Network Functional Composition: State of the Art

Christian Henke

Next Generation Networks Technical University Berlin
Straße des 17. Juni 135
10623 Berlin, Germany
c.henke@tu-berlin.de

Abbas Siddiqui, Rahamatullah Khondoker

Integrated Communication Systems University of Kaiserslautern
67655 Kaiserslautern, Germany
{siddiqui,khondoker}@informatik.uni-kl.de

Abstract—**Network Functional Composition is an approach for a flexible Internet architecture which decomposes the layered network stack in functional building blocks which can be loosely coupled. Functional Composition therefore enables a customized composition of functionality at the edges and in the network in respect to application specific requirements. Functional Composition is an architecture for a Future Internet which provides solutions for many of the challenges that have been identified in the Future Internet debate. Several early and current projects have addressed Functional Composition with different aspects, which have been reviewed here as part of a state-of-the-art analysis. This review shall give an overview about the different projects and categorizes the different approaches.**

I. INTRODUCTION

It is about 30 years ago that TCP/IP was developed in order to merge different networks which used heterogeneous protocols. In the early years of the Internet things were less complex, there were only a limited amount of fixed nodes, mostly under one administrative and trusted domain. The Internet is still evolving these early principles, like best-effort packet switching, end-to-end connectivity and the layered architecture. Nevertheless preliminaries from the beginning years like few number of nodes, mobility as an exception and the missing security support in the network are not appropriate today. Because of missing functionalities the Internet has been patched with new protocols, technologies and middle-boxes that even violate the design principles. This patching usually solves one problem, but evoking several others, e.g. the addresses depletion problem has been temporarily solved with Network Address Translation (NAT) and Classless Inter-Domain Routing (CIDR), but now end2end communication setup has been become more complex and routers have to cope with increasing routing tables. The complexity of the Internet has lead to an inflexible system that has limited abilities to keep up with the constantly changing demand of applications in terms of increasing data rates, reliability, QoS, client and network heterogeneity, and security.

Besides this ossification of the Internet there are also "cyberspace tussles" [1] that hinder new functionalities to be deployed in the network. Nowadays there are many stake-holders in the Internet domain with conflicting interests. ISPs demand for a profit share from over-the-top providers (like google or youtube) which base their business model upon the ISPs infrastructure. ISPs compete for customers but on the other hand they must connect and negotiate terms in order to provide end2end services for their customers. This often leads to the problem that ISPs do not have incentives to deploy new network functionalities for their customers.

Realizing that the inflexibility of the Internet stack and the problem of the "Cyberspace Tussles" may slow down the progression of the Internet there has been many research funding in new Future Internet technologies under the Um-brella of the US NSF supported GENI and the European FIRE program. Besides incremental solutions there are also "clean slate" network approaches, i.e. new paradigms that develop the Internet from scratch. One of these approaches is Functional Composition, which decomposes the layered network stack in functional building blocks and reorganizes the functionalities in a composition framework. Each of the single functional blocks fulfills a specific fine-grain functionality, e.g. forward error correction, reliable transport or forwarding. These functional blocks are then composed into composite functional blocks that will execute more complex tasks like providing a reliable connection with ordered packet delivery over a lossy network infrastructure.

II. POTENTIAL ADVANTAGES OF FUNCTIONAL COMPOSITION COMPARED TO LEGACY IP

Application specific Network Composition Through Functional Composition the network can be composed based on application specific requirements. Instead of offering a one-size-fits all "best effort" IP network, each connection will get a customized functional network which is composed based on the application requirements and the network state. The customized functional network will be assembled by different network services which provide the best service for this application under given network constraints.

Flexibility Functional Composition facilitates the integration of new functionalities into the network. Instead of tight coupling of functionalities within end2end protocols, network services are deployed on arbitrary nodes, are loosely coupled and provide their service to arbitrary other functional blocks. This design originates in the service oriented architecture approach and requires means of service description, discovery and composition. This design approach reduces management efforts and provides a flexible framework to integrate new network services.

Cross-Layer Information Exchange Cross-Layer means that functionalities can be adjusted based on the interaction between different layers. Especially for wireless and sensor networks it has been shown that cross-layer designs allow more efficient power management, increased performance, and enables better QoS support [2]. Because Functional Composition allows arbitrary composition and information exchange between network services, Functional Composition incorporates the benefits of cross-layer design architectures.

Redundancy of Functionalities Due to the layered network stack architecture many protocols on different layers implement similar functionalities (e.g. error correction, retransmission, encryption), because they are unaware of the over- and underlying protocols. With Functional Composition these functionalities may only be instantiated once within the functional network which reduces redundant overhead.

New Network Services The end2end principle of the current network declares the network dumb only mandating the network to forward packets to its destination. Nevertheless network services that are close to the customers premises like content-caches, transcoding can provide benefits for QoS intolerant and bandwidth requiring multimedia services. Also providers can offer security services by scanning incoming traffic or offering secure connection to the next trusted network. With increasing number of mobile clients these network services will become more important because the resource limited clients will be able to offload processing and storage into the network. For example in [3] the authors show how these service can be incorporated in routers.

New Business Models The new network services can also provide a new business model for network service providers. Nowadays network providers are in strong competition because they all offer the same commodity service - packet transport. Through new network services the network providers can differentiate from competitors which guarantees higher margins. The network services can be offered as premium services for customers and over-the-top service providers. In this way network providers can also benefit from applications that run over their infrastructure. Because service providers and customers can choose between the services this will also lead to a thriving evolution of network services where the best performing and efficient services will survive.

III. STATE OF THE ART - CONCEPTS FOR FUNCTIONAL COMPOSITION

Functional Composition has been in the focus of multiple projects and papers. Early approaches like Adaptive[4], DaCaPO[5], FCSS [6] already existed since the mid 1990s. These approaches concentrate more on how the protocol stack at the edges can be designed and composed. Under the Future Internet flag (GENI, FIRE) newer projects aim at a broader scope, where also services within the network should be considered and where the deployment in experimental facilities is envisioned: 4WARD[7][8] ANA[9] AutoI[10] Network Service Architecture[11][12] Net-Silo[13] RBA[14] RNA[15] Self-Net[16] TinyXXL[17], NetServ [18] and SONATE[19]. Instead of presenting all contributions one by one we first want to explain the concepts, approaches and design issues of the projects and later on give a short introduction to some projects and papers.

A. Parallel Virtual Network Stacks

One approach for functional compostion is to provide multiple virtual network stacks (netlets) that are composed at design time by a developer and then choose one netlet based on the application requirements and network situation (cf. 4WARD [7]). This composition is supported by an architecture framework that eases the management and composition of the network services. After composing the network from basic functional blocks the developer then commences test routines which evaluates the characteristics of the composed functional network. The functional network is then described by its functionalities and its characteristics. At runtime a specific functional network (netlet) will be chosen by mapping the application specific requirements and the network state with the characteristics of the netlet. The TinyXXL[17] project takes a similar approach but concentrates on sensor networks.

B. Runtime and Design Time Composition

The projects differ in their behavior at runtime. Whereas in projects like 4WARD [7] the composition of building blocks is conducted by a developer at design time and only one of the composite network services is chosen at runtime, dynamic composition of services at runtime is targeted by other projects [4][5][9][19][13]. This dynamic composition can also be accompanied by adaptations of building block parameters (e.g. compression factor of a compression service) also referred to as knobs [13]. Based upon the network state and the application requirements, specific building blocks are chosen, configured and composed to fulfill the application request. The processing step of the building blocks is called a workflow. This runtime composition targets the optimal solution for the given circumstances. Nevertheless runtime composition requires heavy processing which can e.g. be mitigated by giving building blocks a specific ordering [13], introducing pre-composed services at design time [19] or caching of prior composition work flows.

C. Autonomic Networks

Autonomic Networking targets the self-management of networks to overcome the complexity of protocols and adaptations. In the focus of multiple projects [9][16][10] are the self-* features of autonomic networking such as self-configuration, self-optimization, self-management, self-repair, and self-protection. These features are enabled through autonomic learning and cooperation principles, like self-cognitive cycles. These cognitive cycle are instantiated for network services, nodes and even network domains, that measure their

current state, and make this information available for others in the system. This measurement data will be analyzed for further processing. Based on the measurement analyses, learning algorithms and system policies an execution plan (a composition and configuration of building blocks) is derived and executed. The effects of this composition are then measured and again affect the cognitive cycle upon which a learning process will evolve.

D. Network Service Workflow Signaling

Functional Composition approaches also differ by how the signaling of the workflow to the different nodes that are involved in the communication is conducted. In today's Internet, each packet obtains an IP header with an IP address which is used by the intermediate routers to forward the packets. Functional Composition provides more functionality than pure packet forwarding, it can also incorporate in-network services offered by the network service providers. These in-network services can be of different types

1) Basic type in-network-services that are available at each node (e.g. packet forwarding)
2) In-network-services that are not available at all nodes, but the data-connection requires this service at all involved nodes (e.g. explicit resource reservation)
3) In-network-service that are only available at dedicated nodes but do only need to be passed once (e.g. for transcoding multimedia traffic)

In case in-network-services from a node are used on the path, the node needs to know how to process a packet. This can either be done with a) **in-band signaling** where the packet has attached headers that trigger specific functional blocks on the node (refer RBA [14]) or b) with **out-of-band signaling** where during connection establishment the intermediate nodes get the information how the connection flow should be handled and only a connection identifier is inserted into the packet header (e.g. Network Service Architecture [11]). Out-of-band signaling leads to a connection oriented transmission where all packets need to take the pre-configured and signaled path. Adaptation can only be done from the signaling entity which is aware of the involved nodes. In-band-signaling leaves more freedom to the network with a connection-less transmission. The nodes themselves can decide where to forward the packet and how to route to the next in-network service. Adaptations will be possible by each involved node.
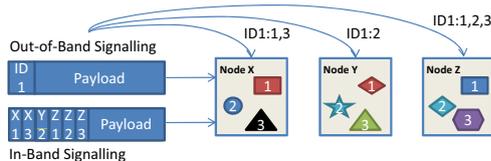

Fig. 1.   Network Workflow Signaling

IV. Functional Composition Publications

A. Early relevant publications

In the 1990s there has been substantial research on dynamic micro-protocol composition. Some of the approaches are shortly mentioned here. Although these approaches do not consider implementing additional services in the network, they demonstrate dynamic protocol configuration and composition based on application request at the end-nodes.

Coyote [20] is motivated by the challenges of mobile computing systems, like service disconnection, location and QoS support. The required functionality for mobile hosts and base stations is structured in protocols which can be implemented in a variety of micro-protocols. Micro-Protocols can then be hierarchically composed with other micro-protocols. The actual micro-protocol chosen is runtime dependent on events which can be system or micro-protocol originated.

Dynamic Configuration of Protocols (DaCaPo) [5] proposes an architecture for dynamically configuring protocols based on QoS requirements and Network Coding. The DaCaPo infrastructure is separated in 3 layers - Transport, Application and Middle C layer. The DaCaPo protocols are then processed in the C layer. The composition is based on an acyclic graph which shows the order of protocol invocation. Which direct sequence is chosen depends on the application requirements. The sequence is then confirmed with the receiver and the modules initialized.

The Function Based Communication Subsystem (FCSS) [6] also shows the decomposition of tasks in functional modules which are not aligned based on the application requirement. In [21] the authors point to a drawback of the above approaches and ask for a generic description so that the new deployment can be facilitated and the implementation customization is kept at minimum. They propose the use of the Service Description Language which is the predecessor of the current Web Service Description Language (WSDL).

B. Role-Based Architecture (RBA)

RBA was proposed in [14] and can be seen as an abstract approach to a non-layered architecture. The RBA is organized in standardized network services, which are called roles. Each role has its own roleID which reflects its functionality. In the initial concept there was only a small and limited amount of roles considered. There can be multiple roles on a single node and a role can also be abstractly distributed over multiple nodes. RBA stresses the fact that the network architecture should not have static layering and therefore the RBA introduces dynamic order of role headers in a packet header. This use of Role Specific Headers (RSH) is the main concept for role interaction and composition in RBA. The RSH semantically define the inputs and outputs of a certain role. They address roles at specific nodes in the way roleID@nodeID; in

case of a distributed role this may also be roleID@* which means that each node along the path with that roleID will process the packet. Therefore RBA uses the concept of in-band signalling. Along its path the packets RSH can be read, modified, added or deleted. A role may also contain an internal role state which can be changed by signaling through another RSH or can modify its activity depending on the set of RSH in the packet. How this internal state can be used is shown by the example of forwarding. RBA does not provide a model for routing packets, but for header processing. In a packet from A to B a role specific header will address the Forward role at all nodes on the path with additional information Forward@*, sourceID, destinationID. How the forwarding is done will be determined by another role that calculates the routing path and sends another RSH that changes the state of the Forwarding role. When a packet arrives the node must decide in what order the roles must be executed. Because some functionalities require specific ordering (e.g. Compress, Encrypt, Expand, Decrypt is not useful) there also needs to be some precedence information carried in the packets. Roles that communicate directly with each other on one node should be composable into an aggregate role, these roles are directly bounded instead of using shared data in RSH. The authors especially emphasize that RBA is a generic model, in the sense that it can be applied on top of any layer, whether all network functionality is split into roles or only the functionality above the Link, IP or even application layer is decomposed, depends on the realization.

C. Network Service Architecture

With [22] [11] [23] [24] [12] the authors present a series of publications that present an automated network service composition approach and implementation. In [11] they propose a first design for a network service architecture. A service controller (for example one for each Autonomous System, organized in a hierarchy) manages a number of network service nodes and upon connection request sets up the service processing sequence and the data transfer between the service nodes for each flow. The request is then passed to the neighboring service controller along the path to the destination so he can further setup the connection. Therefore the Network Service architecture uses out-of-band signaling. This implies that fixed routes for each flow will be taken i.e. connection-oriented packet switching. Each service node has to maintain state for each flow and each packet of the flow is processed equally. The Service Controller receives the user request and parses the information for a Mapping Algorithm. The Mapping Algorithm uses service and resource information from the service nodes (available services, memory, power, bandwidth) and tries to map the user request to service nodes. In [24] they present an distributed service routing algorithm in order to find an (nearly) optimal network path that incorporates all required services. A Service Node Interface allows the Service Controller to send control messages for connection setup to the network service nodes. Initially a network service node registers its available services at the service controller. It then receives a connection request from the service controller and

keeps track which flows require which services. It receives, processes and transmits packets for each flow accordingly. The Service node monitors resources which it sends to the Service Controller. In [23] they describe and implement a service socket API which provides end2end abstraction like the commonly used BSD TCP/IP sockets. The API consists of three methods: 1) a request function that has as input the required service specification, sets up the connection and returns a socket 2) a method that uses this socket, forms a packet and sends data to the next node 3) a receiving method which stores arriving packets in the memory. In [12] they further describe an automatic service composition approach based on semantic description of data and communication characteristics with the Web Ontology Language (OWL). They structure typical network characteristics in a tree hierarchy which can be expressed in OWL. Based on preconditions and transformations they formulate the service composition approach as an AI planning problem. The AI problem is formulated in Golog, a high level programming language which is based on situation calculus which is a logical language for reasoning. They have shown the operation of the network service architecture on the EMULAB testbed and a prototype implementation on the Cisco ISR 2800 router which provides an Application eXtension Platform (AXP).

D. Autonomic Network Architecture (ANA)

The project ANA [9] has been funded by the EU commission for a 3 year period since 2006 and realized a first implementation of a non-layered network architecture. In ANA the functionality is decomposed in network services here called functional blocks that perform the processing. Functional blocks can have arbitrary granularity; the size of a small entity or a whole network stack. Functional Blocks can be accessed via one or multiple Information Dispatch Points (IDP). Data packets are always send to a certain IDP not to the functional block itself. Having multiple IDPs for one FB enables the differentiation between different functions or states that are performed by the functional block, i.e. if IDP A is accessed then the function performs forwarding to an address A whereas when IDP B is used the information is forwarded to a different address B. An Information Channel (IC) is a functional block which is also accessed by an IDP and provides an information channel to some remote entity. ICs can be a cable, radio medium or memory, but can also be a composed service which offers multicast or broadcast. ANA uses the term compartment for different network "instances" which includes nodes with the same functional blocks. One of these compartments may also be legacy IP, but a compartment can also be bounded by different network technologies (wireless, wired), only implement one layer or multiple. Each compartment can have its own communication scheme (incl. addressing, forwarding). Nodes can be part of different compartments, as long as they include the functional blocks for these compartments. These "multihome" nodes can act as gateways for translating the communication between the compartments. An ANA Node includes a MINMEX (Minimal

Infrastructure for Maximal Extensibility) and the playground. Whereas the playground includes all functional blocks, the MINMEX provides packet forwarding between different functional blocks on the node and access to other protocols and compartments. Each functional block knows a successor IDP and, after processing, sends the data to this IDP. Dispatching from IDP to IDP or FB to FB continues until an IDP is called which provides a low-level function which forwards the packet the next node via a network interface. Different Functional Compositions are achieved by manipulation of the IDPs either bounding the IDPs to different functional blocks, deleting or adding them. For loading functional blocks and exchanging service specification ANA uses the Active Service Deployment Protocol [25], which can query the presence of services, (un-)load, (re-)configure, (de-)activate services and service composites. The composition of atomic functional blocks into higher-level service components is done by the use of service control graphs and filters [26]. The control graph is a cyclic directed graph and represents all possible processing paths through the functional blocks whereas the actual processing path is runtime determined. The actual processing path depends on the content of the message, which is matched by message filtering rules and external events (which are used for autonomic adaptation). A first running implementation of ANA can be used as standalone userspace application or as Linux kernel module. ANA has not been designed for performance more as an proof of concept. The implementation provides basic functional blocks, but also provides interoperability with legacy IP.

E. NetServ

The NSF NetServ project [18] which started in Spring 2009, follows a bit different approach to ANA and RBA Functional Composition. Whereas in ANA and RBA the nodes functionality is dependent on the packet header and content, in NetServ the service invocation is outband-signaling driven. The implementation is based on the Click Modular Router [27], an OpenSource C++ project and the OGSI Java Framework in which services are deployed. OSGI supports loading and unloading of building blocks at runtime which can be simple bundled jar files. This setup enables dynamic in network service deployment. The NetServ project concentrates on the implementation of a content distribution network (CDN) enabled by additional services implemented on top of the Click Router.

F. Net-Silo

Service Integration Control and Optimization (Silo) [13] is an US NSF project that also uses elements of fine grain functionality as reusable building blocks. Their main objective is to separate control from data functions to enable cross-layer interactions. A service is fully defined by describing the function it performs, interfaces it presents to other services and its control parameters which are also referred as knobs. Knobs can be tuned dependent on the application requirements. A method is an implementation of a service that uses a specific mechanism to carry out the functionality associated with the service. For instance. "re-sequencing" is one method for implementing the "in-order packet delivery" service. A silo is a dynamic composition of building blocks which is instantiated on a per-flow basis. A control agent which is available at each node is responsible for the composition of a silo (composing a silo refers to determining the subset of services it contains, their order in the stack, and the method implementing each service)and appropriately adjusting all the service and method specific knobs based on the available services, the applications requirements and the networks resources. An ontology, of constraints and dependencies among services, has been established for the fine-grained composition. In [28] the authors present a simplistic service composition approach for the SILO network architecture. It is a recursive approach based on pre-defined precedence constrains of services.

G. 4WARD

In the current EU Project 4WARD the virtualization of network resources are investigated. Network virtualization allows the concurrent operation of multiple network types on the same infrastructure thus providing multiple networks ("let thousand networks bloom") instead of an IP "one-size-fits-all" solution for networking. The 4WARD project wants to enable network designer to construct a multitude of composed networks (called netlets) based on basic network services at design time. A Netlet contains a composition of network services and can be seen as a container to e.g. encapsulate today's or future networks. A Netlet is evaluated before deployment in a test environment and described based on the evaluation results for different network access technologies (e.g. wired, wireless). Such a Netlet can be chosen by a selector dependent on the application requirements which will be compared with the evaluation results of all available Netlets.

H. Other Network Layer Composition Projects

AutoI The Autonomic Internet, AutoI [10] is a running European Union (EU) project which targets a self-managing communication resource overlay for secure, reliable, guaranteed and fast delivery of services. For achieving this objective, AutoI is focusing on designing and developing an open source framework for the composition and execution of better (reliable and guaranteed) services. For doing so, it emerges virtualization of network resources and policy-based management techniques to define, describe and control the internal logic of services.

Self-Net Self-Net (Self-Management if Cognitive Future Internet Elements) is an ongoing European Project which investigates approaches for autonomic self-management of the network stack. Functionality is also split in functional blocks (network elements) which are then dynamically composed [29]. The orchestration of network elements and the selection of network compartments is achieved through cognitive cycles by a Network Element Cognitive Manager (measurement, decide, execute), which empowers self-organization and

optimization by measuring the current situation.

RNA The Recursive Network Architecture [15] uses a different approach by introducing a meta-layer which is a generic protocol layer offering basic services. Each protocol layer stack will then be instantiated from this meta-layer with an arbitrary amount of necessary layers which provides flexibility to insert more functionality into the stack. The generic layer provides means for information exchange between arbitrary layers.

TARIFA Tarifa [30] is a clean slate approach which targets an architecture covering aspects of ubiquitous service provision besides Functional Composition. Services are classified into atomic and composed services. Atomic services are those individual functions commonly used in networking protocols (i.e. flow control, sequence numbers, etc). Composed services are network applications with a wider scope (e.g. printer service, directory service, instant messaging, etc). Composed services are likely to build upon atomic services and/or other composed services. Services are considered to be distributed which could be discovered and executed while passing through the route. In this approach, different negotiation schemes (i.e. 3-way and 4-way handshake) have been proposed to negotiate application request for the desired network functionality and QoS requirements.

V. Conclusion

In this paper, we reviewed potential benefits of Functional Composition as an architecture for the Future Internet. The Functional Composition approach which divides functionality into functional blocks instead of layers provides a flexible architecture which can incorporate demands of future applications. We presented a review of Functional Composition approaches up to very recent projects. This review can be a basis for other researchers in this field to find relevant work and methodologies.

VI. Acknowledgement

References

[1] D. D. Clark, J. Wroclawski, K. R. Sollins, and R. Braden, "Tussle in cyberspace: defining tomorrow's internet," *IEEE/ACM Trans. Netw.*, vol. 13, no. 3, pp. 462–475, 2005.

[2] X. Lin, N. B. Shroff, S. Member, and R. Srikant, "A tutorial on cross-layer optimization in wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, pp. 1452–1463, 2006.

[3] B. S. Davie and J. Medved, "A programmable overlay router for service provider innovation," in *PRESTO '09*. New York, NY, USA: ACM, 2009, pp. 1–6.

[4] D. C. Schmidt, D. F. Box, and T. Suda, "Adaptive: A dynamically assembled protocol transformation, integration and evaluation environment," *Concurrency - Practice and Experience*, 1993.

[5] M. Vogt, T. Plagemann, B. Plattner, and T. Walter, "A run-time environment for da capo," in *Proceedings of INET93 International Networking Conference of the Internet Society*, 1993.

[6] B. Stiller, "Fukss: Ein funktionsbasiertes kommunikationssubsystem zur flexiblen konfiguration von kommunikationsprotokollen," *GI/ITG-Fachgruppe Kommunikation und Verteilte Systeme*, 1994.

[7] L. Völker, D. Martin, I. E. Khayat, C. Werle, and M. Zitterbart, "An architecture for concurrent future networks," in *2nd GI/ITG KuVS Workshop on The Future Internet*, Nov 2008.

[8] L. Völker, D. Martin, C. Werle, M. Zitterbart, and I. E. Khayat, "Selecting concurrent network architectures at runtime," in *Proceedings of the IEEE International Conference on Communications (ICC 2009)*.

[9] "Autonomic network architecture (ana)." [Online]. Available: www.ana-project.org

[10] "Autoi project," online; accessed 15-April-2010. [Online]. Available: http://ist-autoi.eu/autoi/

[11] S. Ganapathy and T. Wolf, "Design of a network service architecture," in *Proc. of Sixteenth IEEE International Conference on Computer Communications and Networks (ICCCN)*, Honolulu, HI, Aug. 2007.

[12] S. Shanbhag, X. Huang, S. Proddatoori, and T. Wolf, "Automated service composition in next-generation networks," *Distributed Computing Systems Workshops, International Conference on*, pp. 245–250, 2009.

[13] R. Dutta, G. Rouskas, I. Baldine, A. Bragg, and D. Stevenson, "The silo architecture for services integration, control, and optimization for the future internet," in *Communications, 2007. ICC '07. IEEE International Conference on*, June 2007, pp. 1899–1904.

[14] R. Braden, T. Faber, and M. Handley, "From protocol stack to protocol heap: role-based architecture," *SIGCOMM Comput. Commun. Rev.*, vol. 33, no. 1, pp. 17–22, 2003.

[15] V. P. Joseph D. Touch, Yu-Shun Wang, "A recursive network architecture," 2006.

[16] "Selfnet project," online; accessed 10-July-2007. [Online]. Available: www.ict-selfnet.eu

[17] A. Lachenmann, P. J. Marron, D. Minder, M. Gauger, O. Saukh, and K. Rothermel, "Tinyxxl: Language and runtime support for cross-layer interactions," in *Proc. of the 3rd SECON 2006*, 2006, pp. 178–187.

[18] S. R. Srinivasan, J. W. Lee, E. Liu, M. Kester, H. Schulzrinne, V. Hilt, S. Seetharaman, and A. Khan, "Netserv: dynamically deploying in-network services," in *ReArch '09: Proceedings of the 2009 workshop on Re-architecting the internet*. New York, NY, USA: ACM, 2009.

[19] P. Müller and B. Reuther, "Future internet architecture - a service oriented approach (future internet architecture - ein serviceorientierter ansatz)," *it - Information Technology*, vol. 50, no. 6, pp. 383–389, 2008.

[20] N. T. Bhatti, M. A. Hiltunen, R. D. Schlichting, W. Chiu, and A. Chiu, "Coyote: A system for constructing fine-grain configurable communication services," 1998.

[21] B. Geppert and F. Roessler, "Generic engineering of communication protocols - current experience and future issues," in *ICFEM '97: Proceedings of the 1st International Conference on Formal Engineering Methods*. Washington, DC, USA: IEEE Computer Society, 1997, p. 70.

[22] T. Wolf, "Service-centric end-to-end abstractions in next-generation networks," in *Proc. of Fifteenth IEEE International Conference on Computer Communications and Networks (ICCCN)*, Arlington, VA, Oct. 2006, pp. 79–86.

[23] S. Shanbhag and T. Wolf, "Implementation of end-to-end abstractions in a network service architecture," in *CoNEXT '08: Proceedings of the 2008 ACM CoNEXT Conference*. New York, NY, USA: ACM, 2008.

[24] X. Huang, S. Ganapathy, and T. Wolf, "A scalable distributed routing protocol for networks with data-path services," in *Network Protocols, 2008. ICNP 2008. IEEE International Conference on*, Oct. 2008.

[25] M. Sifalakis, S. Schmid, T. Chart, and D. Hutchison, "A generic active service deployment protocol," in *In proceedings of the 2nd International Workshop on Active Network Technologies and Applications*, 2003.

[26] M. Sifalakis, "Adaptation and awareness for autonomic systems," Ph.D. dissertation, Computer Department Lancester University, October 2008.

[27] "The click modular router project." [Online]. Available: http://read.cs.ucla.edu/click/

[28] M. Vellala, A. Wang, G. Rouskas, R. D. I. Baldine, and D. Stevenson, "A composition algorithm for the silo cross-layer optimization service architecture," Mumbai, India, December 2007.

[29] D. Wagner, J. Moedeker, and T. Horstmann, "Dynamic protocol functionality in cognitive future internet elements," *Future Network and MobileSummit*, 2010.

[30] X. Sanchez-Loro, J.Casademont, J.Paradells, J. L. Ferrer, and A. Vidal, "Proposal of a clean slate network architecture for ubiquitous services provisioning," 2009.