Experimenter training

Wim Vandenberghe – Piet Demeester

May 7th 2013

Intended audience

- Researchers interested in experimentation on the Fed4FIRE facilities.
- Fed4FIRE will fund two open calls for such experiments.
  - Launch in May 2013 & 2014
  - Open to academia and industry
  - Two funding schemes defined:
    - Selected proposals join the project for one year, max. funding 80 k EUR / proposal
    - Select proposals do not join the project (typical for small experiments), max. funding 25 k EUR / proposal

Outline

- What is Fed4FIRE?
- Introduction to the common experimenter tools
- Introduction to the offered testbeds

What is Fed4FIRE?

- FP7 IP project: Federation for FIRE
- www.fed4fire.eu
- Project summary: Fed4FIRE will bring a common federation framework for Future Internet Research and Experiment facilities that will
  - be widely adopted by different communities (experimentation facilities, experimenters, academia, industry)
  - support powerful experiment lifecycle management (including tools for discovery and reservation, experiment control, measurements, etc.)
  - support key aspects of trustworthiness (federated identity management and access control, accountability, SLA management)

iMinds
CONNECT.INNOVATE.CREATE

What's in it for you (the experimenter)?

- Access a wide range of FIRE testbeds.
- Create experiments that break the boundaries of the different FIRE domains (wireless, wired, OpenFlow, cloud computing, smart cities, services, etc.)
- Easily access all the required resources with a single account.
- Focus on your core task of experimentation, instead of on practical aspects such as learning to work with different tools for each testbed, requesting accounts on each testbed separately, etc.

Fed4FIRE – general info

- Federation for FIRE
- IP project
- 10/2012 - 9/2016

- project coordinated by iMinds
- Total budget: 7.75 MEUR

Project partners

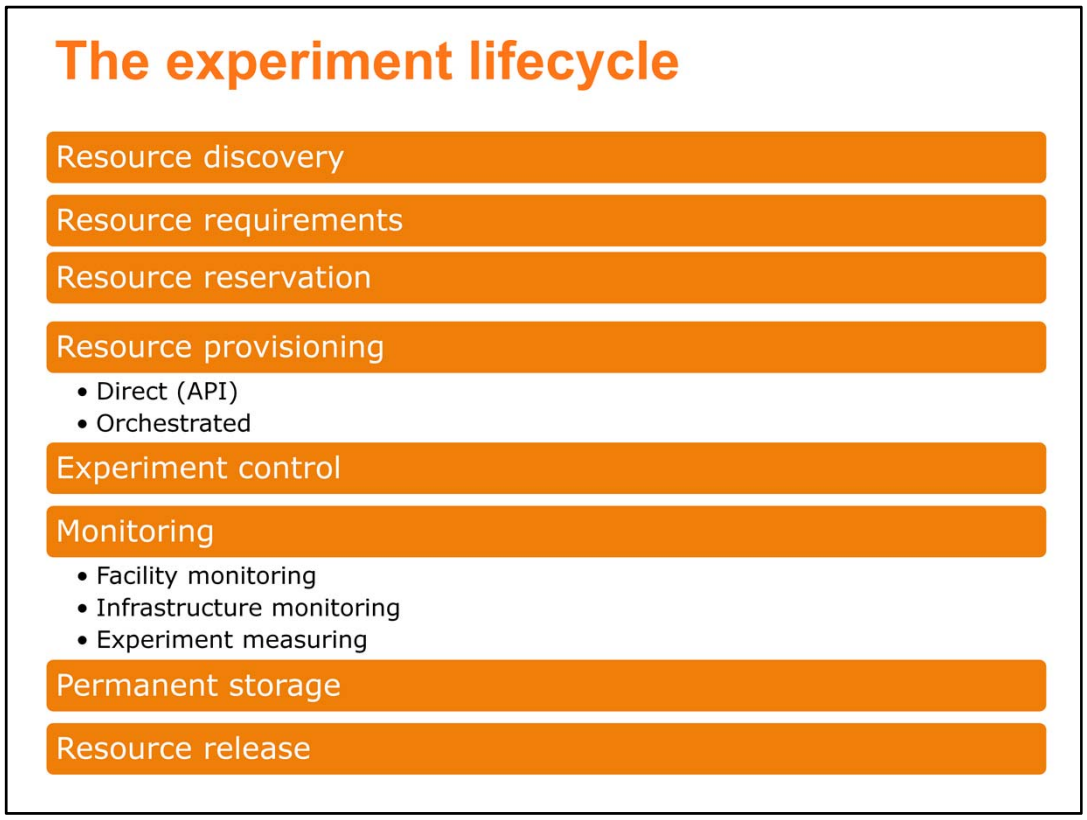
Experimentation Facilities

FED4FIRE

iMinds CONNECT.INNOVATE.CREATE

Outline

- What is Fed4FIRE?
- Introduction to the common experimenter tools
- Introduction to the offered testbeds



Resource discovery: Finding available resources across all facilities, and acquiring the necessary information to match required specifications.
Resource requirements: Specification of the resources required during the experiment, including compute, network, storage and software libraries.
Resource reservation: Allocation of a time slot in which exclusive access and control of particular resources is granted.
Resource provisioning

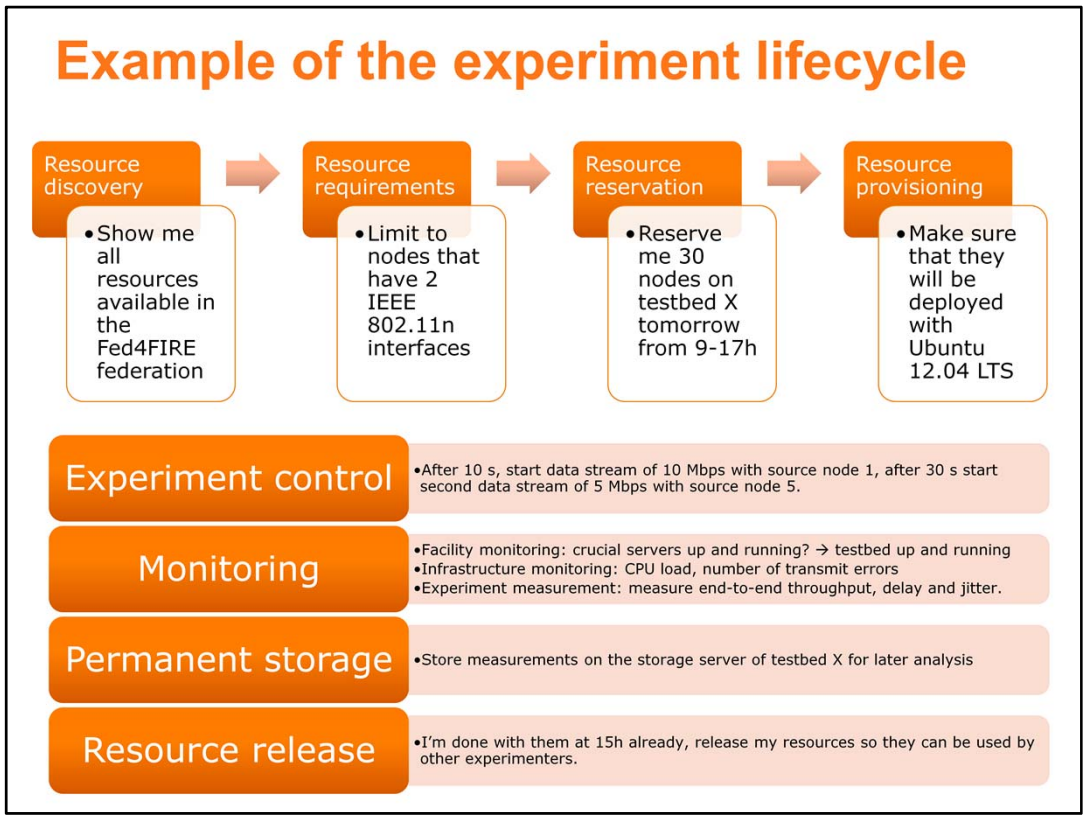
Direct (API): Instantiation of specific resources directly through the facility API, being the responsibility of the experimenter to select individual resources.

Orchestrated: Instantiation of resources through a functional component, which automatically chooses resources that best fit the experimenter's requirements.
Experiment control: Control of resource behavior during experiment execution, involving actions to query and modify resource state, and their correct sequencing.
Monitoring

Facility monitoring: Instrumentation of resources to supervise the behavior and performance of facilities allow system administrators or first level support operators to verify that facilities are performing correctly.

Infrastructure monitoring: Instrumentation of resources to collect data on the behavior and performance of services, technologies, and protocols to obtain measurements in the context of a concrete experiment.

Experiment measuring: Collection of experiment data generated by frameworks or services that the experimenter can deploy on its own.
Permanent storage: Storage of experiment related information beyond the experiment lifetime, such as experiment description, disk images and measurements.
Resource release: Release of experiment resources after deletion or expiration the experiment.



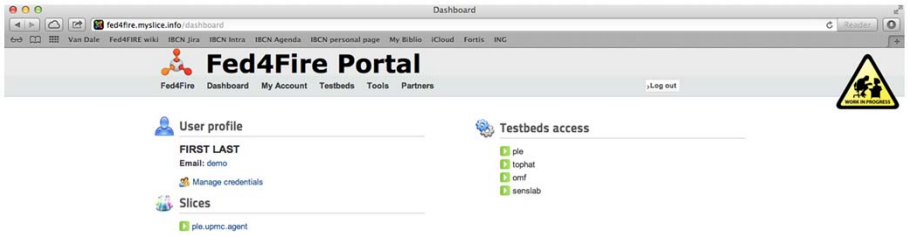
In this example an experimenter has developed a mechanism to automatically create a wifi mesh network (multi-hop network). The experimenter wants to test this at a larger scale, hoping to proof that the new solution can easily forward multiple streams at the same time without sacrificing any performance.

Common experimenter tools (1)

- Discovery, reservation, provisioning
  - Common interface = SFA
  - Tools supported by all testbeds:
    - Fed4FIRE portal
    - Flack
    - Omni
    - SFI

Common tool: Fed4FIRE portal

- Provide information
  - Testbed & tools directory
  - Links to the project website
  - Links to the First Level Support service
- Support registration of new users
- Act as an experimentation tool
  - Discovery, reservation & provisioning
  - Bridge to experiment control tools
  - Will work with most of the Fed4FIRE testbeds in the 1st round of open call experiments



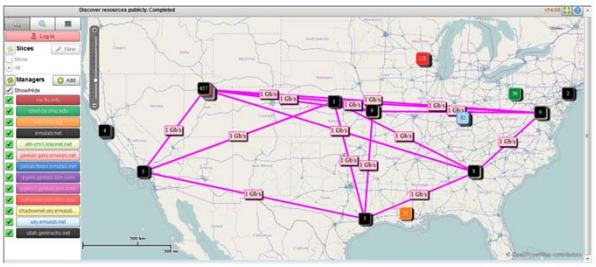
The portal is currently under development and will be a central starting point to access the Fed4FIRE federation. The portal will provide information to the experimenter in several forms. The testbed directory can be considered to be a catalogue, providing a high-level overview of all federated testbeds and their capabilities. The portal will also provide pointers for the experimenters to the project website and to the First Level Support systems (e.g. Trouble Ticket System), if they need help to register or use the portal or encounter problems while setting up experiments.

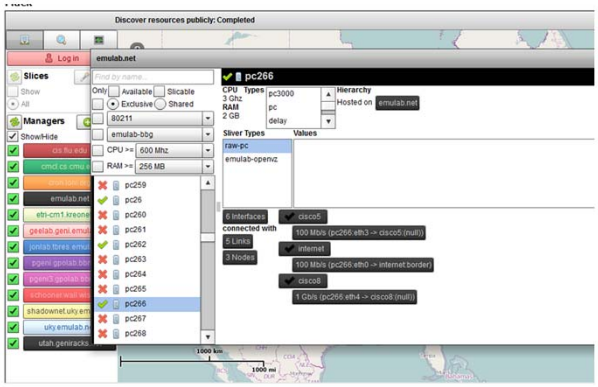
The portal will also be the registration place for new experimenters. Therefore, it will provide an easy way for experimenters to register themselves and to access the federated testbeds. Note however that the testbeds will always determine whether the user can actually access them according to their access policies.

Moreover, the portal will also perform the role of a client tool. On behalf of the experimenter, the portal will forward queries to federated testbeds using SFA delegation mechanism. Using the portal, the experimenter will be able to search, browse and reserve resources across federated testbeds. The portal will also act as a bridge to experiment control tools.

Common tool: Flack

- Visual SFA client for
  - Discovery
  - Provisioning
  - Slice management
    - Create
    - Open
    - Update
- Will work with most Fed4FIRE testbeds in the 1st round of open call experiments

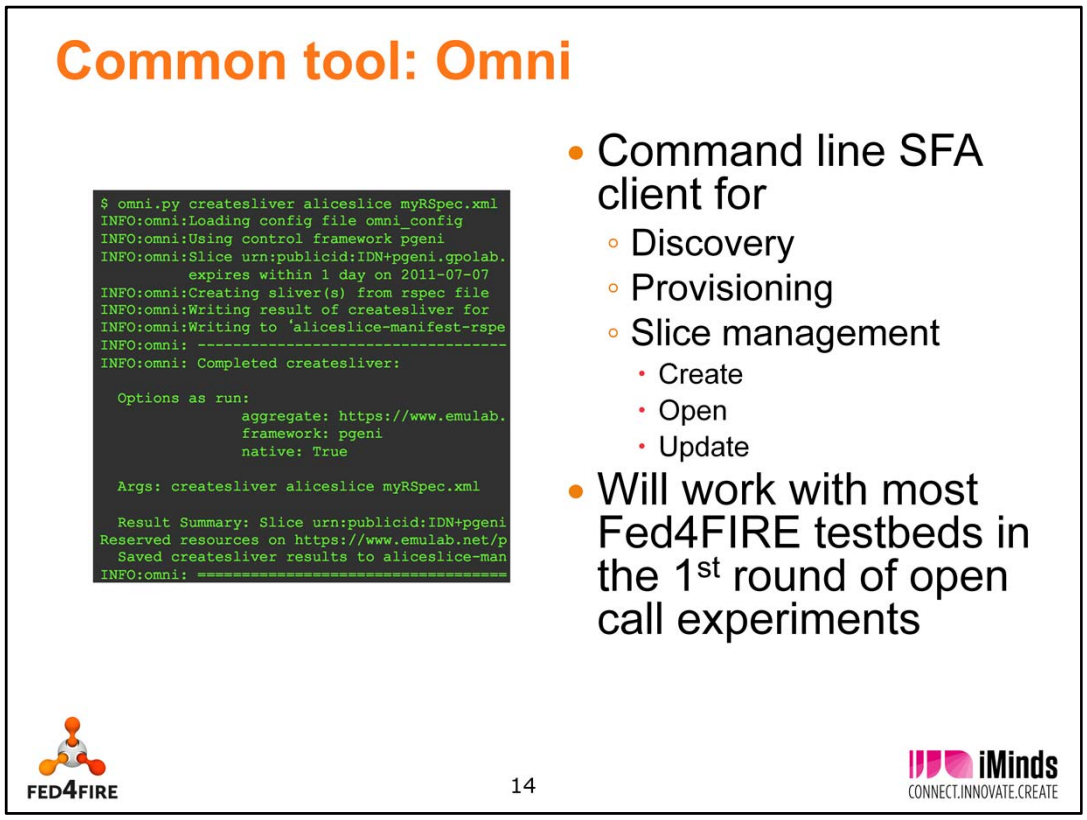




Flack (http://www.protogeni.net/wiki/Flack) is a piece of software developed in the GENI project (USA). It is a visual client for SFA based aggregate managers, such as the ProtoGENI and federated GENI aggregate managers. Flack covers the main functionalities of authentication, discovery and resource provisioning over SFA compliant testbeds. It also allows the experimenter to create, open and update slices.

In theory, Flack should be able to handle Fed4FIRE testbeds out of the box, since both rely on SFA. However, this will be verified at the moment that the testbeds have deployed their SFA interface during the course of development. It is not inconceivable that Flack should be slightly updated to cope with possible implementation or versioning issues that emerge when Flack is tested against the different Fed4FIRE testbeds. If such needed small updates of Flack would be identified, then they will be implemented in Fed4FIRE. This will allow the usage of the Flack tool together with the Fed4FIRE testbeds in the first round of open call experiments.

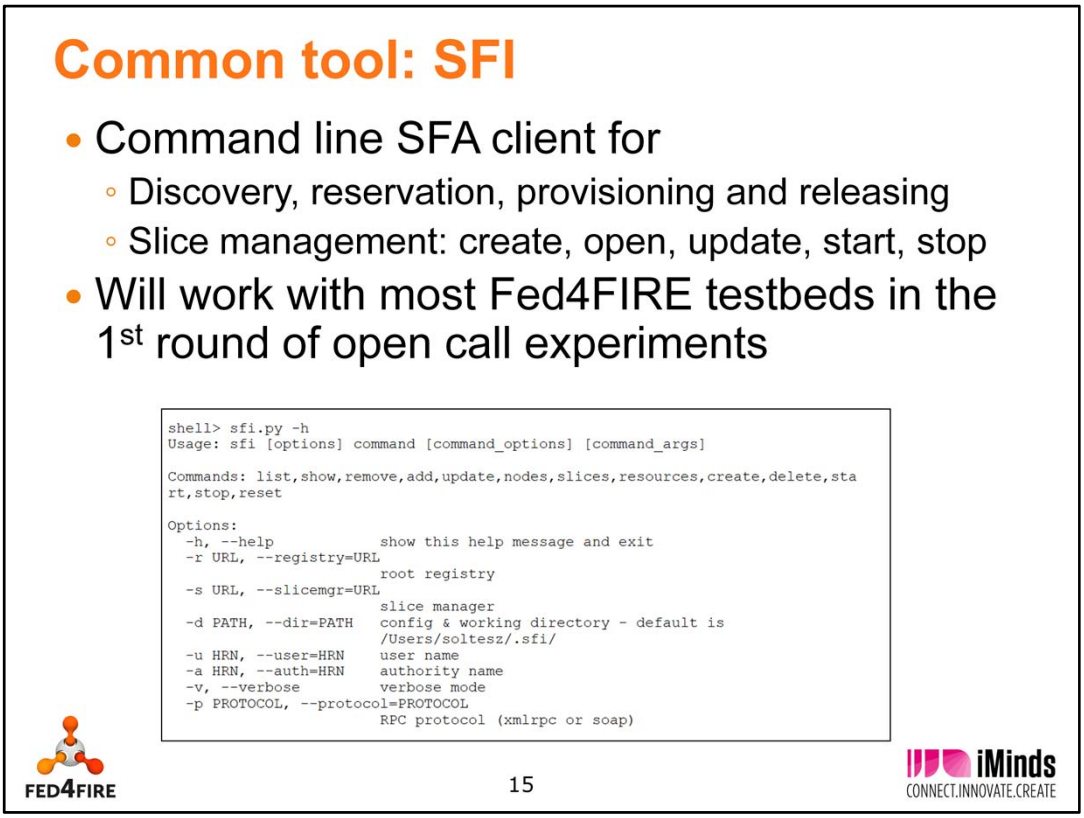
The top figure depicts the high level resource discovery that can be done on a map view. The bottom figure depicts the detailed resource discovery that is possible per site.



Omni (http://trac.gpolab.bbn.com/gcf/wiki/Omni) is a GENI command line tool for discovering and provisioning of resources at GENI Aggregate Managers (AMs) via the GENI AM API. The Omni client also communicates with Clearinghouses (also known as Control Frameworks or CFs) to create slices, and enumerate available GENI AMs. A Clearinghouse is a framework of resources that provides users with GENI accounts (credentials). Users can use these credentials to reserve resources in GENI AMs. Any AM API compliant aggregate should work with Omni. These include SFA, ProtoGENI, OpenFlow and GCF.

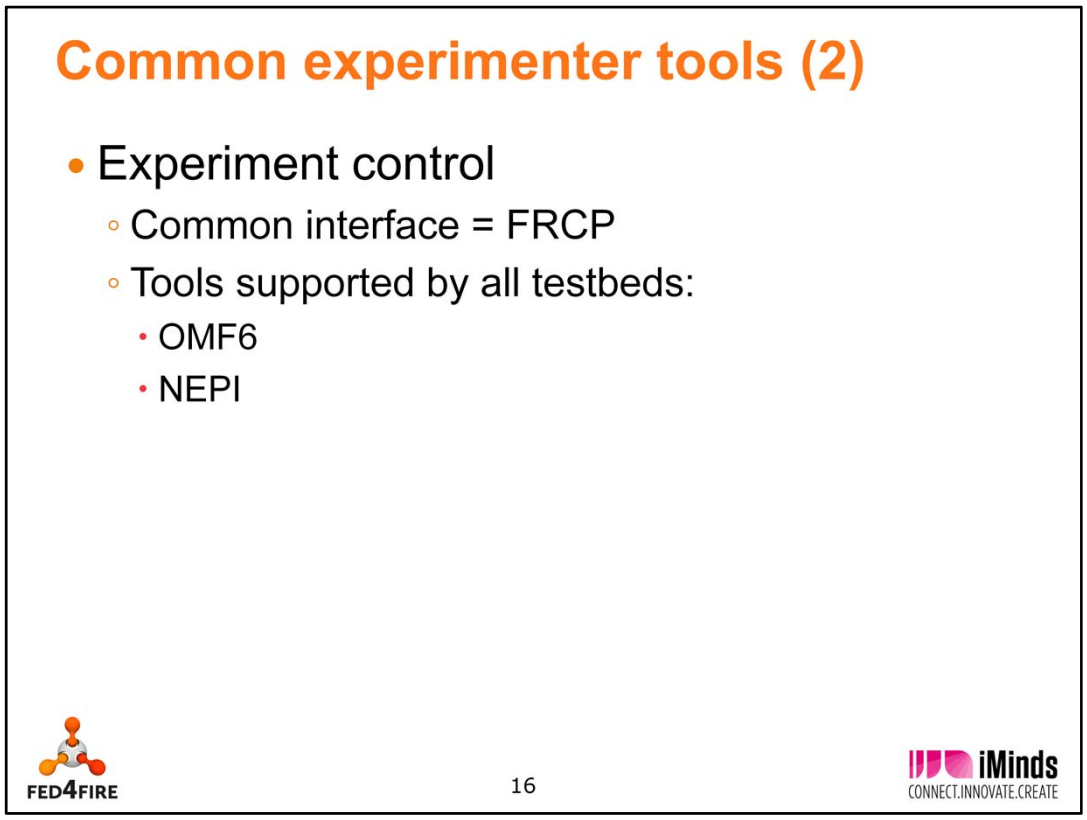
In theory (and similar to the Flack case), Omni should be able to handle Fed4FIRE testbeds out of the box, since both support SFA. However, this will be verified at the moment that the testbeds have deployed their SFA interface during the course of development. It is not inconceivable that Omni should be slightly updated to cope with possible implementation or versioning issues that emerge when it is tested against the different Fed4FIRE testbeds. If such needed small updates would be identified, then they will be implemented. This will allow the usage of the Omni tool together with the Fed4FIRE testbeds during the first round of open call experiments.

The figure depicts a screenshot of an experimenter using Omni



SFI is another command line client for SFA interfaces. It is implemented in python as part of the (freely available) PlanetLab implementation. It provides the functionality to create, update and display a slice. SFI also supports resource discovery, reservation and provisioning. It can also be used to release resources, and to start and stop a slice. To illustrate how SFI is to be used, the help page of the tool is depicted in the figure.

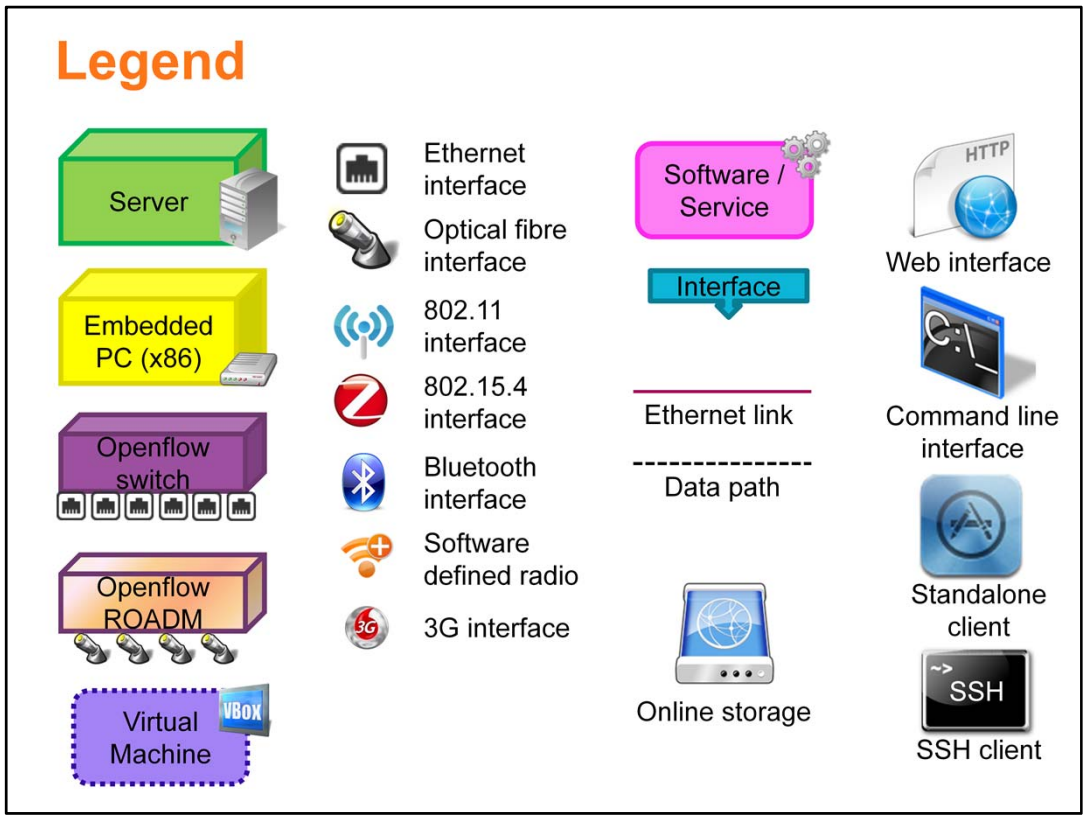
In theory (and similar to the Flack and Omni cases), SFI should be able to handle Fed4FIRE testbeds out of the box, since both support SFA. However, this will be verified at the moment that the testbeds have deployed their SFA interface during the course of development. It is not inconceivable that SFI should be slightly updated to cope with possible implementation or versioning issues that emerge when it is tested against the different Fed4FIRE testbeds. If such needed small updates would be identified, then they will be implemented. This will allow the usage of the SFI tool together with the Fed4FIRE testbeds during the first round of open call experiments.

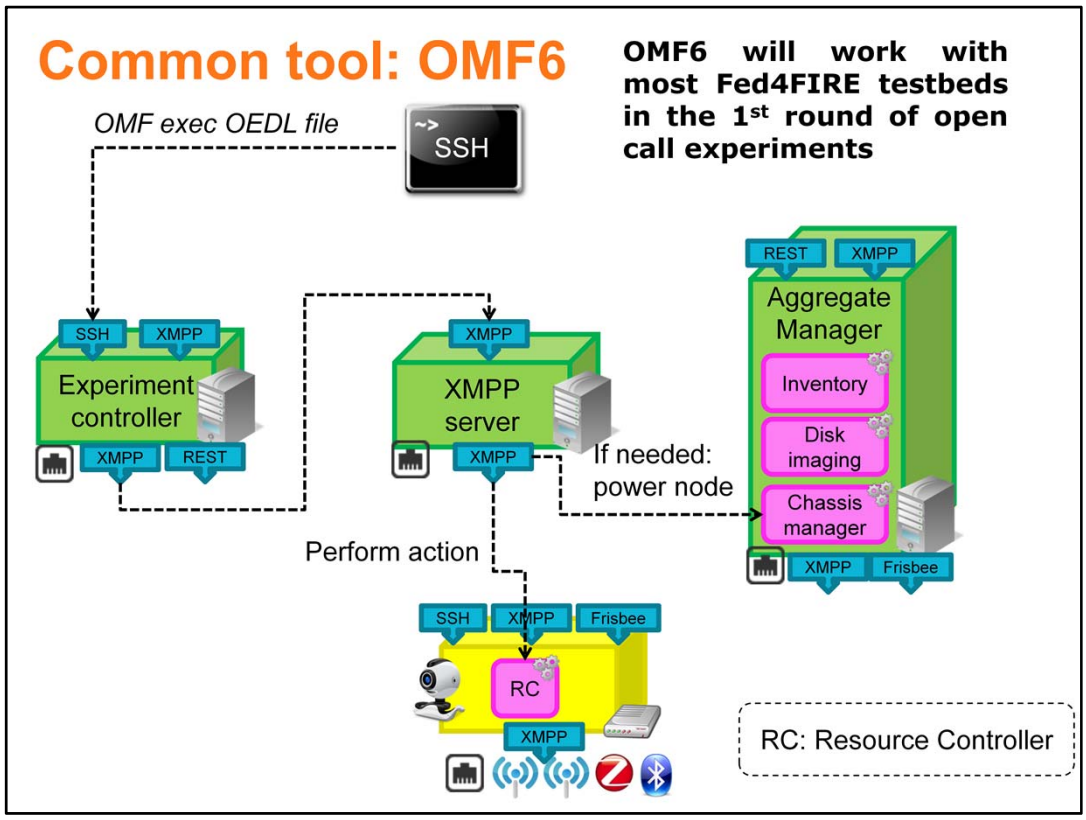


A standardized resource control protocol will permit to control resources provided by federated facilities using different management software in a uniform way.

The novel federated resource control protocol (FRCP) is such a protocol. It consists of a message being sent by a requester to a component (or resource). The component may accept the message and perform the requested associated actions. For the message exchange with the resources (physical or application resources), the necessary resource controller (RC) implementation, supporting the set of defined messages, should be running in the different resources of the facility. FRCP is adopted by Fed4FIRE as the common interface for experiment control.

FRCP is currently being developed by NICTA in the context of OMF6. Since it is a standardized resource control protocol, for which the syntax and semantics of the used messages is open, different experiment control engines can be developed that will rely on FRCP for the actual execution of their needed actions on the actual resources. Such an engine is a user tool where the user can describe experiments, and which is responsible for the experiment orchestration. OMF6 provides its own engine to be used together with FRCP, NEPI is another experiment control engine that is supported by the Fed4FIRE federation.


Legend
Server
Embedded PC (x86)
Openflow switch
Openflow ROADM
Virtual Machine
VBOX
Ethernet interface
Optical fibre interface
802.11 interface
802.15.4 interface
Bluetooth interface
Software defined radio
3G interface
Software / Service
Interface
Ethernet link
Data path
Online storage
HTTP
Web interface
Command line interface
Standalone client
SSH
SSH client



OMF (http://omf.mytestbed.net) is a generic framework which allows the definition and orchestration of experiments using shared (already provisioned) resources from different federated testbeds. OMF was originally developed for single testbed deployments, but has recently been extended to support multiple deployments and the following features. First, it provides a domain-specific language based on an event-based execution model to fully describe even complex experiments (OEDL). OMF also defines a generic resource model and concise interaction protocol (FRCP), which allows third parties to contribute new resources as well as develop new tools and mechanisms to control an experiment (as mentioned on the previous slide). It has a distributed communication infrastructure based on XMPP supporting the scalable orchestration of thousands of distributed and potentially disconnected resources.

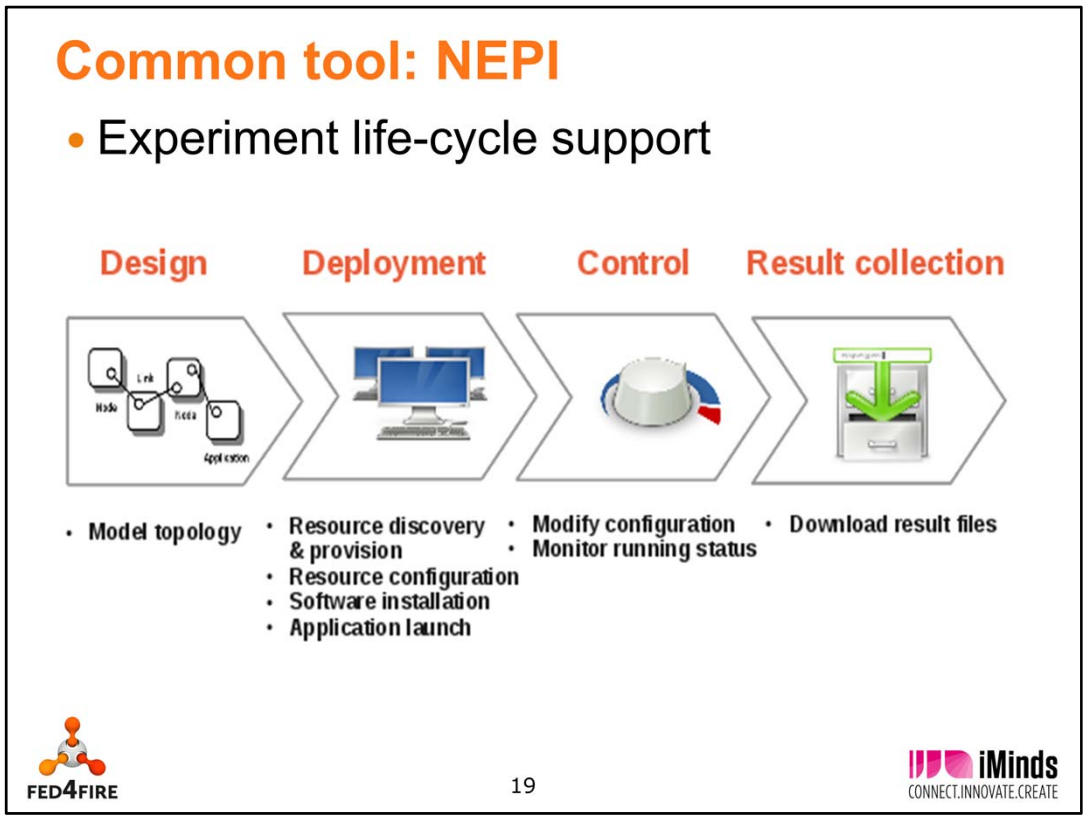
But how does this OMF experiment control work? This is depicted on the slide. At the bottom the resources are depicted, in this case an embedded PC with several wired and wireless connections. On the OMF layer, three entities can be observed. The Aggregate Manager (AM) is responsible for the inventory, disk imaging and chassis management (powering nodes on or off when needed. The second OMF management entity is the Experiment Controller (EC). It processes a description of the experiment scenario (written in the OEDL language), and will automatically trigger the right actions at the right nodes when needed. Although it is part of the OMF management layer from a logical point of view, the EC can both be provided as a service by the testbed, and can be run locally by the experimenter on its own PC. To perform these actions at the resource, the EC will send a message to a daemon running an every resource: the Resource Controller (RC). The RC is capable of executing everything what a user could do manually on the command line. It can also make abstraction of certain commands by wrapping them in OMF commands. An example is the OMF command to change the Wi-Fi channel. Behind the curtains it will determine which wireless driver is used on the resource, and will then select the suitable set of command line commands to execute, depending on the driver. To support this messaging between EC and RC, an XMPP service is used. Hence a XMPP server is added as the third entity of the OMF control framework. The protocol used by the EC to request actions at the different RCs is FRCP.

So the experiment control is executed as follows:

1. The experimenter gains access to the PC that runs the EC (his own PC, or a server at the testbed that is reachable through SSH)
2. The experimenter starts the experiment control procedure by giving the command “OMF exec”. The name of the scenario description that is to be executed is given as an argument. The EC will process this description, and will initiate specific commands at certain resources at the appropriate time.
3. If the target node is powered off, the EC will call the chassis manager service of the AM to power it on. For this it will send an appropriate XMPP message.
4. Once the target node is powered on, the EC will request the RC running at the desired resource to perform a certain command. As mentioned, this can be any command that could also be given on the command line manually. To trigger the RC, an XMPP message is sent from the EC to the RC. This message is compliant with the FRCP protocol.

It is a common misassumption that OMF and OML are the same things, while in fact they are not. OMF is the framework for provisioning and experiment control as described above. OML is an additional software suite targeting experiment monitoring. They are very often deployed together in testbeds, but this is not a strict requirement. From

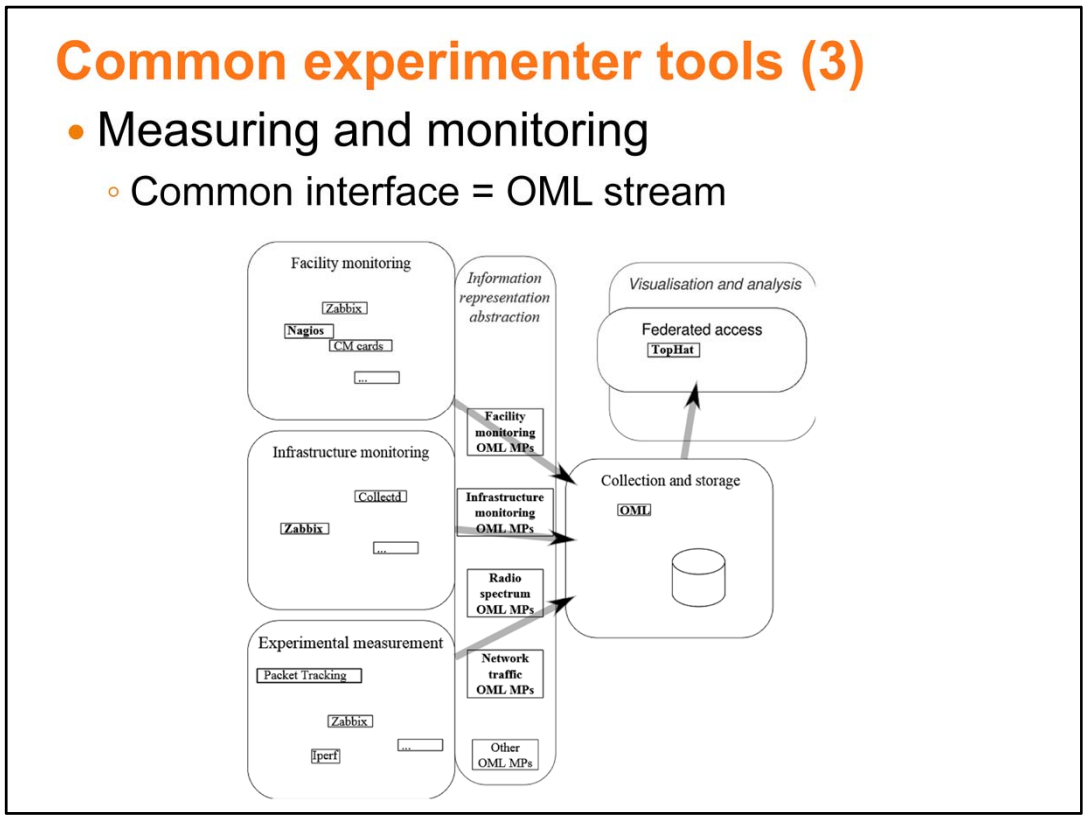
a logical point of view they can be considered as two separate entities. You can run OMF without running OML, and vice versa.



NEPI (http://nepi.inria.fr), the Network Experimentation Programming Interface, is a life-cycle management tool for network experiments. The idea behind NEPI is to provide a single tool to design, deploy, and control network experiments, and gather the experiment results. Going further, NEPI was specially conceived to function with arbitrary experimentation platforms, so researchers could use a single tool to work with network simulators, emulators, or physical testbeds, or even a mixture of them. To accomplish this, NEPI provides a high-level interface to describe experiments that is independent from any experimentation platform, but is able to capture platform specific configurations. Experiment definitions can be stored in XML format to be later reproduced, and modified according to experimentation needs. Experiment execution is orchestrated by a global experiment controller, that is platform independent, and different platform-dependent testbed controllers, creating a control hierarchy that is able to adapt to platform specific requirements while providing an integrated control scheme.

The initial focus of NEPI in Fed4FIRE is to support design, and control through the federated resource control protocol FRCP. This functionality will be available during the first round of open call experiments. To support the whole experiment life cycle, the appropriate course of actions will be defined during the course of the project, when authentication, authorization, discovery, provisioning, reservation and measurements according with other project Work Packages are in more mature state.

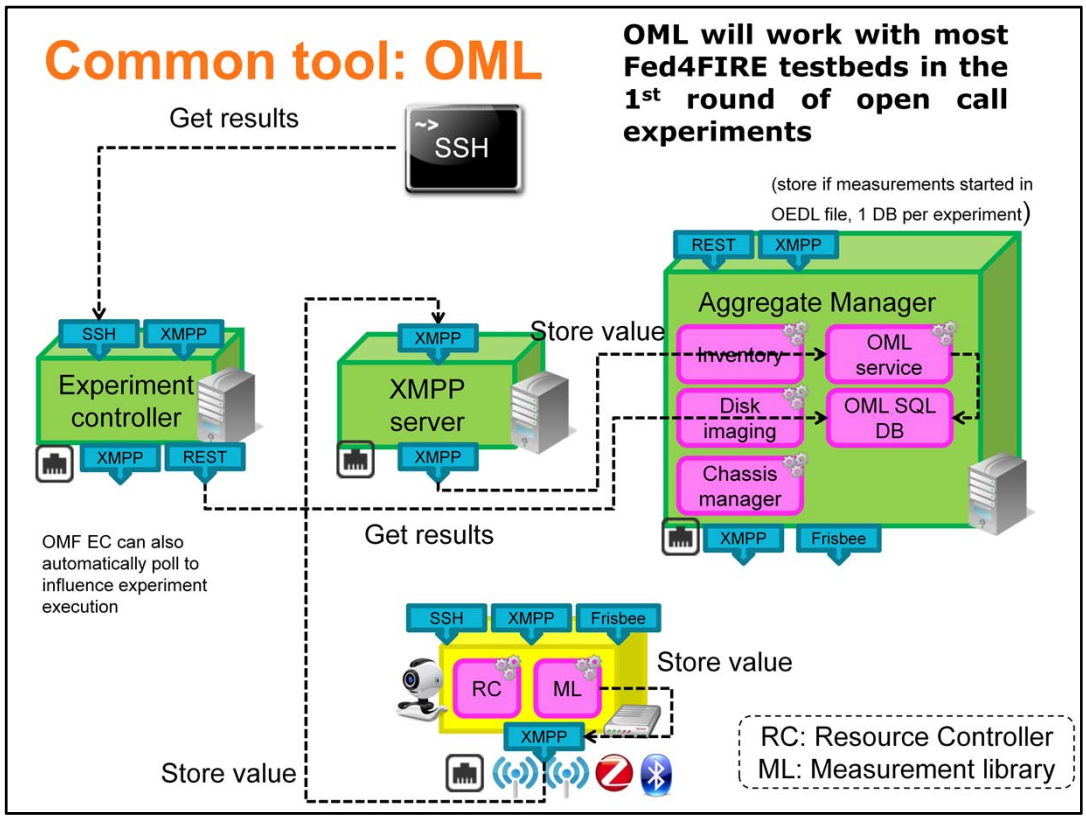
NEPI uses a *Boxes and Connectors* modelling abstraction to construct the experiment design. Each supported experimentation platform defines a set of *box* types, which represents the conceptual constructive blocks of an experiment. These boxes can be associated through named ports called *connectors*. Each connector in a box has a different function. The experiment description is thus constructed out of a graph which has *Boxes* as vertex. The boxes present in the experiment description and the connections between those boxes will define the experiment topology, both at a physical (infrastructure) and application (services) levels. Boxes also have a set of *attributes* that allow defining the experiment configuration, and *traces* that allow defining experiment results to be collected.



There is a large variability in the tools that where already deployed in the past on the various testbeds belonging to the Fed4FIRE consortium. A few commonalities could be found, mostly in the monitoring solutions, where Nagios, Zabbix and Collectd are primarily used. On the experimental measurement side the variability shows the most, with a lot of different and sometimes ad hoc tools. This disparity however can be solved through the use of a middleware measurement system in charge of reporting samples from heterogeneous distributed tools in a unified and centralisable way. Here, a commonality was identified around OML (http://oml.mytestbed.net/projects/oml/wiki). Its lightweight API is also a good match for the instrumentation of the various measurement tools in use.

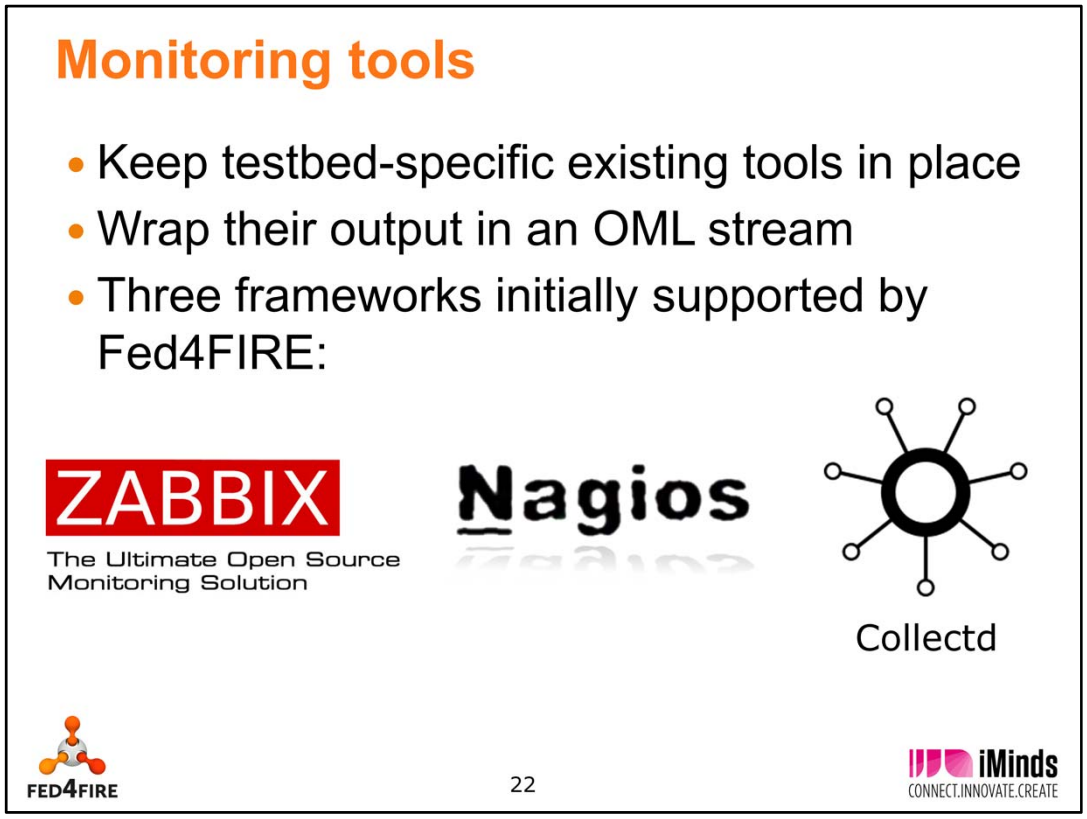
Therefore it was decided that in Fed4FIRE the common interface for measuring and monitoring will be OML. This use of OML as a collection and reporting framework allows instrumenting any sort of system through the abstraction of a measurement point, describing a group of metrics. This abstraction allows more latitude in the choice of measurement tools: as long as they conform to the same measurement point for the same information, their output can be used indistinctly in further analysis. Selecting OML for reporting purposes therefore allows flexibility in the choice of measurement tools, both for monitoring and measurement tasks, as well as for a unified way to access the collected data.

So in practice existing infrastructure and monitoring frameworks that are already in place (e.g. Zabbix, Nagios or Collectd) are kept at the Fed4FIRE testbeds. However, their measurement data is packaged in a common format: the OML stream. Besides, all testbeds adopt OML on their resources to allow experimenters to easily collect experiment measurements. Experimenters can then basically do whatever they want with the OML streams that come out of their experiment: persist them in a database, visualise them, etc.



OML (http://oml.mytestbed.net/projects/oml/wiki) is a generic framework that can instrument the whole software stack, and take input from any sensor with a software interface. It has no preconception on the type of software to be instrumented, nor does it force a specific data schema. Rather, it defines and implements a reporting protocol and a collection server. On the client side, any application can be instrumented using libraries abstracting the complexity of the network communication. Additionally, some of the libraries provide in-band filtering, allowing to adapt the measurement streams obtained from an instrumented application to the requirements of the current observation (e.g., time average rather than raw metrics). Applications for which the code is not available can also be integrated in the reporting chain by writing simple wrappers using one of the supported scripting language (Python or Ruby). After collection from the distributed application, the timestamped data is stored in an SQL database (SQLite3 or PostgreSQL), grouped by experimental domain; a server can collect data from several domains at the same time.

As mentioned before, OMF and OML are separate entities from a logical point of view: you can run OMF without running OML, and vice versa. But their corresponding software libraries are installed on the same entities. As depicted on the slide, OML consists of a service running on the resource, and a service and database running on the Aggregate Manager (AM). On the resource, the Measurement Library (ML) takes measured values as an input, and is responsible for getting them added to the database at the AM. It does so by calling the OML service running at the AM. Again XMPP messaging is applied. Annotations to the measured values such are experiment ID, source ID and so on are automatically added by the OML framework. From an experimenter point of view, it is sufficient to redirect the measured value coming out of your own software or measurement tool to the ML to collect all of them in a single place for future processing.



As already mentioned, there is a large variability in the tools that where already deployed in the past on the various testbeds belonging to the Fed4FIRE consortium. A few commonalities could be found, mostly in the monitoring solutions, where Nagios, Zabbix and Collectd are primarily used. Therefore it was decided to keep these systems in place, and wrap their measurements in OML streams.

Zabbix is an open-source solution for facility and infrastructure monitoring. It supports performance monitoring natively, in addition to facility monitoring and alerting. It also supports an extensive list of operating systems and platforms, including virtual machines. Three types of resource components are available for monitoring: native Zabbix agents, SNMP monitoring, and agentless script-based queries; all data is then aggregated within a central collection server which relies on SQL databases (MySQL or PostgreSQL) for storage.

Nagios is another open-source base for infrastructure monitoring solutions. Unlike Zabbix, it does not support performance monitoring natively. It provides status reports for hosts, applications and networks. Nagios can also be extended through the use of user scripts run by the Nagios Remote Plugin Executor (NRPE). It has built-in support for raising alerts on problematic situations. Data is stored in an *ad hoc* backend, but some plugins allow export to SQL databases. Data can also be processed and exchanged between instances using the Nagios Remote Data Processor (NRDP).
Bothtools can be extended through the use of plugins, and plugins written for Nagios are also reusable with the Zenoss monitoring tool.

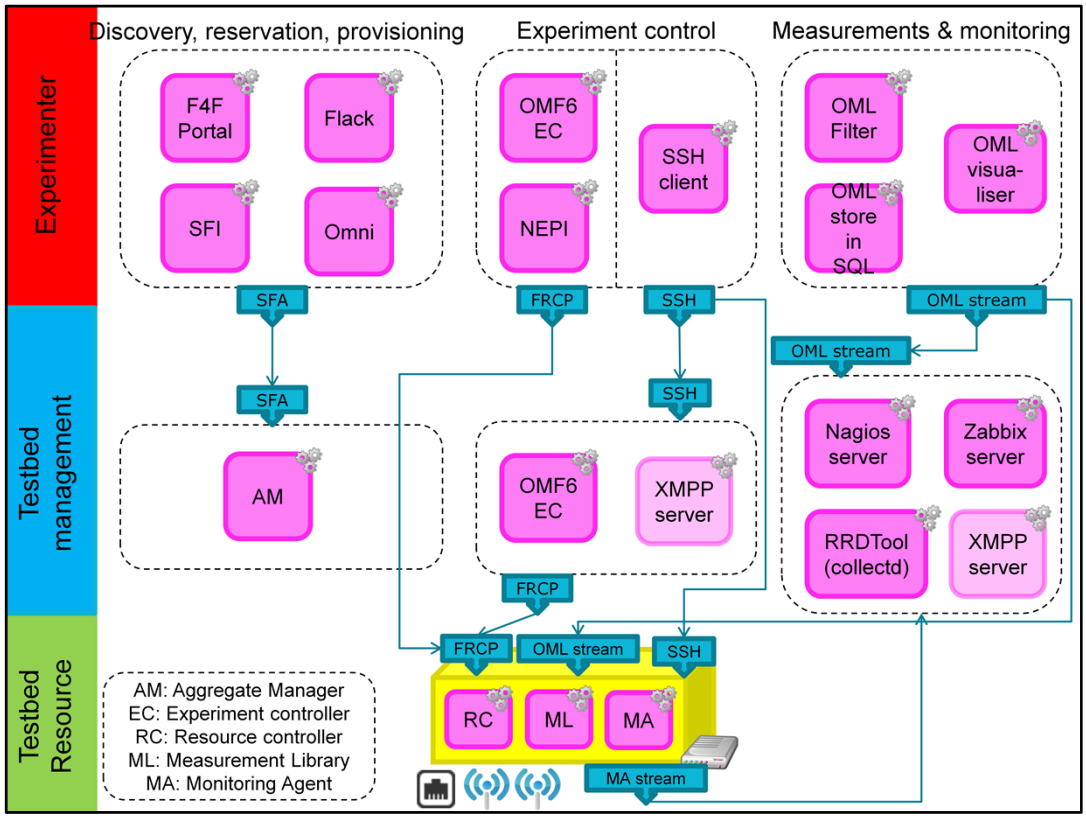
Collectd is an extensible daemon that collects and stores performance data from network equipment and computers it is running on. Through the use of plugins, it can be extended to monitor various aspects of the nodes, such as various common application servers and specific system metrics. It can also query information from SNMP-enabled nodes, and through the libvirt plugin, monitor guest virtual machines. Its default storage backend relies on RRDTool, from which time-based graphs can be generated from external tools. However, a writer plugin is available which makes the data available through OML.

Common experimenter tools (4)

- Security
  - Common interface = SFA X.509 certificates
  - Two approaches possible
    - Affiliated with a Fed4FIRE testbed: use the tools provided by your testbed to create a valid certificate
    - Not affiliated with a Fed4FIRE testbed: will need to register as a new Fed4FIRE user on the portal. In that case the portal will provide the possibility to automatically create a valid certificate for you.

Summary of common experimenter tools

- Three types of tools supported:
  - Discovery, reservation, provisioning
  - Experiment control
  - Measurements & monitoring
- For each type, the experimenter has different options to choose from



As mentioned in the previous slide, there are 3 types of experimenter tools supported, and for each of them the experimenter has several options to choose from.

For resource discovery, reservation and provisioning, the experimenter can choose to use the Fed4FIRE portal, Flack, Omni and SFI. All these tools use the SFA interface to talk to the specific testbed management components (called Aggregate Managers).

For Experiment control, the experimenter can run its own instance of an experiment controller. This can be the OMF6 Experiment Controller, or NEPI. As an alternative, the testbed could choose to deploy its own instance of the OMF EC. In that case the experimenter can log in to that instance using its local SSH client, and run the experiment control at the testbed. As a final option, the user can also directly log in on the nodes using its local SSH client, and perform experiment control actions manually using the console on the resource.

For measurement and monitoring, the experimenter can use several OML tools: filters, persistence tools (store in SQL database) and visualization tools. In case of experiment measurement, these tools will use OML streams that are directly originating from the OML measurement library component that is deployed on the resource. For facility and infrastructure monitoring, the testbed will wrap its existing monitoring infrastructure (being Nagios, Zabbix or Collectd) in an OML stream, which will be used by the OML experimenter tools.

To support all these tools, it is required that testbeds expose their management software through the SFA interface, that they deploy OMF6 experiment control on their testbed, and that they provide one of the mentioned monitoring frameworks and wrap its output in OML streams. On the resources three agents have to be deployed: an OMF resource controller, an OML measurement library and the appropriate monitoring agent that corresponds with their adopted

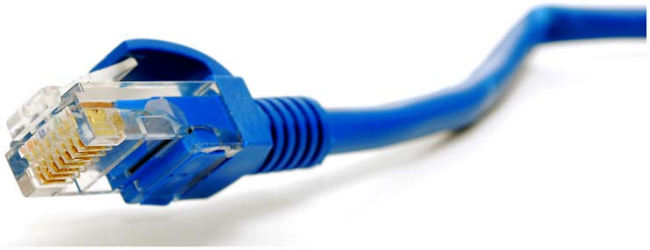
monitor framework.

Outline

- What is Fed4FIRE?
- Introduction to the common experimenter tools
- Introduction to the offered testbeds

The wired testbeds

- Virtual Wall (iMinds)
- PlanetLab Europe (UPMC)



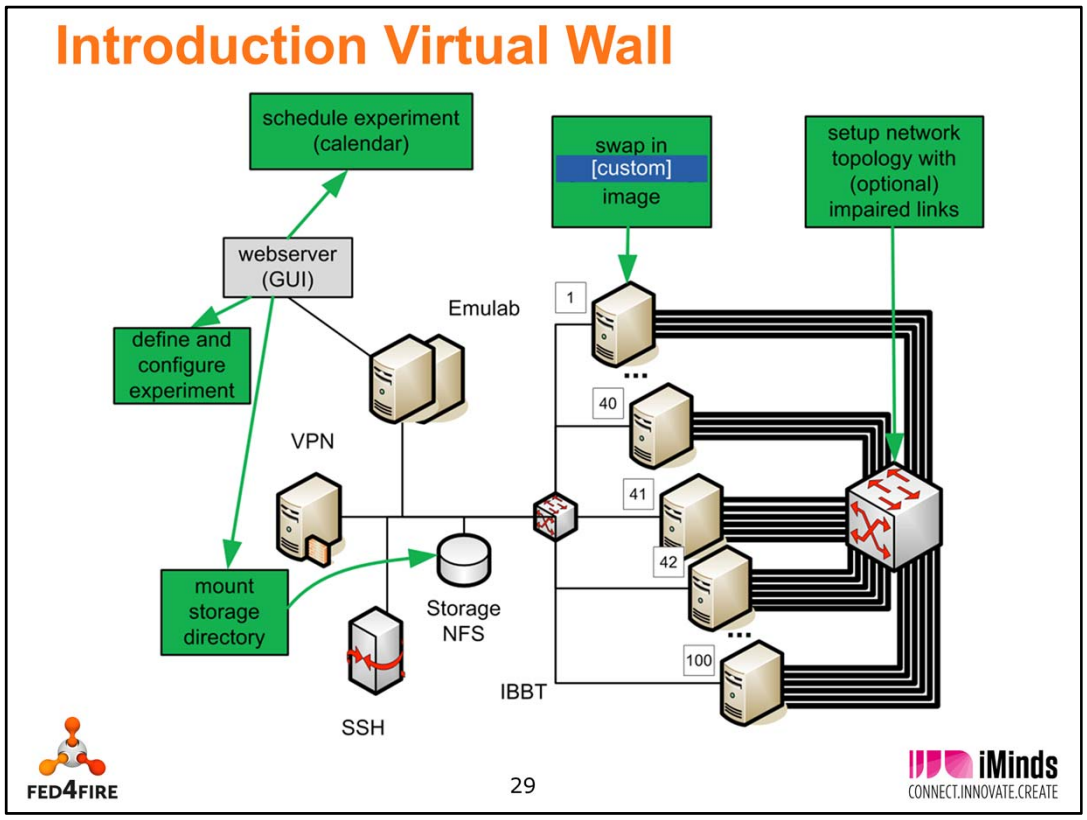
FED4FIRE

iMinds
CONNECT.INNOVATE.CREATE

Virtual Wall testbed (iMinds)

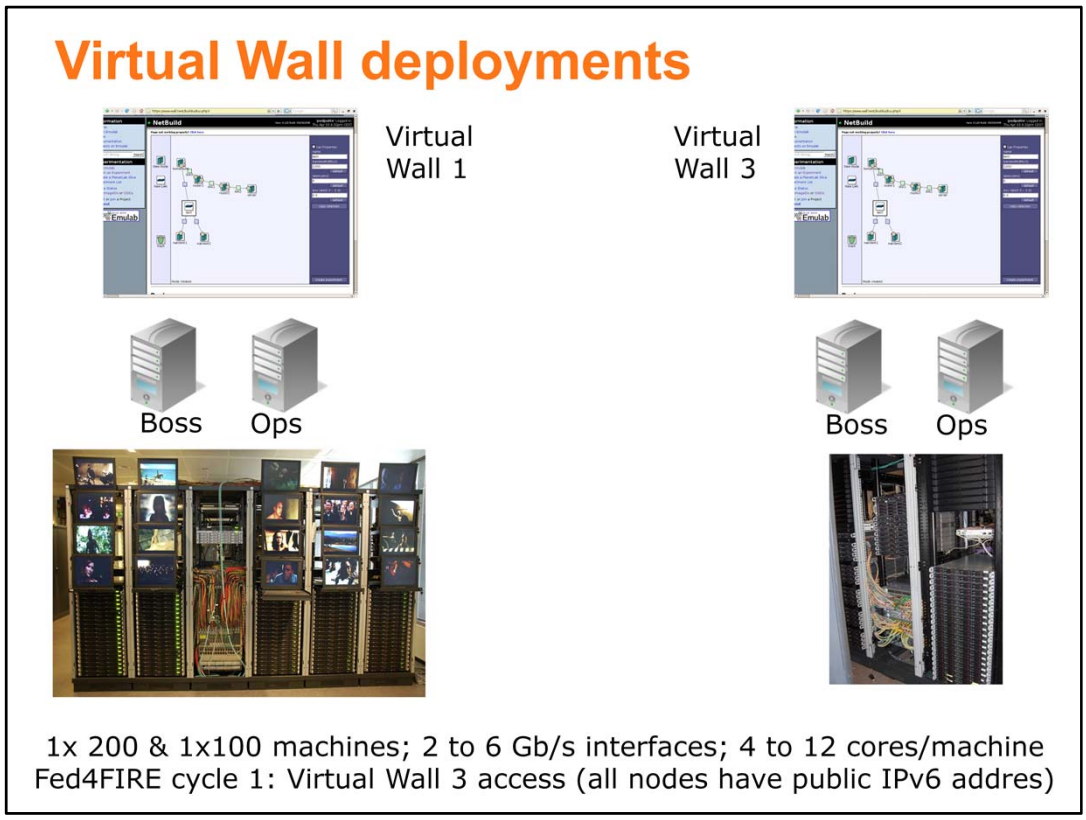
FED4FIRE

iMinds
CONNECT.INNOVATE.CREATE

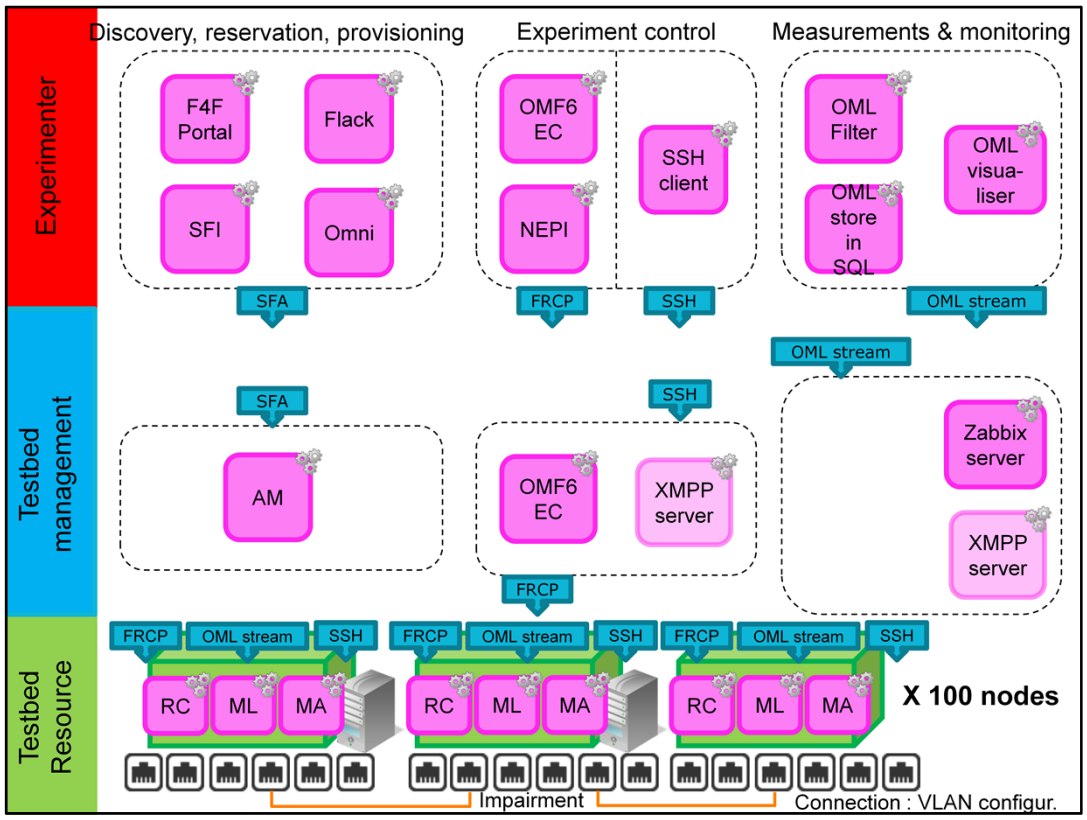


The Virtual Wall **(http://www.iminds.be/en/develop-test/ilab-t/virtual-wall)** is an emulation environment that consists of 100 nodes (dual processor, dual core servers) interconnected via a non-blocking 1.5 Tb/s Ethernet switch, and a display wall (20 monitors) for experiment visualization. Each server is connected with 4 or 6 gigabit Ethernet links to the switch. The experimental setup is configurable through Emulab, allowing to create any network topology between the nodes, through VLANs on the switch. On each of these links, impairments (delay, packet loss, bandwidth limitations) can be configured. The Virtual Wall nodes can be assigned different functionalities ranging from terminal, server, network node to impairment node. The nodes can be connected to test boxes for wireless terminals, generic test equipment, simulation nodes (for combined emulation and simulation) etc. The Virtual Wall features Full Automatic Install for fast context switching (e.g. 1 week experiments), as well as remote access.

Being an Emulab testbed at its core, the possibility to create any desired network topology and add any desired clients and servers makes it possible for the Virtual Wall to support a plethora of Future Internet experiments. These can focus more on networking experiments, e.g. emulating a large multi-hop topology in which advanced distributed load balancing algorithms are tested. However, the Virtuall Wall can just as well support experiments focusing on the application layer, e.g. performance testing of a novel web indexing service on a few server nodes, and creating a large amount of clients that feed it with a very large amount of data during the experiment. An illustration of this high level of flexibility displayed by the Virtual Wall is the tfact that it is also applied in the context of OpenFlow experimentation in its role of a OFELIA island, and in the context of cloud experimentation in its role of a BonFIRE island.



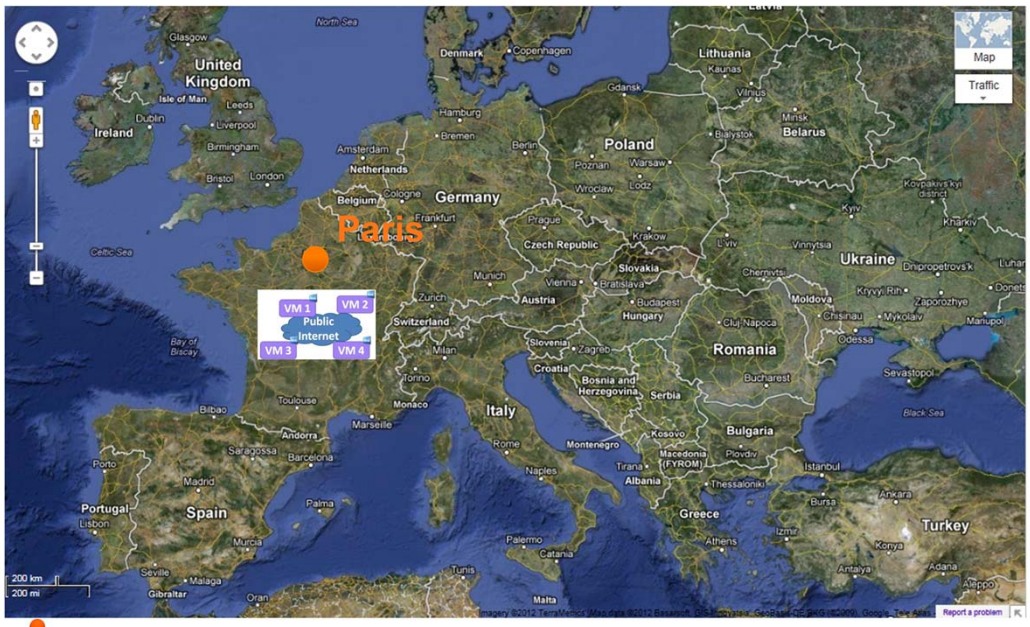
There are currently 2 deployments of the Virtual Wall at iMinds (Virual Wall 1 with 200 nodes and VW 3 with 100 nodes). In Fed4FIRE the initial target is to allow access to the newest instance, the Virtual Wall 3. An important characteristic of this instance is that all the nodes are also publically reachable through the public Internet using the IPv6 protocol.



Compared to the summary figure presented before, the situation at the Virtual Wall testbed is characterized by the following properties:

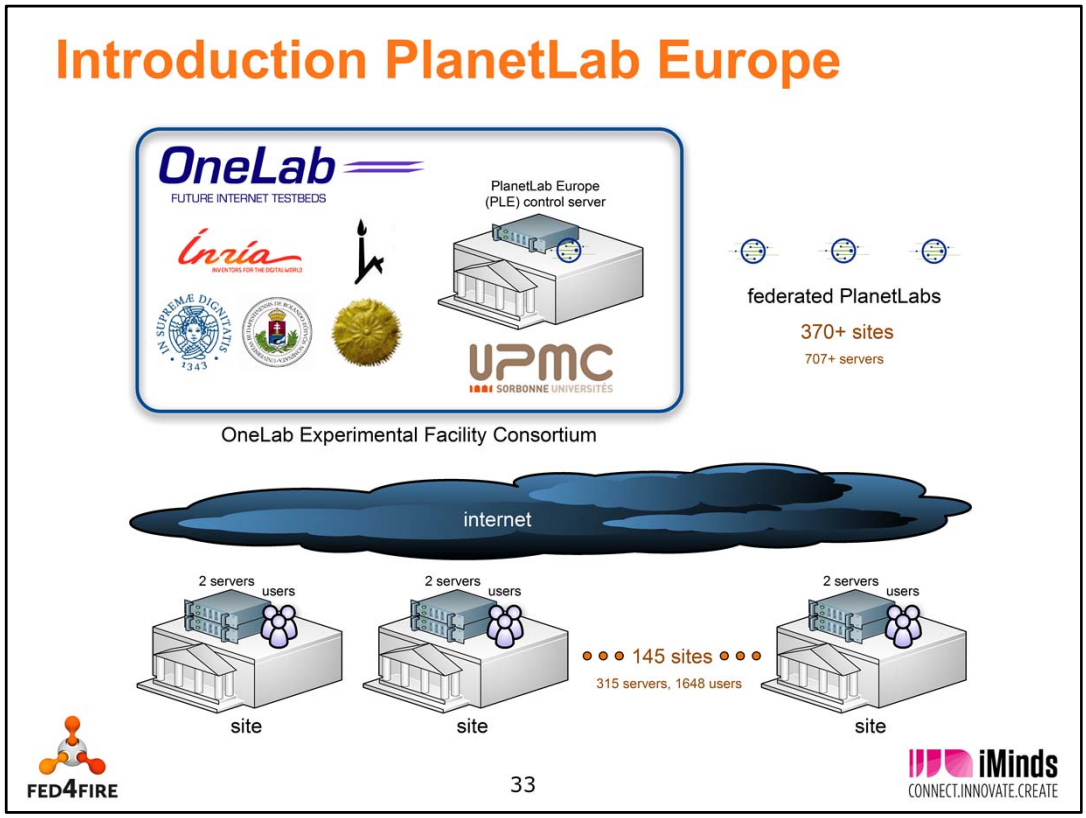
- Only Zabbix is supported as a monitoring framework (not Collectd nor Nagios).
- The depicted resources are Virtual Wall-specific: 100 dual processor, dual core servers, each equipped with multiple gigabit ethernet interfaces. These are all connected to one non-blocking 1.5 Tb/s Ethernet switch. The Virtual Wall system will automatically configure the corresponding VLAN settings in order to build any topology that the experimenter desires. In the example on the slide, the requested topology is that the first and last node should seem directly connected to each other on a LAN with a certain bandwidth and delay configuration. Therefore the system had automatically put an impairment node in between. However, from the perspective of the first and last node, this impairment node is invisible, and they believe that they are directly connected on layer 2.

PlanetLab Europe (UPMC)



FED4FIRE

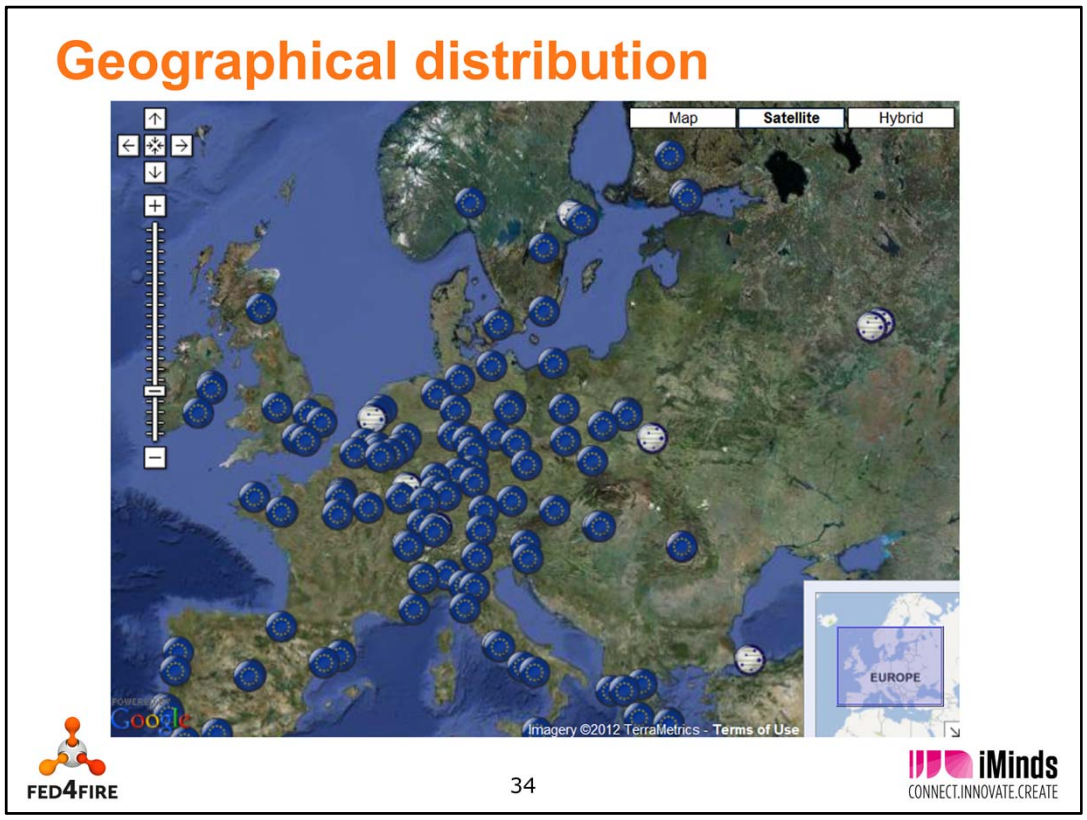
iMinds
CONNECT.INNOVATE.CREATE



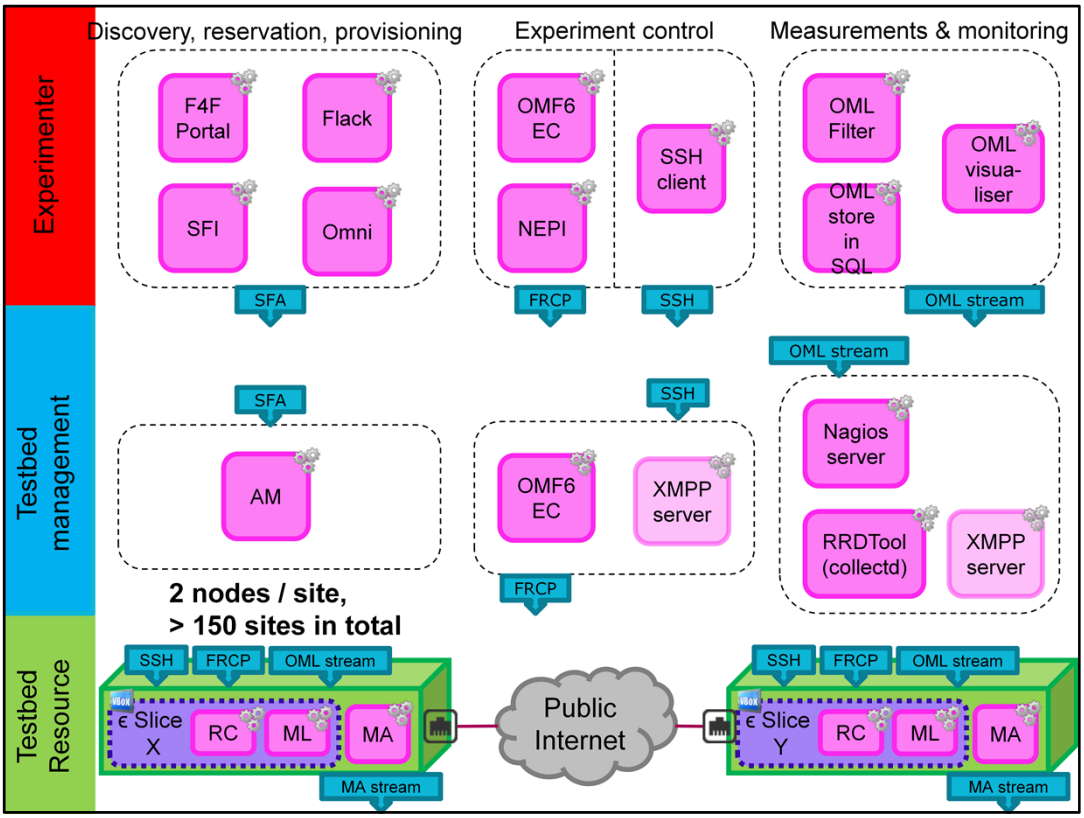
PLE (Planetlab Europe, http://www.planet-lab.eu) is the European arm of the global PlanetLab system, the world's largest research networking facility, which gives experimenters access to Internet-connected Linux virtual machines on over 1000 networked servers located in the United States, Europe, Asia, and elsewhere. Researchers use PLE for experiments on overlays, distributed systems, peer-to-peer systems, content distribution networks, network security, and network measurements, among many other topics.

Organisations can join PLE if they provide two physical servers at their premises, which are directly connected to the internet (no firewalls) and are put under the control of the PLE management software. Once you are a member of PLE, you can request virtual machines on any of the PLE physical servers. These virtual machines will always be directly connected to the public Internet. As an experimenter, you have full control over these virtual machines. The PlanetLab Europe Consortium has over 140 signed member institutions: mostly universities and industrial research laboratories, each of which hosts two servers that it makes available to the global system. These institutions are home to 937 experimenters. On a typical recent day, 244 were connected to on-going experiments.

Through Fed4FIRE, experimenters affiliated to organisations that have not joined PlanetLab Europe will now also be able to perform experiments on PLE.



The nodes belonging to PLE are part of institutions spread over Europe. Furthermore, through specific PlanetLab federations between PLE, PlanetLab Central and the Japanese PlanetLab instance, PLE experimenters can in fact deploy their virtual machines on servers anywhere across the world.

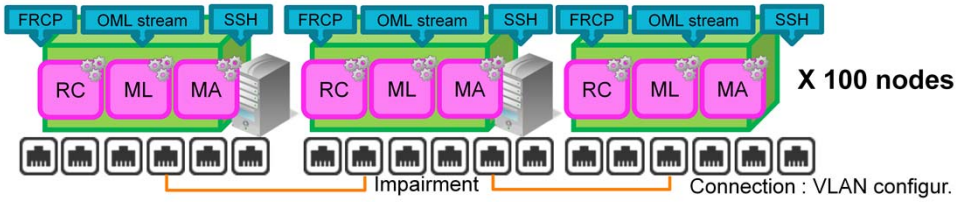


Compared to the summary figure presented before, the situation at the PlanetLab Europe testbed is characterized by the following properties:

- Zabbix is not used. Nagios is used for infrastructure monitoring, collectd is used for facility monitoring
- The depicted resources are PLE specific: 2 physical servers per site on which experimenters can create virtual machines belonging to their slice. These are connected directly to the public internet, without any firewalling or NAT.

Resources overview wired testbeds

Virtual Wall



PlanetLab Europe

The wireless testbeds

- Norbit (NICTA)
- w-iLab.t (iMinds)
- NITOS (UTH)
- Netmode (NTUA)
- SmartSantander (UC)
- FuSeCo (FOKUS)

FED4FIRE

37

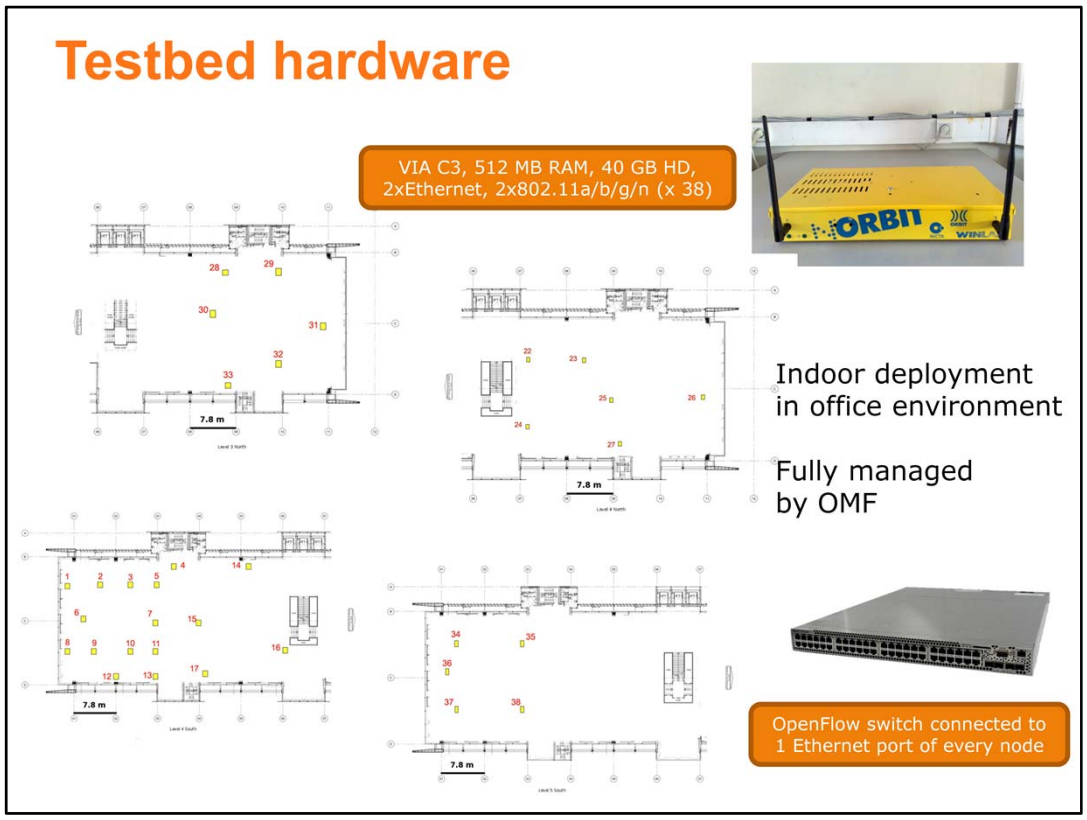
iMinds
CONNECT.INNOVATE.CREATE

Note that SmartSantander (in this case the Santander facility) is of course also well known for being a Smart City services testbed, and not only a wireless testbed.

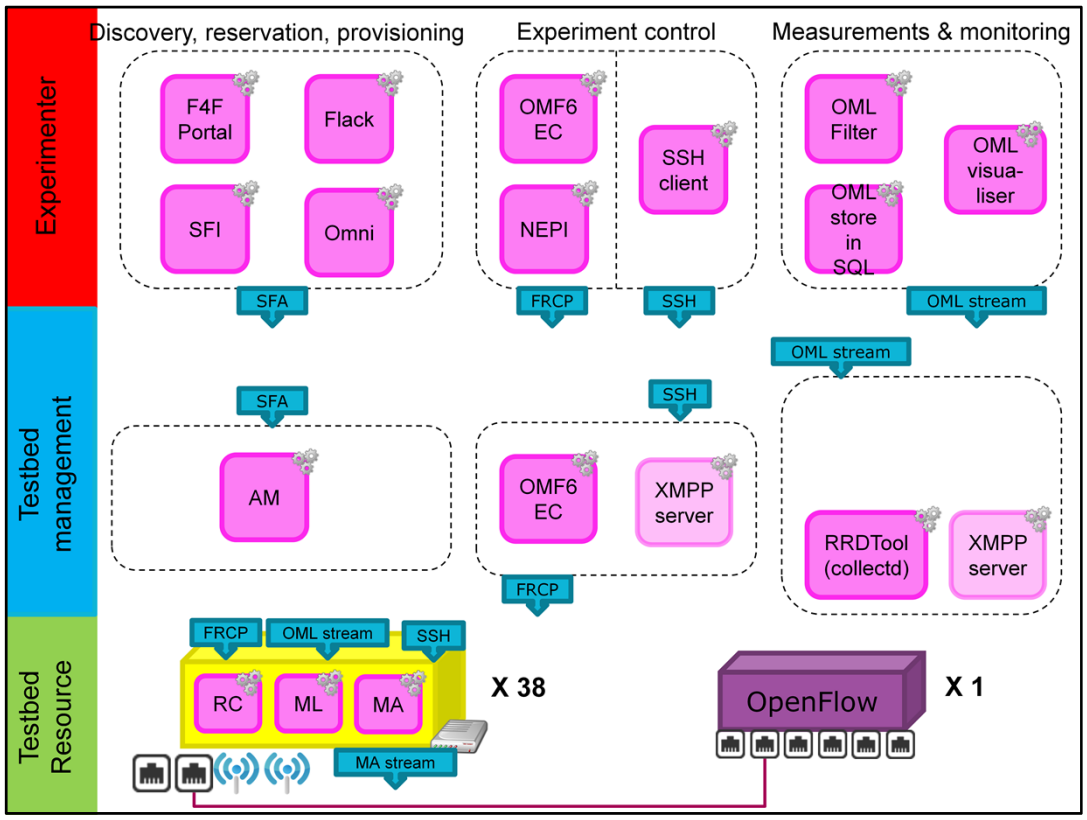
Norbit testbed (NICTA)

FED4FIRE

iMinds
CONNECT.INNOVATE.CREATE



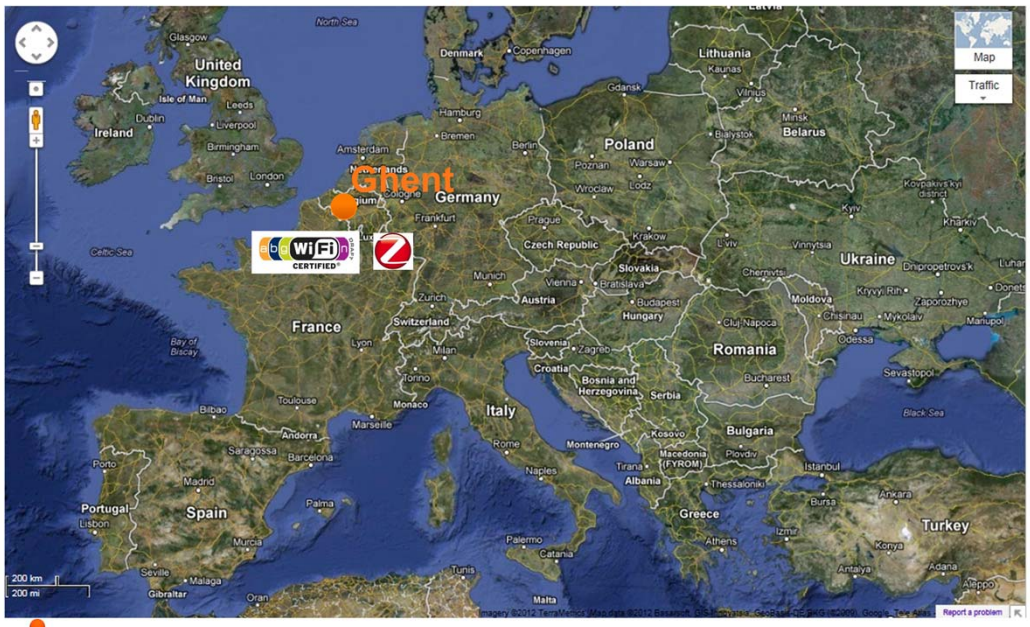
The NORBIT testbed (http://omf.mytestbed.net/projects/omf/wiki/DeploymentSite) is a Wi-Fi testbed located in Sydney, Australia. It belongs to the NICTA group. The testbed consists of 38 nodes equipped with a 1 GHz VIA C3 CPU, 512 MB RAM, 40 GB HD, 2 GigabitEthernet interfaces and 2 IEEE 802.11a/b/g/n interfaces. These nodes are installed indoors in an office environment, and are fully managed by OMF. One of the Gb Ethernet interfaces of all the nodes is attached to a configurable OpenFlow (OF) switch. This provides an experimental 1Gb wired network, in addition to the experimental wireless networks that the nodes can build.



Compared to the summary figure presented before, the situation at the NORBIT testbed is characterized by the following properties:

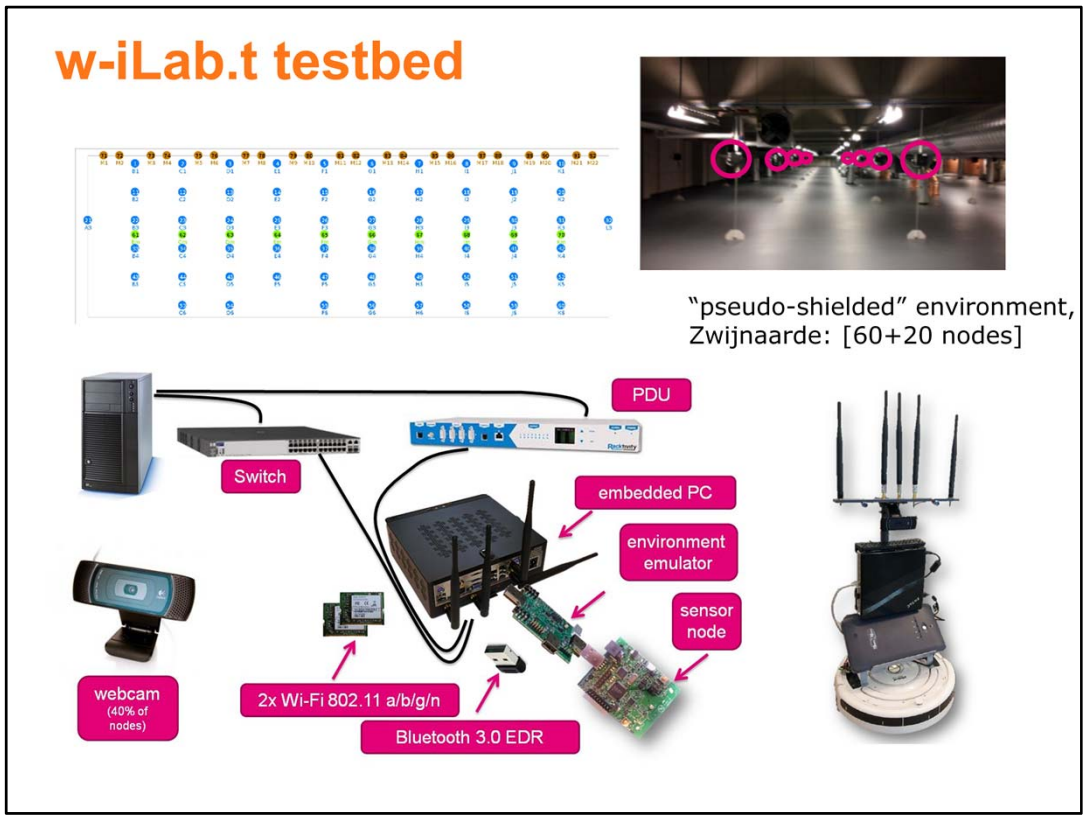
- Only Collectd is supported as a monitoring framework (not Zabbix nor Nagios).
- The depicted resource is NORBIT-specific: 38 nodes equipped with a 1 GHz VIA C3 CPU, 512 MB RAM, 40 GB HD, 2 Gigabit Ethernet interfaces and 2 IEEE 802.11a/b/g/n interfaces. One of the Gb Ethernet interfaces of all the nodes is attached to a configurable OpenFlow (OF) switch

w-iLab.t testbed (iMinds)



FED4FIRE

iMinds
CONNECT.INNOVATE.CREATE



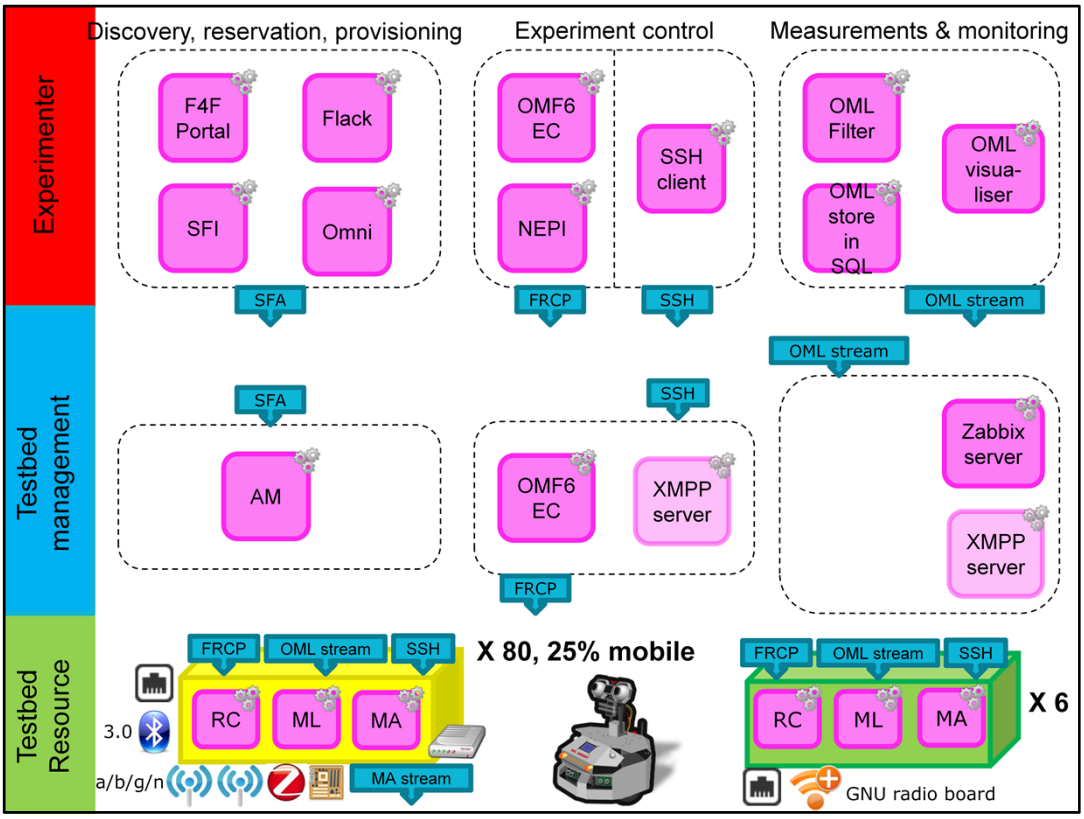
The w-iLab.t testbed is composed of two separate deployments, of which initially only the one called "w-iLab.t Zwijnaarde" will be available through the Fed4FIRE federation for the first round of open call experiments. (http://www.iminds.be/en/develop-test/ilab-t/wireless-lab/w-ilab-t-zwijnaarde) . This testbed is intended for Wi-Fi and sensor networking experimentation. It is located in Zwijnaarde, a district of Ghent, and belongs to iMinds.

It can be found in an unmanned utility room (size: 66m x 22.5m). As opposed to the other w-Ilab deployment (called w-iLab Office) where people work and where there are also operational wireless networks (WLAN, DECT phones) installed, there is a lot less external radio interference in the Zwijnaarde deployment. At this location, hardware is hosted at 60 spots. Every spot is equipped with:

- 1 embedded PC with 2 Wi-Fi a/b/g/n interfaces and 1 IEEE 802.15.1 (Bluetooth) interface
- a custom iMinds-Rmoni sensor node with an IEEE 802.15.4 interface
- an "environment emulator" board, enabling unique features of the testbed including the triggering of repeatable digital or analog I/O events at the sensor nodes, real-time monitoring of the power consumption, and battery capacity emulation.

There are two additional possibilities in the Zwijnaarde deployment:

- A number of cognitive radio platforms (including USRPs) as well as specialized spectrum scanning engines are available at this location. This enables state-of-the art research in the field of cognitive radio and cognitive networking.
- A deployment with 20 mobile robots is planned for 2013. This will allow experiments to be scheduled that include up to 20 mobile nodes (similar to the 60 fixed nodes) to be operated in the environment.



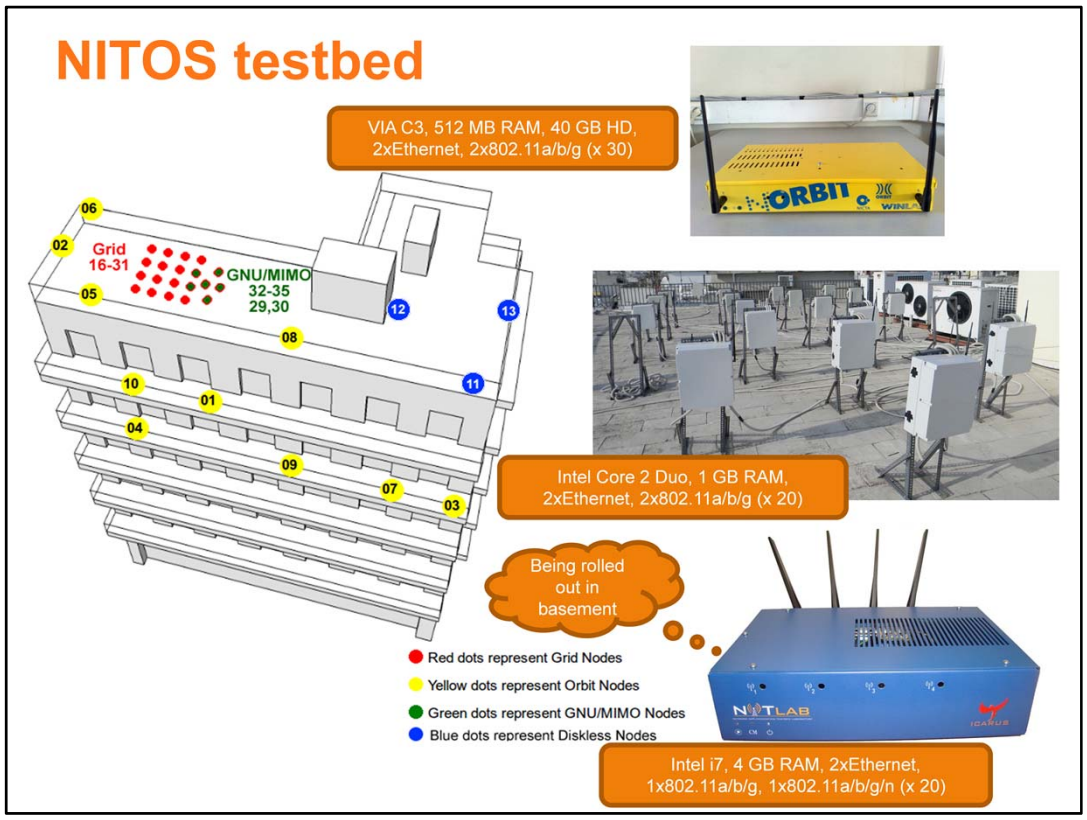
Compared to the summary figure presented before, the situation at the w-iLab.t testbed is characterized by the following properties:

- Only Zabbix is supported as a monitoring framework (not Collectd nor Nagios).
- The depicted resource w-iLab.t specific: 80 Atom based embedded PCs with 2 Wi-Fi a/b/g/n interfaces and 1 IEEE 802.15.1 (Bluetooth) interface, a custom iMinds-Rmoni sensor node with an IEEE 802.15.4 interface and an “environment emulator” board. 20 of those 80 nodes are installed on mobile robots (Roomba). There are also 6 cognitive radio platforms (including USRPs) as well as specialized spectrum scanning engines

NITOS testbed (UTH)

FED4FIRE

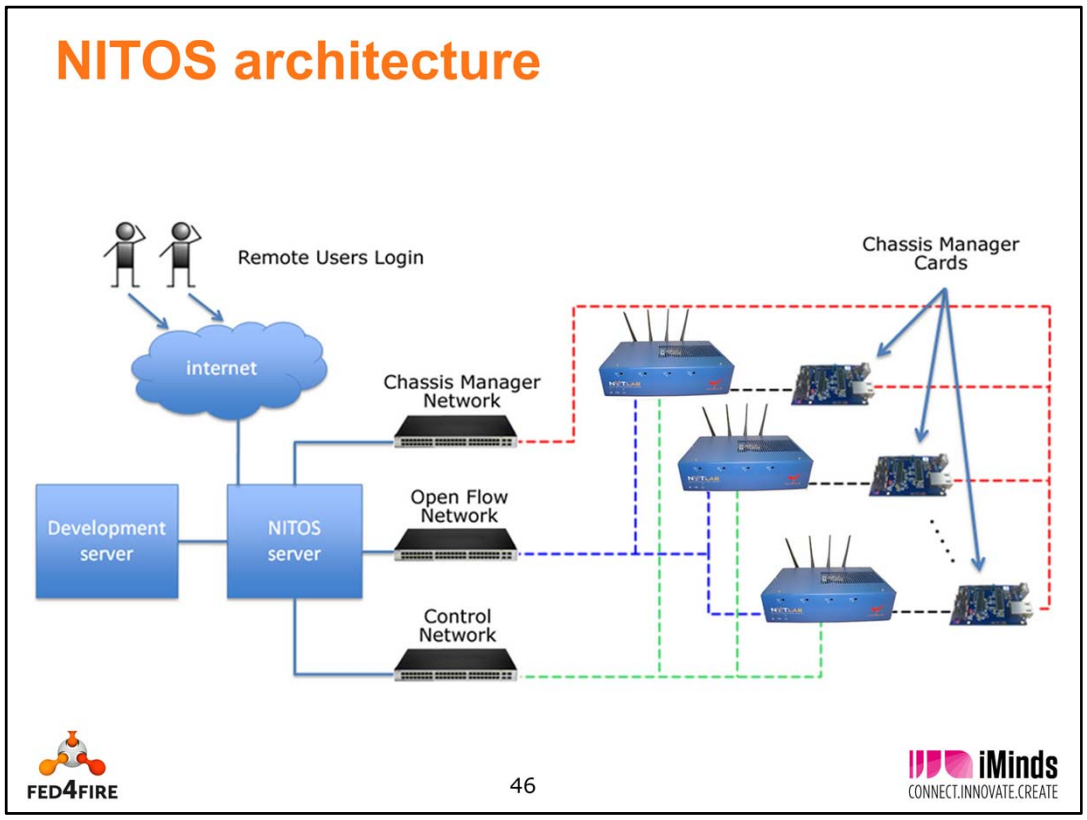
iMinds
CONNECT.INNOVATE.CREATE



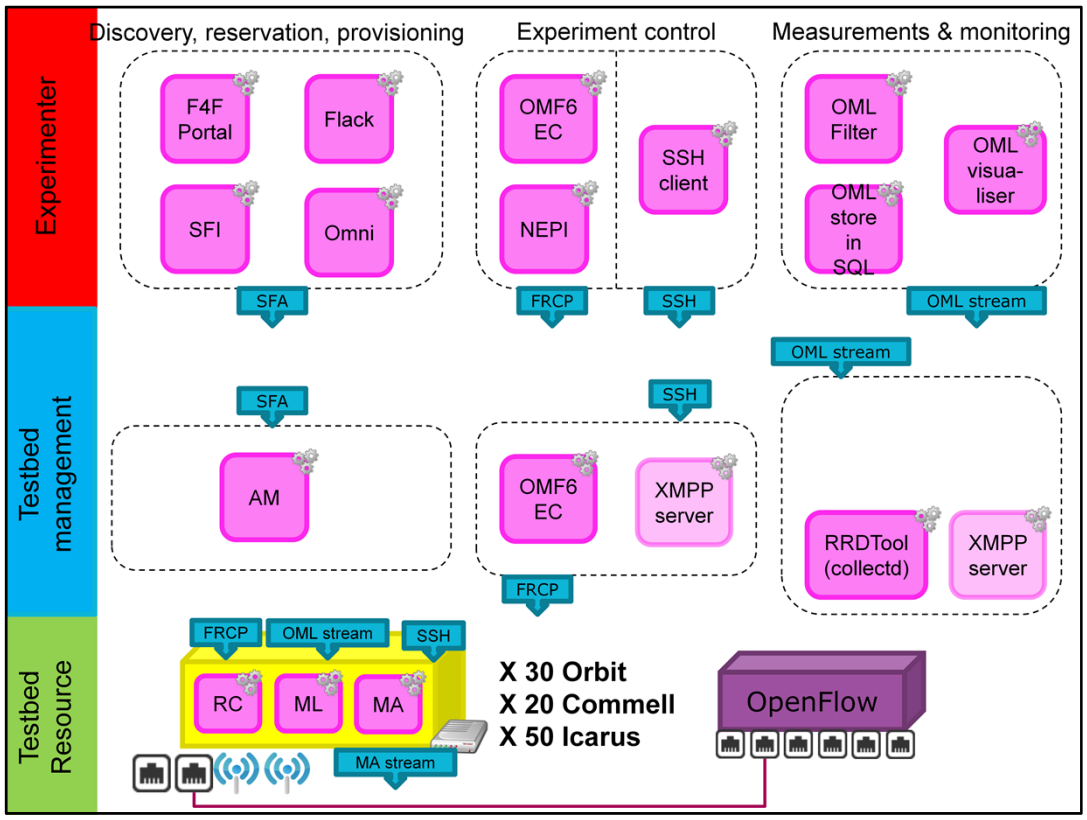
The NITOS testbed (http://nitlab.inf.uth.gr/NITlab/index.php/testbed) is a Wi-Fi testbed of the University of Thessaly, Volos, Greece. It consists of 50 nodes installed in- and outdoors in an office environment, and 50 additional nodes are currently being installed in an indoor shielded environment (basement). Three different hardware types are installed, with different performance characteristics and 802.11 technologies.

- 30 nodes similar to the Norbit nodes: 1 GHz VIA C3 CPU, 512 MB RAM, 40 GB HD, 2 Gigabit Ethernet interfaces and 2 IEEE 802.11a/b/g interfaces
- 20 so-called Commell nodes: Commell LV-67B motherboards, with Intel Core 2 Duo P8400 2,26 GHz processors and 1 GB RAM, 2 Gigabit Ethernet interfaces and 2 IEEE 802.11a/b/g interfaces. 16 of these nodes are deployed in a grid, the 4 others are attached with GNU Radio boards and support MIMO features.
- 50 so-called Icarus nodes: Intel Core i7-2600 Processor, 8M Cache, at 3.40 GHz, 4 GB RAM, 2 Gigabit Ethernet interfaces, 1 IEEE 802.11 a/b/g interface, 1 IEEE 802.11 a/b/g/n interface. These are the nodes that are being installed in the shielded environment.

On each node, one of the two Ethernet interfaces is connected to an OpenFlow switch. This way, the testbed can also be used for OpenFlow experimentation next to the typical Wi-Fi experiments.



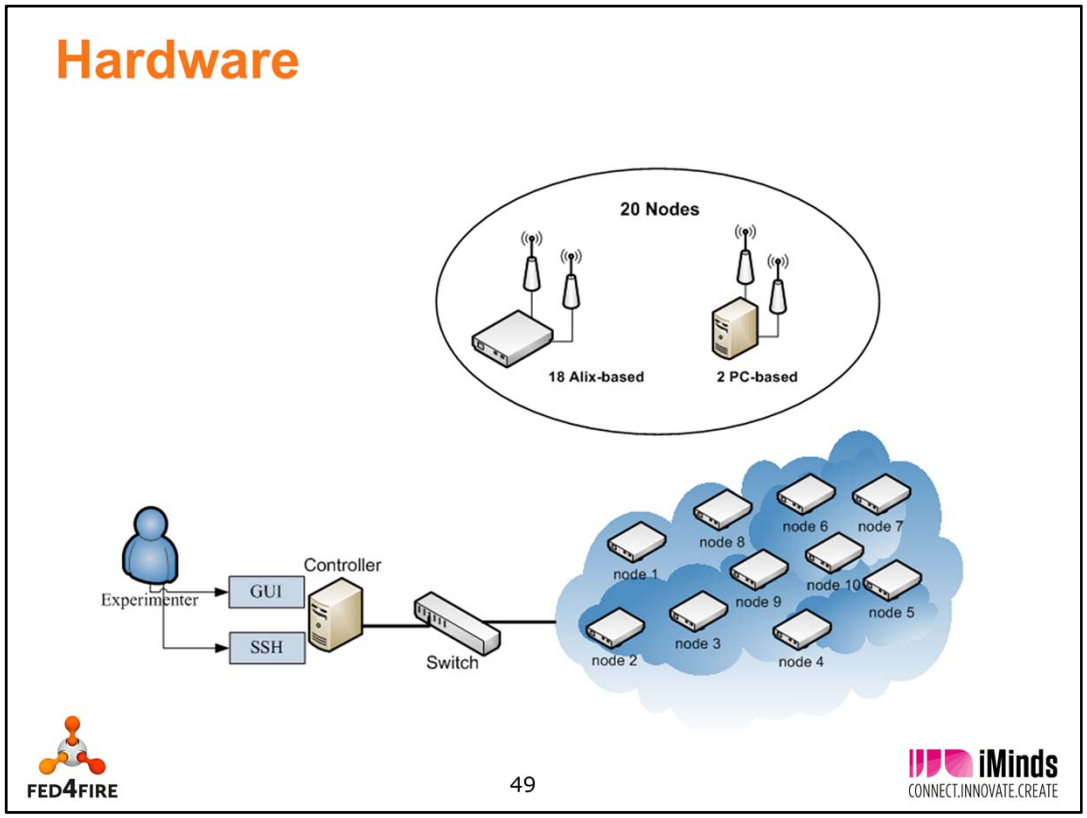
As depicted on the Figure, each NITOS node has two Ethernet ports. One of them is connected to the control netwerk. This allows experimenters to log in on the node using their SSH client, for the EC to communicate with the RC on the node, etc. The second interface is connected to one of several OpenFlow switches, creating an experimental OpenFlow network. This means that although NITOS is originally designed as a Wireless testbed, it can also be used as a wired OpenFlow testbed.



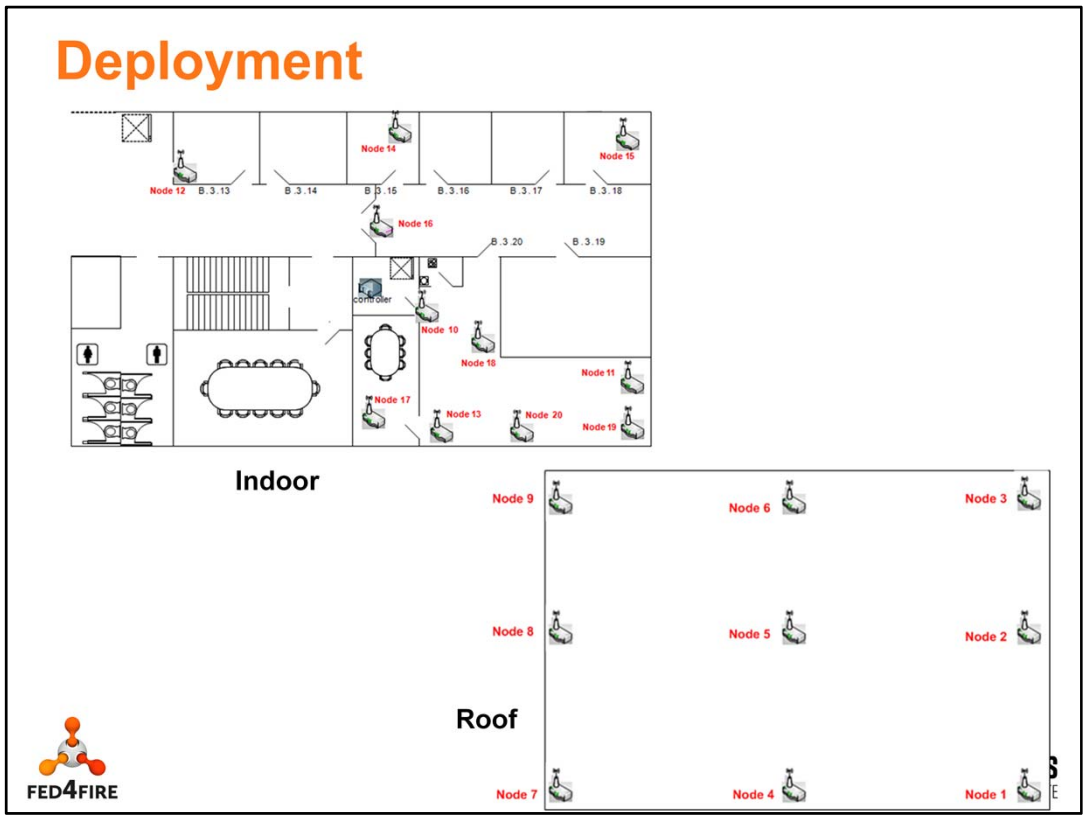
Compared to the summary figure presented before, the situation at the NITOS testbed is characterized by the following properties:

- Only Collectd is supported as a monitoring framework (not Zabbix nor Nagios).
- The depicted resource is NITOS-specific: 30 nodes similar to the Norbit nodes: 1 GHz VIA C3 CPU, 512 MB RAM, 40 GB HD, 2 Gigabit Ethernet interfaces and 2 IEEE 802.11a/b/g interfaces. Also 20 so-called Commell nodes: Commell LV-67B motherboards, with Intel Core 2 Duo P8400 2,26 GHz processors and 1 GB RAM, 2 Gigabit Ethernet interfaces and 2 IEEE 802.11a/b/g interfaces. 16 of these nodes are deployed in a grid, the 4 others are attached with GNU Radio boards and support MIMO features. And 50 so-called Icarus nodes: Intel Core i7-2600 Processor, 8M Cache, at 3.40 GHz, 4 GB RAM, 2 Gigabit Ethernet interfaces, 1 IEEE 802.11 a/b/g interface, 1 IEEE 802.11 a/b/g/n interface. These are the nodes that are being installed in the shielded environment. On each node, one of the two Ethernet interfaces is connected to an OpenFlow switch. This way, the testbed can also be used for OpenFlow experimentation next to the typical Wi-Fi experiments.

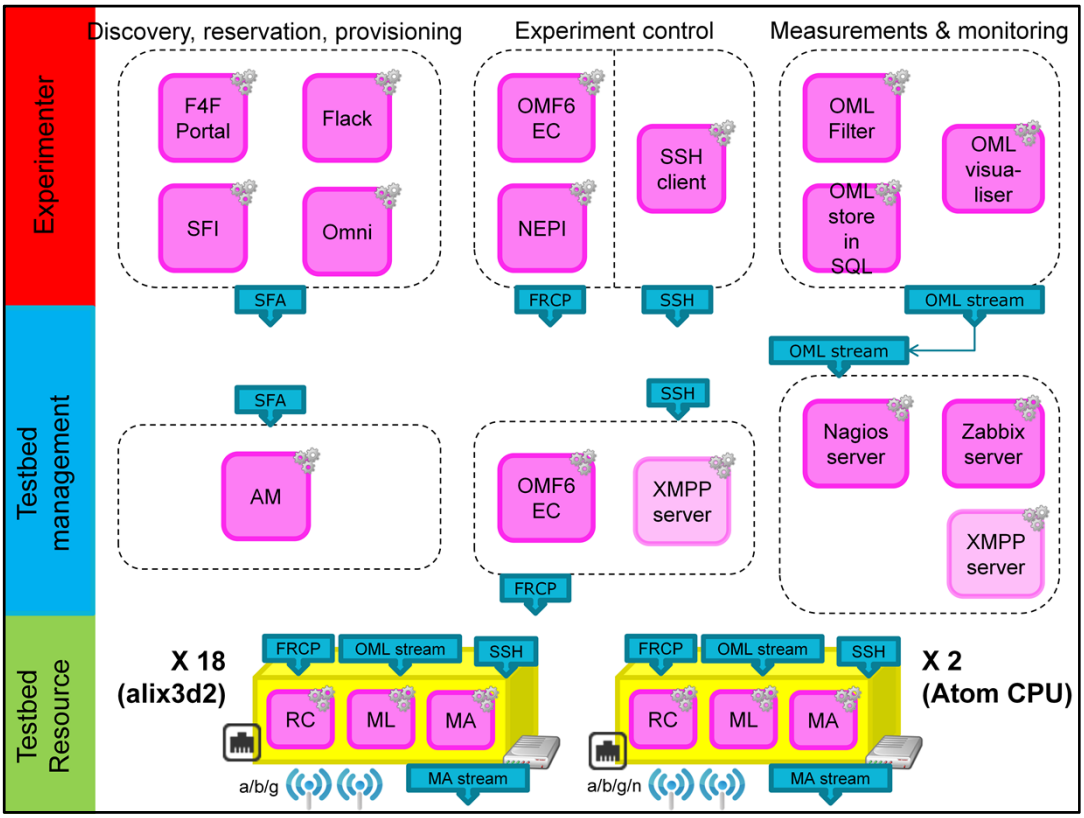
Netmode testbed (NTUA)



The NETMODE testbed (http://www.netmode.ntua.gr/) is a Wi-Fi testbed belonging to the National Technical University of Athens (NTUA). It consists of 20 x86 compatible nodes positioned indoors in an office environment. 18 of these nodes consist of an alix3d2 board with two IEEE 802.11 a/b/g interfaces, one 100Mbit ethernet port, two USB interfaces and a 1GB flash card storage device. The other 2 nodes are more powerful, applying an Intel Atom CPU, a 250 GB hard drive, and providing two 802.11 a/b/g/n interfaces and one Gigabit ethernet interface,.



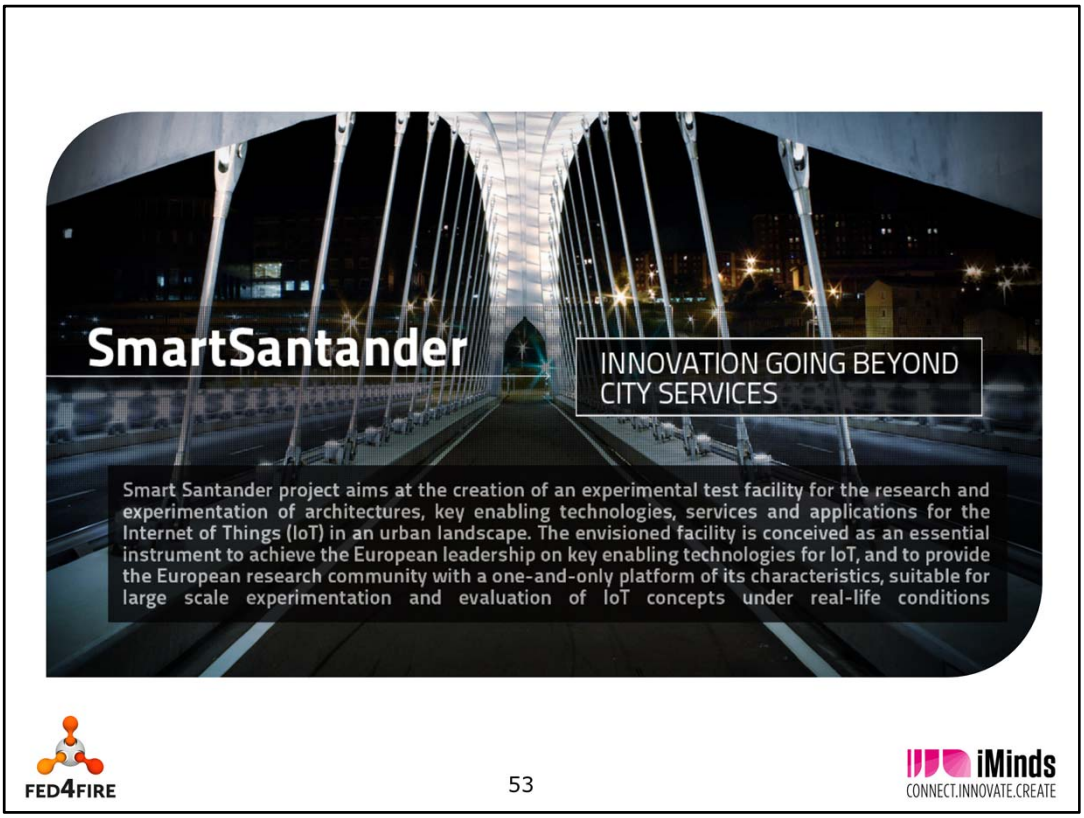
The nodes are scattered around the 3rd floor and the roof of the ECE building at the National Technical University of Athens.



Compared to the summary figure presented before, the situation at the NETMODE testbed is characterized by the following properties:

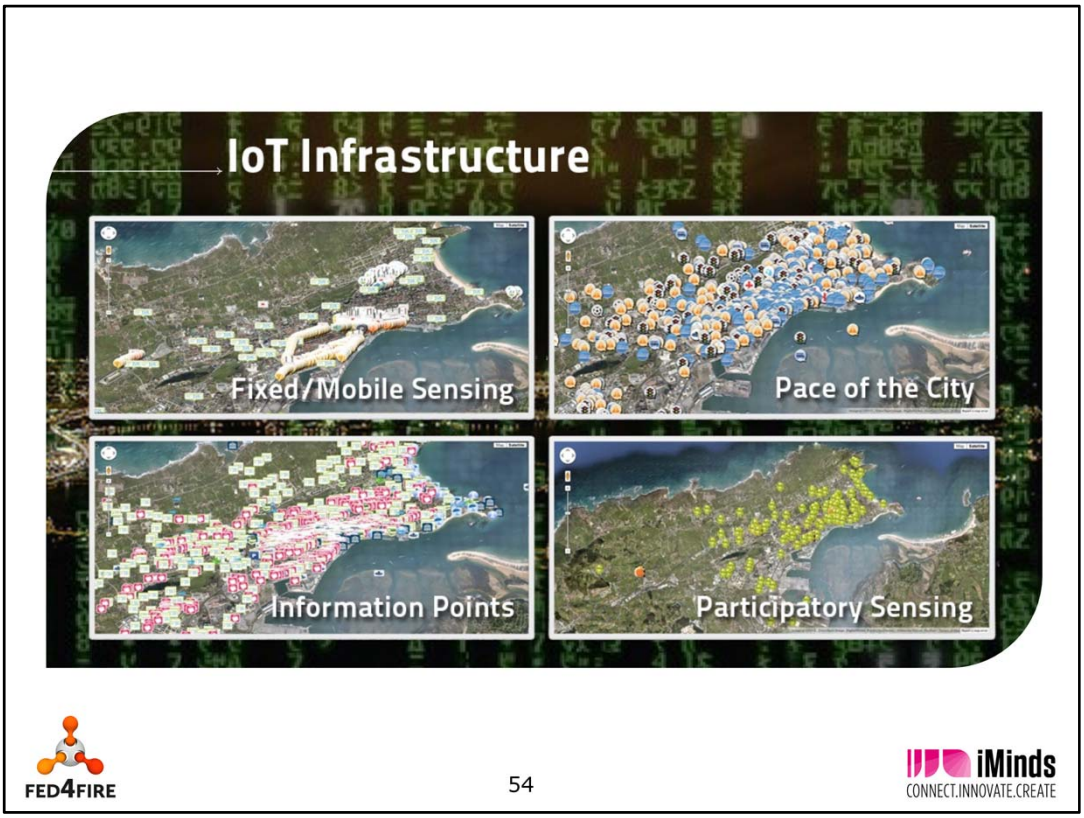
- Collectd is not supported
- The depicted resources are NETMODE-specific: 18 of these nodes consist of an alix3d2 board with two IEEE 802.11 a/b/g interfaces, one 100Mbit ethernet port, two USB interfaces and a 1GB flash card storage device. The other 2 nodes are more powerful, applying an Intel Atom CPU, a 250 GB hard drive, and providing two 802.11 a/b/g/n interfaces and one Gigabit ethernet interface,.

SmartSantander testbed (UC)



The SmartSantander facility is a large scale smart city deployment in the Spanish city of Santander. More detailed information can be found at (http://www.smartsantander.eu/index.php/testbeds/item/132-santander-summary)

The testbed supports two types of experiments: Internet of Things native experimentation (wireless sensor network experiments) and service provision experiments (applications using real-time real-world generated sensor data). At the moment, it is only planned to include service provision experiments in Fed4FIRE



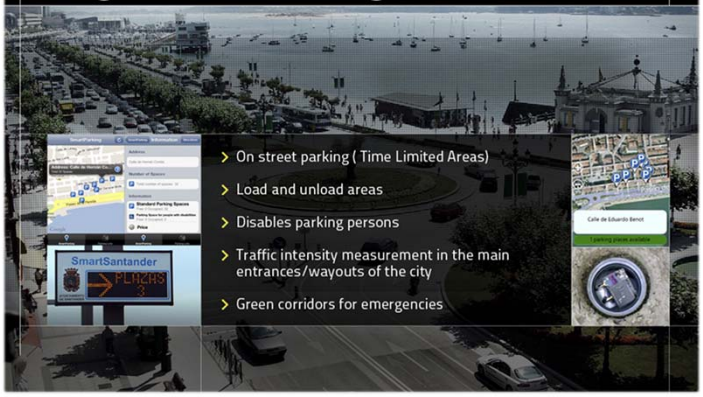
The Santander testbed is composed currently of around 2000 IEEE 802.15.4 devices deployed in a 3-tiered architecture:

- IoT node: Responsible for sensing the corresponding parameter (temperature, CO, noise, light, car presence,etc.).
- Repeaters: These nodes placed high above ground in street lights, semaphores, information panels, etc, in order to behave as forwarding nodes to transmit all the information associated to the different measured parameters. They include two 802.15.4 transceivers in order to separate the native experimentation layer from the management one.
- Gateway node (Meshlium): Device that collects the measurements from a large number of sensor nodes and upload all this data to the SmartSantander servers through their Internet uplink.

However, the IoT infrastructure deployed around the city of Santander does not only include traditional “mota” sensors, but a wider set of sensing points. In this sense, the sensor conception in SmartSantander also covers human interactions with augmented reality passive tags deployed around the city or participatory sensing.

Services from IoT infrastructure

Integral Traffic Management



FED4FIRE

iMinds
CONNECT.INNOVATE.CREATE

Services from IoT infrastructure

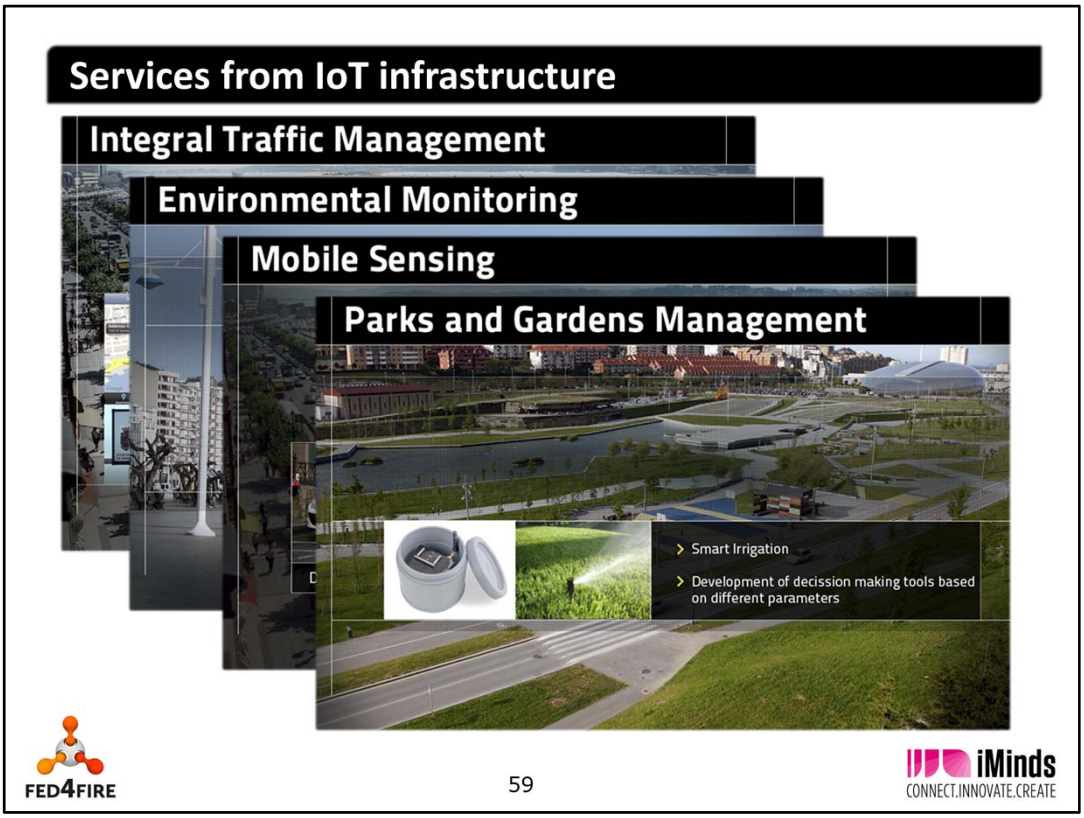
FED4FIRE

iMinds
CONNECT.INNOVATE.CREATE

Services from IoT infrastructure

FED4FIRE

iMinds
CONNECT.INNOVATE.CREATE


Services from IoT infrastructure
Integral Traffic Management
Environmental Monitoring
Mobile Sensing
Parks and Gardens Management
Smart Irrigation
Development of decission making tools based on different parameters
FED4FIRE
59
iMinds
CONNECT.INNOVATE.CREATE

Citizens Generated Services

Participatory Sensing

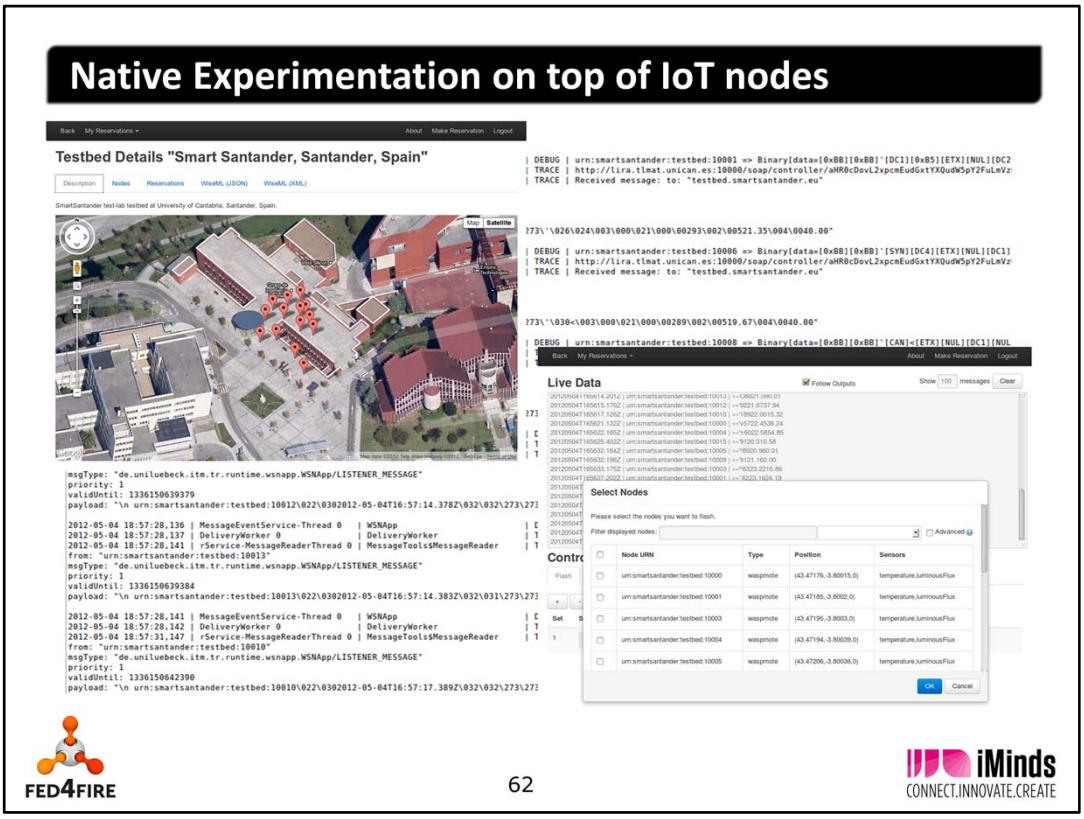
FED4FIRE

iMinds
CONNECT.INNOVATE.CREATE

Citizens Generated Services

FED4FIRE

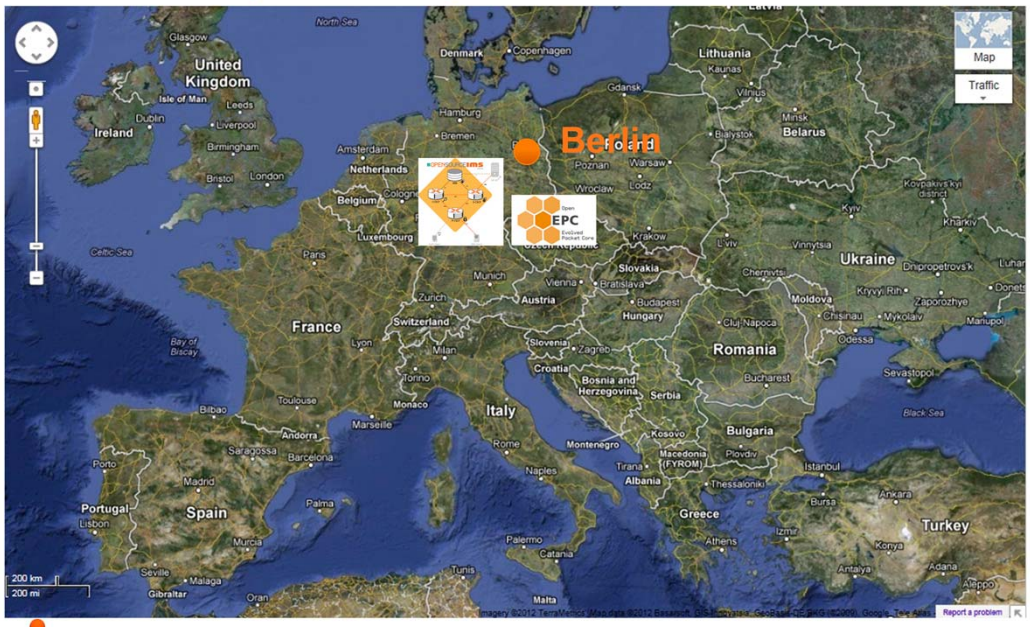
iMinds
CONNECT.INNOVATE.CREATE



As mentioned in the previous slides, during the first round of Fed4FIRE open call experiments, the data originating from all these sources will be provided to the experimenters, but it will not be possible to run wireless IoT experiments.

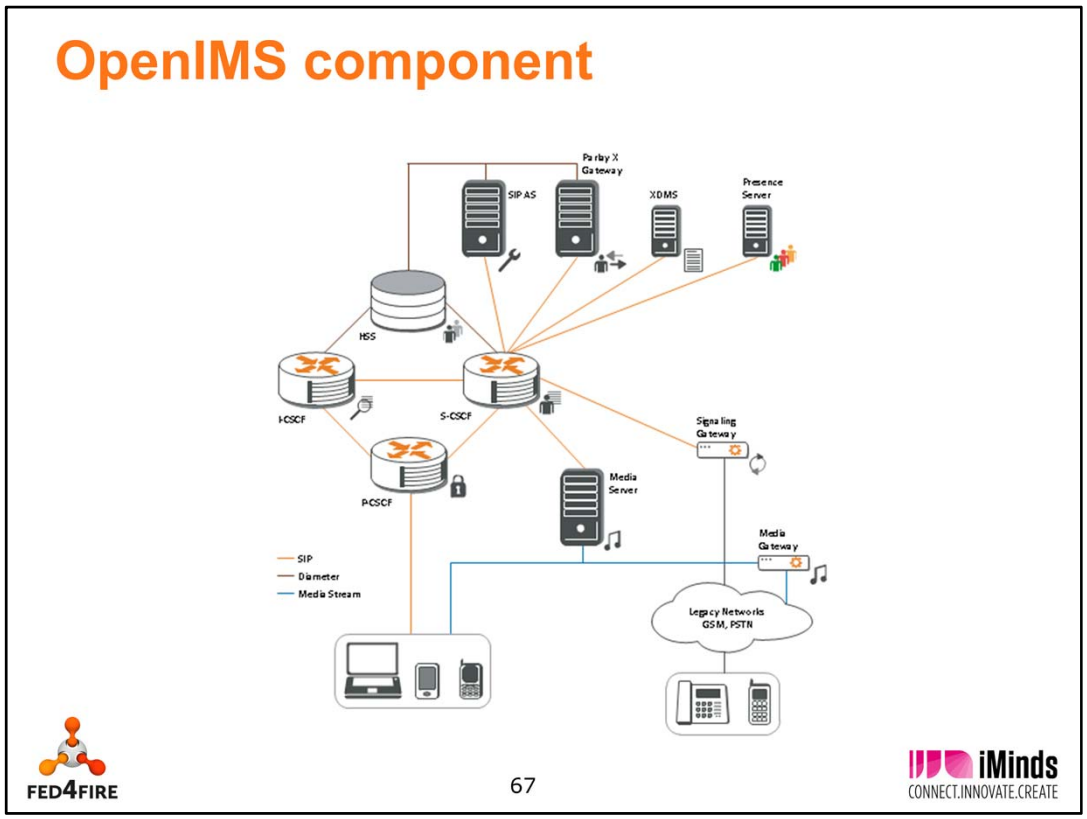
Nodes allowing native experimentation are mainly the ones covering environmental monitoring. Both fixed and mobile sensors deployed in vehicles can be flashed over the air on a multihop fashion (MOTAP) with different programs, thus allowing the researchers to test different experiments on the deployed network. In this case, there is an exclusive resource reservation system prior of getting access to any node.

FuSeCo (FOKUS)

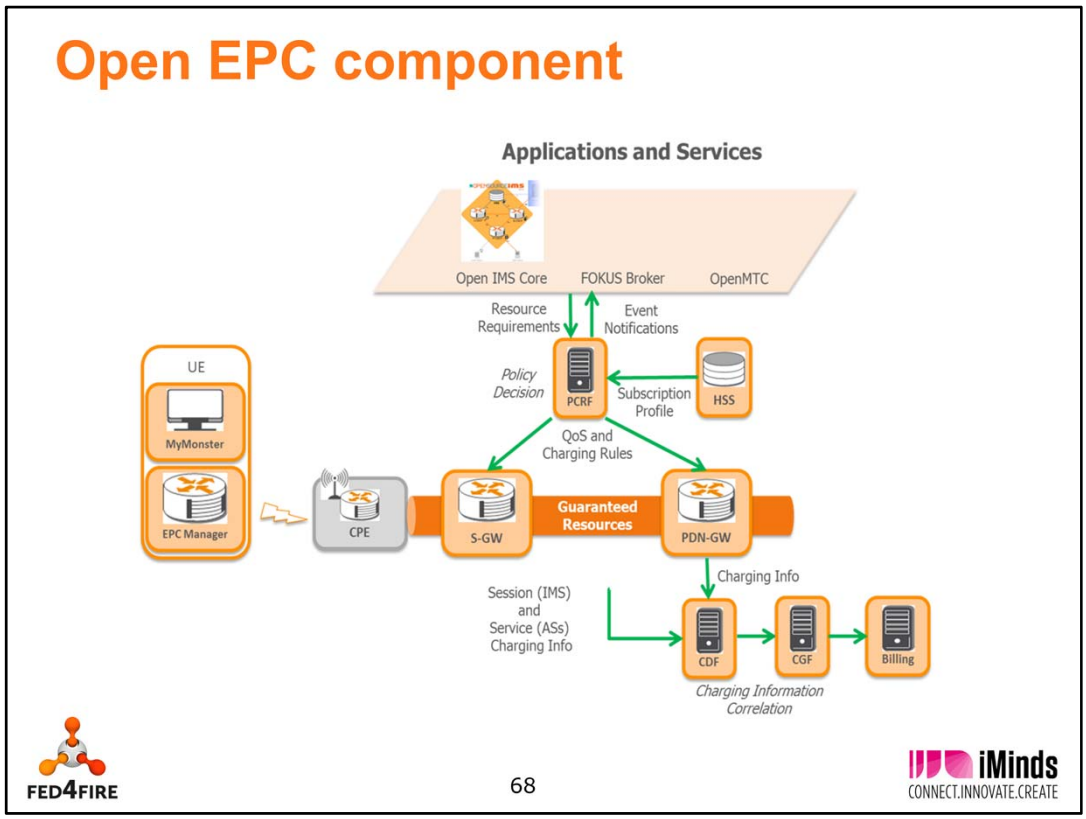


FED4FIRE

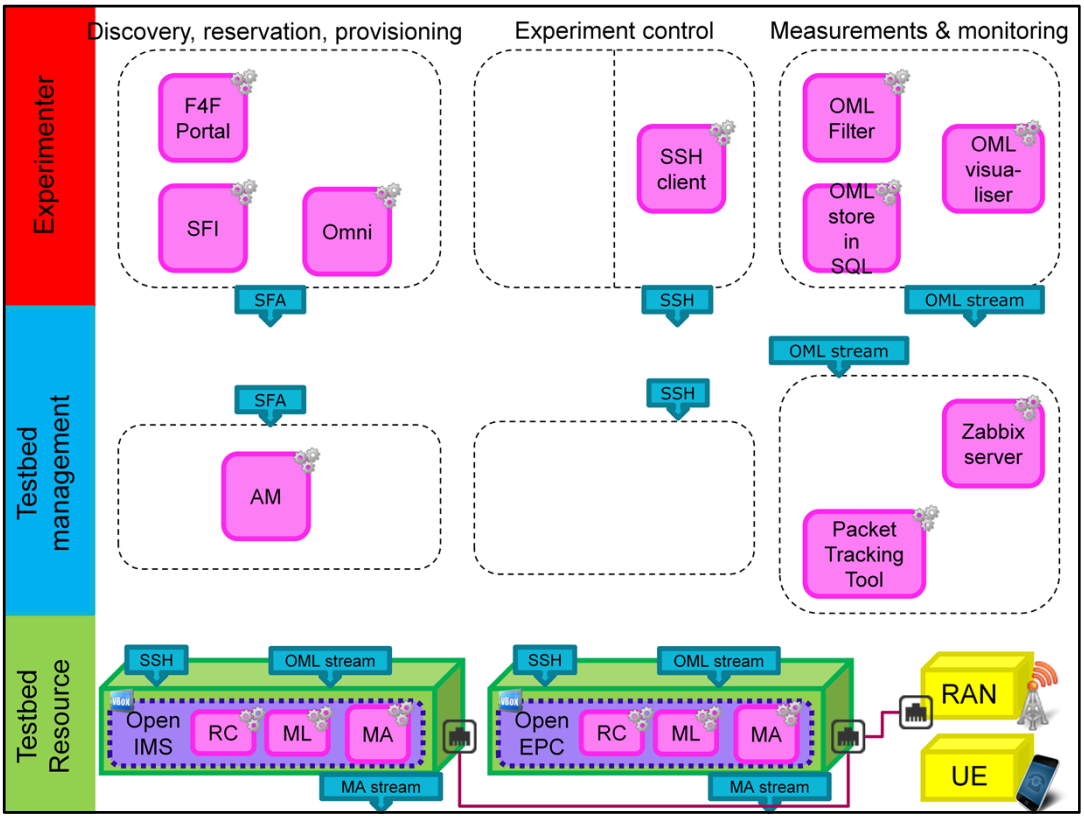
iMinds
CONNECT.INNOVATE.CREATE



The Open IMS Playground (http://www.openimscore.org/) includes an Open Source Project 'OpenIMSCore' which focuses on the development of the IMS core network (HSS and CSCFs). The modular structure of IMS allows full or particular deployment of the OpenIMSCore components, which allows interoperability test of new components against the OpenIMSCore.



FuSeCo Playground integrates various state of the art wireless broadband networks into a 3GPP Evolved Packet Core (EPC) prototype platform, allowing the rapid validation of new networking paradigms, and prototyping of innovative Future Internet and smart city applications. The current focus of standardization within the Third Generation Partnership Project (3GPP) is on building the foundation for future Telco networks to support coming wireless broadband access network types like Long Term Evolution (LTE) or WiMAX and concepts such as ABC (Always Best Connected), inter-technology handovers or (mobile) cloud computing. With the initial launch of Next Generation Mobile Networks of the NGMN Alliance based on the LTE/SAE (System Architecture Evolution) standard which was announced by many operators for 2010, the mobile industry demonstrates its efforts to move a true mobile broadband user experience from promise to reality. With this introduction, technology challenges have to be overcome and the focus of the industry is shifting to the key milestones such as testing and prototyping for deployment and operations, and to the availability of new services and devices.

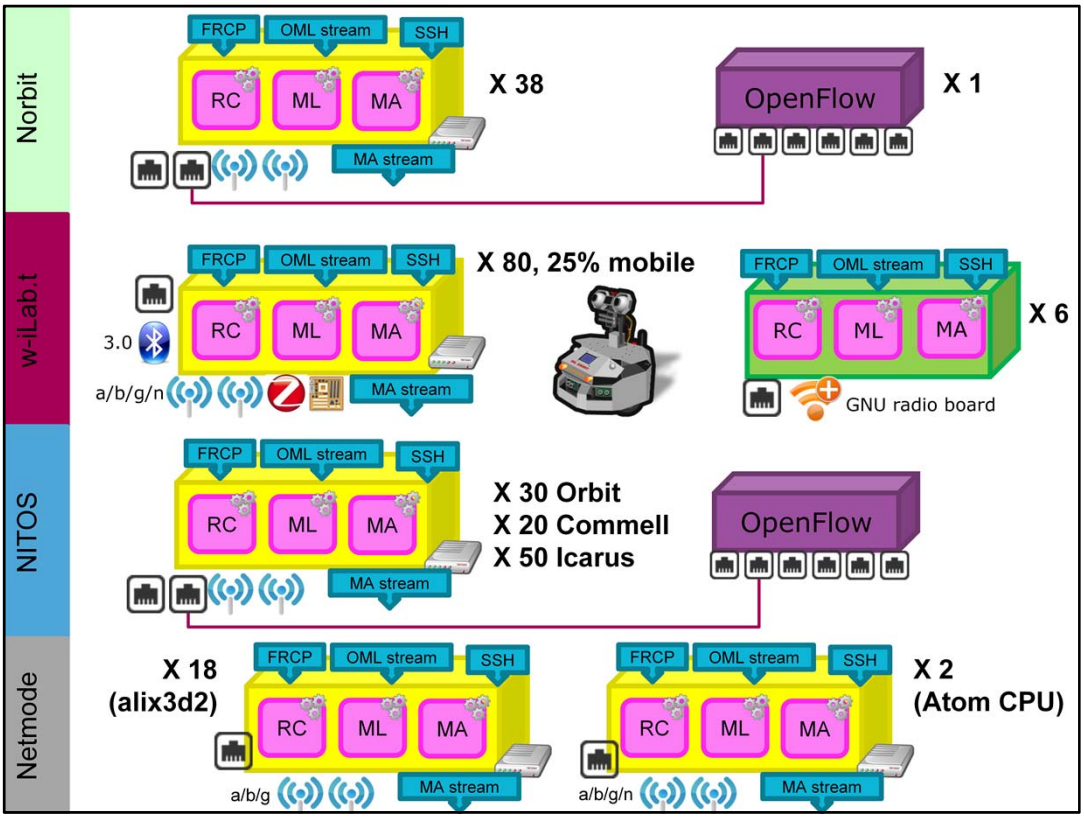


Compared to the summary figure presented before, the situation at the FuSeCo testbed is characterized by the following properties:

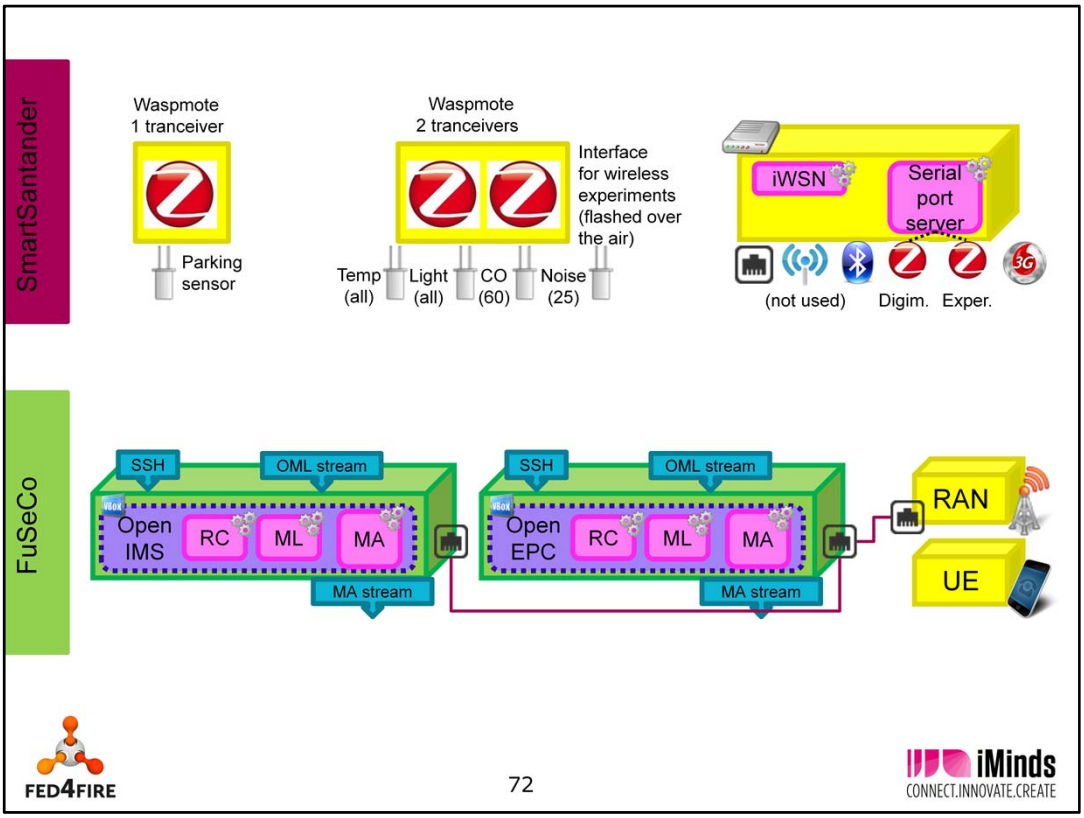
- Flack is not supported
- FRCP is not supported. Therefore OMF6 and NEPI are also not supported.
- Collectd are Nagios not supported, however a new measurement tool Packet Tracking Tool is introduced

The depicted resources are FuSeCo-specific: a fixed number of instances of both OpenIMS and OpenEPC will be provided as a federated service. Experimenters will be able to get exclusive access to such an instance for a given amount of time. During this time they can chance configuration parameters, add additional services, etc. These changes will be reverted when another experimenter gains access to the same instance later on. Every instance will also be characterized by the specific Radio Access Network infrastructure that is dedicated to it. This will determine which instance an experimenter needs to apply for his/her experiment. Also, some User Equipment is available to be included in the experiment. However, experiments that rely on an interaction between human users and UEs connected to the testbed require some sort of physical presence of the experimenter. In practice, such experiments must take place in the vicinity of the wireless infrastructure, which is usually at FOKUS premises in Berlin.

Resource overview wireless testbeds



Resources overview wireless testbeds (1)



As mentioned before, SmartSantander at the moment focuses on making the recorded sensor data available for experimentation. It does not support IoT experimentation using Fed4FIRE (for now).

The OpenFlow testbeds

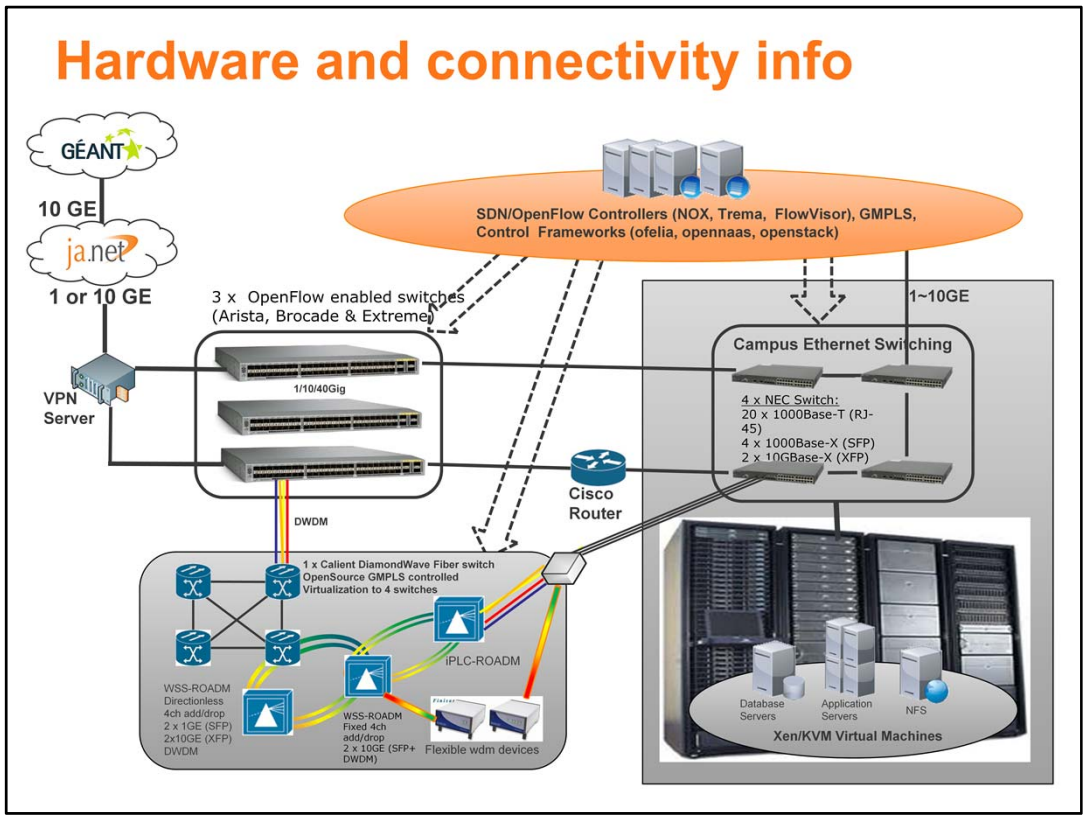
- UBristol OFELIA island
- i2CAT OFELIA island
- Koren testbed (NIA)
- NITOS & Norbit testbeds (see wireless testbeds)

FED4FIRE

iMinds
CONNECT.INNOVATE.CREATE

In this category of testbeds, expseriments are all about OpenFlow, on top of optical hardware or packet switches. In a typical experiment some virtual machines are deployed both as data sources and as data consumers, an OpenFlow path is established between them, and an OpenFlow controller is deployed that can dynamically change the path based on certain conditions. In Fed4FIRE, three different testbeds are focusing on the OpenFlow technology: the Ubristol and the i2CAT islands of the FP7 OFELIA project, and the Korean testbed Koren.

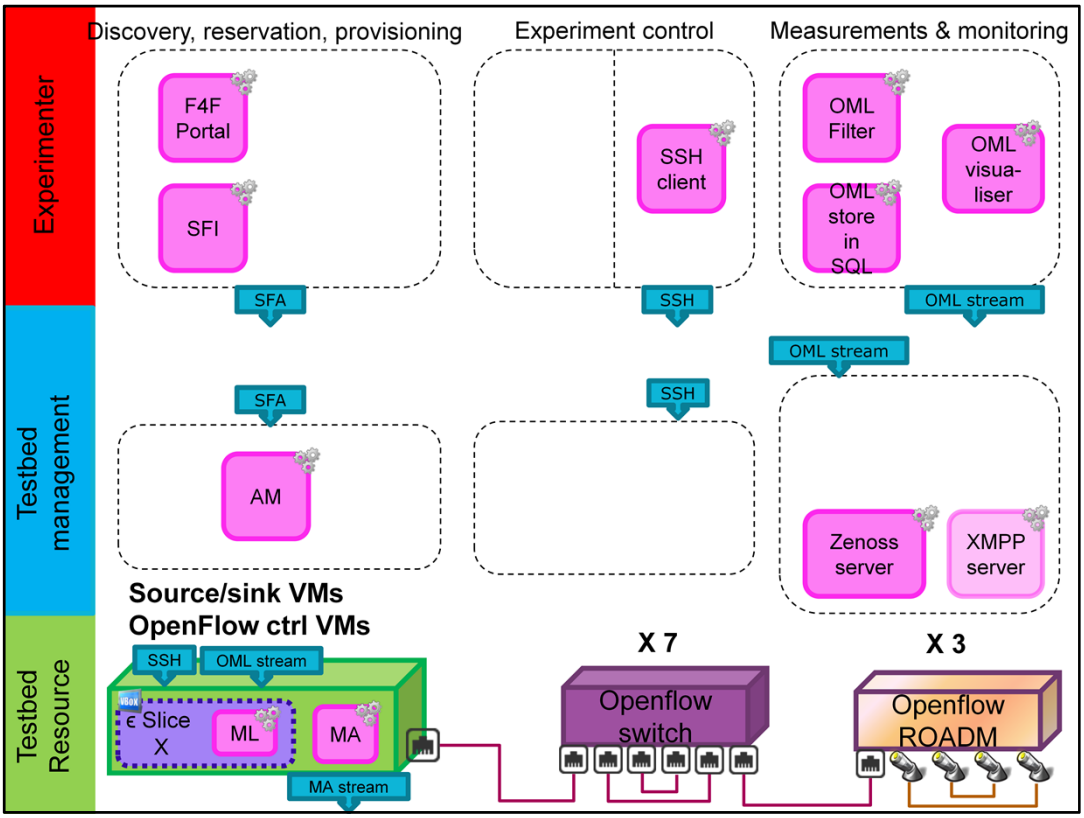
UBristol OFELIA island



In this figure, the infrastructure of University of Bristol (https://alpha.fp7-ofelia.eu/doc/index.php/Testbed_Bristol) is shown from hardware and connectivity perspective.

The added building blocks and component are:

- 3 x OpenFlow-enabled switches: 1 Arista, 1 Brocade, and 1 Extreme
- 4 x Campus Grade Ethernet switches (also OpenFlow-enabled) (NEC IP8800)
- Extra OpenFlow controllers
- 3 x ADVA FSP 3000 DWDM ROADM nodes
- UnivBris-i2CAT connectivity via GEANT (for connectivity test via GEANT)
- Xen/KVM Virtualization Servers



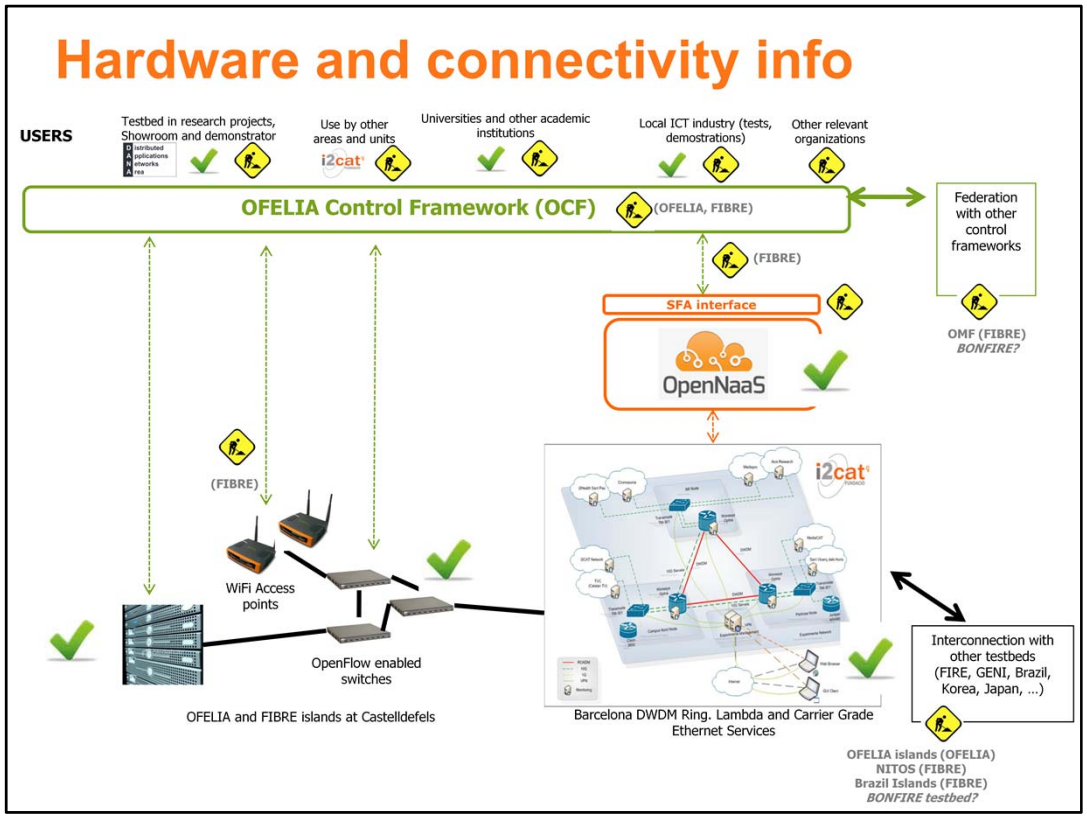
Compared to the summary figure presented before, the situation at the Ubristol Ofelia island is characterized by the following properties:

- Flack and Omni are not supported
- OMF6 and hence FRCP are not supported. Nepi is therefore also not supported.
- Zabix, CollectD and Nagios are not supported. Instead, Zenoss will be wrapped in an OML stream
- The depicted resources are specific: 3 x OpenFlow-enabled switches, 4 x Campus Grade Ethernet switches (also OpenFlow-enabled), 3 x ADVA FSP 3000 DWDM ROADM nodes & Xen/KVM Virtualization Servers which hosts VMs that can act as data source, data sink or openflow controller in an experiment.

i2CAT OFELIA island

FED4FIRE

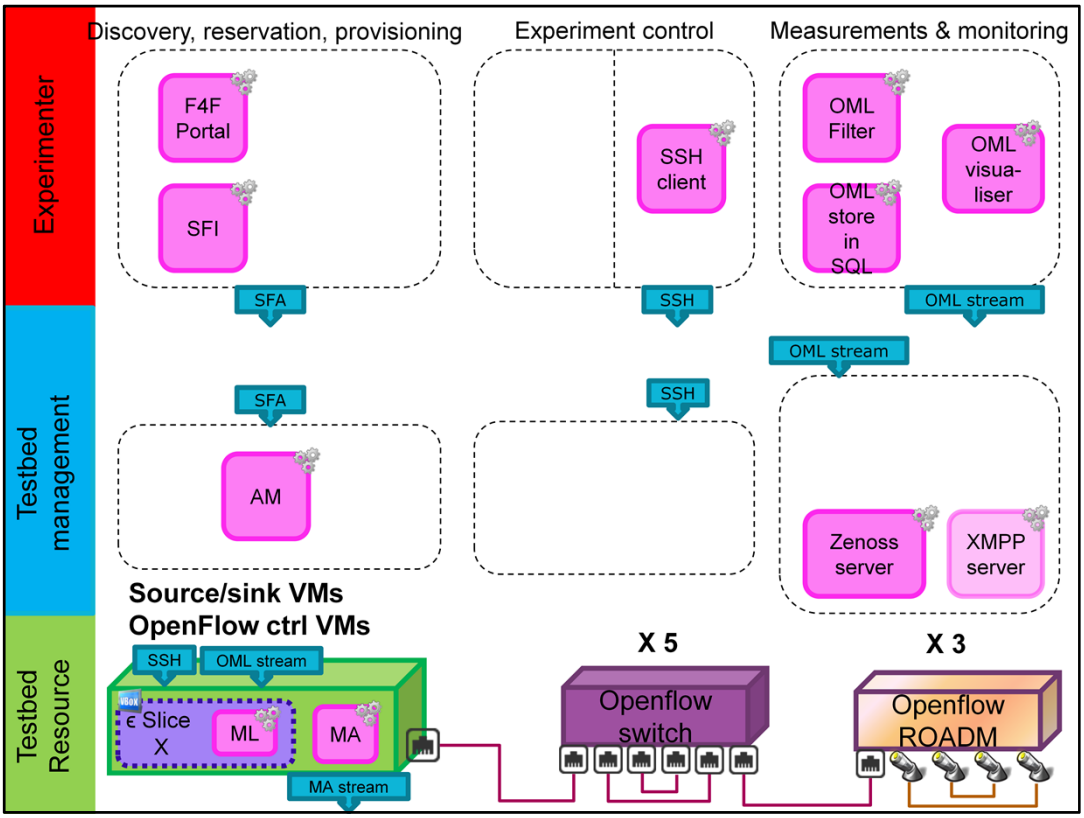
iMinds
CONNECT.INNOVATE.CREATE



The i2CAT OFELIA island provides an open facility to test and validate experimental research aligned with Future Internet technologies. The infrastructure is virtualized in order to offer logical isolated substrates to enable simultaneous disruptive research experiments in productive environments without interfering to parallel research users; following an IaaS (Infrastructure as a Service) model. The main research centres, technical universities and industrial clusters in Catalonia are connected to this facility (called Experimenta) to use, test and offer their experimental services.

The Experimenta open facility is composed by three optical reconfigurable 2.5/10G ROADM-based nodes, establishing a multi-wavelength fibre ring throughout the city of Barcelona; a pool of distributed virtual machines and the dedicated OFELIA (http://www.fp7-ofelia.eu/) island (with at least 5 OpenFlow switches and 5 servers) within i2CAT facilities.

The implementation of the Experimenta facility is an ongoing work. This is indicated by the “under-construction” signs on the figure, meaning that the annotated component is already fully functional but is currently being improved. For example, in the case of OCF and OFELIA and FIBRE islands. Those components are working and being used in experiments, but at the same time, new functionalities are added or improved or bugs fixed in the OCF or new hardware is attached to the testbed.



Compared to the summary figure presented before, the situation at the i2CAT Ofelia island is characterized by the following properties:

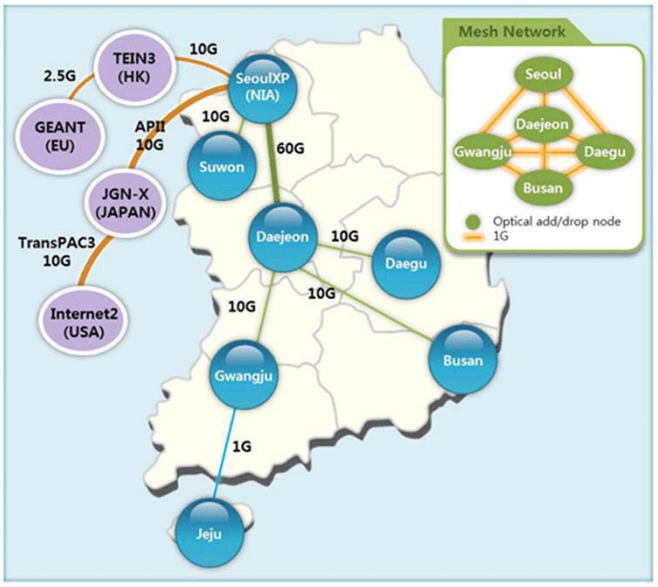
- Flack and Omni are not supported
- OMF6 and hence FRCP are not supported. Nepi is therefore also not supported.
- Zabix, CollectD and Nagios are not supported. Instead, Zenoss will be wrapped in an OML stream
- The depicted resources are specific: three optical reconfigurable 2.5/10G ROADM-based nodes, establishing a multi-wavelength fibre ring throughout the city of Barcelona; a pool of distributed virtual machines and the dedicated OFELIA (http://www.fp7-ofelia.eu/) island (with at least 5 OpenFlow switches and 5 servers) within i2CAT facilities.

KOREN testbed

FED4FIRE

iMinds
CONNECT.INNOVATE.CREATE

Connectivity

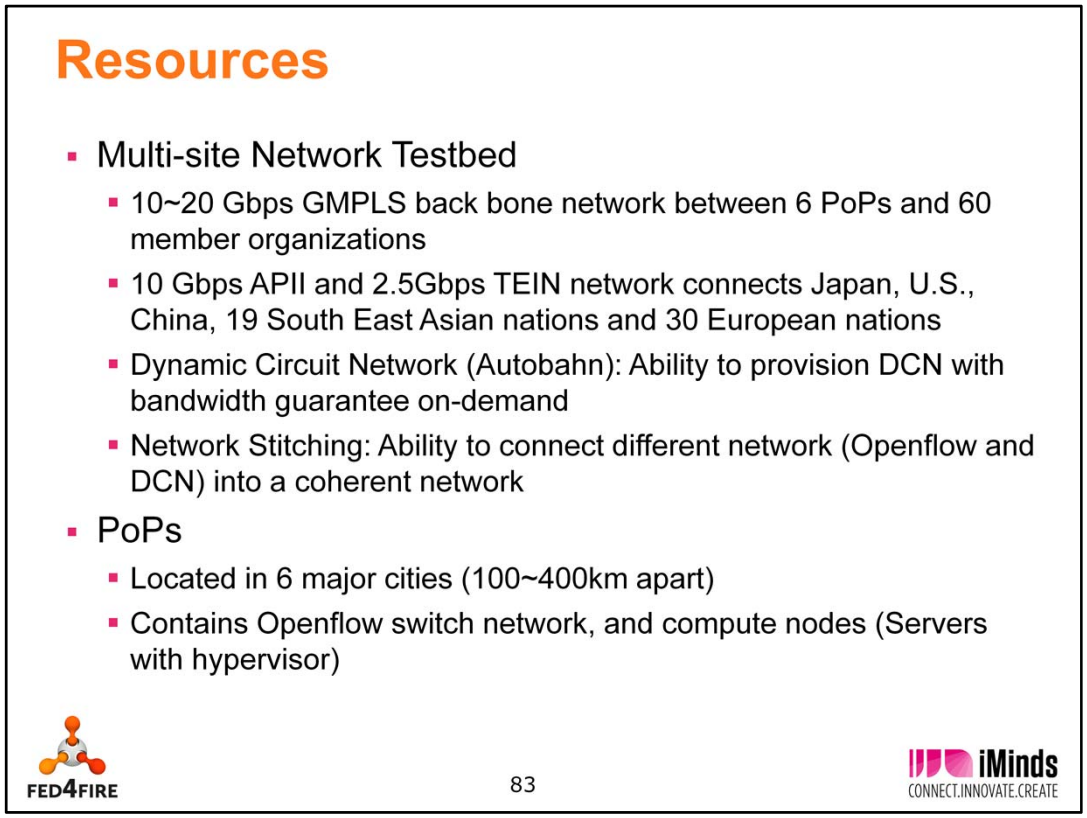


FED4FIRE

82

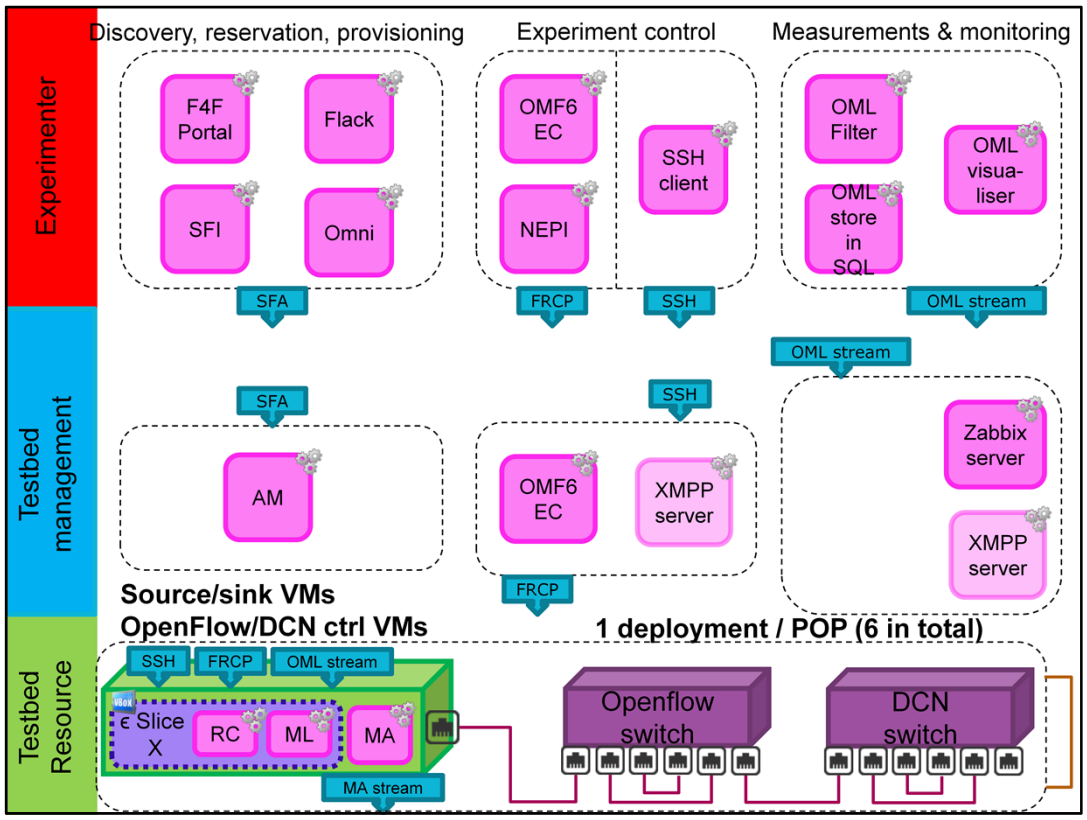
iMinds
CONNECT.INNOVATE.CREATE

As depicted on the slide, the KOREN network is connected to different important international research networks: TEIN3, GEANT, JGN-X and Internet2



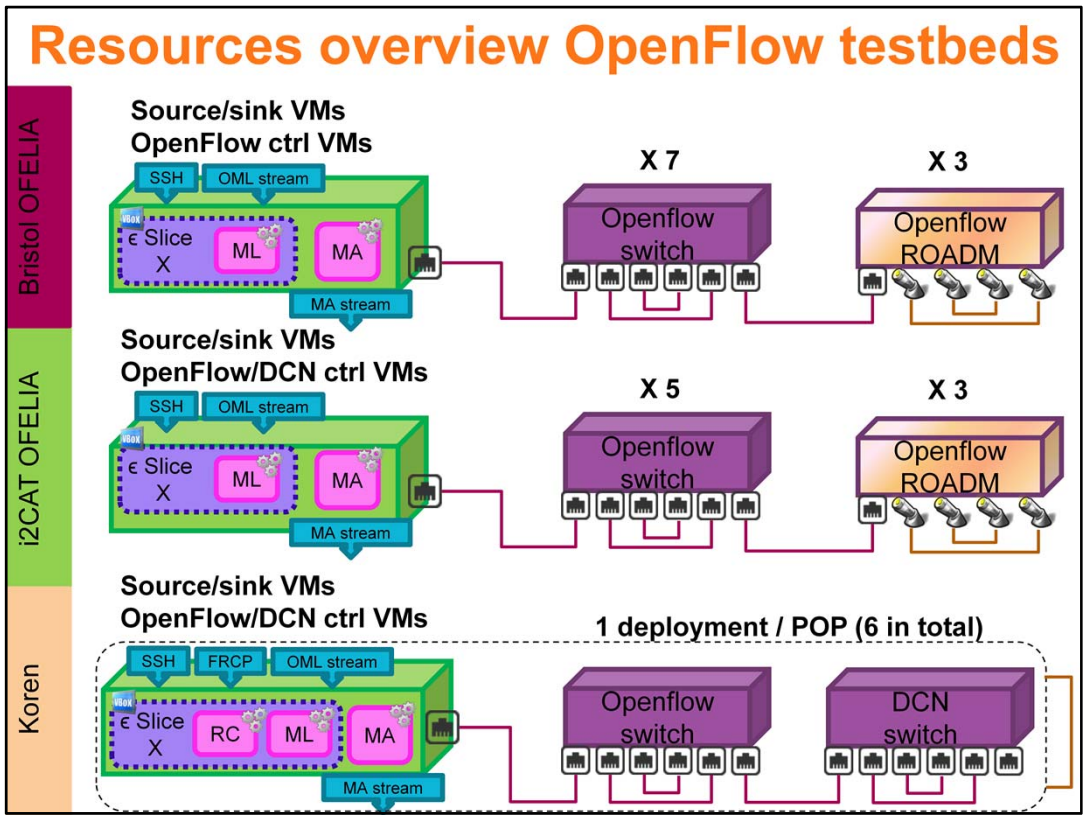
To summarize: Koren is a high-speed research network interconnecting 6 POPs in Korea (with support for bandwidth on demand), and also connecting to other international research networks. At the POPs, there are some OpenFlow switches and DCN switches, and servers that can host virtual machines. An experiment looks as follows:

1. VM's are created that act as data sources and/or sinks in the experiment. These are physically deployed at one or more POPs.
2. At every POP of interest, the experimenter can reserve several ports of the OpenFlow and DCN switches. To control how traffic will be forwarded over these ports, the experimenter deploys a new VM on which he installs a controller such as NOX.
3. If the experiment makes use of several POPs, the experimenter can reserve dedicated bandwidth between them. Because of the high-speed link to Geant, it is also possible to combine this Koren part of the experiment with other components belonging to the Fed4FIRE federation. But bandwidth cannot be guaranteed outside of the Koren Network.



Compared to the summary figure presented before, the situation at the Koren testbed is characterized by the following properties:

- Nagios and CollectD are not supported
- The depicted resources are Koren specific: 6 sites in Korea (POPs) interconnected using optical fibre, possibility to reserve bandwith between the POPs for your experiment. At each POP there are 1 or more OpenFlow switches, one or more DCN switches, and one or more servers to host VMs. These VMs can be used as data sources or sinks, or as controllers for the OpenFlow and DCN switches. As an experimenter you will add a set of switch ports to your experiment.


Resources overview OpenFlow testbeds
Bristol OFELIA
i2CAT OFELIA
Koren
Source/sink VMs
OpenFlow ctrl VMs
SSH
OML stream
ϵ Slice X
ML
MA
MA stream
X 7
Openflow switch
X 3
Openflow ROADM
Source/sink VMs
OpenFlow/DCN ctrl VMs
SSH
OML stream
ϵ Slice X
ML
MA
MA stream
X 5
Openflow switch
X 3
Openflow ROADM
Source/sink VMs
OpenFlow/DCN ctrl VMs
1 deployment / POP (6 in total)
SSH
FRCP
OML stream
ϵ Slice X
RC
ML
MA
MA stream
Openflow switch
DCN switch

The cloud computing testbed

- BonFIRE is a multi-cloud testbed for services experimentation
  - EPCC and Inria cloud sites
  - iMinds Virtual Wall testbed for emulated networks

BonFIRE

FED4FIRE

iMinds
CONNECT.INNOVATE.CREATE

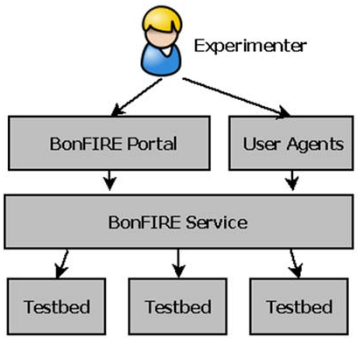
BonFIRE Fed4FIRE testbeds

FED4FIRE

iMinds
CONNECT.INNOVATE.CREATE

Introduction to BonFIRE

- Goal: experiments about cloud computing technology
- Large scale: 400 dedicated cores, more on-request cores available
- Single interface to all testbeds

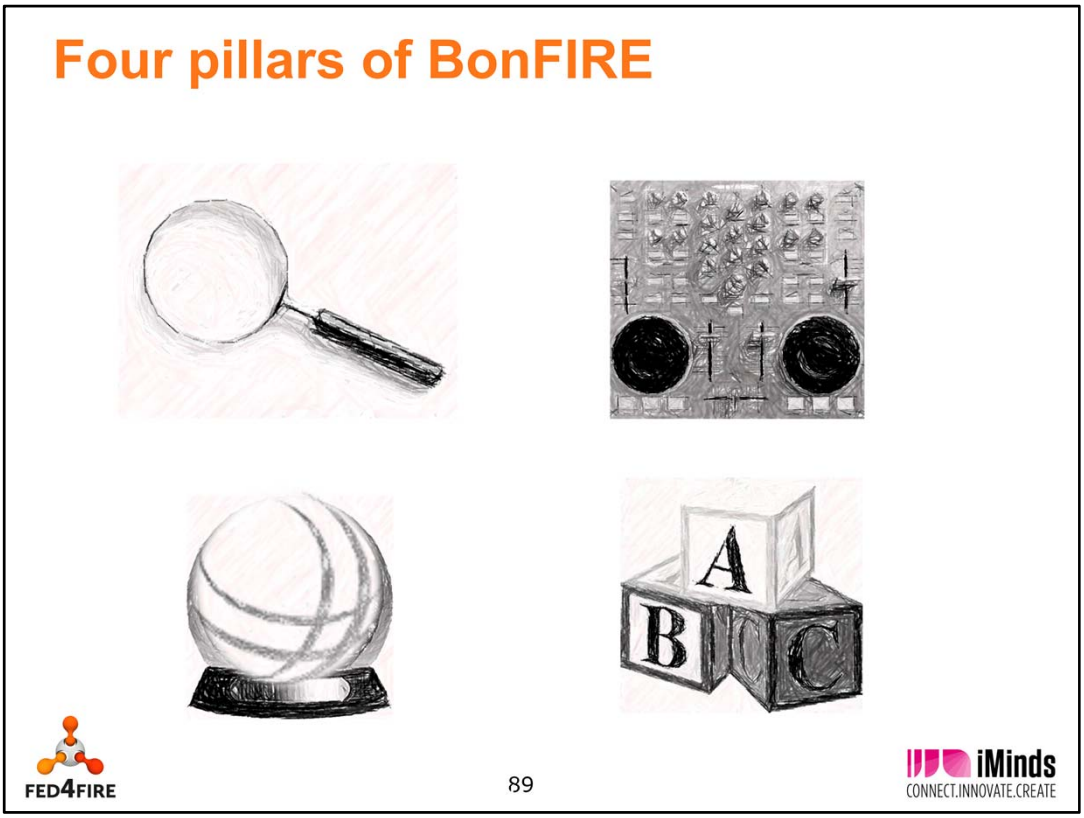


FED4FIRE

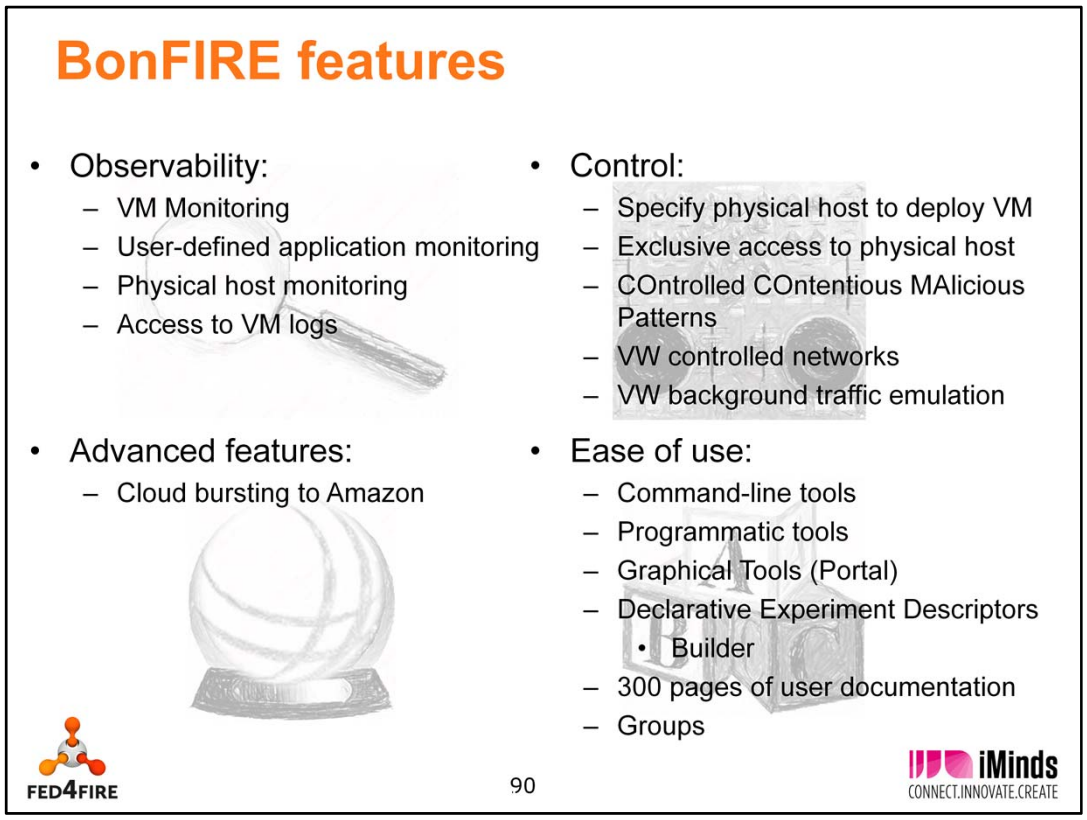
88

iMinds CONNECT.INNOVATE.CREATE

The focus of BonFIRE experiments is on cloud computing experiments. BonFIRE consists of six different BonFIRE testbeds, of which three are represented in Fed4FIRE: EPCC, INRIA and the iMinds Virtual Wall. In the context of cloud computing experimentation, the testbeds are only accessible through the BonFIRE Service APIs, not directly. Note that the Virtual Wall is also offered as a wired testbed as presented before. In that case it can be directly accessed.



BonFIRE has been designed around four principles: Observability, Control, Advanced Cloud Features and Ease of Use.

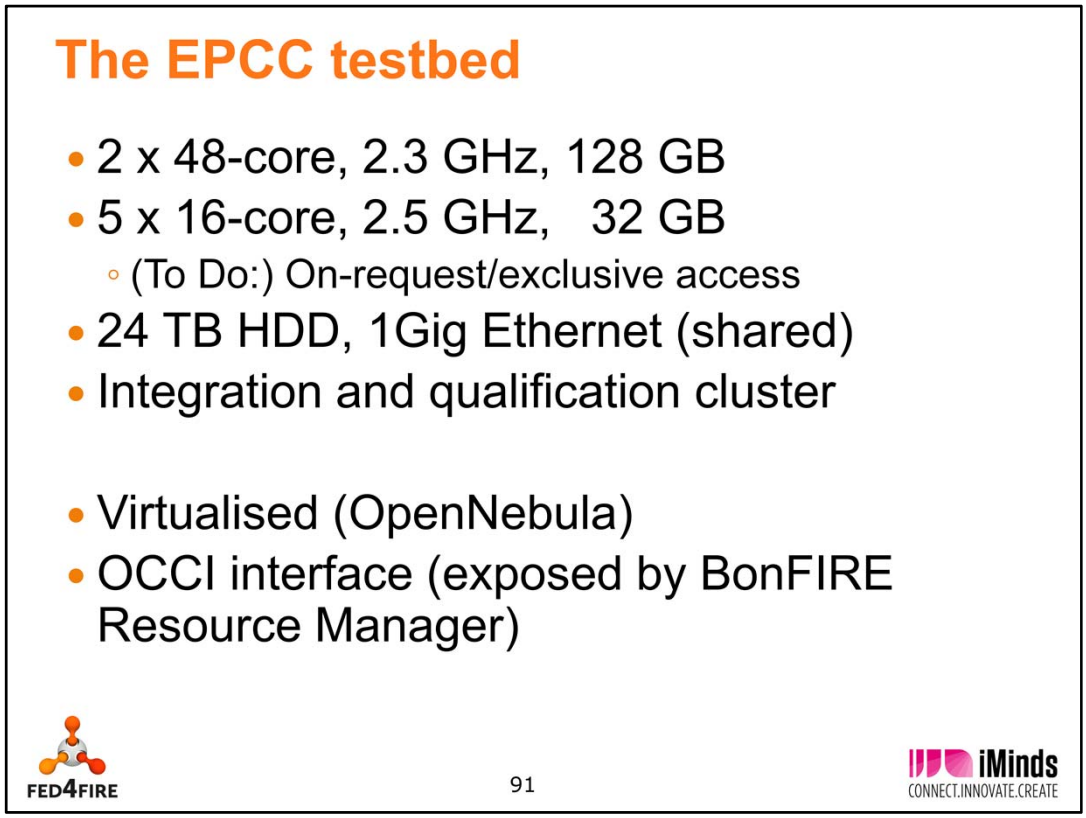


First of all, BonFIRE offers a cloud with ***Observability*** at all layers: physical, virtual and application. Cloud providers tend to hide information; BonFIRE exposes low-level monitoring of the infrastructure and gives access to time-stamped deployment logs.

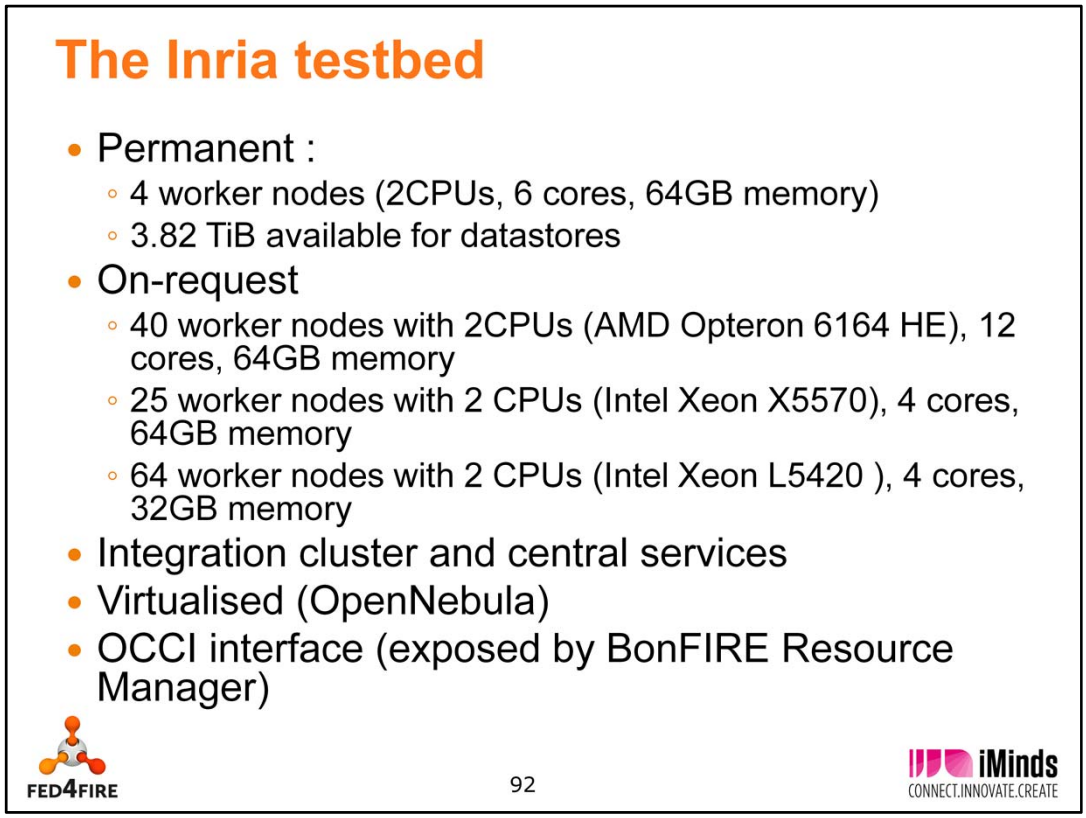
Then, BonFIRE puts the user in ***Control*** . On BonFIRE you can control placement of virtual machines, get exclusive access to physical nodes and even load a physical machine with a user-configured pattern. On the Virtual Wall facility the user can finely control emulated network parameters and inject them with background traffic.

BonFIRE delivers basic cloud features such as on-demand access and elasticity, also recognised the need for ***Advanced Cloud Features*** required by the future Cloud scenarios. Take Cloud bursting for example, where an application running on one cloud elastically expands to use resources on another, potentially Commercial cloud.

Finally although BonFIRE offers a rich set of features we wanted to ensure they are **Easy to use.** We offer a wide range of interfaces, Command-Line, Programmatic, Graphical, and Declarative, with on-line builders and documentation to facilitate adoption and use of the platform. We also offer functionality to allow experimentation groups to share access to their resources.



EPCC provides one of the five BonFIRE cloud facilities, which was commissioned in 2011Q3. Two physical hosts provide 4 AMD Opteron 2.3GHz processors each, with a total 96 cores and 256GB of memory. A separate front-end node offers 24TB of disk, available to all cores through NFS. The site runs the OpenNebula Cloud Manager over the Xen hypervisor, and exposes an OCCI-standard interface for experimentation. In-depth monitoring information at VM and hypervisor level is made available to the experimenters through the Zabbix framework and PI. separate server of a different architecture is also available and is used for development and deployment of the next BonFIRE releases. The site has a shared, 1GB link connection to G T, through the UK NREN JANET. The facility is dedicated to the FIRE project BonFIRE; we therefore intend to develop, deploy and support as much Fed4FIRE common functionality as required and as possible with the available means.



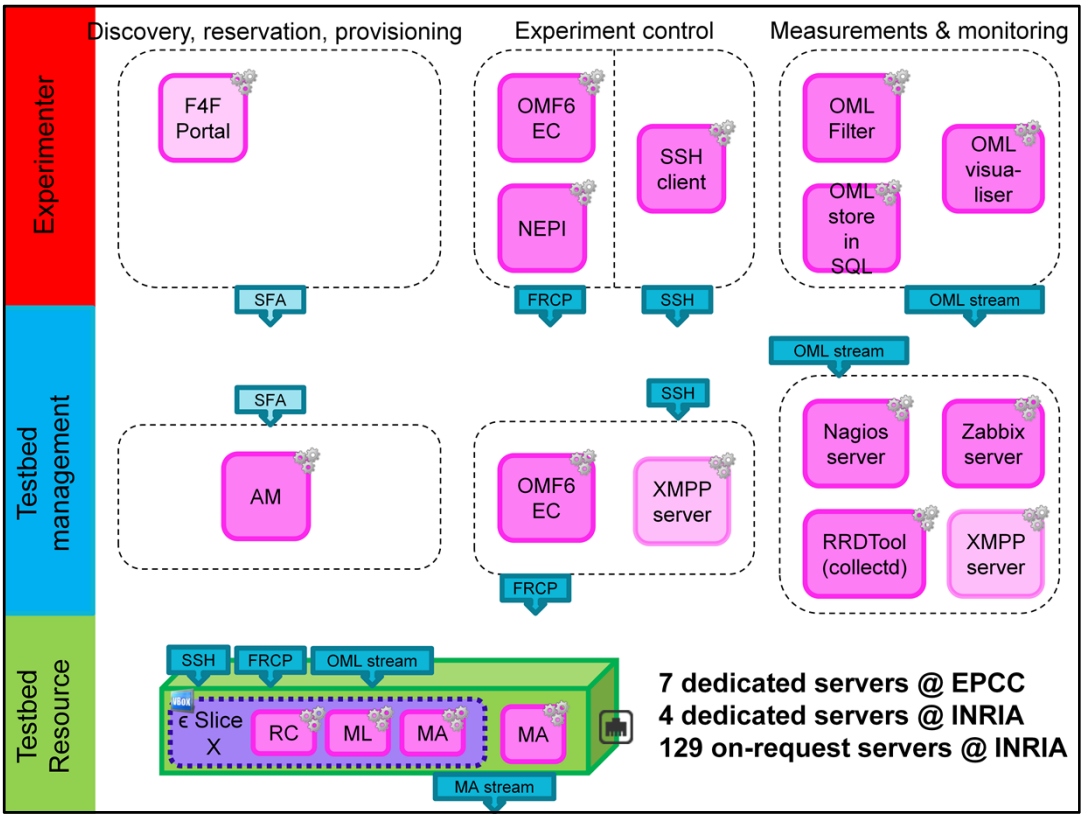
Inria provides one of the five BonFIRE cloud facilities (http://doc.bonfire-project.eu/R3.1/overview/infrastructure.html) . Inria's contribution is composed of a permanent set of resources, shared between users which was commissioned in 2012Q4 and of on-request resources taken from the local Grid'5000 node when at the request of BonFIRE users (BonFIRE usage is just an other reservation for the local Grid'5000 node). The permanent resources are provided by 4 physical hosts provide 2 Intel hexa core machines. A separate front-end node offers 3.82TB of disk, available to all cores through NFS. The site runs the OpenNebula Cloud Manager over the Xen hypervisor, and exposes an OCCI-standard interface for experimentation. In-depth monitoring information at VM and hypervisor level is made available to the experimenters through the Zabbix framework and API. The site has a shared, 1GB link connection to GEANT, through the French NREN RENATER The facility is dedicated to the FIRE project BonFIRE; we therefore intend to develop, deploy and support as much Fed4FIRE common functionality as required and as possible with the available means.

An experiment is worth a thousand words

- Hands-on experience allows experimenters interested in the Fed4FIRE open calls to better understand the BonFIRE offering.
- This is possible since BonFIRE is running a continuous, Open Access Initiative
  - Simple process for you to apply and go on BonFIRE free of charge
  - Just visit http://www.bonfire-project.eu/involved and fill in the short form.

FED4FIRE

iMinds CONNECT.INNOVATE.CREATE



Compared to the summary figure presented before, the situation at the BonFIRE testbeds is characterized by the following properties:

- Limited support for SFA: it only provides the mechanisms needed to include BonFIRE information in the testbed and tools directories on the Fed4FIRE portal, allowing experimenters to get acquinted with the testbed on a high level and to be linked to more detailed online documentation. Actual resource discovery, reservation and provisioning are not supported through SFA. Therefore the Fed4FIRE portal is partially supported, and the other tools (Flack, SFI, Omni) are not supported at all.
- The resources are specific for BonFIRE: powerfull servers to host the cloud VMs. Each virtual machine automatically includes the collectd monitoring agent, and on each physical server there is also an measurement agent running to provide monitoring information about the physical hardware on which the experiment's VMs are running. Every VM instance also supports OMF6, and hence FRCP and hence NEPI. OML is also natively supported on each VM instance. As mentioned before, there are dedicated servers deployed both at EPCC and INRIA, and on request additional servers from INRIA can be added to BonFIRE for a specific experiment.

Summary

- Fed4FIRE provides a heterogeneous set of FIRE testbeds to the experimenter.
- To perform experiments on these, common tools can be used.
- This way, Fed4FIRE allows the experimenter to
  - Create experiments that break the boundaries of the different FIRE domains
  - Easily access all the required resources with a single account
  - Focus on its core task of experimentation, instead of on practical aspects such as learning to work with different tools for each testbed, requesting accounts on each testbed separately, etc.

Questions

info@fed4fire.eu

FED4FIRE

iMinds
CONNECT.INNOVATE.CREATE

Acknowledgement

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