Computer Networks 55 (2011) 987–998

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Functional composition in future networks

M. Sifalakis [b,*], A. Louca [a], G. Bouabene [b], M. Fry [a], A. Mauthe [a], D. Hutchison [a]

[a] Computing Department, Lancaster University, LA1 4WA, UK
[b] Computer Science Department, University of Basel, CH-4056, Switzerland

ARTICLE INFO

Article history:
Available online 19 December 2010

Keywords:
Autonomic systems
Functional composition
Computer networks
Future internet
Adaptability

ABSTRACT

In the future Internet it is likely that diverse service requirements will create a strong demand for the ability to modify a network subsystem's functionality, if possible at runtime and in response to customisation needs. To date, proposals for dynamic tuning of functionality have used various ad hoc techniques for cross-layer optimisation. Several frameworks have also been proposed by researchers from the active networking community; these enable incremental extension of functionality. We believe that past work has only partially addressed the goal of functional composition. In this article we propose an integrated system that is capable of functional adaptation in response to a variety of adaptation objectives. We describe a prototype implementation of our system and an evaluation of the prototype.

1. Introduction

The future internet will be information centric [1] and to an increasing extent technology agnostic. This suggests that information processing and distribution need to be robust and persistent across service (re-)configurations, and decoupled from specific data transport technologies, in order to allow seamless technology migration. Customisation of the data-plane configuration [2], federation and pooling of heterogeneous resources through virtualisation, and generalisation of service interfaces all contribute to an evolution towards the abstraction of service management from transport technologies and resources.

At the highest level of abstraction we need to satisfy the requirements of the human user of a service. This is supported at the intermediate level by a utility function for the composition of service configurations both horizontally (across the network) and vertically (within a node). At lower levels of abstraction the process is subject to constraints of access technologies and resource management policies. Technologies and policies are also subject to change, albeit over longer time scales. In such a dynamic operational environment, mechanisms that minimise constraints and maximise the customisation capacity of network subsystems will play a key role in rapid service evolution.

There has been extensive research into aspects of network service adaptation. Previous work on modular protocols such as Hutchinson and Peterson [3], Zitterbart et al. [4], and Schmidt and Suda [5] have provided solutions for *composition* of protocols at near-zero performance cost. Analogous work by Morris et al [6] delivered a modular platform for *re-configuration* of the entire data plane of a router system. Open architectures from research on active networks such as Wetherall et al. [7], Alexander et al. [9], Clayton and Calvert [10]) and "chemical" network protocol design (Meyer et al. [8]) have contributed ideas for *programming the behaviour* of a node inside the network. Finally, many proposals for cross-layer design advocate dynamic optimisation of existing network subsystems through *information fusion* between service functions across layer boundaries. This previous research has explored orthogonal aspects of adaptation in network subsystem.

The future internet poses a challenging demand for an integrated solution to adaptation. The contribution of this

* Corresponding author.
E-mail addresses: sifalakis.manos@unibas.ch (M. Sifalakis), a.louca@lancaster.ac.uk (A. Louca), ghazi.bouabene@unibas.ch (G. Bouabene), michael.fry@usyd.edu.au (M. Fry), a.mauthe@lancaster.ac.uk (A. Mauthe), d.hutchison@lancaster.ac.uk (D. Hutchison).

1389-1286/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.comnet.2010.12.006

paper lies in engineering a solution that integrates ideas, which enable different aspects of runtime customisation: (*a*) lightweight allocation of system resources to different service realms, (*b*) on-the-fly composition of data processing services at flow-level granularity (end-to-end or at an aggregate level), (*c*) modification of a node's service profile, subject to *in-band* communication, as well as *out-of-band* environmental conditions, and (*d*) on-demand information exchange between service functions to leverage optimal operation. Our focus is on the network subsystem of nodes, since that is where data path processing of packet flows is orchestrated and the highest degree of customisation can be achieved.

In this article we start by exploring related work and position our contribution with respect to past research. Next we present our prototype service composition framework. A two-part evaluation provides first a proof-of-concept validation and, second, performance indicators derived from a user space implementation. This evaluation provides evidence for the viability of our system. A summary and outlook on future work conclude the paper.

2. Related work

In this section we review a representative sample of the substantial body of literature that has addressed the adaptability of networked systems. In so doing we identify key ideas and approaches developed to date, and we relate or compare them with our work.

Architectures such as the x-kernel [3], FCCS [4], and more recently the Click router [6] have demonstrated the flexibility of modular design in customising and re-composing the entire data plane. These early approaches represent composition of services as a graph that interconnects simple modular functions. In our work we use a composition graph for the *declaration* of a cooperation context for service functions, but not for explicitly binding them together. Instead, in our system, service modules execute as independent components than can interact reactively and asynchronously through message-passing. The composition graph describes a composable classifier that executes in an independent component, which schedules and orchestrates interactions among other components. This is a more elaborate approach compared to the simple linking of function calls in a strictly ordered fashion as in the x-kernel and Click. It also allows runtime resource control.

Tau, by Clayton and Calvert [10] is a customisable system that uses dynamically loaded service components and late binding for composing data plane services. Service functions register event handlers and a byte-pattern, which is to be found in a message. The registered events are invoked when a message with the pattern of interest arrives at the destination transport layer. The service functions can be independently scheduled for execution (potentially in parallel). This design has several aspects in common with our work; such as the filter mechanism for associating messages (as events) with service functions, and the internal architecture of the system that resembles our functional composition manager. However, our graph-based classification engine provides more effective orchestration of the runtime composition process than the simple First-In-First-Out (FIFO) event scheduler of Tau, by supporting partial ordering, mutual exclusion, and conditional scheduling of components based on out-of-band or cross-component conditions. A second important difference is that Tau provides customisation of services at the transport layer only, while we consider the complete range of functions in the node's network subsystem.

In Switchware/PLAN [9] the node's network fabric can be configured by packets that carry λ calculus programs and which, upon evaluation, compose on-the-fly control plane services from dynamically loaded library routines. Analogous, yet simpler, functionality is provided by the systems in [7,11]. This idea, as well as the fusion of control and data planes, has been further developed in our framework. In our work all functions are provided by components: the functions that compose the node services, the functional composition process, and resource management. Components communicate by exchanging messages, providing a unified interaction paradigm across the network as well as within a node. This allows running services to have recursive (self-)access to resource management and service composition, through messaging, for re-configuring node behaviour.

A constraining aspect of most active network solutions has been the confinement of possible interactions between service functions within the data path. Packets in the data path serve as the sole shared facility for multiplexing functions, exchanging information and providing some level of synchronisation. This is the general constraint that many cross-layer designs try to overcome by defining ad hoc out-of-band interfaces. "Chemical" network design [8], which emerged from earlier work on active networks [12], allows for interactions between service modules, albeit in a statistical way, to be expressed as by-products of chemical reactions. By contrast, in our work we provide a number of more structured ways for explicit interactions between service components, both in-band (to the data-path) and out-of-band. For the in-band, a typical function call-like meta-interface is provided, through which components exchange both data and any additional parameters. For the out-of-band, service components have the option to advertise control-plane interfaces that can be used by other service components in a publish-subscribe style. The combination of these two approaches addresses the requirements for inter-component feedback control identified by most (if not all) cross-layer frameworks in the current literature (e.g. [13–16]).

The ideas discussed here have been integrated in our framework, which also further develops core functionality from some of our earlier work in [17]. The notion of on-the-fly flow-selected service composition from [17] has been retained in the functionality of a special component, providing a *routing* service for a network of service components. Multiple routers of this kind can be used to create an *inter-network* of composite services enabling multi-level service composition and progressive service abstraction. A service component may modify, at runtime, the service composition by communicating with any of these *router* components to perform control or data plane operations.