depending on the communication latency and physical model of sensed values (given as differential equations).

Runtime Environment - The framework is backed by jDEECo, an implementation of DEECo in Java. This runtime environment includes scheduling of component processes, dynamic grouping of components into ensembles, and distributed knowledge exchange. Technically, jDEECo employs gossip-style network communication to uniformly address IP-based, as well as peer-to-peer broadcast-style WPAN networks.

Case Studies and Evaluation

To date, DEECo has been a successful part of the EU FP7 IP project ASCENS and employed in a number of case studies, including intelligent vehicle navigation, emergency coordination, and ad-hoc cloud deployment. Case studies have confirmed that DEECo represents a significant development simplification while preserving robustness and dependability properties of the designed system.

Future Work

In the next research and development steps, we intend to focus on enhancing the framework of efficient communication means in situations of limited connectivity, handling uncertainty of knowledge, and verification in the presence of dynamicity.

Links:
http://www.d3s.mff.cuni.cz
https://github.com/d3scomp/JDEECo

References:
[1] K. Beetz and W. Böhm: "Challenges in Engineering for Software-Intensive Embedded Systems", in Model-Based Engineering of Embedded Systems, Springer, 2012, 3–14
[2] T. Bures et al. "DEECo – an Ensemble-Based Component System", in proc. of CBSE'13, ACM, 2013, 81–90
[3] J. Keznikl et al. "Design of Ensemble-Based Component Systems by Invariant Refinement," in proc. of CBSE'13, ACM, 2013, 91–100.

Please contact:
Tomáš Bureš
Charles University in Prague, Czech Republic
E-mail: bures@d3s.mff.cuni.cz

The Software-Defined Network Revolution

by Marco Canini and Raphaël Jungers

Thanks to the introduction of Software-Defined Networking (SDN) [1], it is becoming possible, for the first time since the early days of computer networks, for operators to design and implement their own software in order to operate and customize the network to specific needs. But this comes with challenges involving many different disciplines.

Computer networks, such as the Internet, are playing an increasingly important role in our society. Despite this, they are too often designed and managed using ad-hoc techniques that are not as mature as other engineering and scientific disciplines. This is largely a consequence of the dependence on proprietary technologies and lack of open interfaces. Today's networks consist of various types of switches, routers and packet-processing middleboxes. Switches and routers are architecturally composed of two components: a data plane and a control plane. The data plane handles line-rate forwarding of packets that arrive at the device. The control plane handles the logic needed to configure the forwarding rules in the data plane. Traditional switches and routers have proprietary firmware and control logic implementations. This not only inhibits vendor interoperability, but also hampers flexibility, as operators cannot introduce new control plane functionality or protocol into a switch.

With the recent emergence of Software-Defined Networking (SDN), the networking industry is on the verge of a profound transformation. SDN is an emerging paradigm that aims to alleviate these issues by decoupling the data and control planes. SDN outsources the control of packet-forwarding switches to a set of software controllers running on a server cluster. This enables operators to run third-party software or create their own to build networks that meet their specific, end-to-end requirements, such as fine-grained policy enforcement, Quality of Service, and, of course, efficiency [2]. In SDN, control applications operate on a global, logically-centralized view, achieving a higher level of abstraction for managing networks. This view enables simplified programming models to define a high-level policy (i.e., the intended operational behaviour of the network) that can be compiled down to a collection of forwarding rules and installed in the data plane. This is in stark contrast to today's manual, error-prone, ad-hoc approaches.

As such, SDN promises to radically change the way networks are managed and operated. Indeed, for the first time since the early days of computer networks, it is becoming possible to enable network operators to design and implement their own network-controlling software and develop new functionality and services for end-users. A pioneering initiative in this direction is the OpenFlow protocol [3], which allows a software controller to dynamically manage the state and the behaviour of network elements.

However, while offering unprecedented potential, SDN is still in its infancy and much research effort is needed to better define its foundations: What are the abstractions that promote a simple yet expressive and efficient network-programming model? How will these abstractions yield efficient implementations that support rapid reactions to unplanned network events? Is it practically feasible to devise efficient abstractions that may still lend themselves to automated verification of integrated network systems? For instance, is it possible to prove that a network system satisfies key properties that would

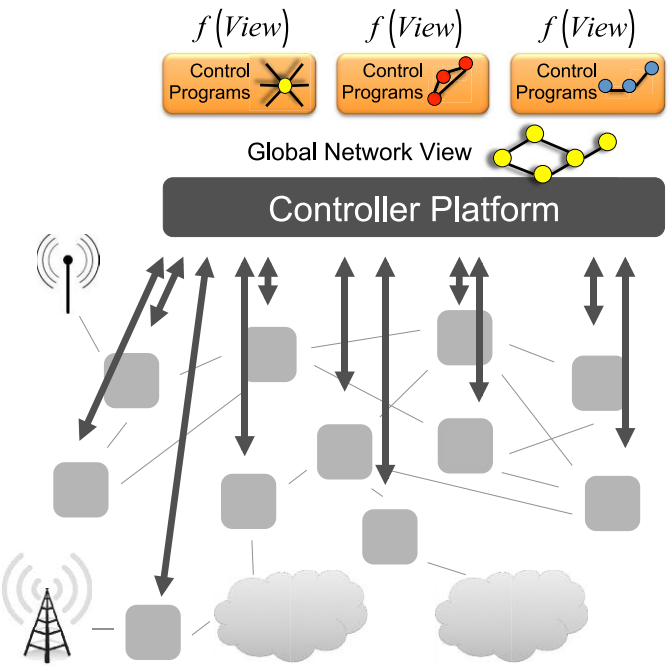

Figure 1: Example Software-defined network architecture: the control plane is separated from the data plane and control applications operate on top of a global (possibly virtualized) network view.

ensure smooth network operation? How to design a logically centralized (i.e., physically distributed) control platform that adequately and cost-effectively provides the required levels of availability, responsiveness and scalability? What control functionality can we keep distributed, what should we centralize? How can we achieve optimal dynamical resource allocations and fairness among users? Also, what methodologies can we devise to troubleshoot these systems when things in practice do not work according to theory (e.g., due to hardware malfunctioning)? Finally, what strategies should we use to deploy SDN into existing networks? We believe that these questions require us to apply principles and techniques from a large body of different disciplines, including Operations Research, Control Theory, Distributed Systems, Programming Languages, Software Engineering, Security, and Embedded Systems.

As such, SDN systems share many challenging technical problems with other Cyber-Physical Systems (CPS): (i) they have to cope with human and unpredictable demand, (ii) they are akin to a Systems-of-Systems structure comprising a two-tier distributed system, in which each node has heterogeneous processing capabilities and a large part of their computational resources are embedded (e.g., router firmware, NPUs, FPGAs), and (iii) constraints from the physical world are strong and various (e.g., link bandwidths and latencies, or failures due to random errors or malicious behaviours). On top of that, SDN systems constitute an opportunity for other CPS that might need to rely on a programmable and modular network of computing and forwarding resources. Perhaps even more than for other CPS, SDN seems to have only one limitation to our ability to control them: our own imagination.

At UCLouvain, we have built a team centred on the new SDN paradigm in order to leverage its possibilities and develop the new generation of computer networks. Our goal is to build a rigorous and systematic methodology for designing and operating SDN systems, and we are convinced that the only feasible approach is to combine advances in Electrical Engineering, Computer Science and Applied Mathematics, ranging from computer networking to software and requirements engineering, optimization and control theory. Our research group includes faculty, students, and post-docs from various backgrounds in these disciplines. More details on this project are available at our website: http://sites.uclouvain.be/arc-sdn/. PhD and Post-doc positions are available within our team.

Link:
http://sites.uclouvain.be/arc-sdn/

References:
[1] N. Feamster, J. Rexford, E. Zegura: "The road to SDN: An intellectual history of programmable networks," ACM Queue 11, 12 (December 2013)

[2] "Ethane: taking control of the enterprise," in Proceedings of ACM SIGCOMM, 2007

[3] N. McKeown et al.: "OpenFlow: enabling innovation in campus networks," ACM SIGCOMM Comput. Commun. Rev. 38, 2 (March 2008), 69–74.

Please contact:
Marco Canini and Raphaël Jungers
Université catholique de Louvain, Belgium.
E-mail: marco.canini@uclouvain.be, raphael.jungers@uclouvain.be