Is SDN the De-constraining Constraint of the Future Internet?

Jon Crowcroft
Computer Laboratory
University of Cambridge
Cambridge, UK
jon.crowcroft@cl.cam.ac.uk

Markus Fidler
Institute of Communications Technology
Leibniz Universität Hannover
Hannover, Germany
markus.fidler@ikt.uni-hannover.de

Klara Nahrstedt
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, USA
klara@cs.uiuc.edu

Ralf Steinmetz
Multimedia Communications Lab
Technische Universität Darmstadt
Darmstadt, Germany
ralf.steinmetz@kom.tu-darmstadt.de

This article is an editorial note submitted to CCR. It has NOT been peer reviewed.
The authors take full responsibility for this article's technical content. Comments can be posted through CCR Online.

ABSTRACT

Dagstuhl hosted a three-day seminar on the Future Internet on March 25-27, 2013. At the seminar, about 40 invited researchers from academia and industry discussed the promises, approaches, and open challenges of the Future Internet. This report gives a general overview of the presentations and outcomes of discussions of the seminar.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Packet switching networks; C.2.6 [**Internetworking**]: Standards

Keywords

Future Internet, Network Design, Testbed, Software Defined Networking, Virtualization, OpenFlow

1. INTRODUCTION

This article reports on a seminar on the Future Internet held at castle Dagstuhl on March 25-27, 2013. The seminar focused on topics in the areas of prescriptive network theory, experimentally-driven research, SDN, virtualization, and OpenFlow.

During the past few years, the vision of a Future Internet created significant momentum resulting in numerous Future Internet initiatives and projects, such as FIND (NSF, USA), GENI (NSF, USA), FIRE (EU), and G-Lab (BMBF, Germany). Beyond technological innovations, Future Internet research called the Internet architecture, it's mechanisms and protocols, and it's evolution into question, giving rise to important discussions of the Internet design fundamentals. While the debate on clean-slate versus evolutionary approaches attracted considerable attention, a definite answer may not be easily obtained as there is little formal basis to reason about network architectures: There is no theory from which the 'right' network architecture can be derived nor is there a benchmark that helps understanding how close to or far from optimal the current Internet is. While it may be difficult to obtain a theory, that is generative in the sense that it facilitates creating a network architecture, its relevance justified considering the potentialities of such a 'prescriptive network theory' at the first day of the Dagstuhl seminar.

Complementary to network theory, the development of large-scale Future Internet testbeds[1] systematically supports experimental research. Well-known use cases of testbeds are to verify solutions via implementation and prototypes, to engineer details, or to approach weakly understood problems. We used the second day of the seminar to elaborate what are the aims of testbeds in Future Internet research. The main questions asked were:

- Do we gain knowledge from current testbeds to advance Internet architecture, protocols, and algorithms? What is the experimental discovery process?
- What are requirements for Future Internet testbeds?
- Can we use current testbeds for Future Internet research or do we need new testbeds? Is a general purpose testbed feasible?

Having touched upon a number of basic and potentially long-term research questions, day three of the seminar focused on the concrete technologies of SDN, virtualization, and OpenFlow. These approaches have the potential to transform the network architecture and, as OpenFlow implementations emerge and (programmable) midboxes become more and more popular, it seems that the process is fully under way. While numerous options for SDN, e.g., on which layer to operate, are subject to current research, a collection of use cases already showed that SDN can achieve large flexibility, interestingly on the layers above and also below the SDN implementation.

In the following we will highlight a number of research questions and viewpoints that emerged in talks, discussions,

[1]The Internet started as a testbed, the ARPANET. Since then, there have been many follow-ups that have been built for various targeted or more general research purposes such as the Terrestrial Wide Band network for Distributed Simulation Internet, the DARTNET for Multicast and Diff-Serve, Internet II for new Internet Services, PlanetLab for distributed application research at Internet scale, and Emulab for large scale realistic emulation, just for a few examples.

and in group work at the seminar. We do not attempt to provide a complete summary of the seminar. For further reference, the program, abstracts, and slides of presentations can be found online [2].

The seminar started off with a one minute madness session where all participants were asked to provide a short statement on the Future Internet, e.g., the biggest challenge, expected paradigm shift, or most promising research topic. The outcome was a list of (as expected) diverse, yet highly inspiring views. Since we cannot reproduce all statements, we clustered them to highlight a few that we find to be representative:

- Following an age of bandwidth we are facing the age of latency pressure [Max Mühlhäuser].
- Data exchange is wasteful. We need self-organizing overlays. The Internet has become today's mainframe [Jörg Liebeherr].
- Ossification is not the issue. Devices in the network restrict innovation at the application level [Laurent Mathy].
- The biggest problem the Internet is facing: (all too often, at times, ...) deployment is ad-hoc and irreversible [Markus Fidler].
- The biggest challenge is to include security by design [Rastin Pries].
- The fundamentals are done. How do external factors (social, economic, ...) affect solutions? Look outside the networking domain [Ruben Cuevas-Rumin].

Further aspects that were raised several times concerned the design of on-demand, flexible, adaptive, and future-proof networks, multi-domain problems, and deployment issues. While there was agreement that patience is required as deployment may take up to several decades, it was felt that we are lacking means to predict the timeline as well as the deployability in general.

In the following sections we will focus on each of the three topics of the seminar: prescriptive network theory, experimental research, and SDN, virtualization, and OpenFlow.

2. PRESCRIPTIVE NETWORK THEORY

The term 'prescriptive network theory' is not commonly used and it may be the lack of such a theory that explains fundamental uncertainties in network design. A prescriptive theory, as opposed to a descriptive theory, is concerned with what should be, rather than with what is. Prescriptive theories are often viewed as high-level guidelines or rules, e.g., for the design of systems. Formally, a prescriptive theory is axiomatic. As such, it is a framework to derive certain conclusions, yet, it cannot be proven in itself.

An important observation in our field is that we are much more concerned with the specifics of the current Internet than with networking as a discipline in general [3, 7]: classes focus on protocols rather than on principles; backwards compatibility and incremental deployment are issues; major network theories, such as teletraffic theory, graph theory, or formal methods, are mostly descriptive and retrospective. Finally, we find ourselves in a dilemma: "We build what we measure. Hence, we are never quite sure whether the behavior we observe, the bounds we encounter, the principles we teach, are truly principles from which we can build a body of theory, or merely artifacts of our creations" [3]. Not surprisingly, numerous basic architectural questions about networking are still unsolved today, e.g., on protocol decomposition, the combination of protocols, or the waist of the protocol stack [6].

Knowing the difficulties a prescriptive network theory is facing, we dedicated the first day of the seminar to this topic to gather viewpoints, potential candidates, and if nothing else to create awareness. After an "introduction to prescriptive network theory" by Markus Fidler, three 'seed' talks were given to spark discussion in group-work: "a formal model for network communication mechanisms" by Martin Karsten; "multi-mechanism adaptation for the Future Internet" by Ralf Steinmetz; and "prescriptive network theories" by Jon Crowcroft. In the sequel we summarize some of the main results that emerged from presentations and group work.

Qualities There is a general agreement that a single, all-embracing network theory may not exist. While neighboring disciplines, such as signal processing or information and coding theory, each resorts to a comprehensive body of theory, that guides the design of systems or at least acts as a benchmark, networking draws conclusions from an interdisciplinary variety of different theories and methods, see [7] for an overview. Furthermore, the expectation at the seminar was that defining a prescriptive network theory is highly challenging, making it an ambitious long-term goal. Ideally, a prescriptive theory is generative, in this way it has a different quality than a descriptive theory, and it is rooted in a set of well-defined axioms, like [5], that enable the formal derivation of conclusions. In this respect, a prescriptive theory differs from design patterns that formalize an innovation before it is scaled out. On the other end of the spectrum, a prescriptive theory can be understood as a set of less formal (justified and maybe agreed upon) high-level guidelines for the design of networks, that may be obtained more easily. The prospects of a more or less formal prescriptive network theory were found to depend largely on the complexity of the problem. Generally, the methodology to approach abstract, architectural questions seems to be much less developed, whereas a rich variety of formal methods exists for specific problems (some of which are solved with considerable success, others are still open).

Multi-mechanisms The term multi-mechanism refers to a set of mechanisms that serve the same functional purpose and hence may be interchangeable with each other (the mechanisms may nevertheless differ in non-functional qualities such as throughput, delay, reliability, or security). Interestingly, it seems that in network protocols the number of corresponding mechanisms is readily comprehensible, e.g., consider the set of basic mechanisms (not their implementation and their policies) for medium access control or for congestion control. Moreover, fundamentally new mechanisms are rarely discovered, suggesting that most basic mechanisms are known by today [3]. Their composition, on the other hand, is much less understood [6] and offers significant opportunities for the Future Internet. Concluding, the notion of multi-mechanisms can facilitate a paradigm shift towards a dynamic and adaptive on-demand composition of mechanisms.

De-constraining constraints David Alderson and John Doyle have coined this phrase [1]. Originally, they used

the idea to explain how the narrow waist of the hourglass figure, often used to visualise the protocol stack for the Internet, represents a constraint that frees up the design space for the lower and higher layers to evolve freely. The analogy is drawn from the nature of DNA and how this freed up biological systems to develop great diversity, constrained merely by fitness to their environment. While traditionally IP was the waist of the hourglass, alternatives emerge above as well as below IP, e.g., HTTP, CDNs, or virtualization [6].

Middleboxes Recent studies show a significant proliferation of middleboxes, such as NATs, firewalls, rate limiters, and proxies, resulting in an Internet that is a layer 4+ interconnection of IP networks. Bypassing, e.g., tunneling over HTTP (TCP port 80), is an unfavorable option and the risk that middleboxes may restrain applications is evident. On the other hand, middleboxes bear great potential for innovation and may be viewed as a de-constraining constraint that has the potential to combine, e.g. DTN, CCN, and IPng.

Latency During the seminar, latency was identified as a major, cross-cutting issue for the Future Internet. It was argued that a prescriptive theory has to include temporal aspects and the need for a 'Shannon theory' of network delays was recognized. While switching and transmission delays are addressed by adding (processing, transmission) capacity that is made available by the technological advancement, propagation delays are ultimately fixed by the laws of physics. Interestingly, delays including propagation delays may also be considered from an architectural viewpoint: since light cannot be sped up, distances may be reduced by caches, proxies, or in-network processing in general; requests can be duplicated and sent out in parallel to combat jitter; the redundancy of paths and options provides choices; and protocol handshakes can be omitted to trade other qualities such as security for delay. Finally, queueing delays can be mitigated by schedulers and limited buffers, where significant results from queueing theory, effective bandwidths, and the (stochastic) network calculus are available to set up systems.

3. EXPERIMENTAL RESEARCH

Much Future Internet research is experimentally driven and centered around a testbed that may become the first running prototype of the Future Internet. These testbeds are frequently implemented as overlay, virtual, and/or Software Defined Networks that run concurrently to a production network using the same hardware. Clearly, testbeds are indispensable to implement running code as a proof-of-concept, whereas their use for understanding networking and for establishing new principles and paradigms is less obvious. Our goal at the seminar was to revisit experimental approaches to gather lessons learned and best practices. To this end, we asked speakers to provide 'seed' talks that elaborate on questions such as:

- Is there something like 'experimentally driven knowledge gain' and what is the scientific method?
- Which insights can the experimental, testbed-based approach reveal?
- What are meaningful use cases of testbeds, e.g., engineering details, concepts weakly understood, prototyping?
- How should a testbed platform look like, which properties must be provided to achieve comparability and validity?
- What makes (research) overlays special (pros & cons)?

Seed talks were given by Martina Zitterbart and Oliver Waldhorst "overlay networks as innovation engines;" Jörg Liebeherr "experimentally driven networking research;" Brad Karp "wide-area distributed system deployments yield fundamental insights;" Markus Hofmann "<provoke> research testbeds considered harmful </provoke>;" and Phuoc Tran-Gia "some reflections on experimentally-driven research and testbeds in future network research." The session was concluded with a very lively podium discussion. Presentations on specific testbeds and aspects were given by Paul Müller "a virtual environment for distributed systems research;" Jonathan M. Smith "NEBULA future Internet;" and Michael Menth "Congestion Exposure (ConEx) – an experimental protocol for the Future Internet." Selected main aspects that were discussed are:

Insights from testbeds Regarding testbed-based research, Brad Karp in his talk raised and reflected on two fundamental questions: 1. "is the act of building a testbed research?" and 2. "do testbeds yield fundamental research insights?" While numerous examples emerged (also in other talks and in discussion) where testbeds revealed unexpected phenomena that fostered original research, the answer to the first question was that in general the act of building a testbed may not be research. (There were other opinions, such as that building a research testbed is by definition research, or that if certain research can only be done with a testbed, the question does not matter.) This negative answer implies, however, that a research testbed cannot be an end in itself and consequently great care has to be taken in deciding whether to build a testbed or not, given the need for significant funds and labor. The hard question that remains is how to justify the decision for or against building a testbed or a prototype for experimentation. A litmus test proposed by Brad Karp is: "what relevant phenomena can't you simulate?" Basic examples include user behavior, feedback, non-perfect behavior of systems, or a lack of knowing what to simulate at all.

The engineering loop In his talk, Jörg Liebeherr highlighted specifics of the experimental discovery processes and important consequences thereof. According to the scientific method, experiments are constructed to verify, refute, or refine a hypothesis to uncover the laws of nature. In contrast, the engineering process implements a system design for experimental evaluation to support, discard, or improve an idea [4]. Consequently, demands on experiments differ largely where reproducibility and controlled experimentation become less important in engineering as the target is a proof of concept using some 'real-world' experiment (and not finding the truth regarding a hypothesis). A concern that is raised by the lack of replication is objectivity as typically design and experimentation are done by the same, interested party (in contrast, e.g., in physics a new finding is accepted only after independent replication). Also, the need for applying sound statistical methods was emphasized.

Scalability of testbeds An aspect that was discussed controversially in different presentations was the scalability, the size, and the inclusion of real users in testbeds. While

Jörg Liebeherr conjectured that there is an optimal size of community testbeds (testbeds where users contribute the resources such as PlanetLab) beyond which resource sharing makes the network 'crawl', it was argued by Brad Karp that the resulting effects, e.g., latencies, make testbeds most useful. Latencies are results of real-world behaviors, coming either from people (e.g., access to webpages, etc.) or from machines (e.g., access to DNS servers, web servers, supernodes, etc.). These behaviors are very difficult to emulate in testbeds unless one has access to real-world user behavior and/or data traces. The importance of the aspect was further supported by Markus Hofmann who emphasized the necessity for testing systems with millions of real users to be able to observe relevant aspects, such as the example of signaling storm mentioned by Phuoc Tran-Gia. On the other hand, Klara Nahrstedt added that one can observe not only changes at the level of millions of real users, but one can see users changing behavior at much smaller scale. Social scientists showed that user behavior changes at group sizes of 2-3, 15, 50, 100, or over 150. This is purely from the social point of view how people behave to each other and may or may not be visible in the digital domain, i.e., if people of these group sizes use interactive technologies (e.g., Skype, or teleconference). One of the reasons why we do not know how technology changes the behavior is that many current interactive technologies do not support more than 10 users. Finally, Phuoc Tran-Gia emphasized the need for testbeds, simulation, and analysis to exist side by side to be able to trade detail for scalability.

General purpose testbeds Besides scalability, the aim for general purpose testbeds was reconsidered. On the one hand, there was agreement that overlays enable and foster innovation within a broad scope. Also overlay-based testbed networks such as PlanetLab were highly valued. On the other hand, it was cautioned that general purpose testbeds may fall short of one's expectations as there will always be contradicting requirements, e.g., controlled experiments vs. need for inclusion of real users.

Contact with reality As already illustrated by the engineering loop, a frequent goal of testbeds is to reproduce real-world behavior. Meeting this goal raises, however, numerous challenges: supporting and maintaining heterogeneous devices in testbeds is cumbersome; programming interfaces may evolve quickly, e.g., on smart phones; including real users is difficult; realistic traffic, mobility patterns, or user behavior are hard to obtain; privacy concerns persist even after anonymization due to risk of de-anonymization; anonymized data may be hard to work with; and re-engineering (human-designed and implemented) systems or mechanisms, e.g. YouTube or Skype, to obtain desired data can be questioned and faces the risk that an update may just change the mechanism under investigation. In conclusion, it was felt that academia depends on the willingness of industry to share traces and data of real-world phenomena.

4. SDN, VIRTUALIZATION, OPENFLOW

The third day of the seminar dealt with the practical technologies of SDN, virtualization, and OpenFlow. The presenters were Wolfgang Kellerer "opportunities and challenges for Software Defined Network systems;" Klara Nahrstedt "Software Defined Networks for distributed interactive multimedia environments;" Laurent Mathy "SDN++: beyond programmable plumbing;" Panagiotis Papadimitriou "towards wide area network virtualization;" and Paul Kühn "automatic energy efficient management of data center servers operated in hot and cold standby with DVDS."

One of the most popular practical topics that emerged during the seminar was use cases of Software Defined Networks (SDN), in its most general sense, not just OpenFlow. The vision for SDN has been well expressed by Nick Mckeown and Scott Shenker [8]. Advances in Computer Science mean that we can generate safer, more secure programs much more quickly and flexibly than in the past. Programming languages, operating systems, especially virtualisation and isolation techniques, and verification tools have all advanced to the point where we can be far more confident about such an approach, even when our critical infrastructure for communications depends on systems built this way. Indeed, we can potentially make much more rapid progress at improving the dependability of the Internet this way than continued incremental patching of the existing systems.

In the past, the internal components of networks – switches, routers, firewalls, traffic management, and so on – have generally evolved in a separate software and hardware ecosystem from end systems. Hence, while commodity price, performance and production of smart phones, desktops, servers and whole data center computing has largely benefited from exactly the same advances, and scale of use of computer science, the infrastructure of the Internet has moved more and more into being a set of niche businesses generating specialised technologies.

Proponents of Software Defined Networking hope to reverse this trend, by moving large fractions of the control plane of the Internet, and even some parts of the data plane into the arena of main stream computer science. Of course, if only for performance reasons, we expect *Big Iron* switches to remain the remit of specialised hardware, since by definition they are likely to continue to exist right on the performance limits of anything we can build. However, many other components (routing, firewalls, new forwarding paradigms) and almost all functions in data and control planes in non-core routers can move back into the commodity world. Thus the vision for SDN is to have a *platform*, in the same way that the iPhone, Android, and Windows have platforms (and stores), on which a million applications ("SDN Apps") can flourish[2].

On the last day of the seminar, we had a session where we tried to enumerate some of the use-cases for SDN, and this is a partial list of what was captured:

1. Multi tenant data centers have a need for specialised networking for different tenants who may have different network service requirements – just picking a few examples, one tenant might need multicast for serving IPTV, while another might need delay bounds for traders or gamers, whilst still another might need in-network processing to support better MapReduce task deterministic throughput.

[2]It is worth noting that some of these ideas have been seen several times before in research activities such as the Active Networks programme, and work on OpenArch including the separation of data and control planes via General Switch Management Protocol (during the emergence of IP over ATM, then subsequently MPLS).

2. Multi user VR and Massive Multiplayer Online Role Playing Games have large scale dynamic network requirements whose parameters keep changing – one could certainly imagine supporting this partly through SDN.
3. Live migration of virtual machines is increasingly used both for load balancing and for high-availability/persistence during scheduled maintenance. VM migration puts a spike load onto the net, and could interfere with routine steady state traffic unless serviced specially, and requires itself, specialised support for continued access (re-routing existing flows, and redirecting new clients, for example). This could be supported as an SDN function.
4. The Internet of Things (IoT) is touted as a big driver for networking (e.g. IPv6 deployment to support the massive increase in globally reachable devices). IoT will also require specialized network functionality (securing access, delay bounding access for some sensor/actuator cyber-physical applications, and many as yet unforeseen applications that will no doubt emerge).
5. Content Distribution Networks (CDN) need management. Some CDNs place difficult load on an ISP – especially CDNs that are all or partly P2P, perhaps interfering with peering arrangements. Managing this could easily be seen as a good match to SDN.
6. Middlebox management is sorely needed. Middleboxes increasingly are the main cause of ossification of the network - the inability to deploy new transport layer protocols or extensions is mainly traceable to the ad-hoc mature of the large number of different middlebox functions deployed throughout the Internet. Bringing these into a coherent framework would allow some semblance of progress to be made. SDN again would be the tool to replace the ad-hoc functionality with a clean slate programmable system with a public, open standard API.
7. This is the metaclass of the SDN use case - management of multiple SDNs will itself be a challenge.
8. Hybrid clouds where the cloud supports a set of user applications and the set of SDN apps is an interesting case of meta-management. Co-existence of the applications and the customised, specialised support for these applications in the Software Defined Network is a key requirement, for example, in today's data centers.
9. Resilience in networks currently relies on ad-hoc approaches to providing replication. SDN can help unify a set of mechanisms under one control plane.
10. Cross-layer design of distributed applications requires potentially more open, possibly reflective APIs in SDN, so that the application and the SDN can co-evolve efficiently.
11. Enterprise infrastructure setup is a key need in large scale private intranets. There are many such systems in the world, and often their owners incur high costs to provide customised network services. SDN offers a way to build more flexible networks that could be matched to an enterprise's needs more as a matter of configuration than bespoke engineering.
12. Improved security may be on the cards if SDN takes on board improved software practices, using safer programming languages, and trusted computing bases, and techniques for information flow analysis, software verification, and so forth.
13. One simple hope for SDN might be to take research results in policy routing (e.g. meta-routing) and do a one-time replacement of BGP.

Many of these cases are driven by some "pressure", viz

Overcommit the economics of multitenancy depend on invisible overcommit – that is, Amazon should be able to pack more and more VMs in without the customers realising that something else is being scheduled in the gaps of their usage.

Latency VR especially (see oculusvr.com) requires far lower latency than going cross-continental permits.

Flow-based I/O instead of soft state VMs can be viewed as a network flow (in the aggregate sense; its not just packets but a collection of TCP/SSL state that needs to be moved around sensibly). CDNs are basically the similar to VM migration – they need to be represented as a service and give the network a chance to load balance globally; Again middlebox management is all about explicit flow management, but *not* just TCP/IP – application level flows are just as important (cookie-based balancing for HTTP as seen in every L7 router, SSL session caching in those terminators, cross-protocol WAN compression, etc);

Global identity IoT needs a way of *naming* all the billions of devices, and not addressing them (i.e. IPv6 is a red herring, but the lack of signposting is the important thing). Also, they need pseudonyms and intentional names, which people are mistakenly cramming into the addressing layer instead of the naming layer.

We also considered SDN in data plane.

1. Data plane middleboxes already interfere with TCP/IP packets in an unstructured way (frequently, to improve operation of protocols in wireless networks such as 3G and 4G nets, but also interfering with the ability to deploy new versions of TCP. SDN could include data plane packet processing, at least near the edges of the network where performance requirements can be met affordably.
2. Fine grain media control could be another SDN data plane activity - e.g. video and audio re-coding for different receivers with different rendering capabilities.
3. Network as a Service (NaaS) for data center (e.g. map-reduce) in-net application code has also been suggested as an SDN data plane task. The TCP incast problem can be solved by processing a fixed number of shuffle phase data packets in switches, with relatively simple tasks.

In addition to numerous use-cases or applications on top of SDN, SDN was also found to have the potential to connect technologies below the SDN implementation. But there are broader questions that we leave unanswered here:

- Can we use SDN to Connect IP and non-IP networks?
- Could we do layer 2 and layer 4 SDN via OpenFlow?
- What could make SDN harmful?
- What are the key SDN business cases?

All of these use cases raise the question whether SDN may be the "de-constraining constraint" of the Future Internet. Thus moving all of the control plane, and parts of the data plane into the same environment as end systems, and creating an eco-system for SDN apps (served by new network appstores) may have this effect on the Future Internet that the approach had in the mobile smart device world.

5. CONCLUSION

This editorial summarizes main outcomes of the recent Dagstuhl seminar on the Future Internet. The seminar picked up a number of fundamental questions regarding the design of networks, where our discipline would benefit from a prescriptive theory, as well as regarding the research methodology with a focus on experimental, testbed-based research. Thirdly, with SDN, virtualization, and OpenFlow, we investigated promising technological aspects of the Future Internet. The main conclusions of the seminar are as follows.

- There was a general agreement that we lack a prescriptive network theory and that the existence of such a theory would benefit our field of research significantly. Expectations are that such a prescriptive theory is a long-term research goal, where major challenges are due to the complexity of networks, the degree of formality that can be achieved (high-level guidelines vs. an axiomatic approach), and the consideration of temporal aspects. Finally, due to the interdisciplinary nature of networking, there will not be a Swiss army knife in network theory, but rather a set of different theories to address individual aspects.
- Testbed-based research was found to be of vital importance to networking as real experiments uncover effects that may be very hard to anticipate otherwise, e.g., due to user behavior, randomness, timing, or the immense complexity of systems. A number of success stories showed how the understanding gathered with testbeds revealed open problems and fostered the invention of new solutions. On the other hand, there was some cautioning to proceed carefully: there are cases where testbed experimentation does not reveal anything new compared to simulations, hence expectations should be formulated beforehand; building a general purpose testbed results in implementing the (not useful) lowest common denominator; and the necessity of using sound (statistical) methods was emphasized. The need for traces/data was identified as a critical aspect as a testbed by itself is not sufficient to perform realistic, e.g., 'million people' experiments. Here, academia urgently needs support from industry to share in some way traces or access to 'training' testbeds that provide these types of data sets.
- On a shorter time-scale, SDN, virtualization, and OpenFlow seem to take place. A number of interesting use cases were formulated at the seminar, that enable innovative applications or bridge existing and future technologies, e.g., next generation IP, non-IP, or Internet of Things.

  It is interesting to speculate how this will evolve, and what the Internet will look like in 2023, and how different it is in 2013 from the way it was in 2003.

6. ACKNOWLEDGEMENTS

We would like to thank all presenters, scribes, and participants for their contributions and lively discussions. Particular thanks go to the team of Schloss Dagstuhl for their excellent organization and support. We also would like to thank Anil Madhavapeddy for his comments on SDN.

7. PARTICIPANTS

Zdravko Bozakov (Leibniz Universität Hannover, DE)
Florin Ciucu (TU Berlin, DE)
Jon Crowcroft (University of Cambridge, UK)
Ruben Cuevas-Rumin (Univ. Carlos III de Madrid, ES)
Hermann de Meer (Universität Passau, DE)
David Dietrich (Leibniz Universität Hannover, DE)
Markus Fidler (Leibniz Universität Hannover, DE)
Philip Brighten Godfrey (University of Illinois - Urbana, US)
Christian Gross (TU Darmstadt, DE)
David Hausheer (TU Darmstadt, DE)
Markus Hofmann (Alcatel-Lucent Bell Labs - Holmdel, US)
Matthias Hollick (TU Darmstadt, DE)
Tobias Hoßfeld (Universität Würzburg, DE)
Brad Karp (University College London, UK)
Martin Karsten (University of Waterloo, CA)
Wolfgang Kellerer (TU München, DE)
Karl Klug (Siemens - München, DE)
Paul J. Kühn (Universität Stuttgart, DE)
Jörg Liebeherr (University of Toronto, CA)
Laurent Mathy (University of Liège, BE)
Martin Mauve (Heinrich-Heine-Universität Düsseldorf, DE)
Michael Menth (Universität Tübingen, DE)
Max Mühlhäuser (TU Darmstadt, DE)
Paul Müller (TU Kaiserslautern, DE)
Klara Nahrstedt (University of Illinois - Urbana, US)
Panagiotis Papadimitriou (Leibniz Universität Hannover, DE)
Rastin Pries (VDI/VDE-IT, München, DE)
Ivica Rimac (Bell Labs Alcatel-Lucent, DE)
Silvia Santini (TU Darmstadt, DE)
Nadi Sarrar (TU Berlin, DE)
Jonathan M. Smith (University of Pennsylvania, US)
Ralf Steinmetz (TU Darmstadt, DE)
Dominik Stingl (TU Darmstadt, DE)
Phuoc Tran-Gia (Universität Würzburg, DE)
Oliver P. Waldhorst (Karlsruhe Institute of Technology, DE)
Klaus Wehrle (RWTH Aachen, DE)
Michael Zink (University of Massachusets - Amherst, US)
Martina Zitterbart (Karlsruhe Institute of Technology, DE)

8. REFERENCES

[1] D. L. Alderson and J. C. Doyle. Contrasting views of complexity and their implications for network-centric infrastructures. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 40(4):839–852, July 2010.

[2] J. Crowcroft, M. Fidler, K. Nahrstedt, and R. Steinmetz. Dagstuhl Seminar on the Future Internet. `http://www.dagstuhl.de/13131/`, Mar. 2013.

[3] J. Day. *Patterns in Network Architecture: A Return to Fundamentals*. Prentice Hall, 2008.

[4] D. G. Feitelson. Experimental computer science: The need for a cultural change. Online, 2006.

[5] M. Karsten, S. Keshav, S. Prasad, and M. Beg. An axiomatic basis for communication. pages 217–228, Aug. 2007.

[6] C. Partridge. Forty data communications reserch questions. *ACM Computer Communication Review*, 41(5):24–35, Oct. 2011.

[7] J. Rexford. The networking philosopher's problem. *ACM Computer Communication Review*, 41(3):5–9, July 2011.

[8] S. Shenker. An attempt to motivate and clarify Software-Defined Networking (SDN). Online video clip. YouTube, 2011.