FEDERICA
WWW.FP7-FEDERICA.EU

e-infrastructure

Federated E-infrastructure Dedicated to European Researchers Innovating in Computing network Architectures

Co-funded by the European Commission within the Seventh Framework Programme. Grant Agreement No. RI-213107

Deliverable DJRA2.3: Virtual network architectures

Version 1.5

Dissemination Level	Public
Contractual Date of Delivery	31 January 2010
Actual Date of Delivery	25 March 2010
Editor	P. Szegedi (TERENA)
Contributors	P. Sjödin, M. Hidell (KTH) L. Giraudo, A. Bianco (PoliTo) M. Grammatikou, M. Dramitinos, G. Stamoulis (ICCS/NTUA) J. Palus, K. Kurowski (PSNC) D.R. Lopez, A. Daryanani (RedIRIS) J. Pons (I2CAT) J-M. Uzé (Juniper)
Reviewers	C. Cervelló-Pastor (UPC) - *on behalf of JRA1* M. Campanella (GARR) M. Potts (Martel)

Abstract

This deliverable DJRA2.3 '*Virtual network architectures*' provides an update to the preliminary investigations of DJRA2.1 concerning the FEDERICA architectural principles, including the federated authentication, authorization and security aspects, the novel architectural paradigms foreseen by FEDERICA, as well as the fairness concepts related to the proposed business model for virtualised infrastructures in general. The deliverable also contains the validation of architectural principles in various experiments (e.g. software routing and flow-based virtualisation) towards defining and prototyping novel paradigms.

Executive summary

The deliverable DJRA2.1 *'Architectures for virtual infrastructures, new Internet paradigms and business models'* was a preliminary, high-level architectural investigation within the FEDERICA project. This deliverable DJRA2.3 *'Virtual network architectures'* contains a report on the work that has been performed by the JRA2 partners (between Months 8 and 25 of the project). This also concludes the main achievements in JRA2 concerning the FEDERICA architectural principles, including the federated authentication, authorization and security aspects, the novel architectural paradigms foreseen by FEDERICA, as well as the fairness concepts related to the proposed business model for virtualised infrastructures in general.

Note that the final deliverable from JRA2(i.e.DJRA2.4 'Final prototype testing' of JRA2) is planned to be just a brief technical report on the results of the final IPsphere-FEDERICA interoperability prototype tests (defined in DJRA2.2) and will be done jointly by JRA2 and SA2 (in particular the NOC) partners.

After an extensive review of existing (and planned) architectural frameworks, concepts, project plans, and implementations reported in DJRA2.1, the FEDERICA related architectural requirements were collected and consolidated with the main FEDERICA principles summarised (*see Section 2.5 of deliverable DJRA2.1*). The FEDERICA physical infrastructure was under construction at that time, therefore, the basic design principles were fed back into the implementation process.

Consequently, JRA2 (being a research activity) was directed towards searching for novel architectural paradigms on virtual infrastructures such as FEDERICA is deploying. Two major research topics of interest to the partners were identified since they were interesting trends focused towards future paradigm shifts These were:

- Application of multi-stage software router architectures implemented on top of virtualised hardware infrastructures (studied by PoliTo)
- Investigation of flow-based traffic management (i.e. OpenFlow) as a novel slicing concept in virtualisation capable network environments (studied by KTH).

JRA2 partners have experimented with both topics and plan to use the FEDERICA infrastructure to validate the design principles pre-defined and enforced during the FEDERICA implementation process. The main considerations, observations and results are summarised in this deliverable. In conclusion, the JRA2's state-of-the-art research activities (performed on the FEDERICA virtual infrastructure) prove that the FEDERICA architecture is agnostic, open and flexible enough to make experiments oriented towards future Internet architectures that may lead to major paradigm shifts in the future (researched by user projects).

The other objective of JRA2 is to establish a technical collaboration framework between FEDERICA and IPsphere. This was initially achieved by prototyping a gateway function (*see deliverable DJRA2.2 and Milestone MJRA2.1 due to Month 15*) for integrating the MANTICORE software (based on the IaaS Framework, which is a descendant of the UCLPv2 project) in the IPsphere model. Note that in FEDERICA, an enhanced version of the MANTICORE software is used to control the V-Nodes while the Juniper SRC software controls the Juniper boxes. The prototype framework allows the user to interconnect (i.e. vertically federate) isolated slices of any virtualised infrastructure (like FEDERICA, OneLab, GENI, etc.) to create a single unified slice under the user's control. Beyond the prototyping work the two major aspects of the interoperability (or federation framework) related to JRA2 activities are:

- Enhanced functionalities of the Authentication and Authorization Infrastructure (AAI) proposed for FEDERICA (studied by RedIRIS and PSNC).
- Business related issues brought up by IPsphere (studied by I2CAT and Juniper) as well as an optimised business model for FEDERICA, and similar virtualised infrastructures (studied by ICCS/NTUA).

What is described in this deliverable is the concept of the proposed AAI infrastructure for FEDERICA and the practical details of the federated SSH access. These were implemented by the Service Activities in the first phase of FEDERICA.

Business models are important for the FEDERICA research and future exploitation of the concept although the project is not primarily aimed at commercial exploitation. JRA2 has proposed and experimented with a FEDERICA business model optimised for virtualised infrastructures. A complete specification of the assumed FEDERICA market mechanisms, the associated business and value-related issues as well as a representative example are summarised in this deliverable. Theoretical research on the fairness issues in virtualisation capable environments is also performed (studied by TERENA).

Although not part of the original project Description of Work, the JRA2 partners have agreed to develop a simulation framework for FEDERICA as additional work. The initial need for this simulator emerged from the fact that the actual usage of the FEDERICA infrastructure was not sufficient at that time to provide enough inputs to perform the fairness, resource allocation, and business-related studies by the Joint Research Activities. That is why the simulation framework needed to be developed while other useful aspects also emerged. Finally, the simulator supports:

- Planning and development of the FEDERICA physical infrastructure (potentially useful for planning any follow-on project of FEDERICA).
- The slice creation process and its effect on the network performance (useful for the NOC and daily operation)
- Better understanding of the users' behaviour in a virtualised environment (useful for business-related research and potential commercialisation).

This deliverable contains a brief description of the FEDERICA simulation framework (written by PSNC).

Note that the simulator is available to all the FEDERICA project partners on the FEDERICA Wiki page.

Table of Contents

1 Introduction and motivations

This deliverable DJRA2.3 *'Virtual network architectures'* contains a report on the work which has been completed by the JRA2 partners between Month 8 and 25 of the project. The main achievements in JRA2 concerning the FEDERICA architectural principles including the federated authentication, authorization and security aspects, the novel architectural paradigms foreseen by FEDERICA, as well as the fairness concepts related to the proposed business model for virtualised infrastructures in general has been described.

1.1 Document overview

The structure of this deliverable follows a similar structure found in the first deliverable (and refers to it in many places), while also providing an update.

- Section 1 '*Introduction and motivations*' provides the document overview.
- Section 2 '*Towards architectural paradigms on virtual infrastructures*' summarises the FEDERICA architectural principles focusing on the JRA2 related issues, namely the IaaS framework, the interoperability prototype, and the business processes covered by IPsphere. The validation of the applied design principles are done by studying novel architectural concepts (such as multi-stage software routers and OpenFlow, which may lead to paradigm shifts in the future) on top of the FEDERICA virtualised infrastructure.
- Section 3 '*Authentication and Authorization Infrastructure for FEDERICA*' includes all the security aspects related to internal risks and external threats. It gives details of the AAI infrastructure proposed for FEDERICA and the practical usage of the federated SSH access to FEDERICA resources.
- Section 4 '*Simulation framework for FEDERICA*' describes the simulation framework developed by PSNC. A brief description of the simulator and its foreseen benefits are explained in this section. The results of a theoretical study on fairness issues in FEDERICA are also given.
- Section 5 '*Business model for FEDERICA*' gives all the details of an assumed FEDERICA market mechanism. A representative example summarises the business model and value chain proposed for virtualised infrastructures and for its users.
- Section 6 '*Summary*' summarises the JRA2 work and the main achievement pointing the way towards any follow-on project of FEDERICA.

1.2 How to read the document

It has been the editor's intention to make deliverable DJRA2.3 as easy to read as possible. It is the concluding deliverable of the JRA2 main activities. It summarises all the major activities and research areas covered by JRA2, focusing rather on the architectural level aspects and not going into every single detail.

All the details of the work performed by the partners under JRA2 can be found in the referred articles, publications, documentations, and previous deliverables.

The reader is also advised to refer back to the previous deliverables and documentation:

- It is highly recommended to start with reading the deliverable DJRA2.1 '*Architectures for virtual infrastructures, new Internet paradigms and business models*' including background information for all the research areas covered by JRA2. This deliverable DJRA2.3 provides an update on those topics.
- For more technical people, it is recommended to read the deliverable DJRA2.2 '*Prototype for interoperability between IPsphere and MANTICORE*'. That deliverable contains the detailed description of the FEDERICA – IPsphere interoperability prototype and also the description of the enhanced MANTICORE functions.
- This deliverable should then be easy to read. It gives an overview of all the JRA2 activities related to both the technical and non-technical issues.
- The detailed information for highly interested readers is available in the appendix and in the referred publications.
- The final (internal) test results of the interoperability prototype will be available in the last JRA2 deliverable DJRA2.4 '*Final prototype testing*'. That one will be a brief technical report, at the end of the FEDERICA project.
- The final JRA2 research results on novel architectures, being performed on FEDERICA slices, will be reported in deliverable DNA2.3 '*FEDERICA Usage Report*'.

2 Towards architectural paradigms on virtual infrastructures

The overall objective of the task TJRA2.1 "*Architectural paradigms for virtualised infrastructures*" under JRA2 was defined at the beginning of the project. It was anticipated that the FEDERICA virtualised infrastructure makes it possible to support innovative research activities that will lead to tangible results within a relatively short timeframe (in contrast to the 10-20 years horizon of "clean slate" initiatives). The innovative research work is part of the JRA2 activity in FEDERICA. It was envisioned that the research activities would address improvements in the following particular areas (*see page 63, Annex I "Description of Work"*):

- Virtualisation support to enable sharing of resources within and between infrastructures in a way that allows virtualised resources to be directly controlled and managed by end-users.
- Support for experimentation with different forwarding and switching paradigms.
- Mechanisms and techniques for the separation of control and data planes, allowing for a fully experimental virtualisation.

The JRA2 partners have considered these areas and proposed related research activities as follows:

- Addressing the area of sharing of resources within and between infrastructures, the IPsphere – FEDERICA interoperability prototype has been developed by JRA2 as the main contribution to the project. The direct control and management of virtualised resources by end-users has been supported by the enhanced MANTICORE software developments.
- The experimentation with different forwarding and switching paradigms has been addressed by important research in software routers. A multi-stage software router concept and its possible implementation on virtualised hardware infrastructure (i.e. a FEDERICA slice) have been studied.
- Finally, in order to investigate the potential mechanisms and techniques for the separation of the control and data planes and to allow for a fully experimental virtualisation, the OpenFlow standard has been brought into the picture. The proposed OpenFlow-based virtualisation architecture study aims to separate the control plane (i.e. flow management function) from the data plane.

This chapter summarises the FEDERICA architectural principles focusing on the JRA2 related issues, namely the IaaS framework, the business processes covered by IPsphere, and the interoperability prototype.

The validation of the applied design principles were done by studying novel architectural concepts on top of the FEDERICA virtualised infrastructure (such as multi-stage software routers and OpenFlow, both of which may lead to paradigm shifts in the future). FEDERICA slices are planned to be requested for the software router research and the OpenFlow investigations. If the research studies are successful, this will prove that the deployed FEDERICA infrastructure is agnostic, open, and flexible enough to make innovative experiments oriented towards novel networking paradigms. Explicitly, it validates the FEDERICA architecture's design principles.

2.1 FEDERICA architectural principles

An extensive review, done by the JRA2 partners, on existing and planned architectural frameworks, concepts, project plans, and implementations related to FEDERICA has been reported in DJRA2.1. The review resulted in a set of requirements which were considered during the FEDERICA architectural design. The main design principles were:

- Versatile, so as to accept different types of technology
- Distributed
- Be part of the Internet, in order to provide a realistic environment and the possibility to test the migration path from the current technologies to the new ones
- Capable of federating with other facilities, to offer a richer environment to its users
- Offer the capability to ensure reproducibility behaviour of its elements
- Offer full control and configuration capability
- Encompass all the network layers, including applications
- Pose the minimum number of constraints to its users

The subsequent application of these principles during the FEDERICA architectural design and the implementation of the infrastructure ensured that the FEDERICA infrastructure became unique, in the sense that it differs from similar virtualisation capable infrastructures of today.

Taking the FEDERICA principles into account, we can say that the well-known and widely used PlanetLab [Pan] did not provide full reproducibility, since it is deployed on a ´best effort´ platform which is the public Internet. The most important limitation of the OneLab/PlanetLab architecture is that the underlying transport network layer is not part of PlanetLab and this lack of control may result in under provisioning and affect experiments. In FEDERICA, a 'real network substrate' has been created together with some engineering methods ensuring the reproducibility of the experiments. Even in Emulab [Emu] the network is a fake, although the experiments can virtually be scaled very well. The current implementation of VINI is based on PlanetLab [Vin]. PL-VINI allows real routing protocols to run on a virtual network topology that is implemented as an overlay on PlanetLab. This feature is also part of the FEDERICA concept allowing research on Layer 3; however, the scope of FEDERICA has also been extended to the lower layers (i.e. raw Ethernet).

Project: / Feature:	OneLab2/ PlanetLab	Emulab	FEDERICA
Allowed Operating system	Fixed	Fixed	User's choice (almost any)
Control of lower layers	No	Emulation	Control down to raw Ethernet
IP	Mandatory	Used to connect then emulation	Used to connect, then not needed
Choice of physical delay and capacity	No	Emulation	Yes (up to 1 Gb)
Guarantees of reproducibility	No	Emulation	Yes
User access limitations	Almost none, at any time	Almost none, at any time	Regulated by a User Policy Board
Cost	Limited	None	None
Scalability	Medium/large	Large	Limited

Table 1 Comparison between FEDERICA and similar facilities (OneLab/PlanetLab/Emulab)

Although FEDERICA follows very similar substrate, slicing and federation principles as those defined in the U.S. initiative GENI [Gen], FEDERICA narrows down the scope in terms of resource types and their behaviours. Compared to GENI, the main simplifications are twofold: the usage of fixed computing resources only, and the variety of user-area resources that are restricted to the existing NRENs' network capabilities.

It is important to note that FEDERICA has limitations in terms of performance because of the substrate design. FEDERICA is not about performance but about design, testing new ideas, and principles; these are the main motivations behind the FEDERICA core concept.

The current implementation of the FEDERICA infrastructure is well described in the deliverables DSA1.1 and DSA1.2. The final infrastructure will be detailed by SA1 in deliverable DSA1.3. Our aim here is not to define the detailed FEDERICA architecture, just to summarise the architectural principles related to JRA2 research. Note that JRA1 is working on the details of these principles and on the possible solutions on how to apply those in the FEDERICA infrastructure (*see deliverables DJRA1.1 and DJRA1.2*). We need to understand the concept of virtualisation used by FEDERICA first to identify the potential new directions of virtualised system design that may cause major paradigm shifts in the near future.

In the following sections the FEDERICA slice creation workflow process and some business-related issues brought up by the potential commercial exploitation of the concept are briefly summarised. As the main contribution of the JRA2 activity, the importance of an IPsphere-FEDERICA interoperability prototype (designed and implemented by JRA2) is also discussed. The summary of these architectural issues considered by FEDERICA may help to understand the research directions taken up by JRA2.

2.1.1 Workflow providing virtualised infrastructure services on-demand

The objective of JRA2 is oriented towards defining and prototyping novel paradigms on virtual infrastructures and new functionalities that are envisaged for a combined approach to network and systems virtualisation. Some current tools already provide virtualisation functionalities of network elements at lower layers. A major challenge to be addressed by JRA2 is including virtual machines based on large computers, together with dedicated routers and virtualised network resources within the same workflow. This may allow dynamically providing in one step a whole experimental scenario to allow researchers to test their protocols or architectures.

Figure 1 depicts the FEDERICA high-level slice creation workflow that combines the virtualisation of traditional network elements (i.e. routers and switches) and systems (i.e. any functionality such as software router, host, DNS server, etc.) implemented on top of computer machines.

The details of the slice creation process and the associated control levels of the architecture can be found in Section 3 of deliverable DJRA2.1.

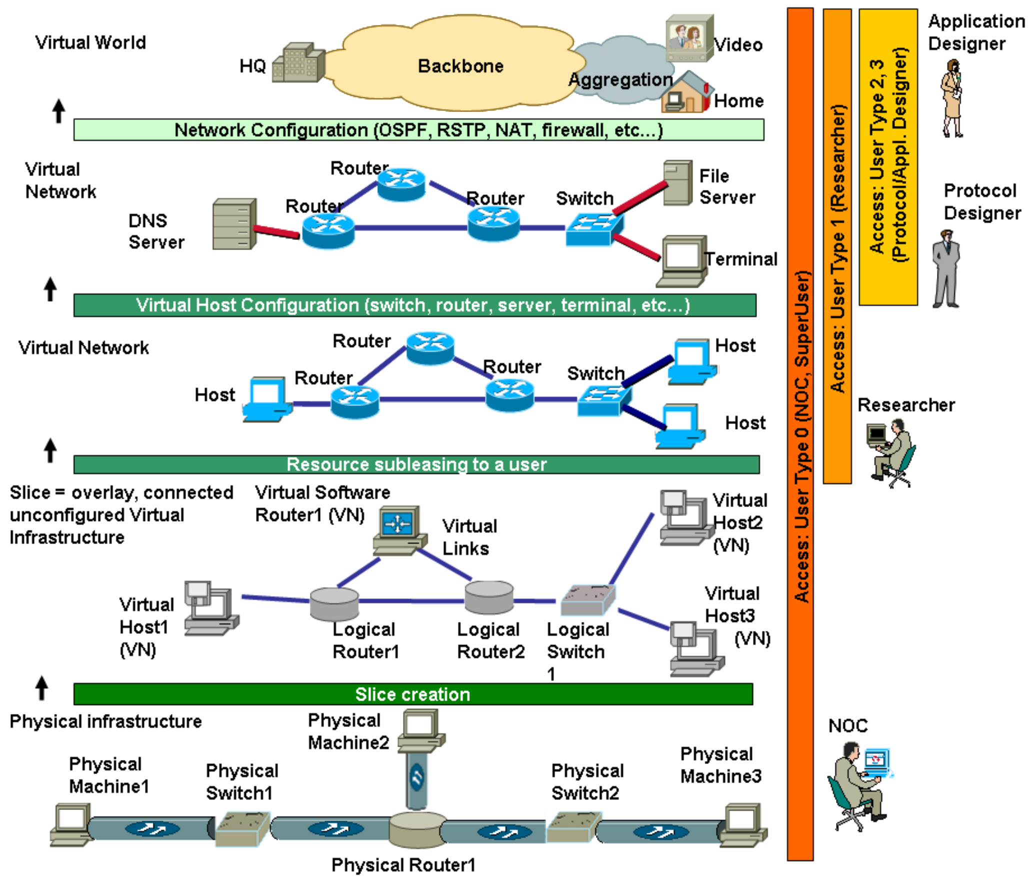

Figure 1 FEDERICA slice creation workflow (high-level)

In conclusion, it was found by JRA2 that the IaaS framework [Ias] is an obvious choice to develop new services for infrastructure virtualisation in FEDERICA because the services could follow the same standards, use common functionalities, be integrated into a global system, and interoperate easily. The structure of the services is very flexible and the orchestration can be restructured. The detailed investigation done by JRA1 concluded that the MANTICORE software [Man] (based on the IaaS framework, which is a descendant of the UCLPv2 project) can easily be enhanced to make it capable of creating and controlling virtual machines in remote hosts running virtualisation software. The joint research architectural studies supplied these conclusions back to the services actively.

2.1.2 Business processes

Following on from this work, the JRA2 activity, which is working more on the conceptual level and looking into the future, the importance of commercial exploitation of the virtualised infrastructures has been identified. The understanding and the potential automation of the business processes are key elements.

Once the physical infrastructure owner is capable of virtualising the infrastructure elements (i.e. the physical resources) and the administrative owner can create and control the virtual resources on-demand (e.g. accessing MANTICORE and/or the Juniper SRC in FEDERICA) in the given domain, the inter-domain decomposition and abstraction of services remains an important step towards the end-users (see Figure 2). The automation issue of the business processes creating multi-domain services are covered by IPsphere [Ips]. IPsphere includes interfaces between the resource-based view of the infrastructure or element owners and the service-based view of the administrative owners.

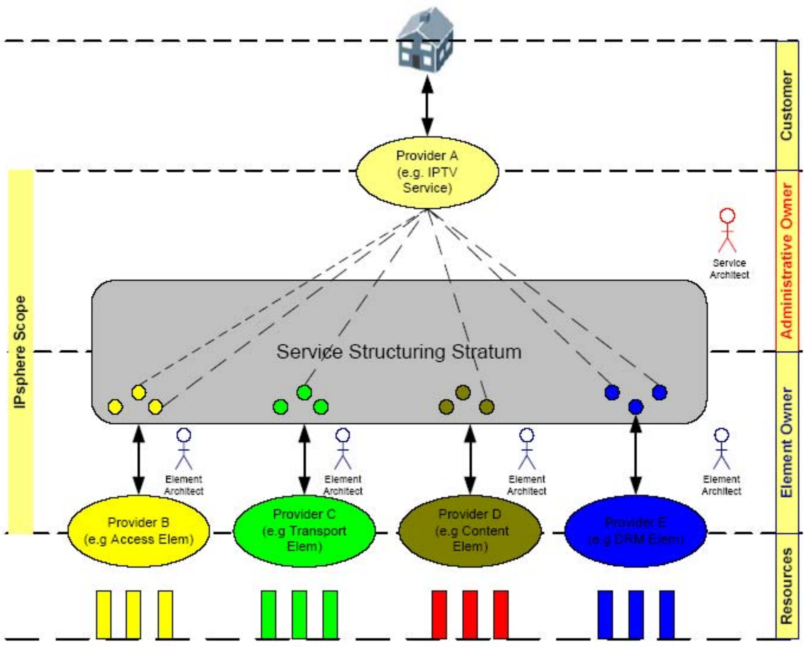
Customer
Provider A (e.g. IPTV Service)
Administrative Owner
Service Architect
IPsphere Scope
Service Structuring Stratum
Element Owner
Element Architect
Element Architect
Element Architect
Element Architect
Provider B (e.g Access Elem)
Provider C (e.g Transport Elem)
Provider D (e.g Content Elem)
Provider E (e.g DRM Elem)
Resources

Figure 2 IPsphere framework context diagram

IPsphere is a standard framework that has the unique objective that enables an automatic business process between multiple stakeholders to activate a service along with associated physical and virtual distributed resources. It is network independent (and thus technology and vendor agnostic), multi-domain, and it constitutes a Service Oriented Architecture layer [Soa] that can be integrated in any existing framework. The IPsphere framework has strong commonalities with the service layer being explored in FEDERICA for the orchestration of virtualised resources from multiple stakeholders.

More details about IPsphere can be found in Section 2.4.2 of deliverable DJRA2.1

2.1.3 Interconnection of virtualised infrastructures

The main objective of JRA2 was to develop an interoperability prototype between IPsphere and FEDERICA (i.e. the MANTICORE software and Juniper SRC used by FEDERICA). *The details of the prototype have been reported in deliverable DJRA2.2.*

The basic concept is that in a fully virtualised, multi-domain environment of the future networks, a comprehensive inter-domain framework capable of negotiating end-to-end paths across several network providers and virtual slices are needed. In general, the IPsphere framework can fulfil this requirement. With the FEDERICA prototype, the aim is to demonstrate and validate that the various virtual slice management tools (e.g. Juniper SRC and MANTICORE used by FEDERICA) and its functions can be integrated into the IPsphere model in a standardised way.

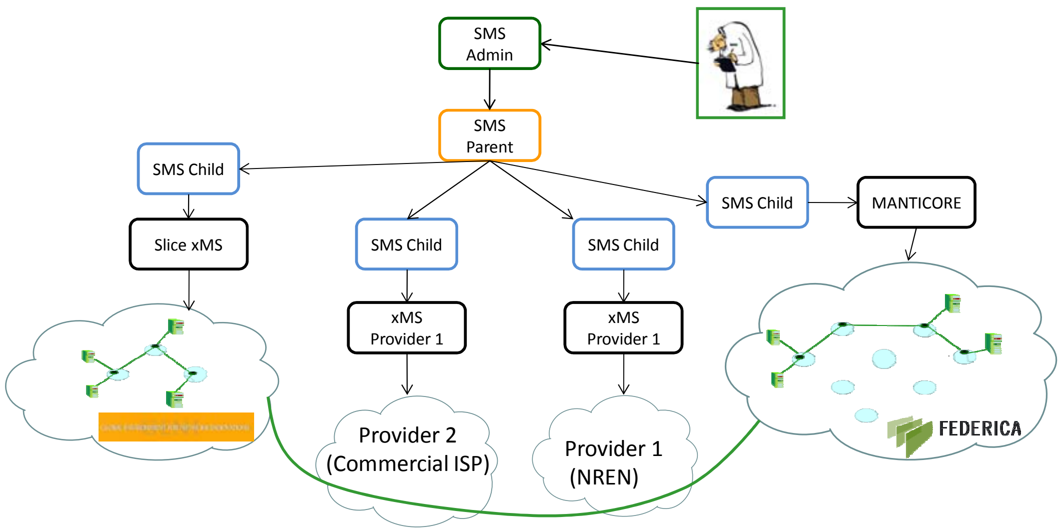

Figure 3 Automatic (user-controlled) service activation over various domains

As one of the service-oriented use cases demonstrates (see Figure 3), the administrative owner (via Service Management System, SMS Admin) can activate the multi-domain service that contains the virtual infrastructure elements (in Service Management System, SMS Child) created and subleased by the element owners in advance.

Additional use cases and the validation of the concept can be found in deliverable DJRA2.2.

In the next section, further research topics (of interest to the JRA2 partners) are introduced. These research topics are focused on the investigation of novel architectural paradigms which can be applied in virtualised environments but can also use FEDERICA slices to perform its research. So, explicitly, these topics are capable of validating the FEDERICA design principles (described above) by supporting users to make innovative research that is reproducible and may lead to revolutionary solutions, and even challenge the structure of the traditional OSI layer of network stack protocols.

2.2 Validating architectural principles by studying novel paradigms

Beside the major tasks of defining the workflow and developing an interoperability prototype taking the virtualised services and automatic business processes into account, JRA2 (being a research activity) was directed towards searching for novel architectural paradigms on virtual infrastructures (such as FEDERICA is deploying).

Two major research topics of interest to partners were identified as interesting trends orienting towards future paradigm shifts.

- Supporting novel switching, forwarding paradigms is the aim of FEDERICA. Software routers (by means of open source software and off-the-shelf hardware) can be the basic enabler of this feature. Software routers are the key component of current FEDERICA and may be essential parts of future routed networks (i.e. the future Internet).

  PoliTo has significant experience in the field of software routers, leading to a viable approach to router building. From the software point of view, the use of open source operating systems (e.g. Linux) has proven to be a good platform on which to improve or implement router functionalities such as buffer management techniques or routing protocols.

- To understand the behaviour of future routed networks, a better knowledge is needed on the micro processes (i.e. packet levels) and on the macro processes (i.e. flow level) of the networks. The identification of traffic flows and the flow-based management of the networks are state-of-the-art research areas. OpenFlow [Opf] is an open standard that allows researchers to implement experimental protocols on packet networks. In JRA2, OpenFlow was considered as potential platform to create virtual IP network slices on a per-flow basis.

  KTH has done an extensive research on OpenFlow, especially on how to use OpenFlow as a flow-based virtualisation platform in future packet networks.

In the following sections, the details of the two research studies performed by the FEDERICA JRA2 partners are discussed. Since both research studies are planned to be implemented on a FEDERICA slice, the work can validate the FEDERICA principle of supporting novel paradigms.

2.2.1 Research on multi-stage software router architecture

Routers are the key components of modern packet networks in particular in the Internet. The demand for high-performance switching and transmission equipment keeps growing due to the continuous increase in the diffusion of information and communications technologies as well as new bandwidth-hungry applications and services based on video and imaging. Routers are able to support the performance growth by offering an ever-increasing transmission and switching speed, mostly due to the technological advances of microelectronics.

Contrary to what has occurred with personal computers where interface and protocol standards have been defined allowing an open and multi-vendor market for the hardware components, the field of networking equipment of routers and switches has always been characterized by the development of proprietary architectures. This leads to incompatible equipment and architectures, especially in terms of configuration and management procedures, as well as the requirement of specific network administrators to handle several proprietary architectures together or to be limited to a single vendor solution only.

This situation produced commercial practices that were not based on free competition and therefore the final cost of equipment is often too high with respect to performance and equipment complexity. Software routers based on off-the-shelf PC hardware and open-source software represent appealing alternatives to proprietary network devices because of the wide availability of multi-vendor hardware, the low cost, and the continuous performance evolution driven by the PC-market economy of scale. Indeed, the PC world benefits from the de-facto standards defined for hardware components that enabled the development of an open market with a wide availability of multi-vendor hardware, low costs offered by the large PC market, wide information available on their architecture, and the large availability of open-source software for networking applications, such as Linux, the Berkeley software distribution (BSD) derivatives, Click Modular Router, XORP, and Quagga [Osr].

Despite the limitations of bus bandwidth, central processing unit (CPU), and memory-access speed, current PC-based routers have a traffic-switching capability in the range of some gigabits per second, which is more than enough for a large number of applications. Moreover, keeping this in perspective, performance limitations are compensated by the natural PC architecture evolution, driven by Moore’s law. However, high-end performance cannot be obtained easily today with routers based on a single PC. In addition to performance limitations, several other objections can be raised to PC-based routers; e.g. software limitations, scalability problems, lack of advanced functionality, inability to support a large number of network interfaces, as well as the inability to deal with resilience issues to match the performance of carrier-grade devices.

To overcome some of the limitations of software routers based on a single PC, PoliTo proposed to create a large router exploiting multi-stage switching architectures, as presented in [Msa], [Msr], [Cmm] (see also Figure 4). The multi-stage router is based on the following three stages:

- Load balancing stage

- Interconnection stage
- Routing stage

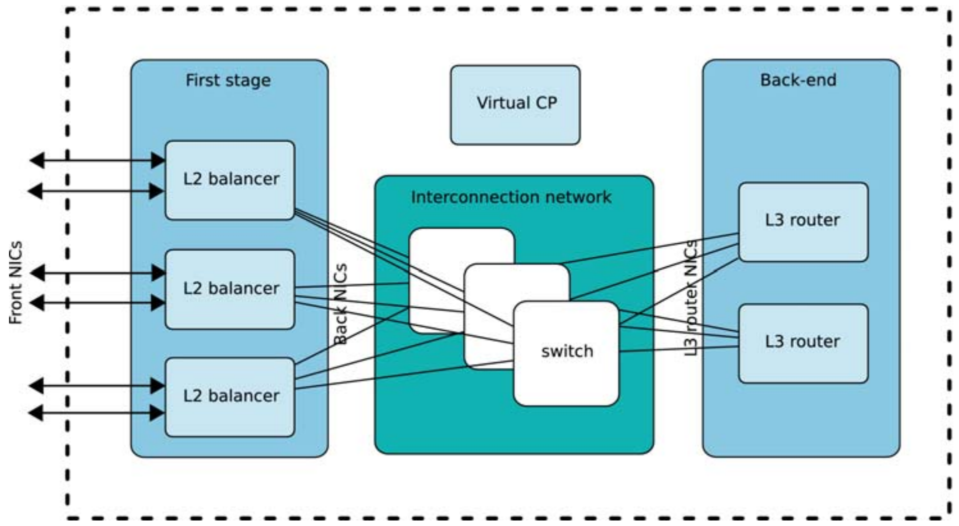

Figure 4 Multi-stage router architecture

Performance measurements show that routing capabilities may scale up almost linearly with the number of elements, as reported in [Msr]. Thus, the main advantage of this architecture is the ability to scale up in performance and the number of ports. Furthermore, higher reliability due to the implementation of recovery mechanisms in the management plane can be obtained. However, the coordination among different elements is difficult to implement.

Some improvements, such as the introduction of energy management mechanisms to switch on/off elements needed/unneeded in the interconnection due to traffic fluctuations, and the utilization of virtualised elements to increase management flexibility and routing capabilities, were considered.

The on-going development at PoliTo is focusing on the realisation of the multi-stage router in a virtualised environment i.e. using virtual machines instead of physical elements to build up the multi-stage router. The main advantages of this approach are:

- Hardware independence. It is possible to move virtual elements from one physical server to another and to adapt deployed resource to the offered load (e.g. consolidation of routers in the minimum number of physical servers to save energy, simplified maintenance of physical servers, etc.)
- Larger flexibility in adding routing power and/or ports. Dynamical addition of resources by renting virtual units from cloud-computing environments (e.g. Amazon EC2 [Aec]).

Virtualisation technologies introduce nice features to multi-stage routers however, some issues need investigating, such as:

- Performance penalties: virtual machine monitors (e.g. VMware, XEN, etc.) suffer from performance limitations due to hardware abstraction and resource contention.
- Additional complexity for the management plane.
- Larger latency when virtual machines reside in different servers and/or in different networks.

The investigation of the novel multi-stage software router architecture implemented in a virtualised environment is performed in two steps. In the first step, the performance issues have been evaluated in the PoliTo laboratory (called LIPAR) taking into account the introduction of virtualisation technologies. In the second step, a FEDERICA slice has been requested to evaluate the architecture in a geographically extended, real network scenario.

Preliminary laboratory test results show that the overhead introduced by Virtual Machine Monitors (VMM) may be large, but it is possible to obtain good throughput for virtual routers by properly tuning virtual Network Interface Controller (NIC) drivers. Performance limitations were detected when using virtual load-balancers, such as Click Modular Router (CMR). In this case, the throughput performance is very poor, since CMR is heavily optimized to work with specific drivers for Intel NICs.

The performance tests running on a geographically extended scenario (provided by a FEDERICA slice) are still on-going as this deliverable is being written. However, it is clear that many aspects can be evaluated better in a real network scenario, such as the possibility to evaluate the feasibility of this approach in a multi-server scenario and the impact of latencies on performance.

The investigation will initially consider the network topology depicted in Figure 5. The multi-stage router is composed by six virtual elements and a virtual switch to interconnect front-end and the back-end stage. Three virtual elements are then configured as traffic generators and synchronised, to test data plane performance and control plane functionalities, similarly to the approach pursued in [Cmm].

Experiments will be conducted in the following areas:

- Feasibility evaluation: the multistage router has already been verified in a virtual environment (VMware, ESXi), where all virtual machines run on a single server.
- Performance evaluation (data plane): it will begin with the evaluation of single elements (L3-router, L2-balancers and eventually traffic generators) and then the throughput of the whole architecture.
- Control plane evaluation: the last phase will focus on the internal control plane, which may be affected by latency introduced by the network environment. The internal configuration protocol and some improvements related to dynamic configuration of load balancing and energy saving will be tested.

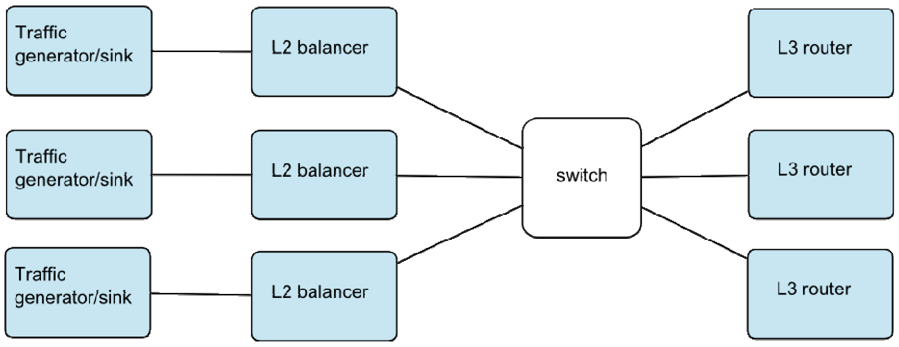

Figure 5 Basic scenario to be tested in a FEDERICA slice

The final research results will be reported in the deliverable NA2.3 (as an internal user project) at the end of the FEDERICA project.

In summary, it can be said that the main benefits of the proposed multi-stage router architecture on top of a virtualised infrastructure are:

- Flexibility and manageability,
- Providing the ability of renting capacity and devices instead of buying and directly managing the network and its components,
- Potential energy savings that can be obtained by hosting several virtual machines in few physical nodes.

Concerning the evaluations, being performed in a FEDERICA slice, the FEDERICA virtualised infrastructure is sufficiently flexible to implement the novel multi-stage software router architecture which exists on top. It also can provide a close-to-real network scenario where real latencies are introduced but in a reproducible, well-controlled environment.

The research on the multi-stage router architecture is innovative, taking into account the fact that it breaks the traditional architectural paradigms applied in commercial hardware router and switch design. However, it still fits into the OSI layered protocol stack and follows the basic FEDERICA virtualisation approach.

2.2.2 Research on OpenFlow as novel virtualisation platform

The research on OpenFlow [Opf], in contrast to the multi-stage router architecture research, is trying to propose a brand new virtualisation concept, different from what is applied in FEDERICA. This means that a novel approach on network virtualisation is proposed and this new approach will be evaluated and tested in the FEDERICA virtualisation capable infrastructure. If this research is successful, it will prove that FEDERICA is capable of performing research on architectures that are completely different from itself, which is a unique feature today.

In the following, a brief description of OpenFlow and how it can be used as a novel virtualisation platform are discussed.

OpenFlow is a technique for setting up flow tables in switches. It defines a protocol by which an external entity can control a flow table (i.e. the forwarding table that exists in the hardware of most modern switches). This allows the external entity to control the switching of packets on a per-flow basis.

From a virtualisation point of view, OpenFlow can be seen as a slicing method. It is a more general and flexible way of slicing compared to current methods (e.g. applied in FEDERICA), since it is not confined to slicing by VLAN, MPLS tunnel, VPN, etc. Since OpenFlow allows slicing on a per-flow basis, OpenFlow virtual networks could be defined in terms of groups of flows.

Another feature of OpenFlow is a control and management model where switches are controlled by an external entity over a secure channel. This separation of control and data planes gives a large degree of freedom in how switches are controlled. This offers interesting potentials of user-defined control policies and algorithms.

Two main research issues concerning flow-based virtualisation architecture are addressed in this section:

- Virtualisation based on flows. Investigation has been done on how virtualisation architectures can be designed based on OpenFlow, as a virtualisation layer that can support different virtualisation paradigms. The main purpose here is to investigate novel flow-based virtualisation techniques using OpenFlow.
- Centralized control plane architectures. OpenFlow uses a centralized controller to set up flows in the network. An investigation how to design this control plane to achieve performance, scalability, and robustness has been done

The novel flow-based virtualisation architecture approach (studied by KTH) is based on OpenFlow. An architectural design has been proposed, where a virtual network layer is built on top of an OpenFlow controller, as illustrated in Figure 6.

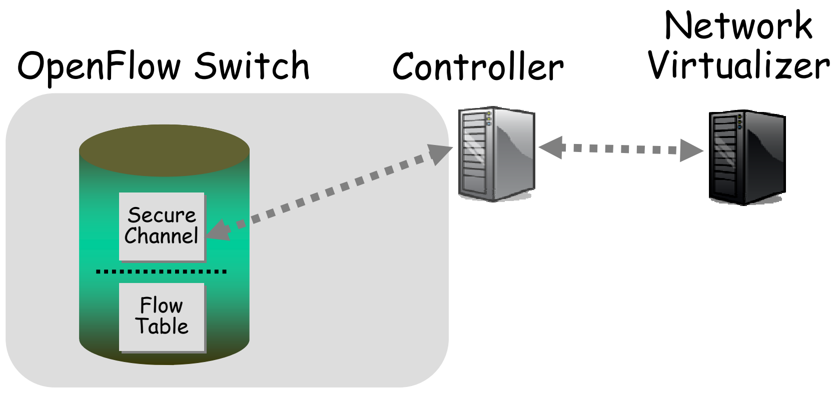

Figure 6 Virtualisation architecture based on OpenFlow

In this design approach, a network virtualiser that translates between flows and virtual networks is defined. The network virtualiser can be seen as a tool for provisioning and managing OpenFlow virtual networks. A virtual network (VN) is then defined in terms of set of flows and functions, and includes the actual flows belonging to the virtual network, routing functions to be used in the virtual network, policy functions for the virtual network, and so on. The design is illustrated in Figure 7.

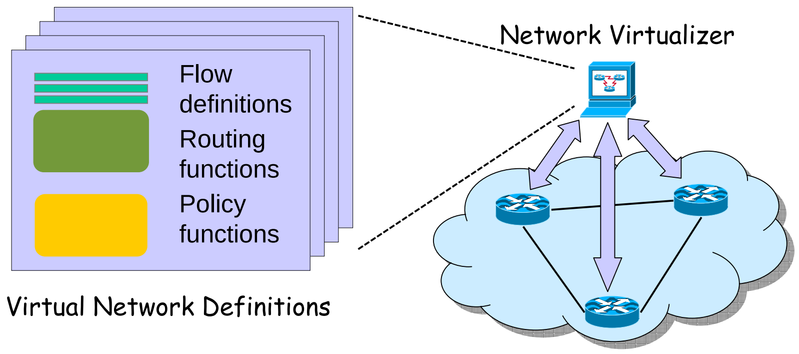

Figure 7 Defining and managing OpenFlow virtual networks

The network virtualiser sets up flow table entries according to specified routing and policy functions. To set up the flow table entries, the network virtualiser translates the VN definition into OpenFlow commands. These commands are transferred to the OpenFlow controller, which runs the OpenFlow protocol to set up the required flow table entries in the network switches. The VN establishment can be done either in a traffic-driven or in a pre-configured mode (or in a combination).

In the proposed OpenFlow-based virtualisation architecture, the control functionality is centralized, so that a controller can be responsible for many OpenFlow switches (see Figure 8).

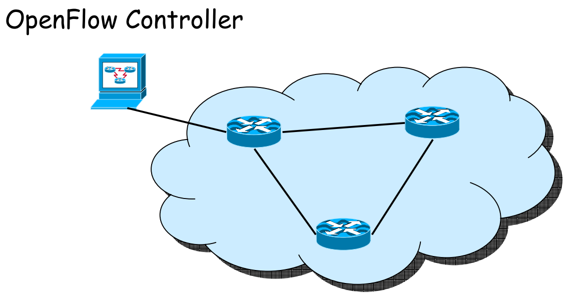

Figure 8 Centralized control in OpenFlow

The centralisation of control functionality presents several opportunities, as well as several challenges. The separation between control and forwarding means that communication overhead between controller and switches will be introduced. Furthermore, the OpenFlow controller tends to be a hot spot for control traffic. Finally, the centralized controller constitutes a single point of failure.

An important purpose with the flow-based virtualisation architecture is to make it dynamic and flexible. As a consequence, it should be possible to set up flows on demand, something which will result in per-flow interactions between controller and switches. This means that the network performance will depend on several issues related to centralized control. Therefore, we need to explore basic issues, such as:

- Controller performance,
- Network topology,
- Controller placement,
- Communication overhead,
- Flow table setup strategy.

In a centralized control plane, issues such as scalability and availability need to be considered. There are several dimensions to scalability. The centralized control plane needs to be scalable both in performance and the number of OpenFlow switches for which a controller is responsible. In addition, scalability in terms of geographical size, may require special attention due to the corresponding increase in communication overhead (and latency). An obvious way to deal with these issues is to use multiple controllers so that they can share the burden on the control plane. Such an approach will also help address the single point of failure in a centralized architecture, since more than one controller will be able to take care of the control plane traffic.

Multiple controllers in the centralized control plane present several interesting research issues, such as:

- Supporting multiple controllers per switch,
- Load balancing between controllers,
- Network partitioning among controllers,
- Geographical placement of multiple controllers.

The investigation of OpenFlow-based virtualisation architecture is performed in two steps. In the first step, the centralised OpenFlow controller issues have been studied by KTH, based on simulations in their laboratory. In the second step, a FEDERICA slice will be requested to evaluate different architectural scenarios.

A basic OpenFlow simulator has been implemented using Omnet++4.0 [Omn]. The OpenFlow simulator has been verified through the simulation of fundamental network topologies with a rather basic modelling of controller properties. It is possible to investigate performance, the number of flows established per second with regards to controller performance, network topology, and controller placement by using the current simulator.

Initial simulations have been made for a simplistic bus network with N switches and one controller, where the total flow establishment time is investigated as a function of the flow rate (load). When the first packet of a flow enters the first switch, a request is sent to the controller to configure the complete path from source to destination for this flow. It is referred to as *full-path* flow establishment. Such simulations have been made for various network sizes (in terms of number of switches on the bus). The results are shown in Figure 9.

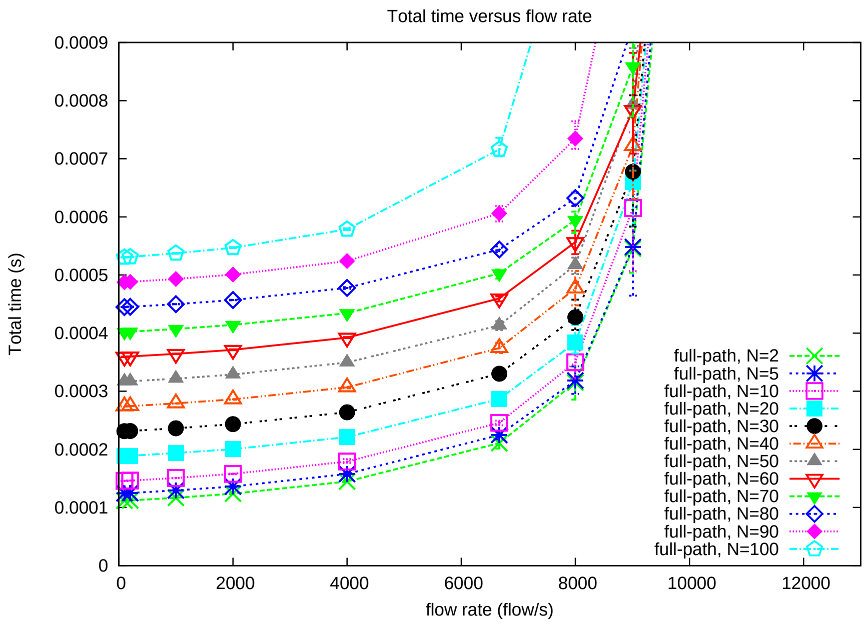

Figure 9 Flow establishment time vs. flow rate for various sizes of a simplistic bus network

An alternative to full-path flow establishment would be to use *hop-by-hop* flow establishment, i.e. that the controller configures only the switch where the request came from. Each hop will then be configured as the first packet of the flow advances along the path from source to destination. Hop-by-hop flow establishment will be investigated in the continuing simulation studies. In the preliminary simulations, we have tried to estimate when the controller itself or the control channel becomes saturated. The current controller model shows that 10000 requests per second can be served. The controller traffic amount required to serve a request, i.e. to configure the switches along the path, depends on the number of switches to be configured. Therefore, depending on the size of the network, either the controller itself or the control channel will get saturated. This is illustrated in Figure 10.

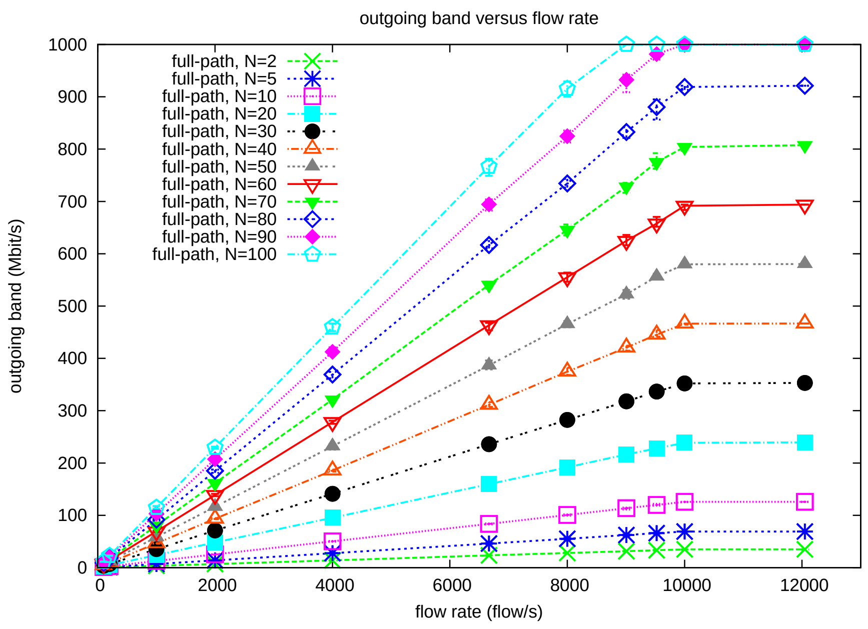

Figure 10 Control channel traffic vs. flow rate for various bus network sizes

The investigation on more advanced network topologies and various ways to use multiple controllers in the network (using a FEDERICA slice) is still on-going.

In addition, KTH is working on improving the controllers' simulation model to achieve a more realistic modelling of controller properties. KTH has defined two different scenarios involving multiple controllers:

- Multiple controllers with load balancing,
- Multiple controllers with network partitioning.

In both of these scenarios, the location of the controllers are important to minimize delays and to maximize performance.

The load balancing scenario is based on multiple controllers, aware of the complete network but located in different parts of the network. All controllers communicate with all switches, and the control plane load is shared between the controllers according to some balancing scheme. This scenario is illustrated in Figure 11.

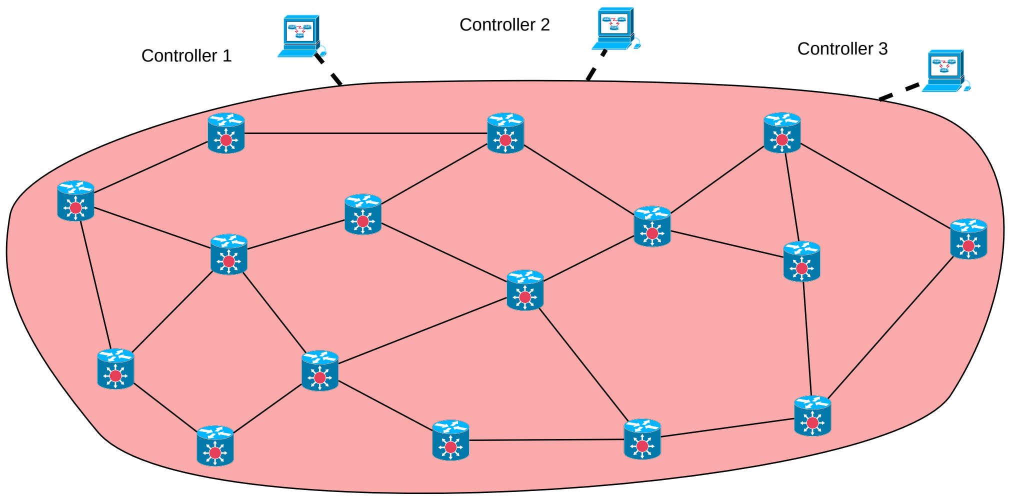

Figure 11 OpenFlow network with multiple controllers

The network partitioning scenario maintains the one-to-one association between switch and controller, so that each controller is responsible for one part (area) of the network. We will study how to best partition the network according to objectives like minimizing average switch-node latency and like achieving fair load sharing among controllers. The scenario is depicted in Figure 12.

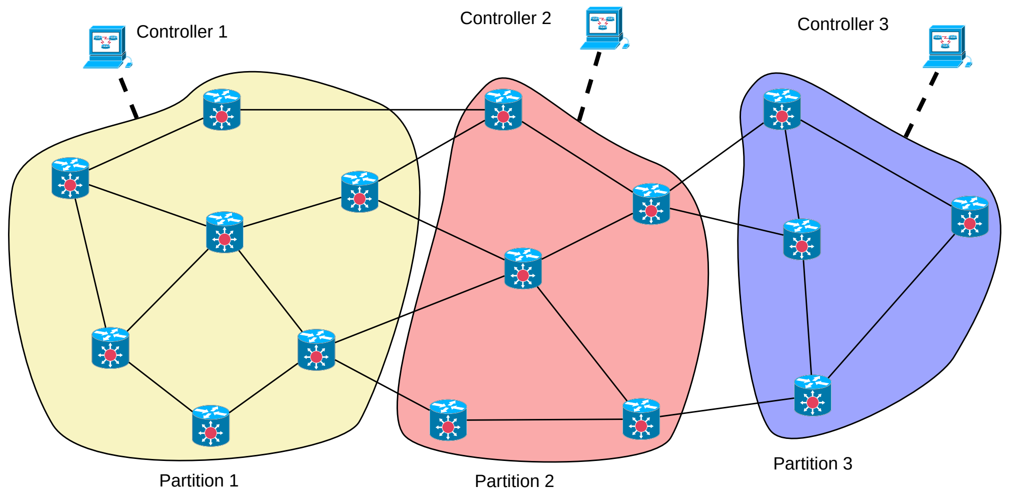

Figure 12 OpenFlow network, partitioned in areas with one controller per area

Regarding the network topologies used for the scenarios above, we considered different alternatives, such as random networks (e.g. the Erdős-Rényi model [Erm]), random location-based networks (e.g. the Waxman model [Wax]), and Internet-like networks (e.g. BRITE [Bri]). The reasonable sizes are up to 200-300 nodes in the simulations. In case of the FEDERICA slice request, the focus will be more on the benefits of the close-to-real network scenario (i.e. real latencies, bit errors, network failure injections, etc.) and not on the scalability.

The final results of the aforementioned research study will briefly be reported in the deliverable DNA2.3 (as an internal user project) at the end of the FEDERICA project.

The research on OpenFlow-based virtualisation architecture is highly innovative, since it breaks the current virtualisation paradigm applied in FEDERICA (and other virtualisation capable infrastructures today). The network slices can be defined by various attributes of the traffic flows, which lead to a much more flexible virtualisation concept. However, the FEDERICA infrastructure is adaptive enough to accommodate such state-of-the-art research studies inside a slice. FEDERICA is capable of virtualising both its control plane and data plane, so that the OpenFlow data plane and the proposed separated controllers can easily be implemented inside a FEDERICA slice. The novel OpenFlow-based slicing method will be eventually tested inside a ‘traditionally’ sliced network. This study can validate the unique feature of the FEDERICA infrastructure as it currently exits.

3 Authentication and Authorization Infrastructure for FEDERICA

The novelty of the FEDERICA concept consists in integrating infrastructure resources, user-accessible open source routers, and nodes within one consistent framework. This is due to the: multi-institutional, multi-vendor nature of the FEDERICA infrastructure domain, the multi-domain aspects brought up by the proposed interoperability prototype, and security (related to both internal risks and external threats). All of these aspects must be analysed and appropriate solutions must be worked out within FEDERICA.

This chapter provides a description of the technologies and solutions that have been chosen for research inside the JRA2 activity. It must be noted that research in Authentication and Authorization Infrastructure (AAI) functionalities in FEDERICA is constrained by the mechanisms used by end users to access FEDERICA resources. Specifically:

- SSH is the currently applied interface and it will continue to be in the short term,.
- The project envisions a web interface (or client application) in the later phase, which guides the activity to focus closely on web and web-service based solutions.

For better understanding of the current outcomes of the research, this chapter begins with some general considerations about AAI requirements in FEDERICA.

3.1 AAI requirements

Since the beginning of the project, the initial requirements for research in AAI have not changed dramatically (*see Section 4 of deliverable DJRA2.1*). They have been only refined and prioritized according to the feedback received from project administrators and end users.

Generally, any AAI solution proposed for the project should be based on the following principles (*Section 3.3 of this deliverable presents specific requirements for Web Services based solutions*):

- Simplicity: it should be easy to deploy, maintain and also enhance the solution in the infrastructure. From a user's perspective, AAI should work seamlessly with the interfaces and resources provided to them.
- Federation: ad-hoc solutions should be the last resort from any perspective. Federations exist, and most potential users are already part of them. They are important in building circles of trust between existing infrastructures, and of great importance is offering features like SSO (Single Sign-On), and unified login, etc.

- Scalability: FEDERICA is bound to grow in users and interfaces in the future. Making the correct decisions at the outset can lead to easier upgrades in the future. As an example, the usage of well-known standards is fundamental.
- Homogeneity: all the user information, whether it is an end-user or an internal member of the project, is located in the same data backend; any proposed solution should rely on this backend for authentication and user information retrieval.

3.2 SSH approach

3.2.1 Federated SSH (fedSSH)

fedSSH [Fsh] is a solution based on OpenSSH, which obtains user public keys from a directory in order to integrate SSH servers within a federated infrastructure. An institution, even if it does not intend to share access with others members of the federation, can take advantage of this solution, thus making the management of the policy access to their nodes easier.

This can be achieved by means of a policy dependent on user attributes rather than on user identifiers. The architecture of fedSSH consists of two main parts:

- The mechanisms to retrieve user public keys stored previously in a directory.
- A web application, managing who is allowed to store his/her public keys into the directory.

fedSSH offers two choices to the first component:

- A small patch (10 Kb) for OpenSSH, allowing it to connect to a Lightweight Directory Access Protocol (LDAP) server and retrieve user public keys.
- A set of scripts and REST-style (Representational State Transfer-style [Res]) Web Services that allow OpenSSH to update its public keys repository from a LDAP server.

Thanks to the web application, users are able to upload their public keys into the directory. This action only can be performed when they and their attributes comply with the authorization policy. This policy comprises of:

- SSH servers deployed in the federation or the institution itself.
- How those SSH servers are going to get users' public keys.
- In each server, a set of rules on the user attributes that must be satisfied in order to allow a user to upload public keys.

A set of nodes can also be joined as a group, making rule management easier. More information is available in [Fsh].

3.2.1 FUSE extension

FUSE (Filesystem in UserSpacE) is a technology [Fus] allowing the development of file systems residing in user space instead of kernel space that can provide file view abstraction over services. CurlFtpFS [Cur] is an example of such a file system which enables access to remote FTP resources through local directory tree. The main features of FUSE include:

- Simple to implement new file system,
- Secure, FUSE process, executed as an unprivileged user,
- Available for different operating systems (Linux, Windows, FreeBSD, etc.).

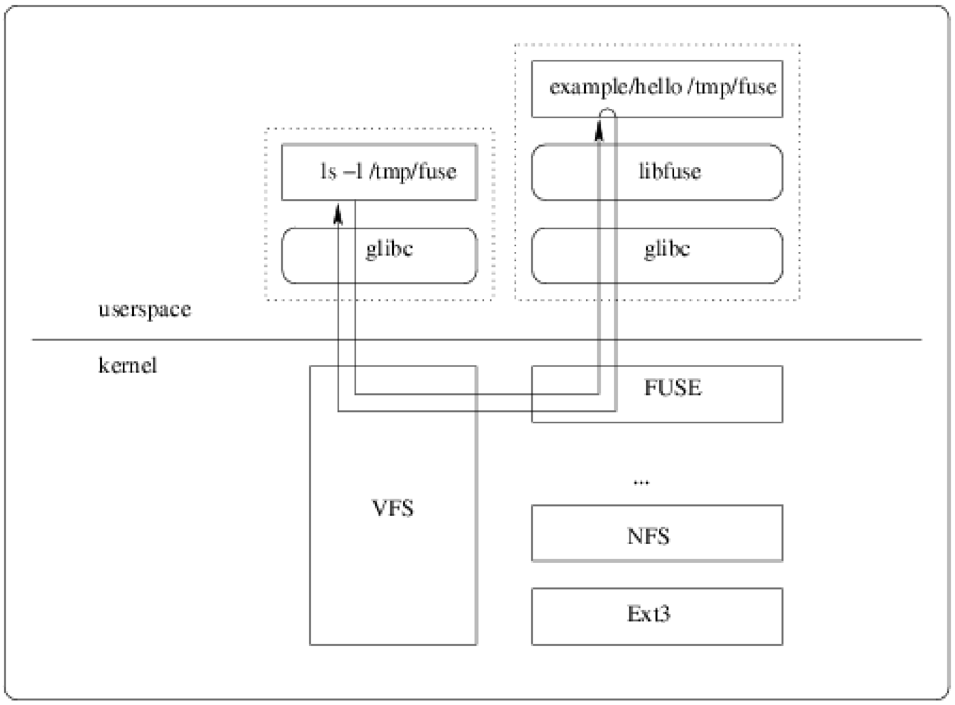

Figure 13 General FUSE architecture

As an alternative to the fedSSH approach, the FUSE file system was developed to provide files with authorized public keys of FEDERICA users. Such solution is also based on user attributes (public key in this case) rather than user credentials. Implementation consists of four file operations:

- *Open, Stat* – retrieves user's authorized key files from configured identity backend and caches it in memory.
- *Read* – read authorized key files.
- *Release* – remove user's file from cache.
- For security reasons, there is no implementation for *Readdir*, therefore, a user's list cannot be obtained.

In order to enable federated access and distinguish users from different organizations a naming convention for usernames was assumed in the form of *<username>.<organization alias>* where:

- *<username>* is the name by which user is represented in his/her organization identity backend
- *<organization alias>* is a short name for configuration entry for his/her organization identity backend (will be described later)

Considering the following example of user *john.doe* both in PSNC and RedIRIS identity databases their authorized key files would be accessed through */authorized_keys/john.doe.psnc* and */authorized_keys/john.doe.rediris* respectively, assuming that the FUSE file system is mounted under */authorized_keys* and both *psnc* and *rediris* are valid aliases. To enable OpenSSH to access these authorized keys, configuration option *AuthorizedKeyFile* in *sshd_config* file must be changed to */authorized_keys/%u.*

One of the key concepts of an authorized keys file system is a *mapping file* that contains configuration for federated organizations. Path to the file is passed as a mounting option. File syntax consists of lines in a form of *<alias name>=<backend address>* where:

- *<alias name>* is a short name for representing an organization, used in username naming convention
- *<backend address>* is identity backend configuration which holds authorized keys for given organization

At this point, the only available backend is *SSH Key Service (SKS)* with web-service access available in the *smoa-identity* package (will be described later) which obtains public key attribute for users present in IdP service. An optimal solution would be to utilize *SAML Attribute Query* from *SAML Assertion Query* profile to maintain maximum interoperability by using a well-known standard. Introducing many types of backends and extending mapping file syntax could also be considered.

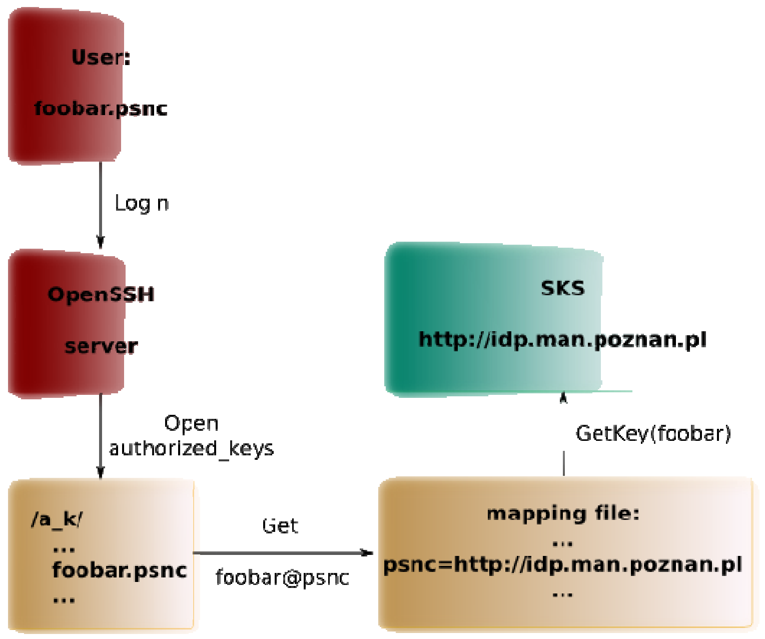

Figure 14 SSH Key Service (SKS) with Web Service access

3.3 Web services approach

In recent years, SOAs (Service Oriented Architectures) [Soa] and the usage of Web Services has increased rapidly, gaining in maturity, stability, and user knowledge. In this respect, its application in FEDERICA could be translated into a dedicated control plane (web-based, or client application) for the infrastructure's end user, where he/she could see and manage the available resources (monitoring information, granted computing, network resources, etc). The interaction between the control pane and the resource end points using Web Services would offer a homogeneous view to the end user without direct interaction, making it a secure environment while improving the user experience.

Nevertheless, some particular security requirements must be imposed on web based solutions so as to be applicable in the FEDERICA ecosystem. These are as follows:

- All requests and responses must include the unique identifiers of the elements issuing them.
- The syntax and semantics of the identity statements should be agreed offline between requesters and responders.
- The supporting message platform should implement the proper mechanisms to ensure integrity, confidentiality, authenticity and non-repudiation.
- It is recommended that encrypted channels to exchange security statements are used.

3.3.1 General architecture

Identity-based security in a Web Services environment requires an identity infrastructure, where a requestor (service consumer) can interact with the different resources (service producers) in a secure way.

From the producer's perspective, it is important to distinguish who has to be identified properly; namely, the requestor itself, and also the user behind it. This leads to two generic models:

- In the first case, the end user is identified at a client application which acts on behalf of the user requesting services from the appropriate service producers. Here a single statement can be used to properly identify the client application and the end user. Figure 15 shows an example workflow, where the user is authenticated at the Identity Provider (IdP). The client application at the host queries directly each resource including the user's identity data (expressed in SAML, and carried through SOAP). The AS is an Authorization Service, where the request can be properly validated (possibly, contacting Attribute Authorities for further information about the requestor).

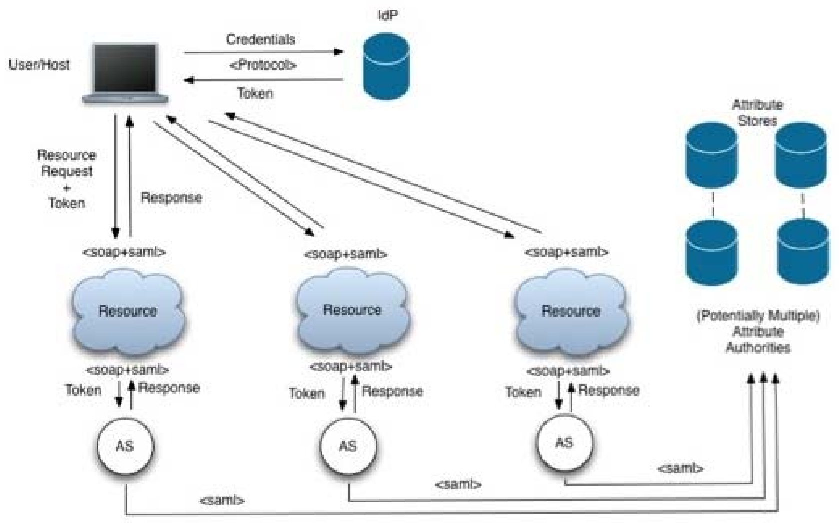

Figure 15 Example workflow for requesting services from appropriate service producers in parallel

- In the second case, the end user is identified at a client application, which acts on behalf of the user requesting services to the first service. This service then forwards the request to the second one, in a chained form. To successfully identify the end user at each service, the initial security statement about the user must be forwarded from service to service. Each link in the chain should also add information about its identity before forwarding the request. An example using SAML, SOAP and AS (as in the previous example) is shown in Figure 16.

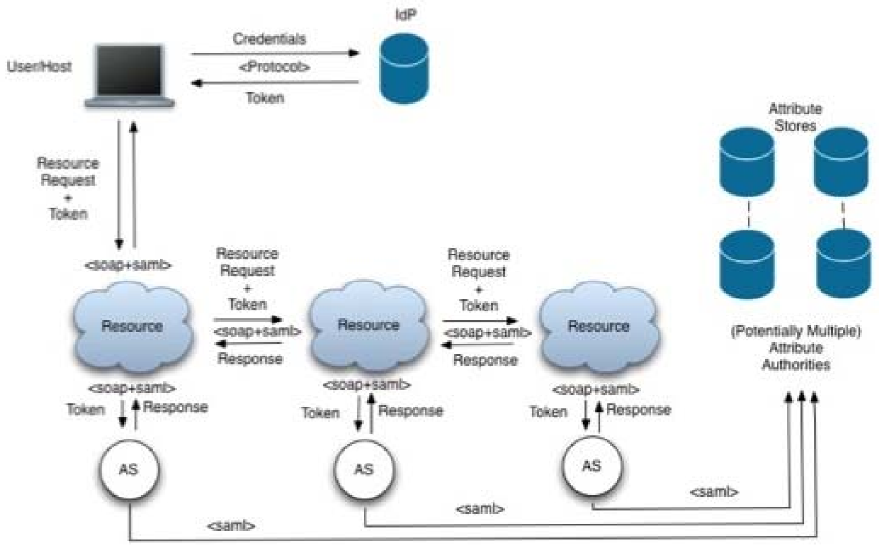

Figure 16 Example workflow for requesting services from service providers one by one as a chain

Two different mechanisms for authenticating users have been considered:

- Direct use of Liberty Alliance protocols [Lib,
- Application of the eduGAIN AA [Edu] infrastructure.

eduGAIN also proposes a mechanism for using the authentication data inside the Web Services infrastructure.

3.3.2 Applying Liberty Alliance protocols

The Liberty Alliance [Lia] was formed to build open standard-based specifications for federated identity and identity-based Web Services. All the specifications are built around SAML 2.0 which is now the industry standard for deploying and managing open identity-based applications. This section will describe the *smoa-identity* package which is the implementation of Liberty Alliance specifications developed at PSNC.

Since the Liberty Alliance is mainly focused on Web Service access, it has a great potential in available features in this area. The *smoa-identity* package consists of modules being implemented in the following Liberty’s specifications:

- Liberty ID-WSF Single Sign On Service – identity provider component issuing SAML 2.0 assertions based on provided authentication tokens. It makes use of SAML 2.0 Authentication Protocol.
- Liberty ID-WSF Authentication Service – similarly to SSO it issues assertions but authentication process relies on SASL protocol (SASL over Web Service).
- Liberty ID-WSF Discovery Service – integrated with identity provider and acts as both a service discovery and assertion issuer. Service Providers register their metadata in Discovery Service and associates users with them based on

their identities. Service Consumers query Discovery Service based on their identities and receive Service Providers' metadata with SAML 2.0 assertions attached.

The main Web Service scenario using Liberty services (see Figure 17) looks as follows:

1. Authenticate either via Single Sign On Service or Authentication Service.
2. On successful authentication an assertion is issued with AudienceRestriction condition set to address of Discovery Service.
3. Using the assertion as an authentication token query Discovery for required Service Provider type.
4. A successful query metadata (or set of metadata) is returned which holds assertion for requested Service Provider (note that metadata structure is used to hold assertion, assertion itself is not used to carry any information about target service).

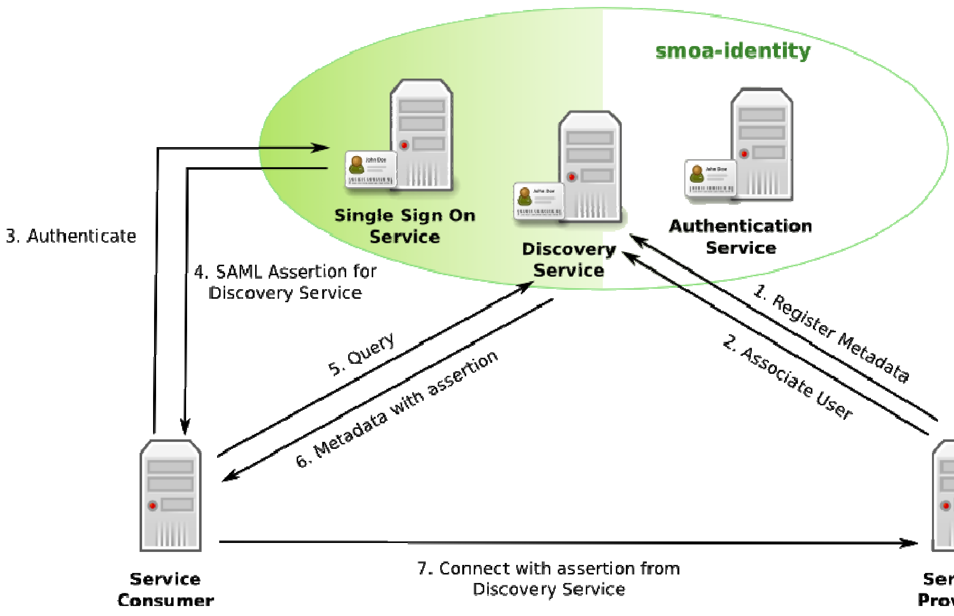

Figure 17 Web service scenario using Liberty services

An example of response from Discovery Service is presented in the Appendix I of this deliverable.

Though Web Service access to FEDERICA infrastructure is an interesting and important research area, a traditional access through web application is still a basic method for users to access FEDERICA resources. Due to the Web Service centric nature of Liberty Alliance specifications, an additional component was developed acting as a web frontend to the *smoa-identity* service. In order to explore methods for creating web applications in federated environment, we tested a development version of the *spring-security* component (which will soon be incorporated into regular spring-security distribution) implementing WebSSO stack with use of SAML 2.0. The key features of the *spring-security* include:

- It can be used as a security layer for any Java web application framework, while being independent from the rest of the Spring project.
- Wide use of well established standards regarding SAML 2.0.
- Very flexible configuration.

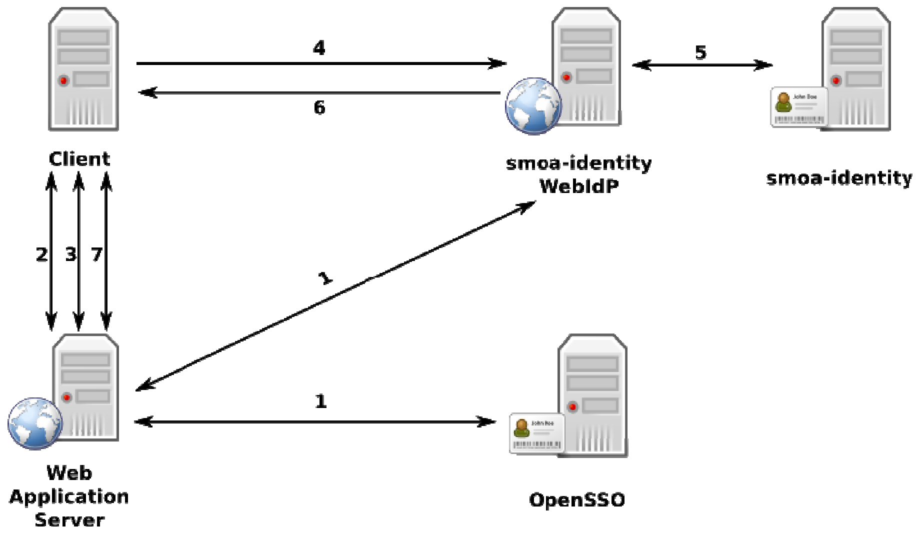

Figure 18 Simple scenario testing the framework

The scenario used to test the framework (see Figure 18) is as follows:

1. During initialization, the *spring-security* framework retrieves SAML 2.0 Identity Provider Metadata from the configured IdPs.
2. The client requests access to web resource on the Web Application Server and is redirected to the IdP selection page.
3. The client selects its IdP and is redirected according to SAML 2.0 HTTP POST or Redirect Binding.
4. The client provides its credentials to its local IdP.
5. The client is authenticated via the SAML 2.0 Protocol in the Single Sign On Service and the SAML 2.0 Assertion is issued.
6. The client is redirected back to the Web Application Server according to the AssertionConsumerServiceURL with the attached SAMLResponse.
7. The client accesses the web resource.

The proof of concept deployment used to test the *spring-security-saml* has shown:

- The project's maturity, despite being in the development phase
- Interoperability with different identity providers, including *smoa-identity* and OpenSSO.

- The ease of creating web applications with federated access.

3.3.3 Applying eduGAIN

One of the outcomes of the GÉANT2 project is eduGAIN [Edu], a tool that enables the sharing of identity data between different federations, creating a new scenario composed of existing organizations and policies by means of an interoperability layer. This scenario is usually referred to as a ‘confederation’.

In the current version of the GÉANT project (GN3), eduGAIN is being established as a service, mainly based on SAML2 WebSSO profiles and using direct connections between the Identity Providers and Service Providers participating in the confederation. It is important to note that the eduGAIN team has selected FEDERICA as one of the use cases to study, therefore supporting also non-web profiles.

The authentication based on eduGAIN WE (Web Enabled) profile is applicable in those cases where a certain software component (the client) is accessed by end users through a web container (e.g. an application server), and the client acts on behalf of the end user when requesting services to other component(s), that will be referred to as ‘resources’ in the rest of this profile. To access the client, users must pass through the procedures applicable to WebSSO, so the container can provide the client with the attributes received during the WebSSO phase.

In this case, user authentication is performed by means of the same web browser used to access the client. After the WebSSO steps, the client is able to send a proof of a user’s identity, as asserted by the user’s local IdP and with the appropriate restrictions to avoid abuse. In summary, the profile provides a method for performing secure identity delegation through WebSSO.

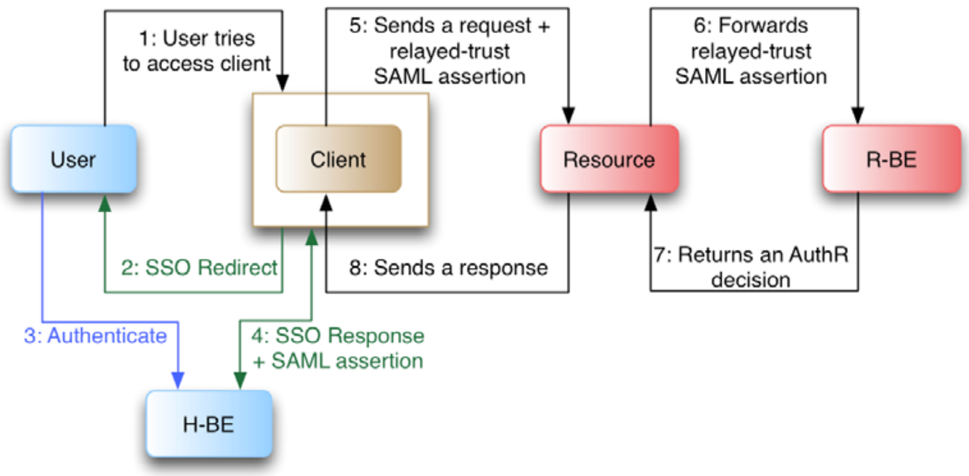
1: User tries to access client
5: Sends a request + relayed-trust SAML assertion
6: Forwards relayed-trust SAML assertion
User
Client
Resource
R-BE
2: SSO Redirect
8: Sends a response
7: Returns an AuthR decision
3: Authenticate
4: SSO Response + SAML assertion
H-BE

Figure 19 Authentication based on eduGAIN WE profile

The client container (through the appropriate federation/eduGAIN mechanisms) redirects the user browser to the appropriate IdP (H-BE in the figure) for authentication. In doing so, it uses the eduGAIN profile for WebSSO.

Applying the local procedures at the home federation, the user authenticates exchanging credentials at their local authentication point. The H-BE sends back to the client container an identity assertion. The container uses whatever local procedure to pass the received data to the client, preserving the original SAML assertions received from the H-BE as part of the SSO response.

The client must use the received SAML assertion to build a relayed-trust SAML assertion (*according to the rules described in the Appendix I*). It must include the identity material to be used in the request sent to the resource(s). The resource(s) receiving this assertion will then pass it for evaluation to their corresponding Authorization Service (R-BE in the figure). The concrete mechanisms for the passing of the relayed-trust SAML assertion from the client to the resource(s) and from there to the R-BE are out of the scope of this profile.

The details of the SAML construct, used in this case, are detailed in Appendix II of this deliverable.

The work on AAI has satisfied the current requirements of the project by following the principles of simplicity, scalability, and homogenization, bearing in mind the security of the infrastructure and the usage of standard technologies. Moreover, it has anticipated the future needs in this area by keeping an eye on the latest trends and what the FEDERICA partners envision for the future of the project.

In order to achieve this, JRA2 has developed software (i.e. improvements to fedSSH and the implementation of Liberty Alliance protocols), tested them in pilot infrastructures, and liaised with other initiatives (such as eduGAIN in the GEANT project).

4 Simulation framework for FEDERICA

In this chapter, the basic motivations and the implementation details of the simulation framework tailored to FEDERICA are discussed. This is additional work (*not part of the original project Description of Work*) which the JRA2 partners have agreed upon based on the progress of the project. The initial need for this simulator was mainly rooted in the fact that the actual usage of the FEDERICA infrastructure was not sufficient at that time (in 2009) to provide enough inputs to perform the detailed fairness, resource allocation, and business-related studies by the Joint Research Activities. That is why it was decided to develop the simulation framework but in the meantime other useful aspects have emerged. Finally, it can be said that the simulator supports:

- Planning and development of the FEDERICA physical infrastructure (potentially useful for planning any follow-on project of FEDERICA).
- Planning of the slice creation process and its effect on the network performance (useful for NOC and daily operation).
- Better understanding of the users' behaviour in a virtualised environment (useful for business-related research and potential commercialization later on).

This chapter contains a brief description of the FEDERICA simulation framework as well as an additional theoretical study on the fairness limitations in FEDERICA.

Note that the simulator source code and executable files are available to all the FEDERICA project partners on the FEDERICA project Wiki page.

4.1 Implications of current FEDERICA usage

The task TJRA2.2 activity in FEDERICA is looking towards the future and searches for novel concepts for fair sharing, security, accounting and business models for virtualised infrastructures in general. This type of research is routed in the experiments with the current FEDERICA virtualised infrastructure but obviously looking beyond that in terms of broad penetration, wide usage, and commercial exploitation of the entire concept. Taking into account the actual FEDERICA infrastructure and the behaviour of its actual users such type of research on future scenarios cannot be done efficiently.

Moreover, the current usage of the FEDERICA infrastructure is far from sufficient to study specific, taut situations with high competition between user requests, starvation of resources, different service level specifications, fairness of the various resource sharing and allocation mechanisms, etc. The actual usage of the FEDERICA infrastructure is depicted well on the monitoring statistic figure visualized by CESNET [Cmo]. In Figure 20, the FEDERICA virtual infrastructure has far more number of virtual resources than the current user projects can accommodate.

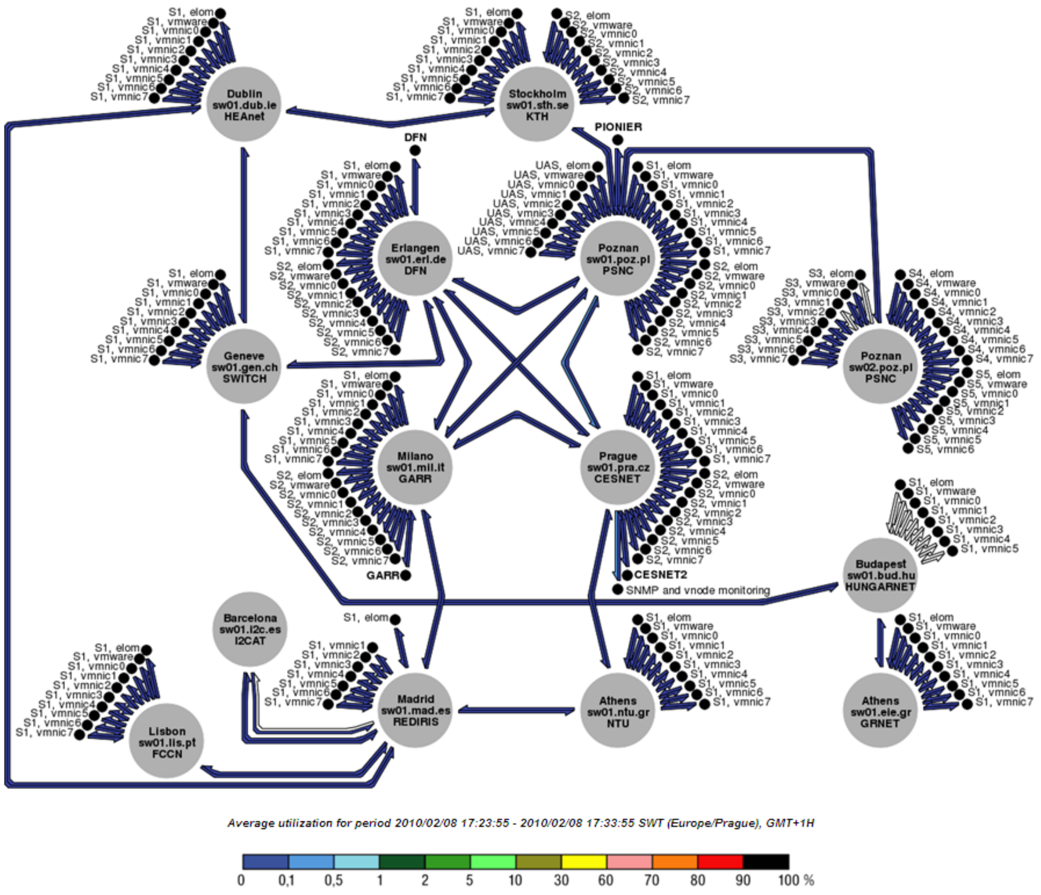
Dublin sw01.dub.ie HEAnet
Stockholm sw01.sth.se KTH
Erlangen sw01.erl.de DFN
Poznan sw01.poz.pl PSNC
Geneve sw01.gen.ch SWITCH
Poznan sw02.poz.pl PSNC
Milano sw01.mil.it GARR
Prague sw01.pra.cz CESNET
Budapest sw01.bud.hu HUNGARNET
Barcelona sw01.i2c.es I2CAT
Madrid sw01.mad.es REDIRIS
Athens sw01.ntu.gr NTU
Athens sw01.ele.gr GRNET
Lisbon sw01.lis.pt FCCN
DFN
PIONIER
GARR
CESNET2
SNMP and vnode monitoring
Average utilization for period 2010/02/08 17:23:55 - 2010/02/08 17:33:55 SWT (Europe/Prague), GMT+1H
0 0,1 0,5 1 2 5 10 30 60 70 80 90 100 %

Figure 20 FEDERICA infrastructure utilisation (as of 08-02-2010 [Cmo])

This over-provisioning of resources is, on one hand, highly beneficial from the Service Activities' point of view (to support as many users as possible) but is, on the other hand, a drawback from the Joint Research Activities' (in particular, some offered research tasks of JRA2) point of view.

This fact was realised by the JRA2 partners who decided to develop a simulation framework for FEDERICA. The simulation framework is flexible enough to take into account different physical infrastructures, various resource types, and a wide variety of user demands. It is also possible to simulate any kind of resource management algorithms and architectures under the framework.

During the development of the simulation environment and the on-going discussions with the Service Activities inside the project, the FEDERICA simulator was found to be essential for performing the JRA2 research studies and very useful in understanding the potential development of the infrastructure and the future directions of strategic decisions. An example is the effect of the new physical locations and resources installed in the infrastructure on the performance and/or throughput of the virtualised systems allocated to the users.

4.2 Simulation framework for FEDERICA

A well-known practice in the commercial world is using specific simulations tailored to the given infrastructure (and broader environment) to study real life situations in production networks. In this section, the FEDERICA simulation environment is described.

GSSIM (Grid Scheduling Simulator) [Gss] is software for performing simulations which provides a comprehensive environment enabling researchers to test resource management algorithms and architectures. For the purposes of the FEDERICA project, GSSIM was adapted to test different approaches towards resource allocation based on a given demand model. Emphasis was put on simulating an exact infrastructure, as close to the problem as possible.

Note that the following description is restricted to the simulation framework only which tries to cope with the real FEDERICA architectural component (defined by JRA1 in deliverable DJRA1.1) and in parallel tries to be flexible enough to implement other (future) components as well.

4.2.1 Description model

The first step to enable simulations of the FEDERICA infrastructure is to provide input data for the framework, consisting of slice requests with a specific probabilistic distribution which tries to match to a real usage scenario. Since there is no official existing slice description model, it was necessary to create a language which would allow expressing users' requirements for resources and topology.

4.2.2 Identifying FEDERICA resources

Simulation framework for FEDERICA infrastructure provides three groups of resources:

- Computing resources (virtual nodes) – virtual machines hosted by virtualisation software characterized by the number of CPUs, network interfaces and amount of memory and disk space. The number of nodes running on a single server is limited by its hardware resources.
- Network resources (routers, software routers, switches) – routers and switches are instances of virtual routers in Juniper hardware. Software router is a dedicated virtual node containing network software with support for custom implementations of routing protocols.
- Network links – a way of communication between above resources with bandwidth limited to 1Gb/s (according to current FEDERICA infrastructure).

A deeper look into resource types shows no particular difference between a virtual node and a software router, or between a router and a switch. In all cases, the same hardware is used to host them (i.e. a virtualisation server for the virtual node and software router, and Juniper hardware for the router and switch). This leads to the conclusion that, without loss of generality, we can divide all the computing and network resources into two classes: *nodes* and *routers*. Furthermore, it was assumed that every virtualisation server and every Juniper router provides a constant number of maximum virtual instances they can host in a single point of time. This assumption implies identical parameters of virtual instances with physical resources equally divided among them.

Although the simulation allows the framework to support reservations for network bandwidth, this feature is not available in the FEDERICA infrastructure, due to the project's architecture details. Each physical network interface is shared by many virtual machines running in virtualisation software which means that the system lacks the mechanism to enforce bandwidth limits. This leads to a problem of measuring network performance in simulation environment. (Note that the performance issues are not taken into account in the case of the real FEDERICA infrastructure either.)

The only metric for expressing link quality available in a simulation framework is the number of reservations made. This leads to the conclusion that ways of expressing network performance must be re-visited and eventually incorporated into the simulation environment.

4.2.3 Slice description language

For the purposes of the simulation environment, a simple XML based language was defined for the slice description.

A slice description example can be found in Appendix III/a of this deliverable.

The description consists of three main parts:

- Slice general information is found in `<slice>` tag, including the unique slice id and slice reservation parameters – duration and time slot boundaries. Note that in this context, *slot period of time* means where reservation may occur, and it must be at least as long as the duration parameter or it may be even longer. .
- Resources description section found in the `<resources>` tag defines the computing and network resources used in the slice. Each resource is described by its type (router or node, as described in previous section) and unique id.
- Topology description section found in the `<topology>` tag defines the connections between resources. Each connection consists of two endpoints that refer to resource id attributes.

4.2.4 Infrastructure model

In order to provide a complete simulation environment for the FEDERICA infrastructure, a model of physical resources and topology should be included.

Organizations participating in the FEDERICA project can be divided into two groups: *Core PoPs* and *Non Core PoPs* each of them having the following hardware resources:

- *Core PoP* – 2 virtual node servers and 1 Juniper router,
- *Non Core PoP* – 1 virtual node server.

For each group, a template was created representing its parameters. It was then applied to every organization creating a complete map of available resources. The description is stored in *workload/HostParameters.xml*. Although it is an internal format of the GSSIM software, it has a simple syntax and can be easily extended in case of infrastructure changes.

An example for a PoP configuration can be found in Appendix III/b of this deliverable.

Similar to the resource description, network topology can be found in *workload/network-topology.txt* containing link descriptions with references to resources defined in previous file.

4.2.5 Running a simulation

The complete simulator source code, executable files and comprehensive manual are available to all the FEDERICA project partners on the FEDERICA project Wiki page

Simulation description file

Before running a simulation, a user must provide an experiment description in the '`simulation.properties`' file located in the project's root directory. It has a standard format of defining Java properties with following options:

- `federicaschedulername` – class name implementing scheduling algorithm,
- `slicesdirectory` – path to directory containing XML files with slice descriptions.

Executing simulation in console

To start a simulation from the console, execute '`run.sh`' script found in the project's root directory.

Executing simulation in Eclipse IDE

Before running a simulation in the *Eclipse* environment, first import the project. To do this: *Select* menu *File->Import,* then *General->Existing Projects into Workspace*, in *Select archive file* select file `FedericaScheduler.zip` and press *Finish*. To run the simulation, right-click on *FedericaScheduler.launch* and select *Run As->FedericaScheduler*.

4.2.6 Development support

In the FEDERICA simulation framework the basic data types are as follows:

- `SliceDescription`

  Represents slice requirements. All the information contained within XML file can be obtained here including reservation parameters, required resources and network topology.

- `SliceResource`

  Single resource defined in the slice description file characterized by its type (node or router) and list of connections to other SliceResources.

- `Location`

  Class representing single organization (such as PSNC, TERENA, CESNET, etc.) where slice resources can be hosted.

- `Connection`

  Depending on the context represents connection between two resources, in case of slice description, or two locations, in case of network topology of FEDERICA infrastructure.

- `Offer`

  Offer from particular virtual machine server or Juniper router (according to type of requested resource). `getResourceAllocations()` method gets list of time slots, corresponding to requested time constraints, within which reservations could be made.

- `ResourceAllocation`

  Returned either in `Offer` object or by `getLinkReservations()`. It represents slot in time beginning in `getStart()` and ending in `getEnd()`. Free and allocated slots of resource in this period could be obtained by invoking `getFreeSlots()` and `getAllocatedSlots()` respectively. For `Offer` objects resource slot represents available virtual instances, for network reservation it is number of reservations made.

- `ResourceManager`

  Basic class for managing reservations of physical resources. The `getOffers()` method obtains offers of possible reservations accordingly to given time constraints. The `acceptOffer()` chooses one of them making reservation of physical resource. If slice requirements cannot be met the `reject()` method is invoked. Physical network topology can be obtained in `getNetworkTopology()` which is used to choose the best path between two locations (according to own algorithm). For convenience a helper method is provided which finds the shortest path between two locations. Reservation of chosen path for period of time is made via `reserveNetworkPath()`. The number of reservation made on network link can be obtained by invoking `getLinkReservations() method.`

In order to implement custom scheduling algorithm into the framework please find the Federica Simple Scheduler implementation document on the FEDERICA Wiki.

4.2.7 Metrics and graphs

After running simulations both textual and graphical statistics can be found in *workload/stats_read* directory including:

- Resource utilization factor,

- Waiting times,
- Number of completed requests,
- Gantt charts with resource reservations.

4.3 Fair sharing of resources

As mentioned before, the current FEDERICA infrastructure has far more resources than the actual user projects can accommodate. According to the estimations (*done by the NA2 'Building and consolidating the user community' activity of the project*) the FEDERICA resources will remain over-provisioned compared to what the user projects will request. This means that FEDERICA does not have to cope with the lack of resources i.e. in situations where the fairness issues are most relevant.

However, FEDERICA should be prepared for such unexpected situations (e.g. in case of a dramatic growth in the number of user projects or appearances of really large projects requesting an increase in the number of resources at one time) or at the very least, FEDERICA should be aware of its theoretical limitations and boundaries in terms of utilization and throughput while ensuring the maximal fairness of the resource usage. That is the reason why simple theoretical models tailored to FEDERICA have been created and mathematically analysed. The results show the theoretical upper limits of the FEDERICA infrastructure that might not be reached in real life situations but shows the scalability of the infrastructure in terms of accommodated user projects. This information could also be useful for the User Policy Board of FEDERICA taking into account practical decisions on user applications (knowing only the required number of virtual resources and the expected length of those usage).

4.3.1 Simple Linear Programming (LP) models

Ensuring fairness is one of the most important aspects in FEDERICA serving a wide variety of user projects (ranging from EC funded large projects to individual academic use). Simple Linear Programming (LP) [Lpm] models have been created and analysed by JRA2 discovering some basic parameters. The initial assumptions have been taken into account are as follows.

The FEDERICA project lifetime is slotted into equal time slots. The resources can be assigned to the users at the beginning of the time slot and cannot be released until the end of the timeslot. The number of resources at each time slot is fixed. The total number of user demands entering the system during the simulation lifetime is also fixed. All demand has the same priority (no different Service Level Specifications) The fairness in this context is defined as the aim of fair serving of the user requests; i.e. allocating resources equally to both shorter, smaller project requests and longer bigger project requests as well as keeping the infrastructure throughput and utilization as high as possible. The simple formulization of the problem is depicted in Figure 21.

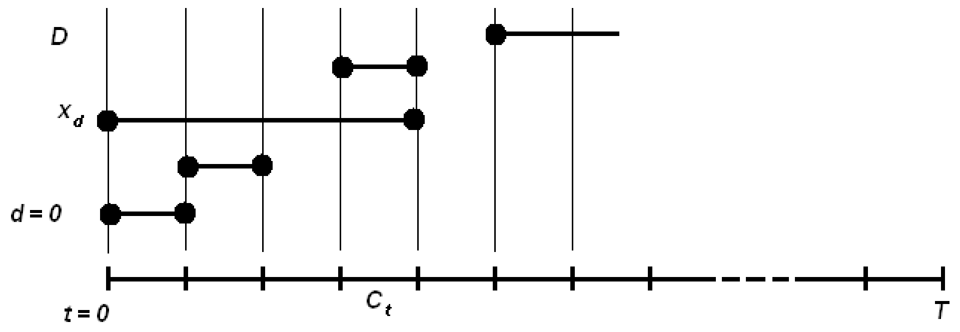

Figure 21 Problem formulization

To reduce the complexity of the problem (but still remain realistic), the following input parameters and constraints have been chosen for the analysis.

- The total number of user demands *D* during the project lifetime cannot exceed 40. The demands are given a priori.
- The length of a time slot *t* is 1 month and the simulation lifetime *T* is 30 month (just like of the FEDERICA project).
- The resource capacity at a time slot C_t is 160 (taking the number of available V-Nodes and virtual router/switch instances into account in case of guaranteed service to ensure the reproducibility in real FEDERICA). Note that the link bandwidth is neglected because it can be split infinitely, in theory.
- The user demands are elastic, that means that each demand can consume any resources at a given time slot (i.e. there is no SLA policy for reserved resources, any service guarantees, etc.)
- To be realistic, the statistical distribution of the latest OneLab nodes' CPU usage statistic [Ols] has been analyzed to determine the parameters of the FEDERICA user demands. It turned out that the holding time of the CPU usage in OneLab (i.e. CPU hours) follows an exponential distribution with the expected value of *1/λ*, as it is shown in Figure 22.

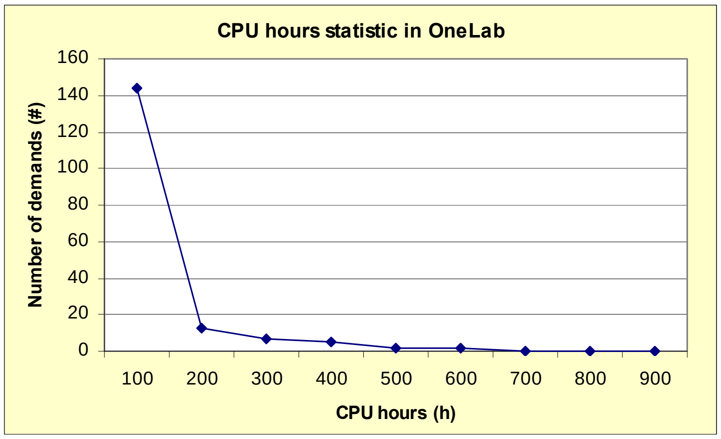

Figure 22 CPU usage statistic in OneLab nodes [Ols]

For the holding time of FEDERICA user demands the same exponential distribution has been chosen. For the demands stating time distribution a simple uniform distribution on *t∈(1..T)* has been chosen. (In case of FEDERICA, peek hours or any trends in regular usage cannot be recognized, so uniform distribution is the proper assumption for this purpose).

- The expected holding time $1/\lambda$ of a user demand is selected to 1, 2 and 3 months in different analyses.

To analyze the theoretical limits under the given constraints four different objective functions have been defined as a LP problem to be solved.

- The first objective function *max_thru* is to determine the theoretical upper limit of the FEDERICA infrastructure throughput (i.e. maximize the throughput under the given constraints). The throughput shows how many virtual resources can be used by the user projects in total.
- The second objective function *max_util* is to determine the theoretical upper limit of the FEDERICA infrastructure utilization (i.e. maximize the utilization under the given constraints). The utilization shows the percentage of time the virtual resources are used by the user projects in total.
- The third and the fourth objective functions are to ensure the fairness of the resource allocation i.e. make sure that the bigger, longer projects do not consume all the resources from the smaller, shorter projects or vice versa. Many smaller projects do not consume the majority of the resources forcing out the bigger, longer project demand requests.
    - The third objective function *log_fair* is a kind of revenue objective that consists of maximizing the sum of natural logarithms of the resource volumes obtained by user projects. The rationale behind using the logarithmic function is that it does not ‘prefer’ the short demand requests (revenue goes to infinite) and at the same time does not ‘prefer’ the high-demanding users either. Note that this is easy to solve but does not provide the optimal fair solution.
    - The fourth solution *opt_fair* is a small algorithm (called in the literature as Max-Min Fairness) that provides a fair solution that is optimal i.e. it is trying to maximize the overall throughput while keeping the minimum number of obtained resources on the globally maximum.

The LP formulations of the objective functions and algorithm as well as constrains can be found in Appendix IV of this deliverable.

4.3.2 Theoretical results

First, the basic parameters, such as the overall throughput, utilization, and acceptance rate of the user requests were calculated

The maximum throughput can be determined by applying the first objective function *max_thru*. This ensures that the user requests will get as many virtual resources in total as possible. This maximal number is taken as 100% throughput. In Figure 23, the first group of columns shows the resource utilization and the request acceptance rate in the case of 100% throughput. The virtual resource utilization is only 76.7% and 15.6% of user requests were rejected. The reason for this is that the throughput maximization prefers shorter demands in time and excludes some longer demands which lowers the resource utilization in total.

The next group of columns shows the theoretical maximum of the resource utilization using the second objective function *max_util*. This ensures that the highest possible number of resources is allocated to the user project in each time slot. The resource usage maximization at each time (88.2%) leads to a lower overall throughput (80.5%) and the highest rejection rate (28.3%) as is shown in Figure 23. This ensures the constant usage of the allocated virtual resources in the long term where the longer projects are preferred against the shorter ones, in time.

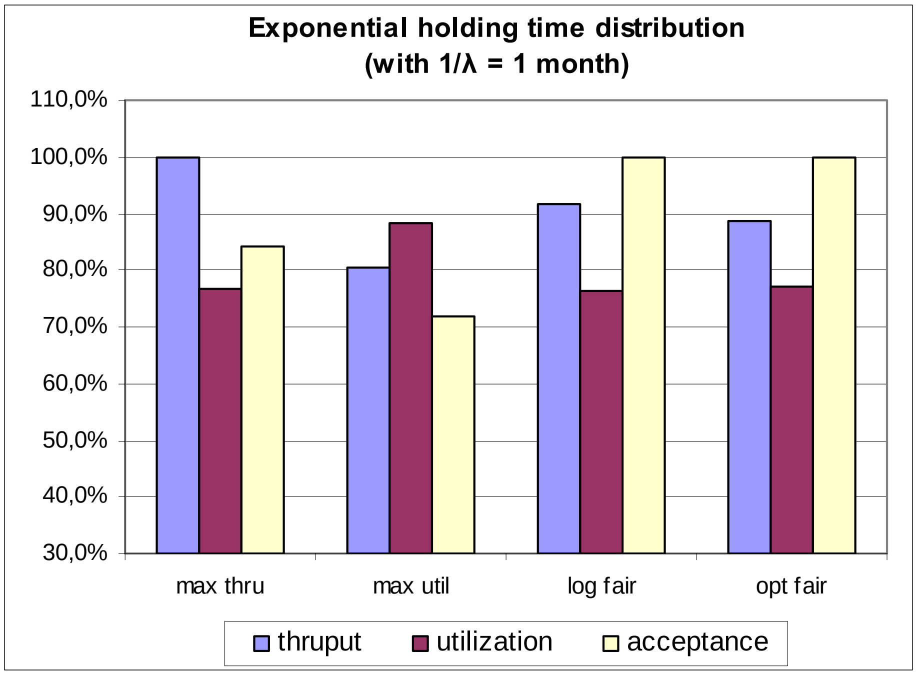

Figure 23 Performance parameters

The fairness strategies can now be applied, having satisfied the upper limits of these basic parameters (100% throughput and 88.2% utilization of virtual resources). The aim is to calculate the performance (i.e. throughput and utilization) parameters while keeping the resource allocation fair (i.e. no preference between longer and/or shorter projects). The *log_fair* objective offers a relatively fair solution. It can be seen in Figure 23 (third group of columns) that the acceptance rate of the user request are 100% i.e. neither shorter nor longer projects have been preferred. The infrastructure throughput is 91.8% which is very high, but the resource utilization is rather low (76.4%). Compared to the results of the *opt_fair* solution, a significant difference cannot be seen. Beside 100% acceptance rate and optimal fairness, the throughput is a bit lower (88.8%) but the utilization is a bit higher (77.3%). This demonstrates that the shorter and longer requests are taken equally.

In Figure 24, the expected number of allocated resources to the user demands is depicted as a function of the average demand holding time.

Two scenarios are shown: (a) if the FEDERICA policy (ensured by the User Policy Board) prefers longer projects which can maximise the utilization but the overall throughput and the acceptance rate will be lower; and (b) if the shorter projects are preferred, since these can maximize the throughput (however, resource utilization is poorer). In fairness to every project request, none of the shorter or longer projects can be rejected but the number of obtained resources by the users is lower in every case.

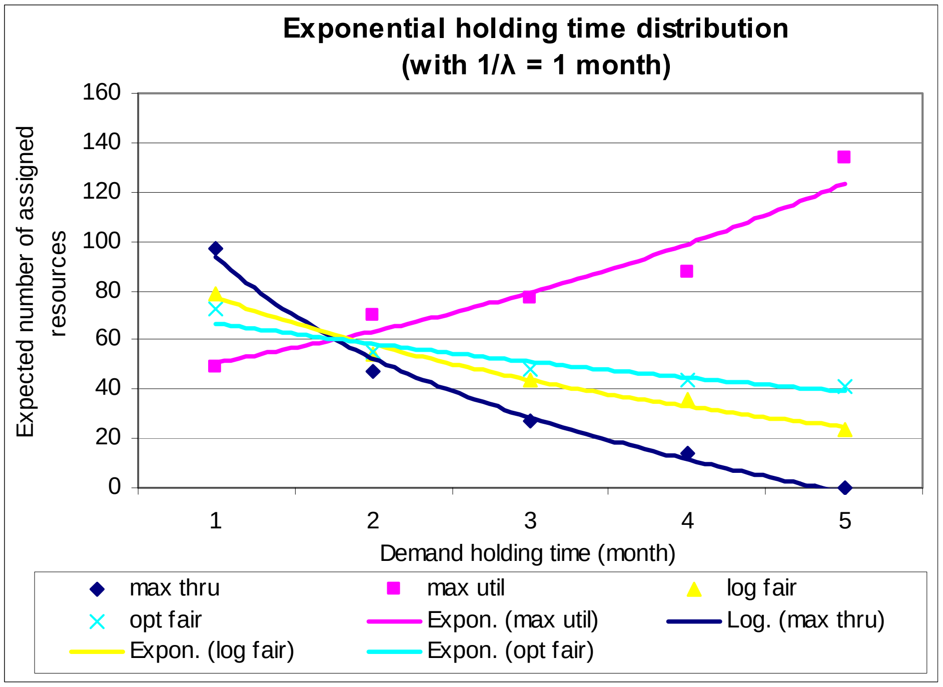

Figure 24 Expected number of allocated resources

In conclusion, it can be said that the *log_fair* objective function gives us useful results in the theoretical limit of the FEDERICA infrastructure performance parameters while keeping the resource allocation reasonably fair. Of course, in real life situations the demands are not known a priori, but this theoretical calculations gives us a idea of what we can expect from the infrastructure, taking into account the given constraints and assumptions.

4.3.3 Final remarks

In this theoretical study, the relatively short and long time requests are distinguished and no different service levels are defined. The fairness policy is trying to ensure that the long time request does not consume all the resources from the short time requests (and vice versa) and in parallel continues to maintain the highest resource utilization and throughput as possible.

Later in the business case study, two service levels are assumed: the guaranteed service (ensuring reproducibility) and the ´best effort´ service for the rest. The fairness statement in that case is defined as: the high-value (guaranteed) requests should not consume all the resources but should allocate some for the low-value (best effort) demands, and as a secondary option, some high-value request can be satisfied by a ´best effort´ service in case of system lack of resources.

See the details of this proposal in Section 5 of this deliverable.

5 Business model for FEDERICA

Although not primarily aimed at commercial exploitation, business models are important for the FEDERICA research and future exploitation of the concept. JRA2 has proposed and experimented with a FEDERICA business model that is optimised for virtualised infrastructures. A complete specification of the assumed FEDERICA market mechanisms and the associated business and value-related issues as well as a representative example are summarised in this chapter.

5.1 Summary of previous work on business issues

An initial, yet very detailed study of business issues and models was carried out in Section 5 of deliverable DJRA2.1. First, related background on business models and participating roles was presented, both in general and with emphasis on systems with a strong IT component. Furthermore, the beneficial effects of virtualisation were identified for such systems. In particular:

- A virtualised infrastructure can be used ‘in-house’ to improve internal efficiency and performance of the IT infrastructure within an organization.
- Multiple organizations can create a ‘pool’ of resources that is shared across the different organizations on an as-needed basis governed by complex policies.

As a consequence, innovative new business models can be built on virtualised infrastructures and IaaS [Ias]. The key element of these business models is the condominium concept applied to the communication infrastructure and equipment. Thanks to virtualisation a user can buy (for example) a port (with its full control) on an expensive piece of equipment without having to buy the cards. Therefore, additional value can be created by the infrastructure brokers. For example, infrastructure brokers can create a search engine that will span across infrastructure providers and discover end-to-end solutions with an associated price for the users. From the point of view of making business through them, infrastructure virtualisation technologies may be considered as an innovative means of creating business channels though the brokers especially in a pan-European scale such as FEDERICA.

Moreover, the main actors in FEDERICA, together with their incentives were identified. FEDERICA being viewed as a provider (or any other similar activity)[1] offers the platform for running experiments in a virtual network, with a virtually private infrastructure. This is a specialized service and to this end, customers of FEDERICA making use of this service could be:

- Industrial or academic institutions or project consortia wishing to test their innovative approaches.

[1] Henceforth, FEDERICA can be interpreted in a broader sense, that is, an activity offering similar services to those provided by this specific project and with a similar approach, because what is discussed and proposed for FEDERICA is also applicable to other such activities.

- Industrial partners offering new software and/or hardware, possibly in collaboration with other industrial partners or projects. The benefits of such customers (providers) may be to test and fine-tune their equipment or software and its interoperability features, to reach a first set of customers, to introduce a new technology, and to help with its adoption, etc.

However, FEDERICA can also be viewed as a platform consisting of network links, hardware (routers, switches, etc.) and software with appropriate capabilities that allow their combination to offer the aforementioned service. Therefore, the providers of such components and infrastructure can also be viewed as FEDERICA customers. FEDERICA can charge them in order to reach the customers aiming to use the platform for experimental purposes. In fact, such customers and providers themselves can also have their own business models. For example, a project could define a complex network topology needed for experimental purposes and reserve a 'slice' of FEDERICA for implementation purposes. This is then offered as a platform to other projects, avoiding the procedure of how to convert the topology of their experiments into a FEDERICA slice. It should be decided whether the analysis of the business models of the FEDERICA customers and providers falls into the scope of the project.

Table 2 summarizes the above issues:

Customer Type	**Incentives**	**Benefits**	**Costs**
Project or Industrial/academic institutions	Test innovative research approaches	Access to the right experimental infrastructure	Possible charges by FEDERICA Overhead to reserve and use slice
Provider of new hardware/software	Testing and promotion of new products	Introduction of new technology to potential customers Testing and fine-tuning of product in a real and challenging environment	Possible charges by FEDERICA Overhead to integrate with platform Disclosure of technology
Provider of 'basic' infrastructure	Attain more customers	Revenue	Possible charges by FEDERICA Overhead to integrate with platform

Table 2 Three main types of customers in FEDERICA

In the sequel to Section 5 of DJRA2.1, an extensive review of related projects was performed, in order to identify similarities with FEDERICA with respect to business issues.

The preliminary study concluded that GENI [Gen] and PlanetLab [Pan] are the most relevant projects to FEDERICA (in terms of business models), because they apply the similar concept of virtualisation. It was explained therein that the selection of the most appropriate approaches depends on the requirements imposed by the users, on the specific features of the FEDERICA service, as well as on the level of demand for the FEDERICA service. Certain alternative solutions were presented that cover the most likely and the most interesting cases for demand along with other requirements.

In this update, some of the business and economic related questions raised in Section 5.3.1 of DJRA2.1 are addressed, and relevant approaches and solutions are proposed. A complete solution for the FEDERICA market mechanism is presented. The solution is sufficiently flexible to allow users to express their resource requirements and their willingness-to-pay and at the same time attains resolution of conflicts in an equitable manner that provides users with the right incentives.

5.2 FEDERICA market mechanism

In this section, a market mechanism is presented for environments such as FEDERICA that offer a subset of virtualised infrastructures. Note that the underlying principles for such a mechanism have already been discussed in Section 5.3.3 of deliverable DJRA2.1, together with several options that were left open.

5.2.1 Resource model

Prior to proceeding with the details of the mechanism, an abstract resource model has to be defined. This model should be broad enough to cope with the current FEDERICA infrastructure resources or other related ones, and should also be capable of accommodating new types of resources introduced in the future. Therefore, we use the abstraction of a set of resources R. The respective number of each type r of resources is denoted as n_r. For example, each link of the infrastructure is such a resource and its capacity is divided to n_r integral units. Although, this notation is general enough to cope with any possible resource type, it is also important to cope with the inherent features of the resources that are currently or expected to be offered by means of FEDERICA. In particular, it is important to take into account the fact that FEDERICA resources are qualitatively heterogeneous:

- Resources such as communication link capacities can be “sliced” (partitioned) to an arbitrary number of units (i.e. these goods are infinitely divisible and the supply of these resources is considered to be “elastic” in the sense that the only actual constraint is the capacity constraint). However, there are no additional technological limitations that must be taken into account, as opposed to the other category of FEDERICA resources (see below).
- Resources that are subsequently referred to as blocking resources. Due to technology limitations, these types of resources are not arbitrarily divisible and can only be divided to an integer number of units. The most typical example of such “blocking” resources is that of a virtual router.

Obviously, the aforementioned nature of the provided resources must be taken into account by the FEDERICA resource allocation mechanism, so that the customers attain meaningful allocations.

After the clarification of the FEDERICA resource issues an appropriate time model has to be defined (i.e. how resources are allocated over time).

5.2.2 Time model

The importance of the time model is that it defines how users may offer/acquire resources. As already introduced in DJRA2.1, a slotted time model (i.e. allocations are decided and are valid for a pre-specified amount of time, referred to as the 'slot') has been selected. This model is actually quite common in all sorts of markets (e.g. oil trading, stock markets) since it allows the synchronization of demand and supply and subsequently the clearing of the market, thus facilitating trade. The fact that allocations are valid for just one slot allows the mechanism to be fast and computationally tractable. It also provides incentives for the rational reservation of resources over time, and in terms of efficiency performs well. This is also demonstrated by the success and wide adoption of this time model in all major economic markets nowadays.

5.2.3 Demand model

In a previous section, the abstract resource notation was introduced but a demand model specification is also needed. Prior to the definition of any mechanism, it is important to have a good idea of the demand for resources in the allocation mechanism. This way, it is easier to tailor the mechanism design decision so as to provide meaningful allocations to the users, facilitating trade, resulting in the efficient operation and high utilization of the system.

As already introduced in DJRA2.1, users require sets of heterogeneous resources to meet their needs, such as some virtual routers and a certain amount of bandwidth over the communication links that interconnect them. Obviously, the scarcity level of these resources is expected to vary significantly. To simplify the presentation, our assumptions are:

- Resources that are under-demanded are given for a fixed price to the users, which is publicly announced and known to all.
- The price of the remaining resources for which there is a competition, should be dynamically decided by the market mechanism, so as to reflect their scarcity and actual value. This is both economically sound and allows the market to have both meaningful prices and efficient resource allocations.

The potential decision on these prices will be elaborated in the following sections where the FEDERICA mechanism is fully specified.

Another feature, which must also be taken into account by the FEDERICA mechanism, is that the user demands can be either flexible or not, in terms of the resource allocations demanded/received. If a user requests a certain set of resources for experimental purposes and the experiment must be repeated under identical conditions, the set of resources must be permanently allocated, and the user demand is not flexible. This imposes stringent requirements for the resource allocation mechanism that should be able to provide this kind of service, and is henceforth referred to as "guaranteed". Alternatively, for other users it may be acceptable to allocate even a subset of alternative resources later in a flexible way. This service should also be made possible and is referred to as "best effort". Lastly, it is also desirable to allow users that initially demand guaranteed service to denote to the resource allocation mechanism that ´best effort´ service is also acceptable when the resource prices are too high for the user to obtain the guaranteed service.

5.3 Complete specification

5.3.1 Types of allocations

In addition to the aforementioned resource, time, and demand issues, the main idea of the mechanism is motivated by the fact that high-value customers should be prioritized first compared to users whose needs are not that urgent and have a low value for an allocation of resources at the current slot. This way, the mechanism can rank the competing service requests according to the users' willingness-to-pay and decide on allocations that are beneficial for the overall social welfare.

Moreover, to avoid "starvation" of low-value customers, even low-value users should be able to get some level of service. However, this is done on a ´best effort´ basis, as opposed to high-value users whose allocations (whenever granted) fully serve their expressed requirements. The guaranteed reproducibility of the experiments performed by a user, in which the same conditions must be met twice, by reserving the same set of virtualised resources is important to note. Furthermore, the mechanism must be flexible enough to provide different semantics on the ´best effort´ service, as detailed below.

To be more precise, a threshold can be defined (e.g. 2/3 = 66,7%) that prescribes the percentage of resources that are allocated (by means of reservations to the high-value customers) for the guaranteed service provision, while the rest (33,3% in this example) is reserved for providing a ´best effort´ service. Obviously, for the blocking resources an integer quantity of units should be actually offered to both types of customers.

5.3.2 Design issues

There are several auction mechanisms that could be used in the context of FEDERICA. The main source of complexity is that each user reserves a multitude of both networking and computational resources and there exits a large range of possible combinations, each with their own advantages and disadvantages. If sealed-bid combinatorial auctions are employed, then the complexity of determining the winner will be prohibitive. On the other hand, if simultaneous ascending auctions are employed (say for each resource auctioned) it is likely that it will last for many rounds, making it difficult for bidders to employ a meaningful strategy due to the multitude of available options. To overcome these difficulties, a non-combinatorial auction has been opted by JRA2 where reservations are valid for just one slot. This is the only viable solution since a combinatorial auction would simply not work in practice.

First of all, it is important to stress the bid definition problem. Given that we have opted for a non-combinatorial auction design, how should a user decide on how to split his/her total budget for a certain set of resources over the individual auctions run for them?

Clearly, the most flexible option (where the users decide arbitrarily on this) is not a realistic option. It would be too complicated for them; it is a non-intuitive and a complex process for the average user for which it is practically impossible to decide on the optimal budget splitting. This could result in inefficient allocations due to erroneous budget splitting which would also make the mechanism unattractive.

A second option would be to allow the user to denote only his/her total willingness to pay and then consider this for the scarcest resource auctioned. However, since a user bid may demand multiple scarce resources (e.g. more than one virtual routers at nodes that exhibit high demand) and also additional resources of a certain quantity, such a rule could result in the wrong incentives for users and bidding strategies that could lead to over-consumption of even the scarcest resources. The decision on the ranking of different bids that compete for the same scarce resource (including others exhibiting different demand where different quantities are wanted) could not be done in a transparent, fair, and incentive compatible way.

Therefore, the last alternative has been selected: each resource type announces a certain weight according to each user's willingness-to-pay, it will then be split accordingly. These weights can be chosen to depict the importance of each resource in a similar way that effective bandwidths in networks are used to determine the difficulty of multiplexing flows with different features at a certain point of the network. For example, if virtual routers exhibit higher demand than communication links over the FEDERICA platform, it could be decided that the weight of virtual routers is 0.70, which is further split among the virtual routers demanded. If the bandwidth is 0.30, it is again further split among the links and the bandwidth units demanded. This is a transparent rule that does not impose uncertainty about the mechanism allocations or any unfairness on the treatment of bids, and it is strategically simple for the users. The selection of these weights can be derived from statistics regarding the spread of demand over the FEDERICA resources that is depicted in the platform statistics. It should also be noted that this splitting of the willingness-to-pay among the various resources is done after subtracting the fixed prices to be paid for the under-demanded ones, if any exists.

Therefore, the bids to be submitted by the users contain:

- Resources and respective quantities sought,
- Respective total willingness to pay,
- A flag demonstrating if ´best effort´ service is acceptable when the bid is not winning at the slot auction.

In summary, it has been specified that the "guaranteed service" allocations are to be decided by means of an auction, while the "best effort service" (whose quality is inferior) is to be less expensive than the guaranteed and provisioned by means of a policy. The next section deals with the detailed auction mechanism.

5.3.3 The auction mechanism

In this section, the steps of the proposed auction mechanism are presented; how the users that compete by bidding for guaranteed service are allocated the FEDERICA resources.

1. For each type of resource, a non-negative reserve price is set.
2. The auction starts by measuring excess demand of the blocking resources.
3. The allocation process (by means of auctioning) begins from the scarcest resource. Once a winner determination has been completed for this resource, the allocation of the next most demanded blocking resource is then determined. This is done by running a sealed bid auction instance for this resource and so on. This process is repeated until all resources are considered.

   Note that for simplicity reasons only resources where demand exceeds supply are considered. Those that do not exhibit excess demand are allocated for a low publicly announced fixed price.
4. For each such resource, a sealed bid auction instance is run where the highest bids win and the guaranteed service is provisioned over the FEDERICA platform. The winner determination phase sorts all bids according to the per-resource unit price and serves the highest of them so that:

   a) Winning bids exceed the reserve price in each separate resource sought,

   b) Are fully satisfied,

   c) Threshold capacity constraint (e.g. 66.7%) is not violated i.e. for each type *r* of resources 0,667*n_r is only allocated.
5. The losing bids are no longer considered for the provision of guaranteed service and are removed from the auctions remaining in this slot. Those ´best effort´ service bids, are transferred to the ´best effort´ queue. Their respective bidders are then notified that their bid has been transferred to the ´best effort´ queue.
6. In cases where demand for the auctioned resources does not exceed supply (due to the removal of bids whose stringent requirements could not be met due to losing in a previously run auction for a resource that was also part of the bidder's request) the transaction price is set to the fixed price of that resource for the ´best effort´ service and the auction essentially blocks no-one.
7. The feedback of the mechanism is the available resources and the current standing prices per resource type.

Note that this mechanism has certain desirable properties. First, it performs well in terms of social welfare, since high-value users are prioritized. It still serves low-value users but with lower precedence, thus providing proper incentives for users to submit high bids. Therefore, it allows the infrastructure to support two layers: one of guaranteed allocation that quickly serves users who urgently need service and are willing to pay for it, and one of ´best effort´ allocation. Finally, the mechanism is flexible enough to work even if the demand is less than the supply of resources; the prices of ´best effort´ services prevail and all services are guaranteed.

So far, the description of the mechanism implies relying on the use of prices and real currency. However, the mechanism definition is broad enough to allow its smooth operation even when this is not the case. Therefore, it is possible to accommodate the following cases:

Option 1: Use of virtual currency

As already explained in DJRA2.1, if an environment such as FEDERICA wishes to offer their resources for free, rather than charging users for actual money, then renewable tokens can be used as an internal control mechanism based on incentives to avoid resource exhaustion. These tokens amount to a renewable right to use per time interval for free up to a certain amount of each of the resources. There are no actual modifications required for the mechanism presented above to work in this case too. Note however, that a solution based on virtual currency is in general considered inferior to one with real currency, due to several reasons: e.g. the uncertainty of the actual value of the "virtual" currency, the associated inflation, the possibility for a secondary market (where low demand users sell their rights to others and can thus make actual money!), and finally the fact that it cannot be used by its owner for infrastructure upgrade, buying/maintaining hardware, etc.

Option 2: No use of currency

It is possible in some contexts to prescribe that pricing is not desirable at all, even with virtual currency. This could be in academic contexts where resources are also under-utilized. In this case, there is no urgent need to involve pricing in the mechanism. However, in order to ensure the viability and growth of the system, it is desirable to introduce an incentive for the partners to contribute resources to the common infrastructure. Therefore, it is proposed that the willingness-to pay in users bids must be replaced by an automatically generated bid which the system computes based on the user's contribution in resources over the past T slots. This way, a user can be certain that he/she will be served satisfactorily from the system and in accordance with his/her needs. The user must also ensure a minimum amount of contribution to the pool of resources. Note also that after time T the contributed resources expire avoiding one-shot contribution by the users who subsequently would have no incentive to contribute more. Moreover, the higher the amount of contributed resources, the higher the computed bid, thus the higher the probability of the user receiving good service. Contrary, low or even no contribution increases the probability of the user's bid being allocated to the "slow" ´best effort´ queue (see next section). Thus, contribution is incited by the way the mechanism is structured.

5.3.4 The ´best effort´ service queue

The auction mechanism for the guaranteed service and for the remaining unallocated FEDERICA resources supporting the ´best effort´ service has been specified above. This section discusses how it performs.

The remaining resources of the platform are shared among all users not previously served, according to a certain policy that is advantageous for FEDERICA users. Let us assume that one of the standard scheduling policies is opted, namely the First Come First Served (FCFS) scheduling policy. This means that losing bids are queued and served whenever it is possible according to the given system load. Note that this is essentially a ´best effort´ service with no guarantees regarding the waiting time. For instance, if the platform is highly loaded, it is possible that the waiting time for users served in a ´best effort´ way may span a significant number of slots.

Regarding the queue management, and in the FCFS queue specifically, there may be at a certain slot unsatisfied bids waiting to be served from previous slots. Thus, the FCFS queue is not cleared on a per slot basis, as opposed to the guaranteed service auction and bids which are considered valid until they are removed by the users that have submitted them. Since such a policy is to be put in place, it must also be decided what happens if a bid (that should be served in the FCFS queue due to the fact that it has the oldest timestamp) cannot be allocated the entire resources demanded due to its large demand for resources and that a subset of them has been allocated for the guaranteed service. In order to increase the utilization of the system, it has been opted that this bid is bypassed at the current slot and will be considered again in the next slot. Due to the difference in demand over time, at some point the required resources may be made available in the future. The fact that the waiting time of such bids may be arbitrarily high is actually an attractive feature! Indeed, users demanding a significant quantity of resources should be incentivized to bid aggressively in order to reserve these resources, rather than attempting to get them for a low price by means of ´best effort´ service.

Prior to adopting the above approach, other policies leading to inherently different semantics regarding the ´best effort´ service were considered. For instance, the remaining unsatisfied bids could be served according to a Round Robin or fair share policy. If that were indeed the case, users would reserve only a fraction of the resources demanded to run their experiments. This would imply a larger execution time and unpredictability for the end users regarding the significance of the actual delivery time of the experiments. .

Finally, to maintain the right incentives, a fixed per-unit price for the ´best effort´ service is specified that cannot exceed the auction reserve prices for each resource type. This price would apply regardless of whether there may be users asking directly for ´best effort´ service or demanding it implicitly due to having lost it in the guaranteed service auction (and having denoted in their bid that ´best effort´ service is desirable). Overall, no matter what the ´best effort´ service semantics are if the quality of serves imposes uncertainty to the users then it is not preferable compared to the guaranteed service as long as the system prices allow the user to receive such a service. This feature prevents the cannibalization of the guaranteed service and also provides proper incentives to users to bid more aggressively.

Note also that each variation of the ´best effort´ service has drawbacks that could be more or less common, depending on the features of the demand exhibited. For instance, FCFS implies that a large request for many resources may result in a possibly very high delay for the rest of the users that wait in the queue. On the other hand, Round Robin and fair share policies where the system is highly loaded may result in “small” allocations that result in limited or zero value for the customers. These features are inherent to these ´best effort´ policies, whose success depends also on a voluntarily good behaviour of the system users, as opposed to auction mechanisms that are very appropriate in providing the right incentives for rational and fair consumption of resources.

5.3.5 Final remarks

The following final observations regarding the resource allocation mechanism are:

- A slotted model is used. This makes the auction for guaranteed service computationally tractable and also simplifies the bidding decisions of the users. Its impact on the ´best effort´ service is also beneficial. Due to the treatment of the bids in the ´best effort´ queue and the demand fluctuations over time, long-term unfairness and starvation of bids are unlikely to occur.
- The guaranteed service is a premium service that can meet the strict requirements of the users and ensure reproducibility of experiments.
- The ´best effort´ service is a less expensive alternative, however, of lower quality.
- The scheduling policy for the ´best effort´ service should be chosen to meet the FEDERICA participant needs. FCFS is a simple yet good solution. Under more sophisticated solutions, if necessary, even multiple ´best effort´ queues could co-exist by partitioning the ´best effort´ resources to thinner slices and allocate them according to different scheduling policies (FCFS, Round Robin, etc.).

The most important remark however comes from the fact that users may wish to reserve certain sets of resources with the reservations spanning over multiple slots. This indicates that there is room for intermediates to appear in the market, namely brokers that have sophisticated knowledge of the demand and price fluctuations of the market and can be successful in reserving resources and accommodating their customer needs. Their role is similar to that of stock market brokers who build a certain portfolio for their customers.

5.4 A representative example

In order to clarify the main issues of the FEDERICA auction, a simple yet complete and representative example is provided in this section. To facilitate the presentation for the reader and without loss of generality, the set of resources, their respective capacities, as well as some of the auction parameters have been simplified.

Let us assume that at slot *t* the following FEDERICA resources are to be allocated by means of the auction:

- 3 routers, namely R1, R2, R3. Routers R1 and R2 can be sliced to 3 virtual routers (VR), while router R3 can be sliced to 10 VR.
- Links R1-R2, R2-R3, R1-R3. The capacity of links R1-R2, R2-R3 is 3 Mbps, while that of R1-R3 30 Mbps.
- The threshold of the guaranteed service is set to 2/3: This means that at most 2/3 of the total resources of the infrastructure will be allocated by means of the auction to users demanding guaranteed service. All the remaining resources that are left after the auction winners have been determined will be transferred to the ´best effort´ queue.
- The reserve price for 1 VR is 5€ and for 1 Mbps of link capacity is set to 1€.
- The budget splitting threshold is set to 0.7 for 1 VR and 0.3 for 1 Mbps. This means that after removing the possible charges for the uncongested fixed price resources, the relative value according to which the remaining budget is to be split is 70% for the total number of virtual routers demanded and 30% for every Mbps of the links.

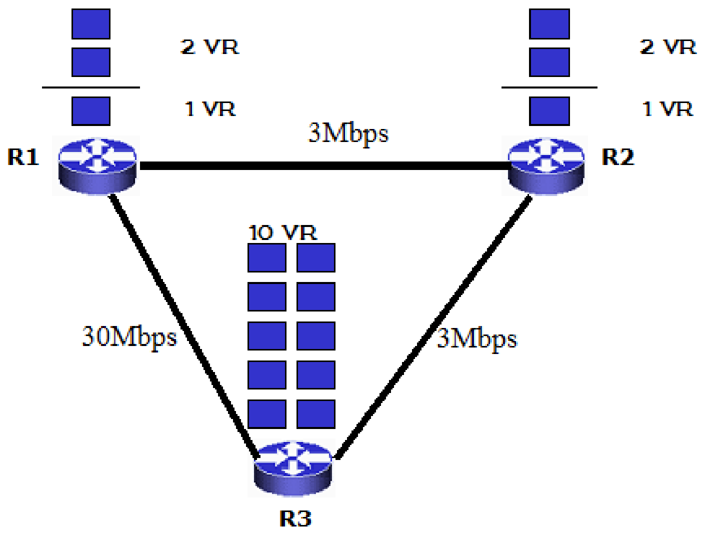

Figure 25 Network example

Let us further assume that router R3 and Link R1-R3 exhibit lower demand than supply; hence users that demand resources there will be surely allocated what they request and will be charged by means of a predetermined, publicly announced fixed price. Therefore, we focus attention on the resources of the FEDERICA platform where there exists competition, namely routers R1, R2 and links R1-R2, R2-R3. The user bids that include these resources are the following:

- B1: 1 VR of R1, 1 VR of R2, 1 VR of R3, 1 Mbps of R1-R2, 1 Mbps of R2-R3. Willingness to pay (wtp) is 100€ and user does not wish only a ′best effort′ service.
- B2: 1 VR of R1, 1 VR of R2, 1 VR of R3, 1 Mbps of R1-R2, 1 Mbps of R2-R3. Willingness to pay (wtp) is 90€ and user can accept a ′best effort′ service if he/she fails to win at the auction.
- B3: 1 VR of R1, 1 VR of R2, 1 VR of R3, 1 Mbps of R1-R2, 1 Mbps of R2-R3. Willingness to pay (wtp) is 80€ and user can accept a ′best effort′ service if he/she fails to win at the auction.
- B4: 1 VR of R2, 1 VR of R3, 1 Mbps of R2-R3. Willingness to pay (wtp) is 30€ and user can accept a ′best effort′ service if he/she fails to win at the auction.
- B5: 1 VR of R2, 1 VR of R3, 1 Mbps of R2-R3. Willingness to pay (wtp) is 20€ and user does not wish only a ′best effort′ service.
- B6: 1 VR of R1, 1 VR of R2, 1 VR of R3, 1 Mbps of R1-R2, 1 Mbps of R2-R3. Willingness to pay (wtp) is 15€ and user does not accept a ′best effort′ service if he/she fails to win at the auction.
- B7: 1 VR of R1, 1 VR of R3, 1 Mbps of R1-R3. Willingness to pay (wtp) is 40€ and user does not accept a ′best effort′ service if he/she fails to win at the auction.

Let us assume that due to the previous runs of the auction (i.e. up to slot *t*) there are the following waiting bids in the ′best effort′ queue:

- BEB1: 1 VR of R1, 1 VR of R2, 1 VR of R3, 1 Mbps of R1-R2, 1 Mbps of R1-R3, 1 Mbps of R2-R3.

Finally, it is worth noting that due to the fact that the auction threshold for the provision of guaranteed services is set to 2/3, two of the three VRs of routers R1 and R2 will be auctioned, along with 2 Mbps of links R1-R2 and R2-R3. Note that in order to simplify the numerical calculations of this example, we consider the fixed prices of the resources where there is no actual competition to be set to 0€.

To finalize the notation used in the remainder of this example, the bid identifier for a set of resources (e.g. B1 as the identifier of the end user who submitted it) is used. Therefore, the user who submitted bid B1 is also referred to as B1.

The auction will begin by estimating the excess demand at the blocking resources i.e. the routers:

- Router R1: Due to bids B1, B2, B3 and B6, demand for R1 is 4 VRs, while the auction supply is 2 VRs. Thus, the normalized competition level is 5 /2.
- Router R2: Due to bids B1, B2, B3, B4, B5, B6, demand for R2 is 6 VRs, while the auction supply is 2 VRs. Thus the normalized competition is 6 /2 = 3, rendering router R2 the scarcest resource.

Next, the auction will estimate competition for the non-blocking resources i.e. the links. By working in a similar fashion, we obtain the normalized competition level for the link R1-R2 to be 5/2 due to bids B1, B2, B3 and B6; that of Link R2-R3 is 6/2=3.

Therefore, the auction instances are to be run in the following order:

1. Router R2,
2. Router R1,
3. Link R2-R3,
4. Link R1-R2.

(Recall at this point that blocking resources are to be considered first.)

The individual bids submitted to the individual auctions, as a result of the 70-30% budget splitting rule, are defined next. Note also that from the above resources B7 only bids for R1; therefore, the bid of B7 should not be split.

	R2	R1	R2-R3	R1-R2
Bids of B1	35€	35€	15€	15€
Bids of B2	31.5€	31.5€	13.5€	13.5€
Bids of B3	28€	28€	12€	12€
Bids of B4	21€	-	9€	-
Bids of B5	14€	-	6€	-
Bids of B6	5.25€	5.25€	2.25€	2.25€
Bids of B7	-	40€	-	-

Table 3 Bids submitted to the resource auctions

As explained in the presentation of the auction mechanism, first a sealed bid auction is run for the 2 VRs of router R2. The auction winners are B1 and B2, as depicted by the shading of the corresponding cells. Moreover, all the users that have participated in the auction and failed to be allocated any resources, namely users B3, B4, B5 and B6 are not considered for guaranteed service provisioning anywhere else in the FEDERICA platform. In this example, this means that only B1, B2, and B7 will keep competing in the auctions for the remaining resources of the FEDERICA guaranteed service. This is exhibited in Table 4, which depicts the current candidate bids for the remaining auctions, after auction R2 is concluded.

	R2	R1	R2-R3	R1-R2
Bids of B1	Auction Concluded	35€	15€	15€
Bids of B2		31.5€	13.5€	13.5€
Bids of B7		40€	-	-

Table 4 Next candidate bids when the first one is closed

The winning bids in the auction for R1 are those of B7 and B1. Both users B1 and B7 will be allocated the required resources and provisioned with guaranteed service. Indeed, user B7 bids in none of the remaining auctions for the links R2-R3 and R1-R2. Also, user B1 is the only remaining user in these two auctions, and thus he/she automatically wins. All the other users are considered having lost in the auction, including B2, whose 1 VR of R2 already won is transferred to the ´best effort´ service queue together with the 2 Mbps of bandwidth left in the links R2-R3 and R1-R2.

A decision needs to be made whether the users who failed to be winners in the auction can be served by means of the ´best effort´ queue. This option only applies to the users with flagged bids for ´best effort´, namely users B3 and B4. These bids have lower precedence than BEB1 that is already waiting in the queue. Therefore, user BEB1 will be allocated 1 VR of R1, 1 VR of R2, 1 VR of R3, 1 Mbps of R1-R2, 1 Mbps of R1-R3, and 1 Mbps of R2-R3. Bids B3 and B4 were submitted at the same slot, and thus have the same time precedence in the queue. However, since B3 has a higher total willingness-to-pay it is examined first. B3 demands (among others) 1 VR in R1, which is no longer available. Thus, B3 will be considered for service after the auction is run for slot *t+1*. All resources demanded by bid B4 are available, and thus the corresponding user can be served. Notice that this happened because the 1 VR of R2 that B2 won in the auction (where B4 was not successful), was subsequently returned in order to increase the efficiency of the mechanism.

5.5 Associated business and value-related issues

In general, FEDERICA facilitates the sharing of resources and the usage of a virtual infrastructure. This brings substantial benefits to all the main actors of the market and indicates that there is room for intermediates to appear in the market, namely brokers that have sophisticated knowledge of the demand and price fluctuations of the market [Bro] and thus can be successful in reserving resources and accommodating their customer needs.

In this section, the key market actors of a brokering model and their business relationships are identified. In addition, some scenarios that demonstrate how the resource allocation mechanism described earlier fits in this context, brings value to the users, and how the FEDERICA platform can be used in practice are described.

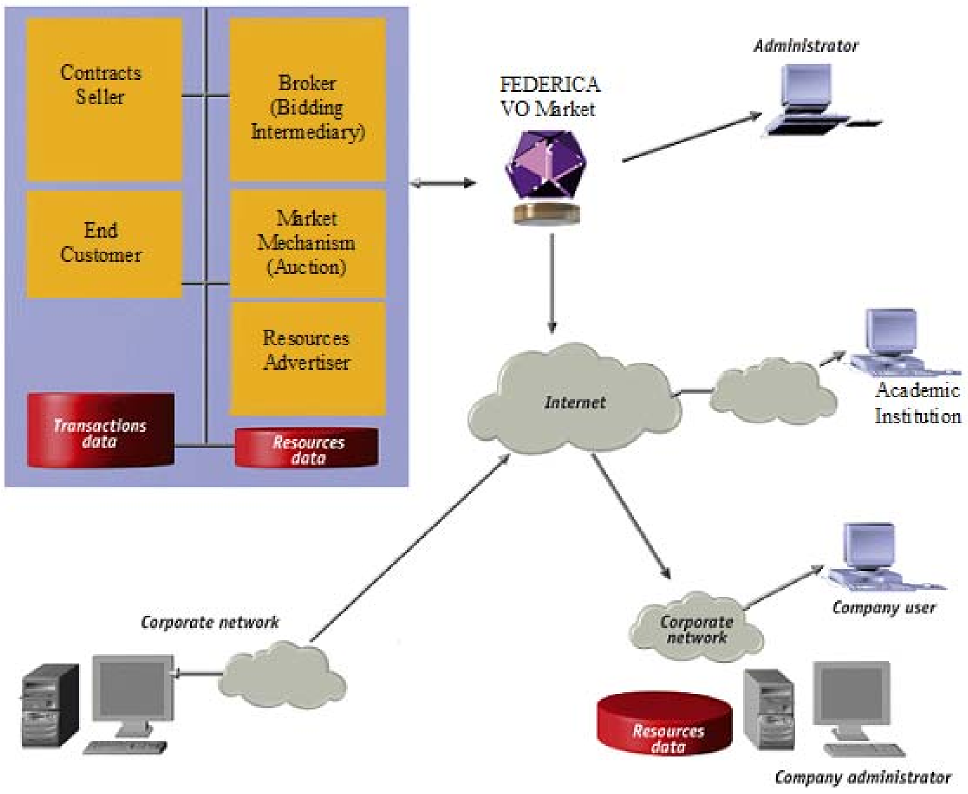

Figure 26 Key business actors of FEDERICA assuming resource broker

As depicted in Figure 26, the key actors are as follows:

- Infrastructure owners: These are the owners of the FEDERICA resources, such as routers, communication links, etc. They contribute to the supply of resources that are to be auctioned through the FEDERICA auction mechanism.
- Brokers/service providers: These are the market participants that reserve resources from the infrastructure owners and use them to offer a brokering service or to provide a computational service to the end users who wish to run certain experiments.
- End users: They constitute the demand side of the market and are comprised of users or institutions that wish to run experiments on top of the FEDERICA platform.

Figure 26 also includes the important role of the Resources Advertiser, because in order to auction the resources, the market must advertise what is available. This does not have to be a separate entity but could potentially be merged with the market mechanism. Similarly, intermediaries such as the contracts seller who is the retailer for the end customer and the broker that bids in the market slots auctions can also be possibly merged. It is worth noting that the use of a common platform brings substantial benefits to all market participants. The common contribution of resources of multiple infrastructure owners over the same virtual platform results in a larger market, as opposed to each provider attempts to sell its own resources. Due to the fact that communication networks, computation platforms and markets in general have strong externalities, it is commonly accepted that unless a certain market reaches a critical mass threshold, its viability is at stake. The multiplexing of multiple infrastructure providers is positive for the size of the market and is thus more likely to attract end user demand, as opposed to the case of bilateral leasing or one-provider markets. Moreover, the virtualisation capabilities enable an even more extensive multiplexing of resources, their higher utilization, the provision of tailor-made solutions for the users at lower prices, due to broader sharing of the cost. Clearly, end users benefit highly from these externalities and features, when there is a healthy and viable market they can rely on that offers them the resources they need in order to meet their needs. Also, the fact that the market consists of multiple providers limits the threat of lock-in for the customers to a certain provider, which is always highly undesirable. Since the market is essentially competitive, and an auction resource allocation mechanism is adopted, it is guaranteed that the market prices will reflect the amount of competition exhibited for the resources.

Finally, intermediaries that provide value-added services also benefit from the way FEDERICA resources are allocated. Due to the simplicity of the resource allocation mechanism, and requirements for a simple, fast, and intuitive market for multiple heterogeneous resources, there is a business opportunity for intermediaries to act on behalf of the users and create by means of bidding, complex networks (configured or un-configured) that can be sold to the end users.

Such intermediaries face the cost of leasing the required resources (i.e. there is a revenue stream from the intermediaries to the resource owners). Based on the resources leased, these intermediaries may offer to end users the option of leasing raw resources or a configured network, by combining resources they have leased. This way, they can meet market demand efficiently and reduce the uncertainty that bidding in independent auctions for resources inherently entails. Given the fact that these intermediaries can multiplex their customers' needs, they can provide efficient service over time and over physical resources by forming meaningful bundles from the resources allocated to them. They essentially serve as a secondary market that resells bundles of resources and services to the users that need them (and do not have to face the risk of bidding themselves) and allows them to add a profit margin to the bundles resold. Note that an additional benefit of the auction brokers in particular is that they can charge end users with fixed-price contracts due to their knowledge of the price fluctuations in the auction derived from past experience/statistics and market analysis. This is also an attractive feature for end users who do not face uncertainty regarding the actual cost of utilizing the FEDERICA resources in the long run.

This business scenario and the respective revenue flows are graphically depicted in Figure 27 for the case that involves one intermediary serving as contracts seller and bidding broker for the end users. In the more general case where these roles would be separated, the Intermediary box would be split in two with the Contracts Seller paying the Auction Broker who would subsequently bid in the auction.

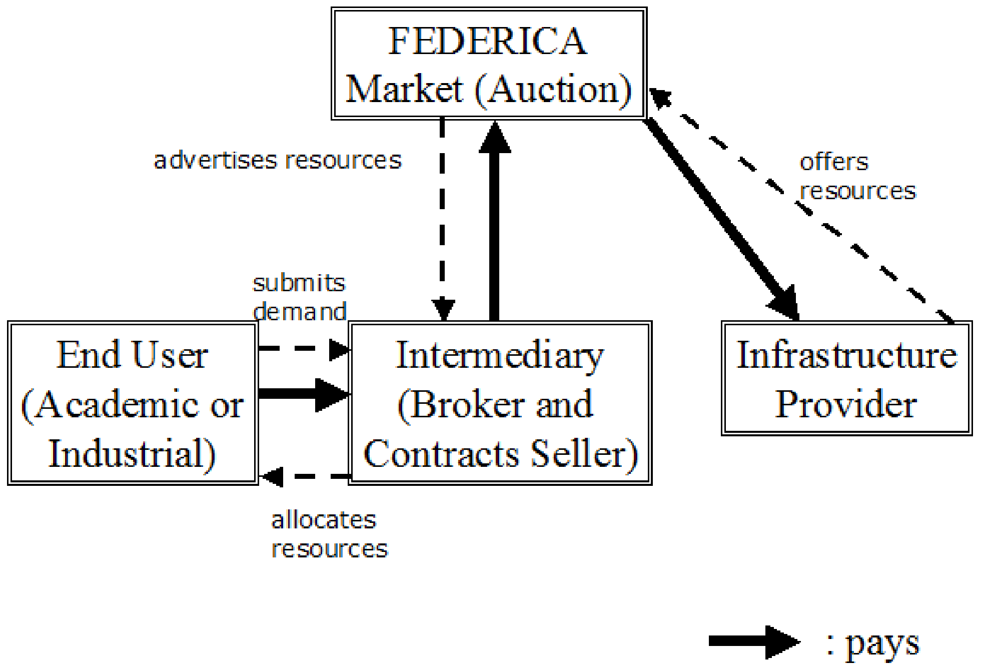
FEDERICA Market (Auction)
advertises resources
offers resources
submits demand
End User (Academic or Industrial)
Intermediary (Broker and Contracts Seller)
Infrastructure Provider
allocates resources
: pays

Figure 27 Flow of value and money in a FEDERICA business scenario

Last but not least, the infrastructure owners also benefit from the existence of these intermediaries since:

- It is not required for them to develop such services, which is not their core business and imposes additional costs for them.
- Market entry to the FEDERICA platform remains simple, since they just need to offer their resources for sale i.e. there are no entry barriers or exclusion effects.
- Due to the competition of the intermediaries, new efficient services are offered in the market, which attracts more demand, thus resulting in higher revenue for themselves.

Finally, it is worth emphasizing that the FEDERICA platform allows the easy offering and purchasing of resources, while each market actor can focus on its own core business and market role, without facing additional transaction costs. In order to illustrate the aforementioned issues, a sample business case scenario is presented below.

Let us assume that a set of service providers offer their resources to the FEDERICA platform. These resources are advertised and auctioned by the FEDERICA market, slot by slot. End customers purchase contracts from the brokers for either raw resources or certain network configurations. Each broker by knowing its customers needs subsequently bids in the marketplace in order to reserve over time the required resources. (The auction resource prices are not necessarily communicated to the end users on a per slot basis, but rather affect the contract prices over longer time intervals.) Note that the urgency of reserving resources in the market actually depends on the kind of contracts these brokers sell: It could be the case that a certain set of raw resources or network configurations are specified in the contract to be delivered within a broader time interval (consisting of more slots than those for which the user demands resources). This facilitates the broker to multiplex more efficiently and better cope with any unfortunate auction results that it may encounter at certain slots in order to satisfy the customer demand. Therefore, end users purchase contracts from the brokers, who subsequently bid in the auction, thus transferring a part of their revenue to the market and consequently to the infrastructure owners.

As a final remark, it should be noted that by adopting an auction market all the issues of resource allocation, price discovery, flow of payments, etc. can be solved simultaneously.

6 Summary

In the final section, the main contribution of the Joint Research Activity JRA2 to the FEDERICA project and also to the future directions of virtualisation capable infrastructures, systems are summarised. Note that this deliverable is the closing report of the JRA2's main activities. In this last phase of FEDERICA JRA2, it acts as an 'internal user' to the project since it is using its own infrastructure to validate the final resource results. Those results will be reported under the networking activity. The final IPsphere – FEDERICA interoperability prototype test results will be reported in the JRA2.4 deliverable at the end of the FEDERICA project.

6.1 Main achievements

All the JRA2 activities are grouped into three tasks. Task TJRA2.1 has been working on novel architectural paradigms. The two major research studies on multi-stage software router architecture and flow-based virtualisation platform are reported in this deliverable.

Task TJRA2.2 has been working on novel concepts for fair sharing, security, accounting and business models. Proper solutions for both the security issues in current and future FEDERICA as well as the business mechanisms optimised for FEDERICA and generalised to any future virtualised infrastructure are given in this deliverable. The fairness issues are also discussed in details.

Task TJRA2.3 has been working on the IPsphere-FEDERICA interoperability prototype. The related achievements are reported in separated deliverables but the concept of interconnected or federated virtualisation capable infrastructures motivated all the aforementioned research areas of JRA2.

6.2 Final words

As it was stated before, the final results of the IPsphere – FEDERICA interoperability prototype test (done jointly by the JRA2 and SA2 partners using a FEDERICA slice) will briefly be reported in the last deliverable DJRA2.4 of the JRA2 activity. The latest results of the multi-stage software router architecture tests and the OpenFlow investigations will be included in the deliverable DNA2.3 of the NA2 activity as internal user slices.

Finally, it is worth mentioning that the virtualisation architectures in general still has a number of challenges and FEDERICA itself can just be seen as a small pilot project implementing and demonstrating the feasibility of the basic concepts on a pan-European scale. Further extension of the physical resource set towards the transmission (e.g. photonic) layers as well as the wireless and access technologies can be envisioned. Enhancements on the resource control and management plane software towards the integration and standardisation of (yet proprietary) cloud middleware are also challenging.

References

[Aec]	http://aws.amazon.com/ec2/
[Bri]	http://www.cs.bu.edu/brite/
[Bro]	Kevin Lai: Markets are Dead, Long Live Markets Sigecom Exchanges, Vol. 5, No. 4, July 2005, Pages 1-10.
[Cmm]	Andrea Bianco, Robert Birke, Jorge Finochietto, Luca Giraudo, Fabio Marenco, Marco Mellia, Azeem Khan, D. Manjunath, Control and Management Plane in a Multi-stage Software Router Architecture, HPSR 2008 (IEEE Workshop on High Performance Switching and Routing), Shanghai China, May 15-17, 2008
[Cmo]	https://tom1.cesnet.cz/netreport/federica_utilization1/
[Cur]	http://curlftpfs.sourceforge.net/
[Edu]	GN2: Deliverable DJ5.2.3,2: Best Practice Guide - AAI Cookbook - Second Edition Guidelines for Connecting to the eduGAIN AA Infrastructure http://www.geant2.net/upload/pdf/GN2-07-023v4-DJ5-2-3_2_Best_Practice_Guide-AAI_Cookbook-Second_Edition.pdf
[Emu]	https://users.emulab.net/trac/emulab/wiki
[Erm]	http://en.wikipedia.org/wiki/Erd%C5%91s%E2%80%93R%C3%A9nyi_model
[Fsh]	http://www.rediris.es/fedssh/doc.html
[Fus]	http://fuse.sourceforge.net/
[Gen]	http://www.geni.net/
[Gss]	http://www.gssim.org/
[Ias]	http://www.iaasframework.com/
[Ips]	Kevin Dillon: 'IPsphere Forum Backgrounder', Chair – IPsphere Forum, 2005 http://www.ipsphereforum.org/
[Lia]	http://www.projectliberty.org/
[Lib]	William Duserick, Fidelity Investments: Whitepaper on Liberty Protocol and Identity Theft, February 20, 2004 http://www.projectliberty.org/liberty/content/download/389/2726/file/Liberty_Identity_Theft_Whitepaper.pdf
[Lpm]	http://en.wikipedia.org/wiki/Linear_programming
[Man]	Eduard Grasa , Xavier Hesselbach , Sergi Figuerola Victor Reijs, Dave Wilson, Jean-Marc Uzé, Lars Fischer, Tomas de Miguel: 'The MANTICORE project: Providing users with a Logical IP Network Service', TERENA Networking Conference 2008, Bruges, Belgium
[Msa]	Andrea Bianco, Jorge Manuel Finochietto, Giulio Galante, Davide Mazzucchi, Marco Mellia, Fabio Neri, Scalable Layer-2/Layer-3 Multistage Switching Architectures for Software Routers, IEEE GLOBECOM 2006, San Francisco, CA, USA, November 27-30, 2006
[Msr]	Andrea Bianco, Jorge Finochietto, Marco Mellia, Fabio Neri, Giulio Galante, Multistage Switching Architectures for Software

Routers, IEEE Network "Advances in Network Systems", Vol.21, No.4, pp.15-21, ISSN: 0890-8044, July 2007

[Ols]	http://everlab.cs.huji.ac.il/planetstats/
[Omn]	http://www.omnetpp.org/
[Opf]	http://www.openflowswitch.org/
[Osr]	Bianco, A. Finochietto, J. M. Galante, G. Mellia, M. Neri, F: Open-Source PC-Based Software Routers: A Viable Approach to High-Performance Packet Switching, 2005, ISSU 3375, pages 353-366
[Pan]	Larry Peterson, Steve Muir, Timothy Roscoey, Aaron Klingaman: 'PlanetLab Architecture: An Overview', PDN–06–031, May 2006 http://www.planet-lab.org/
[Res]	REST: Representational State Transfer, http://oreilly.com/catalog/pwebserperl/chapter/ch11.pdf
[Soa]	http://en.wikipedia.org/wiki/Service-oriented_architecture
[Vin]	Andy Bavier, Nick Feamster, Mark Huang, Larry Peterson, Jennifer Rexford: 'In VINI Veritas: Realistic and Controlled Network Experimentation', ACM SIGCOMM, Pisa, Italy, September 2006 http://www.vini-veritas.net/
[Wax]	Hamed Haddadi: Understanding the Internet Topology http://www.informatics.sussex.ac.uk/research/projects/ngn/slides/msn08talks/haddadi_topology.pdf

Appendix I: Example of a response from Discovery Service

```
<QueryResponse>
 <Status code="OK"/>
 <wsa:EndpointReference>
   <wsa:Address>http://sp.com</wsa:Address>
   <wsa:Metadata>
    <ds:Abstract>SP.COM service</ds:Abstract>
    <ds:ProviderID>http://sp.com/</ds:ProviderID>
    <ds:ServiceType>urn:sp:service:1.0</ds:ServiceType>
    <ds:Framework Version="2.0" />
    <ds:SecurityContext>
      <ds:SecurityMechID>
         urn:liberty:security:2006-08:ClientTLS:SAML V2
      </ds:SecurityMechID>
      <sec:Token>
        <saml:Assertion ...>
                ...
        </saml:Assertion>
      </sec:Token>
    </ds:SecurityContext>
   </wsa:Metadata>
 </wsa:EndpointReference>
</QueryResponse>
```

Appendix II: SAML constructs for Relayed Trust

The SAML construct, used in the eduGAIN WE case, must be able to convey information about the user accessing the resource and fulfil two essential constraints:

- It has to be bound to the client by the IdP, so it is possible to check that the information about the user that it contains has been legally obtained.
- It has to be bound to the resource by the client, so a potentially malicious resource cannot use this information to further impersonate either the client or the user.

To comply with these two requirements, the client will build a SAML assertion expressing data related to the authentication with:

- A valid audience restricted to the resource it is addressed to, through a SAML condition element containing an URI uniquely identifying the resource.
- A statement that specifies the method of relayed trust must be used to evaluate the assertion through a specific value in the SAML construct identifying the subject confirmation method. This value is the following URI in the eduGAIN namespace: `urn:geant:edugain:reference:relayed-trust`

- The SAML assertion(s) received from the web container as evidence for this confirmation process, as part of the SAML element `SubjectConfirmationData`.

A sample SAML assertion following the above procedures for a given client with the eduGAIN component identifier (CId):
`urn:geant:edugain:component:perfsonarclient:NetflowClient10082`

Acting on behalf of a user that it is identified by a BE with Cid:
`urn:geant:edugain:be:uninett:idp1`

And connecting to a resource identified by:
`urn:geant:edugain:component:perfsonarresource:netflow.uninett.no/data`

Should have a SAML 2.0 content as the one displayed below (some line breaks and indentation added to improve readability):

```
<?xml version="1.0" encoding="UTF-8"?>
<Assertion xmlns="urn:oasis:names:tc:SAML:2.0:assertion"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="urn:oasis:names:tc:SAML:2.0:assertion
   file:/Users/andreas/Documents/UNINETT/AAISpecs/SAML-2.0/oasis-
sstc-saml-schema-assertion-2.0.xsd"
   Version="2.0" ID="100001" IssueInstant=”2006-12-03T10:00:00Z”>
  <Issuer>
    urn:geant:edugain:component:perfsonarclient:NetflowClient10082"
  </Issuer>

<!-- An audience restriction, that will restrict this security token
to be valid for
     one single resource only. -->
  <Conditions>
    <AudienceRestriction>

<Audience>urn:geant:edugain:component:perfsonarresource:netflow.unine
tt.no/data</Audience>
    </AudienceRestriction>
  </Conditions>

  <Subject>
    <NameID>aksjc7e736452829we8</NameID>
    <SubjectConfirmation Method=”urn:geant:edugain:reference:relayed-
trust”>
      <SubjectConfirmationData>
        <Assertion xmlns="urn:oasis:names:tc:SAML:2.0:assertion"
         xmlns:xsi="http://www.w3.org/2006/XMLSchema-instance"
         Version="2.0" ID="_200001" IssueInstant="2006-12-
03T10:00:00Z">
          <Issuer>urn:geant:edugain:be:uninett:idp1</Issuer>

<!-- This inner assertion is limited to only be valid for the client
performing the WebSSO
     authentication. This inner assertion cannot be reused or used at
all by others than the
     NetflowClient10082 instance. But NetflowClient10082 can use it
as an evidence when used inside an
```

```
     assertion issued by NetflowClient10082 using the relayed-trust
confirmationMethod. -->
          <Conditions>
            <AudienceRestriction>

<Audience>urn:geant:edugain:component:perfsonarclient:NetflowClient10
082</Audience>
            </AudienceRestriction>
          </Conditions>

<!-- This is the inner Subject and authNstatement proving the
authentication itself. These elements
     and attributes must be identical in the inner and outer
assertion:
     - Assertion/Subject/NameID
     - Assertion/AuthnStatement@AuthenticationMethod
     The inner assertion confirmation Method must be
urn:oasis:names:tc:SAML:1.0:cm:bearer. -->
          <Subject>
            <NameID>aksjc7e736452829we8</NameID>
            <SubjectConfirmation
Method=”urn:oasis:names:tc:SAML:2.0:cm:bearer”/>
          </Subject>

          <AuthnStatement AuthnInstant="2006-12-03T10:00:00Z">
            <AuthnContext>
              <AuthnContextClassRef>
                urn:oasis:names:tc:SAML:2.0:ac:classes:Password
              </AuthnContextClassRef>
            </AuthnContext>
          </AuthnStatement>
          <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
<!-- Signed by the IdP (or Home Bridging element) -->
            <SignedInfo>
              <CanonicalizationMethod Algorithm="…"/>
              <SignatureMethod Algorithm="…"/>
              <Reference>
                <DigestMethod Algorithm="…"/>
                <DigestValue/>
              </Reference>
            </SignedInfo>
            <SignatureValue/>
          </Signature>
        </Assertion>
      </SubjectConfirmationData>
    </SubjectConfirmation>
  </Subject>

<!-- The authNstatement issued by the client itself -->
  <AuthnStatement AuthnInstant="2006-12-03T10:00:00Z">
    <AuthnContext>
      <AuthnContextClassRef>
        urn:oasis:names:tc:SAML:2.0:ac:classes:Password
      </AuthnContextClassRef>
    </AuthnContext>
  </AuthnStatement>
  <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
<!-- Signed by client -->
```

```
    <SignedInfo>
      <CanonicalizationMethod Algorithm="…"/>
      <SignatureMethod Algorithm="…"/>
      <Reference>
        <DigestMethod Algorithm=".."/>
        <DigestValue/>
      </Reference>
    </SignedInfo>
    <SignatureValue/>
  </Signature>
</Assertion>
```

Appendix III: Examples of simulation framework descriptions

a) Slice description example in GSSIM

```
        <!--

            slice description for following topology:

            Node1 - - - Router1 - - - Node2 - - - Router2 - - - Node3

        -->
        <slice id="slice1"
                startTime="2008-11-03T01:00:00.000+01:00"
                endTime="2008-11-03T06:00:00.000+01:00"
                duration="PT2H">
                <resources>
                        <resource type="node" id="Node1" />
                        <resource type="node" id="Node2" />
                        <resource type="node" id="Node3" />
                        <resource type="router" id="Router1" />
                        <resource type="router" id="Router2" />
                </resources>

                <topology>
                        <connection>
                                <endpointId>Node1</endpointId>
                                <endpointId>Router1</endpointId>
                        </connection>
                        <connection>
                                <endpointId>Node2</endpointId>
                                <endpointId>Router1</endpointId>
                        </connection>
                        <connection>
                                <endpointId>Node2</endpointId>
                                <endpointId>Router2</endpointId>
                        </connection>
                        <connection>
                                <endpointId>Node3</endpointId>
                                <endpointId>Router2</endpointId>
                        </connection>
                </topology>
        </slice>
```

b) The following example shows configuration for the PSNC Core PoP:

```
<computingResource  resourceId="PSNC-vnServer1">
   <machineParameters>
         <hostParameter name="cpucount">
               <paramValue>20</paramValue>
         </hostParameter>
         <hostParameter name="application">
               <paramValue>node</paramValue>
         </hostParameter>
         <otherParameter name="reservation">
               <paramValue>true</paramValue>
         </otherParameter>
   </machineParameters>
</computingResource>

<computingResource  resourceId="PSNC-vnServer2">
   <machineParameters>
         <hostParameter name="cpucount">
               <paramValue>20</paramValue>
         </hostParameter>
         <hostParameter name="application">
               <paramValue>node</paramValue>
         </hostParameter>
         <otherParameter name="reservation">
               <paramValue>true</paramValue>
         </otherParameter>
   </machineParameters>
</computingResource>

<computingResource  resourceId="PSNC-router">
   <machineParameters>
         <hostParameter name="cpucount">
               <paramValue>16</paramValue>
         </hostParameter>
         <hostParameter name="application">
               <paramValue>router</paramValue>
         </hostParameter>
         <otherParameter name="reservation">
               <paramValue>true</paramValue>
         </otherParameter>
   </machineParameters>
</computingResource>
```

Appendix IV: Simple LP formulisations of fairness strategies

Constants

D – number of demands

T – project time

C_t – network capacity at time t

$$\delta_{d,t} = \begin{cases} 1 & - \text{ if demand } d \text{ is served by the network at time } t \\ 0 & - \text{ otherwise} \end{cases}$$

Generation of the $\delta_{d,t}$ matrix:
Demand holding time has an exponential distribution with the expected value of 1/λ
Demand start time has a uniform distribution on $t \in (1..T)$

Variables

$x_d \geq 0 \quad ; \quad \forall d \in (1..D), \quad x \in N$

$c_t \geq 0 \quad ; \quad \forall t \in (1..T), \quad c \in N$

Constraints

$$\sum_d (\delta_{d,t} \cdot x_d) \leq C_t \quad ; \quad \forall t \in (1..T)$$

Objective functions according to the four algorithms

- Max_thru:

$$\max\left[\sum_d x_d\right]$$

- Max_util:

$$\max\left[\sum_t c_t\right]$$

- Log_fair:

$$\max\left[\sum_d \log(x_d)\right]$$

- Opt_fair:

$\max[\min[x_d]]$

The opt_fair case can be solved with the following algorithm:

Step 0. $\bar{x}_d := 0$

Step 1. $\max[a]$ subject to $a \cdot \left(\sum_d \delta_{d,t}\right) \leq C_t \; ; \; \forall t \in (1..T)$

Step 2. $c_t := c_t + a \cdot \left(\sum_d \delta_{d,t}\right); \forall t \in (1..T)$

$x_d := x_d + a$

remove all time t where $\min(c_t) \; ; \forall t \in (1..T)$

remove all demand d where $\delta_{d,t} = 1 \; ; \forall t$ removed

Step 3. if no more demands STOP
otherwise GO TO Step 1.