Large-scale SDN Experiments in Federated Environments

G. Carrozzo* R. Monno* B. Belter† R. Krzywania† K. Pentikousis‡ M. Broadbent‡ T. Kudoh§
A. Takefusa§ A. Vico-Oton¶ C. Fernandez¶ B. Puype‖ J. Tanaka**
*Nextworks s.r.l. †PSNC ‡EICT §AIST ¶i2CAT ‖iMinds **KDDI

Corresponding Author: g.carrozzo@nextworks.it

***Abstract*—International cooperation on Software-Defined Networking (SDN), crossing the boundaries of Europe, the Americas and Asia, builds a strong foundation for pursuing experimental research through advanced programmable network testbeds. The EU-Japan jointly-funded project FELIX (FEderated Testbeds for Large-scale Infrastructure eXperiments) considers the definition of a common framework for federated Future Internet (FI) testbeds, which are dispersed across continents. This framework will enable an experimenter to i) request and obtain resources across different testbed infrastructures dynamically; ii) manage and control the network paths connecting the federated SDN testbed infrastructures; iii) monitor the underlying resources; and iv) use distributed applications executed on the federated infrastructures. This paper details six use cases that will be employed to validate the FELIX architecture and software platforms. We present our analysis and end-user considerations, highlighting the necessity to have a global vision of issues within the testbed network. Resource reachability and coherent use of physical connections are key factors in the use cases. This is particularly important when considering the simultaneous use of different technologies such as OpenFlow and the Network Service Interface (NSI) among others.**

I. Introduction

Programmable networks, based on Software-Defined Networking (SDN) principles, decouple the control from the data plane and allow for remote software to take over the control and management of the underlying network. Those networks have become a substantial part of existing Future Internet (FI) testbeds. In such testbeds, researchers around the world are interested in efficient, predictable, realistic, and reproducible environments which can be used to validate their proof-of-concept prototypes and experiment with new algorithms, protocols or network functions. By designing and implementing a suitable framework, the FELIX project (www.ict-felix.eu) aims to facilitate the federation and integration of different network and computing resources residing in a multi-domain heterogeneous environment across different continents. As we will see later in this paper, FELIX uses a hierarchical model for inter-domain dependency management, with resource orchestrating entities responsible for the synchronization of resources available in particular administrative domains.

This paper presents a number of scenarios that can be used for validating and demonstrating the FELIX framework. This will be done using a distributed SDN infrastructure consisting of multiple federated and geographically dispersed SDN testbeds. Taken as a whole, the FELIX federated resources create a virtual infrastructure which spans multiple domains. Six specific use cases are described in this paper, grouped into two major clusters: *Data Domain* and *Infrastructure Domain*.

The Data Domain use cases focus on the efficient use of SDN technologies to provide interconnections across geographically dispersed testbeds with the ability to realize data migration dynamically and efficiently. The Infrastructure Domain use cases are mainly oriented towards the use of a virtual distributed infrastructure which can be employed to migrate entire data processing workloads. This paper reports early-phase work focusing on use-case identification [1] and architecture definition [2]. Future work will address the validation of these use cases on the FELIX federated infrastructure.

The remainder of this paper is organized as follows. Section II details the different resources considered in the FELIX experimental facility. Section III overviews the FELIX architecture and Section IV details the use cases considered. Finally, conclusions and future work are presented in Section V.

II. Resources in the FELIX Federated Testbed

Resources in FELIX include both networking and computing capacities available at geographically dispersed facilities. Resources are under the administrative control of different but cooperating stakeholders. Taken as a whole, the federated resources included in FELIX create a virtual infrastructure which spans multiple domains. Note that this environment is starkly different from the case of a) a single administrative domain with resources geographically distributed across the world, e.g. data centers of a single cloud operator; and b) loosely coupled, interconnected islands which allow for remote access to certain resources. FELIX is primarily interested on network enablers and, in particular, the integration of SDN testbeds with Network Service Interface(NSI)-controlled transit domains. These, in turn, can be used to solve the dynamic establishment and tear-down of network flows (based on L2 switching and L3+ routing/forwarding) across multiple domains and technologies.

A. Virtual Infrastructures through Federation

Monga et al. [3] note that connecting facilities at continental, let alone inter-continental scale, is not a trivial task. They motivate the need for connecting facilities (such as those considered in FELIX) at the lower layers (e.g. L2), thus avoiding the system overheads introduced by the connections established at L3 and above. However, the resulting proposals [3], [4], [5] do not conform to emerging standards, such as NSI [6]. Moreover, this work does not comprehensively consider the elements of each island from a network control perspective, and does not account for policies and trust. We believe that these aspects will play a crucial role in determining the adoption of a framework suitable for federated resources. Our analysis of the latest research literature on this topic has highlighted the need to introduce new APIs and logic for globally distributed heterogeneous facilities (e.g. OFELIA islands and JGN-X RISE testbeds). It is clear that these should capitalize on SDN and NSI mechanisms and protocols to facilitate the dynamic, on-demand establishment of end-to-end cross-continental virtual network infrastructures.

While SDN testbed infrastructures are constructed from the viewpoint of network research and development, it goes without saying that computing and storage resources are also important components in each testbed. FI services can be grouped into two categories: those that use network resources to move data, and those that use the whole infrastructure (including computing and storage resources) to provide network-based services. Therefore, we consider two major classes of use cases for the demonstration of virtual infrastructure based on federated testbed resources. Namely, the first category of use cases are in the *data domain* since the primary focus is the use of data. The second category of use cases are in the *infrastructure domain*, which includes all three resource types in a testbed: networking, storage and computing.

B. Key System Concepts and Definitions

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

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 Fig. 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 island. Examples of zone definitions can be found in widely-deployed Cloud Management Systems (CMS) such as CloudStack, where infrastructure is partitioned into regions, zones, pods, and so on [7]. In addition, OpenStack offers infrastructure partitioning through availability zones and host aggregates [8], [9].

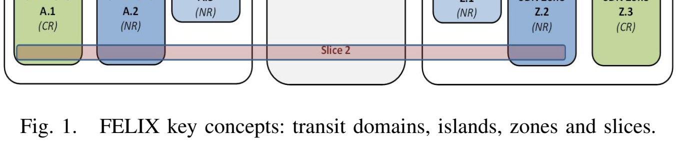

Fig. 1. FELIX key concepts: transit domains, islands, zones and slices.

Transit network domains use NSI to expose either automatically or on-demand the 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 belonging 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 [6] will be used as a means to orchestrate network resources for an experiment setup.

In Fig. 1, a *slice* is a user-defined subset of virtual networking and computing resources, created from the physical resources available in federated SDN Zones and SDN Islands. A slice is isolated from other slices running simultaneously on the same physical resources. It should also be dynamically extensible across multiple SDN Islands. Each slice instantiates the specific set of control tools required for the specific zones it must traverse.

III. The FELIX Architecture

The concepts introduced in the previous section are used in the remainder of this paper to define a modular and multi-layer FELIX architecture. As illustrated in Fig. 2, we use the combination of two different "spaces", i.e. 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 which coordinate the creation of a virtual environment in the heterogeneous, multi-domain and geographically distributed facilities. The elements that belong to this layer operate in a hierarchical model for an efficient multi-domain information management and sharing. The User Space is composed of any tool or application a user wants to deploy to control his virtual network environment or to run a particular experimentation within it. These two logical spaces glue together different functional building blocks, as shown in Fig. 2.

In the FELIX Space, the *Resource Orchestrators* (ROs) are responsible for orchestrating the end-to-end network service and resources reservation in the whole infrastructure. Moreover, ROs should be able to delegate end-to-end resource and service provisioning in a technology-agnostic way. ROs are connected to the different types of *Resource Managers* (RMs), which are in turn used to control and manage different kinds of technological resources. For example, the *Transit Network RM* provides the connectivity between L1/L2 transport network domains and manages physical devices. This management can be achieved using either frame, packet or circuit switching technologies and should support different protocols.

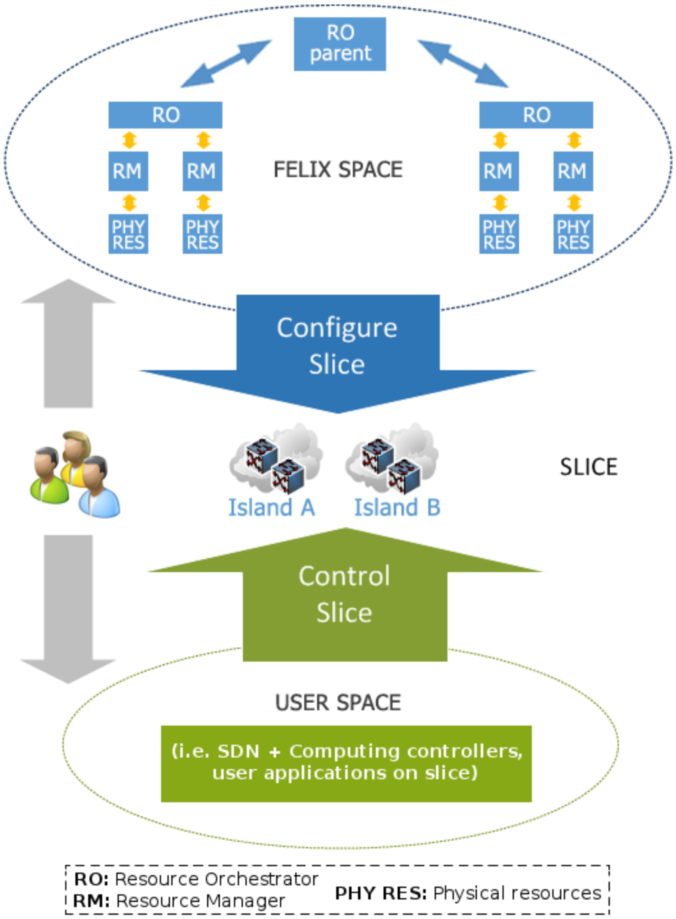

Fig. 2. FELIX Architecture and Spaces

On the other hand, the *SDN RM* manages the network infrastructure composed of SDN-enabled devices, e.g OpenFlow switches or routers. In short, it can control the user traffic environment 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, powering on/off instances, network interface card configuration, etc. Moreover, the FELIX Space can provide some basic functionalities to the FELIX architecture using dedicated modules such as the Authentication and Authorization Infrastructure (AAI) for authenticating and authorizing users, or the Monitoring Functions module to retrieve, aggregate and store metering information.

In the User Space, the *Slice Controller* can dynamically control the physical and virtual resources which belong 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 to completely reconfigure the slice behavior.

The FELIX architecture is the result from a careful analysis of all relevant state of the art in FI projects. In the remainder of this section we briefly discuss advantages and disadvantages, mainly focusing on what is either missing or not fully covered in their proposed design and offered services.

The OFELIA (www.fp7-ofelia.eu) and FIBRE (www.fibre-ict.eu) projects are deeply inspired by SFA concepts and architectures to offer a federation of physical resources, allowing different clients to have direct access to the Resource or Aggregate Managers. This can be viewed as a major issue for scalability within large-scale distributed architectures. Moreover, in these testbeds there is no concept of *Resource Orchestrator* and some configuration or provisioning operations require either human intervention or authorization and authentication policies.

The BonFIRE project (www.bonfire-project.eu) is mainly focused on cloud computing, giving priority to computing rather than network resources. This can result in a lack of a *slice* concept and can increase the complexity for a federation approach. However, it provides dynamic network parameter configuration (i.e. latency) and offers Bandwidth on Demand (BoD) services through AutoBAN.

The FED4FIRE project (www.fed4fire.eu) is a federation of heterogeneous testbeds, mainly focused on how to effectively offer the different services of the testbeds, without a centralized or distributed logical control plane. Furthermore, the network connections between testbeds are fixed and cannot be manipulated using a dedicated system.

Finally, the GridARS [10] and RISE [11] projects introduce the NSI protocol to manage the inter-domain network segment within a dedicated generic developed framework. Unfortunately, these projects seem to be SFA-agnostic increasing the effort to federate resources with them.

Key differentiators of FELIX with respect to the aforementioned FI tesbeds and infrastructures are covered in the reference architecture; which is intended to jointly support both network and computing resources, and in the seamless interaction with NSI-controlled transit domains. The latter is considered a key innovation step towards the dynamic user-controlled construction of highly distributed virtual infrastructures across continents.

IV. The FELIX Use Cases

As mentioned previously, the FELIX usage scenarios are clustered into two groups: Data Domain and Infrastructure Domain use cases. The Data Domain use cases in FELIX consider a virtual infrastructure that focuses on the efficient use of SDN for dynamically and efficiently interconnecting geographically dispersed testbeds across two continents. In the case of the Data Domain, use cases include virtual infrastructure consisting of SDN islands interconnected with dynamic circuit-switched (inter-continental) networks. One important goal is to optimize the use of interconnectivity between testbeds to realize data migration. On the other hand, the Infrastructure Domain use cases describe user scenarios based upon federated resources; placing emphasis on the optimized use of the infrastructure as a whole. This includes the migration of entire workloads during data processing operations.

A. Data Domain Use Cases

Data Domain use cases are primarily oriented towards the efficient utilization of the physical network by taking advantage of SDN and NSI operations for the dynamic interconnection of testbeds dispersed across different continents. The focus here is on the coordination of caching, processing and network services rather than on the exact caching algorithms to be used, which are in the full scope of user priorities and control. The testbeds for these use cases form a virtual infrastructure which consists of SDN islands (L2 domains) interconnected with dynamic circuit-switched networks (multi-domain transit networks). In this large-scale facility, data must be transferred from the origin to its destination end-point, typically in another SDN island. The following subsections summarize each use case and explain how the aforementioned flows of data traverse a real network.

Data-on-Demand: delivery of distributed data by setting data flows over the network. This use case investigates how to process large amounts of data stored in different and distributed sites. For instance, several applications, such as astronomical observations or collaborative investigations, generate huge amount of data which are typically stored in dedicated storage servers or devices in a nearby data center. An application or user, i.e. a data processor, may want to run a post-processing algorithm on the data collected by different data providers. In this context, it is not suitable nor efficient to move the data from the original sites to the final location. On the contrary, it could be more convenient to install a (SDN-based) network controller, which shall have a global view of the whole network topology. This controller could automatically establish links between different end points. It can also be used to guarantee the reliability of the end-to-end communication (with minimum delay and jitter, as needed). In such a way, the optimal use of the physical network resources could be achieved. Fig. 3 depicts the main components of the scenario, the relevant actors and their potential interactions.

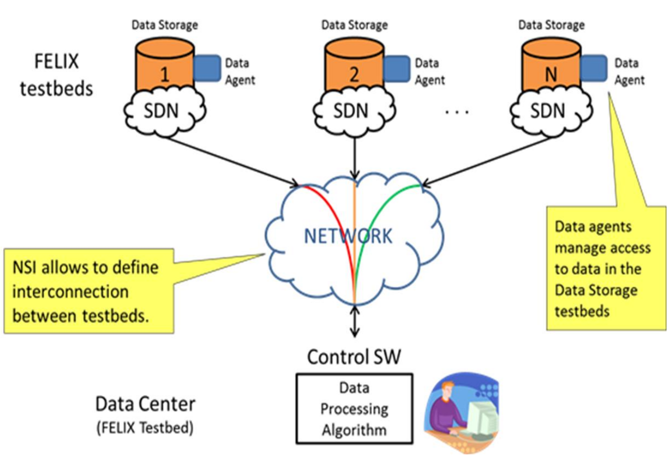

Fig. 3. Data-on-demand: distributing data via programmable data flows

Data preprocessing for minimizing network latency effect for live data. This use case aims to provide near real-time data, e.g. satellite images, to users located in different and distant places without incurring in the large Round Trip Delay (RTD) values typically found with transfers through the public Internet. In these situations, a dedicated platform would be placed near the receiver station and perform a suitable preprocessing of the data. This platform could be able to allocate computing, caching and networking resources at both source and destination islands. It could also be able to implement on-demand and application-driven network services for the specific data transfers, which require well-defined network parameters. Consequently, this approach can significantly reduce the size of data to be delivered across the transit network and improve the overall system performance. Fig. 4 presents an overview of this scenario.

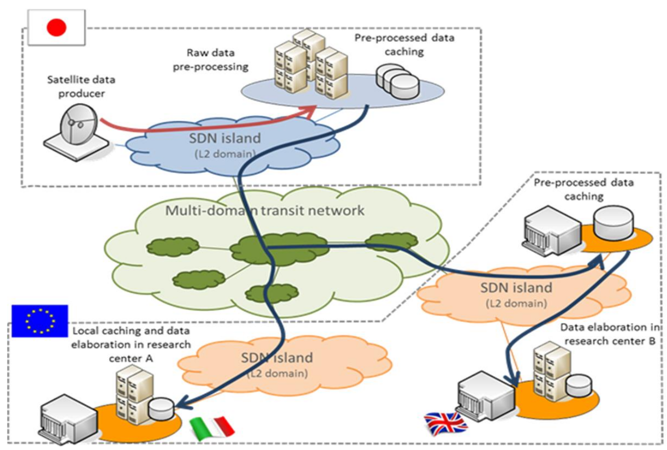

Fig. 4. Data preprocessing: minimizing network latency effect for live data

High-quality media transmission over long-distance networks. In the last few years we are experiencing a rapidly evolution of the media content delivery, especially in the context of the ultra-high definition of the video streaming, i.e. 4K and 8K resolution. This evolution directly relates to a higher quality of the media playback, but also imposes higher bandwidth and lower delay constraints on the network. In this scenario, illustrated in Fig. 5, hardware optimization is required for the transmission and reception of the data content, especially in a very long-distance environment.

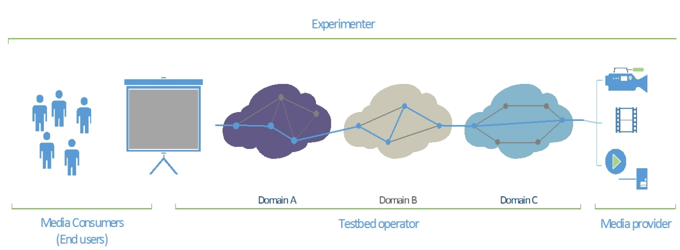

Fig. 5. High-quality media transmission

At the same time, network streamlining is needed both in the transport (NSI-enabled) segments and in the inter-datacenter networks (SDN-enabled). In this use case, all the defects of poor management and control of the network will manifest in visible playback artefacts: jitter, incorrect frame sequencing, transmission disruption, etc. Moreover, strict requirements are necessary to serve 3D video to the user, as two flows have to be delivered separately for left and right eye. In this scenario, proper synchronization is extremely important to achieve a satisfactory quality of service. This is measured through Quality of Service (QoS) and Quality of Experience (QoE) metrics.

B. Infrastructure Domain Use Cases

The Infrastructure Domain use cases are mainly concerned with the services and workloads which can be facilitated by a software platform built on top of the federated resources. It is important to note that both the Infrastructure and Data Domain use cases share common architectural, trust and security assumptions. In the Infrastructure Domain use cases, we consider network, computing and storage resources which can dynamically migrate over the allocated physical environment. This work is inline with recent developments in leading standardization fora, such as IETF and ETSI, where significant attention has been drawn from both industry and academia towards network service chaining and the ability to relocate network functions, infrastructure scale-out and scale-up, as well as continuous service delivery [12]. The remainder of this section introduces the Infrastructure Domain use cases and explain how the services can be deployed in a large-scale facility, such as FELIX.

Inter-cloud use case: data mobility service by SDN technologies. This use case focuses on cloud systems and the services provided by them in carrier-grade, mission-critical areas. This includes electronic administration, medical care and finance. To satisfy the requirements, these complex cloud systems should meet demands of an end-to-end guaranteed service quality, reliability of compliance and energy efficiency. In this context, every single-cloud system is limited by its available resources. This limit can be easily exceeded with a flexible reassignment of resources belonging to different cloud systems. Therefore, it is important to establish a cooperation between data centers, at least on a temporary basis.

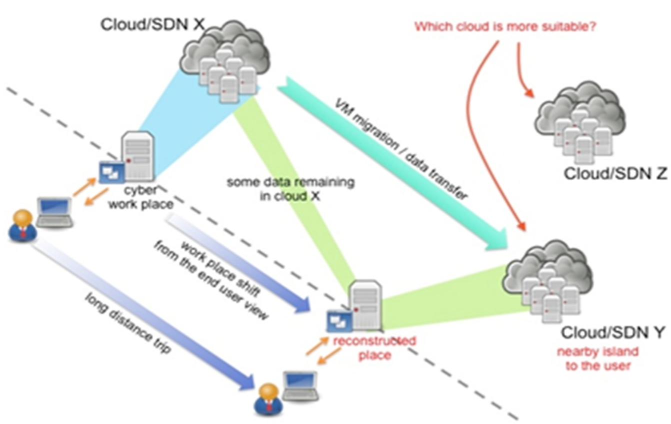

Fig. 6. The inter-cloud data mobility

For example, consider a user who moves to a remote location due to a business trip. The user wants to to use his or her services, in this case based in the cloud, with the same level of quality of experience as if they where using local resources and on par with the experience they have in their home network. Note that in this case, traditional mobility management solutions [13] would not be able to mitigate the expected large propagation delays between the present user location and the data center processing the user's workload. Instead, the scenario illustrated in Fig. 6 points to the fact that it would be preferable to transfer user data (such as credentials, applications and services) to a cloud system closer to his/her visiting place.

Follow the sun (or moon) principles. As detailed in [14], Internet usage curves follow a similar daily pattern everywhere in the world, and there is a natural shift in the load of data centers to places in the world where it is currently daytime. The opposite is true during the night, when data centers are under a different amount of load. This is often referred to as the "follow the sun/moon" principle. Moreover, the prices of renewable energy strongly depend on the availability of wind and solar energy (green energy). As a result, several data centers are moved in locations such as Iceland and Finland and perhaps in the future in desert areas.

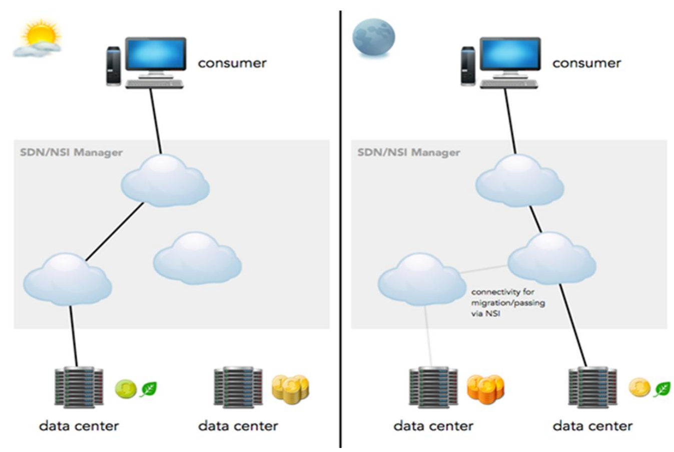

Fig. 7. The "follow the sun-moon" use case

In this case, one could shift the load of one data center to another one following two different approaches (Fig. 7): a) move the entire workload to a more efficient data center basically with a re-routing of the user's traffic, or b) handle the user's requests at the less efficient data centers by delegating the work-flow to more efficient data centers. It is important to note that both scenarios require dynamic and on-demand end-to-end connections between the federated data centers. Moreover, when the workload is moved from one data center to another, a number of different resources (network, compute and storage) need to be configured accordingly.

Disaster recovery by migrating IaaS to a remote data center. This use case is inspired by the Business Continuity Planning (BCP) of services which are key to cloud providers. This is particularly pertinent after the experience of the great east Japan earthquake in 2011. Typically, the cloud systems are managed by Infrastructure as a Service (IaaS) software, such as OpenStack or CloudStack, and provide isolated tenants on physical resources (computers, storage and network) in a data center with multiple IaaS users. A stable and fault-free environment is expected by these users, but under particular conditions, such as a serious disaster, it can be difficult to continue providing the desired services. In such a case, middleware can assist in enabling the migration of the cluster of servers and virtual machines to a remote data center and guarantee business continuity. Another influence used in creating this use case is the Hardware as a Service (HaaS) paradigm, [15], which can dynamically configure virtual IaaS-enabled resources using nested virtualization technologies (e.g. KVM and FlowVisor). These resources can be migrated on the HaaS layer of another data center, as depicted in Fig. 8, coordinating the configuration of the hypervisor resources with the network bandwidth constraints to allow a fast and efficient migration of the IaaS instance from one site to the other.

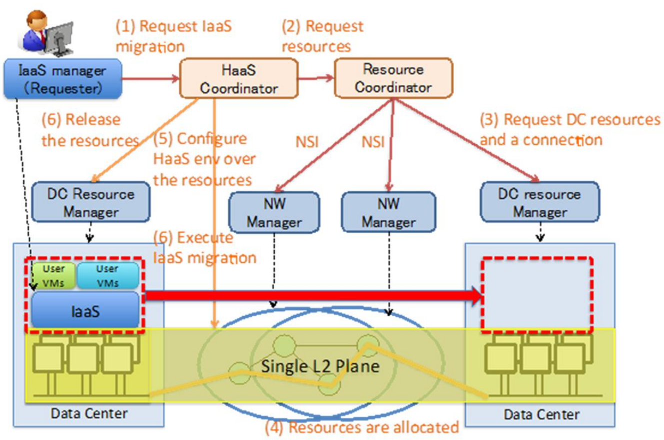

Fig. 8. Disaster recovery through IaaS migration in remote data center

V. Conclusions and Future Work

We presented six use cases for large-scale SDN experiments over cross-continental federated environments. We grouped the set of use cases into two major categories, namely the Data Domain and Infrastructure Domain, in order to better reflect their primary applicability area and stakeholders. These scenarios highlight the necessity to have a single management and control of the intra- and inter- connectivity for the data centers. We believe that they can be considered as a foundation for the development of complex architectural models and software platforms which can manage resources in more efficient ways.

The FELIX Data Domain use cases mostly target the application area of SDN and dynamic interconnections via NSI to improve and innovate both data transfers and consumption among testbeds dispersed across different continents. Data caching, fast delivery, streaming and the related workflow management are key in this group of use cases. The FELIX Infrastructure Domain use cases focus more on the efficient use of federated and dispersed FI resources across different continents, the ability to migrate entire workloads (VMs and data) or infrastructures in a more efficient way (e.g. with energy saving targets) and enhanced features (e.g. data/service survivability in case of disasters).

From the users' perspective, all presented use cases apply to the same and unique FELIX framework architecture, which has to include the common system functionalities derived from specific use cases and users' goals. The current list of use cases are not meant to be exhaustive. In fact, they represent a structured set of the initial outcomes obtained through early project activities and preliminary input to the FELIX architecture definition.

The work in the FELIX project is proceeding towards the consolidation of the architecture of the FELIX system. As part of our future work, we aim to continue the development of the software components introduced in this paper, and validate them using the use cases presented in this paper.

Acknowledgment

This work has been conducted within the framework of the EU-FP7/JP-NICT FELIX project, which is partially funded by the European Commission under grant agreement no. 608638 and the National Institute of Information and Communications Technology (NICT), Japan.

References

[1] R. Krzywania, et al., *Experiment Use Cases and Requirements*, FELIX Deliverable D2.1, September 2013. Available at http://www.ict-felix.eu.

[2] R. Krzywania, et al., *General Architecture and Functional Blocks*, FELIX Deliverable D2.2, February 2014. Available at http://www.ict-felix.eu.

[3] I. Monga, et al., "Dynamic creation of end-to-end virtual networks for science and cloud computing leveraging OpenFlow/Software Defined Networking", *Proc. TNC*, May 2012.

[4] I. Monga et al., "Software Defined Networking for big-data science", *Proc. SCC*, November 2012.

[5] A. Sadasivarao, et al., "Open Transport Switch: A Software Defined Networking Architecture for Transport Networks", *Proc. HotSDN*, August 2013.

[6] G. Roberts, et al., *NSI Connection Service v2.0*. [Online]. Available: http://redmine.ogf.org/attachments/135/NSI%20Connection%20Service%20Protocol_20_draft25.pdf. Last visited: 20 May 2014.

[7] Apache CloudStack documentation (online). *Cloud Infrastructure Concepts*. Available: http://cloudstack.apache.org/docs/en-US/Apache_CloudStack/4.1.0/html/Admin_Guide/cloud-infrastructure-concepts.html. Last visited: 20 May 2014.

[8] OpenStack documentation (online). *Scaling*. Available: http://docs.openstack.org/trunk/openstack-ops/content/scaling.html. Last visited: 20 May 2014.

[9] OpenStack design references. (online). *MultiClusterZones*. Available: https://wiki.openstack.org/wiki/MultiClusterZones. Last visited: 20 May 2014.

[10] A. Takefusa, et al., "GridARS: An Advance Reservation-based Grid Co-allocation Framework for Distributed Computing and Network Resources", *Proc. JSSPP*, LNCS vol. 4942, April 2008.

[11] *RISE project website*. [Online]. Available: http://www.jgn.nict.go.jp/rise/english/index.html. Last visited: 19 May 2014.

[12] W. John, et al., "Research Directions in Network Service Chaining", *Proc. IEEE SDN4FNS*, November 2013.

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

[14] A. Qureshi, et al., "Cutting the Electric Bill for Internet-scale Systems", *Proc. ACM SIGCOMM*, August 2009.

[15] R. Takano, et al., "Iris: Inter-cloud Resource Integration System for Elastic Cloud Data Center", *Proc. CLOSER*, April 2014.

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.