Engineering software for next-generation networks in a sustainable way

Lidia Fuentes
ITIS Software, Universidad de Málaga, Andalucía Tech
Málaga, Spain
lfuentes@uma.es

ABSTRACT

The virtualization and softwarization of network functions is the networking industry's latest achievement. Software-Defined Networks (SDN) and Network Function Virtualization (NFV) propose novel software architectures and development process adapted to for instance mobile networks (e.g., 6G). However, these architectures and processes are mainly defined by the telecommunications community, without much regard for the contributions of software engineering to generic software processes. This paper explores how the fields of software engineering (SE) and telecommunications can work together to improve service virtualization, cloud computing, and edge computing in the context of next-generation networks. It also highlights the potential of SE fields like software architecture, variability, and configuration to greatly enhance the development of virtual network functions (VNFs). On the other hand, the new contributions should be energy efficient, since this is a primary goal in next-gen networks. Finally, current software processes should consider the impact of communication networks on the correct functioning of software products, since network functioning can affect the QoE of users.

KEYWORDS

network softwarization, virtual network function, network self-adaptation, network variability, network configuration

ACM Reference Format:

Lidia Fuentes. 2024. Engineering software for next-generation networks in a sustainable way. In *Proceedings of International Workshop on Software Engineering in 2030 (SE 2030).* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The advancements of computer networks towards wireless technology are extraordinary, proposing popular interconnection protocols like WiFi or BLE. In the last twenty years, several generations of mobile networks have been proposed. Currently, Beyond 5G (B5G) is under development. With each new generation, the latency has decreased, resulting in a better user experience[8]. More importantly, this has opened up new possibilities for software systems. For instance, it is now possible to control a drone or a robot equipped with a SIM card in real-time from a mobile phone. Thanks to the networking technologies a myriad of devices of different types can be connected to the Internet. Common devices like home appliances, house alarms or smart locks are now permanently connected and can be remotely controlled, something considered impossible twenty years ago. These devices generate data that is usually processed in the Cloud, defining what is called IoT (Internet of Things). All these advancements were accomplished mainly by the telecommunication community and are behind the most important advances of the Computer Science area.

On the software community side, the adoption of Cloud Computing has gained significant popularity since the 2010 decade, and more recently it has been combined with Edge Computing or MEC (Mobile Edge Computing)[2]. Software Engineering (SE) focused its efforts on Cloud computing, proposing novel alternatives to classical client-server architectures, such as Service Oriented Computing (SOC), microservices or serverless architectures, to mention a few. Also, there are many programming languages specific for computing devices, like mobiles phones (e.g., Java for Android) or sensors (e.g., NesC). Moreover, the current trend in software technology is to develop more complex systems like cyber-physical systems, which combine physical and computational components. These systems integrate computation, sensing and networking, containing software that interacts with both physical components and their surrounding environment. Examples of these systems are industrial robots, smart grids or healthcare systems. In conclusion, Software engineering (SE) has progressed alongside the growth and development of computer networks.

On the other hand, the current thriving state of Artificial Intelligence (AI) and specifically machine learning can be largely attributed to the exponential growth of devices that can connect to the Internet, especially using mobile networks. Machine learning requires a vast amount of data to work efficiently, and the higher transmission rate of current mobile networks allows this data to be transferred at runtime. The use of social networks and search engines on mobile phones generates massive amounts of data. However, people often overlook the role of mobile networks in the current success of machine learning algorithms[26].

Both communities of Computer Science (CS) and Telecommunication (Tel) contribute to exploit the advancements of Cloud computing, MEC and Edge computing, but from a different point of view. For example, the CS researchers are more focused on how to compose or orchestrate services running in different virtual machines achieving a desired QoS. But, Tel researchers are more worried about vertical scaling of resources in the cloud, so that

SE 2030, November 2024, Puerto Galinàs (Brazil)

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

these resources can be shared among services in the most fair way and guaranteeing a certain latency. Sometimes the efforts of both communities overlap, while at other times their efforts should be complementary, but are not. The current DevOps trends aim to unify the development and operations phases, which means that developers need to be well-informed about the operational environment in which their software products will be deployed. However, developers do not care if the networking requirements are fulfilled or if the network will have enough resources to support the time restrictions in interactions of the distributed software components that compose the system.

Coming back to the evolution of wireless network technology, the advancements developers have made in the last thirty years are extraordinary. From the initial computer networks where the majority of network functionality was implemented in hardware, like routers and gateways, the tendency in last years is towards the complete softwarization of networks. Network developers coming primarily from the area of telecommunications, or telematics engineering, are experts in signalling and network architectures, but not so much in how to leverage SE methods and technologies to successfully address the present challenges that come with network softwarization.

In this paper we will discuss the synergies between SE and next generation networks. For instance, the intrinsic variability of devices, virtualization platforms and communication protocols could benefit from mechanisms like variability models and configuration management. Also, network operation is a self-adaptive system in nature, since the dimensioning of network resources or the deployment of services (aka service placement) in the most appropriate devices largely depend on current network context (e.g., current resources or number of user requests). At the architectural level, SE has much to say since the classical layered network architecture (e.g., OSI/RM or Internet) should be substituted by a more flexible architecture based on microservices. Indeed, the architectures proposed by Tel organisms are a set of boxes and arrows with no clear semantic since they are not formally defined. Also, since one of the goals of B5G is to incorporate AI natively in the network (e.g., machine learning algorithms that predict resources needed in the near future to allocate them in advance), it is no clear where to deploy these algorithms and how to coordinate them to avoid conflicts and also make them to share overlapping components like monitoring in a consistent way.

Finally, most of the aforementioned technologies were developed with the goal of being more environmentally friendly. Edge computing tries to reduce the energy consumption of uploading and downloading gigabytes of data from the cloud by moving the data processing to Edge devices located nearer data producers devices (e.g., sensors or smart phones). 5G technology was designed to consume less energy, by putting in "sleep state" the base stations when network traffic demands are low, or support multiple data streams simultaneously using the same frequency band, which consume less energy. Indeed, energy efficiency is one of the primary goals of the new generations of mobile networks, so network softwarization should respect this goal.

Several years ago, the SE community was highly interested in measuring the energy consumed by different software libraries[18]. However, due to the difficulties in obtaining accurate results and their dependency on uncontrolled factors (i.e., current machine load, temperature, etc.), researchers almost abandoned this task. Other initiatives focused on proposing green practices to the complete development process, but the energy consumed by using networks is always neglected.

In summary, SE traditionally did not care about the impact of communication networks in the developed system, ignoring they impact in their correct functioning or in the QoS like energy efficiency or response time. More importantly, the networking community that is not expert in designing and maintaining software architectures ignores SE methods, leading to proposing poorly documented network architectures that are impossible to reason about and test. Indeed, some bugs were identified[9] concluding that software defined networks specification and implementation need formal verification and runtime testing of network controller behaviour.

2 NETWORK SOFTWARIZATION

Network softwarization is pivoting the reshaping of Next Generation Internet, accomplishing the decoupling of network functions from the underlying hardware. The end-to-end softwarization of networks is at the basis of the 5G networks, pushing a radical change in the design of mobile systems. Exploiting network softwarization is fostering academic research on Beyond 5G (B5G) or the sixth generation (6G) networks, with a compromise of providing diverse tailored network services and emerging cloud-edge applications. The interconnected devices eventually part of the Internet of Things (IoT), Cloud and Edge computing ecosystem, require billions of network connection devices containing general purpose or specific network services that can be implemented in software and run in commodity devices[24].

Traditionally, network functions (e.g., firewalls, proxy servers, or intrusion detection systems) run on powerful and proprietary hardware dedicated to a particular network function. Software Defined Networking (SDN)[14] and Network Function Virtualization (NFV)[1], are widely accepted paradigms to address the new structure of mobile network architectures. SDN pursue to overpass the network functions dependency of hardware by separating the data plane (i.e., SDN Datapath embedded in network devices like routers), from the application plane (i.e., network functions), by putting an intermediary, the SDN controller[1].

On the other hand, NFV is an innovative, but complementary paradigm that promotes the virtualization technology to disengage network functions from dedicated hardware appliances and transform them into software components, so called virtual network functions (VNFs) that run at the application plane (i.e., as SDN applications) on commercial off-the-shelf (COTS) servers (Figure 1). Then, user applications demanding a network service turn out to be a request for running a set of VNFs composing a Service Function Chain (SFC) deployed on the network infrastructure. In this scenario, the European Telecommunications Standards Institute (ETSI) proposed the standard reference architecture for the Management And Network Orchestration (MANO)[1] of VNFs, which specifies the core operation functionality, including the NFV Orchestrator that controls the physical and virtual infrastructures, and the VNFs placement inside the virtual network. VNFs differ from traditional

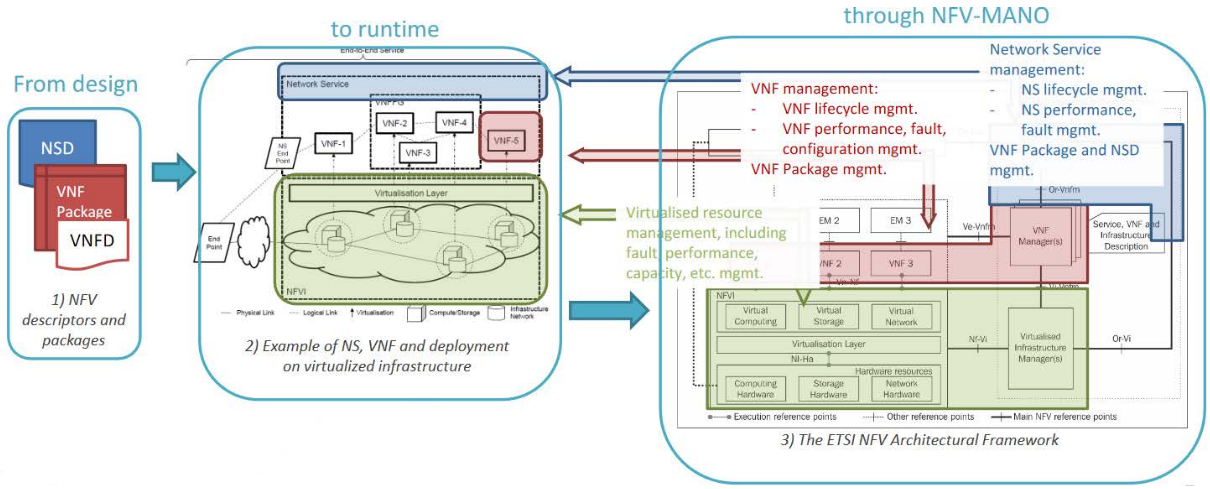

Figure 1: Release 2 specifications of the NFV framework.

Physical Network Functions (PNFs) in that the latter ones refer to legacy network functions implemented in proprietary hardware. And, in a cloud-native world, Cloud-native Network Functions (CNFs) running on cloud servers (e.g., video streaming service) are a good alternative that complement both, VNFs and PNFs. In the hereafter we will refer to all of them as VNFs.

As networks evolve from 5G to 6G a novel paradigm, network slicing (NS)[14] is becoming a fundamental technology that empowers mobile networks with the capacity of supporting a wide range of use cases as distinct, separable, and relatively isolated business processes, running in the user plane[4]. NS approach allows to define multiple logically isolated virtual networks for different services running in a common physical network, encapsulated in containers or in virtual machines. These application services (i.e., VNFs) inside a NS can be tailored for certain applications family (e.g., virtual reality, video delivery or distributed games), domains like IoT, or allocated to a class of customers, a class of service requiring a specific SLA (Service Level Agreement) or to certain mobile network operators. For example, smart factory line needs Ultra Reliable Low Latency Communications (URLLC), but sport event retransmission needs mainly high-bandwidth and dynamic scaling. Combining multiple services from different domains in VNFs chains provided to customers/tenants as a service, is the general trend in 5G/6G. Note, that these services also can combine many of the SLA characteristics above.

The main benefits of B5G/6G network softwarization and virtualization are rapid service innovation, flexible network function configuration and deployment, and improved resource usage, providing tangible economic and technical benefits. Indeed, 6G networks promise to provide a large set of agile services, custom made and providing user-defined Quality of Service (QoS). The typical key performance indicators (KPI) associated with 6G wireless networks are user-experience (QoE), data rate or latency among others. In addition, network practitioners consider energy efficiency of primary importance to compensate for the additional energy cost of softwarization and virtualization. In addition, VNFs deployment and operation should adapt to an ever-changing context, mainly in terms of traffic demand, available resources, network structure and node failures, but always keeping the required QoS. In this dynamic context, when a web application requires to activate a certain SFC composed by VNFC (VNF components), the orchestrator manager is responsible for finding out and instantiating the corresponding technical configuration to make it work, or figure out deployment descriptors when they are missing[29].

Each new release of NFV-MANO architecture specification proposes to incorporate new capacities to smooth the transition to a total software based development and continuous integration of VNFs. As shown in Figure1 the NFV-MANO challenges covers the design and runtime phases of VNFs. While the instantiation of NFV-MANO components is under progress, key areas of focus have been identified, which include[1]:

- Support lightweight virtualization technologies such as OS containers, optimizing NFV Infrastructure (NFVI) abstraction for reducing the coupling of VNFs to infrastructure
- Improving life-cycle management and orchestration, introducing more policy-based management and handling advances in autonomous and self-adaptive networking
- Studying the NFV-MANO service-based transformation defining a Service-based architecture (SBA), Multi-access Edge Computing (MEC) and Cloud-native VNFs
- Simplification of NFV-MANO to ease development and deployment of sustainable NFV based solutions, verification (and certification)
- Continuous VNF integration, software upgrades/updates of the VNF according to Policy models

3 CONFIGURATION ENGINEERING FOR NEXT GEN NETWORKS

Considering that VNFs are essentially software components themselves operating in multivendor environments, Software Engineering methods and technologies have much to say to tackle the aforementioned issues[22, 28]. Concretely, variability and configuration

[1]ETSI NFV-MANO https://www.etsi.org/technologies/nfv

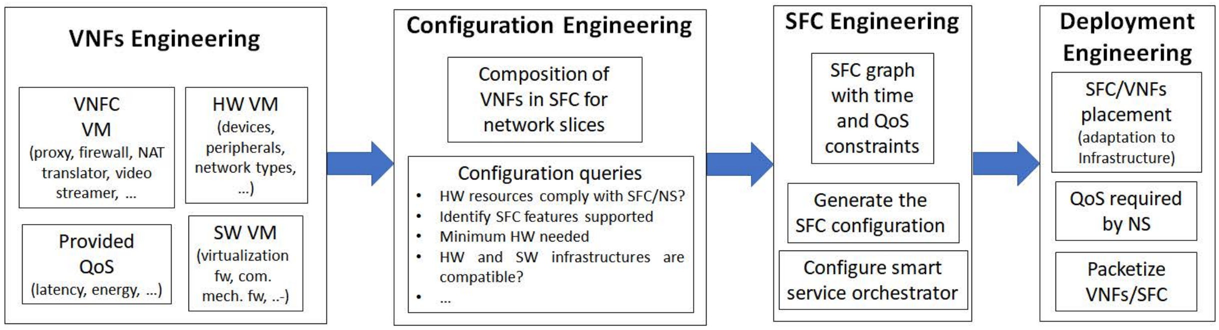

Figure 2: VNFs/SFC/NS pipeline process

management technologies, are an appropriate approach to deal with the high diversity of domain specific network service functions and their interdependencies with legacy VNFs, virtualization platforms and underlying infrastructure. One of the proposals is to produce the corresponding *technical configuration* that makes everything work fine, considering different IoT/Edge/Cloud contexts inside B5G/6G networks.

According to[28] "configuration management is a well-understood, universally practiced discipline, at least in its simplest form: version control. The focus in the high-consequence context is sensibly managing the relationships between features, deployment platforms, and architectural entities". The software product line (SPL) and variability community has been focusing on configuration techniques and methods, like sampling relevant configurations, performance prediction or finding optimal configurations, which usually consider only a static environment with a constant workload [6], but this is not the case of many application domains like mobile networks. Indeed, this community should move towards more realistic environments posed by current practice in networked systems, such as virtualization and containerization, serverless architectures, cloud infrastructure and deployment automation.

B5G/6G networks are a clear application area that needs sound configuration techniques that enable not only to configure a VNFs chain (SFC, service function chain is a sequenced VNFs set to serve a specific web service[4]) adapted to tenants' requirements, but also to configure the software system's environment and infrastructure under variable conditions. Unfortunately, the mobile network community does not address the development and continuous deployment and orchestration of custom-made VNFs from the point of view of variability and configuration management, being unaware of the connections between different configuration activities performed at different times and dimensions, leading to crashes. As an example, the social networks giant Facebook, experienced a blackout that interrupted their services in October/2021 for hours and they concluded that it was caused by: "Configuration changes in central routers that coordinate network traffic between data centers. This change in network traffic affected the way centers communicate and disrupted services".

Therefore, configuration management in IoT/Edge/Cloud contexts is a complex task that cuts across the whole software life cycle, which involves diverse stakeholders (e.g., VNF developers, providers, tenants), artefacts, infrastructures, and should be tackled consistently in a stepwise manner at different times. In the case of mobile networks several configuration dimensions should be considered, starting in the development phase by tailoring the network functionalities, go through the technical configuration required by a virtualized infrastructure, considering dynamic QoS requirements, and, mostly important, a highly variable context (e.g., network traffic, nodes' workload). So, instead of defining a fix VNF catalogue, as suggested by ETSI specification, our proposal is that the NFV orchestrator queries the variability model to determine which VNFs provide a certain quality or require the least amount of resources, among others.

In order to do so, the definition of multi-layer (multiview) variability models[3] would help to specify hardware (e.g., devices resources like memory CPU or GPU, peripherals like cameras, communication protocols with their bandwidth, etc.) and software infrastructure (e.g., virtualization framework like Kubernetes, communication type like MQTT, WebSockets or publish-and-subscribe platforms like Zenoh-flow[6]) and specify the crosstree constraints between both models. The family of VNFs should adapt to the concrete infrastructure available in each network slice and require the QoS or QoE needed for example by a smart factory.

Another objective should be to characterize the different phases involved in the configuration of virtualized mobile networks, during the development, deployment and execution of network services inside a network slice. It is worthy to investigate the variability present in artifacts and activities required by virtualization, containerization, microservices, serverless, cloud computing, resource orchestration, KPIs and network infrastructure, and define a configuration lifecycle adapted to Edge/Cloud/6G environments. The idea is to manage the independent configuration of VNFCs (VNFs' components) and the hardware and virtualized infrastructures in a

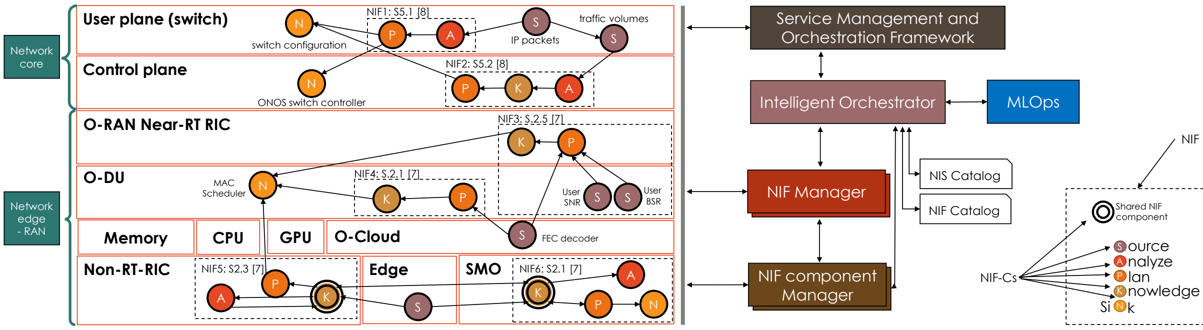

Figure 3: Ni-native architectural concept proposed by the DAEMON project for the ni stratum.[13]

consistent way, promoting reuse across network slices at the user plane.

One work towards this objective is outlined in[10][2]. This approach introduces a special phase called *Configuration Engineering*, which aims to include the configuration of hardware and software infrastructure as a preliminary stage of the SPL application engineering process. The configuration engineering comprises: (i) identification of the capacities of hardware infrastructure to comply with the necessities of software infrastructure and application configuration in terms of resources; (ii) identification of supported or not supported application features (e.g., none of the devices has a camera needed by the application); (iii) identify the minimum of hardware infrastructure and its resources (i.e., devices, machines, networks) needed to support a family of VNFCs fulfilling a good quality of service.

Another approach could be to define different configuration views for different stages (e.g., develop, install, deploy and runtime), artifacts (e.g., variable, code, container descriptions, automation scripts), environments (e.g., network slices, containers, virtual machines), purposes (e.g., artificial intelligence support, testing, quality assurance), as well as the stakeholders (e.g., VNF developers, providers, tenants). Concretely Once the functionality of a network service is defined and several versions are implemented using different programming languages and framework(s), the idea is to automate as possible each view's configuration activities, so that they can be accomplished in a consistent and efficient way. Figure**??** shows the pipeline of SFCs/NS life cycle corresponding to the phases before runtime.

Initially, what correspond to domain engineering in the SPL process, variability models of hardware, software and VNFs is defined, enriched with features and attributes needed to assess the QoS (e.g., CPU type is needed for energy efficiency). Next, the configuration engineering phase is carried out to ensure that the hardware and software infrastructure comply with the SFC requirements. Since network slices are specific for a certain type of application services they require similar SFCs. These SFCs are usually represented by a graph that can be enriched with time, resource or any QoS constraints (e.g., a VNFs must be executed in less than 2 ms, or other two VNFs can be executed in parallel, but before the last VNFs of the graph). If different implementations offer varying quality, it would be interesting to generate a graph configuration that optimizes a specific QoS, such as energy consumption. On the other hand, several mobile network architectures include smart orchestrators, which allocate resources to SFCs over a shared infrastructure. Different resource orchestration policies can be applied, such as vertical scaling of resources, or instantiating containers before they are needed to reduce the deployment time. So, in this phase it would be interesting to select and configure the orchestrators that will be deployed in a certain network slice[12].

VNF providers need to tailor VNFs/SFCs to their tenants needs and deliver their products with the required configuration files to be deployed and operated under various execution environments and infrastructures. Considering VNF providers may have hundreds of tenants, a desired goal is to automate the delivery of packetized SFC products carrying all the necessary configuration descriptors, i.e. the technical configuration, and assure its distribution free of errors. With this information, network functions orchestrators should be able to find out how to deploy network services and achieve the necessary resources to guarantee their correct operation.

In summary, **little emphasis in solving configuration networks problems were dedicated by the SE community, including SPL and Configuration areas**. In some variability models, sometimes appear features that model concepts related to networks, the typical WiFi or Bluetooth alternative, but they are not delved deeper.

4 SELF-ADAPTIVE NETWORK OPERATION

As we have already commented, network slicing and NFV-MANO architecture are two key concepts in modern networking, especially in the context of B5G. Network slicing aims to realize the concept of *Zero-Touch Operation*, which requires end-to-end network automation for 5G. Zero-touch operation for network slicing aims to automate the entire life cycle, including installation, deployment, configuration, and operation. Integrating NS with SBA architectural approach that decomposes network functions into smaller, modular services, the VNFs and SFCs can be more easily deployed and managed. One trend is to consider Network Slice-as-a-Service (NSaaS) creating new services that use virtual resources from the existing infrastructure in isolation[27]. With SBA is easier to create more flexible and scalable network slices adapted to changing context and requirements.

[2] Angel Cañete, PhD thesis, "Energy-aware function and resource management in next-gen networks with variability models", March 2024, to appear

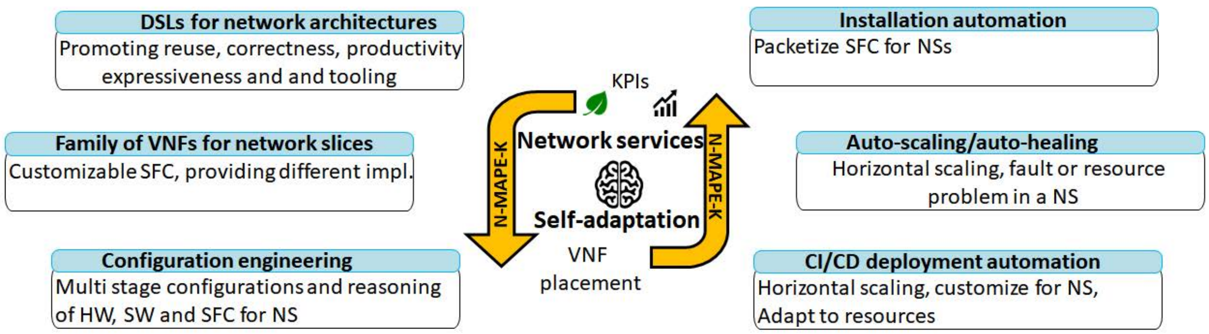

Figure 4: Open issues

In SE an important area of research proposes methods, techniques, processes, and tools to support the construction of safe, cost-effective self-adaptive and autonomous systems that provide self-* properties like self-configuration, self-healing, self-optimization, and self-protection[3]. And that is precisely what they refer to as zero-touch operation in the Tel area. So, NFV-MANO and NS architectures are self-adaptive in nature, having to guarantee Service Level Agreement (SLA) conditions. The instantiation of SFCs for a particular service (i.e., a web service or mobile app) should be accomplished in a self-adaptive manner reaching a compromise between the QoS requirements and the resource availability.

However, the lack of networks domain-specific description languages to describe service characteristics, KPIs, resources, and requirements for slices makes it challenging to translate the high-level description of the service into the concrete slice in terms of infrastructure and network functions[25]. These information should be part of the service descriptor files, so that can be processed at runtime. SFCs are part of the service requirements and it would be desirable to custom their VNFC for each network slice.

Indeed, the dynamic variability of SFC can be expressed as that each VNF composing the chain can change its configuration parameters (e.g., transmission rate) or implementation at runtime. But, this decision depends on current network conditions that need to be monitored constantly, like: (i) current traffic load; (ii) congestion level of network nodes; (iii) availability of edge nodes to offload code if necessary to maintain a quality of service; (iv) number of users and their request rate; or (v) if a cloud machine with high capacity where always is possible to deploy VNFs is part of the infrastructure.

In SE community one of the most used reference models for self-adaptive systems is the well known MAPE-K loop. However, the MAPE-K reference model is almost completely unknown by the Tel community. Recently, was proposed an extension of this model (Networking MAPE-K or N-MAPE-K) considering the peculiarities of the mobile networks[17]. This model was defined in the context of the European project DAEMON[4], "Network intelligence for aDAptive and sElf-Learning MObile Networks", where we were partners. This model was extended considering Machine Learning (ML) techniques (N-MAPE-K[17]), pushed by the increased availability of measurement data and computational resources within mobile networks to develop Network Intelligence (NI) that will enable fully autonomous 6G networks[13]. This initial model was extended with AI functions, where the Knowledge of the N-MAPE-K is being trained or used in inference for the operation of the network following the MLOps paradigm for supervised and reinforcement learning.

This initial model was extended with a Network Intelligence stratum that "complements and interacts with the existing planes in current and next-generation mobile networks, i.e, the user/-data, control, and management planes"[13]. In this second version they add to the traditional supervised training loop, a second training loop dedicated to online learning as part of the B5G network architecture[13]. This new version is focused on NISs (Network Intelligence Service) that complement the functionality of Network Intelligence Orchestration (NIO) exploiting with ML algorithms the monitored data. A NIS is composed of one or more Network Intelligence Functions (NIFs), which are composed of NIFs (Network Intelligence Functions) Components (NIFCs). In order to reuse NIFC components they can be shared by different NIFs (see double circle in figure 3). In this cases there are possible conflicts on operations and/or resources, being the NIO who arbitrates the operation of such components, guaranteeing that the overall goal of the NIS is met.

[3]SEAMS conference

[4]DAEMON project https://h2020daemon.eu/

However, this architecture is not formally specified and the interplay between the network intelligence functions and the self-adaptive loop is not clearly defined. As can be appreciated in figure 3 some of the MAPE-K loop components were renamed, such as Monitoring is Source and Execution is Sink. The meaning of arrows is not specified. The N-MAKE-K loop is replicated in different planes and layers, sometimes as part of a certain NIF. For example, in the user plane of a switch (figure 3) the IP datagrams are monitored obtaining the current traffic volume, which is analysed by the NF1 that generates a plan to change the switch configuration. A new version of the N-MAPE-K was proposed in [15], which is more aligned with the SE principles. However, this version does not explicitly includes the ML pipeline, which is an important limitation.

On conclusion, several extensions of the MAPE-K architecture to incorporate AI functions was already studied by the SE community[16, 31], but none of them address the specific problems of zero-touch B5G networks. However, **according to the survey[30] the 72.2% participants who work in the domain of ICT communication and networks would regularly appreciate support to address their self-adaptive problems**, something I can confirm first-hand.

5 ENERGY-AWARE NEXT GENERATION NETWORKS

Environmental sustainability is one of the innovative targets of future telecommunication systems. At the same time, electricity expenditure is a main cost for mobile network operators already today, and it is expected that the transition to a fully virtualized infrastructure will increase such a cost substantially. In spite of these considerations, little attention has been paid to understanding how network management decisions impact energy consumption. One of the goals is to create profiles for specific Virtual Network Functions (VNFs) and Service Function Chains (SFCs) to accurately calculate their energy consumption. This will introduce a new approach for the design of VNFs that are "energy-aware" by making them smart enough to automatically adapt their energy consumption to the context. Additionally, orchestration algorithms will be designed to account for the energy footprint of VNFs in different locations when deciding their placement. This will help in achieving the best balance between network performance and energy consumption.

Auto-scaling, a critical mechanism for VNF orchestration, systematizes the growth or reduction of the system's capability that is offered for use by VNFs tenants (i.e., user end application or web services). Vertical scaling increases the capacity of existing infrastructure by adding more resources, and horizontal scaling increases the resources of the infrastructure by adding more nodes. Both scaling options and hybrid ones, should consider the cost of scaling up or down resources from the point of view of the energy footprint.

The Edge and Cloud resources provide new opportunities to make a more efficient VNF/SFC and resource orchestration, reducing not only latency, but also the energy footprint, by placing some VNFs closer to the data they process. A number of works focus on the VNF placement problem (VNF deployment) as a cost minimization problem which is solved using an Integer Linear Program (ILP) or Mixed ILP[21]. However, these solutions for VNF placement fail to provide the explicit modelling of SFC[23] or consider the VNFC requirements (such as CPU demands) as part of the placement algorithm. And only few of them consider energy consumption as one of the targeted objectives[11, 15].

In [11] an energy-aware VNF placement algorithm is proposed. The energy footprint of VNFC is estimated by a couple of formulas, one for computation energy and the second one for communication energy. In this case, the energy estimation is used to select the most eco-friendly implementation of a VNFC. Therefore, the accuracy of energy calculation is not important, but rather the comparison of energy footprint between the implementations. Another alternative is to use linear regression models to estimate the energy footprint of code, like GreenMiner[19], but the results highly depend on the hardware where the benchmark was executed not being the greenest approach. Others propose a static-analysis approach that "estimates the energy consumption of API usage in an app, eliminating the need for test case execution"[7].

One green politics that can be applied to reduce the impact of ICT on global warming is to promote the use of machines fed by renewable energy. The VNF placement algorithm of[11] multiplies by a reduction factor the energy estimated for green nodes.

Regarding self-adaptive architectures that use ML algorithms for decision making, they probably involve greater energy consumption especially if they need to train algorithms continuously. In these case proactive strategies based on Control Theory to consider not only the current context, but also the system's expected evolution over time (over a prediction horizon), so their scaling solutions tend to stay valid for longer[5]. In [5] is demonstrated that this politic consume less energy than a reactive one, since the system makes less reconfigurations.

On the other hand, as we already discussed virtualization continues to evolve and being more popular. One of the most used is Kubernetes, "a portable, extensible, open source platform for managing containerized workloads and services"[5]. One recent power meter tool is Scaphandre[20], that is able to estimate the energy consumption not only of an entire machine, but of virtual machine and containers, such as Kubernetes pods. These power consumption measurements can be exported to Prometheus and Grafana. The advantage of this tool is that shows the power consumption in real time, so service orchestrators could make energy aware decisions.

Everybody agrees that the ICT sector should drastically reduce its energy footprint, but it is unclear how to make it in a sustainable way..

6 DISCUSSION AND CONCLUSIONS

Figure 4 pretends to be a summary of the main issues of SDNs and 6G that could benefit from SE methods. Some of the activities are related to design phases and other to installation, deployment and self-adaptation.

We discussed the the importance of creating domain-specific description languages for the upcoming networks. These languages would be used to define various service characteristics, KPIs, resources, and requirements for slices. Additionally, we outline the significance of specifying network architectures and their interactions with virtualization and cloud architectures. Doing so would enable a better understanding and greater expressiveness of these

[5]Kubernetes https://kubernetes.io/docs/concepts/overview/

architectures. Furthermore, the current trend is to define service-based architectures using microservices or serverless architectures.

Mobile Network Operators (MNOs) are implementing a diverse range of Virtualized Network Functions (VNFs) along with their inner Virtualized Network Function Components (VNFCs). These VNFs encapsulate various network functions such as NAT translation, as well as more sophisticated functions including video streaming. MNOs are integrating these VNFs natively into their network resources to enhance their service offerings. The Tel area is aware that these VNFCs should be reusable and shareable across several network slices. It is important to offer different implementations of VNFCs so that network service orchestrators can choose the most appropriate ones based on each NS and current resources.

Configuration engineering allows the automatic configuration and reasoning on several variability models, modeling SFCs, hardware and software infrastructures. This models will help to answer questions like, does a NS has the hardware and software infrastructure to support a certain SFC? By analysing these models based on a certain QoS (e.g., energy consumption), stakeholders can reason about how well NS infrastructure matches current web services needs.

Installation automation requires the massive and automatic installation of packetized SFC products that could be installed in a cloud infrastructure or in other devices. The "it works on my machine" mindset can lead to poor software quality and unreliability in production. To avoid this, network providers should use virtualization techniques and also test SFCs if necessary using emulators. They should also ensure that each SFC has everything necessary to guarantee its correct operation.

SFC cannot be delivered as-is but needs to be continuously supported and updated by network operators. CI/CD (Continuous Integration/Continuous Deployment) should be integrated as part of SDN operation processes, defining an agile software life cycle. Deployment automation is related with SFC/VNF placement and resource scaling. Both processes consist of finding the optimum deployment of a set of VNFCs for a NS infrastructure, considering a certain QoS, such as energy efficiency. Here is also included the evolution of SFCs as long as new requirements of NS or from users appear.

According to MANO architecture, the Network Intelligence Orchestrator has to detect any fault or resource issue and try to recover it. This manager can decide to offload, restart or re-deploy VNFCs to guarantee a certain QoS. And finally, in the center the N-MAPE-K loop uses AI to optimize network slices automatically based on policies that assures a certain SLA. We have also discussed that sustainability is one of the primary goals of 6G, so reducing the power consumption should be part of any intelligent network process.

On the other hand, network resources and capacities should also be incorporated to any software development process. A service provision sometimes fails because the network resources are not enough to accommodate a sudden increase in user requests (e.g., the final game of a football World Cup).

REFERENCES

[1] 2014. *Network Functions Virtualisation standards.* Technical Report.

[2] Nasir Abbas, Yan Zhang, Amir Taherkordi, and Tor Skeie. 2018. Mobile Edge Computing: A Survey. *IEEE Internet of Things Journal* 5, 1 (2018), 450–465. https://doi.org/10.1109/JIOT.2017.2750180

[3] Mathieu Acher, Philippe Collet, Alban Gaignard, Philippe Lahire, Johan Montagnat, and Robert B. France. 2012. Composing multiple variability artifacts to assemble coherent workflows. *Software Quality Journal* 20, 3–4 (sep 2012), 689–734. https://doi.org/10.1007/s11219-011-9170-7

[4] Hassan S.A. Ghaffar R. et al. Akhtar, M.W. 2020. The shift to 6G communications: vision and requirements. *Hum. Cent. Comput. Inf. Sci. 10, 53* (2020).

[5] Inmaculada Ayala, Alessandro V. Papadopoulos, Mercedes Amor, and Lidia Fuentes. 2021. ProDSPL: Proactive self-adaptation based on Dynamic Software Product Lines. *J. Syst. Softw.* 175 (2021), 110909. https://doi.org/10.1016/J.JSS.2021.110909

[6] Gabriele Baldoni, Julien Loudet, Luca Cominardi, Angelo Corsaro, and Yong He. 2021. Facilitating distributed data-flow programming with Eclipse Zenoh: the ERDOS case. In *Proceedings of the 1st Workshop on Serverless Mobile Networking for 6G Communications* (Virtual, WI, USA) *(MobileServerless'21)*. Association for Computing Machinery, New York, NY, USA, 13–18. https://doi.org/10.1145/3469263.3469858

[7] Abdul Ali Bangash, Kalvin Eng, Qasim Jamal, Karim Ali, and Abram Hindle. 2023. Energy Consumption Estimation of API-usage in Smartphone Apps via Static Analysis. In *2023 IEEE/ACM 20th International Conference on Mining Software Repositories (MSR)*. 272–283. https://doi.org/10.1109/MSR59073.2023.00047

[8] Mudit Ratana Bhalla, Anand Vardhan Bhalla, et al. 2010. Generations of mobile wireless technology: A survey. *International Journal of Computer Applications* 5, 4 (2010), 26–32.

[9] Ayush Bhardwaj, Zhenyu Zhou, and Theophilus A. Benson. 2021. A Comprehensive Study of Bugs in Software Defined Networks. In *2021 51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. 101–115. https://doi.org/10.1109/DSN48987.2021.00026

[10] Angel Cañete, Mercedes Amor, and Lidia Fuentes. 2022. Supporting IoT applications deployment on edge-based infrastructures using multi-layer feature models. *J. Syst. Softw.* 183 (2022), 111086. https://doi.org/10.1016/J.JSS.2021.111086

[11] Angel Cañete, Mercedes Amor, and Lidia Fuentes. 2024. HADES: An NFV solution for energy-efficient placement and resource allocation in heterogeneous infrastructures. *J. Netw. Comput. Appl.* 221 (2024), 103764. https://doi.org/10.1016/J.JNCA.2023.103764

[12] Livia Elena Chatzieleftheriou, Marco Gramaglia, Miguel Camelo, Andres Garcia-Saavedra, Evangelos Kosmatos, Michele Gucciardo, Paola Soto, George Iosifidis, Lidia Fuentes, Gines Garcia-Aviles, Andra Lutu, Gabriele Baldoni, and Marco Fiore. 2023. Orchestration Procedures for the Network Intelligence Stratum in 6G Networks. In *2023 Joint European Conference on Networks and Communications 6G Summit (EuCNC/6G Summit)*. 347–352. https://doi.org/10.1109/EuCNC/6GSummit58263.2023.10188297

[13] Livia et al. Chatzieleftheriou. 2023. Orchestration Procedures for the Network Intelligence Stratum in 6G Networks. In *2023 Joint European Conference on Networks and Communications 6G Summit (EuCNC/6G Summit)*. 347–352. https://doi.org/10.1109/EuCNC/6GSummit58263.2023.10188297

[14] Massimo Condoluci and Toktam Mahmoodi. 2018. Softwarization and virtualization in 5G mobile networks: Benefits, trends and challenges. *Computer Networks* 146 (2018), 65–84. https://doi.org/10.1016/j.comnet.2018.09.005

[15] Sedef Demirci, Seref Sagiroglu, and Mehmet Demirci. 2020. Energy-efficient virtual security function placement in NFV-Enabled networks. *Sustainable Computing: Informatics and Systems* 30 (12 2020), 100494. https://doi.org/10.1016/j.suscom.2020.100494

[16] Omid Gheibi, Danny Weyns, and Federico Quin. 2021. Applying Machine Learning in Self-adaptive Systems: A Systematic Literature Review. *ACM Trans. Auton. Adapt. Syst.* 15, 3, Article 9 (aug 2021), 37 pages. https://doi.org/10.1145/3469440

[17] Marco et al. Gramaglia. 2022. Network Intelligence for Virtualized RAN Orchestration: The DAEMON Approach. In *2022 Joint European Conference on Networks and Communications 6G Summit (EuCNC/6G Summit)*. 482–487. https://doi.org/10.1109/EuCNC/6GSummit54941.2022.9815816

[18] Samir Hasan, Zachary King, Munawar Hafiz, Mohammed Sayagh, Bram Adams, and Abram Hindle. 2016. Energy profiles of Java collections classes. In *Proceedings of the 38th International Conference on Software Engineering* (Austin, Texas) *(ICSE '16)*. Association for Computing Machinery, New York, NY, USA, 225–236. https://doi.org/10.1145/2884781.2884869

[19] Abram Hindle, Alex Wilson, Kent Rasmussen, E. Jed Barlow, Joshua Charles Campbell, and Stephen Romansky. 2014. GreenMiner: a hardware based mining software repositories software energy consumption framework. In *Proceedings of the 11th Working Conference on Mining Software Repositories* (Hyderabad, India) *(MSR 2014)*. Association for Computing Machinery, New York, NY, USA, 12–21. https://doi.org/10.1145/2597073.2597097

[20] https://github.com/hubblo-org/scaphandre. [n. d.]. .

[21] Stanislav Lange, Nguyen Van Tu, Se-Yeon Jeong, Do-Young Lee, Hee-Gon Kim, Jibum Hong, Jae-Hyoung Yoo, and James Hong. 2020. A Network Intelligence Architecture for Efficient VNF Lifecycle Management. *IEEE Transactions on Network and Service Management* PP (08 2020), 1–1. https://doi.org/10.1109/TNSM.2020.3015244

[22] Cuong Le and Myungsik Yoo. 2020. An Extended Agent-Based Mechanism For Testing Service Function Chain. In *International Conference on Information and Communication Technology Convergence, ICTC 2020, Jeju Island, Korea (South), October 21-23, 2020.* IEEE, 506–509. https://doi.org/10.1109/ICTC49870.2020.9289215

[23] Yicen Liu, Hao Lu, Xi Li, Yang Zhang, Leiping Xi, and Donghao Zhao. 2020. Dynamic Service Function Chain Orchestration for NFV/MEC-Enabled IoT Networks: A Deep Reinforcement Learning Approach. *IEEE Internet of Things Journal* PP (11 2020), 1–1. https://doi.org/10.1109/JIOT.2020.3038793

[24] Qingyue Long, Yanliang Chen, Haijun Zhang, and Xianfu Lei. 2022. Software Defined 5G and 6G Networks: a Survey. *Mob. Netw. Appl.* 27, 5 (oct 2022), 1792–1812. https://doi.org/10.1007/s11036-019-01397-2

[25] Noel Crespi Meroua Moussaoui, Emmanuel Bertin. 2022. 5G shortcomings and beyond-5G/6G requirements. *International Conference on 6G Networking (6GNet)* (2022).

[26] M. G. Sarwar Murshed, Christopher Murphy, Daqing Hou, Nazar Khan, Ganesh Ananthanarayanan, and Faraz Hussain. 2021. Machine Learning at the Network Edge: A Survey. *ACM Comput. Surv.* 54, 8, Article 170 (oct 2021), 37 pages. https://doi.org/10.1145/3469029

[27] Kibeom Park, Sangmo Sung, Hokeun Kim, and Jae il Jung. 2023. Technology trends and challenges in SDN and service assurance for end-to-end network slicing. *Computer Networks* 234 (2023), 109908. https://doi.org/10.1016/j.comnet.2023.109908

[28] R. Striemer V. Gruhn. 2018. *The Essence of Software Engineering.*

[29] et al. W. S. Atoui. 2019. Virtual Network Function Descriptors Mining using Word Embeddings and Deep Neural Networks. *Symposium on IM pp. 515-520* (2019).

[30] Danny Weyns, Ilias Gerostathopoulos, Nadeem Abbas, Jesper Andersson, Stefan Biffl, Premek Brada, Tomas Bures, Amleto Di Salle, Matthias Galster, Patricia Lago, Grace Lewis, Marin Litoiu, Angelika Musil, Juergen Musil, Panos Patros, and Patrizio Pelliccione. 2023. Self-Adaptation in Industry: A Survey. *ACM Trans. Auton. Adapt. Syst.* 18, 2, Article 5 (may 2023), 44 pages. https://doi.org/10.1145/3589227

[31] Danny Weyns, Bradley Schmerl, Masako Kishida, Alberto Leva, Marin Litoiu, Necmiye Ozay, Colin Paterson, and Kenji Tei. 2021. Towards Better Adaptive Systems by Combining MAPE, Control Theory, and Machine Learning. In *2021 International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS).* 217–223. https://doi.org/10.1109/SEAMS51251.2021.00036