

A Hierarchical Node Management System for Application-tailored Network Protocols and Architectures

Hans Wippel, Thomas Gamer, and Denis Martin
Karlsruhe Institute of Technology, Germany
{wippel, gamer, martin}@kit.edu

Abstract—In many Future Internet visions, a node is assumed to be connected to a multitude of possibly virtualized networks, each being optimized for a specific application. Within such a network, application-tailored network architectures and protocols provide communication services between the nodes. In order to handle such a number of different architectures and protocols, new management paradigms need to be considered, allowing for a flexible and extensible Future Internet. In this work, we therefore propose a management model for the Future Internet that provides flexibility and extensibility regarding node management while mastering complexity by a hierarchical management structure that allows for information aggregation and task delegation.

I. INTRODUCTION

Future Internet initiatives like 4WARD [1], G-Lab [2], and FIND [3] consider a Future Internet scenario in which nodes simultaneously connect to multiple, potentially virtual networks. These networks are mostly tailored to specific use cases and applications and, thus, require optimized protocols and network architectures. Composition constitutes a feasible technique that, e.g., simplifies development of application-tailored protocols. With protocol composition, new protocols are created by combining individual building blocks, each implementing a specific self-contained functionality. In order to run a multitude of composed protocols concurrently and to allow for extensibility by adding new protocols, the network stack of a Future Internet node may look as outlined in Figure 1.

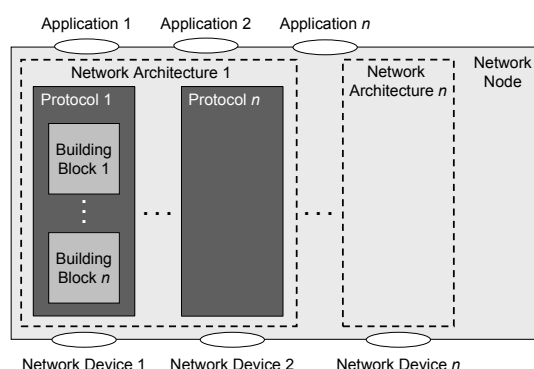


Figure 1. Network stack of a Future Internet node.

Using different network devices, a node is attached to

multiple networks, which are possibly based on different network architectures. Every network architecture contains a set of protocols with a common understanding of, e.g., message formats and addressing schemes. Instead of selecting a specific protocol directly, an application should only need to provide generic communication requirements via an augmented application interface—e.g. requesting a reliable service or a certain quality-of-service. The system then selects suitable protocols based on these requirements and the current network properties. The combination of protocol composition and a framework implementing such functionalities provides flexibility and extensibility necessary for a Future Internet.

The SILO network architecture [4] introduced the concept of management and optimization interfaces (*tuning knobs*) for building blocks (BBs). ANA [5] proposes a network architecture that allows for grouping of BBs into compartments. In the context of management, monitored information provided by BBs can be aggregated by an orchestration BB and handed to interested BBs [6]. INM [7] similarly groups building blocks into management domains and proposes the concepts of co-design (designing services with their management functionalities) and co-location (grouping services and their management functionalities).

Based on these ideas, we want to introduce node management that, e.g., allows to manage resources or to adapt parameters across a multitude of different network architectures and protocols. Such a flexibility poses a serious challenge on the management of a network node. The possibly huge amount of concurrent protocols and network architectures renders manual node management unfeasible. Additionally, new network protocols and architectures might be instantiated on a node at runtime. These protocols and architectures may be optimized for previously unknown use cases. Thus, they possess unknown properties and require unforeseen management mechanisms. For these reasons, we propose a *hierarchical self-management system* for network nodes, which minimizes human interaction, provides flexibility and extensibility, and reduces complexity by aggregating management information and delegating management tasks along the proposed management hierarchy.

II. HIERARCHICAL NODE MANAGEMENT SYSTEM

Before detailing the management model our hierarchical management system is based on, we first define node management and its related management tasks. Finally, we shortly discuss our next steps and give an outlook on future work

A. Node Management

Having already described the main requirements for node management in a Future Internet—flexibility, extensibility, and controllable complexity—we define the necessary node management tasks in this section. Management in general comprises many different aspects, ranging from collecting information, based on which actual management decisions are taken, to the enforcement of those decisions by appropriate actions. Thus, the following definition considers all those management tasks that, in our opinion, a Future Internet management system has to be capable of:

a) *Monitoring*: Monitoring comprises collecting information and statistics about resource utilization, network properties, and protocol details. This information consists of generic statistics like memory, CPU, and network resource usage (which do not require detailed knowledge about architectures and protocols) and statistics like, e. g., lost and corrupted messages or violations of the protocol logic (which require knowledge about the actual protocols and architectures).

b) *Policy Management*: Policies regarding nodes, network architectures, and protocols can be set by end users of a network node in order to enforce, e. g., saving of energy resources on a mobile node.

c) *Event Management*: Event registration and handling are part of Event Management. Events allow for information exchange between components of the management system and execution of custom adaptation and monitoring mechanisms provided by respective protocols and architectures.

d) *Resource Management*: Administration and admission of network, processing, and storage resources are part of Resource Management. Such resources are, e. g., capacity for data flows, new processing threads, or hard disk space. In case of resource shortage or usage violations, resource allocations are adapted or—in the worst case—withdrawn.

e) *Anomaly Detection*: Anomaly Detection analyses protocols and network properties for anomalous behavior, e. g., to detect node and network failures or malicious attacks.

f) *Adaptation*: Adaptation modifies protocol parameters in such a way that varying network properties, like packet loss rate, are compensated and proper protocol operation is maintained according to the application's initial requirements. In addition, adaptation includes the tuning of protocol parameters in order to optimize protocols, e. g., to save bandwidth or other resources on network nodes.

Tasks such as adaptation require a rather deep knowledge of a protocol's parameters and behavior. In a Future Internet with a large number of protocols, such fine-grained tasks cannot be defined at a high abstraction level, i. e., without knowledge about the precise protocol. On the other hand, resources must be shared across architecture implementations that are possibly

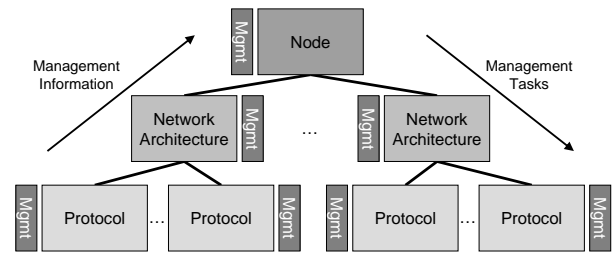


Figure 2. Hierarchical Management Model

completely isolated from each other—which suggests a generic approach for, e. g., resource management and monitoring.

B. Management Model

Based on the mentioned requirements, the previous analysis of management tasks, and the node structure outlined in Section I, we propose the hierarchical management model depicted in Figure 2. On each level of the node structure, dedicated management components are located that achieve the management tasks described previously. On the lowest level a *Protocol Management* component exists for each protocol. This component's management tasks need specific knowledge about the managed protocol. For example, monitoring on this level collects statistics of individual data flows and protocol parameters, adaptation modifies protocol parameters, and anomaly detection can check for inconsistencies of the protocol logic or unusual traffic patterns in individual data flows. Since with our hierarchical approach such detailed protocol knowledge only has to be available in the Protocol Management, easy interchangeability or addition of protocols is guaranteed—assuming that the new protocols come with their own Protocol Management components.

On the second level, there is an *Architecture Management* component in each network architecture supported by the node. The Architecture Management has detailed knowledge about the network architecture only—protocol knowledge is not available here. At this level, e. g., monitoring gathers information about network devices or architecture details, adaptation can modify device parameters or the data rate available for individual protocols, and anomaly detection can detect unusual usage patterns of network devices. Keeping architecture specific knowledge in the Architecture Management allows for modification or addition of architectures without modifying components in the upper level.

Finally, a single *Node Management* component is located on the highest level performing management with focus on the entire node. This component does not need detailed knowledge about specific network architectures or protocols, which are in use by the node. On this coarse-grained node level, for example, monitoring collects information about resource utilization of all the network architectures. In addition, adaptation may restrict resources available to an architecture, a policy could be enforced to save energy on the node, and anomaly detection

could be used to detect attacks against the node across different architectures.

In order to reduce complexity of the entire node management as well as to provide flexibility and extensibility regarding the management, each management component aggregates its *management information* as depicted in Figure 2. This provides a self-contained view on the component, which is passed up to the management component of the next level. This approach ensures that a management component only needs detailed knowledge about its own level. As an example, Protocol Management components monitor information about resource usage of individual data flows of a protocol—thus, information is rather fine-grained. This information then is aggregated resulting in the complete network resource utilization of the protocol, which is subsequently passed on to the Architecture Management. The Architecture Management collects the aggregated information from all protocols of its network architecture. In this way, it gets complete information on the network resources currently being in use. Again, this information is aggregated and passed on to the Node Management, which collects the resource utilization of all network architectures and, thereby, obtains information about the resource usage of the entire node.

In contrast to the management information, which is aggregated upwards, *management tasks* are delegated downwards along the management hierarchy. This way, detailed knowledge about network architectures and protocols necessary for the execution of management tasks, e. g., adaptation of protocol parameters in order to meet application requirements, is only needed in the components of the corresponding management level. This again reduces complexity and ensures high flexibility and extensibility.

C. Next Steps and Future Work

As future work we are planning to implement our node management system, which is based on the proposed hierarchical management model, into an existing Future Internet framework—the *Node Architecture* framework [8]. The resulting architecture, i. e., the framework including the components of our management system is outlined in Figure 3.

Individual protocols in the Node Architecture are encapsulated in *Netlets*. Netlets are divided into two parts: The protocol implementation contains the actual protocol logic, which may be composed out of multiple building blocks (BB). The *Netlet Management* component implements the Protocol Management, i. e., it is responsible for management tasks inside the Netlet. Within this framework, *Network Architectures* are implemented as a set of Netlets using the same multiplexer, which is responsible for functions common to all protocols of the architecture such as framing or forwarding based on addresses. Each network architecture contains an *Architecture Management Netlet* that copes with architecture-wide management tasks. Finally, a *Node Management* component is responsible for management tasks across architectures.

This implementation will allow for evaluation as well as for further enhancements of our approach. By implementing

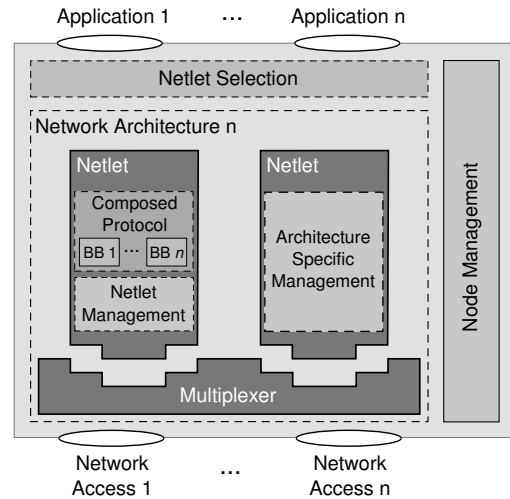


Figure 3. Management System Integration into the Node Architecture

and evaluating simple use cases, we expect to gain practical insights into (1) the design of management interfaces, (2) the definition as well as description of monitoring information and tasks, (3) the aggregation of monitoring information, and (4) the delegation of management tasks downwards along the management hierarchy.

ACKNOWLEDGMENT

This work was carried out in parts within the research projects 4WARD and G-Lab which are funded by the European Commission (within 7th framework programme) and the German Ministry of Education and Research (BMBF) respectively.

REFERENCES

- [1] “4WARD Project Homepage.”, URL: <http://www.4ward-project.eu/>
- [2] “G-Lab Project Homepage.”, URL: <http://www.german-lab.de/>
- [3] “FIND Initiative Homepage.”, URL: <http://www.nets-find.net/>
- [4] R. Dutta, G. N. Rouskas, I. Baldine, A. Bragg, and D. Stevenson, “The SILO Architecture for Services Integration, control, and Optimization for the Future Internet”, in *Proc. of IEEE International Conference on Communications (ICC)*, Glasgow, Scotland, UK, Jun. 2007, pp. 1899–1904.
- [5] A. Keller, T. Hossmann, M. May, G. Bouabene, C. Jelger, and C. Tschudin, “A System Architecture for Evolving Protocol Stacks”, in *Proceedings of 17th International Conference on Computer Communications and Networks (ICCCN)*, St. Thomas, US Virgin Islands, Aug. 2008.
- [6] M. May, M. Siekkinen, V. Goebel, T. Plagemann, R. Chaparadza, and L. Peluso, “Monitoring as first class citizen in an autonomic network universe”, in *Proc. of 2nd Conference on Bio-Inspired Models of Network, Information and Computing Systems (Bionetics)*, Budapest, Hungary, Dec. 2007, pp. 247–254.
- [7] A. Gonzalez *et al.*, “In-network management design”, 4WARD Consortium, Project Deliverable D4.3, Jan. 2010.
- [8] L. Völker, D. Martin, I. El Khayat, C. Werle, and M. Zitterbart, “A Node Architecture for 1000 Future Networks”, in *Proceedings of the International Workshop on the Network of the Future*. Dresden, Germany: IEEE, Jun. 2009.