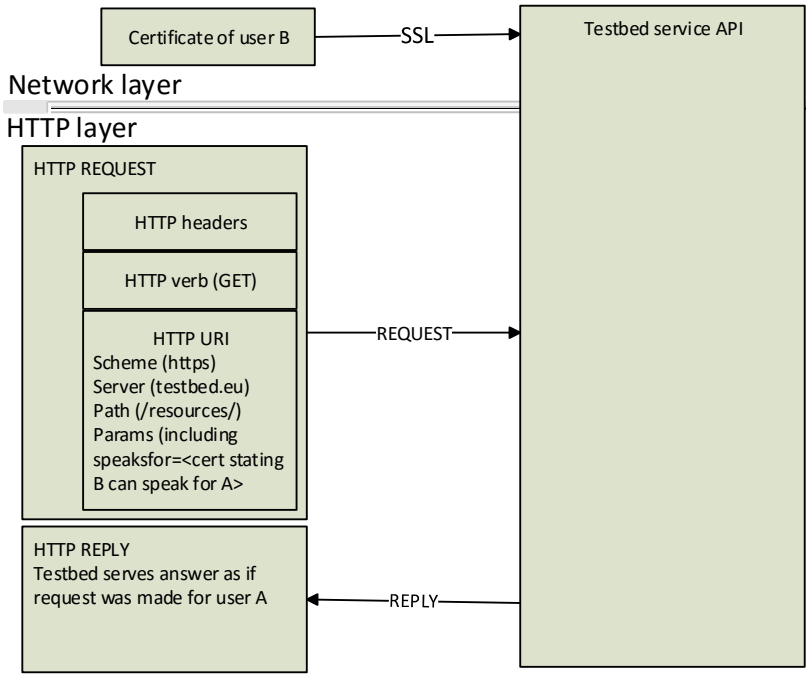

Figure 1: User B (such as federation tool) acting on behalf of user A (an experimenter) using a speaksfor credential)

1.3.2 Federation proxy to service testbeds

It is anticipated that in the future, testbeds federating with Fed4FIRE may benefit from being able to do so using a Fed4FIRE proxy. This proxy will check the credentials of the caller, and forward the call to the native API of the testbed. For example,

GET https://fed4fire-proxy.bonfire-project.eu/locations?speaksfor=base64encodedSpeaksforcredential

will, after having checked the ssl client certificate, call

GET https://api.bonfire-project.eu/locations

To convey information on the user that is requesting the service, it is anticipated that HTTP headers are added to the call to the native API:

- X-Fed4FIRE-caller-urn:
- X-Fed4FIRE-caller-email:
- X-Fed4FIRE-user-urn:
- X-Fed4FIRE-user-email:
- X-Fed4FIRE-url-anchor:

The caller-urn and caller-email parameters are extracted from the ssl credential, and user-urn and user-email are extracted from the chain of speaks-for to identify the final user for which the query is made. Url-anchor is used so that the final API can be made aware of the root url from which the API is

made available on the proxy.

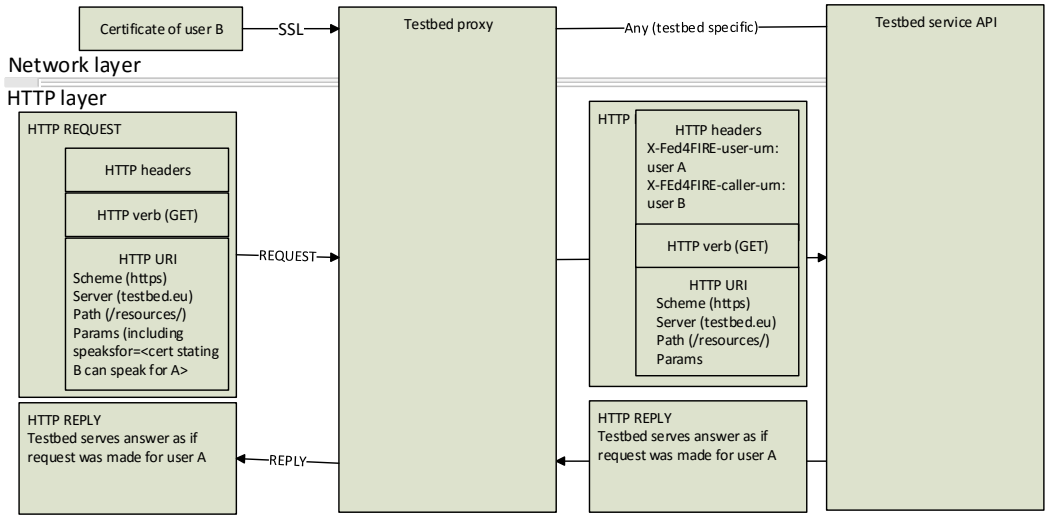

Figure 2: User B (such as a federation tool) acting on behalf of user A (an experimenter) via a testbed proxy implemented in the same domain as the testbed service API. The proxy is doing the authentication and the testbed service API is implementing the service, with the knowledge it is doing so at the request of user B for user A.

1.3.3 Towards ad-hoc federation

If the testbed proxy of Figure 2 is moved out of the administrative domain of the testbed it is acting for, it is not difficult to see that it could act as a proxy for more than one testbed. Indeed, the code of every testbed proxy will be very similar, only differing in the mechanism to securely connect between the testbed proxy and the testbed service API. It is therefore tempting to go one step further, with only one proxy service in the federation. To implement this vision, a means of identifying the testbed that the proxy is working for, and a means of securing the link between the proxy and the native API to avoid opening a security hole are needed.

We envision writing a service associating every testbed with a secured means of connecting the federation proxy to the testbed service API. This association could be created dynamically by testbed owners, creating sub-proxies identified by the first component of urls used to query them. For example, queries to https://proxy.fed4fire.eu/bonfire/someurl would be redirected to https://api.bonfire-project.eu/someurl using a secured link, with the parameters of this secured link defined per testbed.

The key to such a service is to be able to write a federation-wide proxy that can create secured links to the testbed's service API dynamically. There are different means of achieving this:

- Publish the IP and a client certificate the federation proxy is able to use.
- Associate a login and password to use to connect to a testbed when forwarding requests to it, and carry those over using HTTP auth mechanisms (basic or client certificate over ssl).
- Any combination of the above.

It is therefore possible to create an API endpoint in the federation where any testbed could come to

and register a testbed name, a testbed service API endpoint and a mechanism to secure the connection from the federation proxy to that endpoint, and immediately have a working proxy to allow usage of its native APIs by anyone in the federation, provided its service API is configured to trust the proxy Federation service.

1.3.4 Implementation

As the vision presented here was agreed upon late in cycle 3, it was not possible to aim at implementing a common proxy for all testbeds in the context of Fed4FIRE. The base mechanism will be demonstrated on BonFIRE (see section 2.5.3) and Smartsantander during the lifetime of the project. It is also available using Tengu (http://tengu.intec.ugent.be/v1/)

2 BonFIRE

2.1 Facility description

BonFIRE is a federation of test facilities at different locations run by the BonFIRE Foundation. It provides cloud-computing facilities (virtual machines) at EPCC and Inria, together with the network emulation capability from the Virtual Wall run by iMinds.

BonFIRE consists of three systems: an integration system for component development; a qualification system for package testing; and a production system. Experimenters only access the production system.

2.2 Cycle 3: Overview

Efforts to implement the cycle 3 features of Fed4FIRE have focussed on:

- Providing an SFA gateway that will bring BonFIRE into line with the chosen method of managing resources
- Further integration of monitoring with federation services and with the BonFIRE SFA implementation
- Implementing authorisation mechanisms to enable service level federation (see also section 1.3).

2.3 Cycle 3: Experiment lifecycle management

2.3.1 Description, Discovery and Provisioning

As indicated in previous deliverables, the chosen mechanism in Fed4FIRE for describing, discovering and provisioning resources is the Slice Federation Architecture (SFA) standard (specifically GENI AM API v3).

In previous cycles, it was explained that BonFIRE concepts could not be mapped directly to SFA concepts, hence developing a SFA gateway was a necessary step for BonFIRE to adopt the Fed4FIRE standard for resource management.

In cycle 2 a first tentative implementation of an SFA gateway to BonFIRE was developed using SFAWrap[8], which allowed to show a working proof of concept and revealed that SFAWrap did not provide the expected framework to build a SFA gateway for BonFIRE.

Further investigations on SFA ecosystem showed other approaches, in particular the GRAM[9] project which was adapting GCF[10] to provide a SFA gateway to Openstack[11]. Discussions with members of GENI[12] who developed GCF provided great insight and by the end of cycle 2 a working proof of concept of a SFA gateway to BonFIRE was developed based on GCF, leveraging earlier developments made toward SFAWrap.

In cycle 3 efforts were made toward designing the SFA gateway to BonFIRE in a way enabling to contribute back as much as possible to the GCF project, which in the meantime was established as an opensource software under the name Geni-Tools[13]. That way, newly federated test facilities without SFA native support will be able to leverage work done for BonFIRE in their process of becoming fully

federated members of Fed4FIRE.

Hence two tools were developed, GCF-Fed4FIRE, a GCF fork with enhanced pluggability, and GCF-BonFIRE a plugin for GCF-Fed4FIRE implementing a SFA gateway to BonFIRE. All GCF-Fed4FIRE improvements were contributed back to Geni-Tools, and GCF-Fed4FIRE received further development brought to Geni-Tools, such as an improved support of GENI AM API v3 standard.

During M31 of the project stable versions of GCF-BonFIRE and GCF-Fed4FIRE were made available online as a SFA gateway to BonFIRE production facilities, enabling the use of all GENI AM API v3 calls (GetVersion, ListResources, Allocate, Renew, Delete, Provision, Describe) for describing, discovering and provisioning resources on BonFIRE, including their future reservation (a concept introduced in BonFIRE 4.1).

Some integration efforts were made with other Fed4FIRE work packages in order to improve Fed4FIRE user tools (jFed, portal/myslice, YourEPM …) support of the SFA Gateway to BonFIRE in order to provide a smooth experience to the end user.

Further work was done to enable on the fly user registration to BonFIRE for Fed4FIRE users using demo accounts. This feature in planned for deployment on BonFIRE production environment by the end of cycle3.

2.3.2 Experiment Control

Fed4FIRE adopted OMF as the federation wide experiment control, measurement and management solution. OMF is based on an Experiment Controller, one or more Resource Controllers and a communications server which facilitates the exchange of messages between the various Controllers.

OMF supports two kinds of communication servers: XMPP (with pubsub) and AMQP.

In previous cycles, a move from XMPP to AMQP was decided for Fed4FIRE, as OMF usage required more performance from its message passing backend than a federation of XMPP servers could provide. AMQP while being a more centralized environment, allowed great improvement on performance and security.

On BonFIRE, the OMF Resource Controllers (and Experiment Controller if the user decides to use BonFIRE as the hosting infrastructure) are installed and configured in Virtual Machine images. In order to keep supporting images prepared during previous cycles BonFIRE kept its existing XMPP servers running, while allowing newly prepared images to use AMQP.

2.4 Cycle 3: Measurement and monitoring

Most work for measurement and monitoring was completed in earlier cycles. The main work during cycle 3 has been to integrate the work from previous cycles with the SFA work undertaken and described above

2.4.1 Facility Monitoring

The FLS services take advantage of the BonFIRE SFA AM v3 and are able to retrieve the current version of the SFA AM and the available (free) resources of the BonFIRE infrastructure.

2.4.2 Infrastructure Monitoring

In order to facilitate clients querying the database (other Federation services and 3rd parties), BonFIRE adds the Fed4FIRE sliver ID and the BonFIRE Experiment ID to each metric. Thus, Federation services can query metrics based on their sliver ID or BonFIRE Experiment ID.

BonFIRE provides metrics to Central Federation Services and to experimenters using BonFIRE hosted resources.

2.4.3 Experimenter Infrastructure Monitoring

In cycle 3, BonFIRE integrated the experimenter infrastructure monitoring service with the SFA AM. Experimenter infrastructure monitoring metrics are exposed through the SFA AM advertisement rspec. The advertisement rspec can be parsed either by a human or a machine. Experimenters are then able to specify metrics and their destination when they request resources using SFA allocate call. The SFA AM propagates the experimenter's request to the VM level. The local service, which runs on each VM, collects the metrics and propagates them either to a local or remote destination.

2.5 Cycle 2: Trust and security

2.5.1 User registration

In cycle 3, BonFIRE has implemented dynamic creation of previously unknown users. Up to cycle 2, users wishing to use BonFIRE through BonFIRE's aggregate manager (SFA gateway) needed to go to BonFIRE's portal to create an account using their SFA credentials and request to join a group with usage quota on BonFIRE. Groups where here created for all Open Call users when access was granted to Fed4FIRE.

With this new development, users not having registered through the BonFIRE portal are now dynamically registered upon first use on BonFIRE's aggregate manager API and attributed initial quota (enough quota to run through the tutorial and a bit more for initial evaluation of BonFIRE's usefulness for their use-case). A delay persists between this dynamic creation and the time at which they can create their first allocation at the end of cycle 3, but it will be removed shortly. Morevover, as no SFA mechanism exist to request usage quota, they will still need to use an out-of-band mechanism to request that their account be linked to a group with quota. Finally, ssh key handling for these dynamically created users still needs some work.

2.5.2 PDP

BonFIRE adopted and implemented the Control and Management Framework (OMF). BonFIRE users can use OMF to control applications running on BonFIRE compute nodes. However, the OMF solution does not provide authorization and authentication mechanisms. Thus, an attacker can abuse an OMF based testbed. Fed4FIRE's solution was to design and develop Policy Decision Point (PDP). The PDP solution is discussed in D7.2 [6].

During cycle 3, BonFIRE tested and evaluated the PDP implementation. A number of issues were found as a result of stress testing, and these were shared with WP7. These issues are listed as follows.

- In BonFIRE, it is common for an OMF RC not to be ready immediately the resources are allocated. When the resources are allocated, the PDP is informed, and it immediately tries to subscribe to its AMQP subject. If the OMF RC is not available when the PDP tries to subscribe, the PDP "forgets" the OMF RC. The only solution is to restart the PDP.
- The PDP uses lots of RAM and crashes when the process memory limit is reached. The PDP does not release memory frequently enough, which causes the PDP process to request more memory until the limit is reached and the process terminates.
- In the event of an authorisation denial, the PDP does not inform the OMF RC. This makes OMF experiments very hard to run, because the experimenter does not know what has happened.
- Excessive logging of the PDP. The log files contain everything that takes place. This makes it hard to see if something is working or not.

These issues were reported to the PDP developers and the updates made to the PDP to address them are described in D7.6.

2.5.3 Service Level Federation

In addition to the development of the SFA gateway to BonFIRE, where the aggregate manager API calls defined by the SFA architecture are implemented using BonFIRE API calls, BonFIRE has added support for service level federation (despite some mismatch in concepts). As described in section **Error! Reference source not found.**, service level federation allows Fed4FIRE users to access the native API of BonFIRE in a way that supports the native SFA concepts of user credentials and speaks_for credentials. This allows tools, such as service orchestration tools, to interact with the native BonFIRE API while preserving the general trust and security model of SFA.

A version of the BonFIRE API supporting the concept behind the service level federation has been made available for tools developers on November 6th, 2015. This has required a rework of authentication management in that API, in particular with support of dynamic creation of users with enough usage quota for running the standard tutorial and a few trials.

2.6 Connectivity

Cycle 3 does not add further requirements for BonFIRE. Support for experiments with connectivity needs will continue through to M48.

3 SmartSantander

3.1 Facility description

The SmartSantander testbed is an experimental test facility for the research and experimentation of architectures, key enabling technologies, services and applications for the Internet of Things in the context of a city (the city of Santander located in the north of Spain). More than 20,000 IoT devices both fixed and mobile have already been deployed. From the whole SmartSantander consortium, which includes Belgrade, Guildford, Lübeck and Santander, only the Santander deployment is being used within Fed4FIRE.

The SmartSantander infrastructure enables a twofold approach in terms of experimentation: service and native experimentation.

- Service experimentation consists of running experiments and/or applications based on the data gathered by SmartSantander sensor infrastructure and stored in a shared repository. Therefore, these services will be mainly based on data retrieval from this repository. This way, third parties will be able to provide added-value functionalities based on them, hiding the complexity of the SmartSantander infrastructure and only dealing with a high-level interface. In this sense, by using the data retrieved, service experimenters could run data mining procedures in order to infer more elaborated metrics and provide these extensions to, for instance, represent diagrams, maps, etc.
- On the other hand, native experimentation requires a thorough knowledge of the SmartSantander infrastructure and how the different nodes actually work. This type of experimentation is considered a low level experimentation as it directly accesses the nodes and its hardware.

Inside SmartSantander, those two experimentation paradigms are often referred as SEL (Service Experimentation Layer) and NEL (Native Experimentation Layer).

3.2 Cycle 3: Overview

Architectural work carried out on WP2 (T2.1) and WP4 during cycle 3 concluded with the definition of two different federation approaches in Fed4FIRE, depending on how the testbed provide access to experimenters. The first one is based on the usage of an SFA AM, whereas the second one is based on the usage of a service oriented interface on top of the testbed infrastructure. While SFA technology is used by the majority of testbeds under Fed4FIRE umbrella, it is not applicable for all types of testbeds. In addition, some testbeds already have a pre-existent management solution that can be adapted without the need of deploying a new management stack. By offering different federation approaches, each testbed can select the one that better fits their experimentation model.

As already stated in previous deliverables, from the two experimentation modes SmartSantander can provide only SEL layer is involved as part of Fed4FIRE project. NEL experimentation will probably be federated in the future following SFA approach. However, as a data centric type of experimentation, the characteristics of SEL do not fully correlate with the SFA principles, designed to be resource centric. For this reason, work carried out in SmartSantander during cycle 3 has been focused on the

enhancement of SEL layer federation following the service oriented approach. Definition of an advanced federation level for testbeds following this approach has been done during cycle 3, so the main objective for SmartSantander during this development cycle has been to reach that status inside Fed4FIRE. In order to accomplish this task, integration with YourEPM service orchestration tool at different levels has been completed. Finally, a plan for achieving Fed4FIRE advanced federation level through the deployment of an SFA AM has also been done, although development has not been finished due to resources constraints.

3.3 Cycle 3: Experiment lifecycle management

3.3.1 Description, Discovery and Provisioning

The two subsections included in this section references how SEL model federates following the two different approaches Fed4FIRE architectural work proposes.

3.3.1.1 Federation following own testbed API approach

Definition of this federation approach was firstly introduced in Fed4FIRE during cycle 2, although it was not properly included in the architectural deliverable until D2.7 (Third Federation Architecture). That deliverable includes both light and advance federation level definitions, but no technical details (beyond a generic "YourEPM tool compatibility") are included for the second one. During cycle 3 this has changed and the specific technical requirements for advanced federation level have also been defined. The main difference between those two federation levels resides in the fact that advanced federated level is achieved by testbeds whose resources can be controlled with Fed4FIRE federated tools, while light federated testbeds provide access through their own tools by using Fed4FIRE compatible credentials.

SmartSantander SEL experimentation already achieved light federation during cycle 2 by exposing SmartSantander IoT API allowing users/experimenters to access it using a PKCS#12 Fed4FIRE credential (derived from X.509 traditional one). In cycle 3 efforts were made towards achieving advanced federation level through a service oriented model federation approach.

Experimentation lifecycle management functionality via SmartSantander IoT API was heavily redefined during cycle 2 to accommodate latest trends in the testbed and requirements extracted from the Fed4FIRE Open Call experiments. D4.5 includes a detailed description of those modifications and their implications. Due to the amount of effort needed to implement those new functionalities and integrate them into Fed4FIRE, a complete release was not feasible during cycle 2, hence developments have continued during cycle 3. In fact, some of the work done at the beginning of cycle 3 is also described in D4.5. It is important to remark that, in order to cover some new requirements coming from Fed4FIRE architectural work, certain parts of the new SmartSantander architecture needed to be delayed and are still work in progress after cycle 3. Work done on this area during Fed4FIRE cycle 3 includes:

- Finish and deploy a production ready version of the SmartSantander Phenomena and Units of Measurement Directory (PUD) [https://api.smartsantander.eu/phenomena] and [https://api.smartsantander.eu/uom]. These components are an important part of the description and discovery stage.
- Finish and deploy a production ready version of all the SmartSantander SEL asynchronous

subsystem (see Figure 3). This experimentation subsystem allows the user to generate subscriptions to sensor measurements, hence provisioning its experiment. New developments have been done in several components, including the backend infrastructure to support subscription persistence and evaluation, the SmartSubscriber worker modules to generate the notifications and the different Async Notifiers modules to send the notifications by using the specific network technology. During cycle 3, RestHooks and OML notifiers have been completed and are available to experimenters in production.

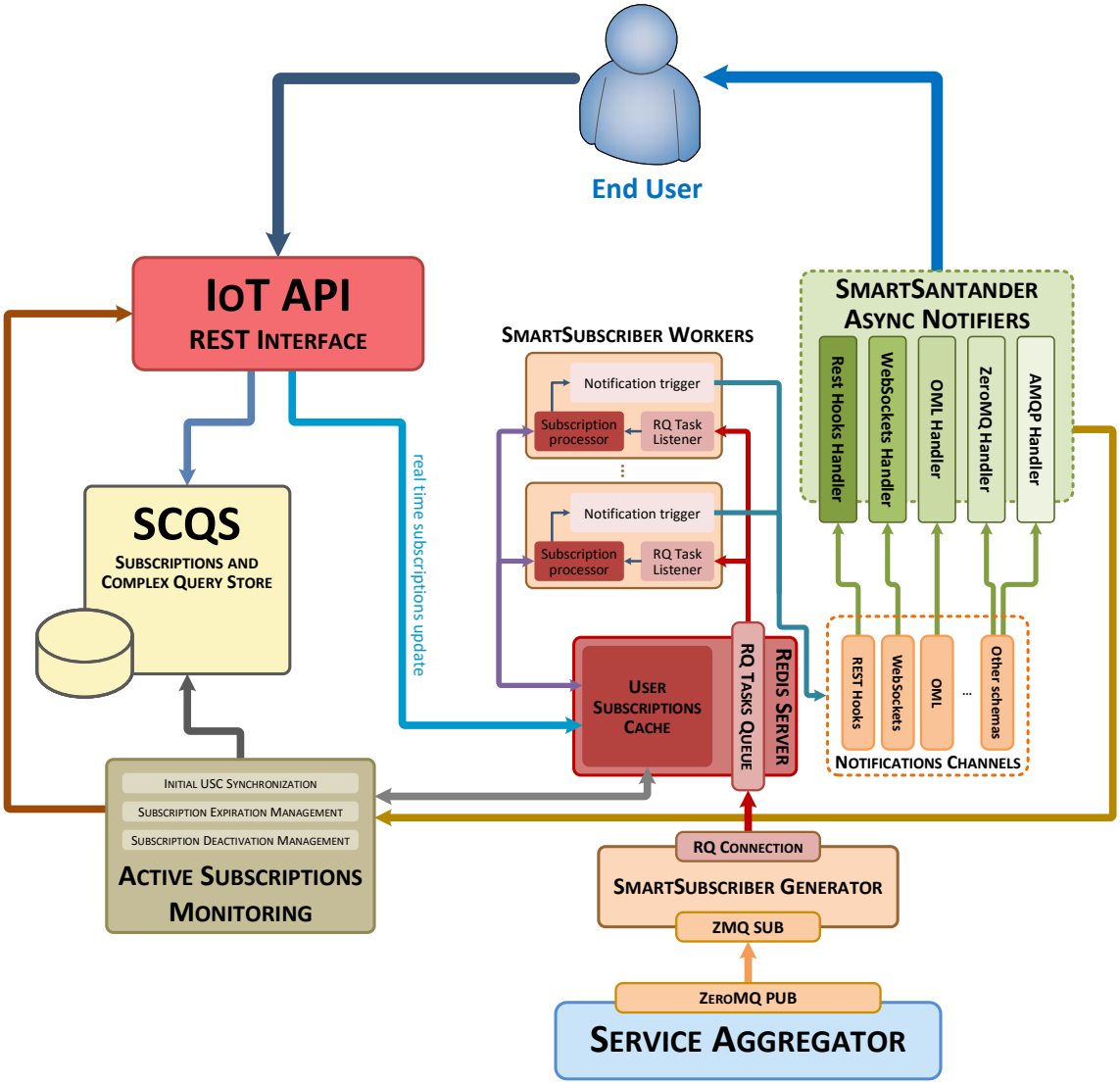

Figure 3. SmartSantander SEL Asynchronous Subsystem

- Partially deploy the second iteration of the SmartSantander IoT API. This REST interface is the basis of how a user/experimenter interacts with SmartSantander. During cycle 3 the different modules to provide access to both PUD and SEL Asynchronous Subsystem have been completed and deployed in production. Figure 4 shows a screenshot of the IoT API documentation hub [https://api.smartsantander.eu/docs]. Work is still in progress to finish the

whole IoT API tree, which relies on some uncompleted SmartSantander components.

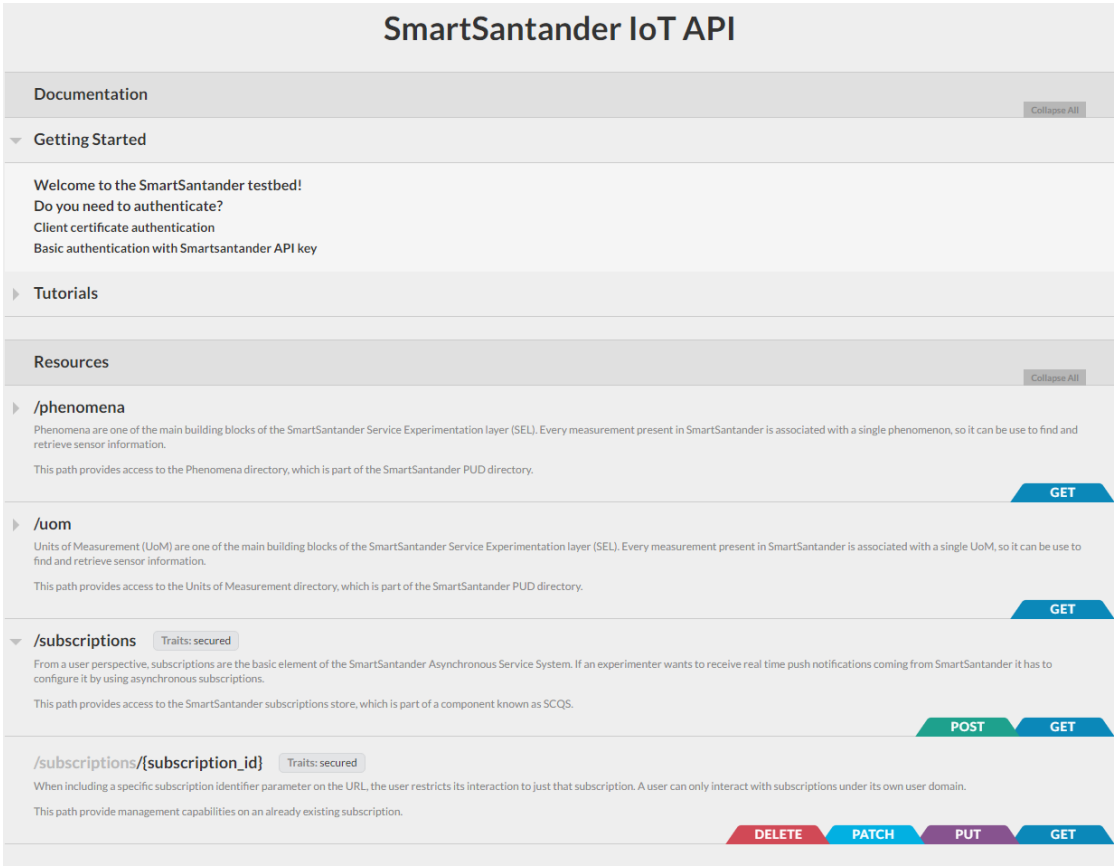

Figure 4. SmartSantander IoT API v2

- Improvements on SmartSantander Synchronous subsystem to allow access to historic sensor information. However, a production ready version is not yet available at the end of cycle 3.

As it has been introduced before, during Fed4FIRE cycle 3 YourEPM has emerged as a tool that allows federated experimentation under a service oriented context. In this sense, a new advanced federation level is defined based on the compatibility of a testbed service with this tool. One of the requirements to achieve this compatibility can be considered as part of the description stage (although it can also be seen as part of the interconnectivity task): to describe the exposed API using an accepted service description language. In the case of REST APIs, the service description language accepted by YourEPM is RAML v0.8 [http://raml.org/]. At the beginning of cycle 3, Swagger v2 [http://swagger.io/] language was proposed to be the one supported. However, due to some incompatibilities with XML schemas it has been replaced by RAML. In this sense, SmartSantander IoT API was described using Swagger before that decision, so the whole production API description has been rewritten using RAML. Although this is not a very complex task, it needs to be done manually, so is a time consuming task. The complete RAML resulting description can be found in [https://api.smartsantander.eu/description/raml/smartsantander-iot-api.flat.raml].

3.3.1.2 Federation following SFA AM approach

Despite the fact SmartSantander SEL experimentation model can't be fully mapped to SFA concepts, the analysis and design of how to provide an SFA based experimentation layer on top of SEL asynchronous subsystem was started during cycle 2 and has been completed during cycle 3. The provided functionality using SFA AM interface will not be as rich as the one based on IoT API, but it will empower SmartSantander integration with the rest of Fed4FIRE facilities. In addition, this will allow SmartSantander to reach advanced federation level following the SFA AM approach.

The decision has been taken to deploy an SFA AM based on FiTeagle [http://fiteagle.org]. By following a similar schema to the one already defined in D6.2, known as "Infrastructure monitoring for experimenters", the idea is to use resource provision stage to create SmartSantander asynchronous subscriptions based on RSpec contents. The drawback of this approach lies in the fact that, as RSpec description is organised by resources, subscriptions will always be resource based and flexibility decreases.

Although some initial developments have been started during cycle 3, due to the effort needed to implement some other Fed4FIRE requirements, such as speaks-for compatibility, this has been postponed to the future. As advanced federation level can be achieved following the service oriented approach this does not represent an obstacle in SmartSantander SEL federation.

3.3.2 Experiment Control

The data orientation of the SmartSantander SEL experimentation blurs the boundaries between resource discovery, resource provisioning and experiment control stages. The experimentation model SEL layer offers turns the different experiment phases into tightly coupled ones, and there is no real difference between provisioning and controlling the experiment. As isolating particular functionalities only related to one of this steps is not easy, some of the improvements presented on previous section can be also considered as part of this one.

As mentioned before, at the end of cycle 3 only asynchronous experimentation has been released on production in SmartSantander and has been integrated in Fed4FIRE. In this kind of experimentation, experimenters create subscriptions to different sensors to be notified when a new sensor measurement is generated. Figure 5 shows an example of one of this subscriptions. As part of the subscription model the experimenter also needs to define the technology and the endpoint to be used on notifications. At the end of cycle 3, RestHooks and OML technologies are available on production, and WebSockets is still under development. The contents of these subscriptions, together with the subscription management functionality provided by the IoT API can be considered parts of the experiment control stage in SmartSantander. The implementation and deployment of those control interfaces have been finished during cycle 3, and all of them are documented in the SmartSantander documentation hub. Work on this task has been already covered on the previous section, so no further explanations is included here.

Figure 5. Subscription example

3.4 Cycle 3: Measurement and monitoring

3.4.1 Facility Monitoring

This was already implemented in previous development cycles, so no further work has been carried out on this topic during cycle 3.

3.4.2 Infrastructure Monitoring

During Fed4FIRE cycle 2, an OML component was introduced as part of SmartSantander SEL asynchronous architecture. This component has been finished and deployed during cycle 3 and it will be in charge of application monitoring data. However, most of the metrics that are calculated in SmartSantander SEL layer are derived from the collected sensor observations, so they do not suit with Fed4FIRE infrastructure monitoring concept. In practice, in the case of SmartSantander SEL layer, infrastructure monitoring mechanisms and experiment monitoring are equivalent.

Due to the inherent characteristics of SmartSantander SEL experimentation, the system has been designed to address experiment monitoring requirements. Dynamic infrastructure monitoring will come as a by-product. Both of them imply the creation of subscriptions through the IoT API, although the criteria used for each of them and the post-processing to be performed will probably differ.

3.4.3 Experiment Monitoring

Traditional OML-based Fed4FIRE experiment monitoring is not applicable to SmartSantander SEL. This experimentation does not offer the experimenter a way of running applications, so OML-instrumented applications are not available.

However, as explained in previous sections, OML is one of the different compatible technologies of the Asynchronous Service System. Due to this fact, this SmartSantander subsystem can be considered as an OML instrumented application by itself, where the OML notifier module behaves as an OML MP.

This OML MP is then user configurable via subscription definitions.

The status of this task has been commented in previous section: it has been finished and deployed on production during cycle 3.

3.5 Cycle 3: Trust and security

3.5.1 User registration

No improvements have been done on this area during cycle 3. Authentication on the SmartSantander IoT API can be don based on the Fed4FIRE PKCS#12 derived client certificate. This way, it can authenticate different Fed4FIRE involved identity provider users and authorization can be performed based on the certificate fields.

3.5.2 PDP

No further work has been done on Fed4FIRE PDP component. As OMFv6 is not needed in SmartSantander SEL experimentation, the PDP component is not required.

3.5.3 Service Level Federation: Speaks-For credentials

As explained before, in order to achieve Fed4FIRE advanced federation level it is mandatory to offer compatibility with YourEPM tool. One of the requirements (already covered) is to provide a RAML description of the service to allow M2M communication between YourEPM and that service. The other one is to being able to accept calls from a user on behalf of another user following the speaks-for pattern. This allows authorized tools, such as the service orchestration tool, to securely interact with the testbed while preserving the general trust and security model provided by the usage of X.509 client certificates.

During cycle 3, SmartSantander has implemented compatibility with this authentication schema on the IoT API. A user with a valid Fed4FIRE credential can sign a Speaks-For credential for a tool (or another user). This credential, which is a signed XML document, includes the complete information of the user Fed4FIRE credential together with a reference to the tool certificate, hence the whole trust chain can be built.

As this security schema has been adopted from GENI, Fed4FIRE have also adopted the tool to generate a Speaks-For credential. This tool, which is web-browser based [https://github.com/duerig/xml-signer], directly retrieves the user credential private key from an identity provider or allows the user to manually enter it. From a user's perspective, exposing its private key through a web browser is not a good security pattern, even though everything is executed locally. To overcome this problem, as part of the speaks-for related developments in cycle 3, UC has released an open source tool to generate and validate this kind of credentials [https://github.com/psotres/speaks-for].

3.6 Connectivity

SmartSantander work on connectivity is focused on inter-testbed service orchestration. This task is based on the ATOS' YourEPM tool integration and it has already been covered on previous sections. To summarize, this task has been completed during cycle 3.

4 FUSECO Playground

4.1 Facility description

The FUSECO Playground is a comprehensive testbed which provides multi radio access technologies ranging from Wi-Fi 802.11 ac, to 2G and 3G as well as LTE. The objective here is to allow proof of concept or experiments in a small-scale mobile operator network, rather than scalability experiments. Additionally the FUSECO Playground offers OpenEPC (see http://www.openepc.com) as a Service, which is required to run end-to-end experiments in a mobile operator network. The EPC clients are running on small size PCs which are equipped with USB modems to connect to the radio network. Furthermore, physical servers are attached to the operator backhaul network, where the experimenter can run applications that are required for its experiments.

The Cloud infrastructure, which is based on OpenStack, allows the experimenter to launch virtual instance for their experiments. On top of OpenStack the experimenter can run a virtualized OpenIMS core and virtualized IMS clients. Recently the OpenMTC platform was added as a service to the FUSECO Playground. OpenMTC is a prototype implementation of a Machine-to-Machine (M2M) middleware aiming to provide a standard-compliant platform for M2M services, to develop innovative M2M and Internet of the Things (IoT) applications.

4.2 Cycle 3: Overview

The main efforts in cycle 3 have been concentrated at the following tasks:

- Adoption of the Resource Description Framework (RDF) including RDF-based RSpecs definition to discover and provision available resources at the FUSECO Playground.
- Deployment of a new release FITeagle federation framework release which supports advanced reservation. This provides the capability for experimenters to reserve resources and scheduling experiments.
- Additionally more LTE equipment and resources have been added to provide a comprehensive testbed for LTE experimentation. Consequently new resource controllers were implemented to support the control of the added LTE resource during experiments.

4.3 Cycle 3: Experiment lifecycle management

4.3.1 Description, Discovery and Provisioning

The FUSECO Playground has chosen to offer the TLS-secured XML-RPC-based GENI SFA AM API v3 to offer resources to the Fed4FIRE federation. The implementation is based on the FITeagle framework, developed by TUB, and was successfully deployed in Cycle 1 using default XML-based GENI RSpecs. Given the focus of Fed4FIRE Task 5.2 to enhance these RSpecs by suggesting alternative approaches, in Cycle 2 the FUSECO Playground offered some of its resources to be operated by a proof-of-concept version of FITeagle v2, which supports the use of RDF-based graphs to describe resources. In Cycle 3 this implementation matured, so that the FUSECO Playground resource can now be described by both XML-based and RDF-based RSpecs using the SFA AMv3 API, including extensions to query for specific resources. Further, continuous integration testing with Fed4FIRE user tools have been executed and demonstrated to assure compatibility with other software components.

4.3.2 Experiment Control

In addition to the OpenFIRE-based XMPP server, a WildFly-based AMQP server has been installed within the FUSECO Playground for experiment control using OMF. A number of Resource Controllers are available for selected resources. The AMQP version 1.0, however, is not compatible with the latest version of OMF 6, as it requires AMQP version 0.9. Additionally, the OMF 5 compatible REST-based LTErf resource controller was installed for controlling eNodeB-related resources. Furthermore the experiment control capabilities were enhanced to support seamless handover between different access networks on the UE level. Additionally experimenters are now able to control the LTE radio signal through specification of attenuation from 0 to 127dB. Finally QoS parameters of the EPC network configurable to set bandwidth, delay or packet loss.

4.4 Cycle 3: Measurement and monitoring

4.4.1 Facility Monitoring

Everything was completed in last cycle and reported in previous deliverable.

4.4.2 Infrastructure Monitoring

Everything was completed in last cycle and reported in previous deliverable.

4.4.3 Experiment Monitoring

The FUSECO Playground integrated during Cycle 3 RSpec extensions that allow users to specify an OML endpoint, to which experiment related information can be pushed to. The experimenter can choose if the monitoring information are send to the OML collector provided by the testbed or if the information are pushed to its own OML server.

4.5 Cycle 3: Trust and security

4.5.1 User registration

Everything was completed in last cycle and reported in previous deliverable. FUSECO Playground trusts all Fed4FIRE identity providers while operating own certificate authority and generating internally users on demand when external users from a trusted federation uses the testbed.

4.5.2 PDP

The PDP component is deployed at the FUSECO Playground and basic functionality tests have been conducted. However this feature cannot be considered as production ready, as detailed evaluation like proof of concept and performance use cases need to be utilized first.

4.6 Connectivity

As already described in the previous deliverable the FUSECO Playground IT infrastructure is dual stack enabled. A L2 connectivity to another testbed facility was not requested during the period of cycle 3, therefore it was decided to postpone it until some experimenter use case is demanding this. A dedicated L2 connectivity to the GEANT, implies extra fees that Fraunhofer had to pay to the local NREN.

5 FIONA

5.1 Facility description

FIONA is a cloud platform for creating, improving and using virtual robots, which represent a new way of interacting with technology that includes access to information, web-based services and also actions over intelligent environments.

FIONA allows users to create and use intelligent virtual robots and/or to make them more interactive, more striking and smarter. Developers, researchers, companies, geeks and designers from around the world contribute to the platform uploading their killer feature, a specific behaviour, an amazing character or the best of their knowledge in a topic, which are encapsulated as “Sparks” and that can be combined to create different behaviours and personalities for the characters to enable diverse skills and capabilities.

While the end-user of the platform is able to run it on any device, FIONA’s cloud infrastructure runs the whole intelligence of the agent, and also is the platform to share the experience gained by the individual agents.

FIONA provides the mechanisms to encapsulate code of HMI technologies. It provides the required infrastructure to grab remote audio, video and text (input sensors) from a remote location and provides it to different modules (ASR, face recognition, etc.) to process an output (3D character video, TTS) which is streamed to the user.

There are three environments into Fiona:

- Sparklink: to create your Spark and upload it to Fiona. At the Sparklink people can combine different Sparks to create their avatar. It’s a drag and drop thing.
- SparkStore: It’s an online store where users share, either for free or charging for it, the Sparks they created or buy new ones from others. The main difference from other online stores is that users only share Fiona modules.
- Sparkrender: This is the final stage. Once the avatar is complete, users can post it on their website through the Sparkrender.

5.2 Cycle 3: Overview

Among the progress towards cycle 3 described in the report on second cycle developments of the services and applications community, Adele Robots has finished the integration of the user registration with Fed4FIRE certificates in the production environment, and finished the tutorials of basic experiment showing the testbed.

Regarding Adele Robots’ goal of integrating FIONA with Fed4FIRE service orchestration tool, based on ATOS’ YourEPM platform, this has not been possible since YourEPM only supports API REST, which is not provided by FIONA.

5.3 Cycle 3: Experiment lifecycle management

5.3.1 Description, Discovery and Provisioning

FIONA platform has certain characteristics that make the developments on it having additional complexities:

- FIONA does not provide API REST, this has made impossible the integration with the current implementation of YourEPM.
- FIONA's platform developments have to go through 3 different environments until it is ready for end-users: development environment, integration environment and production environment. Uploading from one environment to the next one implies all the necessary regression testing.

For creating and setting up FIONA's virtual robots, the platform offers the users its own drag and drop tool, where the sparks that give the different functionalities of the robot are chosen and set. From this tool (SparkLink), FIONA internally creates an internal file, avatar.xml, which will be interpreted while running the virtual robot and which is not provided to the user.

During cycle 3, Adele Robots developed for Fed4FIRE's experimenters, the possibility of introducing the configuration file avatar.xml directly in FIONA, without having to use FIONA's SparkLink tool. This would provide experimenters with a homogeneous interaction with others testbeds in Fed4FIRE. This is done through the Aggregate Manager integrated with jFed.

During this cycle, Adele Robots has studied the different possibilities for implementing the Aggregate Manager, among others, SFA and SFAWrapper, finding SFA, Slice Federation Architecture (SFA) standard (specifically GENI AM API v3), the most appropriate. The development of AM was done on top BONFIRE's GCF-Fed4FIRE implementation.

During M38 of the project stable implementation of SFA was made available online to FIONA production facilities, enabling the use of all GENI AM API v3 calls (GetVersion, ListResources, Allocate, Renew, Delete, Provision, Describe, Shutdown Status and PerformOperationalAction) for describing, discovering and provisioning resources on FIONA, including their future reservation (a concept introduced in BonFIRE 4.1).

Through jFed, experimenters can discover and provision resources in FIONA for their experiments.

During cycle 3 Adele Robots has developed the possibility of making reservations of slices, so that users can reserve in advance the resources they want to experiment on at any given time.

5.3.2 Experiment Control

We have studied an approach to implement Experiment Control with OMF. But since the experiments carried out in FIONA require involvement of humans to validate the results, there has not been a clear mapping of the concepts included in frameworks such as OMF and FIONA concepts.

5.4 Cycle 3: Measurement and monitoring

5.4.1 Facility Monitoring

Ongoing work is being done to support facility monitoring.

5.4.2 Infrastructure Monitoring

Ongoing work is being done to support infrastructure monitoring.

5.4.3 Experiment Monitoring

No experiment monitoring is planned to be supported in FIONA, since Experiment Control cannot be done.

5.5 Cycle 3: Trust and security

5.5.1 User registration

During cycle 3, the user registration using Fed4FIRE's certificates has been uploaded to the production environment. Authentication can be done based on the Fed4FIRE PKCS#12 derived client certificate. FIONA can authenticate different Fed4FIRE involved identity provider users and authorization can be performed based on the certificate fields. New users are created in FIONA on the fly when presented a valid certificate from trusted CA (iMinds).

5.5.2 PDP

No further has been done on Fed4FIRE PDP component.

5.6 Connectivity

During cycle 3 Adele Robots has integrated FIONA with jFed, in order to enable user to include FIONA in their experiments from jFed.

6 Conclusions

Progress in the service and application test facilities of WP4 has proceeded well in the reporting period. The emergence in the architecture of different levels of federation has enabled each test facility to implement the features of the architecture that provide access at the most appropriate levels to meet the needs of experimenters. The test facilities are well placed to enter the sustainability phase of the project.

Tools and components aimed at making service-level federation easier have been developed. Take-up of these by experimenters during the sustainability phase of the project will be encouraged by the project.

The overlapping development cycles in the project have allowed the test facilities to carry out developments in each cycle with a view to how they will tackle the developments required by the following cycle.

The expansion of the user base through further experiments has given the test facilities an opportunity to interact with real users. The response from users to Fed4FIRE has largely been positive from the experimenters.

7 References

[1] Fed4FIRE Architecture Workpackage D2.1 “First federation architecture”

[2] Fed4FIRE Experiment Lifecycle Management Workpackage D5.1 “Detailed specifications for first cycle ready”

[3] Fed4FIRE Measuring and Monitoring Workpackage D6.1 “Detailed specifications for first cycle ready”

[4] Fed4FIRE Trustworthiness Workpackage D7.1 “Detailed specifications for first cycle ready”

[5] Fed4FIRE Service and Applications WorkPackage “MS4.1 First design specifications for facilities”

[6] Fed4FIRE Trustworthiness Workpackage D7.2 “Detailed specifications for second cycle ready”

[7] Control and Management Framework http://mytestbed.net/projects/omf/

[8] SFAWrap, http://sfawrap.info

[9] GRAM GENI Rack Aggregate Manager, http://groups.geni.net/geni/wiki/GENIRacksHome/OpenGENIRacks http://www.opengeni.net/opengeni-software/

[10] GCF reference implementation of the GENI Aggregate Manager API, http://trac.gpolab.bbn.com/gcf

[11] OpenStack, http://www.openstack.org

[12] GENI Aggregate Manager API Version 2, http://groups.geni.net/geni/wiki/GAPI_AM_API_V2

[13] Geni-Tools, https://github.com/GENI-NSF/geni-tools

[14] Fed4FIRE Architecture WorkPackage D2.7 “Third federation architecture”

[15] Fed4FIRE Service Directory, https://portal.fed4fire.eu/portal/servicedirectory

[16] SmartSantander IoT API hub, https://iot.smartsantander.eu:8085

[17] Swagger description language specification, https://github.com/swagger-api/swagger-spec

[18] FI-Lab, The Open Innovation Lab, http://lab.fi-ware.org

[19] Activiti BPM Platform, http://activiti.org/

[20] RAML, the simplest way to design APIs, http://raml.org/