Dynamic network (re-)configuration across time, scope, and structure

Roland Bless
Institute of Telematics,
Karlsruhe Institute of Technology

Bastian Bloessl, Matthias Hollick
Secure Mobile Networking Lab,
TU Darmstadt

Marius Corici
Fraunhofer FOKUS,
TU Berlin

Holger Karl
Hasso-Plattner-Institute,
University of Potsdam

Dennis Krummacker*, Daniel Lindenschmitt†, Hans D. Schotten*†
*German Research Center for Artificial Intelligence (DFKI GmbH)
†University of Kaiserslautern

Lara Wimmer
IHP—Leibniz Institute for
High Performance Microelectronics

***Abstract*—Today's wired and wireless networks offer data transportation and service delivery over a wide range of locations, devices types, and usage scenarios. While they do adapt to changing situations, they are, at their core, essentially still static. In this paper, we attempt to collect challenges and opportunities to substantially broaden the adaptation options for networks with respect to both data transport and service delivery. We discuss options ranging from device technology to network architectures.**

I. Staticness of today's networks

Wired and wireless networks today serve two key purposes: data transport and service delivery. They comprise devices ranging from end user devices (e.g., Internet-of-Things devices or smartphones) over base-stations, switches and routers to servers (e.g., in a remote data center, at the network's edge, or as a fog of servers on user devices). At all these levels, devices and entire networks adapt to new situations. Nonetheless, we argue that today's networks are essentially still static, at least compared to adaptation options in reach of today's technology.

A. Examples for adaptation today

Let us illustrate some common adaptation options. In a simple case, the network is static and only the applications' traffic adapts, e.g., by using a congestion control protocol. More realistically, the network also adapts to changing traffic patterns, e.g., by rerouting flows via traffic engineering or by provisioning additional capacity. But this happens on much longer timescales: While congestion control happens within round trip times (milliseconds), capacity provisioning takes minutes or hours—e.g., when providing additional wavelength division multiplexing (WDM) lightpaths in a fixed network or when additional frequency bands are made available in a mobile network's cell—or weeks and months, when actual physical infrastructure needs to be deployed; digging new fibre optics cables can take an unforeseeable amount of time.

But in any of these adaptations, the devices of the network are fixed in their behavior. They can certainly be upgraded with new software versions, they can also be widely configured and programmable hardware (e.g., via P4[1] or eBPF [2]) opens the door to more flexibility, but in essence, devices stay fixed.

Similarly, the topology of networks stays essentially fixed, reacting to changes in demand only reluctantly; techniques like dynamic backhauling via additional frequency bands are well known (mmWaves, [3], [4]) but effectively do not touch a given network structure, just add additional links. Ideas like mutually adapting WDM configuration and applications [5] have been explored but are not widely deployed.

And again similarly, the composition of network functions is essentially fixed; Network Function Virtualization (NFV) has opened the door to scaling and placing network functions at suitable locations in a suitable number, and even to integrate new services into a network perceived as a service deployment platform. But management and orchestration are still fixed to a pre-chosen functional decomposition both inside a Radio Access Network (RAN) and between RAN and core, typically dictated by a standardized architecture. While there are ideas about configurable orchestration with adaptable structure (e.g., SONATA [6]) or transition-enabled networks [7] we are still looking at a fairly rigid network architecture.

B. Dimensions for adaptation

Based on the previous examples, we observe that there are (at least) three dimensions along which to discuss adaptation.

Time: Adaptation decisions can happen on vastly different *timescales*. This can be advantageous as different timescales can decouple nested control loops or learning processes, making the slow process effectively static from the perspective of the quick process. This typically helps system stability. While always adapting on very short timescales is tempting, the conceptual, algorithmic, and operational challenges are formidable. Nonetheless, we believe that the time has come to think about adapting on shorter timescales.

Scope: Over which geographic, technological, administrative, etc. scope does adaptation extend? In a geographic sense, do we adapt only local operation (e.g., adapting wireless resources inside a single cell), neighboring entities or even regional operation (e.g., adapting wireless resources over all cells of a suburb)? In a technological sense, do we adapt only within the scope of a single technology, or across technologies (e.g., different radio transmissions); similar for administrative

boundaries. In a contextual sense, do we only consider a single user, a single type of traffic, etc. or do we consider a broader scope?

Structure: Is structural adaptation limited to, e.g., a single layer of a network? Or does it encompass multiple layers? Does it consider the structure of a network itself, e.g., how to change from a coreless, purely device-to-device network to a conventional core-based one by integrating such networks together. Does it, in particular, consider a network as *both* a data exchange and a service provisioning facility, and consider both aspects jointly? Is adaptation only concerned with internal network operation or also with user-level applications running inside this service provisioning facility?

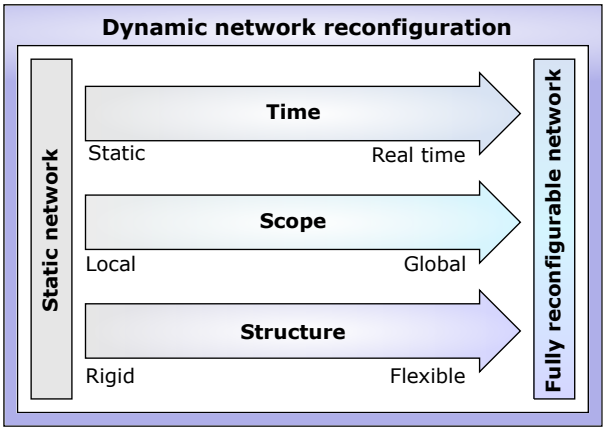

Figure 1. Three dimensions of adaptation: time, scope, and structure

Figure 1 highlights the adaptation dimensions *time, scope, and structure*. We acknowledge that many of these aspects have been considered in past research—organic computing, bio-inspired systems, on-the-fly computing, autonomous networks, self-managed or zero-touch networks, active, open or fluid networks all spring to mind. We nonetheless believe that it is necessary to *jointly consider these three dimensions time, scope, and structure* to develop future networks.

Section II describes foreseeable changes in how networks are conceived. Section III highlights selected exemplary challenges. Section IV speculates about possible solution approaches.

II. Foreseeable changes

We expect changes in three categories: criteria for acceptance of solutions (Section II-A), new types of traffic and applications (Section II-B), and technological developments (Section II-C).

A. Acceptance criteria and scenarios

Acceptance criteria change over time. Recent trends include: 1) *improved environmental awareness*, e.g., by improving energy efficiency or better integration of renewable energy, 2) *load-proportional cost* (monetary, energy, time, etc.) instead of worst-case provisioning, for both CAPEX and OPEX, 3) *technological sovereignty* and the necessity to deal with vastly different legal frameworks worldwide, 4) *reusability* to enable the usage of the same hardware while seamlessly upgrading the network, not requiring new investments with every new generation of network functions.

As a forward-looking scenario, *provider-independent operation* gets more and more traction. This is driven by scenarios like 1) operation of *disconnected groups of devices*, 2) *campus networks* for different use cases (factory floors, construction sites, etc.) possibly in remote or quickly changing setups, 3) *disaster recovery*, where a network needs to survive partitioning. Such examples require a new network architecture where *independent subgroups* can work as a *coreless network* [8], potentially joining together, separating again, and merging into an existing infrastructure network when suitable. Conventional cellular/mobile architectures with rigid division into RAN, core, etc. are certainly not up to this task. Rather, this needs to be amended by concepts like *self-backhauling* where the participating devices create the backhaul themselves, without recurring to existing infrastructure. This can happen inside a given technology (say, 5G standalone network) or across multiple technologies (e.g., using some long-haul radio technology available on some devices).

It could also imply that networks have to be constructed on-the-fly from whatever resources are readily available, forming what we call *anyway infrastructure*: a neutral infrastructure that can be created without being primarily designed as a dedicated network. How a suitable network architecture could look like is up for research. Notably, this should not only apply to data exchange but also to storage and computation—e.g., easily deploying a micro-scale data center at a construction site is today practically infeasible and would need to happen without preplanning and up-front dimensioning.

Moreover, conceiving this as a mere network configuration problem is too simple. For instance, once networks merge or split, we also need to *reconsider a functional split on-the-fly*: functions that had to be run on user devices while not in contact with an infrastructure network should perhaps be pushed onto the infrastructure. Conversely, once a split-off of a group becomes likely, both code and state need to be provided to such a group to ensure that it can continue operating as an isolated network, without recourse to infrastructure resources.

In a broader sense, security, privacy, and trust requirements constantly change, with existing trust models likely no longer tenable and the impact of over-the-top players as well as (possibly adversarial) governmental agencies on these questions becoming bigger and bigger.

B. Traffic and application needs

While in the past, our community has not always been fully successful in predicting traffic and application needs (to whit, consider WAP in 3G or IMS in 4G), we still dare to speculate that the following trends will happen.

Traffic diversity: Traffic will become more diverse, going beyond 5G's classes like eMBB, URLLC, and MMTC in an *unforeseeable manner* (much like Youtube was unforeseen when IMS was designed). Hence, we need to ensure that our

networks can easily adapt to new types of traffic. A prime example is the integration of sensing capabilities into a mobile network. It is a wide open question how this integration in combination with novel application capabilities will influence traffic needs.
Another example is rapid adaptation to new requirements in non-public networks, e.g., on a factory floor. In scenarios such as industry/automation, entirely static networks are still the norm today to allow for service guarantees [9] Moreover, production processes must not be interrupted. That requires that adaption is not only fast, but also that the transient periods are carefully handled to ensure the system is always in a safe state. Representative use cases are studied by the 3GPP [10].

Service delivery: The trend to using a network as a service delivery platform, truly realizing ideas of edge and fog computing for general-purpose applications, still has a chance to happen, finally bridging the gap between pure Network Services and Application Services. A key driver could be edge learning with its newly emerging traffic patterns. At the same time, this also gives new opportunities for jointly shaping traffic, application deployment, and infrastructure. We elaborate on that idea in more detail in Section III-E.

In short, the best way to prepare for the unseen and unknown is to design a wide range of adaptation options.

C. Technology push

In addition to acceptance and scenario pull, we foresee a considerable technology push on multiple fronts. We can only list a few examples here.

Intra-device flexibility: Radio access hardware is becoming much more flexible. One enabler is the use of quickly reconfigurable Field-Programmable Gate Arrays (FPGAs), enabling the migration of functions between general-purpose processors, digital signal processors and custom FPGA implementations. E.g., forward error correction schemes can be dynamically adjusted to take up more or less space on a basestation's FPGA, depending on how much traffic in a cell needs which degree of reliable transmission. Another example is to move digital signal processing functions to an FPGA (for low latency and low jitter) or to a GPU (to leverage some AI/ML processing of the signals). The goal is to make best use of the capabilities of a device and extend its capabilities at runtime if needed ("functional morphing" [11]).
Another example would be to provision protocols and algorithms for unusual modes (like emergency communication) only when needed, without taking up space in normal operation. On user devices, this could be potentially achieved by providing different firmware for ASICs to support different usage situations. While such ideas of "protocol downloads" exist for a long time, the key difference here lies in the tight integration with accelerators of various ilks.

Multiple technologies: Supporting multiple radio technologies over a wider range is likely to become easier and more pervasive also in infrastructure devices. This allows to migrate traffic between different frequency bands more easily, trading off, e.g., capacity vs. coverage.

New device types: *Satellite networks* can provide more backhaul options at a location than is conceivable today, specially using low-orbit, high-capacity optical links (e.g., EDRS, HydRON). Satellites with storage for content delivery are the logical next step. Closer to earth, *drones* serve similar purposes on different timescales, scopes, and structures [12]. E.g., providing some server capacity inside a drone is technically feasible but raises questions of power consumption, drone mission time, handoff of application state from one drone to its replacement drone, etc.

III. Challenges

Addressing all these goals and changes presents formidable research challenges. We start this section with brief examples and then discuss selected aspects in more detail.

A. Brief examples

Fluid optimization goals with complex transients: Once networks adapt to different scenarios, their optimization goals can change radically, e.g., from a connectivity-first goal in disconnected operation to a capacity-first goal once reconnecting to an infrastructure network. This is difficult to handle in conventional optimization techniques but machine learning-based approaches might be more amenable to such situations. Moreover, transitions between such operational regimes need to be handled: Even while a network changes from one operational point to another, some basic operation should always be ensured. This, in turn, might be more challenging to solve based on machine learning as such transients are unlikely events and hence cannot easily be explored. It might become necessary to use hybrid (conventional + machine learning) optimization approaches, where it is not clear how to switch between them.

In-band network management, adaptation-aware monitoring: Management, monitoring, and control cannot happen via dedicated, separate networks (as is commonly done inside data centers). Instead, we need in-band approaches that use the network itself, spanning across vastly different timescales, scopes, and structures. In addition, monitoring systems need to be aware of adaptation actions (possibly even ones from other subsystems) lest they generate false alarms, mistaking adaptation for failure.

Unknown consequences: In a simple system, the consequences of any control action are deterministic, fully known and easy to foresee. This will certainly not be the case here. Instead, any control or adaptation action might have unforeseen consequences, but valiantly exploring any such option might have unacceptable consequences for operations.

Functionally scalable architectures: The idea of *coreless networks* has been described above. One ensuing challenge is how to decide which function(s) need to be activated in which adaptation situation. E.g., there are multiple adaptation steps happening from a simple device-to-device network over a small group to an integration into an existing cellular network. It is not clear at which stage which functions that are today provided by a RAN or core network are needed and how to

ensure that they are consistently provided (e.g., how to deal with conflicting state when merging two networks).

Software development: Software running inside a network can operate the network itself ("network-facing services") or providing actual applications to end users ("user-facing services"). In either case, such software will be confronted with much more profound changes than conventional software today. As a consequence, it will no longer be possible to ensure correct operation by extensive up-front testing, but novel approaches are needed.
In today's distributed applications, partitions or unreachability are usually considered a failure scenario. But the mindset needs to change to think of such cases as the rule, not the exception. This requires effort in training and educating programmers to deal with such cases and to translate experience from cloud-based, large-scale distributed applications to the telecommunication environment. It also might become necessary to more easily compose protocol stacks than today, to inject functionality as needed while still dealing with the ensuing security risks.

Business models: We expect a rethinking of business models. Business ideas around slicing have pointed in the right direction, but they were still operator-focused with operators being owners of all relevant resources. We expect that model to dissolve further, with more competition to happen and opening up of markets. E.g., spectrum does not necessarily have to be sold to few entities once sufficiently good adaptation techniques are in place, effectively overcoming technical limitations that necessitated near monopolies in the past.

New notions of trust and dependability: A common requirement is to have "fully trustworthy" and "fully dependable" networks. While it is alluring to make such claims, we need to recognize that it is impossible to reach this goal with existing technology/approaches. Rather, need to face the fact that there is a (high) price to pay for any of these requirements, that they are not needed for all scenarios and, hence, that this price should not be shared by all users.

Resource management: Many aspects turn out to be resource management problems: how to allocate physical resources to virtual needs, how to do so across time, scope, and structure? The following examples will go into more details.

B. Reachability and connectivity

As the network topology and thus its infrastructure become more dynamic, control of network elements (base stations, switches, routers, and end-systems) must follow the underlying changing infrastructure. This requires reachability and connectivity among the resources.

We conjecture that an integrated address assignment and routing scheme is a promising approach here. Unique addresses help during network partition and merge. Integrated with that, a distributed routing protocol disseminates reachability information inside the network, along with a matching forwarding protocol that ensures that all resources are indeed reached by whatever messages. This includes, but is not limited to, connectivity between any ordinary resource and any "controller" resource, where the controller in charge of a resource might change rapidly. Challenges for this routing/forwarding protocol are the sheer number of resources, highly dynamic or moving resources as well as its efficient operation across different types of networks that could be sparse or dense (like data center fabrics) and any mixture thereof.

C. Adaptive controller topologies

In a network that exchanges data and executes services, a lot of aspects need to be controlled and managed. For simplicity, we summarize under "controller" elements like an SDN controller, an MPLS path computation element (PCE), or an NFV management and network orchestration (MANO) system. A known issue is that in typical real-world deployments, the size of the controlled system is too big to be handled by a single instance; also, single-point-of-failure considerations make single-instance approaches unacceptable. Therefore, distributed implementations of such controllers are commonplace.

To further reduce load or improve response times, a typical mechanism is to introduce hierarchies or peer-to-peer structures of such controllers, where separation of concern happens via topological, administrative, or geographic scope. This induces a *controller topology*. A fixed topology, e.g., based on the network topology, is again commonplace and simple to envision. In prior work, concepts like a *load-adaptive* controller hierarchy has already been researched (e.g., in 5G-PICTURE). While that is useful, it is insufficient: for a concept as envisioned here, we also need a *structure-adaptive* controller hierarchy. Depending on how the network adapts and changes its topology, we also need to adapt and change its controller topology. E.g., assuming responsibility for the control function of a single device once it joins into a larger network, or provisioning a new controller once a subset of devices is about to or already has separated from a larger network. Key challenges are likely the need to deal with partitioning and rejoining of this controller topology at unforeseen points in time, raising typical distributed system issues. We emphasize that this should range from the simplest borderline case—a single device without connectivity—to the global Internet.

D. Dealing with accelerators

In most of today's control and orchestration systems, resources are usually considered to be of different types, but essentially with more or less the same fluid, infinitely reusable usage model: a link's capacity can be just as arbitrarily subdivided as a CPU's instruction cycles. Moreover, interactions between different resource users are more or less ignored (with some upcoming research challenging whether this is too big a simplification). This model is no longer true for storage (a full disk stays full), but it definitely breaks down for accelerators, e.g., FPGAs or GPU/TPUs.

Characteristically, the deployment model for accelerators is quite different. It can take considerable time to reprogram an

FPGA, and interaction between different modules running on it can be negligible or prevent deployment entirely. Multiple code versions for each module will be necessary, with different tradeoffs between resource consumption, latency, throughput, energy consumption, etc. when running the same function on different hardware.

A typical example can be real-time requirements: computational expensive real-time signal processing might be well suited on an FPGA; ML interference from trained models with soft real-time requirements could fit on a general-purpose CPU or perhaps even inside a packet processing pipeline; ML training is a clear fit for GPUs. But typically, we are unable to provide all types of resources in ample amount and, hence, have to accept compromises when deploying a function on less ideal hardware. Depending on available resources like RAM, it might even be necessary to switch to different algorithms, say, use smaller ML networks with different training methods when ample computational power but little memory is available.

E. Jointly optimizing services and network configuration

We conjecture that in the future, data exchange and service/application delivery becomes much more tightly interwoven. Here, we highlight one example aspect: The joint optimization of 1) service deployment, in the sense of serving a user population with a microservice composed of individual services, scaling the number of these constituting services and deciding where to run them, 2) traffic engineering, in the sense of deciding how to route that traffic from user to first service and between services, and 3) decide where to provide networking and computing infrastructure; consider examples like adapting WDM lightpaths or sending drones with radio backhauling and computational capacity on board to wherever they are needed.

The key idea here is not to think of these as two or three separate optimization problems. Conventionally, that would be easy: optimize infrastructure on a long timescale and think of it as being fixed for the optimization of scaling/placement/traffic routing. Clearly, this is a viable approach and feasible to integrate into existing MANO frameworks. Instead, we suggest to think of it as single, joint problem on the same timescale and across multiple scopes. Challenges here abound. E.g., how to deal with state management when the drone hosting said state needs to be swapped out against another once it runs out of battery? What are good tradeoffs between having the service-hosting drone following a single user vs. serving a group of users that moves in different directions? How to tradeoff latency against energy consumption? Is there any commercially viable business model for such scenarios?

IV. Solution approaches

It is clearly not the goal of this whitepaper to provide solutions to all the challenges outlined above. Nonetheless, we attempt to outline some important considerations here.

A. Problem structure

To properly address the aforementioned challenges, a system needs to have a good account of its constituting parts, both in terms of devices and services. For devices, a very expressive self-assessment of a device's capabilities is needed, e.g., what parameters can be controlled on a radio level (transmission power, frequencies, memory, processing power, etc.), what control exists over movement (e.g., 3D mobility, possible flight duration, etc.). Similarly, explicit annotations about a service request will be needed (e.g., intended user population, constituting services, multi-tenant capability yes/no, state to be managed, real-time requirements, etc.). None of this is really surprising, but it becomes more important in such a highly adaptive environment.

In addition, we need explicit representation of administrative control and administrative boundaries; e.g., which device may be controlled by which entity, possibly in which contexts. Similarly, explicit representation of borderline conditions are required (e.g., no-fly zones for drones). While this is less standard, it is also not conceptually too difficult and similar ideas have been around (e.g., radio maps in cognitive radio).

More challenging is the need to argue about possible network configurations and, in particular, allowed and disallowed subsets. Conceptually, that might be doable. In practice, this constitutes a formidable challenge to identify such combinations even from a global, out-of-system perspective. In a realistic, in-system perspective, this becomes exceedingly difficult, given conflicting technical and commercial interests.

B. Solution techniques

Even assuming all input parameters being available, finding concrete solutions is still challenging as well. We briefly outline just two aspects.

Division of concern: Putting all these aspects into a single optimization problem is clearly infeasible. It would be huge, unclear where to collect input, unclear how to assign responsibility and control, unclear how to keep input data up to date and distribute commands in time, etc.
Instead, the problem needs to be divided in simpler subproblems, along one or several dimensions of time, scope, or structure. Obvious subdivisions could be between device control, resource control, network control, and service control (with due consideration for typical problems of such nested control loops like oscillations). But that runs the risk of deteriorating into a conventional approach again. For instance, if we were to decide first on which device an algorithm should run, we are already limiting the decision space even if that device were equipped with both FPGA and CPU; then deciding between FPGA and CPU is merely an afterthought but certainly a considerable reduction in problem size.
The actual challenge here will be to find good divisions into subproblems that only sacrifice marginal amounts of optimality in return for big improvements in feasibility.

Speculative control and management using digital twins: To address the challenge of unknown consequences of adaptation actions, we conjecture that we will need to undertake speculative control or management actions, in the sense of the well-known exploitation-exploration tradeoff commonly considered in machine learning. However, as the consequences

of any such action can be disastrous, it could be prudent to first try out such actions inside a "digital twin"-style representation of the network, running an online simulation of "the network(s)" in possible states, similar to the notion of fictitious self-play known from game theory. This poses challenges regarding the accuracy of such an online simulation, how to update it, the time to invest in such speculation, and when to actual take risks in the real environment, not just in simulation.

AI-powered reasoning: In the past, a major obstacle for versatile context management systems was how data is provided by the supplier, e.g., their logical representation, their format and update frequency. Due to this, for most information processing traces from data source to sink, some kind of proprietary translation is required. But this renders the whole idea of a flexible data provisioning platform futile. Instead, incorporating AI-driven pre-processing can generalize synthesis and representation of data so that it can be inquired by different consumers in various ways [13]. As an example, consider an autonomous robot control system. It needs sensor readings from multiple sources (the robot, its environment, etc.); depending on the robot's current operation mode (moving, at rest), resolution and update frequency need to be adapted. Simultaneously, the robot's digital twin has its information needs, but very different update frequencies, different formats, etc. A smart data provisioning system needs to bridge these gaps towards different data consumers without overloading a participating component.

C. Realization issues

Even once all the conceptual problems of description, monitoring, reachability, optimization, and control should have been solved, there are still plenty of practical realization issues to be addressed. We only mention two examples.

Charging and business models: How and whom to charge for what? E.g., drones might be operated by independent business entities as an on-demand infrastructure, but who pays for them? Operators, because they save money by not having to roll out a worst-case infrastructure? End users, because they directly demand the service? Who assumes the risk for operational instability? There are plenty of business development questions to be answered.

Legal issues: In a corefree/operator-free network, who ensures legal duties like "legal intercept" or "user censorship" (which might be legal and required in some parts of the world, European values notwithstanding); or who, on the contrary, ensures that technology is not used for such purposes?

V. Conclusions

Our current networks are too rigid to deal with foreseeable changes in user needs, acceptance criteria, scenarios, and technological options. We believe that future research has to focus on adaption actions along multiple dimensions: time scales, scopes of various kinds, and structural adaptation. In this whitepaper, we have pointed out some adaptation challenges. We are looking forward to successfully tackle these challenges together with the research community at large.

Acknowledgment

The authors acknowledge the financial support by the German *Federal Ministry for Education and Research (BMBF)* within the project »Open6GHub« {16KISK003K}.
The list of authors is arranged in alphabetical order.

References

[1] P. Bosshart, D. Daly, *et al.*, "P4: Programming protocol-independent packet processors," *SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 3, pp. 87–95, Jul. 2014. DOI: 10.1145/2656877.2656890.

[2] M. A. M. Vieira, M. S. Castanho, *et al.*, "Fast packet processing with eBPF and XDP: Concepts, code, challenges, and applications," *ACM Comput. Surv.*, vol. 53, no. 1, pp. 1–36, Feb. 2020. DOI: 10.1145/3371038.

[3] A. Ortiz, A. Asadi, *et al.*, "SCAROS: A scalable and robust Self-Backhauling solution for highly dynamic Millimeter-Wave networks," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 12, pp. 2685–2698, Dec. 2019. DOI: 10.1109/JSAC.2019.2947925.

[4] D. Yuan, H.-Y. Lin, *et al.*, "Optimal and approximation algorithms for joint routing and scheduling in Millimeter-Wave cellular networks," *IEEE/ACM Trans. Netw.*, vol. 28, no. 5, pp. 2188–2202, Oct. 2020. DOI: 10.1109/TNET.2020.3006312.

[5] P. Wette and H. Karl, "Using application layer knowledge in routing and wavelength assignment algorithms," in *2014 IEEE International Conference on Communications (ICC)*, Jun. 2014, pp. 3270–3276. DOI: 10.1109/ICC.2014.6883825.

[6] S. Dräxler, H. Karl, *et al.*, "Sonata: Service programming and orchestration for virtualized software networks," in *2017 IEEE International Conference on Communications Workshops (ICC Workshops)*, 2017, pp. 973–978. DOI: 10.1109/ICCW.2017.7962785.

[7] B. Alt, M. Weckesser, *et al.*, "Transitions: A protocol-independent view of the future internet," *Proceedings of the IEEE*, vol. 107, no. 4, pp. 835–846, 2019. DOI: 10.1109/JPROC.2019.2895964.

[8] A. Malik, X. Xiao, *et al.*, "Towards coreless wireless mobile networks," in *2018 IEEE Globecom Workshops (GC Wkshps)*, Dec. 2018, pp. 1–6. DOI: 10.1109/GLOCOMW.2018.8644192.

[9] M. Ehrlich, D. Krummacker, *et al.*, "Software-defined networking as an enabler for future industrial network management," in *2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA)*, IEEE, vol. 1, 2018, pp. 1109–1112.

[10] *Study on communication for automation in vertical domains*, 3GPP TR 22.804, 2020.

[11] A. Sterz, M. Eichholz, *et al.*, "ReactiFi: Reactive programming of WI-fi firmware on mobile devices," en, *Art Sci. Eng. Program.*, vol. 5, no. 2, Oct. 2020. DOI: 10.22152/programming-journal.org/2021/5/4.

[12] N. Nomikos, E. T. Michailidis, *et al.*, "A UAV-based moving 5G RAN for massive connectivity of mobile users and IoT devices," *Vehicular Communications*, vol. 25, p. 100 250, Oct. 2020. DOI: 10.1016/j.vehcom.2020.100250.

[13] *ETSI ENI ISG Specifications*, ETSI GS ENI 005 (System Architecture) and related, 2020. [Online]. Available: https://www.etsi.org/standards-search#TB=857.