Support for heterogeneous dynamic network environments through a reconfigurable network service platform

Eli De Poorter, Ingrid Moerman and Piet Demeester

Ghent University - IBBT, Department of Information Technology (INTEC)
Gaston Crommenlaan 8, Bus 201, 9050 Ghent, Belgium
eli.depoorter@intec.ugent.be

***Abstract*—In a future internet of things, an increasing number of every-day objects will be connected with each other. These objects can be very diverse in terms of their network protocols and communication technologies. As more resource constrained devices such as sensor networks and PDAs are added to our environment, supporting efficient communication between these heterogeneous devices becomes a key challenge in next-generation networks. To realize this challenge, this paper presents a reconfigurable network framework (IDRA) that supports direct connectivity between heterogeneous co-located devices, without the need for complex translation gateways.**

I. Introduction

New communication technologies are introduced and deployed on a regular basis. Even common everyday objects nowadays come equipped with (wireless) communication possibilities. As a result, several authors have described an 'internet of things' view of the future, in which every object is connected with every other object [1] (Figure 1). By connecting these different objects, intelligent next-generation applications such as wireless building automation applications [2] or e-health applications [3] become possible.

However, as the number of communicating objects increases, so does the number of co-located communication technologies. When multiple networks operate in the same geographical environment, co-located networks overhear transmissions from multiple networks. Most often, overhearing these transmissions results in harmful interference and performance degradation, since the overheard transmissions can not be interpreted by devices that are not part of the originating network. This is especially a problem in 'last mile' home and office networks. A typical example is the interference in the free license ISM band, which is used by a variety of communication technologies such as IEEE802.1 (WiFi), car alarms, baby monitors, IEEE802.15.1 (bluetooth), cordless DECT phones and IEEE802.15.4 (zigbee) personal body area networks.

Even when co-located devices use the same radio technology, direct communication between devices is not always supported. For example, existing sensor and actuator networks often use propriety network technologies that are incompatible with technologies from other vendors, even though the devices

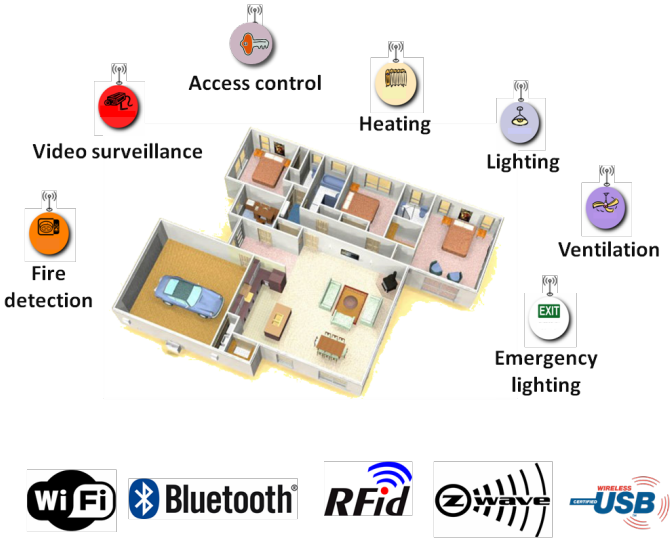

Fig. 1. In the vision of the internet of things, everyday objects will all become interconnected using a variety of communication technologies. These objects can use different communication technologies, different packet types and different network protocols.

use the same radio chip. Currently, communication between heterogeneous devices is supported using one of the following methods. (i) Multiple (proprietary) communication stacks can be installed on a single device. The main disadvantage of this approach is that the memory overhead of using multiple communication stacks is significant. As a result, this approach is not well-suited for resource-constrained devices. (ii) An alternative approach is the use of technology specific translation gateways. All communication is routed through these (remote) translation gateway. These gateways terminate the connection from one network and set up a new connection to a second network. However, translation gateways break the end-to-end communication paradigm and have proven to be inherently complex to design, manage and deploy [4]. In addition, setting up a communication path to the (remote) gateway results in additional packet overhead, which in turn leads to increased interference, lower throughput and a lower network lifetime.

In this paper, we will discuss a third option: an architecture that is small enough to fit on resource-constrained embedded devices and that can be used to support direct communication between heterogeneous co-located devices.

II. Related work

Before describing the IDRA architecture, this section first gives an overview of related work, i.e.: architectures that are designed to support *direct network connectivity* between *heterogeneous networks*. Two main approaches are discussed: (i) incremental 'evolutionary' architectures and (ii) clean slate 'revolutionary' architectures.

A. Evolutionary internet-of-things approaches

Advocates of an evolutionary approach to a heterogeneous internet of things create new architectures by reusing as many components as possible from existing networking solutions. In their vision, the current internet should 'evolve' into an architecture that is more suited for a heterogeneous networks. A first approach is to gradually improve the existing communication stacks, replacing one function at a time, whenever the need arises. A typical example is the introduction of IPv6 addresses to replace current IPv4 addresses. For this approach to be successful, architectures should be easily extensible. Otherwise, this approach results in a difficult adoption of new technologies, as shown by the problematic transition into IPv6 we are witnessing at the moment.

An alternative evolutionary approach is the use of virtualized network components. Network virtualization [5] is used to present underlying network layers in a uniform way towards a high level application. Different devices are connected by forming virtual networks on top of existing networks: logical links are created between distributed systems using native internet routing and standard IP addresses. Well known examples are Virtual Private Networks (VPN) [6] or peer-to-peer applications [7]. The FP7 4WARD project [8] considers virtual networks to be a fundamental part of the design of future internet devices. The project includes virtualized network solutions for in-network management, generic connectivity and content-centric information objects [9]. Similarly, the MAGNET project [10] offers network virtualization at both layer 2 and layer 3, whereas the ITEA2 usenet project [11] focuses on network virtualization for machine-to-machine communication.

In the context of an internet of things, network virtualization can be viewed in two ways [12]. First, these techniques can be used as a tool for evaluating new disruptive architectures on a large scale using existing networks. Secondly, virtualization can be regarded as a fundamental part of next generation architectures, whereby multiple 'overlay' networks coexist by creating different logical networks for communication purposes [13]. However, for directly connecting heterogeneous networks (such as described in our vision of the internet of things), the use of virtualization techniques has the following disadvantages. (i) Network virtualization is not yet proven to be highly scalable, since setting up an overlay network is often difficult and time-consuming. (ii) Virtualization techniques are often too complex and inefficient to be implemented on resource-constrained embedded devices. And (iii) virtualization techniques are often too high in the protocol stack to efficiently bridge networks that use different communication technologies.

B. Revolutionary internet of things approaches

Opponents of the evolutionary approach emphasize the need for a redesigned, clean slate architecture that inherently copes with next generation network challenges [14], [15], sometimes even abandoning IP based addressing in favor of different addressing schemes. Clean slate initiatives are not always meant to be used directly in new devices, but can be used to sketch a revolutionary new perspective, which can then be brought into existing networks. Several approaches have been proposed.

(i) *Database centric architectures* hide the heterogeneity of underlying networks by only allowing access to network information using database operations. For example, the SENSEI project [16] solves the inaccessibility of low-resource end devices by collecting all data from the end devices and making it available in a centrally accessible remote database. Unfortunately, this approach often results in significant network overhead.

(ii) *Content centric architectures* focus on describing the information that is exchanged between networks. For example, the SemsorGrid4Env project [17] focuses on the development of a semantic middleware layer. At the network layer, network protocols are implemented semantically using a 'descriptive language' [18] that focuses on functionality rather than implementation. Unfortunately, support for directly connecting different networks at low network layers is still lacking.

(iii) *Cloud computing approaches* try to offload resource intensive tasks to more capable nodes. Typically, cloud computing can offer infrastructure, platforms or software as a service to less capable devices [19], [20]. Since cloud computing is regarded as a high layer service, this approach does not solve connectivity challenges.

(iv) *Modular approaches* have also been proposed, whereby the protocol stack is divided into different modules which can be combined to create new network protocols with different functionalities. As such, these approaches can easily integrate new network technologies by updating a single module. Modular frameworks, such as SNA [21], can be used to design new network layers. However, most existing modular frameworks compile these distinct modules into a static network layer. In addition, current modular approaches do not focus on supporting connectivity in heterogeneous environments. Thus, although promising, existing modular approaches offer no additional run-time flexibility when compared to traditional layered approaches. In contrast, the NewArch project [22] discusses how a flexible internet architecture can be created whereby different roles can dynamically be combined at run-time to form 'heaps' [23] which can be adapted to the needs

of the network. Unfortunately, the project did not result in a practical proof-of-concept implementation.

C. The need for new architectures

As shown in the previous sections, several research projects are currently involved with the design of new network architectures. However, an economically viable solution might still be a long way off:

- Though several research projects are currently involved with future internet research, most of these efforts focus on the design of a (high-speed) future internet backbones. These solutions are not suitable for use in resource-constrained environments.
- As cited in [24] 'too many future internet proposals are just extensions of existing protocols or architectures'. As such, these proposals lack the innovation to cope with specific internet of things challenges.
- Finally, too many proposals remain 'paperware': there is a definite lack of implemented prototypes [14], [25].

As such, more practical implementations are needed before a decision can be made regarding the feasibility of an all-encompassing internet of things solution. Especially for resource-constrained devices, there is still room for several improvements. More specifically there exists not yet a simple architecture that

- enables optimized communication **at a network and link level** between co-located heterogeneous networks without the use of complex translation gateways;
- has been implemented and evaluated as a prototype in a large scale experimental setting;
- is compact enough to fit even on low resource embedded devices;
- is fully clean slate, but is also backwards compatible with legacy networks;

In the following section, we will discuss how our IDRA architecture fills this gap.

III. IDRA: A RECONFIGURABLE ARCHITECTURE

To support direct communication with neighboring devices, the following challenges need to be solved. (i) The co-located devices can use MAC and routing protocols with a different behavior (e.g.: reactive versus proactive routing). (ii) The co-located devices can use different packet types (e.g.: IPv4, IPv6 or zigbee packets).

To limit the memory overhead, the architecture should make it possible to transparently combine (at run-time) any of networking protocol with any packet type. This section discusses how the IDRA architecture supports this goal. A conceptual representation of the IDRA architecture is given in Figure 2.

A. Decoupling of protocol logic and packet representation

In heterogeneous networks, the use of different packet types can be required for communication with different neighboring devices. To transparently support multiple packet types on a single device, IDRA has *decoupled the protocol logic from packet representation*. Network protocols do not interact directly with packets, but can instead retrieve information about any received packet through a 'packet facade' (Figure 2b). Through this packet facade, standardized *packet attributes* (metadata) can be added, updated or requested. This metadata can represent header fields such as 'destination', 'quality-of-service' or 'time-to-live', but can also be used to describe extra context information such as the packet origin or destination location, the packet owner or additional descriptive information about the packet.

To interpret the structure of received packets, the packet facade uses one or more 'packet descriptors'. These packet descriptors describe at which header offset packet attributes are stored, as well as which representation is used to store the packet attributes (little/big endian, number of bits, encryption method, etc.) (Figure 2c). This way, network protocols can interact with any packet type as long as the correct packet descriptor is available. By providing the packet descriptors of legacy devices, IDRA services can also interpret packets from legacy networks and interact with legacy packet types. Since there is no direct interaction between network services and packet descriptors, the network protocols can request information from a wide range of packet types.

A packet identification service is responsible for selecting the correct packet descriptor to process incoming packets. The following packet identification methods can be used.

- A publicly available, standardized packet type identification can be added as a unique identification field to each outgoing packet.
- Alternatively, networks that utilize multiple communication technologies can associate a packet type with each interface (e.g: an 802.11 packet type for the WiFi interface, an 6lowpan packet type for the IEEE 802.15.4 interface, etc).
- Some radios offer hardware address recognition features that can be used to identify the address of the sending node. In this case, a shared neighbor table can be used to describe the expected packet type of each neighboring node. This approach is not possible for networks that use non radio-compliant MAC headers or networks that include address-free communication interfaces (such as USB interfaces).
- Finally, a last option is to compare incoming packets with the descriptors of existing packet descriptors using bitmap operations.

This wide range of identification methods ensures that network designers that use the IDRA architecture can always choose the most optimal method for identifying incoming packets. The IDRA system automatically drops all packets that are not recognized by any of these packet identification services.

Finally, IDRA includes automatic packet conversion features. To ensure that outgoing IDRA packets can be interpreted by neighboring devices, an entry can be made in the neighbor table that indicates the preferred packet type for each neighbor. When the specified outgoing packet type is different from the current packet type, the system automatically dismantles the

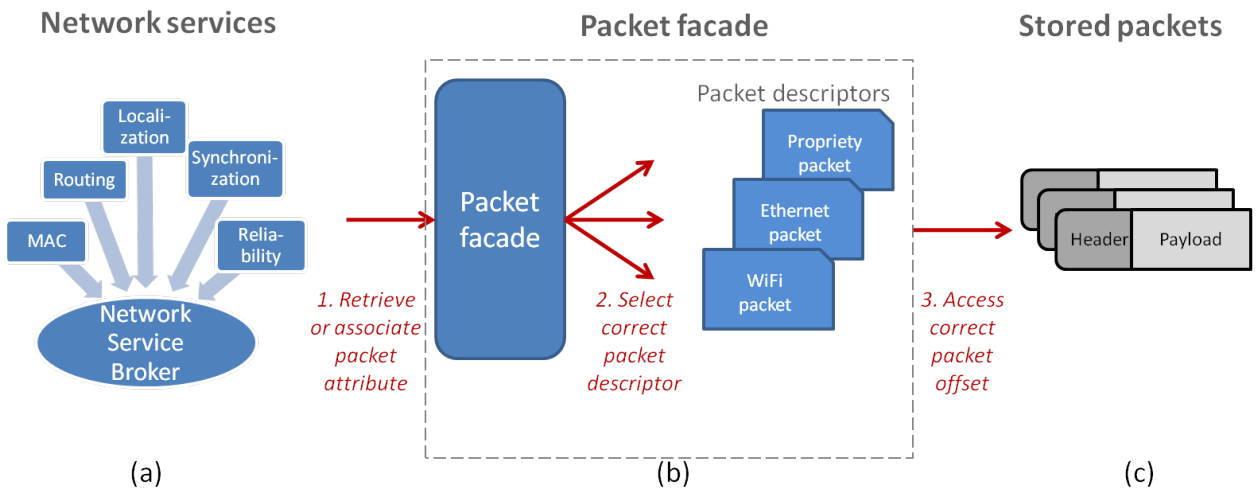

Fig. 2. Network services can transparently interact with any packet type. (a) Network services can associate metadata with, or retrieve metadata from, stored packets using the packet facade. (b) Only the packet facade requires knowledge about the packet format. As long as the correct packet descriptor is available, the packet facade knows how and where metadata is stored. (c) Finally, the packet facade accesses the correct header offset or the packet payload.

packet: all packet attributes are retrieved and stored sequentially. A new packet type is created and all packet attributes are associated with the new packet. As such, packet conversion is fully transparent for the network protocols.

B. Information driven network protocols

To limit the dependence of network protocols on specific (radio) technologies, IDRA network protocols are technology independent. When network protocols wish to exchange information with a remote device, they do not create a packet. Instead, network protocols hand over to the IDRA system any information they wish to send. IDRA uses the packet facade to create the actual packet, encapsulates the information in the payload and stores the resulting packet in a system wide queue. Using the neighbor table, IDRA will use the correct packet structure to encapsulate the payload. This information driven approach ensures that all information exchanges between network protocols are technology independent.

C. Support for multiple MAC protocols

In heterogeneous environments, neighboring devices sometimes use different MAC protocols. The interaction between the MAC protocol and the communication interface is also descriptive in nature. For example, a MAC protocol can describe when and how each packet is allowed to be transmitted (e.g: the maximum tolerated background noise, the scheduled sending time, the radio frequency, etc) and how the communication interfaces should be configured (listening frequency, power state, etc.). A conflict resolution scheme is used to detect conflicting settings and inform the MAC service of undesired behavior. As a result, multiple MAC services can reside on the same node, each with one or more associated communication interfaces.

D. Network service broker

Due to the information driven nature of IDRA network protocols, the network protocols do not need to implement any code for creating packets, interacting with packets or storing packets. As a result, network protocols in IDRA are significantly smaller in terms of memory requirements [26] than network protocols in traditional layered architectures. The network protocols behave similar to small services, which can be added and activated whenever required. To cope with neighboring devices that use different network protocols, network services can be dynamically added, removed or updated according to the needs of the network. As such, it is feasible to add network services on a per-need base (see Section III-D), depending on the characteristics of the neighboring devices.

Rather than having a strict execution order (such as in layered networks), network services are activated only when they needed. To his end, IDRA includes a simple service broker. Based on the packet characteristics, the service broker can identify which packet types should be processed by specialized network services. For example, proprietary packets can be processed automatically by a proprietary routing service, or a key-distribution service can be activated when a device is detected that requires secure communication. This approach shares several concepts with service-oriented architectures (SOAs) [27], where network services can be combined to reach a specific goal.

E. Protocol independent QoS support

Heterogeneous networks typically use a wide variety of quality-of-service (QoS) solutions. To ensure that IDRA network services have a small memory footprint, IDRA includes a separate module that implements all QoS solutions. The QoS module manages the queue and decides which packets should be processed first, which packets can be dropped (in case

Network service	ROM	RAM
Link level duplicate detection	460	26
Label management	294	4
AODV routing protocol [28]	2664	240
HYDRO routing protocol [29]	1924	692(+28 per route)
Positioning algorithm [30]	1584	1832
Low power listening MAC protocol [31]	822	176
Synchronized MAC protocol [32]	1126	184

TABLE I
MEMORY REQUIREMENTS (IN BYTES) OF THE CURRENTLY AVAILABLE IDRA NETWORK SERVICES.

of congestion), which network services should process high-priority packets and which packets should be transmitted first. Since all packets are stored in a shared packet queue, the QoS module can monitor all available packets. As a result, the QoS module has a clear view on the number of packets, their current processing state and their expected delay. Through the packet facade, the QoS module can read and modify the attributes of relayed packets at any processing stage. This information can be taken into account for intelligent packet selection and dropping strategies. Similarly, network protocols can request all QoS related attributes and act upon them to the best of their abilities.

IV. IMPLEMENTATION

According to [14], [33], the development of actual prototypes is crucial to prove the merit of future protocol architectures. Most of the concepts above have been implemented in a proof-of-concept architecture [26]. The following performance metrics demonstrate the feasibility of the IDRA architecture for use in heterogeneous, resource-constrained networks.

- *Memory overhead*. The memory overhead of the overall IDRA architecture (including queue management, the network service broker, the packet facade, a IEEE802.15.4 packet descriptor and an IPv6 packet descriptor) is less than 30 kB ROM and 4 kB RAM. As such, the concepts can be used even on resource-constrained devices.
- *Processing overhead*. On a resource-constrained TMoteSky device [34] using a 8MHz processor, the total overhead to process a packet is about 6-7 msec [26]. About 1 msec of this processing overhead is caused by the use of the packet facade. Packet conversion is more complex, but should only be supported in devices that border two different networks. On devices with more processing power, the processing overhead becomes negligible.
- *Network services*. By delegating common tasks, such as buffer management, packet creation, QoS management and aggregation, IDRA network services have a low memory footprint [35]. Table I gives an overview of the memory requirement of several network services that are currently available. Due to the low memory requirements of the IDRA network services, it is feasible to combine a multitude of network services, even on resource constrained nodes.

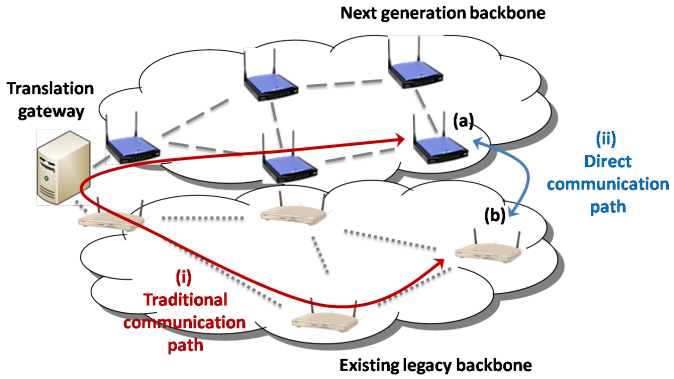

Fig. 3. The coverage of an existing legacy network is expanded by installing an additional next generation backbone using a different communication stack. By using the IDRA framework, the next generation network can converge with existing networks and use direct communication paths, thus prolonging the operational lifetime of legacy networks.

- *Feasibility*. Finally, the architecture has been evaluated in the DEUS project [36] using a large scale testbed of 200 TMoteSky nodes and in two real-life network deployments (the arts center 'Vooruit' and a home for the elderly). Devices were installed that use different routing protocols (DYMO, HYDRO or AODV) [37]. Depending on their neighbors and the packet metadata, the nodes automatically selected the appropriate routing protocol, thus enabling direct connectivity between these different devices (Figure 3). As a result, even on such a large scale, devices running IDRA are capable of efficient direct communication by using packet conversion and a dynamic network service broker on each intermediate hop.
- *Using existing network protocols in IDRA*. Converting existing layered network protocols to IDRA services typically requires two changes to existing implementations. (i) Whenever a legacy network protocol generates a packet, the protocol should hand over an information exchange to the IDRA architecture instead. (ii) Whenever a network protocol interacts with a packet, this interaction should take the form of retrieving or updating a packet attribute through the packet facade.

IDRA currently includes several network services such as routing, MAC, topology control, duplicate detection, packet identification, packet ownership, quality-of-service and localization services, which can all be combined as required by the network. In addition, network negotiation, network detection and network management services are under development. IDRA is freely available online as an open-source project at http://idraproject.net.

V. CONCLUSION

Our everyday environment is equipped with an increasing number of communication technologies, from high speed internet backbones to 'last mile' access and cable replacement technologies such as WiFi, bluetooth or UMTS. This trend

will likely not change, prompting the need for internet of things architectures that inherently cope with this diversity. This paper gave an overview of existing promising architectural approaches. However, at the moment, no solutions exist that enable efficient direct communication between co-located resource-constrained devices that use different communication technologies.

This paper argues that this gap can be filled by the IDRA architecture: IDRA enables direct connectivity between devices that utilize different network protocols and/or packet types. The following contributions of IDRA towards a future internet architecture were discussed in greater detail. (i) By decoupling the protocol logic from the packet representation, network services can interact with any packet type. (ii) IDRA copes with changing network and application requirements by introducing a dynamic service broker responsible for selecting the most optimal network services. (iii) The complexity of IDRA is low enough to implement even on low resource devices. And finally, (iv) IDRA has fully been implemented and evaluated, which proves the feasibility of the proposed concepts. As such, the IDRA architecture is a promising candidate to integrate legacy devices with network developments, or to connect heterogeneous next generation networks in a straightforward way.

VI. Acknowledgments

This research is funded by the FWO G.0298.08 and G.0291.09 grants and through the IWT SymbioNets project.

References

[1] The internet of things. ITU Internet Reports, 2005.

[2] P. De Mil, T. Allemeersch, I. Moerman, P. Demeester, and W. De Kimpe. A scalable low-power wsan solution for large-scale building automation. In *Communications, 2008. ICC '08. IEEE International Conference on*, pages 3130–3135, May 2008.

[3] Hairong Yan, Youzhi Xu, and M Gidlund. Experimental e-health applications in wireless sensor networks. volume 1, pages 563 –567, jan. 2009.

[4] K. Mayer and W. Fritsche. Ip-enabled wireless sensor networks and their integration into the internet. *In Proceedings of the First international Conference on integrated internet Ad Hoc and Sensor Networks. InterSense '06. Nice, France*, vol. 138, May 30 - 31, 2006.

[5] N.M. Mosharaf Kabir Chowdhury and Raouf Boutaba. A survey of network virtualization. *Computer Networks*, 54(5):862 – 876, 2010.

[6] S. Khanvilkar and A. Khokhar. Virtual private networks: An overview with performance evaluation. *IEEE Communication magazine*, Vol. 42(10):pp. 146–154, October, 2004.

[7] K. Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim. A survey and comparison of peer-to-peer overlay network schemes. *IEEE Communications Surveys & Tutorials*, pages pp. 72–93, Second Quarter 2005.

[8] The FP7 IST 4WARD project, "Architecture and Design for the Future Internet". http://www.4ward-project.eu/.

[9] FP7 IST 4WARD project, D2.1 Technical Requirements.

[10] Ramjee Prasad, editor. *My personal Adaptive Global NET (MAGNET), ISBN: 978-90-481-3436-6*. Springer, Signals and Communication Technology, 1st Edition (22 Dec 2009), XXXIV, 435 p., Hardcover.

[11] The itea2 usenet project, https://usenet.erve.vtt.fi/.

[12] N. M. Mosharaf Kabir Chowdhury and Raouf Boutaba. Network virtualization: State of the art and research challenges. *IEEE Communications Magazine - IEEE ComSoc,*, no. 47, no. 7:pp 20–26, Jul. 2009.

[13] N. Niebert, I. El Khayat, S. Baucke, R. Keller, R. Rembarz, and J. Sachs. Network virtualization: A viable path towards the future internet. *Wireless Personal Communications*, vol. 45(4):pp. 511–520, June 2008.

[14] Anja Feldmann. Internet clean-slate design: what and why? *SIGCOMM Computer Communication Review*, Vol. 37, No. 3.:pp. 59–64, 2007.

[15] James Roberts. The clean-slate approach to future internet design: a survey of research initiatives. *Annals of Telecommunications*, Volume 64, Numbers 5-6, June, 2009.

[16] FP7 Sensei project. http://www.ict-sensei.org/.

[17] FP7 SemSorGrid4Env project. http:// www.semsorgrid4env.eu/.

[18] David Chu, Joseph M. Hellerstein, and Tsung te Lai. Optimizing declarative sensornets. In *SenSys '08: Proceedings of the 6th ACM conference on Embedded network sensor systems*, pages 403–404, New York, NY, USA, 2008. ACM.

[19] M. D. Dikaiakos, D. Katsaros, G. Pallis, A. Vakali, and P. Mehra. Guest editors introduction: Cloud computing. *IEEE Internet Computing*, Vol. 12(5), Sep. 2009.

[20] Luis M. Vaquero et al. A break in the clouds: Toward a cloud definition. *ACM SIGCOMM Computer Communication Review, ISSN:0146-4833*, Volume 39, Issue 1:Pages 50–55, January 2009.

[21] Arsalan Tavakoli, Prabal Dutta, Jaein Jeong, Sukun Kim, Jorge Ortiz, David Culler, Phillip Levis, and Scott Shenker. A modular sensornet architecture: past, present, and future directions. *SIGBED Rev.*, 4(3):49–54, 2007.

[22] David Clark et al. NewArch Project: Future-Generation Internet Architecture. http://www.isi.edu/newarch/, 2003.

[23] Robert Braden, Ted Faber, and Mark Handley. From protocol stack to protocol heap: role-based architecture. *SIGCOMM Comput. Commun. Rev.*, 33(1):17–22, 2003.

[24] NSF FIND (Future Internet Design) research program. FIND Informational Meeting: Lessons from FIND 2006. http://www.nets-find.net/Meetings/FirstPIMeeting/FirstInfoMeeting.ppt, November 7, 2006.

[25] NSF FIND research program. Observer Panel Report 2009. Vint Cerf, Bruce Davie, Albert Greenberg, Susan Landau and David Sincoskie. http://www.nets-find.net/FIND_report_final.pdf, April 9, 2009.

[26] Eli De Poorter, Evy Troubleyn, Ingrid Moerman, and Piet Demeester. IDRA: a Flexible System Architecture for Next-Generation Wireless Sensor Networks. *Submitted to Wireless Networks, Springer*, 2010.

[27] Thomas Erl. *Service-Oriented Architecture: Concepts, Technology, and Design*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2005.

[28] Ad hoc on-demand distance vector (aodv) routing. networking group request for comments (rfc): 3561, http://tools.ietf.org/html/rfc3561, July 2003.

[29] A. Tavakoli and D. Culler. HYDRO: A Hybrid Routing Protocol for Lossy and Low Power Networks. *IETF Internet Draft, http://tools.ietf.org/html/draft-tavakoli-hydro-01*, September 10, 2009.

[30] Pieter Becue, Jen Rossey, Pieter De Mil, and Ingrid Moerman. Generic architectural framework for hybrid positioning (poster teaser). *2010 International Conference on Indoor Positioning and Indoor Navigation (IPIN), Zurich*, 30 august 2010.

[31] J. Hill and D. Culler. Mica: a wireless platform for deeply embedded networks. *IEEE Micro*, 22(6):12–24, November 2002.

[32] W. Ye, J. Heidemann, and D. Estrin. An energy-efficient MAC protocol for wireless sensor networks. In *21st Conference of the IEEE Computer and Communications Societies (INFOCOM)*, volume 3, pages 1567–1576, June 2002.

[33] NSF GENI project - Global Environment for Network Innovations. http://www.geni.net/.

[34] http://www.snm.ethz.ch/Projects/TmoteSky TMoteSky Datasheet.

[35] An overview of the currently available network services in the IDRA architecture. http://idraproject.net/protocols-and-applications.

[36] The deus project: Deployment and easy use of wireless services. http://www.ibbt.be/project/deus.

[37] Deus project leaflets: Wireless sensor network. https://projects.ibbt.be/deus.