2011 Eighth IEEE International Conference on Mobile Ad-Hoc and Sensor Systems

VITRO architecture: Bringing Virtualization to WSN world

Monica Navarro
CTTC
Barcelona, Spain
Email: monica.navarro@cttc.es

Marcello Antonucci
Selex Sistemi Integrati
Rome, Italy
Email: m.antonucci@infosis.it

Lambros Sarakis, Theodore Zahariadis
TEIHAL
Chalkida, Greece
Email: sarakis,zahariad@teihal.gr

***Abstract*—This paper presents the architecture of the VITRO system concept developed under the VITRO project for the realization of Virtual Sensor Networks. The aim of the proposed architecture is to enable the realization of scalable, flexible, adaptive, energy-efficient and trust-aware Virtual Sensor Network platforms, focusing on the reduction of deployment complexity and on advanced interoperability mechanisms. The efforts have been put towards specifying a Service Provisioning architecture and mechanisms for advanced sensor and middleware design.**

***Keywords*-WSN, virtualization, system architecture**

I. Introduction

Over the past few years research interest on Wireless Sensor Networks (WSNs) has continued to grow accompanied by a constantly increasing interest for real WSN deployment and commercial exploitation of the services that these networks provide. Although research and development efforts have focused on WSN systems that are dedicated for a specific application, this trend is being replaced by resource-rich Wireless Sensor Network deployments that are expected to provide capabilities in excess of any application's requirements. In this framework, the concept of Virtual Sensor Networking is an emergent approach that enables the dynamic collaboration of a subset of sensor nodes and helps the proliferation of new services and applications beyond the scope of the original deployment. The concept of Virtual Sensor Networks (VSN) [1, 2] introduces a new and dynamic collaboration paradigm that requires to accommodate multiple logical network instances over a single physical network infrastructure with the ultimate goal of a) supporting applications with different requirements both in terms of nodes and communication functionalities and b) utilizing in an efficient manner the available network resources. Furthermore, virtualization of network resources and services has been identified as key technology for Future Internet architecture, to accommodate the IoT vision [3, 4]. Whilst mechanisms are in place to facilitate virtualization in the core Internet network [4], major challenges remain in the embedded fringe networks. On the one hand, these have to do with individual functionalities of a VSN system related to middleware, routing, security, trust and energy awareness of protocols and mechanisms at various layers of the communication protocol stack. On the other hand, significant challenges arise from the development of a holistic approach to VSN, which should take into account not only individual enhancements related to the aforementioned functions but also the need to federate resources across networks belonging to different administrative domains as well as the need to support promising business models.

Following this trend to overcome the constraints of application specific WSN deployments to be more efficient, flexible and adaptive in a wide sense, the VITRO project [5] aims at describing and implementing an integrated system architecture, along with the appropriate engineering tools and methods to extend the notion of a WSN towards the realization of large-scale, cross-organizational VSNs supporting a wide range of applications.

A design choice of VITRO has been the selection of the Internet as the physical bearer between sensor platforms and the applications. The motivation for this choice is that VITRO-compliant sensor networks can be considered as a large network, whose core is the Internet, where VITRO-compliant client applications run. At the periphery of this global network there are the single wireless sensor networks used by those applications. In this new context, where sensors are deployed by some organizations and then used by other organizations and where virtualization may imply virtualization of services but may also have a more physical implication like virtualization of node resources, two are the main drivers for the architecture design:

- *Advanced sensor design* able to support advanced features in several fields (energy saving, routing capability, middleware support, etc.), which requires the definition of special architecture for the sensor where flexibility is a key feature.
- *Middleware design* to mediate between applications and sensors. Under the paradigm that sensors do no longer know the application that will use them, and an application does no longer know the sensors that it will use, major issues that the system architecture must address are: how does a new application find the sensors that it needs, how does it negotiate for the right to use those sensors with the organizations who deployed and administer them, how can an application react to changes in the network? That is, the mechanisms for matching between the applications (user) and the

978-0-7695-4469-4/11 $26.00 © 2011 IEEE
DOI 10.1109/MASS.2011.96

resources (used) have to be specified.

The proposed system architecture has tackled both these issues, by defining in detail a new reference architecture for the sensors and an architecture for the middleware that binds together the sensors and the applications. Also, with the aim of not leaving out the numerous current WSN deployments, VITRO has included a third driver in the architecture design by including proprietary and legacy, non-VITRO-compliant sensor networks. VITRO design, wherever possible, has avoided imposing specific characteristics or features to be mandatory implemented within the sensor node, and has instead allowed those to be implemented somewhere else on the Internet, provided that the accordance with a specific interface is ensured. The motivation for this choice is to facilitate early adoption of VITRO services among existing WSNs that with very limited investment, may implement adaptation servers for their WSNs, and open them to being used by VITRO applications, favoring quickly bootstrapping of the global VITRO market.

The remainder of the paper is organized as follows. Section II describes the proposed VSN general architecture. The architectural components of the VITRO gateways and sensor nodes are presented in Section III and, finally, concluding remarks are given in Section IV.

II. VITRO GENERAL ARCHITECTURE

In this section the VITRO general architecture is presented. First, we introduce the main VITRO actors and then we review the functional components of each of them. Previous to detailing the architecture we shall introduce some definitions:

- *Virtual Sensor Network (VSN)*: A VSN is the seamless grouping of WSNs also called Wireless Sensor Islands.
- *Wireless Sensor Island (WSI)*: A WSI is any grouping of one or more legacy, proprietary or VITRO-aware sensor networks, that may belong to different administrative domains and are able to communicate, respectively, via a legacy, proprietary or VITRO-aware Gateway Node (GW) offering at least one service[1].
- *VITRO service*: Defined as any combination of one or more operational and/or supporting services. As *operational service*, we define any actuation or sensing capability offered individually by a sensor node or collectively by a WSI. As *supporting services*, we define functions that support the provisioning of the operational services. The supporting services may be offered by one or more WSIs or it may be hosted outside the WSI by external service providers. Many VITRO-services may be combined to offer new VITRO-services (as service mash-ups).

[1]Autonomous sensors, which are uniquely addressable from outside a WSI can be considered as a simplified WSI.

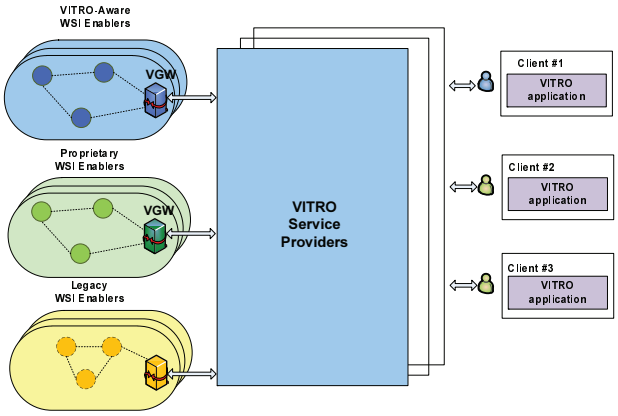

Figure 1. VITRO actors

- *Resource*: As resource, we define any physical or logical entity of a sensor node or of a WSI, which can be allocated, utilised and released in order to realize a service. As such, resources may be considered as service enablers.

Three main actors integrate the VITRO general architecture:

- **WSI Enablers**: These are actors offering the WSI. A WSI enabler may own, lease, manage and/or administrate one or more WSIs and offer at least one operational or supporting service. These services may be offered either free of charge or under specific contracts. In the latter case, guaranteed contracts and/or service level agreements may be offered.
- **VITRO Service providers**: These are actors offering the VITRO services. A VITRO service provider may be a WSI Enabler himself, utilize open access WSIs, have established permanent or negotiate on-demand service contracts with one or more WSI Enablers. The main functions of the VITRO service provider are the service publishing, negotiation, provisioning and monitoring.
- **Users**: These are the actors that negotiate and exploit the VITRO services, potentially under a Service Level Agreement. A user may have a permanent or on-demand service contract with one or more VITRO Service providers.

The VITRO business model foresees an open market environment, which enables open competition and flexible service provisioning. We shall remark that the VITRO business model is not constrained to Fig. 1. In fact, the business model supports a fully cascading architecture, where a VITRO application may utilize a hierarchy of several tiers of VITRO Service Providers along with external third party Service Providers [6]; while various VITRO service providers may also establish permanent contracts - or negotiate them on demand - with WSI enablers, or other

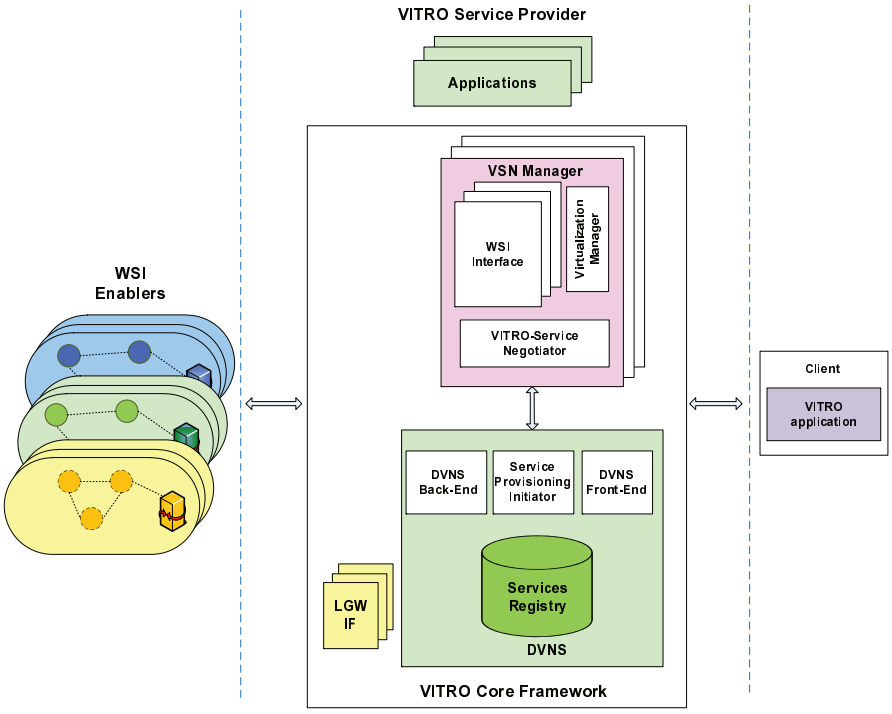

Figure 2. VITRO core framework: DVNS and VSN Manager basic decomposition

VITRO service providers or even with third party service providers. Nevertheless, within the VITRO project we have focused on the business model illustrated in Fig. 1.

A. VITRO Service Provisioning

One of the main challenges addressed during the architecture building process has been finding a solution for VITRO Service provisioning, whose proposed architecture is shown in more detail in Fig. 2.

Each VITRO Service provider offers a number of applications through a VITRO Core framework, that consist of one Dynamic Virtual Network Server (DVNS), a number of instanced Virtual Sensor Network (VSN) Managers and may also host some Legacy Gateway Interworking Functions (LGW-IF). In brief, the VITRO DVNS is responsible for discovering/registering and publishing VITRO-services to the end users and initiate service provisioning; the VSN Manager is responsible for service negotiation, session establishment and monitoring; while the LGW-IF enables the interaction with legacy WSIs.

More specifically, the core part of a **DVNS** is the *Service Registry*. The Service Registry is a database where all known/registered WSI Enablers and the VITRO-services, known to this VITRO-service provider, are listed and described. In addition to the service name, ID and description, the Service Registry has knowledge of the WSI Enabler(s) that offer the services, their gateways, etc., and in particular knows which Gateway(s) is/are responsible for each service.

The additional DVNS components support the Service Registry functionality (detailed in Table I) in the following manner. The *DVNS Back-end* component is responsible for updating the Service Registry on a periodic or on-demand basis. In particular, using crawling techniques the DVNS Back-end queries all known/registered WSI Enablers in order to discover new services or remove services that are not offered any more. It may also enable collaboration between DVNS hosted by different service providers in case a contract is established. In addition, the WSI Enablers and/or the offered services may be directly registered to the Service Registry via the DVNS Back-end component. The discovered or registered VITRO-services are then published by the *DVNS Front-End* to the VITRO end-users/subscribers. The DVNS Front-End is also used for querying and retrieving VITRO-services based on end-user criteria. Depending on the business model, open access, restricted or premium services may be offered to different types of users. As soon as a VITRO-service has been preliminary qualified by the end-user, the *Service Provisioning Initiator* component is invoked to authenticate the user and check the user permissions. Then it initiates a VSN Manager instance. In more detail, the Service Provisioning Initiator will contain a user database with the needed information in order to check user access rights and authentication privileges, get

Table I
MAIN FUNCTIONALITIES OF DVNS COMPONENTS

Component	Functionality
DVNS Back-End	register/discover new WSI Enablers, performing crawling of known/registered WSI for new operational and supporting services and implementing ontology-based representations of services' description.
DVNS Front-End	publishing of VITRO-services in a transparent way to the end-user applications, processing of user queries and translation to VITRO-services and interactive ontology navigation features.
Service Provisioning Initiator	authenticate the user and check the user permissions.

the user's contracts service level agreements, obtain from the services registry all necessary information and initiate a VSN manager instance.

The **VSN Manager**, responsible for service negotiation, session establishment and monitoring, may also perform service renegotiation and accounting/billing of the VITRO services. It consists of the following subcomponents: the VITRO-Service Negotiator, the Virtualization Manager and the WSI Interface. The interface between the VSN Manager and the DVNS is implemented by the VITRO-Service Negotiator. This component receives from the Service Provisioning Initiator in the DVNS all necessary information in order to negotiate and establish a new VITRO-service. This information includes a list with the operational service(s) requested by the user, the WSI Enablers that offer such operational service(s), the WSI entry points that are responsible for the relevant operational and supporting services, etc. The VITRO-service Negotiator queries all WSI and gets up-to-date knowledge of the relevant operational and supporting services, along with the relevant WSI capacity to support a new instance of the operational service. Based on this information and additional non-functional information (e.g. user profile, access permissions, contracts, prices, etc.), the VSN Negotiator will negotiate with the user application the VITRO-service provisioning. In this phase, the User may refine his requirements and negotiate the VITRO-service establishment. If the service negotiation is not successful, the session is closed and the VSN Manager instance is dissolved. Otherwise, the Virtualization Manager is invoked.

At contract start-up, the Virtualization Manager communicates with the various known/registered WSIs and requests the allocation of the necessary resources. This module is also responsible for hiding from the end-user application the actual WSI sensor nodes and the real service(s) [2], and offer the VITRO-service in a transparent way, appearing as a single operational service of the VITRO service provider. Finally, each WSI Interface is a logical process that is responsible for interfacing the WSI and hides any specificities of the wireless sensor island.

At run time, and based on the Contract Service Level Agreement (SLA), the Virtualization Manager may communicate with the WSIs that offer the requested operational services and check the session status/quality. Additionally, the WSI Interfaces may inform the Virtualization Manager if an error has been reported or trapped. In case of failure to meet the negotiated contract, the Virtualization Manager will communicate with the VITRO-Service Negotiator. The Negotiator will initially try to adopt countermeasures by negotiating with the WSI Enablers a new session establishment. If anomalies cannot be compensated and the SLA is broken, the Negotiator may try to re-negotiate with the end-user application or close the connection with an error message (e.g. mail notification).

Finally, the **Legacy Gateway Interworking Function** enables the interconnection between Legacy WSIs and the VITRO network architecture. An entity outside the WSI should offer a LGW-IF, either by the WSI Enabler or by a Service Provider. Alternatively, each VITRO Service Provider should implement instantiations of VGW as interconnection functions for LGW. The LGW-IF should offer the logical VGW functionality (i.e. status and negotiation of services and resources, ontological representations), without realizing the remaining physical layer functions.

In addition to the VITRO Core framework, the VITRO Service provider may offer applications to the end-users, collaborating VITRO service providers or third party service providers. These applications are customized to each VITRO service provider and form the Service provider's portfolio offering.

III. WIRELESS SENSOR ISLANDS

Looking in the Wireless Sensor Islands the VITRO architecture defines in detail the VITRO gateway and the VITRO sensor node. These two components' functionalities have been particularly addressed in the project to enable the appropriate engineering tools realize the virtualization requirements. The full detailed description of the functional blocks of these components and their interfaces can be found in the project deliverable [6]. In this paper, we highlight the main contributions and provide a descriptive overview.

A. VITRO gateway

The VITRO Gateway (VGW) is the device in charge of bridging between one VITRO-aware WSI and the VITRO Service Provider. It handles the bridging in terms of communication protocols (e.g., Medium Access Protocol,

[2]which may run in multiple WSIs

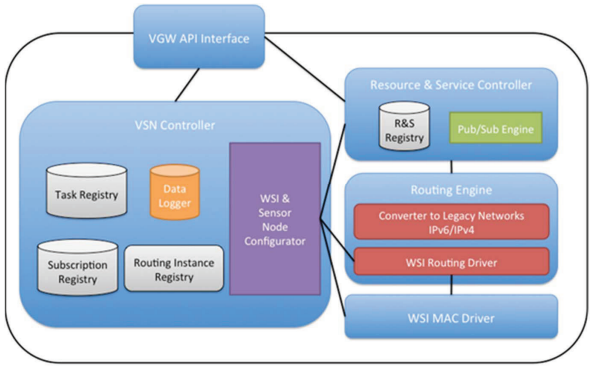

Figure 3. VITRO Gateway functional decomposition

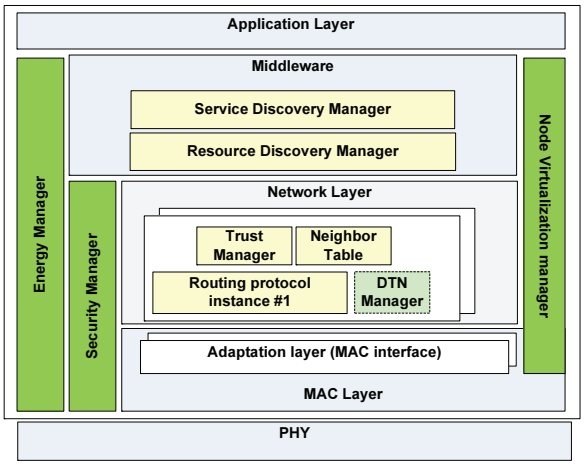

Figure 4. VITRO sensor node functional decomposition

IPv4/IPv6, routing protocol), as well as for upper layers (e.g., middleware, application). The VGW manages the discovery of operational and supporting services, as well as available resources, within the WSI. If required, the VGW notifies VITRO Service Provider(s) about the available services and resources. The functional view of the VITRO Gateway is shown on Fig. 3, whose main components are described next:

- **Interface Handler**: this is an interface used by any external component to instantiate a VSN configuration within the WSI, retrieve the list of available resources/services, retrieve a history of data, or execute an action onto a specific node.
- **VSN controller**: this module receives the instructions to configure, within the WSI, all aspects related to the enforcement of VSNs for supporting VITRO services. It maintains various registries including a registry of the subscriptions per VSN to a set of operational and supporting services provided by the WSI, a registry of tasks that can be deployed on the sensor nodes involved in the operation of a VSN, a data logger, which maintains a history of the data published by nodes involved in an enforced VSN (e.g., measurements, alarms, etc.) and a registry of the routing instances configured in the WSI. The VSN controller embeds a component named WSI Sensor Nodes Configurator to configure the functioning of individual sensor nodes or networking protocols at the scale of the WSI. When receiving the configuration instructions to instantiate a VSN, this configuration component interacts with: the Routing Engine, the MAC Driver module, in order to instantiate a specific MAC scheduling and the Resource and Service controller, in order to subscribe to specific publishing services and manage the resources.
- **Resource and Service controller**: this module negotiates with the VITRO Service provider and keeps a registry of all available operational and supporting services and the status of all WSI available resources. It uses a scalable publication/subscription engine in order to handle published events and inform the VSN Manager (actually the Virtual Connectivity Manager) about the subscribed services and the allocated resources status, as well as pushing actions to specific nodes.
- **Routing Engine**: it receives from the VSN controller the instructions to configure one or more routing instances, with one or many VITRO applications per routing instance. This engine includes a Routing Driver which is in charge of handling the operation of the WSI Routing Protocol(s), and handling for each protocol, the possibly multiple instances of routing plane. In addition, the engine encompasses a Routing Converter which is in charge of converting received packets from the WSI into the appropriate format to be sent over the legacy network.

B. VITRO sensor node

The functional architecture of a VITRO sensor node is described in terms of the protocol stack shown in Fig. 4. In this architecture, virtualization is realized through efficient management of a) node's services and resources and b) functions at the network and MAC layers. Furthermore, it is done in a way that takes into account availability of security features and requirements for energy consumption.

Virtualization at the node level is undertaken by the Node Virtualization Manager (NVM). This component stores, maintains and manages the service requests served by the sensor node (in collaboration with the Middleware component) as well as the current availability of resources (in collaboration, for example, with the Energy Manager and components at network, MAC and physical layers). Moreover, NVM interacts with components at middleware layer to store/update the services offered by the sensor node

in a dynamic way.

The role of the middleware layer is twofold: first, to provide an abstracted and standard view of supported operational services, and second to enable service and resource discovery management within a WSI. As such, the VITRO middleware hides the hardware and software implementation details from the application layer and allows seamless data management between nodes. Additionally, it encompasses mechanisms to enable management of the discovery of services and resources within the WSI and provides, through specific services, the capability for re-purposing existing sensor network deployments and creating VSNs, according to user needs.

In what regards the functions of the network layer, VITRO is focused on routing protocols capable of supporting virtualization through the realization of multiple coexistent routing instances. Each one of these instances may have a Neighbouring Table component, which is used in the routing/forwarding process. In this respect, one of the major contributions of VITRO is the selection of proper routing metrics to accommodate different and sometimes contradicting requirements set by different applications, in such a way that consistency, optimality and loop-freeness routing requirements are met. Tackling with security-related characteristics, the proposed routing solution will specify trust metrics that will be able to investigate and exclude malicious nodes from the traversed path to the destination node. Furthermore, the trust-aware routing protocol will make use of possible security services offered (e.g. encryption, integrity, and authentication) in order to select the optimal path, according to the user request. All information related to trust management is stored and processed by a certain component, called Trust Manager. In cases where there is a lack of continuous network connectivity, the VITRO approach may optionally make use of a Delay Tolerant Network (DTN) mechanism which will be handled by the DTN Manager. This component takes care of communication when a disconnection in the network cannot be repaired by the routing protocol's maintenance mechanisms.

To support the concept of virtualization, different MAC layers may be instantiated in a wide variety of applications. Furthermore, an interface to Energy Manager is foreseen in order for the MAC layer to manage issues related to RF energy (e.g. transmission power and sleep times). Multiple instantiations of the MAC Adaptation Layer will act as an interface to upper layers (Network layer) and cross-layer entities (Energy Manager and Node Virtualization manager) to tune MAC layer parameters.

Finally, management of security and energy-related features are handled by the Security and Energy Managers, respectively. The former is responsible for security measures present in the node, while the latter is responsible for the adaptation of the transmission power level as well as for the provision of information regarding remaining battery time.

IV. Conclusion

We have presented the system architecture for the VITRO system extending the concept of virtualization down to the sensor nodes while ensuring advanced performance and service exploitation. The global virtualization architecture is built around some software components that allow user applications to negotiate, execute and monitor a service obtained by composing basic services provided by sensors, and the architectural design of the physical VITRO aware sensor node. The proposed architecture accommodates the flexibility and adaptability required to achieve an energy efficient realization of Virtual Sensor Networks through the definition of both horizontal and vertical functional layers in the sensor node. In particular, allowing for different instantiations of the MAC and routing protocols to support the concept of virtualization and efficiently enable the different VSN applications that may run on the node.

Acknowledgment

This publication is based on work performed in the framework of the Project VITRO-257245, funded by the European Community. The Authors would like to acknowledge the contributions of colleagues from Hellenic Aerospace Industry, Thales Communications SA, Telefonica Investigation Y Desarrollo SA, Centre Technologic de Telecomunicacions de Catalunya, Research Academic Computer Technology Institute, Technological Educational Institute of Chalkida, Wlab SRL and SELEX Sistemi Integrati S.P.A.

References

[1] A. Jayasumana, Q. Han, and T. Illangasekare, "Virtual sensor networks - a resource efficient approach for concurrent applications," in *International Conference on Information Technology*. Washington, DC, USA: IEEE Computer Society, 2007, pp. 111–115.

[2] R. Tynan, G. OHare, M. J. OGrady, and C. Muldoon, "Virtual Sensor Networks: An Embedded Agent Approach," in *International Symposium on Parallel and Distributed Processing with Applications*, Sydney, Australia, 2008, pp. 926–932.

[3] N. Niebert, I. Khayat, S. Baucke, R. Keller, R. Rembarz, and J. Sachs, "Network Virtualization: A Viable Path Towards the Future Internet," *Wireless Personal Communications*, vol. 45, no. 4, pp. 511–520, 2008.

[4] N. M. Mosharaf, K. Chowdhury, and R. Boutaba, "A survey of network virtualization," *Computer Networks*, vol. 54, no. 5, pp. 862–876, 2010.

[5] "VITRO project," http://www.vitro-fp7.eu.

[6] VITRO-consortium, "D2.3: Vitro system architecture and specifications," FP7-ICT-257245 VITRO, Tech. Rep., Jul. 2011.