|  |  |
| --- | --- |
| D:\Mohr\Arbeitsverzeichnis\Europäische Kommission\Mobile Communications & Technology Platform\EU Technology Platform\Logos NetWorld2020\NetWorld2020 large logos\NW 2020 LBB\NW2020.jpg | logo-5G-positive_ia-text |
|  | February 2020 |

**Strategic Research and Innovation Agenda 2021-27**

**European Technology Platform NetWorld2020**

**“Smart Networks in the context of NGI”**

**DRAFT  
work in progress – not to be distributed**

**2020**

# Table of Contents

[Table of Contents 2](#_Toc40879267)

[4 System Architecture 3](#_Toc40879268)

[4.1 Evolution of Networks and Services 3](#_Toc40879272)

[4.2 System Architecture Vision: Towards Smart Green Systems 5](#_Toc40879273)

[4.3 Virtualised Network Control for Increased Flexibility 7](#_Toc40879274)

[4.3.1 Programmability is Control 7](#_Toc40879275)

[4.3.2 Separation of control/controllability 8](#_Toc40879276)

[4.3.3 Multi-Tenancy and Ownership 8](#_Toc40879277)

[4.3.4 Known Unknowns 9](#_Toc40879278)

[4.3.5 Self-Preservation 9](#_Toc40879279)

[4.3.6 Research Challenges 10](#_Toc40879280)

[4.4 Re-Thinking the Data & Forwarding Planes 10](#_Toc40879281)

[4.4.1 Design Considerations for an Evolved Data & Forwarding Plane 10](#_Toc40879282)

[4.4.2 Key Research Questions 13](#_Toc40879283)

[4.4.3 Recommendations for Actions 14](#_Toc40879284)

[4.5 Efficiency and Resource Management 14](#_Toc40879285)

[4.5.1 Network Slicing versus Network Capacity Planning 15](#_Toc40879286)

[4.5.2 Slicing Requires Conflict Resolution 15](#_Toc40879287)

[4.5.3 Elasticity: Slicing Efficiency Requires Runtime Scheduling 16](#_Toc40879288)

[4.5.4 Towards Green ICT 16](#_Toc40879289)

[4.5.5 Research Challenges 17](#_Toc40879290)

[4.6 AI/ML-based System Evolution 18](#_Toc40879291)

[4.6.1 Proliferation of AIaaS in Network Operations 18](#_Toc40879292)

[4.6.2 AIaaS Proliferation in Service Provisioning 19](#_Toc40879293)

[4.6.3 Related Research Challenges 20](#_Toc40879294)

[4.6.4 Recommendations for Future Actions 21](#_Toc40879295)

[4.7 Deep Edge, Terminal and IoT Device Integration 21](#_Toc40879296)

[4.8 References 24](#_Toc40879297)

# System Architecture

## Evolution of Networks and Services

Distributed computing has taken a significant step forward with the development and utilization of the Internet in many industries, pushing the digitization of processes and opening opportunities for creating or improving many business-to-business (B2B) and business-to-customer (B2C) processes. It does so, however, on the back of an Internet, whose core design is almost 50 years old, starting in the 1970s on very basic assumptions of an end-to-end connectivity between two remote machines, usually denoted as *client* and *server*. Inter-domain connectivity, enabled through the overall IP suite, allowed for reaching any machine through a multi-tier architecture of autonomous systems (ASs). This basic principle, unchanged to this day, had to shoulder the burden of *service routing*, i.e., associating a request to an instance of a service name. This had to be mapped to a combination of hostname and service path and, ultimately, a machine locator, i.e., IP address, bound to that service name.

While unchanged in principle, many things have evolved from this basic picture of Internet connectivity. In the following, we differentiate three aspects, namely the *nature of communication* over the Internet, the *nature of services* (and their relation) and the *nature of provisioning* in the serving endpoints that are being reached via the Internet.

The *nature of communication* over the Internet has changed significantly from the single-client-single-server model. Today, many such servers are hosted in large-scale *data centres*, exposing services via a data centre’s internal routing mechanisms to the wider Internet – here, the client communicates to the data centre (over the Internet) rather than the server directly, said data centre serving as a *point of presence* (PoP), enabling a service provider to host the service without having to own or operate their own resources. In recent years, those PoPs have been moved closer to end users in an attempt to reduce costs (e.g., for inter-domain transfer) as well as latency (by being closer located to the relevant users), particularly for services such as over-the-top (OTT) video or social media. This move has been driven by large-scale service providers, such as Google and Facebook, but also by *content delivery networks* (CDNs). These companies have deployed their own PoPs and, by selling excess capacity, have established themselves as large cloud players. By pushing data centres towards the network edge, communication in the Internet has significantly concentrated on the customer access networks with, for instance, an estimated 61% of Asia Pacific Internet traffic expected to being served through CDNs alone by 2021 [C4-01]. Netflix’s estimated 15% share of the Internet traffic is mostly served through localized PoPs [C4-02]. Extrapolating this to other content platforms (e.g., Amazon, Disney+, as well as country-specific platforms such as BBC iPlayer), we can project the amount of traffic originating and terminating in customer access networks to be easily around *90% of the overall generated traffic*downstream to end users. In essence, **the nature of communication has moved from servers towards services, the realization of which, in turn, moves closer to the end-user.**

When it comes to the *nature of services*, advances in software engineering broke up monolithic code blocks that served services with a single locus of consistency into smaller, independent pieces of cooperating *microservices*. Hence, the centralized client/server model has evolved into a *chains of (collaborative) transactions****,*** with typical challenges like *atomicity*, combined *resource management*, and *execution correctness* of the transactions. This, in turn, has created the desire to extend the basic DNS+IP service routing in place today by network support for such chaining, as witnessed by the ongoing Service Function Chaining (SFC) work in the IETF [C4-03]. This application-level trend goes hand-in-hand with the realization that a network cannot just limit itself to blindly forwarding packets; it needs to take an active role in, e.g., providing security (firewalls), assist in service routing (load balancing, redirecting), or traffic shaping. All this is, essentially, software that needs to operate on a stream of packets, just like many application services do. In consequence, this increasingly establishes application- and network-level services at an equal footing with utilizing the increasing *in-network processing & computation* capabilities. However, at present, a proper control framework for such in-network processing is still missing – while IETF ANIMA [C4-04] establishes a virtually separate control plane, it hides compute resources behind application functions. Some work has started, e.g., the recently established IRTF COIN (Computing In-Network) research group [C4-05] or IETF FORCES [C4-06] (separation of forwarding and control elements). Overall, **the nature of services has moved from monolithic services towards chains of collaborating microservices, at both application- and network-service level.**

Along with changes in the nature of services, the third aspect are changes in the *nature of* *service provisioning.*While microservices (networking or application-level) can be provisioned directly on bare metal, *virtualization*has opened up new opportunities. Since a long time, it has been driving the hosting model in clouds and PoPs; the evolution towards more lightweight virtualization approaches, e.g., through containers or unikernels, has increased the dynamicity of serving instances on a pool of available compute resources. Large-scale services, such as Gmail, YouTube and others, use this approach by dispatching service requests at the DC ingress to dynamically created micro-services, which in turn are based on container-based virtualization. The 5G community has realized the power of such flexibility and enabled its 5G Core specifications to use service-based architecture (SBA), which adopts the micro-service model for realizing vertical industry specific control planes over a cloud-native infrastructure, within a so-called *telco cloud*. *Service routing* becomes key here for the dispatching of service request, e.g., to establish a data traffic session quickly to the right service instance in the data centre of the mobile operator. Given proper service routing, the data centre can easily be distributed, giving mobile operators a decisive competitive advantage over conventional cloud operators in localizing services, as already observed above as a trend in the Internet. We observe that **the nature of service provisioning has changed towards virtualization, for both application services and network services.**

Many major Internet players, such as Google, have long recognized this trend and focused their attention on improving service access in the customer access network (to their POPs hosting their services). QUIC [C4-07], as an example, initially was implemented in the Chrome browser on top of UDP as a differentiator for Google services; standardization in the IETF only followed the initial deployment in millions of Chrome browsers. The intention here was clear, namely, to improve the invocation of services that support the (initially proprietary) extension, with the access network becoming even a dumber pipe and utilizing service end point instead for everything from name resolution to service invocation.

Complementing virtualization of service elements, *network* *programmability* has enabled programmatic changes of forwarding operations post-deployment. In consequence, programmability enables the functionality of all/some network elements, network functions and network services to be dynamically changed in all segments of the network infrastructure (i.e., wireless and wired access, core, edge and network cloud segments. Therefore, network programmability supports different and multiple execution environments at the forwarding plane level, those execution environments enabling the creation, composition, deployment, the actual execution and management of network services and/or network functions.

The *digitization of processes* has been proliferating in many industry branches, significantly diversifying the use cases for communication technologies beyond the often consumer-oriented focus of typical Internet services (such as social media or OTT video). Communication technologies have penetrated manufacturing, supply chains, vehicular engineering, health technologies and governmental services, among others. The Internet-of-Things (IoT) has created a vibrant industry sector with a plethora of service scenarios well beyond the consumer-oriented Internet. This has broadened the scope of services and both functional and extra-functional service requirements. The questions are a) if the existing networking model, with its one-size-fits-all approach, can support this mix of services, and b) whether custom-tailored, in-network service provisioned as in-network service chains are a superior model. These questions go well beyond the addition of a small set of QoS parameters to different data flows or the usage of network slices as isolated parts – it considers the whole set of resources and service semantics. As a trend, **new service types are realized by integrating application and network services and their provisioning, across all types of networks.**

Another key aspect is the assumed *service invocation model*. While we already discussed the transition from pure client-server to collaborative model, the ‘language’ chosen for the transactions performed in said collaborative chains also varies. Although arguments have been presented that HTTP/REST may be seen as the new waist of the Internet [C4-08], the reality of many service invocation frameworks and protocols persists. Those range from request-response models (such as in HTTP), over pub-sub models (with HTTP/2 enabling some functionality) and message passing abstractions to remote memory access models (to create the abstraction of a large yet distributed computer with shared local memory). Similarly, there is an abundance of service discovery protocols (Bonjour, UPnP, …), none of which are interoperable, and few of which are applicable outside very specific environments. We can observe from this situation that *distributed computing has not converged* onto a single universal invocation framework that can be used to connect to any other compute resource. Furthermore, each service invocation framework usually comes with its particular lower layer protocols onto which to map the service invocation itself (e.g., HTTP->TCP->IP), often leaving IP as the only common denominator. Therefore, **services choose the best means of interacting with each other, while relying on basic means to route service requests.**

A final aspect is the changing *nature of the relationships* between the entities providing these services. Currently, systems providing services are mostly assumed to be trusted (or not), and reliable (with occasional faults), but the overall trend we are witnessing is to an increasingly more complex environment, where multiple providers compete with different (albeit similar) offers, with not exactly the same levels of guarantees and trust. Hence, the overall system can only provide *trustworthy end-to-end services* by relying on high system dynamicity to adapt to variable trust relationships across the different system components. **A service environment of determined trustworthiness needs to be set up by dynamic and intelligent methods over subsystems or micro-services of variable trustworthiness.**

The key takeaway from these trends is that collaborative services in the Internet have moved on significantly since devising the key fundamentals of network forwarding that underpin the transfer of bits over the Internet.

## System Architecture Vision: Towards Smart Green Systems

With the general move towards collaborative services, the main problem is to overcome the traditional yet obsolete separation of the entire compute-and-communicate infrastructure into separate domains (logic: network vs. application; business: telcos vs. clouds; silos: automotive vertical vs. manufacturing vertical; …). Chiefly, if the original Internet was about inter-networking, i.e., bit transport between different networks, **future research must address the inter-computing**, i.e., service execution between different systems.

Like the Internet of today, the Internet of the future will be a complex planetary system made of myriads of physically interconnected elements, logically broken in separate islands, each possibly applying different security policies, routing mechanisms, access mode to application services. With more and more intelligence and computing power available per resource, resources will be configured and orchestrated dynamically (i.e., also reprogrammed in runtime), both to deploy/support new services and to better match the requirements of services running over the network. With this however, *unlike the Internet of today*, the Internet of the future will exhibit much higher dynamics, notably in its own topology, and will better support virtualization, paving a possible path for its own technology evolution, very difficult today at a global scale. With that *higher dynamics* and the *co-existence of virtual and physical entities*, the physically separate policy islands of today will often overlap in resources in the future.

A massive number of devices will be connected and will generate and exchange very large quantities of data. Useful insights will be generated based on the automatic analysis of all that data (e.g., using machine learning methods, ML). The infrastructure that supports society (IoT, cyber-physical systems) will be integrated with the Internet, which will help improve the effectiveness and efficiency of both, e.g., using adaptation through learning. *It is paramount to approach ML systemically to correctly assess the relevant trade-offs*: ML instrumentations per se require massive data transfers, are computation-intensive and, ultimately, consume massive amounts of energy. *Relying on siloed solutions and dedicated implementations limits the usefulness of ML, while it increases the cybersecurity risks (attack surface).*

These trends imply that the future network technology will have to support the Internet economy and the particular needs of the cyber-physical infrastructure alike. It will have to work with virtual objects and remote objects, the density, distribution, longevity and interconnection of which in any area can vary a lot (cmp. DC virtual machines against physical L3 routers). It will have to integrate local and remote objects and different connectivity modes seamlessly. It will have to handle its own constituting nodes and services of transient nature, which can disappear and reappear, possibly at a different location and in zero time, be multiplied and shrunk without notice, etc. At the same time, this future network will be expected to operate as a facility: it will be relied upon by private users, businesses, critical branches and governments. Therefore, it will have to be resilient to both failures and security threats, in a world, where autonomic operations for both services and infrastructures, and in particular AI/ML techniques, will be widely used. Open standards will be required, while governments will want to impose limits and regulations on the usage of all the data required to drive these new systems. In this context, overcoming the digital divide will be a key driver for technology evolution, and personal freedom and rights will need to be assured across all media.

Here, **flexible provisioning and elastic execution on dynamic and changing resource pool emerge as key challenges for the future system architecture**. Flexible provisioning refers to the generality of the infrastructure and its capability to onboard and execute essentially any ICT service. The generality of the infrastructure, as opposed to reliance on service-dedicated components, is important to increase *infrastructure sustainability* in time and *degrees of freedom for multiplexing gains*. Execution elasticity refers to an efficient adaptation during the execution, i.e., in runtime, and allows selection of best suitable links and components, to preserve the expected service properties while limiting overprovisioning. In particular, elasticity, as capability of adjusting resources used in service execution, is key to enable **truly green networking**, as it allows to redirect requests to resources with better ecological sustainability and to limit the overall resource footprint while preserving the service throughput. Given the resource mix, we have assume that elasticity and flexibility also apply to infrastructure resources. Hence, working with individual resources is limiting and not sustainable; rather, allocations and executions should refer to the resource pool as a whole. This in turn requires pervasive, resilient resource control.



Figure ‑ The Smart Green Networks Concept

**We envision a Smart Green Network as a programmable system based on a single, unifying controllability framework spanning all resources a tenant is authorized to control, including from previously separate and heterogeneous domains, e.g., enterprise and telecom networks, virtual and physical, data centres and routers, satellites and terrestrial nodes, etc. The unifying controllability framework will glue the disparate resource islands to one system of the tenant supporting smart flexible instantiation and adaptive, elastic and correct execution of any service on the resources (Figure 4‑1).**

Hence, the key challenges that the Smart Green Network controllability layer must solve are: the aspects of control over multiple general-purpose, distributed, network control operating systems; the availability of powerful abstractions from resources to services; new naming schemes for virtualised resources; dynamic and automated discovery; structurally adaptive logical interconnection; multi-criteria routing in networks of different densities; intent-based open APIs and highly configurable policies to control the resource and service access and dynamics; isolation of application’s execution environments and performances; efficient scheduling of requests to resources; a high degree of automation and support of self-\* principles (*self-driving networks*); and distributed yet trustworthy ML instrumentations.

In addition to time-proven algorithm design approaches to provable and understandable behaviour, the Smart Green Networks concept will also use existing AI/ML algorithms as well as propose new, *network-suitable, distributed AI/ML*, to implement data-driven closed control loops that can enable cognitive and comprehensible system behaviour. The training and validation of such technologies require the availability of *cross-technology and cross-sectorial datasets* that do not exist yet. The networking research community needs to build those datasets, agreeing how they are generated, accepted and accessed.

Overall, it is imperative to:

* **Allow dynamic pooling of resources from diverse participating systems, devices and objects;**
* **Integrate autonomics to enable self-organized, resilient programmability and elastic, correct service execution;**
* Offer programmable analytics and cooperative machine learning to the service layer through open interfaces.

Keys to the realization of this vision are discussed in the next sections of this document:

* programmable infrastructures composed of versatile devices and subsystems (4.7),
* integration of AI/ML at the system level (4.6),
* efficient yet correct runtime resource allocations (4.5),
* extensible and flexible data plane protocols and storage/compute solutions (4.4),
* and pervasive operational resource control (4.3).

## Virtualised Network Control for Increased Flexibility

### Programmability is Control

Future infrastructures must be extremely flexible in operations and elastic in resource usage. Programmability of resources is the only way to achieve this. However, different from configuration management, **programmability requires runtime resource control**, i.e., a way for a program executed somewhere to receive some infrastructure event and to possibly tell to a given resource what to do, both proactively and reactively, including in runtime. The requirements on any control plane are classically intrinsically linked to the requirements on the data plane. Yet, with programmability, *any* data plane becomes possible, and hence, both functional and extra-functional requirements on the control plane are enormous. For a control plane used for software-defined infrastructure operations, network structure, the available functionality, transported payloads, data rates for the latter, the latencies of exchanges, the resilience and the security are difficult to predict.

**A programmable system must provide an autonomic programmability** after deployment. There are several pragmatic reasons for that: first, setting up such a versatile and resilient control plane manually is not a skill readily available in any environment; second, this approach would be delicate, as one would need to predict future needs correctly. The main reason however is fundamental: *autonomic organization is imperative to support infrastructure dynamics, which programmability as such creates*. Any programmability solution not able to self-organize or adapt is, therefore, incomplete. Network and system control cannot rely on rigid approaches, as any such approach would only be suitable for particular environments (e.g., centralistic control, particular hierarchies, etc). Instead, *novel solutions are required capable of organizing control flows and control-related processing dynamically among all controllable system elements*, i.e., across multiple domains, systems and layers. This includes initial self-organization, self-preservation during runtime facing external and internal events and *structural adaptation*. Modern ICT infrastructures need to provide dynamic resource management to fulfil different SLAs and to achieve E2E service assurance. Rigidity in any aspect limits the degrees of freedom and, hence, limits the optimality.

With infrastructure programmability (often referred to as “network virtualization” or “network slicing”, not to be confused with the “5G slicing” concept), the decoupling of the platform delivering the service and the service elements reaches a new level. While IP networking has decoupled services from network infrastructure by putting all services on the same technological foundation (the TCP/IP suite) and by pushing the service logic to the edge, network virtualization brings additional degrees of freedom in flow processing and combines edge and network in one logical entity: it is possible to have different flow processing logics active at the same time within the same physical infrastructure, usually in the form of software elements (different configurations, different active modules) deployed on top of more generically capable hardware resources. Whereas today’s networks rely on specific flow processing machines (e.g., IP routers or Ethernet switches), whose flow processing capabilities are intrinsically linked to the purpose of the device, network virtualization breaks this barrier by allowing to define different flow treatments on the same network node and by concurrently reusing any given link for flows of different “slices” or services requiring different assurances. The same applies to the compute nodes.

### Separation of control/controllability

This immediately raises a completely new question of a *service-independent control of resources per se*: as all infrastructure capacities are, in principle, service-independent, we need a novel means to make sure that the execution of any service-specific element on an infrastructure element is durably possible. In other words, while a router routes and a switch switches, and there is hardly anything to verify about that, programmability allows to tell a node to route, while this same node was not a router before, yet had other roles and tasks. It must be verified that it routes correctly over time despite possible task overlap. Classically, control was always integrated in a particular solution logic (on the respective OSI layer or abstraction level) and directly projected to resources dedicated to realize (a part of) that solution. Previously, as existence and function of a node used to be the same, so was their control. For example, network service errors can be traced down to network element errors, by using network service control means. With programmability however, this changes drastically. **We need to understand resource control as a new, paramount domain:** since node and links generally do not have single predefined functions, **there is a new requirement to allocate, monitor, migrate and execute/run several service elements on a shared, per se service-agnostic, infrastructure**. We call it *controllability.*

Additional complexity arises from the insight that, generally, an allocated function does not translate to a single infrastructure element, but can be sustained by resource capacities distributed over the infrastructure. Due to scalability and availability requirements, most network functions rely on hugely distributed realisations, causing the allocation, extension, monitoring or migration of a network function much more challenging than the question of copying a software state from one node to another.

### Multi-Tenancy and Ownership

Network virtualization is resource sharing. Therefore, service footprints, projected to physical resources involved into the execution, are expected to overlap, constituting multi-tenancy in the overall system.

Multi-tenancy in management and control is generally hard, as it contributes to a so-called “split brain” problem: conflicts are likely to happen at the resource level, when several independent owners assign tasks to a shared resource (pool). Such conflicts can be in resource capacities (e.g., two tenants trying to book 2/3 of the resource each), or they can be of semantic nature (e.g., close port followed by open port). In control, multi-tenancy is harder to resolve, because of the potential time-criticality of the commands. **This calls for autonomic, system-integrated, runtime mechanisms for either conflict resolution or conflict avoidance, both in allocations and execution**. Candidate mechanisms per se should cater for multi-tenant operations and the expected system dependability and size. In particular, they cannot rely on single entities or centralistic approaches. This makes the design of such mechanisms generally harder and optimality as a goal questionable. Besides, while trying to provide service guarantees, such mechanisms should not sacrifice system availability and be aware of energy efficiency.

In spite of its expected pervasiveness, resource control solutions need to respect and maintain boundaries of the responsibilities, power and rights for each stakeholder in the ecosystem, as these are key for a secured, guaranteed SLA enforcement. The problem is that with network virtualization tenants can change their control scopes dynamically. Therefore, the classical notion of ownership is not well adapted to the problem space. Instead, the notion of **ownership through controllability** seems better suitable. This notion extends classical ownership through resources obtained through dynamic allocations, booking, and “leasing”. For instance, while resource limits of a virtual machine are up to the owner of the executing host, the definition of processes within the virtual machine is up to the owner of that virtual machine. Suitable control solutions should enforce this principle, also in the sense of (secure) isolation.

### Known Unknowns

To support different realizations for semantically identical entities and to hide implementation complexity, a general key challenge is to separate enforcement (the ”how” part) from the decision (the “what” part). Given multi-tenancy and dynamicity, it is necessary to investigate the ways, in which the control boundary evolves between the objective (e.g., a number of decisions at a given point in time) and its realisation (e.g., considering the operational limits of realising any decision being made, the actually available resources, etc.).

Insisting on perfect knowledge in the described environment will often be in contradiction to the operational reality. Therefore, **solutions should be prepared to work with some degree of “fuzziness”**, i.e., with incomplete data, with data of different freshness, with unreliable postulates. That is why **adaptation is more important than optimality** in this regard. Generally, decision modules need intrinsic flexibility and call for software control elements, realising an adaptive control over the resources they manage. Changes in control objectives are reflected in the existing software, which, in turn, can establish additional software elements in order to react to changes in the control objectives. The enforcement, e.g., of flow handling or computation instalment, is realised by the resource owner, possibly self-constrained by objectives imposed by the physical infrastructure and its operational environment. With all this, the overall system will nevertheless need to fulfil the service requirements.

### Self-Preservation

Given the importance of the controllability framework for the overall operations and its central position in the architecture, it is crucial to devise dependable, i.e., reliable and secure, solutions. In particular, the roles with respect to the programmability (controllability) and service operations (control) should be verifiable, and necessary protections must be applied to both control channels and control end-points, acknowledging decentralization, multi-tenancy and known unknowns, i.e., also dynamics in the overall span of the control plane and dynamics in the available infrastructure resources.

A running control framework must be able to adapt to such changes, e.g., include and remove resources, adjust its own resource usage yet still protect its own integrity. Besides, the execution of its constituting parts in possible remote, virtual objects on devices physically owned by other tenants calls for either trustworthiness verifications of such executing devices or for systemic approaches to mitigate dependency on any particular component.

The self-preservation solution must also counter so-called *self-inflicted errors* inherent to programmability: a running “program” of a tenant could have negative impact on the resource control framework per se. For instance, it could overload crucial control elements (e.g., putting controller under high load leading to timeouts), influence control transport channels (redirecting traffic) or the control plane structure (e.g., blocking control plane traffic to and from nodes and disconnecting controlees from controllers, etc). **Establishing system integrity and self-preservation in runtime for a distributed, dynamic resource control sub-system is one of the research challenges**.

### Research Challenges

Challenges on resource control in Programmable Infrastructures include:

* Resource control emerges as an initial glue that first allows operators to programme their infrastructures, i.e., as an initial new service that allows to allocate, monitor, execute and remove service elements on/from sets of nodes and links. To avoid vendor lock-in and to allow truly end-to-end slicing, it is exactly this glue that requires standardisation, and not any domain-specific management interface.
* Resource control must be able to reach out to all resources controllable by a tenant and be capable to check the states and operations of all service- or slice-specific elements on those resources. Besides, the realisation of the resource control itself should follow the insights from above, i.e., it must be distributed over all controllable nodes and must support elasticity of itself (reaching out to new elements, adaptability, including in structure, self-preservation, conflict awareness).

Because of the novel degree of decoupling of service elements from the infrastructure, the central problem of programmability is not to make a blueprint, but to be able to execute any requested blueprint on top of a shared, distributed infrastructure composed of different capacities, occupied by loads from other executed services or slices. Such a distributed guaranteed execution under contention and with concurrency is extremely challenging and, currently, can only be solved on very small scales.

## Re-Thinking the Data & Forwarding Planes

### Design Considerations for an Evolved Data & Forwarding Plane

The original design of the IP-centric data plane of the Internet focused on three key fundamentals (i.e., principles – key design choices), namely ensuring *global reachability* through a *robust* packet forwarding mechanism that would provide a *best effort* service to higher layers [C4-09]. Those higher layers would complement the basic mechanisms through aspects of, e.g., reliability, error control, but also support for specific service invocation models.

From the discussion in Section 4.1, we derive a number of design considerations for data plane solutions that would ensure a continued support for the evolved services and interactions we have been seeing in the Internet, depicted in Figure 4‑2.

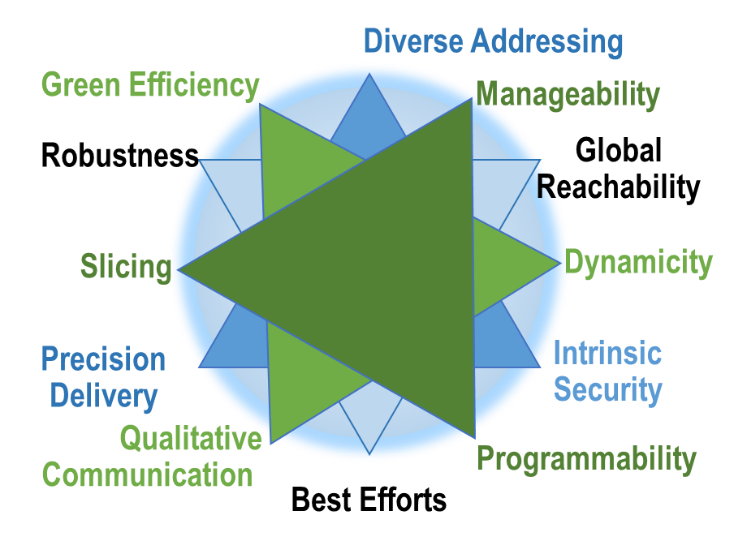
****

Figure ‑ Dataplane Evolution - Design Considerations

We exclude from our considerations the approach to *deployment of the solutions*, therefore not specifically addressing the possible *evolutionary* vs *clean slate* nature of re-thinking the data & forwarding plane in order not to constrain the research albeit pointing out that the *feasibility of solutions* will ultimately need to consider the evolutionary nature of any deployment in existing infrastructures.

It is important to note that evolved data & forwarding plane solutions do not need to necessarily address all considerations and we can already see examples for proposed solutions [C4-10][C4-11][C4-12][C4-13][C4-14][C4-15] considering certain aspects described here:

1. **Dynamicity:** As observed in Section 4.1, many relationships are bound to become ever shorter lived, driven by *virtualization* approaches, with the possibility of network resources to appear and disappear frequently. This introduces aspects of *dynamicity* into the relations that significantly depart from the long-lived locator concept that underpins IP, which assumed a long-lived relation between a client and a *portal* of information in the Internet. Instead, the assignment of forwarding relationships must align with the ability of the corresponding SW component to change relationships, or else the data plane will only inadequately support the advances we see in complex SW systems utilizing the Internet, e.g., through container-based micro-services.
2. **Green efficiency:** While we recognize that many of our considerations can be and partially have been realized through a myriad of *add-ons*, *extensions* to and *overlays* on top of Internet protocols, we strongly believe that *green efficiency* is a consideration that must be added to the design for an evolved data plane, even to the point where the selection of suitable mechanisms ought to include an *energy efficiency KPI*at the same level of today’s focus on performance KPIs such as throughput or delay. Overprovisioning and the aforementioned overlaying of solutions to improve on otherwise limited designs have played too long a role in communication networks for it to continue in the light of the increasing policy trends to fight against climate change, such as Europe’s Green Deal [C4-16]. While providing a flexibility in change (through yet another overlay), it has also led to complexity in management and the *inefficiencies* caused through indirections over many shim layers that make up the final communication relation. This not only stands in the way of achieving true high throughput and low latency communication, required by many emerging services, but also drives the ratio of ICT in the energy consumption [C4-17].
3. **Qualitative Communication:** Relationships will not only become more dynamic in nature but also more complex in terms of *inter-dependencies*. The current model in the Internet treats relationships at the application or session layer, realized through independent connections, managed through protocols like TCP and others, with separate resource management schemes. This leads to inefficiencies in cases where one sub-relationship is transferred well compared to the other, spending efforts on, e.g., error control, for a sub-relationship that is reduced in value due to reduced performance of another sub-relationship. The result is often overall loss of end user experience that ultimately decreases the value of the communication. This qualitative communication is crucial to be taken into account when designing data plane solutions in order to be able to optimize the use of resources spent on the overall relationship rather than the sub-parts of it. Leaving this handling purely to the application or session layer leads to inefficiencies of resource usage, which can be avoided through application awareness, e.g., additional in-packet metadata at a lower part of the data plane, expanding on existing concepts such as service function chaining (SFC) [C4-03] albeit for parallel not sequential transactions.
4. **Security** plays an important part in data plane mechanisms and the current Internet has well recognized this with security considerations having become essential in every protocol solution standardized, for instance, in the IETF. However, the fundamental of building *security on top of an otherwise unsecured packet forwarding* has not changed, therefore focussing efforts on end-to-end security of the application-level content, but not the *security nor the privacy of the packet forwarding operation* itself (who is talking to whom, compared to what is talked about). Consequently, this has enabled for long mechanisms such as IP geo-tracing as well as enabling spoofing and therefore denial of service attacks. Mitigating methods deployed are add-ons to the otherwise unsecured IP, require extra effort rather than basing themselves on an *intrinsically secure* design per se where security of end points and networks alike is ensured together with the *privacy of the interaction* between communicating end points, striking the right balance between accountability and anonymity. Decoupling “security appliances” from the analysis of events and policy-based decision-making is another aspect to consider. Tiny security-handling functionalities embedded into virtual entities should monitor events, collect information and transfer it to suitable functions (possibly based on AI/ML) capable of more powerful analysis and anomaly detection, which in turn would enforce policy based-decisions back to the local actuators.
5. **Precision delivery:** the best effort nature of the current IP suite does not suffice for a number of the new emerging services, e.g., for Beyond 5G. Therefore, it will need to be extended in order to capture new demands for specific performance characteristics, such as *strict delay and latency bounds* for system control, human interaction and many other services as well as *on-time bounds*. This requires the control loops involved to ensure the specified performance requirements of various applications, particularly for access networks with widely varying performance characteristics such as wireless. Those control loops will also need to enable trading off latency at the control level against the necessary operations at the data plane.
6. **Diverse Addressing:** While the universality of higher layer service concepts over a single addressing scheme has been praised as key for the Internet protocol, we assert that the support for *diverse addressing* will need to replace this aspect of the current Internet in order to improve on efficiency when supporting the many new services, while still *ensuring the global reachability* that the current Internet has achieved. This could lead to solutions for *optimized Internet-of-Things* communication (with *smaller identifiers* being used for efficiency purposes), while preserving inter-domain access to the IoT resources. As another example, instead of relying on an interaction between DNS and IP routing, adding initial latency to the service exchange (and leading to problems in future service invocation if service relations might dynamically change), research in, e.g., routing on labels [C4-18], information-centric networking [C4-19] and solutions on *semantic addressing* [C4-20] have shown that those latencies can be significantly reduced through name-based addressing, pushing name information to the far edge of the network as a trade-off (which can be accommodated through increasing availability of storage, even in mobile devices), while still scaling to significant network sizes, particularly in the recognition that much Internet traffic is being localized, as discussed in Section 4.1. In addition, changes in named relations become merely an *ingress routing decision*, being removed as a burden from the DNS, for instance, therefore significantly *increasing flexibility* in routing when the service instance serving a named relation is changing in the light of virtualization of service endpoints, as discussed in Section 4.1.

The aforementioned considerations for *designing suitable packet delivery solutions* need to furthermore consider the following aspects when *being realized for and deployed* in the emerging communication infrastructure:

1. **Manageability:** All the above characteristics will require suitable instrumentation to monitor and validate the delivery of promised assurance levels. Furthermore, telemetry capabilities, i.e., the process of measuring, correlating and distributing network information, are required (and will need to be enabled at the data and forwarding plane level) to gain the visibility of network behaviour to improve operational performance over conventional network Operations, Administration, and Management (OAM) techniques to enable full network automation.
2. **Programmability:** Operators will need to be provided with the methods to dynamically govern the forwarding plane in order to rapidly and easily introduce new network services or to adapt to new enhanced and modified contexts, e.g., through insertion of programmable metadata into packet headers traversing the network. Such programmability particularly aims at providing the desired overall green efficiency by moving from HW updates to SW upgrades instead, including executable code to be injected into the execution environments of network elements in order to create the new functionality at run time with the required security characteristics.
3. **Slicing:** Resource management needs to be utilized to enable easy and efficient execution of multiple and different types of delivery mechanisms, possibly each with different guarantees for KPIs/QoS/ stringent non-functional requirements of network services at a given time on the same infrastructure but across separated subset of resources in the shared resource pool for realization of the desired functionality. Such “slices” may offer uniform capability interfaces to entities and network functions, abstracting the autonomous slice components, which may be loosely coupled, with different functional and non‐functional behaviour. A challenge to address is the realization of large-scale and multi-domain data plane deployments in sliced environments, including aspects of identifying the participating resources being used.

### Key Research Questions

The following research questions are not purely limited to the data and forwarding planes but address wider holistic systems aspects, leading to the following research challenges:

1. **Which layering in which part of the network?** To cater to the often starkly different ‘scopes’ of communication, ranging from localized sensor communication over POP-based access to OTT services to truly global communication, the question on layering is crucial in the light of an *efficient/green* implementation of the overall system. With the desire to support *diverse addressing*of the data plane, the question needs consideration as to *what layer best realizes the semantically different forwarding operation(s) most efficiently, taking into account not only the individual service itself but also the overall system efficiency from the perspective of resources that provide that service.*
2. **What is the role of soft architecting?** With the proliferation of software-centric approaches to networking, allowing for a much higher degree of post-production as well as post-deployment programmability (cmp. Section 4.2), the question arises *what the deployed architecture really is* or *if everything manifests its own (soft) architecture?* Assuming such soft-architecting, as discussed in Section 4.3, the desire to agree on a common substrate, on top of which all such (soft) architectures reside, still remains, similar to the origin of the Internet protocol albeit with a possibly different answer. Instead of the commonality being that of a common postal system between locations, such *commonality* could be the interconnecting bus-like system between resources, where resource control becomes fundamental, while global transport and global routing degrade to applications, many of which can run in parallel. Any answer to that common substrate, however, should still provide the right set of fundamentals among those outlined in Section 4.4.1 that align with the services at hand. *In other words, while soft-architecting is a promising evolution path, ultimately, the considerations above need to be applied to and solved by the global “glue” at the resource layer, be it a control bus or the delivery system itself.* A possible advantage of a solution based on a resource control is a clear set of and a better understanding of the requirements of the latter.
3. **What are the tussle boundaries of the overall system?** Tussles [C4-21] are caused by interactions of players as defined through the interfaces of the overall systems, with each player often pursuing their individual interest. Understanding the boundaries of tussles, the mechanisms to express them and those to resolve them, is crucial for the overall working of the system. Much has been done to study the tussles of the Internet (and its main players) but postulating a system of high *dynamicity* also postulates one of changing relations, particularly when it comes to *trusted* relations. Enforcement through trusted third party is often a mechanism that will not do in such often ad-hoc relationships and *solutions will need to realize more suitable, equally dynamic and ad-hoc mechanisms to ensure an otherwise trustworthy execution of the overall system, while also preserving the privacy and ensuring the security of the individual participants.*
4. **What data is required to make the data plane work (well)?** Any data plane solution, including existing ones, works on a set of metadata, such as identifiers, as well as state, such as link data. While much of this data is vital for the basic operations realized in the data plane itself, it is also required for *control plane* decisions (e.g., for load-depending resource allocations across the network) and for realizing *management goals* (e.g., matching long term demand to supply information). With this in mind, data plane solutions must not focus solely on hitting the key fundamentals outlined in Section 4.4.1 but also enabling a fruitful interaction with the corresponding parts of the overall system that ensure the working beyond the pure transport of relationship information.

### Recommendations for Actions

The following list are suggestions for important actions towards realizing the research agenda for DP/FP evolution, not claiming to be exhaustive:

1. *Call for internationalized efforts*: given the challenge to evolve the data /forwarding planes, European efforts should liaise or even directly collaborate in internationalized research efforts, i.e., in the creation of solutions not just the exploitation in standards or OS communities. This could be realized through targeted **international calls** (e.g., EU-China, EU-US, …) on data/forwarding planes technologies as well as through the creation of **international expert groups**, e.g., in coordination and support actions.
2. *Call for experimentation*: although strong theoretical foundation is desired for any change of fundamental data/forwarding planes functionality, strong **experimental evidence** and **large-scale open testbeds** are crucial to show feasibility but also foster adoption through the operational community. This could be realized through an evolution of the original FIRE efforts or a similar trial phase as in 5GPPP. Open experimentation data/forwarding facilities are required for a large number of **third-party experimenters** of promising solutions and possibilities for looking, e.g., beyond 5G - an Internet of experiments (IoE).
3. *Call for data/forwarding planes research repository*: in order to foster the adoption of evolved data plane technologies, experimentation (see item 2) will need to ensure **replicability** in other, possibly pre-commercial or otherwise research, settings. This could be ensured through making evidence **data and code base availability** mandatory for certain aspects of data plane research (e.g., for certain TRLs upwards), including **migration solutions** that will allow legacy IP-based applications and IP-Services to be used with the new enabled forwarding plane capabilities.
4. *Call for clean slate research*: following the argumentation in other efforts, such as FP7 FI, NSF FIND, the evolution of core Internet technologies requires a combination of an **evolutionary and revolutionary** approach. This could be achieved through setting aside specific **clean slate** or greenfield funds for testing more revolutionary approaches to the data plane evolution.
5. *Call for funding data / forwarding planes research* *in solutions* along the considerations discussed in Section 4.4.1, such as those providing precision delivery in extension to existing best effort. Examples for such research aspects are
   1. Precision packet delivery (with QoS) to extend/complement best effort delivery;
   2. Intrinsically secure, i.e., authenticated and accountable, packet delivery;
   3. Semantic routing, extending current endpoint-based routing for lower latency and higher flexibility delivery of service requests;
   4. Deployment on tenant-specific (in-)network service functions;
   5. Inter-connection of compute/storage resources at Layer2, with focus on customer access networks while interconnecting to Internet-based clouds;
   6. Programmability of forwarding under control triggered by management.

## Efficiency and Resource Management

Efficiency in terms of managing the resource pool of a communication system is essential for controlling costs and therefore OPEX in offering communication services. With Total Cost of Ownership (TCO) becoming a major design target, e.g., for 5G, and the push for *sustainability* of telecommunication infrastructures, the role of efficient resource management will increase significantly in future deployments. This translates to several new problem spaces, currently unaddressed, underestimated or completely overlooked in both the industry and academia.

### Network Slicing versus Network Capacity Planning

As network slicing promises a sheer endless customisation of network-spread functionality, it becomes difficult to plan the capacity of network infrastructures in the same way as today. Whereas operators currently use their combined empirical knowledge regarding both infrastructure and the expected service (and its prices), network slicing turns this principle upside-down: while the infrastructure operator remains neutral to the service, the slice owner is expected to translate the *service to capacity requirements* onto the infrastructure capabilities, an exercise that lacks a reliable general methodology. Incapable of correctly translating service to capacity requirements, slice owners are likely to engage in a cloud-like operation model: start small, then expand or reduce contracts as you go. The *elasticity of the slice therefore is a central requirement*. This fact together with the required radical reduction of the service creation time (from 90 days to 90 minutes, as, e.g., per 5GPPP KPIs) underlines the upcoming shift from planning of the infrastructure to continuous (and likely dynamically adapting) runtime operations on the latter. In simple terms, network planning and network slicing are misaligned, as the former, driven by the presumed physical deployment, operates within completely different time frames than the latter, which exhibits on-demand elasticity.

Hence, **while the initial planning provides the larger operational bounds within which slicing can operate, it is the runtime (continuous, real-time, hot) management and control that determines the efficiency and therefore the costs of the sliced service**. If network slicing wants to succeed in the above sense, the employed technologies must embrace this change and provide mechanisms and practices that feed runtime control over a longer timeframe back into the planning and investment cycle for network infrastructure.

Independently of scale, slicing renders the infrastructure usage and occupation much more diverse and more dynamic. This emphasises the requirement for continuous operation of the real-time management or control, while infrastructure control and management are required to handle the dynamics in a new, currently unsupported manner. This includes handling node and service element loads, departures, additions, errors and the like.

Runtime management and control ultimately still drives the longer-term planning that we can see today in networks. Following our cloud analogy, the longer-term demand and supply pattern emerging from the many tenants of a data centre still drives the planning, and therefore investment patterns, for sufficient build-out of the cloud. Similar feedback must exist for slicing-based network infrastructure albeit situated in a many point-of-presence nature of resources, utilised over a possibly huge area of requirements on those resources.

### Slicing Requires Conflict Resolution

To better support multi-tenancy and to allow efficient resource sharing, especially at bigger scales or facing known unknowns, **consistency and concurrency of slice allocations and slice executions should be addressed** at the systemic scale in runtime. Indeed, concurrent resource-competing or semantically contradictory requests at either allocation time or during (elastic) execution must be dealt with to avoid partial slice operation, generally being useless and, hence, waste of resources, while requiring **novel mechanisms for networked garbage collection** to free up any erroneous resource allocation during such conflict resolution.

While mechanisms exist for handling concurrency at individual component/node level, guaranteed slice allocations would require novel, system-wide mechanisms. Herein, fundamental systemic limits are to be properly addressed at large scale, since strong consistency of allocations (e.g., through consensus, atomic commit protocols with locking, etc.) might otherwise lead to a decrease of availability (starvation effects) and therefore reduce the supported dynamics in slice allocation and elasticity.

Inspired by distributed database management systems and distributed Internet services, **novel research should consider multi-level guarantees for services and service-level redundancy**. In spite of the similarity, the central insight here is the difference in the definition of consistency for databases and systemic allocations: while databases treat replica of the same object (which makes concurrent writes to replica R1 and R2 problematic), systems work with redundant, independent objects (e.g., concurrent allocations on two equivalent yet different paths are non-problematic). Given the observed increase in systemic redundancy (e.g., network density, trend to regional data centres), this insight promises better scalability of guaranteed allocations without sacrificing availability. Hence, **novel approaches could explore the suitability of concurrency-preserving schedules** (e.g., with commitment ordering) **for programmable networked IT systems.**

### Elasticity: Slicing Efficiency Requires Runtime Scheduling

When addressing efficiency, *Total Cost of Ownership* KPI and *green ICT*become important aspects to consider. Given a slice blueprint, one must find suitable resources in the infrastructure and make a reasonable long-term allocation of the blueprint on the selected resources (as per slice lifecycle). This topic has received a considerable attention and is often referred to as *”virtual network embedding”,* with both simplified greedy solutions and optimised heuristics (with tuneable sub-optimality bounds) being available. However, the overall resource allocation problem of network slicing is twofold, and the second part is unsolved, relating to the question of *elasticity of slices*. Indeed, to achieve slice properties not readily provided in the serving infrastructure (e.g., elasticity, but also availability, resilience, latency guarantees, etc.), slice embedding will be usually broader than the purely functional requirements of the blueprint. Therefore, for every entering flow, a simplified, yet more dynamic and online question of the resource allocation problem will arise: **which of the suitable function-equivalent infrastructure resources should be involved into the treatment of an incoming flow**? Note that this cannot be solved within the slice, if the infrastructure owner promises (and sells) extra-functional properties of the allocated slice; in other words, such provisioning will be done in the infrastructure, transparently to the slice owner.

The answer to this question of **runtime service scheduling** is paramount to address the TCO KPI, as solutions to this problem would allow to overprovision slices, without the need to overprovision the underlying infrastructure. The runtime service scheduling therefore is the answer to the questions of elastic and dynamic allocations, currently unsolved. Moreover, if an efficient solution to this problem can be found, network slices can and, for efficiency reasons, should be implemented as dynamically scheduled entities rather than exclusive reserved (and therefore possibly wasted) resource pools for tenant.

### Towards Green ICT

In recent years, the *ecological conscience* has generally increased in Europe. Backed by political and economic initiatives both by the Commission (e.g., Renewable Energy Directive, Green Deal) and the Member States (e.g., German *Energiewende*), the main trend is *to reduce the dependency on conventional energy sources* (nuclear, fossil) *to the advantage of renewable energy supply*(wind, photovoltaics, hydroelectricity). Given the decreased flexibility in the energy production of the latter, *this shift must be accompanied by smart energy demand management functions*, resulting in a strong push for Smart Grids in the energy sector. That is where ICT is generally regarded as an important enabler (e.g., using 5G MTC and network slicing). However, swapping power sources does not address the power consumption of the consuming infrastructure as such.

Given the increased reliance of the society on ICT infrastructures, these have emerged as essential consumers. For instance, while 5G is 10 times more energy efficient than 4G in transmission, recent studies suggest that, by 2025, 5G alone can increase the anyhow growing energy demand in the data centres by up to 3,8 terawatt hours (TWh) in addition [C4-22]. Even though this effect is due to the increased “popularity” and not to a shortcoming of 5G per se, undeniably, **energy efficiency of data centres emerges as a central preoccupation for resource management**. While overprovisioning is a simple and popular method in networking (e.g., in fibre optics, it is a simple mechanism for both network development and service quality increase), overprovisioning is not a valid approach for the computing domain. Indeed, modern DCs reduce the required compute power for the same load, using DC-internal schedulers (e.g., Apache Mesos, Kubernetes K8).

Novel methods are required to overcome the limitation to a single DC and should embrace path and compute allocations together, in order to exploit infrastructure diversity. **Future research should explore and develop approaches to elastic resource management** in addition to the current trends limited to green energy power supply for data centres (using smart grid’s demand management) and the “recycling” of waste heat from the DC cooling systems. Such novel approaches could generally rely on elasticity mechanisms, i.e., runtime redirection of incoming service requests to best suitable infrastructure components with the goal of increasing the throughput on the same resource footprint. Preferred redirection to eco-powered components can be integrated into runtime service scheduling.

This theme translates to the overall ICT sector and ICT infrastructures in that **green or sustainable ICT cannot be achieved without a profound consideration for resource management**. Given the steady increase in the dynamics and the diversity of services, pre-planning and fixed allocations of any kind (dedicated devices, pre-provisioning, long-term configurations, mapping to particular nodes, single points of failure) are doomed to overprovisioning, which, for the same service load, requires more resources to be deployed, maintained and powered up in the infrastructure. This wastes energy and is ultimately not sustainable.

### Research Challenges

On the opportunity side, **programmable ICT infrastructures increase the degrees of freedom** in service-to-infrastructure mapping and, therefore, **could yield more sustainability both in time (flexibility) and in energy (elasticity)**. On the challenges side however, the mixed compute/ storage/ networking environments, even under the assumption of pervasive controllability, require suitable solutions with respect to resource management: the heterogeneity of resources makes it harder to rely on single mechanisms, as different domains apply their own approaches internally, and often do not exhibit this knowledge externally. Also, a given unique approach will likely not fit the requirements of different resource types. Besides, the scale of the overall infrastructure makes it hard to rely on any consistent, up-to-date picture of the current consumption vs. load, as described above.

Challenges in this area can be summarised in the following:

* The question of runtime service scheduling in programmable ICT systems is paramount, as it permits both to provide superior extra functional properties of the supported allocations (“slices”) and to lower the Total Cost of Ownership. Indeed, the TCO of a slicing implementation using only fixed-quota assignments (meaning that the sum of the resources consumed by all slice instances will define the necessary infrastructure resource footprint) would be horrible, roughly comparable to hardware slicing. **The dynamic resource assignment problem,** as a quest for a more efficient infrastructure sharing, including computing, networking and energy resources, **is difficult because of heterogeneity, partial or outdated information, its runtime nature and the absence of any central party or mechanism** (like ordering or synchronized clocks).
* The answer to the job scheduling in large networked systems requires a lot of fundamental research, such as leveraging existing solutions from data centre research and applying them at network scale with multitenancy and concurrency. Suitable **conflict handling mechanisms are required** here, especially if guaranteed execution is required. Utilizing insights from distributed systems research, **the major goal should not be optimality, but rather improved efficiency**: given the size of the infrastructure, *1 % efficiency increase might translate to hundreds of millions of Euros/Watts/additional users*/etc. Given the assumption of sub-optimality, novel **mechanisms for networked garbage collection can be considered**.
* The elasticity of slicing has to propagate towards subscriber level and even application level. For instance, an application could use different slices during its session in order to best utilise the network as well as to provide superior quality of service with respect to slice offerings. In the view similar to application-driven networking, an application could also explicitly ask for a “slice” suitable to its needs. This rules out any pre-provisioning and can only be reasonably implemented in public infrastructures like the telecommunication networks, if the provision of the slices is highly dynamic yet resilient. Thus, **application requirements need not only signalling but also suitable translation to constraints, under which the slicing control can operate to meet the applications’ needs**.

## AI/ML-based System Evolution

Utilizing knowledge gained over a longer time is well-established in the industry. OTT services have long been using AI/ML techniques, albeit operating largely on data sets derived from the services and their users directly. At the level of improving network operations, self-\* solutions have advocated the use of operational insights to adapt network functionality without intervention from either human operators or users.

Given the vast amount of data available in complex network environments albeit in a distributed fashion, AI/ML is well suited to produce new insights into emerging behaviour patterns in such distributed environments. To this end, suitable AI/ML techniques are applied, provided as a service capability towards (a) *operations of networks* and (b) *improvements of service provisioning and functionality* itself. In other words, we see a strong evolution of future networks from a mere communication and computing infrastructure to an integral part of the overall knowledge pool that can be used to improve functioning of networks and services alike; *AI-as-a-Service (AIaaS)* provides this capability in a prosumer-centric notion.

### Proliferation of AIaaS in Network Operations

We foresee AI/ML playing an increasingly important role in network management***,*** with the aim of reducing costs, increasing productivity, deriving more value, and improving customer experience. A range of learning techniques can be used to predict the behaviour of the network and its users to better provision resourcesby avoiding today’s typical over-dimensioning. In terms of OPEX optimization, where energy consumption is one of the major cost items for network operators, AI/ML will leverage ***“****data lakes****”*** to analyze performance and optimize energy consumption versus quality of service. We furthermore see a strong alignment with the move towards fully virtualized network functions, where AI/ML capabilities are utilized to ensure reliable controllability in a fully automated manner, specifically to:

* Instantiate a complete end-to-end network that includes, e.g., the RAN, mobile core, other forms of access networks (DSL, etc.), transport network, as well as the Data Network. This network may be logically separate and/or isolated for certain aspects like services, users, etc.
* Extend such an instantiated network by computationally and storage resources where suitable, spanning from edge computing to backoffice data centres.
* Incrementally deploy network services in the operator's network (or elsewhere) in logically separated and/or isolated manner from the other already deployed services.
* Deploy and provide network services to other operators and/or service providers when requested, via open interfaces. This way, other operators and/or service providers can re-sell/extend the provided network services.
* Realize fast lifecycle management (LCM), automatically triggered based on vendor-independent FCAPS management.
* Instantiate new components into a live production network in a plug-and-play manner.
* Terminate one or more network slices or service(s).

AI/ML-based network control – as a way to implement fully automated Smart Networks – seem like a must for future networks rather than a nice-to-have. To wit, the scale of deployments made possible by function virtualisation, the extreme split in micro or atomic functions and the proliferation of more and more functions at the edge create network deployments of unprecedented complexity, challenging to manage and control with current decision support tools. Down the road, we see a need to overcome the current juxtaposition of conventional *model-based* approaches (who have, after all, driven the Internet for decades) with still untested but promising *data-driven* approaches and come up with integrated, hybrid solutions. Possibly, data-drivenness could compensate for fuzziness and uncertainty while model-driven approaches could provide a solid operational foundation.

The system challenge here is to develop a future network with *Full Automation*, which reduces and tries to eliminate any human intervention. In principle, such automation can be achieved, once exact behaviour of all components is understood and expressed in a suitable model. In practice, however, for the highly complex and interwoven system outlined here, such a full-model description is not feasible, rendering *model-driven* automation and control impractical. For such situations, data-driven approaches leveraging powerful AI/ML systems might come to the rescue. One challenge here is to determine which data to use for what control aspect, using which AI algorithm. For example, there is a challenge that AI/ML is seamlessly applied to network control, to run automated operations of network functions, network slices, transport networks, in an end-to-end scope.

Moreover, a thoroughly integrated AI scheme would open up new venues, how to think about operating a network in general. For example, suppose good to very good predictions (load, failures) were available. Then, the possibility arises to implement predictive behaviours in the network, to make available a network control intelligence capable of mitigating failures, the usage load, etc. and quickly adapt network configuration to be always available at the target performance levels requested by the applications. Basically, we could switch from closed-loop control to open-loop control (or, at least, to receding horizon control – also sometimes termed “open-loop feedback control” – where the optimizing control strategy is recomputed in the light of new observations leading to new predictions over a forward shifting time horizon).

### AIaaS Proliferation in Service Provisioning

Beyond the use of AI/ML for improving on network operations directly, AI/ML will enable innovative features when provisioningfuture digital servicesfor homes, businesses, government, transport, manufacturing smart cities and other verticals. At the same time, we expect a significant increase in the amount of machine-to-machine (sensor) communication monitoring smart cities, Industry 4.0, smart energy, etc. These changed traffic patterns will drivethe move of computational and memory/storage resources from huge data centres towards the edge of the network, therefore impacting network designs to support this move. New services powered by AI/MLmay also bring significant socio-economic impacts together with improved sustainability models for Network Operators. For prosumers, we foresee the proliferation of *personal data platforms* that are tightly connected with network services, allowing Internet users to control their data.

Key to reaping these benefits lies in utilizing the knowledge derived from the vast pool of network datain the services provided over the future telecommunication infrastructure but also utilizing the *highly distributed processing capability* that an AIaaS offering would provide. It is crucial to also understand the impact of M2M traffic, generating e.g., smart city data, etc., which will shape system designs. Both aspects drive the *provisioning of data into the system*as well as *complementing processing capability* of the network with service-level ones. With this, we see AIaaS capabilities of the infrastructure merge with those capabilities at the data and processing level that vertical customers will bring to the table. As a consequence, we see an *emerging data marketplace* that goes beyond raw data (such as location traces) but is lifted to knowledge and insights provided by network operators to its service provider customers. For instance, radio measurements at the deep level of small-cell base stations can provide insights on physical objects that in turn can be utilized by service providers for consumer-facing services that would have otherwise required dedicated hardware deployments or other means of realization. However, key to making an AIaaS useful for service providers, clear and open interfaces, both for data provisioning but also the reasoning logic, are required. Furthermore, *control over the distribution of and access rights* to both data and processing is crucial for the alignment with privacy regulations that both network operators and service providers will need to adhere to.

### Related Research Challenges

While such AI/ML-driven or self-driving networking can start using existing AI and ML protocols, algorithms and approaches, it will gradually require network-specific adaptations in several regards. Below are some of the challenges we can identify in pursuing an AIaaS vision:

* One aspect is the **availability of network-typical and network-characteristic** **datasets** for training and validation. There is no commonly agreed reference dataset to use in research or development to compare different approaches against each other, nor is there a good understanding which data is actually needed to drive an AI/ML scheme, which features need to be extracted from an operational network.
* Similarly, current experience shows that the **procedures to train and validate** AI/ML algorithms and the architectures they use are mostly focused on static pattern recognition (e.g., images, sounds, diagnostics of fixed analysis data…) and are therefore not well adapted to the nature of dynamic networks. We need schemes suitable for changing environments, changing number of users, changing topology, etc. – properties not typically found in popular ML algorithms.
* Even with suitable datasets and algorithms in place, **there is the need to extend the currently mostly centralized AI/ML algorithms to be distributed** to accommodate the distributed deployment in (often multi-domain and multi-technology) networks. This, in turn, will introduce challenges to ensure *scalability, consistency, consensus* and *convergence* of both data as well as reasoning in such distributed environment, while privacy regulations will mandate auditable solutions. Complementing this need for supporting the distributed realization of AIaaS is the opportunity provided by the move towards *Edge and Fog Computing*that we can already see in 5G. This opens the opportunity to complement the resources of cloud computing data centres to analyse the expected vast amount of network data; it could even do so while better adhering to privacy demands through *localizing the processing* of raw data.
* In such a scenario, there are trade-offs between data volume to be transported vs. localized or distributed energy consumption and computational capacity; latency for training vs. latency for action; questions about ensemble learning when locally learned insights should be merged and generalized. For both learning and control in ML, we need a *meta-control*that allows fordeciding, which data is fed into a learning scheme, where and which learned models are distributed to which place in the network for taking control decisions. This is similar to provisioning micro-services in general. However, it might have quite different data-rate/computational/latency/resiliency requirements compared to an application-level microservice. In other words, *AIaaS will need its own control plane logic* built upon the control plane capabilities of the infrastructure itself.
* Meta-control immediately raises the question of self-application: **can ML be used to decide on ML**? This idea is currently gaining ground in the Auto-ML community, where ML is used to learn hyperparameters of ML. Here, we need ML to learn, how to apply ML to a network. Clearly, there is considerable risk of oscillations, feedback loops, etc.
* The **scope of AI/ML schemes will also need to be investigated.** One possible, perhaps naïve approach is to have one set of AI functions/data sets that is applied only to a segregated, intra-service based scheme (“*sliced AI*”), which is easy to realize and ensures data privacy, but squanders possible optimization potential. Removing redundancy and going to a cross-service, cross-network, integrated AI/ML (“*integrated AI*”) scheme is promising, yet fraught with complex design choices.
* Given the increasing multi-domain and multi-technology deployment of infrastructure, AIaaS will require the capability for *multi-domain orchestration* of distributed processing, meaning end-to-end interoperability is a must (cmp. Sections 4.2, 4.3). This requires greater standardization efforts and further progress in the functional architectures.
* Furthermore, **aspects related to security** beyond the conventional application of AI as a tool, e.g., ensuring data flow provenance and distribution within the system, and dealing with AI-enhanced (-amplified or even -rooted) attacks are essential.

### Recommendations for Future Actions

Based on the challenges above, we recommend research into the following aspects:

* making **available network-characteristic datasets** for training and validation;
* agreed procedures to **train and validate** the AI/ML algorithms;
* **distribution of AI/ML algorithms** instead of using centralized AI/ML algorithms, in order to apply AI/ML to a network, considering placement and distribution of AI/ML functions within a network;
* **meta-control procedures** applied for learning and control in AI/ML to decide which data is fed into a learning scheme;
* **integration with AI/ML features provided at the edge of the network**, e.g., when provisioning future digital services for homes, businesses, government, transport, manufacturing smart cities and other verticals;
* devise architectures, approaches and algorithms for **sliced vs. integrated AIaaS*;***
* **development of use cases** for new services powered by AI/ML at the network and service provider level;
* **development of network management techniques** embracing the AI/ML predictions;
* **support performance analysis and optimization** methods for energy consumption versus quality of service analysis, e.g., through an AI/ML enabled “data lake” approach;
* support for new AIaaS services and applications that require, e.g., **multi-domain orchestration** of distributed processing and end-to-end interoperability;
* address of **security and privacy challenges** with evolving regulation;
* support the **provisioning of data** required for AI/ML learning phases, particularly from network infrastructure functions;
* address the **scaling requirements**, e.g., through partitioning mechanisms, to enable efficient AI/ML data processing to provide timely responses required by AIaaS solutions.

## Deep Edge, Terminal and IoT Device Integration

Architecturally, the ‘deep edge’ with its IoT as well as end user or vertical industry devices well integrates into the vision of Section 4.2 by becoming part of the common resource pool, provided as a non-decomposable set of resources by some edge entity, such as an end user, industrial site owner, or a building owner. Following the ‘ownership through control’ mantra, described in Section 4.3.1, we therefore envision tenant-specific resource usage to expand into the deep edge with the same control and data plane considerations, as discussed in Sections 4.3 and 4.4 respectively, and resource management considerations, as discussed in Section 4.5, applying to all those resources. In other words, in principle, we see aspects of controllability of those edge resources to equally apply together with the general programmability for the realization of compute tasks as well as for data and forwarding plane operations through those resources.

However, some edge resources might not directly fit into this vision. For instance, IoT will introduce particular, service-dedicated, possibly intelligent yet resource-constrained components (micro-electronic, battery driven components), which will need a particular consideration for the integration with the rest of the system. Indeed, such IoT components and devices might impose additional requirements on, e.g., volatility and longevity, punctual presence at any moment, persistence, generality, capacities, connectivity, interfaces and APIs from/towards the system. Hence, they might not support direct integration and require particular solutions instead (e.g., gateways or subsystems).

Generally, edge resources often provide human- or generally task-centric input and output capabilities, expressed in a plethora of sensory capabilities, situational awareness, quality of experience perception, which make these resources very useful for integration into the overall vertical application. This yields a *richness* of resources that is challenging when being integrated into a common resource worldview. Unlike the emerging COTS (customer-off-the-shelf) platform basis in other parts of the communication system, e.g., in the core, the edge provides a more *diversified and heterogeneous environment* with many device platforms and their supported local connectivity technologies (e.g., WiFi, BT, LiFi, and others), all of which are provided through a plethora of programming environments. **Future research will need to develop a suitable common model of system-wide representation akin to ‘device drivers’ in existing computing platforms.**

This resource richness at the edge, however, often comes with a **limitation** **in capability**, e.g., in terms of available processing cores in smartphones that can be utilized in the common resource pool. Given that devices at the edge exhibit a high heterogeneity ranging from a simplistic sensor and IoT devices to edge data centres, other typical limitations include energy/battery, form factors, human-machine interface, storage, physical security. This stands in stark contrast to the perceived limitless resource capabilities in data centres as well as core networks and, therefore, impacts the *decomposition* of computational tasks over a resource pool that is geographically and physically limited. As a consequence, the aforementioned *controllability* will need to be ensured through the realization of a suitable *control agent* that integrates the (edge) resource pool into the larger system but also interfaces with the (edge) resource pool to adequately govern the resource usage in the light of the resource-specific characteristics in terms of constraints and dynamicity. Here, research into the *minimal requirements* in terms of processing and communication needs and the realization of those requirements as *novel control agent realizations* will need to ensure that integration into the overall control fabric of the larger system to align with our vision of a smart network as laid out in Section 4.2. Furthermore, *resource scheduling* requires extra consideration in the presence of potential resource scarcity, particularly when combining specific input/output capabilities into the scheduling decision. Scarcity may be increased when utilizing specialized resources, such as GPUs or NPUs, rather than general purposes ones. We may also find that *locality of the resources* becomes crucial when applying policies for, e.g., localized processing for privacy reasons. Scheduling solutions are required that provide suitable trade-offs between moving data to functions or vice versa, possibly under locality constraints. Ultimately, a scheduling decision in favour of one tenant may result in detrimental performance of another, calling for solutions to resource scheduling that likely extend beyond those operating on a large pool of resources with uniform capabilities. **Future research will need to address these edge-specific constraints through suitable scheduling mechanisms that take those constraints into account, while relying on edge-specific control agents enabling the enforcement of the policies underlying the scheduling solutions.**

The *dynamicity* of (edge) resources is another aspect to deal with as an edge-specific constraint. While edge infrastructure, such as in an industrial site, can obviously be very well managed and long-lived, we also foresee edge resources of a much higher *volatility*, particularly when considering end-user provided resources, therefore creating a *limitation in availability* in contrast to, e.g., long-lived data centres. Those resources could be switched off, temporarily disconnected or simply become unavailable, e.g., if linked to human behaviours or policies (such as “do not make my phone available, if battery drops below 15%”). From a control perspective, *maintaining the basic control fabric* needs to take such dynamicity into account, while the *scheduling* will need to react to disappearing and reappearing resources alike to operate at a defined optimum of resource usage. From a data plane perspective, volatile resources need consideration when *routing packets* but also when *establishing in-network state* for forwarding operations. While volatility of resources and dynamics are already covered by the controllability framework presented in Section 4.3, **future research will be required to delve into the systems of systems aspect of such controllability**, given that individual subsystems might not be fully independent.

Furthermore, *governance* of edge resources (and their provisioning through entities like individual users and localized industries) differs vastly from the often long-lived contractual relationships we can identify in the core network business. Instead, the addition and usage of resources with such volatile and temporary nature requires means for *contractual management*, including methods for billing, accounting as well as authorization of use that align with the dynamicity of the envisioned relationship. *Distributed ledger technologies* and *eContracts/ smart contracts* will likely lend themselves to being applied in this world of (possibly highly) ephemeral resource utilization with the appropriate means to keep the resource owner (e.g., the end user) in the loop in order to preserve *digital sovereignty* but also enable *participation in the digital market****,*** akin to the changes in the energy market but likely much more dynamic. An important challenge for entering contractual relations is the *advertisement of resource capabilities*. While today’s solutions are mainly focussed on the pure ability to communicate (e.g., through advertising a radio bearer), solutions are required that expand the negotiation towards clearly articulated *demands* beyond ‘just communication’ that can be dynamically matched against the *supply*. For instance, attaching to a WiFi access point is futile, if connectivity to particular backend services is not enabled at this edge resource. *Efficiency* is key here, avoiding unnecessary signalling between components. Particular consideration must also be given to *security*, both towards the tenant utilizing the resources and those providing them. With tenant-specific instructions eventually being executed on what are possibly end-user provided devices, *accountability* for this usage is key for accepting such usage in the first place, complementing (edge) platform capabilities such as secure enclaves to ensure trustworthy execution at the level of the computational instructions themselves. **Through research in this space, we foresee future solutions to enable an edge resource market that would allow for auctioning the availability of resources to tenants very much like the bidding for white space on a webpage as we know today, basing all interactions on a trusted, auditable, and accountable basis that caters to the dynamics experienced at the edge.** For this edge resource market to emerge, policy descriptions with their rules and constraints will need to be specified in a form that can be enforced by the infrastructure on the services, since direct human oversight is not feasible at this scale. **This will require research into novel programming models and (e.g., policy) languages that not only support all of these services, applications and deployments but also cater to the expected dynamics of the market itself**. Deploying and managing a large set of distributed devices with constrained capabilities is a complex task. Moreover, updating and maintaining devices deployed in the field is critical to keep the functionality and the security of the IoT systems. To achieve the full functionality expected of an IoT system, research should be done in advanced network reorganization and dynamic function reassignment. **Research is needed for providing new IoT device management techniques that are adapted to the evolving distributed architectures for IoT systems based on an open device management ecosystem.**

The continued growth in video applications including augmented reality (AR) and virtual reality (VR) required by among others the emerging applications described in Section 3.1, requires new architectural approaches and solutions. Surveillance and monitoring further complicate the space, as will the growth in real-time sensor data e.g., for industry and smart cities. The ongoing shift of TV distribution from broadcast to the Internet will accelerate, requiring at least a 10x increase in video traffic volume with increased performance and resolution. The implications on application level networkingare tremendous: we will need to integrate video services with the web content framework, delivery model and APIs, with effective use of ultra-dense and diverse wired and wireless networks. Video provenance will become a key issue to combat ”fake news” and the effects of AI/ML-generated video that can subvert legitimate content. Strong security and integrity of applications, network transport and in-network processing will be required. A future key development in the system architecture can be the deep integration of application and service functionality pervasively within the network. The concepts of fog computing apply, but also software-defined networks, network function chaining, virtualization and container provision. There are numerous challenges in developing this vision: service discovery is essential, due to the large and distributed amount of data management and processing services. Existing mechanisms are not sustainable and alternative routing algorithms may help scale the routing infrastructure, but there are many open questions on how these will work. For example, the system architecture will have to become far more dynamic, since the network of the future will be addressing billions of sophisticated data management and processing services. Moreover, service provisioning, management and security are critical to effectively manage billions of devices, ensuring they are suitably configured, running appropriate software, kept up-to-date with security updates and patches, and run only properly authenticated and authorized applications. Security models must evolve. Secure boot, code signing and cryptographic verification of the execution environment will become critical, alongside tools to manage and control data access, management and provenance. Authentication of services and service providers, while accounting for resource usage, is an essential part of the economics of the network of the future. Micropayments will become a key part of the system as the infrastructure to support in-network services and applications is not free. Privacy and data management, and the location of processing and data to match legal and moral restrictions on data distribution, access and processing, will be increasingly important. Many of the services and applications will operate on, process and deal with personal data that is increasingly (and rightly) subject to strict regulation, control and limitation. Strong tools do not exist to describe in human language, legal language or code how data can be processed, located and distributed. Policy descriptions, rules and constraints will need to be specified in a form that can be enforced by the infrastructure on the services, since direct human oversight is not feasible at this scale. In addition, novel programming models and languages are required to support all of these services, applications and deployments.

Research challenges in this area include:

* **Delivery model and API**s, with effective use of ultra-dense and diverse wired and wireless networks (cmp. Sections 4.3 and 4.4);
* **effective management of billions of devices**, ensuring they are **suitably configured, running appropriate software, kept up-to-date with security updates and patches, and run only properly authenticated and authorized applications.**
* **Privacy and data management,** and the location of processing and data to match legal and moral restrictions on data distribution, access and processing, will be increasingly important.
* **Policy descriptions, rules and constraints** will need to be specified in a form that can be enforced by the infrastructure on the services.

## References

[C4-01] “62% of all Internet traffic in Asia Pacific will cross Content Delivery Networks (CDNs) by 2021”, online: https://www.prnewswire.com/news-releases/62-of-all-internet-traffic-in-asia-pacific-will-cross-content-delivery-networks-cdns-by-20211-300528121.html. Last accessed May 20, 2020.

[C4-02] “Netflix Eats Up 15% of All Internet Downstream Traffic Worldwide”, Available online: <https://variety.com/2018/digital/news/netflix-15-percent-internet-bandwidth-worldwide-study-1202963207/>. Last accessed May 20, 2020.

[C4-03] IETF SFC WG. Available Online: <https://datatracker.ietf.org/wg/sfc/about/>. Last accessed May 20, 2020.

[C4-04] IETF ANIMA WG. Available Online: <https://datatracker.ietf.org/wg/anima/about/>. Last accessed May 20, 2020.

[C4-05] IETF COIN RG. Available Online: <https://irtf.org/coinrg>. Last accessed May 20, 2020.

[C4-06] IETF FORCES WG. Available Online: <https://datatracker.ietf.org/wg/forces/about/>. Last accessed May 20, 2020.

[C4-07] IETF QUIC WG. Online: <https://datatracker.ietf.org/wg/quic/about/>. Last accessed May 20, 2020.

[C4-08] L. Popa, A. Ghodsi, I. Stoica, ‘HTTP as the narrow waist of the future internet’, Hotnets-IX: Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks October 2010.

[C4-09] D. Clark, “The Design Philosophy of the DARPA Internet Protocols”, Proc. ACM SIGCOMM ‘88, Computer Communication Review Vol. 18, No. 4, August 1988, pp. 106–114.

[C4-10] S. Ren et al., “Routing and Addressing with Length Variable IP Address”, Proc. of ACM SIGCOMM NEAT workshop, 2019.

[C4-11] R. Li et al., “A Framework for Qualitative Communications using Big Packet Protocol”, Proc. of ACM SIGCOMM NEAT workshop, 2019.

[C4-12] R. Li, “Network 2030 and New IP”, 15th International Conference on Network and Service Management, IEEE NOMS 2020. Online: <http://www.cnsm-conf.org/2019/files/slides-Richard.pdf>. Last accessed May 20, 2020.

[C4-13] Pouzin Society, “RINA: Building a Better Network”, available online: <http://pouzinsociety.org/education/highlights>, Last accessed May 20, 2020.

[C4-14] D. Trossen et al., “Service-based Routing at the Edge”, available online: <https://arxiv.org/abs/1907.01293v1>. Last accessed May 20, 2020.

[C4-15] SCION Internet Architecture, online: <https://www.scion-architecture.net/>. Last accessed May 20, 2020.

[C4-16] European Commission, “The European Green Deal”, Available online: <https://ec.europa.eu/info/sites/info/files/european-green-deal-communication_en.pdf>. Last accessed May 20, 2020.

[C4-17] “Between 10 and 20% of electricity consumption from the ICT\* sector in 2030?”, online: <https://www.enerdata.net/publications/executive-briefing/expected-world-energy-consumption-increase-from-digitalization.html>. Last accessed May 20, 2020.

[C4-18] M. Caesar et al., “ROFL: routing on flat labels”, ACM SIGCOMM Computer Communication Review, August 2006, https://doi.org/10.1145/1151659.1159955.

[C4-19] G. Xylomenos et al, “A Survey of Information-Centric Networking Research”, IEEE Communications Surveys & Tutorials (Volume: 16, Issue: 2, Second Quarter 2014).

[C4-20] Trossen et al., “Name-Based Service Function Forwarder (nSFF) Component within a Service Function Chaining (SFC) Framework”, Internet Society, RFC 8677, online: <https://datatracker.ietf.org/doc/rfc8677/>. Last accessed May 20, 2020.

[C4-21] D. Clark, J. Wroclawski, K. Sollins, R. Braden, “Tussle in Cyberspace: Deﬁning Tomorrow’s Internet”, Proc. ACM SIGCOMM 2002. Available Online: http://conferences.sigcomm.org/sigcomm/2002/papers/tussle.pdf. Last accessed May 20, 2020.

[C4-22] Tim Höfer, Sebastian Bierwirth und Reinhard Madlener, „C15 - Energie Mehrverbrauch in Rechenzentren bei Einführung des 5G Standards“, available online: <https://www.eon.com/en/about-us/green-internet.html>. Last accessed May 20, 2020.