

# Network Virtualization: The missing piece

Karsten Oberle, Marcus Kessler, Manuel Stein,  
Thomas Voith

Alcatel-Lucent Bell Labs  
70435 Stuttgart, Germany

[karsten.oberle@alcatel-lucent.de](mailto:karsten.oberle@alcatel-lucent.de), [marcus.kessler@alcatel-lucent.de](mailto:marcus.kessler@alcatel-lucent.de),  
[manuel.stein@alcatel-lucent.de](mailto:manuel.stein@alcatel-lucent.de),  
[thomas.voith@alcatel-lucent.de](mailto:thomas.voith@alcatel-lucent.de)

Dominik Lamp, Sören Berger  
Rechenzentrum Universität Stuttgart  
70550 Stuttgart, Germany  
[dominik.lamp@rus.uni-stuttgart.de](mailto:dominik.lamp@rus.uni-stuttgart.de),  
[soeren.berger@rus.uni-stuttgart.de](mailto:soeren.berger@rus.uni-stuttgart.de)

**Abstract**— current service platforms or frameworks, e.g., Cloud solutions, do not take the infrastructure, necessary for the execution of the service, sufficiently into consideration. They take resources like network connectivity for granted and do not provide an integrated networking approach considering Quality of Service (QoS) or other real-time aspects of the message exchange between possibly thousands of components.

This paper presents the concept of a fully managed network virtualization framework to provide the required connectivity between components within a virtualized service platform respecting all service requirements, e.g. as expressed by interactive real-time services, on transport layer.

*Network Virtualization; Service Oriented Infrastructure; Real-time service support*

## I. INTRODUCTION

In recent years the virtualization paradigm has gone through its first critical transition that brought about its fusion with the Service Oriented Architecture (SOA) into what is now commonly referred to as the Service Oriented Infrastructure (SOI) paradigm.

SOIs build upon previous advancements in Distributed Systems, Grid Computing, Cloud Computing, Virtualization, SOA and related technologies. Capabilities merged under the banner of SOI offer a solution serving long-standing business needs, but also meet increasing demand for infrastructures enabling fast and flexible deployment of new services.

However, typical current SOI realizations, e.g., Grid or Cloud solutions, do not take the network infrastructure, necessary for flawless service interaction, sufficiently into consideration. In most cases, those frameworks focus on processing of batch jobs with no timing and location constraints, which can be executed in huge remote data centers. They primarily manage computing related resources like CPU and RAM and persistent Storage [e.g. Amazon Web Services], but network connectivity is typically taken for granted while network Quality of Service (QoS) aspects (e.g., jitter, delay) of the data exchange is usually not considered. Consequently, the data exchange between changeably deployed components can not be comprehensively treated [1]. On the other hand, Telecom Service Delivery Platforms (SDPs) currently lack the motivation for Cloud computing, hence focus on capability

exposure & interworking [2] and network management optimization through content-based routing, service discovery and selection to reach Pareto-optimal network utilizations [3].

This paper presents the concept of a network virtualization framework bridging these gaps by dynamic on demand provisioning of network resources based on individual, highly abstracted service requests. This includes provisioning of required connectivity between components in a virtualized service platform as well as taking QoS demand considerations and enforcement (as expressed by, e.g., interactive real-time services) into account.

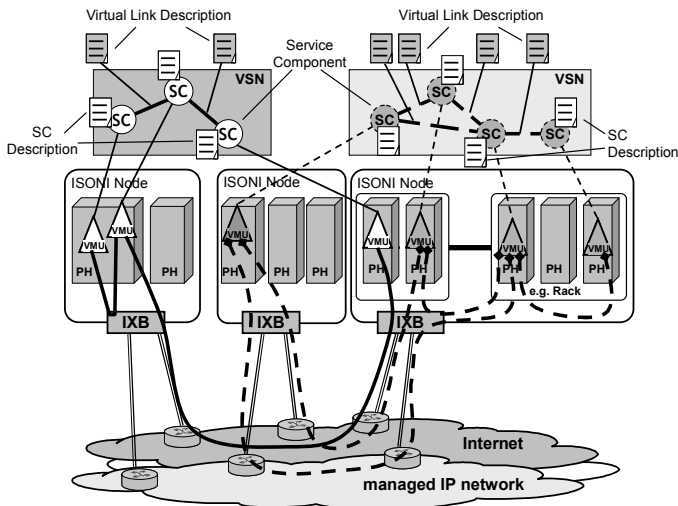
The paper starts with an introduction of SOIs, thus providing the context of network virtualization within virtualized service platforms. The subsequent chapter deals with the concept of network virtualization, starting with a general motivation and ending with a description of a Network Resource Supervision Framework. Finally, the chapter on proof-of-concept implementations provides the latest results of concept verification as being currently carried out in [4].

## II. SERVICE ORIENTED INFRASTRUCTURE

The basic purpose of an Intelligent Service Oriented Network Infrastructure (ISONI) [5] is to considerably reduce the complexity for service providers/developers to roll out new network based services as it takes care of the automatic deployment of the services on best fitting resources distributed in a network. Additionally, the ISONI will provide means to isolate different deployed services from each other in order to prevent unwanted crosstalk between them.

A major task of the ISONI is to completely separate the management of all kind of hardware resources distributed in a network from that of deployed services and their associated service components. By that, the actual status and distribution of resources are hidden from the service developer's view. The infrastructure provides him with fully virtualized resources, including the network resource. That enables a service developer to deal with a complicated network of resources in a simplified way at a level of high abstraction. This full virtualization of a network of distributed resources is the essential prerequisite to get the freedom of resource and service management we need to serve the purpose of the ISONI as described above.

Figure 1 illustrates the principal service deployment performed by the ISONI. The upper part of the figure shows the highly abstract view of two example services in form of “Virtual Service Networks” (VSNs) as provided by the service developer; the examples are distinguished in the figure by dashed lines for the second VSN. Each VSN is composed of multiple so called Service Components (SCs) interconnected by virtual links each specified by a virtual link description. The lower part of Figure 1 shows the mapping of the highly abstracted resource requests (two VSNs) onto the network of real resources, execution environment (Virtual Machine Units) and real links through the network(s) between those VMUs. The ISONI eXchange Box (IXB) is ISONI’s central building block for network virtualization, which is described in more detail in section III.B.



**Figure 1: Deployment of services on the ISONI**

### III. NETWORK VIRTUALIZATION - THE CONCEPT

#### A. Motivation of the concept

As indicated in the introduction typical current frameworks do not sufficiently consider the network infrastructure necessary to provide network resources dynamically on service request and to keep strong quality guarantees as demanded by, e.g., interactive real-time services.

Network virtualization is a promising approach to cover individual and dynamic resource provision while keeping strong individual QoS requirements and optimizing the overall resource usage. In particular, resource virtualization provides:

**Aggregation:** Physical resources can be aggregated, spanned, or concatenated to build a virtually enlarged resource, a clustered resource or a resource pool

**Segmentation:** Several different Applications share one physical resource to increase resource utilization.

**Isolation:** No unwanted crosstalk between applications in case of resource sharing caused by program crashes, sniffing, attacking, etc.

**Encapsulation:** Hiding of resource complexity due to the provision of simplified generic interfaces; moving whole virtual environments for reasons of load balancing, hardware maintenance, and redundancy in a transparent way.

The important features here with respect to network virtualization are segmentation, isolation, and encapsulation. Segmentation allows several different services to share a physical link with given QoS properties. Encapsulation enables service developers to design service specific overlay networks at a high level of abstraction, thus disburden them from dealing with highly complex physical network infrastructures. Finally, means for isolation are imperatively needed to suppress any unwanted crosstalk between services sharing physical links.

Techniques for network virtualization have already some history. E.g., VLAN, VPN and tunneling mechanisms in general are network virtualization techniques, but they have been invented to interconnect remote locations of organizations and enterprises in order to enable secure and seamless application interworking. They are setup statically by means of ‘classic network management’ to serve individual use cases. They are expensive and certainly lack the dynamics we target with our framework.

In summary, we introduce a network virtualization framework with a special focus on dynamic on-demand provisioning of network resources based on individual highly abstracted service requests. A particular focus lies on the requirements of interactive real-time services. The framework enables sharing of transport resources between numerous service specific isolated overlays.

This sharing and usage optimization of transport resources has more impacts. Besides decreasing capital expenses (CAPEX) due to increasing resource usage and lowering operational expenses (OPEX) due to encapsulation, our framework serves the more and more important demand on saving energy.

#### B. Network Resource Supervision Framework

In this paper, we provide an insight into the different aspects of ISONI’s network resource supervision framework.

The main aspects include: Network resource management, connectivity and network support for migration, Quality of Service, individual Flow Control and finally monitoring and health supervision.

One basic task of the ISONI is to considerably reduce the complexity for service providers/developers to roll out new network-based services as it takes care of the automatic deployment of the services on best fitting resources distributed in a network.

Another major task of the ISONI is to deploy and instantiate the application developer’s service on the ISONI. The ISONI will accomplish this task automatically and autonomously. For that the ISONI needs an abstract description of all the requirements of the service on the execution environment, including the description of the interconnections and their individual QoS demands. The level of abstraction of this description is as high as possible to ease its creation, while

still allowing for automatic matchmaking. In particular, the creation of the description will not require special knowledge about the network infrastructure. This description has to be delivered to the ISONI in form of a VSN description.

A network service is composed of multiple software modules deployed to several machines. The VSN description capsules one or more service components and their configuration in a vertex description, called ISONI Service Component description. For each ISONI Service Component description, a concrete realization is derived. A typical realization is based on virtual machines called Virtual Machine Units (VMU) in the ISONI context. The ISONI management assigns resources to these VMUs, and schedules, deploys, and interconnects them. The virtual interconnections and the applied addressing scheme for service networks running on the ISONI are further detailed in the following chapter. It covers the exposure of the services to other networks too.

#### 1) *Connectivity and network support for migration*

The ISONI uses standard transport resources such as the Internet or dedicated transport resources between ISONI nodes with predefined QoS properties (provided by Network Provider(s)), augments them with own shaping and scheduling mechanisms, and assigns bandwidth of individual links to VSNs to account for the QoS requirements provided in the VSN description.

There are three differently managed types of network ISONI can rely on to provide the required connectivity between individual service components with specific and individual requirements in terms of QoS guarantees: best effort [The Internet], routed IP networks [e.g. MPLS], and leased point-to-point connections [e.g. Ethernet over SDH]. The renting/owning of different types of network resources allows an own ISONI QoS management as presented in chapter “Quality of Service”.

For proper resource allocation, an accurate view of the current usage of all these resources, components, and connections is vital. Due to scalability reasons, the resource management is decentralized wherever possible and split into the three hierarchical levels: Physical Host level, Node level, and Domain level. Those levels are described below.

The *Physical Host (PH)* is part of an ISONI Node providing computing resources. Any kind of hardware can serve as PH. Examples are 19” x86-servers or ATCA-frames as generic computing platforms, DSP-farms with hard-coded functionalities, storage, etc.

The *ISONI Node* represents the second hierarchical level of the ISONI management middleware. A node consists of a number of PHs. They are typically connected via a high network bandwidth and low latency network connection.

The realm providing all ISONI capabilities is called *ISONI Domain*, which belongs to an ISONI provider. It represents the highest level of hierarchically organization of the ISONI management middleware.

In order to allow the ISONI to police resource usage of a VSN while guaranteeing the availability of the required resources, the VSN developer must specify all required

interconnections between the ISONI Service Components in the VSN. The specification includes the required parameters of the interconnections. With this information, the ISONI is able to setup a VSN that:

- prevents unwanted communication (“crosstalk”) between Service Components in a VSN
- prevents crosstalk between VSNs

Due to this isolation, the ISONI Service Components of a VSN can neither accidentally nor deliberately access components of another VSN. The VSN *as a whole* also cannot misbehave, i.e., consume more than the allocated resources, as both the link usage and the resource usage of computational resources are controlled by ISONI.

To abstract from the underlying network infrastructure and isolate VSNs against each other, a layered addressing scheme has been developed to provide a flexible connection between the addresses assigned to the ISONI SCs and the actual hardware that is used to run that SC.

#### **Virtual Address Layer**

The address layer that is used inside the VSN description is treated as a *virtual address layer*. Addresses at this layer are not required to be globally unique, as they are only considered in the context of the (unique) namespace that is assigned to each VSN by the ISONI. From a conceptual point of view, any arbitrary layer 3 protocol could be used for ISONI SC interconnections in the VSN at the virtual address layer. As the ISONI has to have a basic understanding of the addressing concept used in a VSN in order to perform routing decisions, the ISONI proof of concept implementation uses an approach specifically based on IP at present. The VSN Creator has to specify the virtual addresses that will be used by the ISONI SCs in the VSN description.

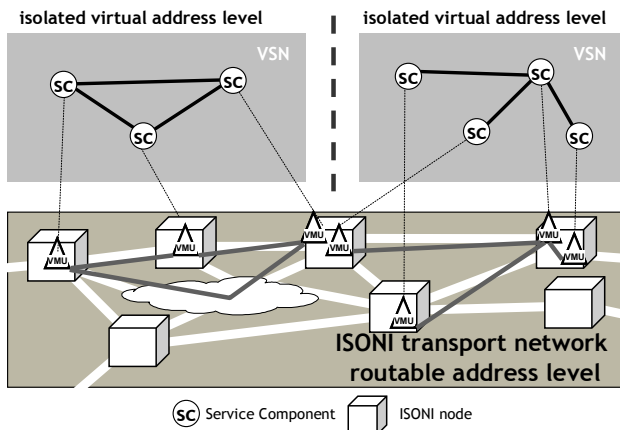
#### **Physical Address Layer**

The IXB of each ISONI Node is attached to one or multiple transport networks. The network traffic exchanged between ISONI SCs is encapsulated by the IXB depending on the used network path in relation with the used transport networks. This allows the ISONI to deploy SCs on any ISONI Node that is capable to execute this ISONI SC. The separation of virtual address space and routable address level (the physical address space of the transport network) allows to migrate running VMUs live within and between ISONI Nodes. At the end of such a live migration the impacted virtual ISONI SC traffic is switched over by using a newly prepared network path.

The correlation of the described address levels is depicted in Figure 2. All ISONI SCs in the VSNs are deployed to concrete ISONI Nodes in form of a concrete VMU configuration consisting of operating system, libraries etc. Alternatively, ISONI could select concrete deployments depending on dedicated hardware like DSPs. For execution, each VMU is deployed to a PH.

In a practical realization, the VSN concept and the encapsulation of the inter-SC traffic in tunnels provides the feature of dynamically changing the flow to another host. In case of a live migration of a VMU, ISONI switches over by

using another network path. So moving a VMU and adjusting the tunnels results in a complete virtualization of the environment. Of course ISONI is also aware of its QoS constraints during the migration.



**Figure 2: ISONI address layers**

2) *Network resource management*

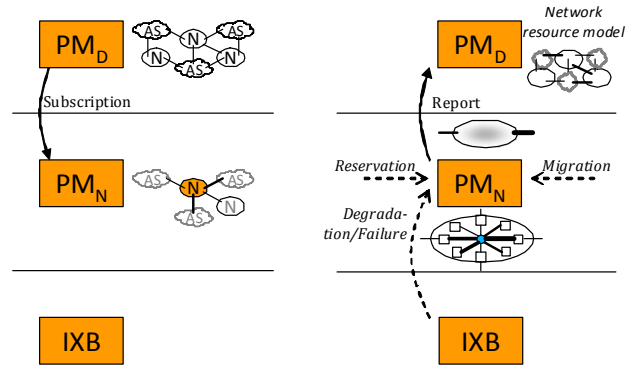
An important advantage of the IRMOS ISONI over other distributed execution platforms is the consideration of applications’ network requirements for component positioning. The ISONI Path Manager Architecture conciliates all network-related tasks of a managed network infrastructure like resource discovery, path selection, configuration and supervision, to sustain individual guarantees for concurrently running real-time services on a common infrastructure without interfering each other.

Concurrent execution of VSNs requires traffic isolation, which is realized with the innovative namespace concept and the network virtualization elements (IXBs) as described in the previous section. The Path Management manages the IXBs’ capacities and controls virtual link segmentation and isolation for scheduled VSNs at the virtualization layer of the ISONI composite structure.

The ISONI architecture defines nodes as an aggregation of PHs to enable a more efficient and domain-wide management of resources. The 2-level hierarchical Path Management layers of the ISONI Path Manager Architecture are aligned with the hierarchical segmentation and abstraction of an ISONI Domain, i.e. it consists of:

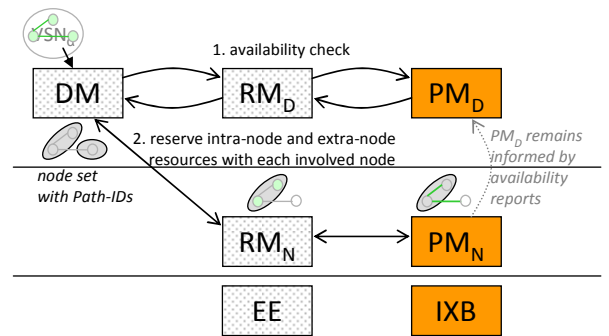
- Path Manager Domain (PM<sub>D</sub>)
- Path Manager Node (PM<sub>N</sub>)

In accordance with the 2-level hierarchy, an abstract domain-wide model representation of the network infrastructure was developed to allow for light-weight domain-level management. The PM<sub>D</sub> maintains an abstract view of the Node network capacity and available node interconnections, in order to support the domain-level Resource Manager (RM<sub>D</sub>) during node selection for possible reservations. In accordance to the node-level abstraction, the PM<sub>N</sub> hides realization details of its connectivity and realization of intra-node network isolation from the domain-level management entities. Thus, the PM<sub>N</sub> delivers aggregated reports of its network availability schedule to the PM<sub>D</sub>.



**Figure 3: Availability Reporting**

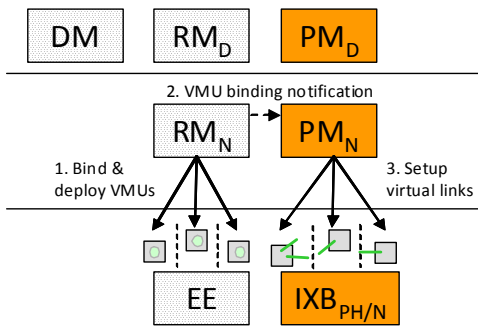
The resource availability reporting process between the node- and domain-level is depicted in Figure 3. The figure shows the 2-level hierarchical path management layers (PM<sub>D</sub> & PM<sub>N</sub>) and the network layer, realized by IXBs. On the left, the PM<sub>D</sub> is depicted having an abstract view of the connectivity inside ISONI Nodes (N) and of wide-area network (WAN) connections between nodes or of interconnections with Autonomous Systems (AS). Each PM<sub>N</sub> reports its resource availability to the domain level (right side of Figure 3) initiated by subscription of the PM<sub>D</sub> to respective PM<sub>N</sub>’s (left side of Figure 3). The PM<sub>N</sub> maintains the resource availability status of its internal and external connections. When events such as VSN reservation, VMU migration, Path degradation or element failures change a Node’s network resource availability, the PM<sub>N</sub> delivers an updated status report to the PM<sub>D</sub>.



**Figure 4: Availability Check & Reservation**

Figure 4 sketches the VSN’s reservation process coordinated by the VSN-specific Deployment Manager (DM) instance. Triggered by a VSN instantiation request, the DM instance checks with the domain-level resource management for a potential deployment (1. availability check). After the RM<sub>D</sub> has selected possible sets of nodes that would accommodate the computational and storage requirements of a VSN, the PM<sub>D</sub> validates each node set by mapping the node-spanning virtual links to Node connectivity correlations that fulfil the network requirements of the links. Furthermore, the probability of successful VMU interconnection inside each Node is estimated on the basis of those virtual links that remain node-internal and the internal capacity report of the targeted node. If the network capacities of a node set can accommodate the VMU interconnections, the node set is reported valid to the RM<sub>D</sub> along with the path selection for node-spanning links.

Upon retrieval of a valid node set proposal from the resource availability check with the domain-level management entities, the DM starts a reservation transaction (2. in Figure 4). It generates some sub-VSNs from the VSN description for each involved Node. This sub-VSN is sent to the corresponding  $RM_N$  of each ISONI Node involved in the proposed node set. The  $RM_N$  determines the necessary node-internal computational and storage availability status and passes the request on to the  $PM_N$  to have the network availability status confirmed. The reservation clears a node's available computational resource and available external interface for the deployment of the sub-VSN. Upon successful reservation, the  $PM_N$  updates its network allocation schedule for the targeted VSN execution time frame, including just assigned external capacities for links towards extra-node endpoints and eventually reports an availability status update to the  $PM_D$ .



**Figure 5: VSN Deployment**

At deployment time of a scheduled VSN, the node-level management ( $RM_N$  &  $PM_N$ ) autonomously deploys its assigned sub-VSN as depicted in Figure 5. The link setup by the  $PM_N$  is preceded by the  $RM_N$ 's selection of the PHs to define the node-internal endpoints of the virtual links. The start-up is completed with the node-internal interconnection of the VMUs. During execution, the  $PM_N$  remains responsible for the supervision of the virtual link states. Accordingly, the node-level management stops and resaves the VSN deployment data upon scheduled termination time of the VSN.

### 3) Quality of Service

IP-based ("internet like") networks internets were designed for elastic applications that tolerate variations in throughput and loss. Now, they are used to support high volumes and various traffic mixes including real-time and non real-time applications. Real-time traffic is sensitive to delay and delay variations (jitter), which generates higher demands on provided network service known as quality of service (QoS). But quality is more than just delay and jitter. It may also cover ensuring throughput, loss rate guarantees and availability of a service in general. ISONI counteract loss rate guarantees and availability of service by taking adequate actions based on monitoring and health supervision described later, like re-allocation of the application to a different ISONI node. The Internet just provides one type of service well known as *best-effort*. Two distinct approaches have been made by IETF for adding enhanced network services to the Internet in reaction of the increasing necessity for traffic management and congestion control. On the one hand there is the Integrated Services (IntServ) approach [7], which limits the demand per network flow and reserves the resources to meet the QoS. Intserv

provides end-to-end QoS that requires keeping states per flow and an individual reservation per flow, which can ensure QoS in respect to delay, jitter and throughput, but it does not scale. The reservation per flow requires a reservation protocol (RSERV). On the other hand there is the Differentiated Services (DiffServ) approach [8], which classifies traffic in groups, i.e. it allows class-based handling of aggregated flows. The DiffServ approach scales, but it does not provide absolute QoS guarantees. ISONI provides both the scalability and absolute QoS guarantees. On the overlay network level ISONI uses traffic classes similar to the DiffServ approach, which allows the handling of VMU traffic in classified traffic groups called ISONI QoS classes. Each ISONI domain may choose its own overlay QoS traffic management strategy, which may result in different numbers of ISONI QoS classes realized by each ISONI domain. ISONI routes the traffic of each virtual link according to its QoS requirements specified in the virtual link description by assigning adequate ISONI QoS classes. An ISONI QoS class refers to the packet scheduling, queuing, or policing behavior of the virtualized inter VMU traffic and ensures the same behavior on each ISONI Node within its domain. It is important for the (re-)allocation of VMUs, especially for live migration, to find the same behavior of traffic management in other ISONI nodes. Together with the policing of used resources by the flow control mechanisms, the ISONI can give QoS guarantees from edge-to-edge of the ISONI domain. The managing of resource reservation is given with the VSN deployment, which discharge the deployed application to deal with reservation protocols as in the IntServ case. It is the merit of synergy between network virtualization and managed deployment, which enables ISONI providing a scaling edge-to-edge QoS environment.

### 4) Individual Flow Control

Each ISONI Node manages the allowed traffic for each transport network access group (illustrated in Figure 6), which is intended to be used for inter VMU traffic. A transport network access group is a generic functional architecture term [9], which is an abstraction of one or several physical network links accessing a transport network. Therefore, the allowed amount of traffic for each transport network access group in correlation with the ISONI QoS classes is specified by the ISONI operation and maintenance. Each ISONI node keeps track of each reserved network resources specified within the Virtual Link Description (illustrated in Figure 1). New VSN deployments are only accepted by an ISONI node, if sufficient resource capacity is available. To avoid interferences among individual virtual link traffic following the sharing the same physical network resources, the traffic of each virtual link is policed by ISONI flow control as depicted in Figure 6 (the diagram convention has been taken from G.805 [9]). The flow control applies to all ISONI QoS classes without any exception. The flow control is absolutely essential for ISONI to ensure given QoS guarantees.

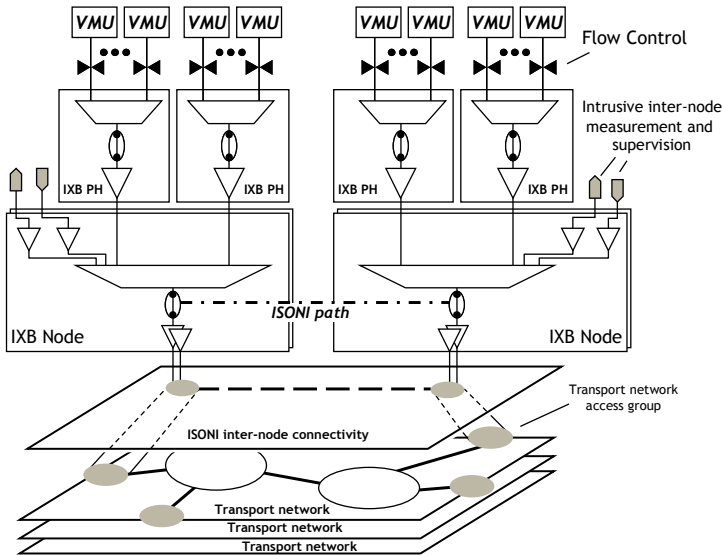


Figure 6: ISONI multiplex and network hierarchy

#### 5) Monitoring and health supervision

ISONI relies on existing available transport networks and on leased lines. ISONI ensures availability of network resources by continuous health supervision of used network links. Health supervision also detects degradation of network links by continuous measuring of certain QoS parameters. ISONI can switch between deployed virtual links to save network links. Due to the virtualized connectivity the switch-over can be realized in different manners:

- Seamless, by using transport network access groups (e.g. bonded network links)
- Switch-over, by using a prepared spare link
- Establish new link

If necessary, ISONI may consider re-allocation of parts of a VSN to other ISONI nodes ensuring the availability of a service.

ISONI also continuously monitors the QoS related parameters of the setup virtual links. The measurements therefore are applied preferably non-intrusively. In case of intrusive measurement, ISONI takes representative measurements, which will be assumed for all virtual links using this network path. Figure 6 shows an example of an intrusive inter-node measurement (e.g. one way delay measurement - OWD) entwined in both directions.

#### IV. PROOF-OF-CONCEPT IMPLEMENTATION

The proof-of-concept implementation is supported by the European Funded Project IRMOS [4].

The current implementation is focused on the path management, whereas the first focus was on the low level components like the IXB for evaluating the configuration and isolation of different parallel deployed VSNs. A remote configuration of VSNs is already possible. Tests to

dynamically change the tunnel configuration for support of migrating VMUs are already been carried out successfully.

Within the IRMOS project, a first proof of concept demo including the automatic VSN deployment is anticipated for early 2010 which include a first version of the path management architecture.

#### V. CONCLUSIONS

We have presented a network virtualization framework with a special focus on dynamic on-demand provisioning of network resources based on individual highly abstracted service requests. The specific requirements of interactive real-time services have been highlighted. The framework enables sharing of transport resources between numerous service specific isolated overlays.

The main aspects presented are: Network resource management, connectivity and network support for migration, Quality of Service, individual Flow Control and finally monitoring and health supervision.

Future work especially in the context of the IRMOS project will focus on further conceptual work and proof of concept implementation of essential innovative parts such as the Path Manager.

#### ACKNOWLEDGMENT

This research is partially funded by the European Commission as part of the European IST 7<sup>th</sup> Framework Program through the project IRMOS under contract number 214777.

#### REFERENCES

- [1] Konstantinos Tserpes et al., "An Open Architecture for QoS Information in Business Grid", [http://www.nextgrid.org/download/publications/Business\\_Environments\\_and\\_QoS%20Correlating\\_Customer\\_Requirements\\_with\\_Provider\\_Capabilities.pdf](http://www.nextgrid.org/download/publications/Business_Environments_and_QoS%20Correlating_Customer_Requirements_with_Provider_Capabilities.pdf)
- [2] Shao, Xu; Chai, Teck Yoong; Lee, Teck Kiong; Ngoh, Lek Heng; Zhou, Luying; Kirchberg, Markus, "An Integrated Telecom and IT Service Delivery Platform", Asia-Pacific Services Computing Conference, 2008. APSCC '08. IEEE 9-12 Dec. 2008
- [3] Callaway, R.D.; Devetsikiotis, M.; Viniotis, Y.; Rodriguez, A, "An Autonomic Service Delivery Platform for Service-Oriented Network Environments", Communications, 2008. ICC '08. IEEE International Conference on 19-23 May 2008
- [4] IST FP7 IRMOS Project, <http://www.irmosproject.eu/>
- [5] Marcus Kessler, Andreas Reifert, Dominik Lamp, Thomas Voith, "A Service-Oriented Infrastructure for Providing Virtualized Networks", Bell Labs Technical Journal 13(3), 2008, Published by Wiley Periodicals, Inc.
- [6] IRMOS Project D6.1.1 Formal description language for application requirements of Execution Environment, USTUTT and other partners, November 2008
- [7] IETF, RFC 2475, An Architecture for Differentiated Services (diffserv)
- [8] IETF, RFC 1633, Integrated Services in the Internet Architecture (intserv): An Overview
- [9] ITU-T, G.805 (03/2000), Generic functional architecture of transport networks