Whitepaper

The FELIX architecture for large scale SDN and NSI experiments

April 2015

1 Introduction

Network programmability via Software Defined Networking (SDN) tools and dynamic on-demand network service provisioning have emerged as key ingredients of Future Internet architecture. It is also common conviction that testing of these innovative solutions for network control needs to occur over large-scale testbeds in order to emulate near real-world testing conditions and allow for the wider technical, industrial and societal impact. Significant efforts by several projects in Europe, Asia and the US have been made over the last years to create experimental research infrastructures that are reusable and can incorporate the latest network technologies as they emerge. The FELIX project [1] is part of this international research experimentation infrastructure strategy (in Europe under the Future Internet Research Experimentation – FIRE – framework), with a special focus on SDN and Network Service Interface (NSI) developed by the Open Grid Forum. FELIX aims at facilitating the federation and integration of different network and computing resources controlled via SDN and NSI in a multi-domain heterogeneous environment across, initially spanning Europe and Japan. To achieve this, the FELIX consortium has designed and is implementing an architecture that extends and advances assets previously developed in other Future Internet projects (e.g. OFELIA), by realizing the federation concepts defined in SFA [2] and implemented by GENI [3] with a combination of recursive and hierarchical configurations for orchestration, request delegation and inter-domain dependency management. Resource orchestrating entities are responsible in FELIX for the synchronization of resources available in particular administrative domains.

This whitepaper presents the FELIX architecture and its key features with the aim of introducing capabilities and opportunities offered by the infrastructure and its control framework. Specifically, this whitepaper details the key building blocks of the FELIX architecture, the main interface specifications and suggestions for potential developers implementing the architecture in real testing environment.

The remainder of this document is organized as follows: Section 3 describes key concepts considered in the FELIX experimental facility, Section 4 details the FELIX architecture components, Section 5 refers to the FELIX source code and its public availability while Section 6 presents where to find more information on FELIX and the next events during which our consortium has planned public demonstrations of the architecture described here..

2 Rationale for FELIX experimental testbed federation

A number of research testbeds have been built over the past year to stimulate Future Internet research. Three of them are particularly relevant on the SDN area: OFELIA, FIBRE and GridARS.

The ***OFELIA*** project [12] established a pan-European experimental network facility which enables researchers to experiment with real OpenFlow-enabled network equipment and to control and extend the network itself in a precise and dynamic fashion. The OFELIA facility uses the OpenFlow protocol (and related tools) to support virtualization and control of the network environment through secure and standardised interfaces. Ten interconnected sites form a diverse OpenFlow-based infrastructure that allows experimentation on multi-layer and multi-technology networks. A key objective is to provide experimental facilities which allow for the flexible integration of test and production traffic by isolating the corresponding traffic domains inside the OpenFlow-enabled network equipment. This creates realistic test scenarios and facilitates the seamless deployment of successfully tested technology into the real-world. OFELIA consists of two layers. The physical layer is comprised of the computing resources (servers, processors) and network resources (routers, switches, links, wireless devices and optical components). Resources are managed by the OFELIA Control Framework (OCF). Furthermore, the control framework layer contains components which manage and monitor the applications and devices in the physical layer. Aggregate Managers and Resource Managers are

crucial components of this layer, which can be seen as the combination of three components: Expedient is the GUI and allows the connection and federation with different Aggregate Managers via its plugins; Aggregate Managers (AMs) enable experimenters to create both compute and network resources via the VT AM and OF AM respectively; Resource Managers directly interact with the physical layer, provisioning compute resources (OFELIA Xen Agent) or flow rules to establish the topology (FlowVisor).

The ***FIBRE*** project [13] federates testbeds distributed across Europe and Brazil. The FIBRE-EU system builds on top of the OFELIA OCF and incorporates several wireless nodes based on commercial Wi-Fi cards and Linux open source drivers. On the other hand, the FIBRE-BR testbed includes nine Brazilian partners interconnected using private L2 channels. The VLANbased L2 physical link between Europe and Brazil is provided by GÉANT, Internet2 and RedCLARA. Unlike OFELIA, the FIBRE infrastructure is managed by different types of control and monitoring frameworks (CMFs). Indeed, FIBRE includes and enhances testbeds from other projects like OFELIA, OMF and ProtoGENI, which have been modified with the necessary software components to align their northbound interface to a common specification. The FIBRE project has opted to have two top-domain authorities, one in Brazil and one in Europe, to manage and own resources in the respective continents. These inter-connected authorities interoperate to allow the federation of BR and EU testbeds. The FIBRE architecture is composed of several multi-layer building blocks which are briefly summarized next. The SFA Registry is a database able to store the information related to users and projects, and to manage the certificates provided by the authorities. The MySLICE tool is used as the graphical (web) user interface for administration and experiment management. The SFA gateway is designed to translate the user's requests to a common API and provide slice management functions. FIBRE reuses and enhances the Aggregate Managers (AM) previously developed in OFELIA related to OpenFlow (OPTIN AM) and Xen-based (VT AM) resources and introduces a new AM to manage optical switches (ROADM) devices.

In Japan, ***GridARS*** [15] provides a reference implementation of the Open Grid Forum (OGF) Network Services Interfaces Connection Service (NSI-CS) protocol standard [4]. NSI is a web service-based interface for reserving, provisioning, releasing and terminating a network service, such as an end-to-end connection, via a two-phase commit protocol. GridARS can coordinate multiple resources (services), such as a network connection, virtual machines and storage spaces, via the NSICS protocol. It provides experimenters a virtual infrastructure, which spans several cloud resources, realised by multiple management domains including commercial solutions. GridARS consists of three main components. First, the Resource Management Service (RMS) is based on NSI-CS and consists of Global Resource Coordinators (GRCs) and Resource Managers (RMs) for Computers (CRM), Networks (NRM), and Storage (SRM). Coordinating with GRCs and RMs, RMS can coordinate heterogeneous virtual resources on multiple cloud environments. GRC has a co-allocation planning capability, which determines a suitable resource allocation plan. Second, the Distributed Monitoring Service (DMS) allows experimenters to monitor the virtual environment allocated to them. DMS does not have a central database, but gathers distributed monitoring information, tracking the hierarchical RMS reservation tree using the reservation ID, automatically. DMS consists of Aggregators (DMS/A) and Collectors (DMS/C). Each DMS/A gathers monitoring information from related DMS/As or DMS/Cs distributed over multiple domains, and provides the information to the requester. Each DMS/C monitors the reserved resources periodically, filters the monitoring information according to the domain policy, and provides the requester with the authorized information. Finally, the Resource Discovery Service (RDS) collects static resource information items from each resource domain and provides the aggregated information. The RDS implementation is based on the Catalog Service Web (CSW), defined by Open Geospatial Consortium (OGC) as an online XML-based database. Each resource domain can POST its static resource information, such as network topology, number of VMs, and storage spaces.

This brief review of research experimentation infrastructures with SDN control tools indicates that either they have been based on the OCF or on NSI framework but never together, thus limiting the flexibility of testbed interconnections at geographical scale.

The joint EU-Japan collaboration project **FELIX aims to bridge this gap and develop for the first time a testbed federation that successfully spans across these technological and geographical boundaries and incorporates both approaches in a recursive and scalable model.**

3 Key FELIX system concepts and definitions

The foundation of the FELIX experimental facility consists of the key system concepts summarized in this section.

FI experimental facilities (or SDN-controlled network domains) are controlled by dedicated software, exposing interfaces that can be used by a federation framework to orchestrate resources in a multi-domain environment. The SDN-controlled network domains are illustrated in Figure 1.

An ***SDN island*** is a set of virtualized network and computing resources under the same administrative ownership and control. It may consist of multiple SDN zones, each characterized by a specific set of control tools and interfaces. Each SDN Zone is a set of resources grouped together by common technologies and/or control tools and/or interfaces, e.g. L2 switching zone, optical switching zone, OpenFlow protocol controlled zone, and other transit domain zones with a control interface. The major goal of defining SDN zones is to implement appropriate policies for increasing availability, scalability and control of the different resources of the SDN islands. Examples of zone definitions can be found in widely deployed Cloud Management Systems (CMS) such as CloudStack, where infrastructure is partitioned into regions, zones [10], pods, and so on. In addition, OpenStack offers infrastructure partitioning through availability zones and host aggregates [11].

Transit network domains use NSI to expose either automatically or on-demand control of the connectivity services and, optionally, exchange inter-domain topology information. On-demand interconnectivity with a specific granularity must be provided in order to federate resources that belong to distant experimental facilities. In FELIX, it is assumed that all experimental facilities will be interconnected with networks running NSI-compatible network controllers. The NSIv2.0 standard interface will be used as a means to orchestrate network resources for an experiment setup.

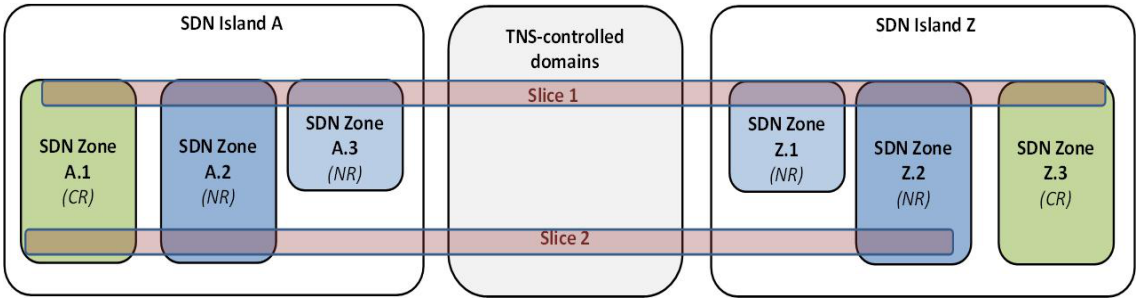


Figure 1 FELIX key concepts: transit domains, islands, zones and slices

In Figure 1, a ***slice*** is a user-defined subset of virtual networking and computing resources. Each slice is an abstraction created upon the set of physical resources available in the federated SDN Zones and SDN Islands. Every slice is isolated from other slices running simultaneously on the same physical resources, thereby avoiding interferences from other separate experiments. A slice should also be dynamically extensible across multiple SDN Islands. Each slice instantiates

only when needed the specific set of control tools required for the specific zones it must traverse.

The slice concept originates from the Slice-Based Federation Architecture (SFA) that defines a number of abstractions to identify provisioned resources, enable their aggregation and identify entities to manage resources (i.e. slivers, slices and component managers). SFA also provides a minimal set of structures and interfaces, which consist of two parts: 1) a specific data type per resource encapsulated in an RSpec to define, for example, a computing node, and 2) a list of methods following a specific workflow in order to reserve and provision any resource. These interfaces and data models facilitate and standardize the process of providing a federated slice composed of heterogeneous resources located in any other testbed in the federation. In order to do this, every SFA-compliant testbed must share the same interfaces and data models with its federated members to be able to understand resource requests. More information and samples can be found on the FELIX project implementation deliverables, available at [1].

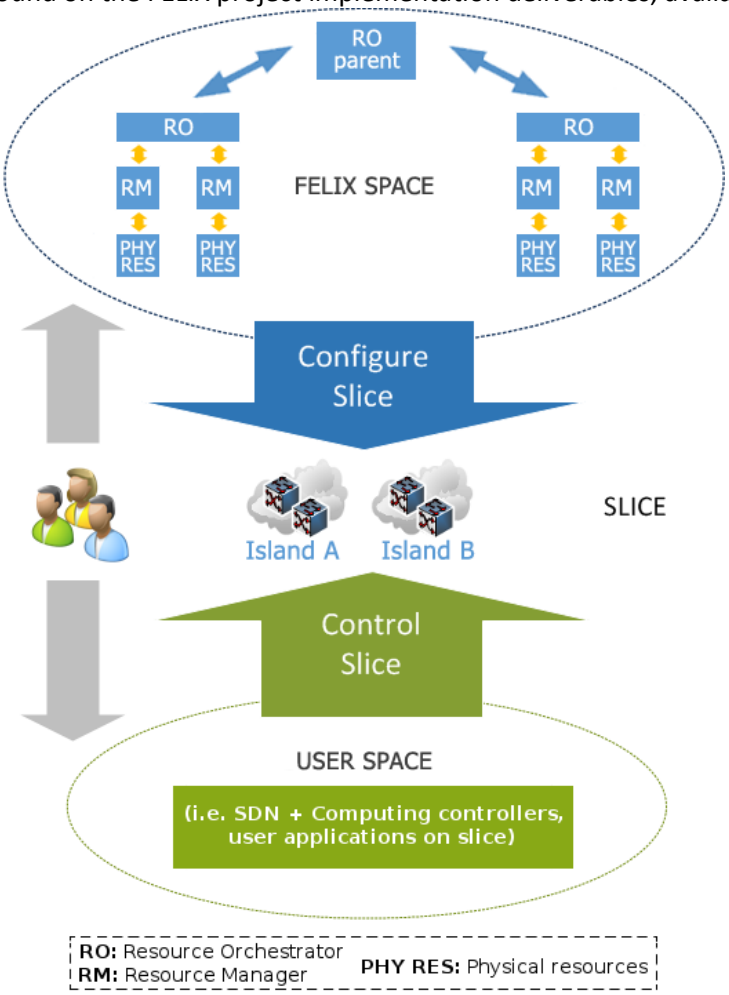


Figure 2 FELIX Architecture and Spaces

4 The FELIX Architecture

The FELIX architecture is the result of a careful analysis of the state of the art in relevant FI research projects in both Europe and Japan. From the European side, the OFELIA [12], FIBRE [13] and Fed4FIRE [14] projects were taken into account thereby providing a working approach to large-scale distributed systems and federation (e.g. through SFA and GENI), as well as addressing federation between heterogeneous testbeds. From the Japanese testbeds, GridARS [15] and RISE [16] were considered in order to manage seamlessly the establishment of dynamic inter-domain communication through the NSI protocol. Taken together, we have defined a modular

and multi-layer architecture for the FELIX control framework. As illustrated in Figure 2, we use the combination of two different 'spaces', namely the FELIX Space and User Space, which cooperate to build, manage, control, and monitor a large-scale virtual infrastructure.
The modules of the FELIX architecture address a number of common design requirements that are inherent and key to any successful federated Future Internet testbed.

- *Resource Orchestration*: This entity performs the necessary orchestration of the different types of virtualized resources present in the testbeds, such as compute, storage or network nodes.
- *Resource Allocation Planning*: This entity controls the reservation or allocation status for any resource subject to it. Both experimenter and administrator aspects are considered, for instance, allocation time desired by the experimenter or load balancing and complete provisioning cost for a given resource.
- *Provisioning*: The system must be able to instantiate and initialize the resources requested by the experimenter. The infrastructure at the different islands must provide applications with a virtual flat environment that behaves like a dedicated cluster and in which some specific user-space resource information, like IP addresses, are made available to the experimenter after proper instantiation and configuration tasks.
- *Domain Resource Management*: The heterogeneous resources that are provided by their corresponding resource management systems need to be managed accordingly, within that same domain.
- *Authentication, Authorization and Accounting*: In order to achieve a controlled environment where any action is authorized and can be traced back in detail, we need to ensure that a) an actor has a valid claim on the presented identity, b) any action is exclusively performed by an authorized actor and c) every action is tracked by the accounting system.
- *Monitoring*: The framework must provide the user with a coordinated set of monitoring data for both virtual and physical resources provisioned or located in different domains. This monitoring data is used primarily for the accounting of the infrastructure use, but also for enhanced control applications based on traffic measurements.
- *User Access*: The federated experimental facility provides friendly interfaces to simplify the operations of the experiment lifecycle for the user, as well as facilitating the general management of the testbed resources for the administrator.
- *Interoperability*: Interoperability is the key feature behind all architectural and development work in FELIX. Therefore project partners decided to implement the GENI architecture and specifically, GENIv3-compatible software interfaces of all FELIX components developed in the project. This decision implies FELIX Framework as such is fully compatible with other FI architectures by exposing standardized interfaces towards client applications (e.g. other testbeds or end users' applications).

4.1 Spaces in the FELIX architecture

As illustrated in Figure 2, we use the combination of two different 'spaces', namely the FELIX Space and User Space, which cooperate to build, manage, control, and monitor a large-scale virtual infrastructure.

The FELIX Space is composed of management and control tools that coordinate the creation of a virtual environment in heterogeneous, multi-domain, and geographically distributed facilities.

The elements in FELIX Space operate in both hierarchical and recursive models for efficient multi-domain information management and sharing.

In the FELIX Space, the Resource Orchestrators (ROs) are responsible for orchestrating the end-to-end network service and resources reservation in the entire infrastructure, as well as delegating end-to-end resource and service provisioning in a technology-agnostic way. ROs are connected to the different types of Resource Managers (RMs), which control and manage different kinds of technological resources similar to the concept of Component Manager in SFA. For example, the Transit Network RM and Stitching Entity RM provide connectivity between L1/L2 transport network domains and manage physical devices by using frame, packet, or circuit switching technologies; whilst able to support different protocols. The SDN RM manages the user traffic environment and the network infrastructure, composed of SDN-enabled devices, by updating the flow tables of the physical devices. In addition, the Computing RM is responsible for setting up and configuring computing resources, i.e. creating new virtual machine instances, network interface card configuration, etc. Moreover, the FELIX Space can provide essential functionalities to the FELIX architecture using dedicated modules such as the Authentication, Authorization and Accounting (AAA) for authenticating and authorizing users, or the Monitoring Functions module to retrieve, aggregate and store metering information from networking and computing resources to be used as feedback in the experiments.

The FELIX User Space is composed of any tool or application a user wants to deploy to control his or her virtual network environment or to run a particular experiment within it.

In the User Space, the Slice Controller can dynamically control the physical and virtual resources belonging to the user's slice environment. In other words, it can request more bandwidth, virtual CPU or RAM, add new resources such as storage, or even completely reconfigure the slice behaviour.

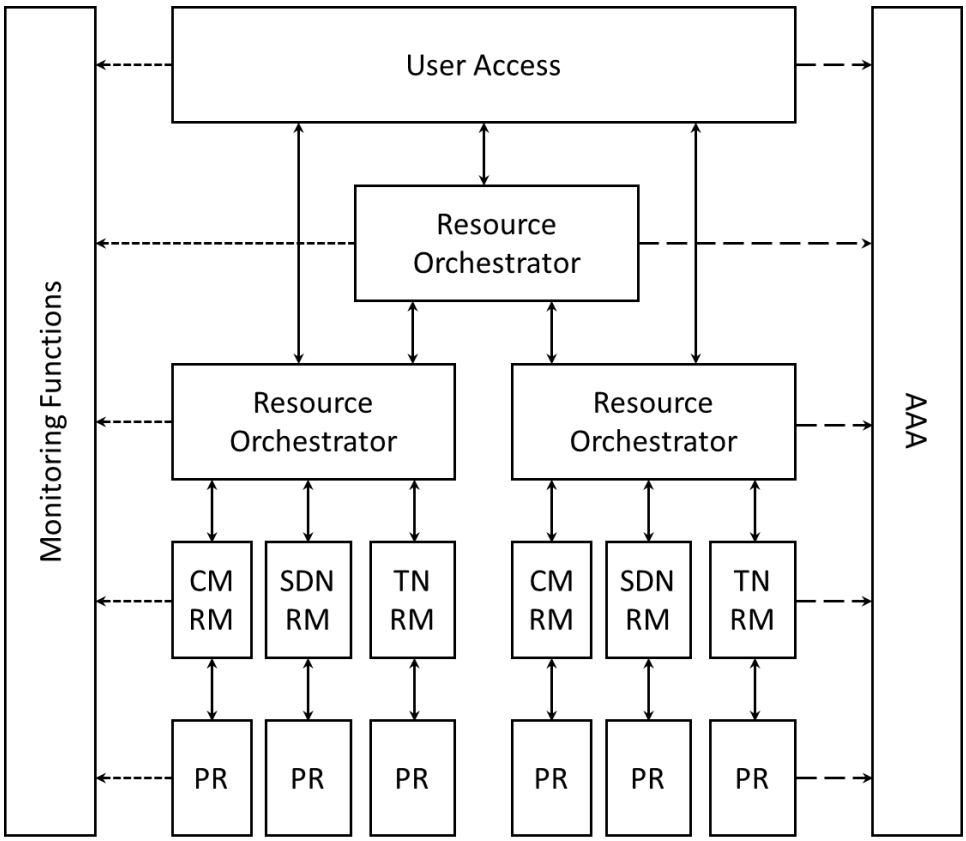
The two FELIX logical spaces glue together splitting the infrastructure management and slice control responsibilities, as shown in Figure 2.

4.2 FELIX Architecture components

The FELIX Architecture is composed of a number of modules that implement the functionalities identified in the common design considerations. These modules or ‘building blocks’ are intended to be as generic as possible in order to deal with different environments. Figure 3 shows a schema with the different modules and the interactions between them.

Resource Orchestrator (RO)

The FELIX RO is a key element of the FELIX architecture and the cornerstone of the management and orchestration system design. We consider that the RO operates over a federated testbed infrastructure of SDN ‘islands’, which are interconnected by Transit Networks subject to dynamic configuration through the Network Services Interface (NSI). The RO module is responsible for orchestrating the end-to-end network service as well as for instructing the resource reservation and provisioning for the entire FELIX infrastructure.



AAA: Authentication, Authorization & Accounting
CR: Computing Resource
PR: Physical Resource
RM: Resource Manager
SDN: Software Defined Network
TN: Transit Network
SE: Stitching Entity

Figure 3 Building blocks of the FELIX Architecture

There are two different levels at which the RO may operate: a) the upper layer, right below the User Access level, and b) the layer immediately underneath. In a typical scenario, the RO in the upper layer can operate at continental level, whilst the ROs in the lower layer may communicate with the RMs or with other ROs.

The main functionalities of RO are as follows: i) proxying requests between the experimenter and the RMs, ii) recursively delegating requests between ROs of other federated infrastructures according to pre-defined policies, iii) maintaining an updated and aggregated topological view of its managed, underlying infrastructures, and iv) verifying proper workflow and notifying the experimenter of any detected error condition.

Request forwarding for allocating and provisioning resources is performed in a technology-agnostic way within the infrastructure and depends on the conditions defined in the federation policy engine previously configured for the particular domain. It is therefore necessary to ensure similar interfaces for each orchestrator. RO must also interpret the experimenter's request to be able to perform any request forwarding as well as evaluate the set of actions received from the user for correctness and notify the user of error conditions.

An internal view of the cross-island topology (for computing, SDN and NSI nodes) is necessary for the RO to properly forward requests, for instance by detecting where computing

resources can be provisioned to meet the experimenter's requirements. Such topological information is filled from the lower layers (its scope is the resource management) to the upper layers, where this abstraction and mapping takes place. This resource discovery procedure minimizes the data being transmitted by taking advantage of the data interchanged between the modules during the expected workflow for SFA testbeds.

Transit Network Resource Manager (TN RM)

The TN RM enhances the FELIX architecture with mechanisms for network connectivity within and between particular domains. In order to deliver the network services in the FELIX architecture, the TN RM must be integrated with its southbound interfaces within a particular network domain. Such a domain can use different L1/L2 technologies and may be controlled by specific interfaces, systems such as the Network Management System (NMS), or protocols that are technology-dependent and unique in each case.

A single TN RM must communicate with a single RO in order to i) advertise resources under its control, ii) receive requests, and iii) notify the RO about success and failure events. A single TN RM is responsible for a group of particular network resources, which belong to a network domain and are usually managed by a single entity, i.e. a network administrator or NMS.

TN RM usually manages L1/L2 transport networks that are composed of physical devices using frames/packets or circuit switching technologies and support different protocols, e.g. MPLS/GMPLS. In order to support inter-island connectivity between existing OFELIA islands in Europe realized with VPN services over the Internet, the TN RM also supports the management of VPN set up and tear down procedures.

In the FELIX architecture, the TN RM southbound interface is based on the NSI-CS protocol for L1/L2 transport networks, a proprietary interface for L3 VPN services, whilst the northbound interface uses SFA-based APIs that can be understood by the RO.

Stitching Entity Resource Manager (SE RM)

The SE RM is a software element of the FELIX architecture that controls the Stitching Entity (SE), a network element providing the necessary translation mechanisms for a slice setup on top of the L2 protocol stack. SE RM hides from a user the actual complexity of the multi-domain transport network topology. SE must provide at least one of the following network functions: i) QinQ, to encapsulate slice traffic into transport network Ethernet frames, or ii) a VLAN translation mechanism to hide from users the actual VLAN tagging, which is used by carrier networks while interconnecting two or more FELIX islands.

The SE RM communicates with a single RO to i) advertise an internal topology and capabilities of the SE under its control, ii) receive requests, and iii) notify the RO about success and failure events.

A single SE RM must be implemented in each FELIX island and is responsible for single or multiple SEs, which belong to a network domain and act as an entry point to the island infrastructure.

SDN Resource Manager (SDN RM)

The domains in FELIX provide SDN-enabled infrastructures, usually equipped with OpenFlow-enabled network devices. This provides experimenters with tools to control their network behaviour in a programmatic way, that is, by defining a set of flow rules through a software controller that communicates with the physical network devices. Each rule defines a matching condition (any OpenFlow header, such as VLAN, and a value to match against) as well

as an action; and can be either polled from the controller by the network device, or directly inserted into special tables within the latter.

These switches and routers are configured and controlled through the SDN RM. This module can interact with a special purpose controller (e.g. FlowVisor) that is able to proxy the OpenFlow packets to the corresponding user controller thus keeping each environment for experimentation properly isolated from others. The SDN RM also allows the administrator to observe the experimenter's set of rules (which define her 'virtual network') and grant or revoke it.

Computing Resource Manager (CR RM)

The CR RM provides experimenters with mechanisms to configure, instantiate, and operate on computing resources and gives administrators the means to manage and monitor them.

This module interacts with the underlying infrastructure (virtualization servers and associated hypervisors) through an agent that acts as a proxy between the FELIX Control Framework side and the hypervisor. Therefore, the agent is able to communicate with the hypervisor and perform the operations provided by the former, such as creating, deleting and changing the status of the machines; informing of the status of each operation; and synchronizing the status of such resources between the infrastructure and management layers.

Authentication, Authorization and Accounting (AAA)

AAA makes available the necessary mechanisms to authenticate and authorize users and provides accounting for the user actions. These actions may be used from any other module (see Figure 3).

The FELIX architecture implements a Certificate-Based Clearinghouse (CH) [17] that establishes the root of a trust chain. By using a certificate-based approach, the architecture has the flexibility to federate different SDN islands easily and allows verifying the identity and privileges of all actors in the FELIX architecture.

The CH comprises a set of related services supporting AAA operations and acts as a location to lookup information about members, slices and other available services in the testbed. These services are offered through the Member Authority for certificate and credentials management, the Slice Authority for slice registry and privilege-mapping against its members, the Project Service for experiment registry and role evaluation, the Logging Service for accounting purposes and finally the Service Registry to register the aforementioned services. These authorities – derive from SFA principals – and the services complement the processes of i) registration and management, ii) authentication and authorization, and iii) accounting.

For federating purposes, these services can be accessed via the Common Federation API [18], allowing thus compatibility with tools like OMNI and testbeds complying with the GENI Aggregate Manager API [18].

The AAA module is closely related to the User Access and is ultimately responsible for granting access to resources, as authentication and especially authorization procedures are invoked on many operations. It may also be extended through policies, which are a set of rules defined by administrators to tailor the upper-level control on resources and testbeds usage.

Monitoring Functions

The monitoring functions are responsible for retrieving monitoring data for heterogeneous resources (e.g. compute, SDN, TN nodes) in the diverse testbeds of the

federation, as well as aggregating and storing it. Monitoring data can be categorized in two types: facility monitoring and infrastructure monitoring.

The facility monitoring data encompasses the basic status information about the facility, such as the status of the servers' availability or network connectivity; and may also include the status of the functional components in the FELIX Control Framework, which is aggregated from RMs and ROs. This information can be offered through the user portal for both experimenters and administrators.

The infrastructure monitoring checks the status of the resources currently available or provisioned, as well as some data for the experiments. This includes availability of virtualization and networking hardware or other information such as uptime and resource usage statistics.

The slice monitoring developed in FELIX builds upon state of the art monitoring tools, as well as testbed specific (facility) monitoring tools. Among these tools we use SNMP and OpenFlow flow space monitoring statistics. Monitoring data, is provided not only graphically, but also via a Monitoring API so that the statistics can be directly used by control mechanisms to make routing decisions based on traffic measurements or link status. The use of monitoring data is key to implement reactive decisions in all use cases considered in the FELIX project, guaranteeing feedback to the Planning and Provisioning tools to ensure that the required slice resources are available and can be used for specific workloads.

User Access

The FELIX Graphical User Interface (GUI) offers intuitive access to the lifecycle management of an experiment for experimenters and general management operations for administrators. To do this, the User Portal communicates with the underlying modules of the FELIX architecture to use a subset of their functionalities to authenticate and authorize users, as well as track their actions and provide them with any requested resource management, operation or observation.

The experimenters are thus able to list the available resources, define a subset of resources and allocate or provision them, performs operational actions on the resources, retrieve a description of the resources made available to them as well as release the resources when no longer needed.

On the other hand, the administrators may configure and manage resources, define different types of policies, grant or revoke resource requests and monitor different sets of resources, among others.

5 The FELIX software for infrastructure federation

FELIX has developed most of the components of the architecture described above from scratch on top of some key artifacts previously delivered by the FP7 OFELIA project. In particular, the FELIX consortium has developed all the logic and interfaces of Resource Orchestrator and the various Resource Managers (CR RM, SDN RM, SE RM, and NSI RM) re-using the Aggregate Manager framework developed in FP7 OFELIA. Also some Graphical User Interface components from OFELIA based on Expedient will be re-used and extended to work with the new RO, RMs and monitoring agents.

The FELIX code sources are open to the public since January 2015 over GitHub, specifically under the I2CAT organization (https://github.com/dana-i2cat/felix). FELIX has leveraged the tooling already available in GitHub to setup an infrastructure to support the initial community of users and developers around FELIX.

Users can download the software and follow the installation instructions in the FELIX GitHub wiki (https://github.com/dana-i2cat/felix/wiki) in order to get a working FELIX implementation for Resource Orchestrator and the various Resource Managers. The

documentation on the GitHub wiki is intended to serve as complement to the detailed instructions and design explanations contained in the official FELIX deliverables. The open FELIX site on GitHub also features an issue tracker where users can submit bug or feature requests.

Developers can clone the FELIX repository and browse/edit the code using their favourite development environments.

The procedures for accepting contributions from external users are still not officially established within the consortium, being the major focus at the time of writing this whitepaper on realization of use cases and demonstration of the full framework stack.

The license set for this code is APACHE 2.0, which has been evaluated by all the partners as a good solution for the type of code to be published and its poten□al use in FIRE communities around the world.

6 More on FELIX

More information on FELIX status and hints for developers along with plans for showcases and events is available through project website (http://www.ict-felix.eu). It serves as main collector for testbed details, latest software releases (jointly with our GitHub repository), guidelines and support for software development and implementation for third-party testbed, and much more.

The project will also execute public demonstrations of its architecture at a number of relevant events in the next months, like: EUCNC 2015 (Paris-FR, July 2015), EWSDN 2015 (Bilbao-PT, Sept. 2015), ICT2015 (Lisbon-PT, Oct. 2015, under evaluation), SC'15 (Austin-TX, USA , Nov. 2015).

References

[1] FELIX Project website: http://www.ict-felix.eu

[2] L. Peterson, R. Ricci, A. Falk and J. Chase, Slice-based Federation Architecture (SFA) v2.0, July 2010.

[3] GENI Aggregate Manager API. <http://groups.geni.net/geni/wiki/GeniApi>.

[4] R. Krzywania, W. Bogacki, B. Belter, K. Pentikousis, T. Rothe, G. Carrozzo, N. Ciulli, C. Bermudo, T. Kudoh, A. Takefusa, J. Tanaka and B. Puype, *Experiment Use Cases and Requirements*, FELIX Deliverable D2.1, September 2013. Available at http://www.ict-felix.eu.

[5] R. Krzywania, W. Bogacki, B. Belter, K. Pentikousis, T. Rothe, M. Broadbent, G. Carrozzo, N. Ciulli, R. Monno, C. Bermudo, A. Vico, C. Fernandez, T. Kudoh, A. Takefusa, J. Tanaka and B. Puype, *General Architecture and Functional Blocks*, FELIX Deliverable D2.2, February 2014. Available at http://www.ict-felix.eu.

[6] G. Roberts, T. Kudoh, I. Monga, J. Sobieski, J. MacAuley and C. Guok, *NSI Connection Service v2.0*. May 2014. Available at http://www.ogf.org/documents/GFD.212.pdf.

[7] I. Monga, E. Pouyoul and B. Tierney, *Dynamic creation of end-to-end virtual networks for science and cloud computing leveraging OpenFlow/Software Defined Networking* in *Proceedings of the 2012 TERENA Networking Conference (TNC'12)*, May 2012.

[8] I. Monga, E. Pouyoul and C. Guok, *Software Defined Networking for big-data science*, in *Proceedings of the 2012 High Performance Computing, Networking, Storage and Analysis Conference (SCC'12),* November 2012.

[9] A. Sadasivarao, S. Syed, P. Pan, C. Liou, A. Lake, C. Guok and I. Monga, *Open Transport Switch: A Software Defined Networking Architecture for Transport Networks* in *HotSDN workshop in ACM Special Interest Group on Data Communication (SIGCOMM'13)*, August 2013.

[10]Apache CloudStack documentation. *Cloud Infrastructure Concepts*. <http://cloudstack.apache.org/docs/en-US/Apache_CloudStack/4.1.0/html/Admin_Guide/cloud-infrastructure-concepts.html>.

[11]Scaling Openstack. <http://docs.openstack.org/openstack-ops/content/scaling.html>.

[12]M. Suñé, L. Bergesio, H. Woesner, T. Rothe, A. Köpsel, D. Colle, B. Puype, D. Simeonidou, R. Nejabati, M. Channegowda, M. Kinde, T. Dietzf, A. Autenriethg, V. Kotronis, E. Salvadori, S. Salsano, M. Körner, S. Sharma, *Design and implementation of the OFELIA FP7 facility: The European OpenFlow testbed*, *The International Journal of Computer and Telecommunications Networking*, December 2013.

[13]T. Salmito, L. Ciuffo, I. Machado, M. Salvador, M. Stanton, N. Rodriguez, A. Abelem, L. Bergesio, S. Sallent, L. Baron, *FIBRE - An International Testbed for Future Internet Experimentation* in 32º *Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC'14)*, 2014.

[14]W. Vandenberghe, B. Vermeulen, P. Demeester, A. Willner, S. Papavassiliou, A. Gavras, M. Sioutiss, A. Quereilhac, Y. Al-Hazmi, F. Lobillo, F. Schreiner, C. Velayos, A. Vico-Oton, G. Androulidakis, C. Papagianni, O. Ntofon, M. Boniface, *Architecture for the Heterogeneous Federation of Future Internet Experimentation Facilities* in *Proceedings of Future Network & Mobile Summit Conference*, 2013.

[15]A. Takefusa, H. Nakada, T. Kudoh, Y. Tanaka and S. Sekiguchi, *GridARS: An Advance Reservation-based Grid Co-allocation Framework for Distributed Computing and Network Resources*, *Lecture Notes, Computer Science of the2008 Job Scheduling Strategies for Parallel Processing (JSSPP'08),* April 2008, vol.4942, pp. 152-168.

[16]*Y. Kanaumi, S. Saito, E. Kawai, S. Ishii, K. Kobayashi and S. Shimojo, RISE: A Wide-Area Hybrid OpenFlow Network Testbed,* IEICE Transactions on Communications, January 2013, vol. E96-B, no. 1, pp. 108-118

[17]U. Toseef, A. Zaalouk, T. Rothe, M. Broadbent and K. Pentikousis, *C-BAS: Certificate-based AAA for SDN Experimental Facilities* in *Proceedings of the 2014 European Workshop on Software Defined Networking (EWSDN'14),* September 2014.

[18]*Common Federation API for any GENI-compatible federation.* November 2013. <http://groups.geni.net/geni/wiki/CommonFederationAPIv2>.

[19]W. John, K. Pentikousis, G. Agapiou, E. Jacob, M. Kind, A. Manzalini, F. Risso, D. Staessens, R. Steinert and C. Meirosu, *Research Directions in Network Service Chaining* in *Proceedings of 2013 Future Networks and Services (IEEE SDN4FNS'13)*, November 2013.

[20]K. Pentikousis and P. Bertin, *Mobility Management in Infrastructure Networks*, *IEEE Internet Computing*, September 2013, vol.17, iss. 5, pp. 74-79.

[21]A. Qureshi, R. Weber, H. Balakrishnan, J. Guttag and B. Maggs, *Cutting the Electric Bill for Internet-scale Systems* in *Proceedings of the 2009 ACM Special Interest Group on Data Communication (SIGCOMM'09)*, August 2009.

[22]R. Takano, A. Takefusa, H. Nakata, S. Yanagita and T. Kudoh, *Iris: Inter-cloud Resource Integration System for Elastic Cloud Data Center* in *Proceedings of the 2014 International Conference on Cloud Computing and Services Science (CLOSER'14)*, April 2014.

Acknowledgements

This work was conducted within the framework of the EU-Japan FELIX project, which is partially funded by the Commission of the European Union and the National Institute of Information and Communications Technology (NICT), Japan. Study sponsors had no role in writing this report. The views expressed do not necessarily represent the views of the authors' employers, the FELIX project, the Commission of the European Union, or the National Institute of Information and Communications Technology (NICT), Japan.

PARTNERS

Poznan Supercomputing and Networking Center
Poland

National Institute of Advanced Industrial Science and Technology
Japan

Nextworks
Italy

Fundacio Privada i2CAT, Internet I Innovacio Digital A Catalunya
Spain

SURFnet bv
Netherlands

European Center for Information and Communication Technologies Gmbh
Germany

iMinds VZW
Belgium

KDDI
Japan

The scientific/academic work is financed from financial resources for science in the years 2013 - 2016 granted for the realization of the international project co-financed by Polish Ministry of Science and Higher Education.