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**

D6.4 - Detailed specifications regarding monitoring and measurement for third cycle

Work package	WP6
Task	Task 6.1, Task 6.2
Due date	28/02/2015
Submission date	10/03/2015
Deliverable lead	Yahya Al-Hazmi (TUB)
Version	Final
Authors	Yahya Al-Hazmi (TUB) Tim Wauters (iMinds) Olivier Mehani (NICTA) Donatos Stavropoulos (UTH) Ciro Scognamiglio (UPMC) Javier G. Lloreda (Atos)
Reviewers	Julien Lefeuvre (Inria) and Halid Hrasnica (EUR)

Abstract	This document provides an overview of the measurement and monitoring services in Fed4FIRE and the respective stakeholders. Based on their commonalities, it presents implementations steps for the third, final development cycle of the project in terms of monitoring and measurement aspects of the Fed4FIRE federation.
Keywords	Measurement, Monitoring, Secure OML, Common Information Model, Data Visualisation

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)	

Version History

Version	Date	Contribution	Authors, contributors
ToC	23.01.2015	Initial ToC	Yahya Al-Hazmi (TUB)
v.01	03.02.2015	Table of content with assignments	Yahya Al-Hazmi (TUB)
v.02	06.02.2015	Section 2.2.2	Javier G. Lloreda (Atos)
v.02	13.02.2015	Sections 2.1 and 2.2	Tim Wauters (iMinds)
v.02	16.02.2015	Sections 3.2.1, 3.3.1 and 3.4.1	Olivier Mehani (NICTA)
v0.2	16.02.2015	Section 2.3	Yahya Al-Hazmi (TUB)
v.02	17.02.2015	Introduction, Sections 3.1	Donatos Stavropoulos (UTH)
v.02	22.02.2015	Section 3.2.2	Yahya Al-Hazmi (TUB)
v.03	27.02.2015	Sections 3.3.2, 3.4.2 and 3.4.3	Yahya Al-Hazmi (TUB)
v.03	27.02.2015	Sections 3.2.3, 3.3.3 and 3.4.4	Ciro Scognamiglio (UPMC)
v.04	28.02.2015	Section 3.5, Conclusion, Executive Summary, and editorial changes in the whole document	Yahya Al-Hazmi (TUB)
v.05	28.02.2015	Review version	Yahya Al-Hazmi (TUB)
v.06	03.03.2015	First internal review	Julien Lefeuvre (Inria)
v.07	06.03.2015	Second internal review	Halid Hrasnica (EUR)
v.08	10.03.2015	Review comments are addressed, Yahya extended Sections 3.1	Yahya Al-Hazmi (TUB) Olivier Mehani (NICTA) Tim Wauters (iMinds) Donatos Stavropoulos (UTH)
Final	10.03.2015	Final Version	Yahya Al-Hazmi (TUB)

Disclaimer

The information, documentation and figures available in this deliverable, is written by the Fed4FIRE (Federation for FIRE) – project consortium 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.

The Fed4FIRE project received funding from the European Union's Seventh Framework Programme for research, technological development and demonstration under grant agreement no FP7-ICT-318389.

Executive Summary

This deliverable presents the third and final cycle implementation of the Fed4FIRE monitoring and measurement architecture done by Work Package 6. The deliverable focuses on significant add-on features that are implemented in cycle 3 to complete the monitoring and measurement architecture as well as fulfilling requirements from other Fed4FIRE work packages and communities.

Fed4FIRE monitoring and measurement architecture includes three main services: facility health and status monitoring, infrastructure resources monitoring, and experiment measuring. Their design and implementation steps have been covered in the specifications of the first and second implementation cycle of the project, in D6.1 and D6.2 respectively. This will not be repeated in this document, but rather a brief summary is provided.

It was decided from the first cycle to use OML as a common framework for data collection and reporting across the federation. However, the native OML framework was not designed for a deployment as it is needed in Fed4FIRE. Therefore, some extensions and enhancements are planned in the third cycle of the project. Two main extensions related to data collection and reporting are planned:

- Support secure and authenticated data streams through encrypted channels and differentiated access control to storage backends by the involved parties, and
- Support semantic, common and meaningful data collection and representation according to a well-defined, ontology-based information model.

Furthermore, the Manifold framework was selected in the second cycle as a data broker for monitoring consumers (such as SLA management module and reputation engine) to query their data from the OML collection resources. This is planned to be extended in cycle three (the third main extension to the architecture) in order to:

- Allow experimenters to easily query data from their collection resources as well as visualising the data graphically through the portal.

It is to be noticed that these three extensions will support the main Fed4FIRE monitoring and measurement services, namely the facility monitoring, infrastructure resources monitoring, and experiment measuring.

This deliverable presents in details these extensions made in accordance with the requirements, along with possible approaches that can be taken for their implementations and the selected ones. The implementation methodologies and steps are discussed and the parties involved in each activity are also represented.

The methodologies and steps to implement the identified add-on features are listed in Table 1.

Functional element	Implementation strategy
Secure OML	• Develop a special release of OML that supports authenticated channels and differentiated access control (NICTA) • Deploy the secure OML if OML not yet available or replace the native OML by this release (all participants with support from NICTA/UTH) • Adjust OML clients to be compliant with the new release (all participants with support from NICTA/UTH)
Common information and data model	• Develop a common information and data model inspired by the Semantic Web ontology techniques. A Monitoring Ontology for Federated Infrastructures will be developed that models the main, significant monitoring and measurement concepts, terms and their relationships (TUB) • Develop a semantic OML framework that implements the developed ontology (TUB)
Secure, semantic OML	• Provide one release of OML that supports both add-ons (security and semantic) (NICTA/TUB) • Deploy the secure, semantic OML (depends on the usage of ontologies in other Fed4FIRE work packages (WP5 and WP7), if RSpecs will be replaced by ontologies and some federation services will adopt ontologies, the associated parties should deploy this release with support from NICTA/TUB/UTH) • Accordingly, adjust the used OML clients to be compliant with the new release (as before the associated parties with support from NICTA/TUB/UTH)
User-friendly data access and visualisation	• Extend the Manifold that acts as a data broker in order to allow experimenters to query their data (UPMC) • Extend Fed4FIRE Portal , based on MySlice and Manifold, to visualise experimenters data graphically (UPMC)

Table 1 - Implementation strategy of the functional elements

Acronyms and Abbreviations

AM	Aggregate Manager
API	Application Programming Interface
CPU	Central Processing Unit
DB	Data base
ETSI	European Telecommunications Standards Institute
FLS	First Level Support
FRCP	Federated Resource Control Protocol
GENI	Global Environment for Network Innovations
GUI	Graphical User Interface
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
ID	Identifier
IETF	Internet Engineering Task Force
IP	Internet Protocol
JSON	JavaScript Object Notation
JSON-RPC	Remote Procedure Call for JavaScript Object Notation
KPI	Key Performance Indicator
NMWG	Network Measurement Working Group
MS	Measurement Streams
MOI	Measurement Ontology for IP traffic
MOMENT	Monitoring and Measurement in the next generation technologies
NAPT	Network Address Port Translation
NOVI	Network innovations Over Virtualized Infrastructure
NTP	Network Time Protocol
Num	Number
OML	Orbit Measurement Library: an instrumentation system allowing for remote collection of any software-produced metrics, with in line filtering and multiple SQL back-ends.
OMSP	OML Measurement Stream Protocol
OWL	Web Ontology Language
QoS	Quality of Service
PKI	Public-Key Infrastructure
PSK	Pre-Shared Key
RAM	Random-access memory
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
RSpec	GENI Resource Specification
SLA	Service Level Agreement
SFA	Slice Federation Architecture
SNMP	Simple Network Management Protocol
SQL	Structured Query Language

SDN	Software Defined Networking
SPARQL	SPARQL Protocol and RDF Query Language
SSN	Semantic Sensor Network
TCP	Transmission Control Protocol
TDB	Tuple Database
TLS	Transport Layer Security
Tx	Transmitter
UDP	User Datagram Protocol
UML	Unified Modeling Language
URI	Uniform Resource Identifier
VM	Virtual Machine
XML	eXtensible Markup Language
XML-RPC	Remote Procedure Call for eXtensible Markup Language

Table of Contents

List of Figures

List of Tables

1 Introduction

This deliverable presents the specifications regarding the third cycle development in WP6. These specifications include the evaluation of possible mechanisms for the new implementations in the third cycle, the details of the selected mechanisms and their implementation steps. More specifically, the limitation of security support in the OML framework is addressed and the steps towards a Measurement and Monitoring Information Model are analysed. The rest of the document is structured as follows. In Section 2 requirements derived from the "Federation Architecture", other work packages and WP6 itself, are presented. Section 3 includes evaluation of new mechanisms, together with their design and implementation details. These include extensions needed to the OML framework for supporting identity management and access control and providing secure OML authenticated channels. Furthermore, an Ontology-based information model is presented and the necessary implementation steps for providing information models, data models and semantic OML are discussed. Further, data querying and visualisation solution is presented along with its implementation steps in order to allow users to have user-friendly access to their data. Finally, a conclusion of the deliverable is given in Section 4.

2 Inputs to this deliverable

This section revives specific information from some previously submitted Fed4FIRE deliverables. The input includes requirements and as well specifications in terms of monitoring and measurement that are considered when defining the third and final cycle development of the Fed4FIRE measurement and monitoring Architecture.

2.1 Architecture (WP2)

This section presents the monitoring architecture, as defined in D2.4 "Second Federation Architecture" [1] and verified in D2.7 "Second Federation Architecture" [57]. Fed4FIRE identified the following types of monitoring and measurement (Figure 1):

- **Facility monitoring**: this type of monitoring is used in the First Level Support (FLS) to see if the testbed facilities are still up and running. It is based on monitoring the key components of each testbed facility. The availability status of the key components of a particular facility are reported to a central location at the federation level that calculates the overall status of that testbed facility and represents its GAR (Green, Amber, or Red) status to the FLS.
- **Infrastructure monitoring:** this type of monitoring is about monitoring the facility infrastructure which is useful for stakeholders mainly the experimenters and some federation services such as Service Level Agreement (SLA) service, reputation service, and probably others. Compared to facility monitoring, infrastructure monitoring is characterised by a finer granularity, providing information about specific resources instead of the facility as a whole. Compared to experiment measuring, infrastructure monitoring is characterised by the fact that this concerns data that an experimenter or other federation service could not collect himself.
- **Experiment measuring:** this type represents measurements that are done through the use of measurement frameworks or tools that the experimenter uses, and can be deployed by the experimenter himself. The only hard requirement on testbeds supporting experiment measuring is the fact that they should support the export of such measurement data as an OML stream.

The corresponding architecture as defined in D2.4 [1] is depicted in Figure 1 and Figure 2. We refer to D2.4 and D2.7 for the details and summarize the main concepts here. Figure 1 illustrates the individual architectural components of the monitoring and measurement solution and their interactions. The components that are to be supported are depicted for the testbed side, the federator side and the experimenter side. Definitions and descriptions of these components are provided in D6.2 [2].

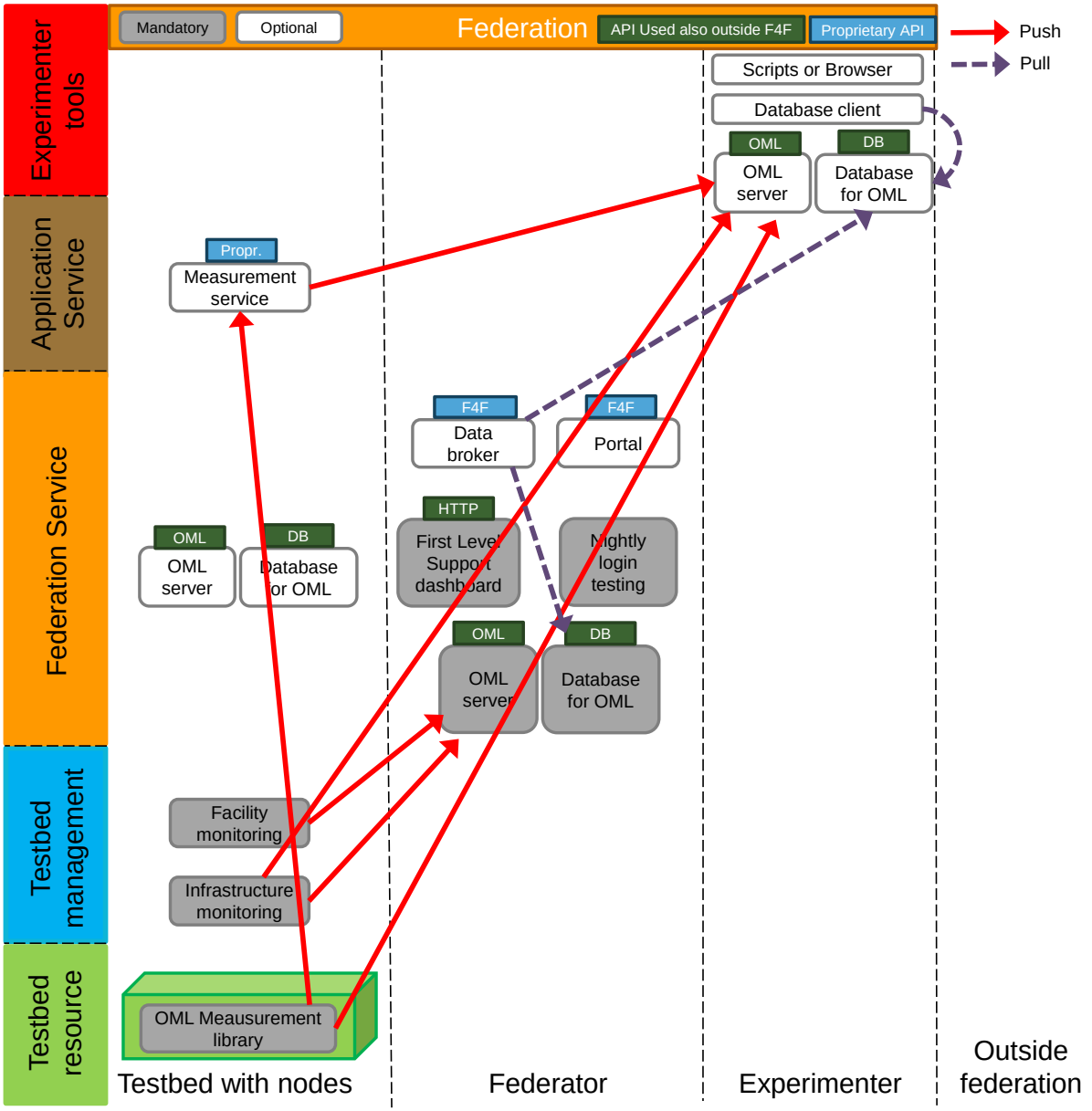

Figure 1: Monitoring and measurement architecture for cycle 2

Figure 2 shows the relation between the monitoring and measurement architecture and the SLA, reputation and reservation services. Facility monitoring data as well as infrastructure monitoring data are pushed by the testbed provider to both services SLA and reputation per experiment basis, while historical based data is pushed to the reservation service.

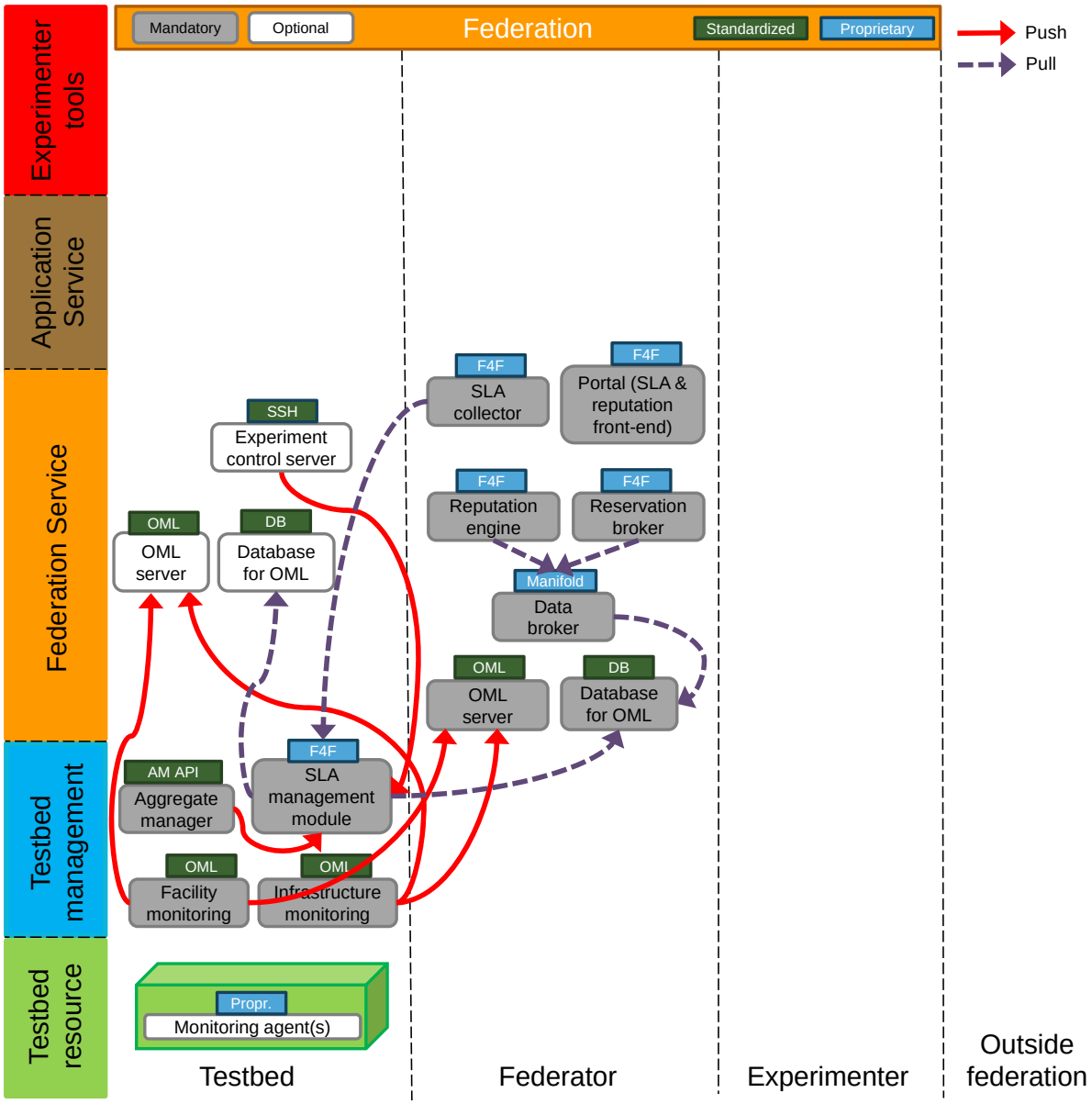

Figure 2: Relation between the monitoring and measurement architecture and the SLA, reputation and future reservation mechanisms as defined in cycle 2.

2.2 Requirements imposed by the other work packages

2.2.1 Requirements from the Infrastructure and the Services and Applications communities

In the joint Fed4FIRE deliverable D3.4/D4.4 “Third input from community to architecture”, the third set of requirements has been gathered from both the Infrastructure and the Services and Applications communities’ perspective. This collection mainly comes from the first Open Call experimenters, as well as from the submitted proposals to consequent Open Calls. The requirements

on measuring and monitoring cover procedures and tools supporting the observation and measurement of system and experimentation facility properties and also deal with other topics: accounting, decision making in resource management and accountability (who used what when, capacity planning and management). The experimenters have provided scores to indicate the relative importance of the requirements included in the table below.

2.2.1.1 Cycle 3 survey requirements

The scores in the table below represent the average values, from 0 (not important) to 3 (must have).

SURVEY REQ ID	REQ ID	DESCRIPTION	COMMENTS	GLOBAL AVERAGE
'5-11'	ST.2.011	That the overhead of any monitoring and measurement tool is minimal. These tools should have a negligible impact on the results of my experiment.		**2.40**
'5-9'	ST.2.009	That other aspects related to the successful execution of my experiment are continuously monitored, and that I am automatically informed in case of any errors. Examples are: when a selected resource could not be instantiated, when there is a problem with the interconnectivity between the used testbeds, when a used testbed goes down during the experiment, when there is a sudden peak of wireless interference, etc. This might be important when analysing anomalies in the experiment results.		**2.20**
'5-7'	ST.2.007	That the overall health status of the different testbeds (testbed up or down, has free resources left, etc.) is continuously monitored by the federation, and that in case of issues I am informed of this. Fed4FIRE will deal with errors and exceptions occurred during an experiment and raise alerts providing comprehensive information back to the experimenter, including information from several facilities. Capability to collect information from different testbeds in order to furnish an open platform with all the gathered product data.		**2.03**
'5-3'	ST.2.003	That by default some common characteristics of my resources are stored automatically for later analyses during experiment runtime (CPU load, free RAM, Tx errors, etc.).		**1.93**
'5-2'	ST.2.002	That Fed4FIRE makes it easy for me to retrieve and store data that I measured during the runtime of the experiment. This means that it should be easy to store my measurement somewhere in a way that the data is clearly related to the experiment ID, but without needing to establish connections to certain databases manually from within my code, and without needing to know the specific experiment ID that belongs to my current experiment.		**1.90**

SURVEY REQ ID	REQ ID	DESCRIPTION	COMMENTS	GLOBAL AVERAGE
'5-8'	ST.2.008	That the overall health status of the different testbeds (testbed up or down, has free resources left, etc.) is continuously monitored by the federation, and that in case of issues the corresponding testbeds try to solve them asap.		**1.87**
'5-12'	ST.2.012	That I can store and access my experiment monitoring data and other measurements on a data service on the federation, which is accessible during the experiment (temporarily data storage by the federation). Fed4FIRE must provide storage facilities for the experimenter to retrieve historical data concerning an experiment data-set and results. Data centres have to keep log of data stored and access to external resources. It is important for audits and historical data tracking	This requirement involves data management and archiving. Questions arise about the long-term storage of experiment data - access control, management, curation, who pays for long term storage.	**1.87**
'5-10'	ST.2.010	That when an error requiring manual intervention is reported to me as part of the previous step, that I am guided through the process for recovery.	Operational errors, such as failure of the infrastructure, impossibility to store or retrieve data are some examples of these errors	**1.83**
'5-14'	ST.2.014	That access to my stored data is properly secured. Experiments must be kept confidential if required, the privacy of experiments, data sets and results should be guaranteed.		**1.80**
'5-17'	ST.2.017	That I am made aware if my storage capacity is running out.		**1.80**
'5-4'	ST.2.004	That for the above monitoring, that I can select and configure how this data should be collected (always at a specified interval, only after a certain event or alarm, define some specific filters, etc). Fed4FIRE must provide tools to create, view, update, and terminate monitoring configurations related to shared resource types or experiments in real time. Monitoring data should be reportable for visualisation and analysis purposes with several reporting strategies (only alarms, all data, filters, etc.) in real time in order to provide accurate information and ease the analysis process. The experimenter might create own aggregated/composite monitored elements out of the available ones when designing the experiment, deciding what to monitor, defining some filtering possibilities as well, and providing the destination endpoint to send the information to. Monitoring information must cover information from different facilities and services. Monitoring metrics should be compatible across different facilities. For example, Monitoring computing & network resources' capacity. The monitored data during an experiment runtime will be available to the experimenter for all components involved in the experiment. If not aggregated, at least monitoring information from all involved testbeds must be provided. Testbeds must be able to publish infrastructure status through an API	As most of the monitoring in cycle 1 is done by using OML streams, experimenters should be able to create and/or configure new/existing OML Measurement Points in real time (maybe via FRCP)	**1.70**

SURVEY REQ ID	REQ ID	DESCRIPTION	COMMENTS	GLOBAL AVERAGE
'5-13'	ST.2.013	That Fed4FIRE makes it easy for me to retrieve and store data that I measured during the runtime of the experiment. This means that it should be easy to store my measurement somewhere in a way that the data is clearly related to the experiment ID, but without needing to establish connections to certain databases manually from within my code, and without needing to know the specific experiment ID that belongs to my current experiment.		**1.70**
'5-1'	ST.2.001	That the internal clocks of resources across multiple testbeds are synchronized very accurately; Comment from MobileTrain: "Different experiments need different time accuracy. Sometimes NTP is enough, whereas other times it is necessary a GPS module on each node."		**1.53**
'5-5'	ST.2.005	That I might create own aggregated/composite monitored elements out of the available ones when designing the experiment, deciding what to monitor, defining some filtering possibilities as well		**1.40**
'5-16'	ST.2.016	That I can share my stored data with specific others (individuals and/or groups), or even make them publically available		**1.37**
'5-6'	ST.2.006	That information about external wireless interference during the execution of my experiment is automatically provided for me.		**1.00**

2.2.1.2 Cycle 2 pending requirements

Infrastructure community

The scores in the table below represent the average scales used in cycle 2, with low (L), medium (M) and high (H) priority.

REQ ID	DESCRIPTION	COMMENTS	CYCLE 2 PRIORITY
I.2.205	Fed4FIRE should give the possibility to store metadata about the data of the experiments.	This allows easy lookup of experiment results, and eases the assessment of the meaning of the data. This seems most valuable when looking for shared data of other experimenters.	M
I.2.107	The user must be able to request on-demand measurements. In order to do so, they will need to express that they want agents with such on-demand polling capacities	The same information can be retrieved by looking into the output of the monitoring and measurement tools that will continuously provide measurements during the experiment run-time. However the on-demand measurement is more convenient during experiment development and debugging.	H
I.2.108	Infrastructure providers will need to evaluate experimenters' measurements request automatically in order to know if they can be met. If not, the experimenters should be informed about this.	If the measurement is not available, the returned zero or random values will most likely be noticeable by the experimenter. However, a formal notification of missing measurements (e.g., because a given metric is not applicable in all domains) is more convenient.	M

REQ ID	DESCRIPTION	COMMENTS	CYCLE 2 PRIORITY
I.3.401	Fed4FIRE should provide a method for querying and reporting the reliability of a testbed in terms of provided hardware, software and present wireless interference.	A possible approach could be to monitor facility resources to observe service experience. Regular questioning of the experimenters about their experience could be another possibility. In this case attention should be given to minimizing the burden on the experimenter, while making sure that untrue vicious feedback is not considered. Anyway, both the monitoring and the feedback approaches would need specific functionality to be in place in the Fed4FIRE federation.	M
I.3.402	In Fed4FIRE experimenters that create an experiment will need to provide a short high-level description of the experiment and its purpose. This allows infrastructure providers to keep track of the usage of the infrastructure, and enables them to report about this to their funding sources.		M
I.3.403	Fed4FIRE should provide the possibility to trace network traffic back to the originating experiment. This is useful when misuse of the infrastructure has been detected and the corresponding experimenter should be sanctioned (e.g., by revoking his/her account). The fact that accountability mechanisms are in place will automatically increase the level of trust that infrastructure providers can have in Fed4FIRE experimenters which are unknown to them.	FIRE facilities can be powerful tools, and misuse should most definitely be handled adequately.	M

Services and Applications community

REQ ID	DESCRIPTION	COMMENTS	CYCLE 2 PRIORITY
ST.2.018	Different services in different testbeds should be used with different identities (e.g. a shopping profile can be integrated with a supporter profile but waste cans require different identifiers)		L
ST.2.019	There has to be a mechanism that assures service users and experimenters that the providers are trusted.	Information providers must be identified and Fed4FIRE must ensure that they are who they say they are	M
ST.2.020	In addition to basic authorisation information, additional profiles related to a specific application must be definable for one service user		L
ST.2.021	The real id or owner of the device, who decides whether the information is sent, collected, etc. (external resource involved in an experiment) has to be kept hidden for other Service Users in order to avoid any legal problems. An alias or hash code should be used instead. A mechanism to unbind data from owner or real device identity for experimenters or service users is required.	This requirement is about the privacy of service users	L
ST.2.022	Provide the means to get end user opinion on service deployed on top of the facilities provided on Fed4FIRE in order to measure the reputation of the facilities and services provided. Facilitate interaction between experimenters and federated facilities.	This is linked to reputation	M
ST.2.023	Service providers can register to notifications on changes on information of their interest		L
ST.2.024	Fed4FIRE will make the distinction between requests of local users, PhD students from other institutes (research), students (practical exercises), in order to know what kind of experimenter is logging in and from where and apply policies accordingly. Fed4FIRE must provide secure mechanisms to retrieve	Identification of user roles – policies could define rights users may have on resources - e.g. PhD students have fewer rights than a principal investigator. If they have complementary expertise, which is usually the case, they might be accessing different testbeds (not the same) for the same experiment, but this should	H

REQ ID	DESCRIPTION	COMMENTS	CYCLE 2 PRIORITY
	information produced by experimenters during experiments or to use resources privately. This access should be controlled according to these profiles. Moreover, Fed4FIRE must provide the means to authenticate different users belonging to several organisations that might collaborate in the same experiment. Of course, within the limits accepted by all parties in the federation, testbed owners should be able to control access to their testbed. Finally, within the limits accepted by all parties in the federation, testbed owners should be able to grant credentials to experimenters outside the scope of the federation for local usage.	not be restricted.	
ST.2.025	Testbed owners must present to funding parties data about: * scientific relevance of their testbed, probably by tracking publication linked to past usage and success stories * socio-economic impact, in particular usage of the platform by industry and SMEs * type of experiments run using their testbed * user provenance (geographic and type of users)		M

2.2.2 Requirements on Service Level Agreements (WP7)

Requirements on monitoring from the SLA point of view are related to improve the coverage of a wider range of monitoring metrics in order to extend the number of SLAs that testbeds can offer. Although resource availability is expected to be the only metric guaranteed by testbeds which adopt SLAs for cycle 3, the SLA Management module is implemented in a generic way to accept and evaluate any type of metric provided by the testbed. Every metric that can be directly monitored by the testbed can be retrieved by the SLA Management module from the testbed's monitoring database which then checks whether it is under the guaranteed value or not. Examples of these kinds of metrics are resource availability, minimum available memory for VMs or minimum network throughput.

Furthermore, more complex metrics can also be guaranteed, improving the coverage of SLAs for the future. These metrics, referred as derived metrics, are based on the result of applying certain computations over direct metrics, i.e. mean value over a period of time, ratio of previous metric values, etc. The SLA Management is able to compare a retrieved metric value against the guaranteed value and perform the corresponding notification in case a SLA violation occurs. Therefore, in order to offer SLAs for derived metrics, the testbeds are required to calculate those derived metrics themselves and make them available to be retrieved by the SLA Management module.

Metrics will be retrieved from the testbed's monitoring database during the experiment duration. The SLA Management module will be notified for the start and stop of the evaluation directly from the client tool used by the experimenter, instead of being notified by the AM, following the architecture requirements for cycle 3. Further details on the SLA process are provided in the WP7 deliverable D7.4.

2.3 Requirements from WP6 (measuring and monitoring)

This section gives an overview on the requirements addressed by WP6 for the third cycle implementation. The focus of the third cycle lies on the implementation of three new services that significantly advance the monitoring and measurement services: i) providing a secure data collection and access control, ii) common monitoring and measurement information model, and ii) user-friendly data access and visualisation.

2.3.1 Secure data collection and access control

One of the critical and important requirements addressed in Fed4FIRE is to have secure channels for monitoring data collection and transmission. Further, it is required that only those authenticated clients can report data according to given permissions (write access control).

Indeed, as OML framework [3] is used for the data collection, transmission and storage, WP6 aims at implementing suitable mechanisms to make OML more secure as it is natively not designed to support such requirements. It does not support identity management. However, by using the native OML implementation, it is possible that multiple OML clients reporting data from sources claim whatever identities they wish, i.e. even if a client is authenticated and authorized to report whatever data, it can push data on behalf of other users claiming their identities. This leads to a possible atmosphere for misbehaviours and misreporting by malicious users about others.

From this perspective support for a secure reporting of OML data is required as well as providing suitable solution for identity management of clients, especially if they are owned and controlled by different users that report data to either one single server or multiple ones that are also controlled by other parties.

It is to be noticed that our focus in this document lies on the write access control but not on the read access control. This is because the latter is already supported in previous cycles through the read access control capability of PostgreSQL database that is used as backend for OML server, where both compose an OML collection resource. The common use of such OML collection resource is that it is deployed by one data consumer, i.e. either deployed by the federator for collecting data from testbeds for First Level Support, or deployed for SLA management system or even by an experimenter that receives his data from multiple sources. In this perspective, there will not be multiple users who will read data from the OML collection resource only if the owner grants them access which is supported already by PostgreSQL database.

2.3.2 Common information and data models

There are a large number of measurement and monitoring tools available in the literature. They differ from each other in their application areas, i.e. where measuring or observing is taking place. This document will not present the available tools, their types, features or even taxonomies as these

are out of the scope of the Fed4FIRE monitoring and measurement. However, only measurement and monitoring solutions within the context of the federated Fed4FIRE testbeds are of our concern.

Diverse measurement and monitoring solutions are used in the concerned testbeds. A testbed might include diverse tools for monitoring different application areas, i.e. a tool or more can be used to monitor the infrastructure resources, while others are used for monitoring networking parameters and connectivity, some others for monitoring the running services, and so on. Further, some testbeds even provide capabilities for experimenters to measure and observe the performance and behaviour of their applications, services, or technologies under test. In this perspective, within one single testbed, diverse tools might be used. The diversity of the tools is not only on their usage and application areas but also on their design and implementations. As a result, they have different data structures and schemas, databases, and provide the data in different formats through various access possibilities. Again, the variety and heterogeneity of tools increase in federated environment as it is the case of Fed4FIRE federation.

Such diversity and heterogeneity lead to provisioning of the data in different formats and data representations. This would then lead to misinterpretation and semantic interoperability problems. This issue is known in the database domains where heterogeneous database systems are used [4]. There are however in the literature a couple of mechanisms allowing databases integrations and interoperability. This is of course out of the scope of WP6 and the requirements driving its implementation, however, some knowledge from this domain can be applied as input for this work. As long as there is disagreement on meanings, interpretations or intended use of information, semantic heterogeneity arise [5].

To shed light on the heterogeneity of data representation, we list in the following a set of possible heterogeneity examples that we face in the context of monitoring and measurement data and metrics within the Fed4FIRE federated testbeds:

- Naming conflicts: Tools could use different names to represent the same concept. Example, the number of CPUs can be named by tool A as “num_cpus”, while in tool B named “cpu_count”, etc.
- Unit conflicts: Tools could use different values to represent the same concept. Example, tool A provides the amount of the free memory in kilobytes while tool B in bytes.
- Different schemas: Tool A provides only the name of the measured metric with its measured values, while tool B provides the name, measures, unites, and further information.
- Different programmatic access: Tool A provides the data through a JSON-RPC API while tool B via a direct access to its database (e.g. MySQL) or through GUI frontend.
- Different implementations: Even if tools expose the same type of APIs, they might have different implementations and thus differently called. And similarly, tools that use the same databases might have various schemas.

In the first and second Fed4FIRE implementation cycles, the OML framework is used as a common interface across the federation allowing providing the data collected from different sources (either from one single testbed or from many) in a common format. In this way, the OML framework solves

part of the problem induced by the diversity and heterogeneity of the used tools. However, as it allows representing the data in arbitrary schemas, there is no common way of presenting the data in a meaningful way, for example, through unified schemas and vocabularies. This is a need for common information and data models that are used to represent the monitoring concepts and relationships in a unified manner.

2.3.3 User-friendly data access and visualisation

Experimenters will have their data collected in their collection resources. They can retrieve their data through either command-line database clients or query tools.

However, it is preferable to allow experimenters to have an easy, user-friendly access to their data. Experimenters should have the ability to retrieve their data in any easy way and if possible visualize these data.

3 Implementation of the third cycle's architectural functional elements

In this section, we briefly summarize the tools for monitoring and measuring chosen from the first and second cycle and introduce the new implementations for measurement and monitoring mechanisms of the third cycle. The OML framework has been already chosen from the first cycle as the backbone of our measurement and monitoring architecture for data collection and reporting. In this deliverable we will focus on extending the OML framework in a way that it can support authentication of measurement streams and differentiated access control to storage backends depending on the source testbed. This will address the limitation of security and ensure the authenticity of measurement streams. Concerning the monitoring and measuring information model, we plan to align our efforts with WP5, regarding the definition of an ontology covering the federation's aspects and provide one specifically for measurement and monitoring metrics. Having the data securely collected and stored in a common way with the help of the common ontology-based information model, we plan to extend the monitoring solutions allowing experimenters to access their data in a user-friendly manner.

3.1 Monitoring and measuring tools already selected from the first and second cycles of Fed4FIRE

Before presenting the new plans for third cycle implementation, the purpose of this section is to provide a short nominal reference of the tools already chosen from the previous cycles and how they are used to provide the three main Fed4FIRE monitoring and measurement services for various users. A more detailed report of these tools and the ones that were not finally selected can be found in the previous deliverables (D6.1 [6] and D6.2 [2]).

For the purposes of measurement collection, transportation and storage, the adoption of OML was a natural decision since most of the provided tools were already OML-enabled and the instrumentation of new tools is fairly simple as a few examples have already been provided in the Appendix B of D6.2 [2]. These tools can be divided in the categories of Facility/Infrastructure Monitoring, and Experimental Measuring. In the former category, prevalent tools like Zabbix [7], Nagios [8] and collectd [9], are used by the majority of the testbeds for monitoring the health of their facilities as well as the infrastructure resources. Integration with OML for these tools has already been provided through specific OML wrappers that are available in the appendices of the previous deliverables D6.2 and D6.3 [10]. In the latter category, tools like Iperf, iostat, vmstat, collectd, Packet Tracking [11] and more are used for facilitating the procedure of measurement collection during the process of experimentation.

The common denominator of all these tools is the instrumentation with OML which provides a standard and common way of collecting and storing measurements, using for instance SQLite or the

PostgreSQL backend of an OML server. Selecting OML for collecting and storing purposes, provides flexibility in the choice of measurement tools, both for monitoring and measurement services, since non OML instrumented tools can be easily wrapped and provide their metrics through standard OML streams. For the purposes of accessing the collected data, the Manifold [12], which is used as the main framework of TopHat [13] and MySlice [14], acts as a data broker (a complementary tool to OML) providing an easy way to query data stored in a single or multiple OML collection endpoints. This can be integrated with the Fed4FIRE portal or any other component such as reputation engine, reservation broker, etc. and allows these federation services to retrieve data from their respective OML collection endpoints.

Fed4FIRE monitoring and measurement architecture is designed in a generic way without specifying concrete list of monitoring metrics or measurements to be delivered by each testbed. So, it is left to each testbed to provide those metrics or information of interest for the different monitoring consumers: experimenters and federation services (FLS dashboard, SLA monitoring, reputation, etc.).

In the case of federation services, some are interested in facility monitoring while others in both facility and infrastructure resources monitoring. However, federation services differ from each other in the required monitoring information. It is left to the service developers and testbed providers to agree on lists of metrics to be monitored. For more details please refer to D6.2 (Section 3.5).

In the case of experimenters, they are interested in the three types of monitoring services:

- Facility monitoring: they can get facility health and status information displayed through the FLS dashboard.
- Infrastructure resources monitoring: a testbed willing to provide infrastructure resource monitoring service to experimenters provide this capability per resource type through RSpecs [16] as part of the resource description (this is supported by WP5). Experimenters can then request on-demand this service. Once requested, the AM at the respective testbed provides infrastructure resources monitoring data exported as OML streams to experimenters' OML collection resources. For more details please refer to D6.2 (Section 3.5).
- Experiment measuring: it is left to each testbed to provide whatever tools or frameworks for experimenters to measure their experiments. Experimenters can on-demand request this service and should have the ability if needed to get their measurement data collected as OML streams.

To conclude, the design and implementation of the three main services have been already provided in previous deliverables for the first and second cycle specifications. They covered most of the common and significant requirements provided by other work packages as well as some of the requirements included in this document in Section 2.2, as WP6 was aware on such requirements such as possibilities for experimenters to get monitoring data of their experiments during and after their lifetimes, etc. However, there are other requirements that should be supported by testbeds individually, e.g. provisioning of information about particular (testbed-specific) metrics that are of experimenters' interest. Finally, some other requirements are considered in this deliverable to be supported in the third implementation cycle such as secure data collection and access control, data visualisation, etc. All remaining requirements from WP6 and from other workpackages are considered in the new extensions of the architecture for cycle three.

3.2 Evaluation of possible mechanisms for the new implementations in the third cycle of Fed4FIRE

The main targeted solutions in the third implementation cycle within WP6 support:

1. Identity management and access control,
2. Common information and data models, and
3. User-friendly data access and visualisation.

In this section, we discuss possible approaches for implementing these solutions, while the selected mechanisms for each solution and their implementation steps are discussed in Sections 3.3 and 3.4 respectively.

3.2.1 Identity management and access control

As aforementioned in Section 2.3.1, it is required to support identity management and access control in the Fed4FIRE monitoring and measurement solution that is mainly based on the OML framework.

In order to simplify the explanation of the issue and how it was solved, in the following sections we consider the facility monitoring service where each testbed reports on the health and status of the key components at the testbed to a central OML collection server at the federation. It is required that each testbed is only allowed to report data within their own experimental domain. This reporting is done through OML, which does not currently support any form of access control. In order to cater for this requirement, multiple solutions can be envisioned, with various levels of complexity, scalability, and security implications.

The experimental domain is the only piece of information that needs to be known to write data to any database. A testbed or other malicious entity could forge a domain which doesn't belong to it, and write fake data to another database, as represented in Figure 3.

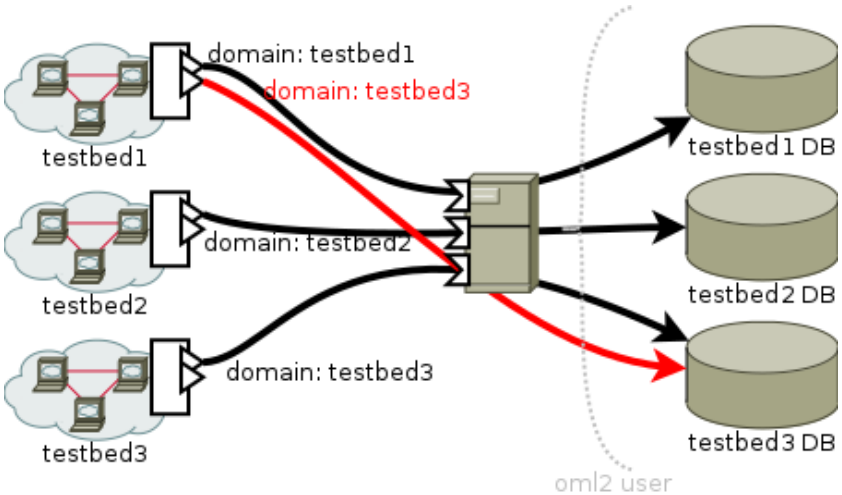

Figure 3 - Issue representation, where testbed1 (acts as malicious entity) sends fake data (represented in red line) about testbed3 which is then accepted and stored

The problem is essentially to offer a finer granularity in write accesses to the storage backend so applications running in one testbed cannot successfully send Measurement Streams (MS) in a domain that does not belong to them, and prevent them from successfully write the samples into the corresponding database. Potential solutions align along two axes:

- Differentiating MSs from different testbeds, or
- Authenticating MSs for access control.

3.2.1.1 Measurement stream differentiation

Each instance of the OML collection server connects to the storage backend as a unique user. Typically, this user is given write permissions to all the databases and adds records to the relevant tables in databases which name matches that of the experiment domain reported in the client's MS.

The idea behind MS differentiation is to run multiple instances of the OML collection server – one per testbed. Each instance will connect as a separate user to the storage backend, with limited permissions only allowing it to write to the database belonging to a specific testbed. Only testbeds are considered in this case. However, with MS differentiation, only known testbed would be able to report measurements, so rogue non-testbed users would already be locked out.

To differentiate MSs, and direct them to the relevant collection point (i.e., a different TCP/IP for each, rather than the default 3003), a set of static NAPT translations rules can be installed on the host running the collection server. Based on the source IP range of the MS, its originating testbed can be identified, and the TCP stream transporting it redirected to the relevant collection server.

By running separate instances of the collection server, each with limited access to a subset of the databases, and distribution MSs to the relevant server depending on their source testbed, malicious senders are no longer able to reach a database they do not have access to, as the backend will forbid the collection server to write to a database which doesn't belong to its user, as illustrated in Figure 4.

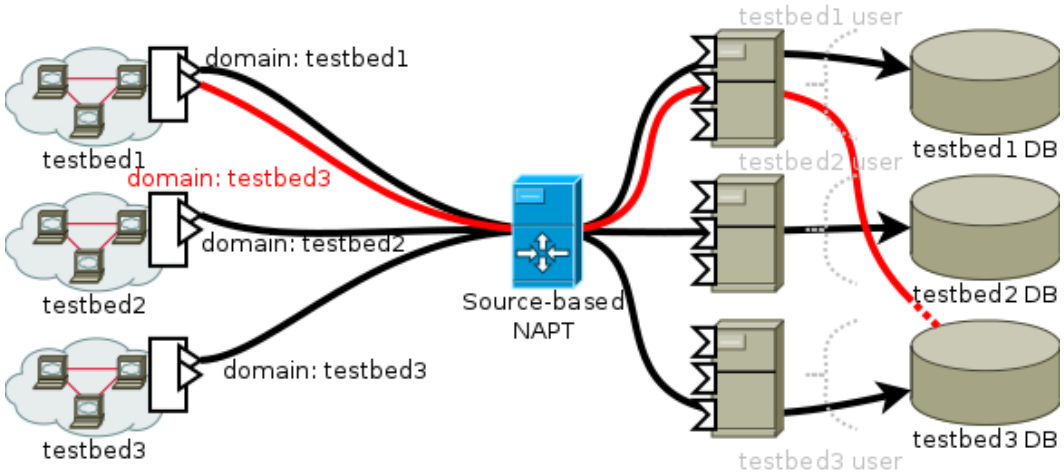

Figure 4 - Measurement stream differentiation, where testbed1 (acts as malicious entity) sends fake data (represented in red line) about testbed3 which is rejected by the database backend

The advantage of this solution is that it does not require any extension of the OML Measurement Stream Protocol (OMSP [17]), nor does it require modification of the software components of the reporting chain. As such, this solution can be readily deployed. The main drawback of this approach is that it is very limited in scalability, as one user per testbed needs to be created in the storage backend, and one more collection server needs to be run per testbed.

As this solution shifts the responsibility of enforcing access control to the storage backend, it also works under the assumption that this backend does support user-based Access Control Lists (ACLs). This is not always the case, which therefore limits the applicability of the solution. From a security standpoint, while communicating using a spoofed IP address is not trivial (though it is much easier when sharing a network with the destination), it is not impossible, and would therefore leave open some attack vector for malicious parties to add forged samples to the datasets.

3.2.1.2 Measurement stream authentication

Rather than passing on the task of access control to the backend, the OML collection server could also be in charge of the access control itself. In order to perform this task, the collection server requires more information to authenticate the source of the measurement streams.

Identification is already provided in the form of the experimental domain, and authentication could be provided by the addition of some credentials such as a password, X.509 certificate [18], or pre-shared key [21]. All these solutions share the requirement that the server needs to be configured with a mapping from credential to allowable domain (the X.509 approach could however embed this information within the certificate). The addition of a plain-text password as part of OMSP would be a straightforward way to enable some sort of authentication, and would only require lightweight changes to the client/server code and OMSP. An important drawback of this approach is that OMSP is currently a clear-text protocol. This means that the password would be sent in clear, and could be readily captured and reused by a malicious party.

The establishment of an encrypted session to carry OMSP traffic is therefore required if credentials have to be exchanged. Transport Layer Security (TLS) is the state-of-the-art protocol for encrypted bi-directional connections [19]. Implementing this protocol within OML is a heavier task than simply embedding clear text passwords in the MS, but provides additional benefit beyond supporting authentication. Of particular interest is the fact that the entire content of MSs can be protected from eavesdroppers, which OML cannot offer at the moment.

Using TLS to provide an encrypted channel for OMSP also allows more built-in authentication methods to be considered. Of particular relevance are pre-shared keys and X.509 certificates. The latter are already in common use to authenticate basic protocols such as HTTP [20]. They however require a substantial infrastructure deployment to support a Public-Key Infrastructure (PKI). While potentially interesting to support more flexible federation in the longer term, a full PKI infrastructure to support authentication of testbed's streams is not currently required. A lighter-weight option is to rely on pre-shared key between injection and collection points [21].

By requiring the inclusion of a secret to authenticate a testbed, the collection server can perform access control and only forward data to the backend if it comes from authorised servers.

This method is considered as the most suitable approach for the Fed4FIRE need which is going to be implemented. Detailed information about this application of this approach for Fed4FIRE need and its implementation is given in Sections 3.3.1 and 3.4.1 respectively.

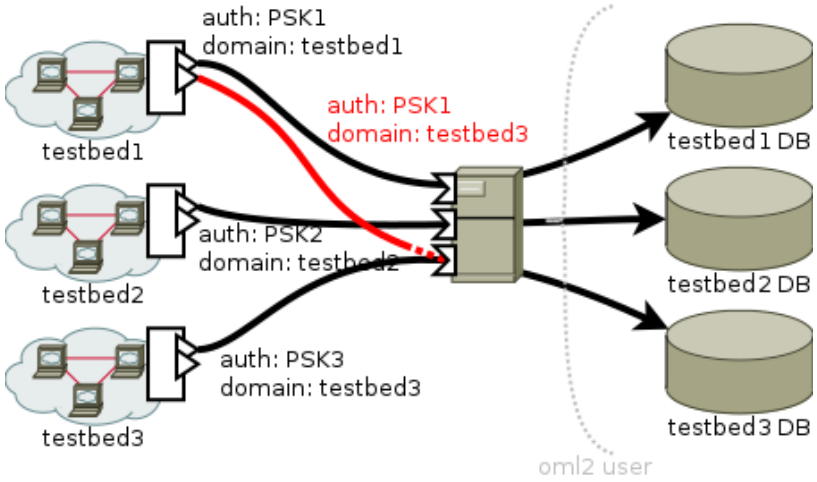

Figure 5 - Measurement stream authentication, where testbed1 (acts as malicious entity) sends fake data (represented in red line) about testbed3 which is rejected by the OML collection point

3.2.2 Common information and data model

According to the requirement addressed in Section 2.3.2, this section presents evaluated approaches that could be followed for implementing a common information and data model that covers the common measurement and monitoring concepts and terms within the scope and context of Fed4FIRE federation testbeds, architecture, and usage.

Before describing possible approaches to implement a common information and data model, we first introduce related definitions along with its relevant aspects such as monitoring, measurements, metrics, monitoring and measurement information and data model.

Monitoring: many definitions for monitoring are available, the most convenient definition is the one defined by the ITIL Service Management [22] as follow “repeated observation of a configuration item, IT service or process to detect events and to ensure that the current status is known”. In this perspective, monitoring within the context of Fed4FIRE is the process of constantly observing and recording information about resources (virtual and physical devices, system, process, application, network, traffic flow, etc.) to determine their status, utilization, and performance. This information is used by various experimenters, FLS, and the federation services.

Measurement: according to Weiner [23], “Measurement is a systematic, replicable process by which objects or events are quantified and/or classified with respect to a particular dimension. This is usually achieved by the assignment of numerical values.”

Monitoring information or measurement information: with this we refer to any piece of data that carries state information of a monitored entity. Note that information and data are used interchangeably.

Information model: is used by designers to model entities and their relationships within a given system at a conceptual level.

Data model: we follow the definition of NOVI [24], a data model "describes protocols and implementation details, based on the representation of concepts and their relations provided by the information model".

It is often a problem for many people to differentiate between information models and data models, many people interpret both concepts as the same thing, while in fact they serve different purposes. Compared to the information model, the data model can be seen as a lower level of abstraction that includes more details [25]. Further, a data model is represented in formal data definition languages specific to the protocol being used. In this perspective, as conceptual models can be implemented in different ways, multiple data models can implement one single information model.

So far we gave a brief overview on terms and definitions relevant to the work covered in WP6 regarding information modelling for monitoring and measurement.

Many challenges are faced while analysing possibilities of creating an information model for complex, federated environments as it is the case of the Fed4FIRE one. This is due to their diversity and heterogeneity. Examples include the heterogeneity of domains, the use of diverse tools, offering virtualised and real resources, different application areas, various use and access policies, etc.

However, there are different approaches to develop information and data models. We discuss in the following sections different approaches that can be seen as possible methods for developing the required common information and data models. However, before introducing these approaches, it is to be noticed that their main goal is to overcome the heterogeneity of data representation that leads to multiple conflicts as well as sematic problems.

3.2.2.1 Integration of heterogeneous databases

The integration of heterogeneous databases is one of the possibilities to have unified access to the data. Such integration is not a trivial task due to the different structures and semantics of the integrated schemas. There are many methods for integration of databases in the literature. These vary from one another depending on the associated requirements and usages. There are methods based on developing mediated, global schema at integration level and local wrappers above the databases allowing users to have transparent access to data [26]. A wrapper is a program that is specific to every database that translates/converts the data from local schema into the global one. Other methods are based on providing multi-layered schema architectures (i.e. starting from local

schema, export schemas, import schemas up to global schemas) where each layer presents an integrated view of the concepts that characterise the layer below [27][28].

The integration of databases approach assumes defining particular architecture design for data access. It does not scale well if we have diverse set of data sources as well as not flexible and generic enough to accommodate new data sources easily. Further, this approach lacks on semantic information, expressiveness, interoperability of data exchange between components, etc. Generally speaking, this approach is not valid for our demand as it focuses on databases and not on providing a solution for common representation of data collected and stored by arbitrary monitoring and measurement tools and frameworks.

3.2.2.2 Common data schemas

Another approach that could be used is to define a common data schema for the entire infrastructure, i.e. in the case of Fed4FIRE including the common concepts, terms and relationships in the Fed4FIRE federation concerning monitoring and measurements. There are several efforts on developing a common way for communication and monitoring data exchange, but they are limited to particular domains and/or designed for specific infrastructures and their needs.

For example, the Open Grid Forum Network Measurement Working Group (NMWG) worked on providing sharing knowledge about measurement tools and metrics [29]. The group developed an infrastructure allowing the communication between different systems in terms of network measurement data and knowledge. They focused on defining common vocabularies used to provide information about different measuring tools. For each of these tools, they defined a particular XML schema [30], called RELAX NG (Regular Language for XML Next Generation)[1]. The information is then sent in XML code defined by that schema. The information representation schema of PerfSonar [31] is based on the NMWG's XML schema that is used to represent network performance monitoring data. According to [32], these approaches have two drawbacks: i) they are based on the XML schema that provides only a common syntax but does not allow deducing any information from monitoring data; ii) they are limited to a set of services, i.e. don't cover every common monitoring and measurement concepts exist in the application areas of interest in Fed4FIRE federation.

There are other data representation formats such as those provided by the Internet Engineering Task Force (IETF) like the IP Flow Information Export Charter (IPFIX)[2] and the Simple Network Management Protocol (SNMP)[3]. Many other information models for IP traffic measurement are analysed and discussed in [33].

These solutions are limited to network measurement domains only and implemented for specific proposals which make them very complex to be adopted in a broader context. In addition, these works do not include any semantic information that can be used in combination with some rules to make decisions.

[1] http://www.relaxng.org/
[2] https://datatracker.ietf.org/wg/ipfix/charter/
[3] https://trac.tools.ietf.org/html/draft-chisholm-snmp-infomode-00

3.2.2.3 Ontology-based modelling

To enable common data representation or data access, various solutions exist as discussed in previous two sections (3.2.2.1 and 3.2.2.2). However, due to different requirements and applied contexts, some may have different (mismatched) concepts, structures, modelling methods and languages. Such task-specific and implementation-oriented solutions with a lack of a shared, common understanding lead to i) poor, limited interoperability, and ii) wasted efforts reinvesting the same subject matter.

The way to overcome this issue is to reduce or eliminate conceptual and terminological confusion through a common, shared understanding and definitions of things. This will allow not only interoperability but also reusability and reliability. For this purpose, the ontology-based modelling approach of the Semantic Web [34] is recognized as a valid approach that should be by definition as generic and task-independent as possible [35], and thus, allows developing a common, generic, extensible information model. Dealing with the information at a semantic level is believed to be a more powerful solution as it enables, in addition to representing data in a common way, some degree of deduction and automatic reasoning over the concerned monitoring data [36]. The advantage of ontologies compared to those solutions presented in Section 3.2.2.2 is analysed and shown within the context of the European MOMENT Project [37].

Before describing this approach, we first introduce the ontology definition.

Definition of Ontology: the mostly referenced definition of ontology is defined in [38] as: “a formal, explicit specification of a shared conceptualization”. This means that an ontology defines a set of formal, explicit vocabularies and definitions of concepts and their relationships within a given domain.

However, as our focus lies on monitoring and measurement, we can define monitoring ontology as follows.

Monitoring Ontology: is a formal representation of resources being measured, measurement metrics, measurement data, and data units, as a set of concepts and the relationships between those concepts.

Ontologies allow defining formally the common terms and relationships in taxonomies (subclass–superclass hierarchies) including rich semantic meanings. They include shared, reused vocabularies that are widely used in other ontologies. Further, they allow describing the information at different abstraction levels, i.e. it allows the definition of specific classes (representing monitoring concepts, terms, functions or capabilities) derived from generic ones.

Unlike other conceptual modelling approaches such as the Entity-Relationship modelling [39] and the Unified Modeling Language (UML)[4] class diagrams, ontologies enables flexible, hierarchal-structured,

[4] www.uml.org

semantically-rich and extensible, task-independent information modelling. The weakness of the Entity-Relationship approach is its expressiveness as well as the difficulty to describe complex systems and/or hierarchal classes. In contrast, the UML class diagrams can be used for modelling hierarchal structure but are not completely supported by “formal” logic and miss (or hard to enable) powerful features such as reasoning and the possibility to represent multiple relationships between two objects (only possible through aggregation) compared to ontologies.

3.2.3 Monitoring information querying and visualization

Information collected through OML is stored in a database, PostgreSQL has been chosen as backend. A way to visualise the data is to directly connect to the PostgreSQL database and execute the appropriate queries. This approach however has some issues, for once we can't aggregate this data with information coming from other sources easily; for instance we would like to merge information on resources coming from an SFA Aggregate Manager with the monitoring data. Also accessing the database remotely can be problematic and prone to security issues.

One approach is to build a data broker that can be used to query and retrieve those data. This data broker will already implement the required queries and will transform the data so it can later be merged with information retrieved from other sources.

Visualization of the measured data can be in the form of raw data (e.g. a table) or a graph that will show the variations of a particular value in time. This is the last step after querying the OML database and transferring the data, and it is usually done from a tool accessible to the final user. Because we already use tools to access testbed resource information it is only appropriate to build some sort of module for such tool so that a user can visualise information about a resource alongside monitoring information. Another approach would be to give access to the OML database to the user, or at least to the raw data, so that it can be exported in different formats for a subsequent analysis and display.

3.3 Details of the new selected mechanisms in the third cycle of Fed4FIRE

This section compared to the previous one focuses on the mechanisms selected in WP6 for implementing the functional elements that fulfil monitoring features and requirements addressed for third implementation cycle.

3.3.1 Secure, authenticated measurement streams

As discussed in Section 3.2.1, the TLS-PSK approach is the most promising to provide secure and authenticated MS transport. Indeed it not only provides authentication, as required to implement access control, but also encryption and a migration path to heavier and more decentralised federation infrastructures such as PKI-based federation infrastructure.

In this approach, the client would perform a TLS handshake using the experimental domain's name as PSK identity. In addition to the implementation of TLS, the collection server will need to be extended with a way to configure mapping of each pre-shared key to the domain it provides write permission to. Conversely, the client will need to be extended with a similar way to pass the pre-shared key to the underlying TLS socket.

As all the additional negotiation is performed when establishing the underlying encrypted channel, this solution does not require any modification of OMSP.

TLS-PSK negotiation is added as part of the client and server protocol. Figure 6 represents the TLS-PSK negotiation workflow. The pre-shared key is configured ahead of time at the server and testbed. The OML elements pass this information down to the internal components in charge of TLS-PSK, which is done as per the standards. Once the encrypted channel has been successfully authenticated and set up, the oml2-server forwards measurement data to the relevant databases. The proposed approach is similar in concept to STunnel [40]. However, the tighter coupling with the OML code offers several advantages: it doesn't require deploying additional tunnelling elements on the client and server, and allows the server to use some of the session information for authentication. STunnel provides encryption, but some rerouting in the ilk of stream differentiation would still be needed afterwards if no authentication can be performed at this stage.

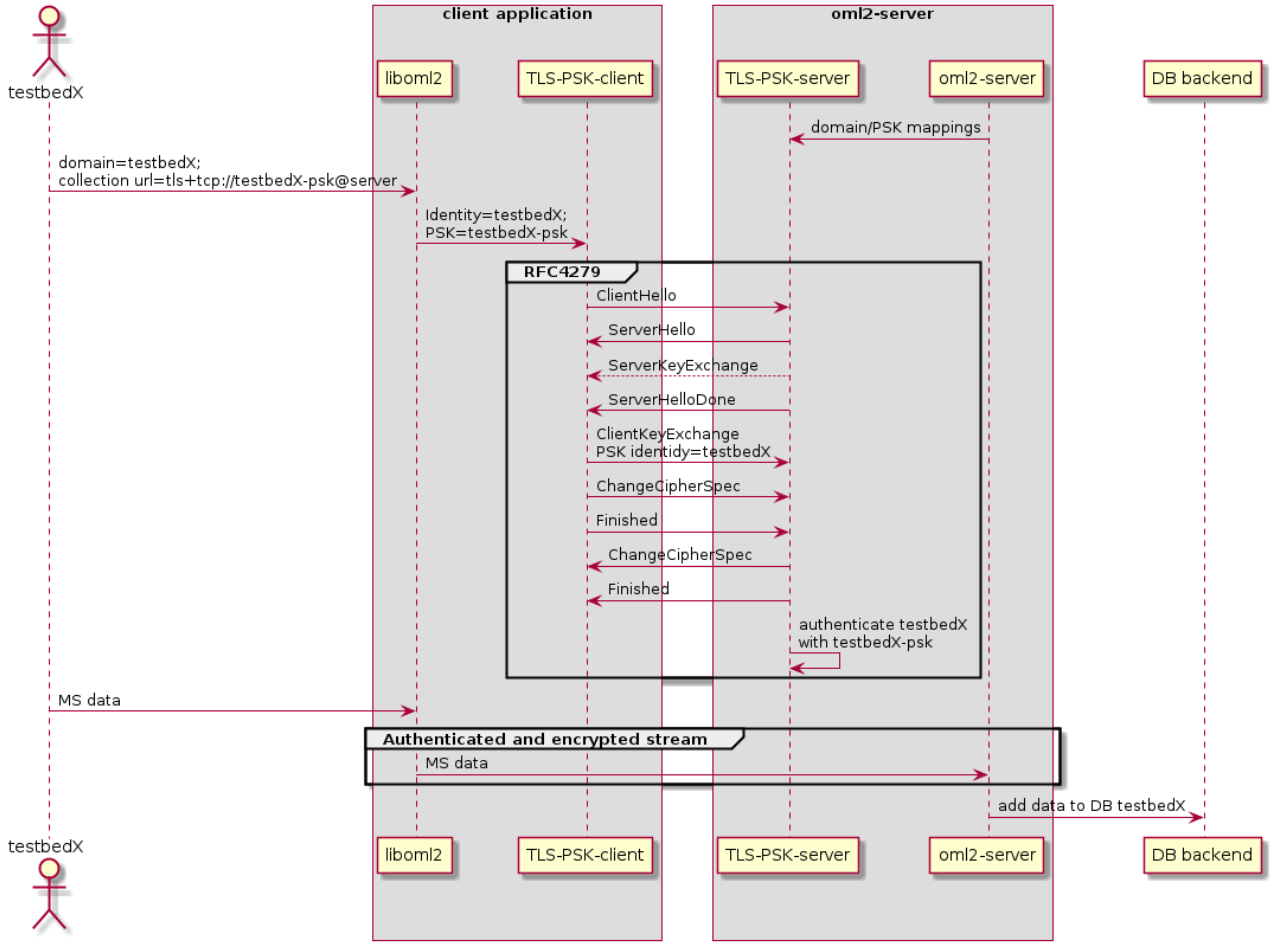

Figure 6 - TLS-PSK negotiation workflow

3.3.2 Ontology-based information model

According to the discussion on possible approaches for modelling common information and data models for monitoring data presented in Section 3.2.2, ontologies are considered as the best approach amongst others. However, this section gives an overview on how information and data models can be developed using ontologies and their respective languages.

Again ontologies are used to describe real world things in formal and explicit vocabularies. Those of WP6 concerns are related to monitoring and measurement concepts and terms. In order for these to be modelled using ontologies, we first list and briefly describe the main ontology components:

- **Classes**. Refers to concepts. A class represents group of **individuals** that share common characteristics, or also called instances of the class. A class can include several **subclasses**. For instance, we can have a class representing tools that can be called “tools” and includes subclasses “monitoring tool”, “filters” and “coverers”. There is a wide range of tools that acts as “monitoring tool” like Zabbix, Nagios, collectd; these can be considered as individuals.
- **Relations**. Describe the relationships between different concepts (classes and subclasses). There are different types of relations, some represent taxonomies such as “kind of” or “part of” and some represent associative relationships like “has name”. Concepts usually have **properties** that can represent relations. For instance, the class “monitoring tool” could have properties like “has name” and “measure”. Each property has domain and range.
    - **Domain**. Links a property to a class or an individual.
    - **Range**. Links a property to a class or data range (attribute).
- **Attributes**. Describe the parameter, features or characteristics of a class. They are represented as **data type properties**. For instance, a class “monitoring tool” has attributes like "has version” and “has licence”.
- **Axioms**. Represents assertions (including inference rules) allowing the insertion of new facts and predicates.

Ontologies are represented by the Resource Description Framework (RDF) [41] which is a general purpose language as data model that allows describing and organising the data in graph-based (subject-predicate-object) format as shown in Figure 7. It is also referred to these graphs as triples. Figure 8 illustrates an example on how information is represented through graphical format, where any KPI has usually measurement data and measured by a measurement tool.

Figure 7 - RDF Graph

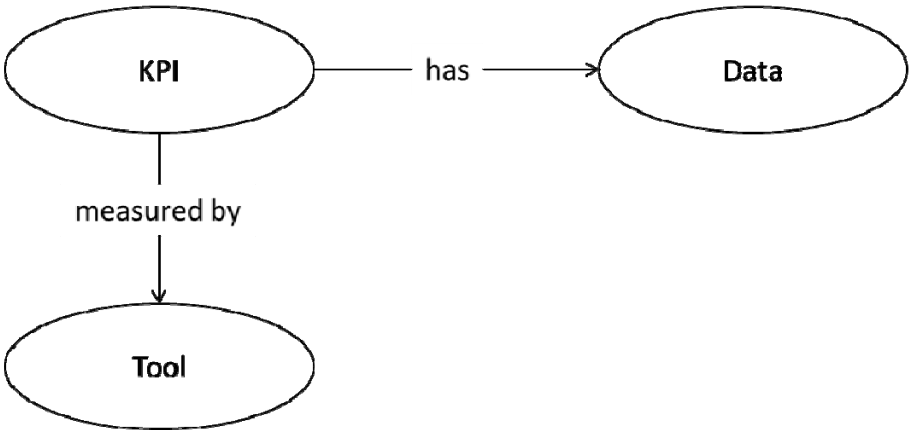

Figure 8 – Information represented through RDF graphical format

However, RDF provides basic vocabulary that is then extended in RDF Schema (RDFS) [42] which adds a data-modelling vocabulary for RDF. Besides, ontologies could be represented by many languages [43] such as Frame Logic (F-Logic), Ontolingua, Knowledge Interchange Format (KIF), LOOM, and Web Ontology Language (OWL).

The OWL [44], a semantic Web language, is the widely used and powerful language for representing knowledge (publishing and sharing ontologies) on the Web.

RDF uses XML Syntax[5] for expressing (serialising) these triples as XML document. Other serialization methods are also possible such as RDF-Turtle[6] that allows writing graphs in a natural text format, RDF N-Triples[7] which enables a line-based, plain text format, and Notation3 (N3)[8] which is readable RDF format.

Semantic ontology-based information models are extensible to include additional vocabularies for formally describing and modelling entities and relationships of anything "real" in the world. Among the main strengths of this approach are its trends of being standard and expressive, focus on interoperability, schema unbound and extensibility [45]. Information model can be defined through an ontology that uses for instance RDF as a data model; with RDFS that provides basis of vocabulary OWL allows extending the schema structure and additional vocabulary. These can be seen as a framework for modelling all forms of data (describing the data through vocabularies) and for allowing data interoperability through shared conceptualizations (ontologies) and schema.

Figure 9 gives a view on how an information model about anything real can be defined through an ontology, which can then use any suitable data model (e.g. RDF) that in return uses any of the possible serialization languages for expressing the data.

[5] http://www.w3.org/TR/rdf-syntax-grammar/
[6] http://www.w3.org/TR/turtle/
[7] http://www.w3.org/TR/n-triples/
[8] http://www.w3.org/TeamSubmission/n3/

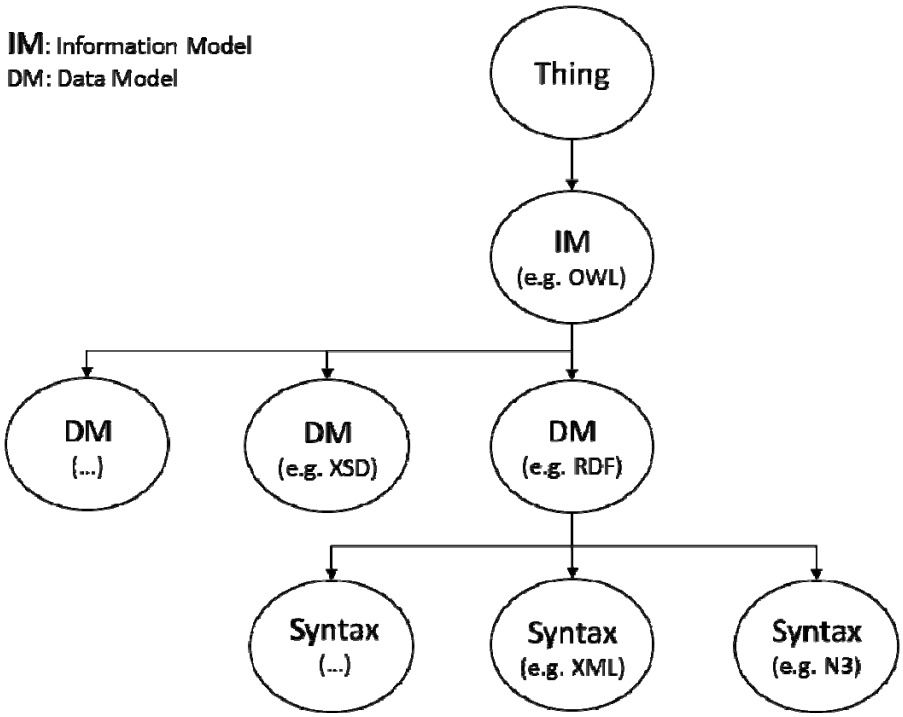

Figure 9 - Relationship between information model, data model and syntax

Further, ontologies support hidden information that couldn't be deduced using other modelling approaches. Figure 10 shows an example of such hidden information "represented in red" that could be made through ontology reasoning capabilities. Such hidden information can be deduced from the data available explicitly, due to the fact that the CPU Load is an instance of the KPI class and both have measurement data.

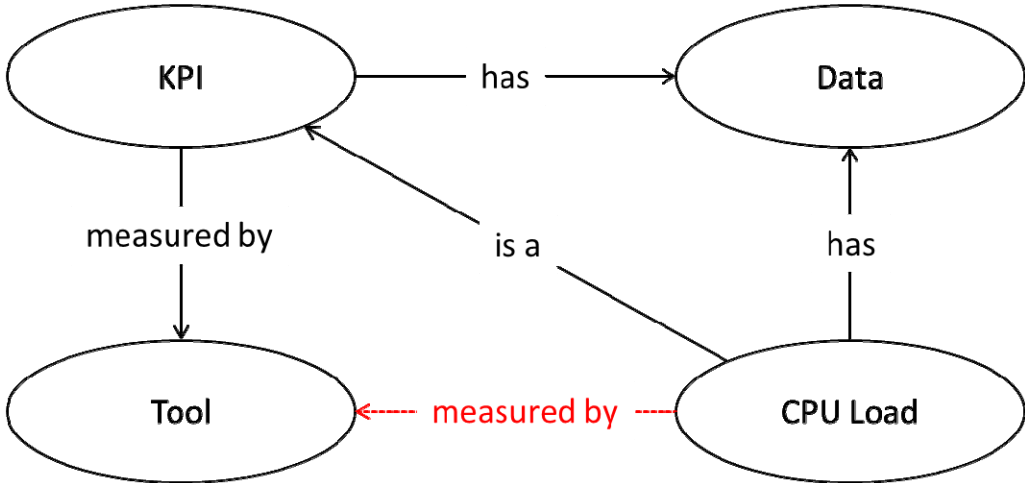

Figure 10 - Hidden information (in red) can be deduced through ontology reasoners

The target information model discussed in this document focuses only on measurement and monitoring concepts within the context of Fed4FIRE federation. This means it should cover the main monitoring and measurement services, resources being measured in the domains of interest, common measurement metrics, and other related aspects.

3.3.3 User-friendly data access and visualization

3.3.3.1 OML query support in Manifold

The chosen approach for querying and retrieving measurement data from an OML database is to use Manifold. Manifold is already able to aggregate information from different sources and of different nature with the use of gateways that act as a data broker between the actual service (in this case the OML PostgreSQL database) and the final application (e.g. MySlice [14]).

Manifold uses an XML-RPC based protocol for its API and an SQL Query like language to build the query requests. A Manifold query could include a request for measurement information alongside general information about resources or slices. Manifold will then asynchronously execute the requests (for instance one to the OML server the other to the SFA AM) and aggregate the results into a unique set of data.

3.3.3.2 OML information display in MySlice

Several tools exist to ease the process of table and graph creation, especially for the web, so the approach here is to integrate such tools in MySlice.

MySlice is a web application build on the well-known framework Django [15] (Python) that uses Manifold as a backend. The modular nature of Django has permitted to build add on software for MySlice in the form of plugins. With such approach a new plugin can be developed and implement the functionalities required to display measurement information retrieved via the backend (Manifold).

Further development include the creation of an easy to use interface that can be used by users to create a database that can be used to store monitoring and measurement information from the running user experiment. Such interface would also permit the user to query and display the stored measurements.

3.4 Implementation Steps

This section gives a detailed overview on the methodologies and steps taken to implement the selected approaches for the three target solutions discussed in previous section.

3.4.1 Secure, authenticated OML channels

OML internally relies on its OComm Sockets library to provide low-level socket-based connection management. Both the client library for injection points and the collection server rely on OComm. It is therefore the most relevant place to implement TLS. This will be done by leveraging existing libraries such as GnuTLS.

Beyond TLS, provision of pre-shared keys, and their mapping to experimental domains will be provided by extending the parameterisation options of client and server elements, respectively. Support for a configuration file will be added to the collection server, so mappings between specific pre-shared keys and domains (i.e., PSK identities) can be recorded, in addition to other operational and runtime parameters such as credentials for the storage backend – these can currently only be provided on the command line.

On the client side, pre-shared keys will be specified through the collection URI, either on the command line or in the existing configuration file [49]. The experimental domain is already provided separately on the command line. This information will be reused directly as the PSK identity for the TLS handshake.

The OComm socket abstraction can be extended to seamlessly offer encrypted TLS tunnels to the liboml2 and oml2-server as shown in Figure 11. A limited number of additional configuration parameters from the uses are needed to support TLS-PSK.

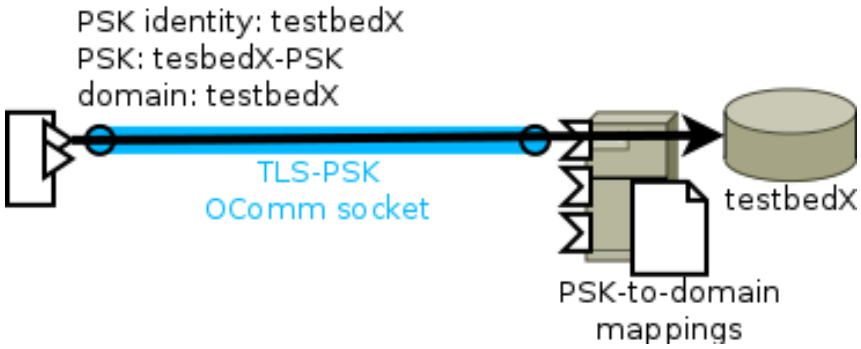

Figure 11 - OComm-TLS-PSK

3.4.2 Common information and data model

Implementation of an ontology based information model should follow the methodologies used to develop an ontology. Several methodologies are discussed in [50], however the common and the best methodology to develop an ontology has to follow the steps listed in Table 2.

Step	Definition	Fed4FIRE WP6 Procedure
1	Define the areas of interest and the scope of the ontology.	The area of interest in our case is the monitoring services and measurement metrics in federated infrastructures as it is the case of Fed4FIRE federation. We limit the ontology scope to those services provided based on the overall requirements concerning monitoring and measurement covered by Fed4FIRE WP6.
2	Check possible reuse of existing ontologies.	There are a couple of ontologies that have been implemented for monitoring that are considered in our development. These are the ontologies developed by the European projects

		MOMENT [51], NOV [52]. Further, the standardized ETSI MOI [53] is also considered. These are limited to particular domains with narrow cope (i.e. MOMENT focus on network monitoring only, ETSI MOI on IP Traffic Measurements while NOVI covers monitoring virtualized federated infrastructures) compared to Fed4FIRE scope that covers wide and heterogeneous domains. We also consider the SSN ontology [54] that focuses on the sensor network. Further, these were built according various requirements that differ from those of Fed4FIRE. These hinders direct adoption of these ontologies, however they will be used as a basis for the target ontology developed in Fed4FIRE WP6 as several of their models as well as vocabularies (classes, relations, properties) will be reused.
3	Identify the important concepts and relations	According to the Fed4FIRE requirements, the important concepts (monitoring services, metrics, domains, etc.) and relations (to measured resources, measurement units, relationships to reservations, SLA, trustworthy reputation, etc.) are identified
4	Define classes, subclasses, and then then arrange them in a taxonomic hierarchy	We will follow a top-down approach by starting with the definitions of the most general concepts and subsequent specialization of the concepts. We will utilize some already provided by related ontologies and extend them to cover all domains in the Fed4FIRE federation. For instance, wireless domains, Cloud computing and SDN domains were not yet covered by other ontologies.
5	Define properties of classes	We will define the most important ones first that fulfil the need of Fed4FIRE in all its federated domains
6	Define restriction of properties and rules for their domains and ranges	Following the Fed4FIRE needs and requirements these will be done.
7	Create individuals of classes	Create all individuals concerned in Fed4FIRE federation.

Table 2 - Ontology development methodology

The target ontology will facilitate the interoperability in the federated testbeds involved in Fed4FIRE in terms of enabling a common monitoring data representation. As is shown in Figure 12, Fed4FIRE federation includes several testbeds that have to provide data about the three main types of measurement and monitoring services are depicted: facility monitoring, infrastructure monitoring and experiment measuring through a common interface (achieved through the OML framework). These are required by multiple stakeholders such as the First Level Support (FLS) operators, the federation services and the experimenters. The FLS dashboard needs high level infrastructure monitoring (represented through the facility monitoring) to check the status of a service, to see whether a service is up or down at certain time, which is indicated in timestamp. SLA management

service requires monitoring information about the used infrastructure resources (e.g. servers, switches, links) and the data should include the resource name, name of the measurement metric (e.g. CPU load), value, unit, timestamp, etc. Experimenters might require infrastructure resource monitoring service as well as the experiment measuring, where the latter provides data about the experiment under test, e.g. the performance of service or application being tested.

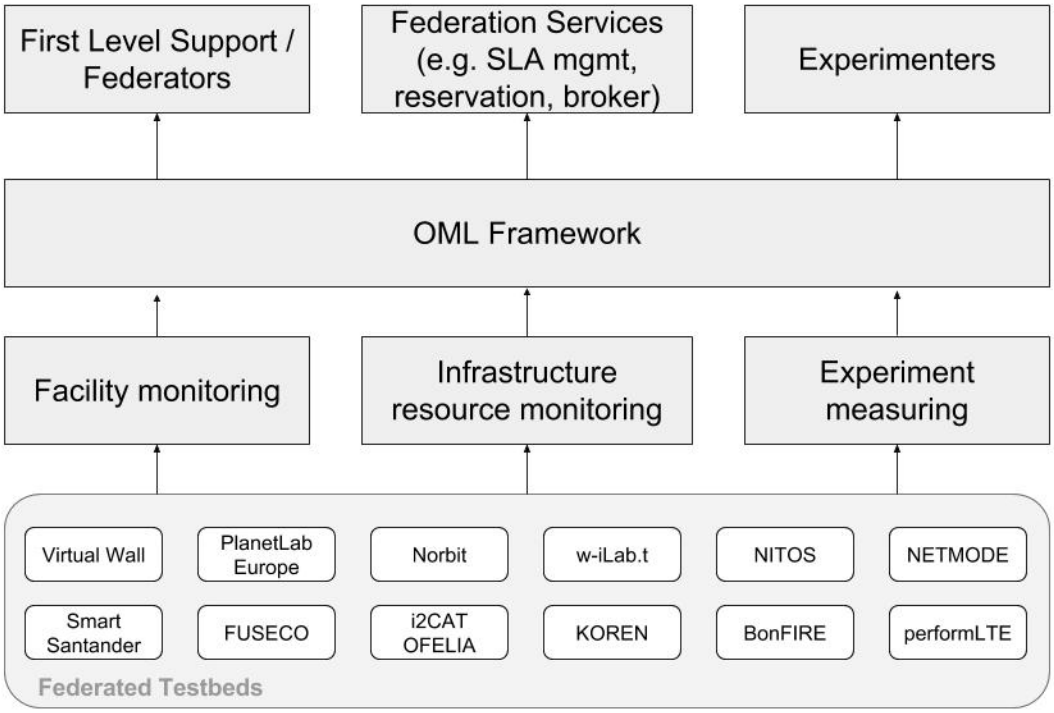

Figure 12 - High-level representation of monitoring requirements in a Fed4FIRE federation

Again we will limit the scope of the ontology not only for simplicity but also to focus only on the urgent and significant needs of Fed4FIRE monitoring and measurement services. However, we will take into consideration a main advantage of ontologies which is the reusability of the target ontology beyond Fed4FIRE project to be used by other similar federations. In this context, we are actively involved in the Fed4FIRE ontology task force that collaborates with other ontology experts, from the Global Environment for Network Innovations (GENI)[9] context as well, to develop an ontology for describing federated infrastructures and their resources called Open-Multinet Upper Ontology[10] as part of our activities within the Open-Multinet[11] initiative.

Besides, we will also consider only the main stakeholders of monitoring and measurements in Fed4FIRE; those are the experimenters, the federation services such as SLA management, trustworthy reputation, and reservation broker, and federation operators/administrators to have high-level information about the availability and status of the participating infrastructures through the FLS dashboard.

[9] http://geni.net
[10] http://open-multinet.info/ontology/
[11] open-multinet.info/

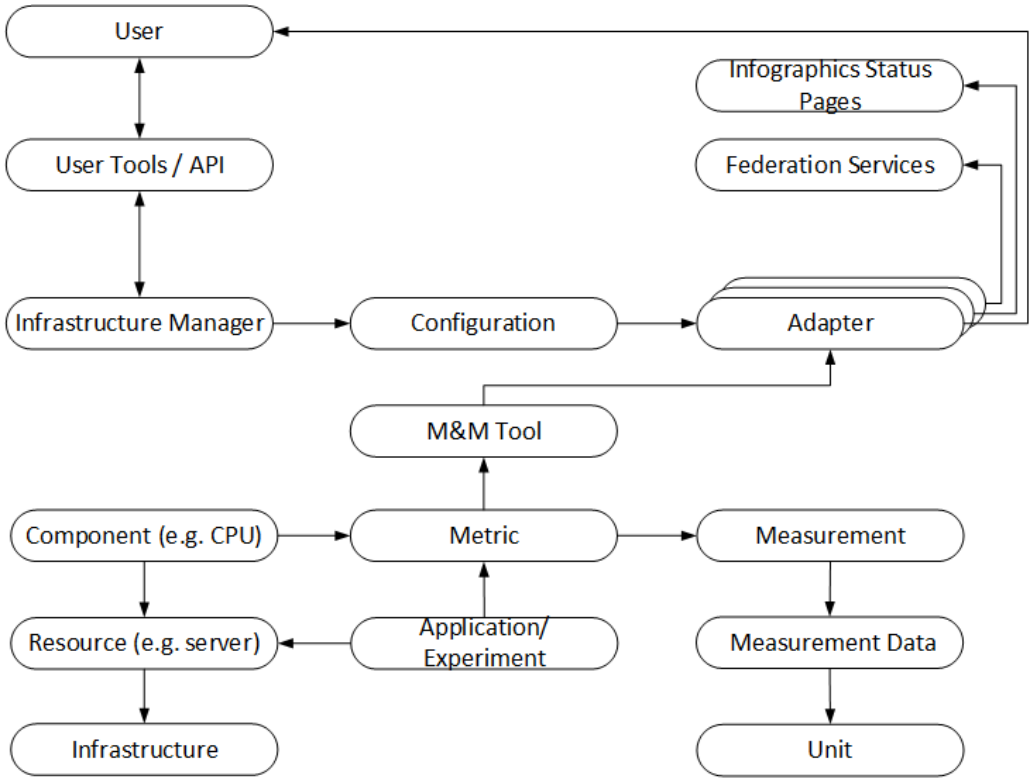

Figure 13 - High-level overview on main components of interest for the target information model

Figure 13 illustrates main components that are relevant to the aimed information model. Ontology support by some components is done within the context of WP5 (aggregate manager, federation services like SLA and reservation, Portal, modelling resources and components, infrastructures, experiments) and WP7 (some federation services like reputation). WP6 will focus on monitoring and measurement tools and metrics, measurement data, and data units.

The information model developed here will therefore include several main classes representing these concepts. An initial design of the upper ontology is represented in Figure 14 where four main models are indicated.

The **Metric** model represents anything that is being measured. It is defined in a generic way to cover the common measurement metrics in the concerned application areas. It defines metrics whose information change dynamically such as CPU utilization, memory consumption, packet delay and loss, etc., as well as metrics whose information may change very infrequently in time (e.g. CPU core counts in a machine).

The **Data** model describes the main concepts related to monitoring and measurement data (like simple measurements, measurement values, etc.) as well as information of relevant concepts such as the name of measured resource/application, measurement duration, used protocols, etc.

The **Unit** model defines the data unites (e.g. bit, byte, second, bps) covering both unit prefixes binary and decimal, as well as different levels (nominal, ordinal, interval, and ratio) and dimensions (basic and derived).

The **Generic Concepts** model represents generic concepts such as measurement location, timestamp, etc. The monitoring ontology is directly linked to other ontologies developed in Fed4FIRE within the context of WP5 and WP7 concerning resource, component and services.

For explaining the interactions of these main models with external ontologies (developed within WP5 for instance), Figure 14 illustrates further models (represented in dotted symbols) along with some relations between them and the models of the monitoring ontology. The figure is created in a way to represent an example that is described as follow: imagine if we have a server (modelled under Resource) that we want to represent measurement data (modelled under Data) with the proper unit (modelled under Unit) about the load (modelled under Metric) of its CPU component (modelled under Component) which is retrieved by an OML wrapper resource (can be modelled as an individual or subclass of Resource) and push the data to the reputation service (can be modelled as an individual of Service).

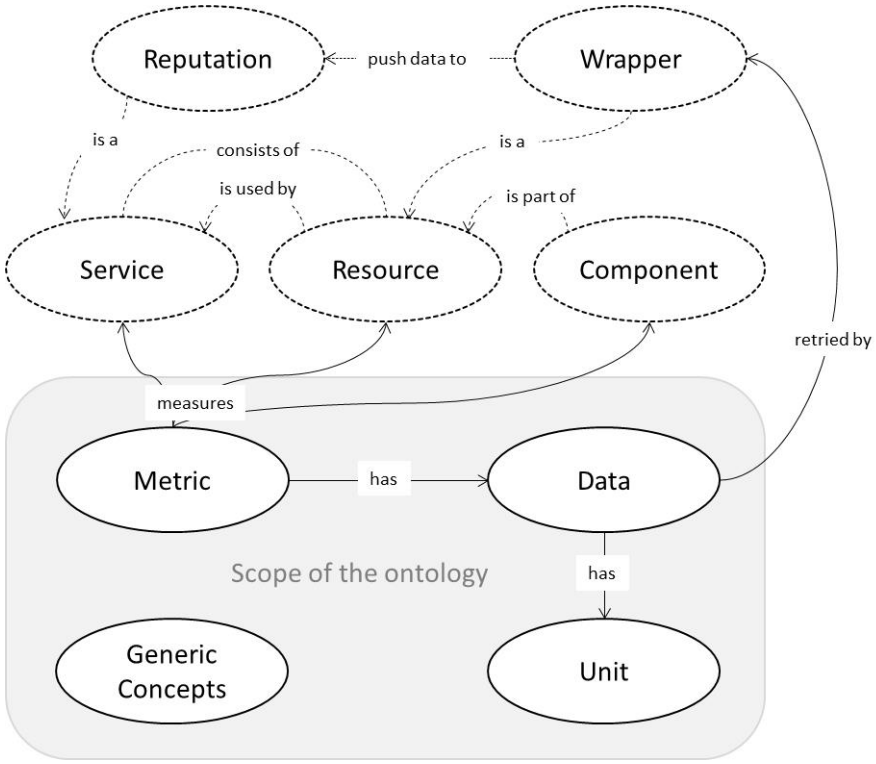

Figure 14 – Initial design of the Monitoring Ontology for Federated Infrastructures (MOFI) including its four models along with an example of their interactions with other external ontologies

The OWL is used to represent the ontology together with the RDF and RDFs that are together represent the data model we will use. We will use the RDF Turtle for data serialization.

3.4.3 Semantic OML

The target ontology-based information model discussed in previous section is implemented within the context of the OML framework allowing semantic-based data representation and exchange.

Initial steps toward implementing a semantic OML have been done within the context of the FIRE OpenLab project[12]. However, only a proof-of-concept release of OML that supports semantic was developed [55] without being really validated and used in real deployment. Figure 15 shows the architecture of the semantic OML. This release includes two main extensions:

- Based on the OML scaffold[13] component which is usually used to generate skeleton OML application source code (represented as OML clients) based on a template provided by the user. The scaffold is extended not only to generate suitable measurement points following the developed ontology in order to build the insert statements semantically (serialized as RDF triple). Further, it is extended to check the correctness of the used semantic schema with the used ontology.
- Extend the OML server through add-ons where the endpoint database is an RDF endpoint that supports SPARQL 1.1 Update [56]. The SPARQL server Fuseki[14] is used which allows serving RDF data over HTTP protocol, together with TDB Triple Store as RDF endpoint.

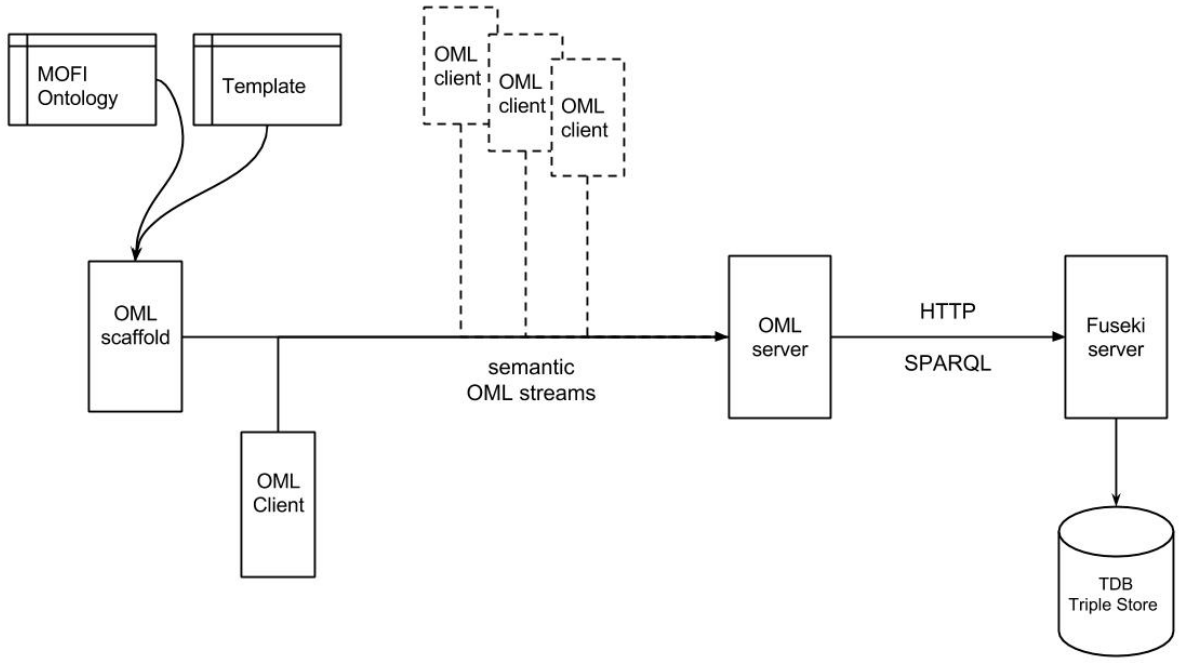

Figure 15 - Semantic OML

[12] www.ict-openlab.eu/project-info.html
[13] http://oml.mytestbed.net/doc/oml/latest/oml2-scaffold.1.html
[14] jena.apache.org/documentation/serving_data/

This implementation will be further developed not only to provide a stable and powerful release but also to support the semantic in further client libraries in other languages like Python and Ruby beside the currently available client library in C.

In addition, OML wrappers are used to collect measurement data, convert them into semantic data representations (RDF-serialized) and export them to the collection resources (OML server) following the developed ontology are to be implemented for the three main monitoring services (facility monitoring, infrastructure resources monitoring and experiment measuring).

3.4.4 User-friendly data access and visualisation

3.4.4.1 Implementation of OML query support in Manifold

A Manifold gateway implements the necessary logic to retrieve information from a particular source, using the endpoint defined transfer protocol, and aggregate the retrieved data with another source by the use of a common ID, i.e. the resource human readable name (hrn).

The OML gateway would implement the necessary logic to interact with an OML server and permit to query it and retrieve the monitoring information. The idea is to translate Manifold queries, which represent the global request made by the user, to SQL queries for the OML database. The information retrieved is transferred using the XML-RPC based protocol upon witch Manifold is based.

A first version of the OML gateway is implemented in Manifold and currently under development with additional features that will permit to easily include future changes in the OML implementation like authorisation and identity management.

3.4.4.2 Implementation of OML information display in MySlice

MySlice utilises Manifold as backend to retrieve information from different sources and then display it through a web based interface to the user. The main work in MySlice involves integrating the Manifold queries needed to interact with an OML server and creating the necessary views to display such information.

Those views can display monitoring information in the form of raw data or aggregated and displayed as a graph. Visualisation will be done by the use of HTML pages. MySlice already uses a Javascript tool called querytable that can be used to display data in the form of tables and permits various operations on it (ordering, filtering, searching etc.).

Querytable uses as data source format JSON, such format is well supported in Manifold and can be used with minor modifications on its structure for displaying measurement information. Graph visualisation can be achieved by using a Graph Javascript library as the Google Charts, already used in MySlice to display some statistical data.

3.5 Coordination

The OML is adopted by Fed4FIRE to provide a unified abstraction for the way data from heterogeneous monitoring and measurement tools are grouped together in meaningful sets, and represent these sets in a common manner to their consumers. However, the native OML does not support authentication of data streams. Further it was not designed for deployment in a federated environment where different parties report and access the collected data through shared deployment, and thus, does not support differentiated access control to storage backends. These limitations will be overcome through a new release (secure OML) provided by NICTA.

Further, although OML does not enforce any schemas to be used for representing the data, there is no standard way to represent the information. In the first and second cycles of Fed4FIRE, Fed4FIRE defined shared schemas used by the OML wrappers that gather data from local monitoring tools and export the data as OML streams following the commonly used schemas. This was still possible as we focused only on a set of measurement metrics required for the FLS about the health and status of facilities and required by federation services (SLA and reputation) about the availability of the used resources. In the third cycle, we expect a large number of various metrics that are required for experimenters and additional metrics for the federation services. To this end, in a complex and heterogeneous environment such as Fed4FIRE, there is a need to define a common information model covering the common monitoring and measurement concepts, terms, metrics and relationships. For this purpose, TUB will define an ontology-based information model that focuses on monitoring and measurement. However, it will rely on other ontologies developed within WP5 that model resources, components and services that will be monitored or even interested in having monitoring data. The monitoring ontology will be then implemented in the Fed4FIRE monitoring and measurement architecture, where special OML that supports the semantic together with semantic-oriented OML wrappers (that represent the data following the ontology model) and triple store database are implemented. TUB will lead this implementation.

So far, we presented two releases of OML (secure OML and semantic OML). But, TUB and NICTA will ensure having only one stable release of OML that support both security and semantic.

UTH will support testbeds on the usage, deployment of OML as well as on writing OML wrappers for their need.

Furthermore, as aforementioned, Fed4FIRE will allow experimenters to easily query their monitoring data and visualise them as graphs through the Portal. This will be implemented by UPMC. Through the Portal experimenters will have the ability to visualise their data collected either in SQL database or even in triple store semantic database.

4 Conclusion

This deliverable presents the specification as well as the design of the third, final cycle implementation of the Fed4FIRE measurement and monitoring architecture. This architecture is in line with the one defined in D2.4 [1] "Second federation architecture" and verified in D2.7 [57] "Third federation architecture" concerning monitoring and measurements.

In the third, final implementation cycle, the architecture will be extended to include three significant add-on features: i) secure data collection and reporting and differentiated access control, ii) representing the data following a common information and data model that is then implemented by the OML framework, and iii) user-friendly data access and visualisation capabilities.

Table 3 identifies the three main add-on features that are planned for the third, final implementation cycle of the Fed4FIRE monitoring and measurement architecture.

Functional element	**Implementation strategy**
Secure OML	• Develop a special release of OML that supports authenticated channels and differentiated access control (NICTA) • Deploy the secure OML if OML is not yet available or replace the native OML by this release (all participants with support from NICTA/UTH) • Adjust OML clients to be compliant with the new release (all participants with support from NICTA/UTH)
Common information and data model	• Develop a common information and data model inspired by the Semantic Web ontology techniques. A Monitoring Ontology for Federated Infrastructures will be developed that models the main, significant monitoring and measurement concepts, terms and their relationships (TUB) • Develop a semantic OML framework that implements the developed ontology (TUB)
Secure, semantic OML	• Provide one release of OML that supports both add-ons (security and semantic) (NICTA/TUB) • Deploy the secure, semantic OML (depends on the usage of ontologies in other Fed4FIRE work packages (WP5 and WP7), if RSpecs will be replaced by ontologies and some federation services will adopt ontologies, the

	associated parties should deploy this release with support from NICTA/TUB/UTH) • Accordingly, adjust the used OML clients to be compliant with the new release (as before the associated parties with support from NICTA/TUB/UTH)
User-friendly data access and visualisation	• Extend the Manifold that acts as a data broker in order to allow experimenters to query their data (UPMC) • Extend Fed4FIRE Portal, based on MySlice and Manifold, to visualise experimenters data graphically (UPMC)

Table 3 - Implementation strategy of the functional elements

References

[1] Fed4FIRE deliverable – D2.4 “Second federation architecture”. Available online at: http://www.fed4fire.eu/fileadmin/documents/public_deliverables/.

[2] Fed4FIRE deliverable – D6.2 “Detailed specifications for second cycle ready”. Available online at: http://www.fed4fire.eu/fileadmin/documents/public_deliverables/.

[3] Olivier Mehani, Guillaume Jourjon, Thierry Rakotoarivelo, and Max Ott, "An instrumentation framework for the critical task of measurement collection in the future Internet," Computer Networks, vol. 63, pp. 68-83, Apr. 2014. DOI:10.1016/j.bjp.2014.01.007. Available online at: http://www.nicta.com.au/pub?id=6718.

[4] A. P. Sheth et al., “Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases”, ACM Computing Surveys, vol. 22(3), pp. 183–236, 1990.

[5] Sheth, Amit P. and James A. Larson, "Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases", ACM Computing Surveys, vol. 22(3), pp.183–236, September 1990. DOI: 10.1145/96602.96604.

[6] Fed4FIRE deliverable – D6.1 “Detailed specifications for first cycle ready”. Available online at: http://www.fed4fire.eu/fileadmin/documents/public_deliverables/.

[7] Zabbix – open source monitoring system, available online at www.zabbix.com, last accessed on February 17, 2015.

[8] Nagios monitoring tool, available online at www.nagios.org, visited on February 17, 2015.

[9] Collectd, Available online at http://collectd.org/, last accessed on February 17, 2015.

[10] Fed4FIRE deliverable – D6.3 “Report on first cycle development”. Available online at: http://www.fed4fire.eu/fileadmin/documents/public_deliverables/.

[11] Multi-Hop Packet Tracking. Available online at: http://www.av.tu-berlin.de/pt, last accessed on February 17, 2015.

[12] Manifold Framework. Available online at: http://trac.myslice.info/wiki/Manifold, visited on February 17, 2015.

[13] TopHat. Available online at http://www.top-hat.info/, last accessed on February 17, 2015.

[14] MySlice. Available online at: https://myslice.info/home.

[15] Django is a high-level Python Web framework. Available online at: https://www.djangoproject.com.

[16] GENI RSpec v3. Available online at: http://groups.geni.net/geni/wiki/GENIExperimenter/RSpecs, last accessed on December 09, 2013.

[17] "OML Measurement Stream Protocol (OMSP)". Available online at: http://oml.mytestbed.net/doc/oml/2.11/doxygen/omsp.html.

[18] International Telecommunications Union, "Information technology – open systems interconnection – the directory: Public-key and attribute certificate frameworks," Recommendation X.509, Nov. 2012. Available online at: http://www.itu.int/rec/T-REC-X.509/en.

[19] Tim Dierks, Eric Rescorla, "The transport layer security (TLS) protocol version 1.2," Internet Requests for Comment, RFC Editor, Fremont, CA, USA, RFC 5246, Aug. 2008. Available online at: http://www.rfc-editor.org/rfc/rfc5246.txt.

[20]R. Housley and P. Hoffman, "Internet X.509 public key infrastructure operational protocols: FTP and HTTP," Internet Requests for Comment, RFC Editor, Fremont, CA, USA, RFC 2585, May 1999. Available online at: http://www.rfc-editor.org/rfc/rfc2585.txt.

[21]H. Tschofenig and P. Eronen, "Pre-shared key ciphersuites for transport layer security (TLS)," Internet Requests for Comment, RFC Editor, Fremont, CA, USA, RFC 4279, Dec. 2005. Available online at: http://www.rfc-editor.org/rfc/rfc4279.txt.

[22]ITIL V3 Glossary of Terms and Definitions, Office of Government Commerce (now UK Cabinet Office) (2007). Available online at: http://www.best-management-practice.com/gempdf/itil_glossary_v3_1_24.pdf.

[23]Weiner, J. 2007. “Measurement: Reliability and Validity Measures.” Mimeo, Bloomberg School of Public Health, Johns Hopkins University, Baltimore. Available online at: ocw.jhsph.edu/courses/hsre/PDFs/HSRE_lect7_weiner.pdf.

[24]A. Salvador, et al., “A Semantically Distributed Approach to Map IP Traffic Measurements to a Standardized Ontology”, International Journal of Computer Networks & Communications, pp. 13-31, vol. 2(1), 2010. ISSN 0975-2293.

[25]A. Pras and J. Schönwälder, “On the Difference between Information Models and Data Models”, RFC 3444, University of Twente, University of Osnabrueck, January 2003.

[26]A. Levy, “The Information Manifold Approach to Data Integration”, In IEEE Intelligent Agents, 1998.

[27]M.P. Reddy, et al., “A Methodology for Integration of Heterogeneous Databases”, IEEE Transactions on Knowledge & Data Engineering, pp. 920-933, vol. 6 (6), December 1994.

[28]Magdi N. Kamel and Moshe Zviran, “Heterogeneous Databases Integration in a Hospital Information Systems Environment: A Bottom-Up Approach”, AMIA.Inc, 1992, pp. 363.

[29]J. Zurawski, et al., “A Scalable Framework for Representation and Exchange of Network Measurements”, In Proceeding of the 2nd International IEEE/Create-Net Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (Tridentcom 2006), pp. 241-254, Barcelona, Spain, 2006.

[30]S. Gao, et al., “W3C XML Schema Definition Language (XSD) 1.1 Part 1: Structures”, W3C Recommendation 5 April 2012, available online at: http://www.w3.org/TR/xmlschema11-1/.

[31] A. Hanemann, et al., “Perfsonar: A Service Oriented Architecture for Multi-domain Network Monitoring”, In Proceeding of the Service-Oriented Computing (ICSOC 2005), Amsterdam, The Netherlands, December 2005, LNCS Vol. 3826, Springer Verlag, 2005. Avalable online at: https://www.perfsonar.net/.

[32]J.E. López de Vergara, et al., “Application of Ontologies for the Integration of Network Monitoring Platforms”, In Proceeding of the 1st European Workshop on Mechanisms for Mastering Future Internet, Salzburg, Austria, July 2008.

[33]ETSI GS MOI 010 V1.1.1 “Measurement Ontology for IP traffic (MOI); Report on information models for IP traffic measurement”. Available online at: http://www.etsi.org/deliver/etsi_gs/moi/001_099/010/01.01.01_60/gs_moi010v010101p.pdf.

[34]T. Berners-Lee, et al., “The Semantic Web,” Scientific American, vol. 284, no. 5, pp. 34–43, 2001.

[35]P. Spyns, et al., “Data modelling versus Ontology engineering”, ACM SIGMOD Record, vol. 31(4), pp. 12-17, December 2002.

[36]J.E. López de Vergara, et al., “Ontology-based Network Management: Study Cases and Lessons Learned”, Journal of Network and Systems Management, pp. 234-254 vol. 17(3), Springer, 2009.
[37]Moment Deliverable - D2.3 Monitoring and Measurement in the Next generation Technologies. Available online at: http://telscom.ch/wp-content/uploads/Moment/Deliverables/FP7-MOMENT-WP2-D2.3.pdf.
[38]R. Studer, et al., “Knowledge engineering: Principles and methods. Data & Knowledge Engineering”, vol. 25 (1-2), pp. 161-198, March 1998.
[39]P. Pin-Shan Chen, “The Entity-Relationship Model-Toward a Unified View of Data”, ACM Transactions on Database Systems, pp. 9-36, vol. 1(1), March 1976.
[40]STunnel. Available online at: https://www.stunnel.org, last accessed on March 5, 2015.
[41]G. Klyne, J. J. Carroll, and B. McBride. Resource description framework (RDF): Concepts and abstract syntax. W3C Recommendation. W3C, 2004.
[42]D. Brickley and R. V. Guha. Resource Description Framework (RDF) Schema Specification 1.0: W3C Candidate Recommendation 27 March 2000. Tech. rep. W3C, 1998.
[43]M. Andrea, et al., “Formalizing Knowledge by Ontologies: OWL and KIF”, CNR, IIT Department, Via Moruzzi 1, I-56124, Pisa, Italy.
[44]D. L. McGuinness and F. van Harmelen. OWL Web Ontology Language Overview. Tech. rep. W3C, 2004.
[45]M. Bergman, “Advantages and Myths of RDF”, AI3 - Adaptive Information, April 2009. Available online at: http://www.mkbergman.com/483/advantages-and-myths-of-rdf.
[46]G. Klyne, J. J. Carroll, and B. McBride. Resource description framework (RDF): Concepts and abstract syntax. W3C Recommendation. W3C, 2004.
[47]D. Brickley and R. V. Guha. Resource Description Framework (RDF) Schema Specification 1.0: W3C Candidate Recommendation 27 March 2000. Tech. rep. W3C, 1998.
[48]D. L. McGuinness and F. van Harmelen. OWL Web Ontology Language Overview. Tech. rep. W3C, 2004.
[49]liboml2.conf(5) – OML2 client library configuration file format. Available online at: http://omf.mytestbed.net/doc/oml/2.11/liboml2.conf.5.html.
[50]N. Casellas, “Methodologies, Tools and Languages for Ontology Design” Legal Ontology Engineering, Law, Governance and Technology Series, Springer, vol. 3, pp. 57-107, 2011. DOI: 10.1007/978-94-007-1497-7_3.
[51]A. Salvador, et al., “A Semantically Distributed Approach to Map IP Traffic Measurements to a Standardized Ontology”, International Journal of Computer Networks & Communications, pp. 13-31, vol. 2(1), 2010.
[52]J. van der Ham, et al., “The NOVI information models”, Future Generation Computer Systems, vol 42, pp. 64–73, January 2015.
[53]European Telecommunications Standards Institute (ETSI) Industry Group Specification MOI (Measurement Ontology for IP traffic), Version 003. Available online at: http://www.etsi.org/deliver/etsi_gs/moi/001_099/003/01.01.01_60/gs_moi003v010101p.pdf.
[54]M. Compton, et al., “The SSN ontology of the w3c semantic sensor network incubator group”, Web Semantics: Science, Services and Agents on the World Wide Web, pp. 25-32, vol. 17, 2012.
[55]Semantic OML code. Available online at: https://github.com/alco90/soml.
[56]SPARQL 1.1 Update. Available online at: http://www.w3.org/TR/sparql11-update/.

[57]Fed4FIRE deliverable – D2.7 “Third federation architecture”. Available online at: http://www.fed4fire.eu/fileadmin/documents/public_deliverables/.