Future Internet Architecture:
A Service-Oriented Approach

Paul Mueller, Bernd Reuther, Markus Hillenbrand
University of Kaiserslautern, 67653 Kaiserslautern, Germany
Integrated Communication Systems Lab.
{pmueller,reuther,hillenbr}@informatik.uni-kl.de

The Internet as we know it today is a success story in itself. There has always been an answer to all requirements, no matter whether caused by new applications or technologies. Applications for example demand confidentiality and QoS, management demand protection and control whereas technological progress demands integration of wireless and high speed networks. But over time the formerly simple and clear Internet architecture became a patchwork of new balconies, detours, wormholes, workarounds and bypasses.

When designing mechanisms and technology for a future Internet, one will take into account all current and foreseeable demands. But we expect that there will be never a fixed set of mechanisms and techniques fulfilling all kinds of demands. In consequence even a completely new designed future Internet will be subject to ongoing evolution. To avoid an architectural patchwork similar to today's Internet, there must be evolutionary principles allowing deliberate extensions and replacement of functionality.

We propose a service-oriented architecture (SOA) for a future Internet, because the Internet can be considered an inherently distributed software system. Hence concepts of software engineering can be applied when designing a future Internet. Service-oriented architectures define structures of loosely coupled self-contained[1] elements (i.e. services), which are well suited to build secure, dependable, flexible and adaptable software systems. We call our approach SONATE (Service oriented Network Architecture).

Lessons learned from SOA for designing a future Internet:

1. *Use self-contained services and data.* This means to find appropriate granularity of services and data chunks. For example TCP covers several functionalities which might be used independent of each other. Functionality like "sending an acknowledgment" does not make much sense without the context of a full retransmission strategy. Earlier efforts in micro protocol research may provide hints for adequate granularity of services on a network level.

2. *Use explicit descriptions instead of implicit assumptions* (fostering loosely coupled services). The current Internet is based on several "well known" protocols. But presupposing protocols (i.e. rules and formats) implicitly lay down several technical details which are hard to change afterwards. As implicit assumptions raise the grade of coupling, we propose to use explicit descriptions where ever possible: a) explicitly describe required services, e.g. "forwarding of data", "reliable transmission" or "flow control"; b) explicitly describe types of data. Structures of coherent data should identify themself, similar to the concept of Information Elements in ISDN/ATM, enabling extension and replacement of data as needed. For example seamless rewriting of addresses, adding (switching) labels or address handles will be possible.

3. *Use well defined interfaces between services* (fostering loosely coupled services). Each service must have a well defined interface, hiding its internal mechanisms and data structures. Protocols are the building blocks of the current Internet. But interfaces between different protocols – if they exist at all – do not hide protocol mechanisms. For example the BSD Socket Interface does not hide which transport or network protocols are used, this hampers exchanging protocols because adjacent layers also require adaptation.

The most interesting clean slate approaches related to the work presented here can be seen in the role-based approach (RBA) [1] conducted in the NewArch project [2] and the SILO approach introduced by Dutta et. al. [3]. The role-based approach represents a non-layered architecture to the design of network protocols and organizes communication in functional units referred to as "roles". Roles are not hierarchically organized, and thus may interact in many different ways; as a result, the metadata in the packet header corresponding to different roles form a "heap" not a "stack" as in conventional layering, and may be accessed and modified in any order. The SILO approach also introduces a non-layered design based on silos of services assembled on demand and specific to an application and network environment. Furthermore it offers a more flexible header structure than the RBA approach.

SONATE, RBA and SILO are similar approaches; all three avoid layering and aim to define a highly flexible architecture. The main motivation for RBA and SILO was to address the frequent layer violations that occur in the current Internet architecture. The SONATE approach is triggered by service-oriented architectures in the

[1] There are more properties of SOA, but loose coupling and self-containment are considered most important.

application domain; the challenge is how to implement these ideas in the lower layers of the network. In addition to RBA and SILO the definition of interfaces between services is in the focus of SONATE as well as synergistic interaction of applications with the network.

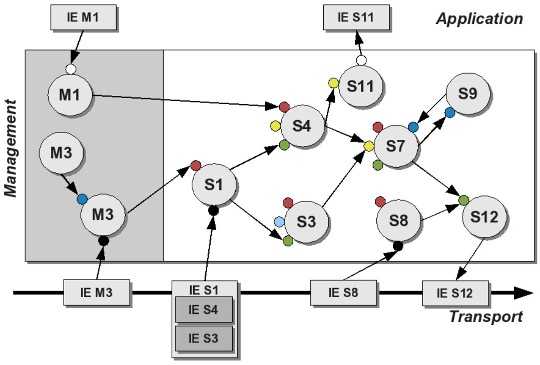

Figure 1: Scheme of interacting services with well defined interfaces using independent Information Elements.

References:

[1] Braden, R., Faber, T., Handley, M., "From Protocol Stack to Protocol Heap -- Role-Based Architecture". HotNets-I, Princeton, NJ, October 2002

[2] NewArch Project: Future-Generation Internet Architecture, URL: http://www.isi.edu/newarch/

[3] Rudra Dutta, George N. Rouskas, Ilia Baldine, Arnold Bragg, Dan Stevenson, "The SILO Architecture for Services Integration, controL, and Optimization for the Future Internet." In Proceedings of IEEE ICC 2007, June 24-27, 2007, Glasgow, Scotland.