871780 — MonB5G — ICT-20-2019-2020

Deliverable D2.1
1st release of the MonB5G zero touch slice management and orchestration architecture

Document Summary Information

Grant Agreement No	871780	**Acronym**	MonB5G
Full Title	Distributed Management of Network Slices in beyond 5G		
Start Date	01/11/2019	**Duration**	36 months
Project URL	https://www.monb5g.eu/		
Deliverable	D2.1: 1st release of the MonB5G zero touch slice management and orchestration architecture		
Work Package	WP2		
Contractual due date	31/01/2021	**Actual submission date**	15/02/2021
Nature	Report	**Dissemination Level**	Public
Lead Beneficiary	ORA-PL		
Responsible Author	Sławomir Kukliński (Orange Polska)		
Contributions from	Sławomir Kukliński (Orange Polska), Lechosław Tomaszewski (Orange Polska), Robert Kołakowski (Orange Polska), Adlen Ksentini (Eurecom), Aiman Nait Abbou (Aalto), Amina Boubendir (Orange France), Anne-Marie Bosneag (LMI), Cao Than Phan (b-com), Christos Tselios (CTXS), Francesco Devoti (NEC), George Guirgis (eBOS), George Tsolis (CTXS), Hatim Chergui (CTTC), Lanfranco Zanzi (NEC), Luis Sanabria-Russo (CTTC), Mohammed Boukhalfa (Aalto), Sarang Kahvazadeh (CTTC), Tarik Taleb (Aalto)		

Disclaimer

Copyright message

CONTENTS

List of Acronyms and Abbreviations

Acronym	*Description*
3GPP	Third Generation Partnership Project
ACT	Actuator
ACT-F	Actuator Function
ACT-S	Actuator Sublayer
AE	Analytic Engine
AE-F	Analytic Engine Function
AE-S	Analytic Engine Sublayer
AI	Artificial Intelligence
ANM	Autonomic Network Management
AP	Access Point
API	Application Programming Interface
BSS	Business Support System
CIS	Container Infrastructure Service
CISM	Container Infrastructure Service Manager
CLA	Closed-loop Automation
CN	Core Network
CNF	Cloud Native function
CPU	Central Processing Unit
CSMF	Communication Service Management Function
DCAE	Data Collection, Analytics and Events
DDoS	Distributed Denial of Service
DE	Decision Engine
DE-F	Decision Engine Function
DE-S	Decision Engine Sublayer
DMO	Domain Manager and Orchestrators
DMaaP	Data Movement as a Platform
DSP	Digital Service Provider
ECA	Event Condition Action
EEM	Embedded Element Manager
EM	Element Manager
E2E	End-to-end

eMBB	Enhanced Mobile Broadband
ENI	Experiential Networked Intelligence
eTOM	Enhanced Telecom Operations Map
ETSI	European Telecommunications Standards Institute
FCAPS	Fault, Configuration, Accounting, Performance, Security
FL	Federated Learning
FP7	The Seventh Framework Programme
GANA	Generic Autonomic Network Architecture
HT	Holt Winters
ID	Infrastructure Domain
IDM	Infrastructure Domain Manager
IDMO	Inter-Domain Manager and Orchestrator
IDSM	Inter-Domain Slice Manager
INM	In-Network Management
IOMF	Infrastructure Orchestrated Management Functions
ISG	Industry Specification Group
ISM	In-Slice Management
ISRB	Inter-slice Resource Broker
ITU	International Telecommunication Union
ITU-T	International Telecommunication Union Standardization Sector
KPI	Key Performance Indicator
LCM	Lifecycle Management
LSTM	Long Short-term Memory
MaaS	Management as a Service
MAN-F	Management Function
MANO	Management and Orchestration
MAPE	Monitor Analyze Plan Execute
MBTS	Micro Base Transceiver Station
MDA	Management Data Analytics
MDAS	Management Data Analytics Service
MdO	Multi-domain Orchestrator
MEC	Multi-access Edge Computing
MEO	MEC Orchestrator
ML	Machine Learning

MLaaS	MonB5G Layer as a Service
mMTC	Massive Machine Type Communications
MNO	Mobile Network Operator
MS	Monitoring System
MS-F	Monitoring System Function
MS-S	Monitoring System Sublayer
MTP	Mobile Transport Platform
NB	Naïve Bayes
NIST	National Institute of Standards and Technology
NFV	Network Function Virtualisation
NFVI	NFV Infrastructure
NFVinfo	NFVI Runtime Information
NFVO	Network Function Virtualisation Orchestrator
NGMN	Next Generation Mobile Networks
NR	New Radio
NSaaS	Network Slice as a Service
NSD	Network Service Descriptor
NSI	Network Slice Instance
NSMF	Network Slice Management Function
NSO	Network Service Orchestrator
NSP	Network Service Provider
NSSI	Network sub-Slice Instance
NSSMF	Network Slice Subnetwork Management Function
NST	Network Slice Template
NWDAF	Network Data Analytics Function
OAI	Open Air Interface
OAM	Operations, Administration and Maintenance
ONAP	Open Network Automation Platform
OPS	Operations
OODA	Observe, Orient, Decide, Act
OSM	Open Source MANO
OSS	Operation System Support
PaaS	Platform as a Service
PNF	Physical Network Function

PoC	Proof of Concept
QoE	Quality of Experience
QoS	Quality of Service
RAN	Radio Access Network
RL	Reinforcement Learning
RFBs	Reusable Functional Blocks
RL	Reinforcement Learning
RNN	Recurrent Neural Network
S2R	Strategy to Readiness
SA	Service Assurance
SBA	Service Based Architecture
SDN	Software Defined Network
SECaaS	Security Service Platform
SecPaaS	Security Platform as a Service
SFL	Slice Functional Layer
SLA	Service Level Agreement
SM	Slice Manager
SML	Slice Management Layer
SNMP	Simple Network Management Protocol
SO	Slice Orchestration
SOD	Slice Orchestration Domain
SON	Self-Organizing Network
SSLA	Security SLA
TD	Technological domain
TMF	TeleManagement Forum
TMN	Telecommunications Management Network
uRLLC	Ultra-Reliable Low-Latency Communication
V2X	Vehicle to Everything
VIM	Virtual Infrastructure Manager
VNF	Virtual network Function
VNFinfo	VNF Runtime Information
VNFM	Virtual network Function Manager
VS	Vertical Slicer
ZSM	Zero-touch network and Service Management

Executive summary

This deliverable describes the initial MonB5G zero-touch slice management and orchestration framework that aims to facilitate the deployment of a massive amount of slices in different administrative and technological domains.

The future 5G networks are projected to support massive numbers of network slices working concurrently, which, together with the already high complexity of the network slicing solution, makes the tasks related to management and orchestration problematic. The elevated requirements for coverage, bandwidth and latency as well as inter-domain operation further exacerbate the complexity of network management, making already devised, standard, human-centric managing solutions insufficient and ineffective. The currently widespread centralised approach to network management also negatively impacts the separation and security of network slices as well as the complexity of the central managing entity. Furthermore, centralisation also increases the overhead related to slice management data that has to be sent to the management system during the slice operation.

Zero-touch management is perceived as one of the key concepts that can significantly contribute to the simplification of the human-based tasks for network slice management and orchestration. With the extensive usage of the AI-driven mechanisms, its goal is to provide self-managed networks with little to no human interaction. The concept is currently heavily researched by the standardisation bodies.

The proposed initial version of the MonB5G architecture addresses the aforementioned issues by enabling functions distribution and providing strong separation of management of network slices' run-time and orchestration domains. The concept facilitates self-managed slices composed of self-managed functions, further extended to slices created in multiple orchestration domains. The issue of management complexity is addressed by using AI at multiple levels to achieve specific management goals and to minimise interactions between architectural entities, e.g. by means of hierarchical closed-loop controls and aggregated Key Performance Indicators (KPIs). The heavy emphasis is also put on the security, management programmability, and energy efficiency aspects of network slicing. Furthermore, the proposed concept is deeply rooted in the already devised network slicing management and orchestration solutions that have been developed by other EU projects or research and standardisation bodies.

In the deliverable, the initial MonB5G zero-touch slice management and orchestration framework that facilitates deployment of the massive amount of slices in different administrative and technological domains is presented. The main goal of the proposed approach is to achieve the scalability of network slicing management. To that end, we have used a distribution of AI-driven management functions at multiple levels of the management hierarchy (node, slice, domain, and inter-domain). At all these levels, the management-related is processed that leads to reduced information exchange in comparison to the centralised solutions. The exchanged information between the management levels is becoming abstracted (intent-basd0 and finally it leads (at the inter-domain level) to KPIs exchange. The use of the AI-driven In-Slice Management concept (ISM) has reduced the number of slice external interfaces (the management plane of a slice has become a slice template) and provides a perfect separation of the slice management plane that cannot be achieved in the 3GPP approach to network slicing management. Moreover, the approach has made it possible to use the intent-based interfaces for the slice management by slice tenant or slice management provider. The Monb5G allows for the implementation of management as dynamically deployed PaaS (i.e. a set of orchestrated functions that are devoted to a specific slice template). Such an approach reduces slice footprint but provides a weaker management plane of slices isolation than the ISM approach.

Instead of a single orchestrator based approach, we have used a multi-domain orchestration and separation of each slice's runtime management (made by ISM) from domain resource management that is provided by a domain manager and orchestrator (a combination of resources orchestrator and resource-oriented OSS/BSS). The implementation of slice management as a part of a slice (i.e. a set of

VNFs) provide better scalability of slice management performance and allows for the programmability of slice management services during slice lifetime. According to the ITU/TMN FCAPS approach, we assume the security is a network/slice management services, therefore, is treated just like one of the management services (not an external add-on). A novelty of our approach is the capability of the ISM to trigger its slice modification request (typically based on slice-specific analysis), enabling that way proactive management operations and contributing to the agnostics of the slice orchestrator. In the 3GPP and the ETSI MANO approach, such operation is triggered by the centralised OSS/BSS. In MonB5G, the slice orchestrator is mostly focused on domain resources and is linked with its OSS/BSS that performs appropriate, resource-oriented functions. It is agnostic to slices orchestrated by it. The split into separately managed and orchestrated domains reduce the overall management traffic. In order to reduce it more, we have used the well-known, KPI-based approach to exchange for monitoring information between domains, and the intent-based approach for configuration changed. The MonB5G approach allows for resource brokering and energy-efficient operations. For that purpose, we have modified the existing interfaces between the infrastructure and other components of the architecture.

This document is an initial step towards the development of the architecture that could enable the implementation of network slicing solutions on a massive scale in commercial use cases as well as create new business opportunities in the field of 5G network slicing and management and orchestration of telecommunication networks. In the document, we have included an overview of the related architecture, outlined the initial concept and described some implementation options. In the future, the architecture will exploit the MonB5G project progress, especially concerning the implementation of monitoring functions, analytic and decision engines, and its updated, final version will be included in the Deliverable 2.4.

1 Scope

This deliverable reports on the activity of the MonB5G project's Work Package 2 (WP2) describing the initial concept of Zero-Touch Distributed Slice Management Architecture. The document includes the current state of the art regarding management and orchestration, standardisation efforts regarding network slicing, autonomic management and the most important projects and their achievements related to slice management and orchestration. The most popular management and orchestration platforms such as Open Network Automation Platform (ONAP) or Open Source MANO (OSM) have also been described. The core of the document is the description of the initial MonB5G architecture, which is followed by its mapping to selected use cases and example of implementation.

1.1 Novelties

The proposed preliminary concept is, according to our best knowledge, the first one that addresses the scalability and robustness of network slicing management and orchestration by using a distributed and programmable management architecture. AI-enabled management operations are adopted at different levels of the management hierarchy. The novel approach to slice management has also been incorporated, i.e. AI-driven slice management functionalities can be embedded as a part of a slice providing in that way higher elasticity in the creation and deployment of diverse slice types. The framework also provides a strong separation of concerns, which contributes significantly to complexity reduction and easier administration of slices, especially in case of multi-domain slices, deployed over different infrastructure domains belonging to several owners. Altogether, the above-mentioned features enable making a significant step towards self-managed network slices.

1.2 Deliverable structure

The structure of this deliverable can be summarised as follows:

- Section 2 describes the motivation that drives the creation of a new network slices management and orchestration approach;
- Section 3 presents related work concerning network management, including the fundamentals, autonomic and cognitive networks, network slice management and orchestration and autonomic slice management frameworks proposed by standardisation bodies or developed within EU projects;
- Section 4 is devoted to the description of the initial MonB5G architecture with respect to the overall principles of the frameworks as well as details regarding its internal components, mechanisms and operation;
- Section 5 presents the MonB5G features in the light of ETSI Zero Touch Network and Service Management (ZSM) requirements as proposed by ETSI;
- Section 6 presents some implementation details of the MonB5G architecture for selected use cases;
- Section 7 concludes the document.

2 Motivation

MonB5G aims to provide a new architecture to achieve scalable and automated management and orchestration of high numbers of parallel network slices as envisioned in 5G and beyond. The MonB5G pillars are: (i) a highly distributed management and orchestration system, deployed over several entities involved in the LCM of network slices, namely MANO, NSMF, MEO, and the slice itself; (ii) data-driven mechanisms, based on distributed machine learning algorithms, to enable self-management and self-configuration of network slices, towards reaching the principle of scalable zero-touch network management.

2.1 Issues concerning management and orchestration

The network slice management differs from classical network management schemes as it requires administrating not a single but multiple network domains. Current MANO solutions mainly focus on centralised approaches, which introduce scalability problems, especially in the presence of multiple administrative and/or technical domains as those envisioned in the network slicing context. In this scenario, the communication between the orchestration entity and the distributed networking entities will be characterised by significant delay and non-negligible traffic overhead, thereby preventing the implementation of standard polling-based monitoring processes. Any resource management decision should be compared to an almost real-time global view of the mobile infrastructure in order to avoid misconfigurations. To guarantee up-to-date monitoring information during the resource allocation process, as well as to allow online reconfiguration operations in response to unexpected network dynamics, an efficient and lightweight decisional process involving closed-loop feedbacks must be in place. The slice setup would likely affect resources spanning across multiple data centres and networking domains. The current MANO framework lacks mechanisms for managing all these attributes properly. The high variability dictated by the mobile network eco-system requires the orchestration process to be both location- and context-aware, namely, it should be able to determine how the different networking functionalities operate based on their location, as well as exploit monitoring information to obtain detailed reports on the current status of the multi-domain resource availability and/or utilisation, e.g. in terms of processing and storing data.

A network slice can be seen as the composition of a set of sub-slices belonging to different technological domains (e.g. RAN, transport, cloud, edge, core network). To overcome the centralised approach, each technological domain may be assigned with one (or more) management element, or agent, logically closer to the pool of resources to be orchestrated, thus enabling faster detection and reaction of domain-specific problems. The idea of having a hierarchical orchestrator is gaining momentum as an enabler for the flexible distribution of management tasks among entities belonging to different network domains. A hierarchical orchestration would enable adaptive function delegation supporting different levels of centralisation of monitoring, analysis, and decision-making tasks, based on the current operation status and the necessary degree of reconfiguration.

Given the hierarchical structure, management decisions could be taken at lower levels limiting the monitoring overhead and reducing the reaction time. When needed, monitoring information may be directly extracted and locally aggregated by the distributed agent, which will also be in charge of reacting to unexpected scenarios enforcing reconfiguration policies. If not enough, the agent will perform an initial inference task and provide the upper layers with more refined information (rather than raw monitoring data) to ease the problem solving and reduce the communication resource consumption. In this regard, the number of layers and time-scale of information exchange among adjacent levels (horizontal and vertical) are particularly critical aspects to consider due to their impact on the overall capability of the system to promptly react to changes (e.g. in case of performance degradation or faulty operation of a network slice, as well as quickly identifying security-related issues). Therefore, finding the best

combination of a number of layers and function delegation in the orchestrator hierarchy is crucial. As a starting point, a sub-optimal solution with a static number of available layers could be employed. However, we envision AI as the key enabler of such distributed management to devise an automatic solution able to optimally disaggregate and distribute the different management tasks to each hierarchical layer according to the current and future network status.

2.2 Management scalability issues

Scalability is a typical problem for network management. In the era of virtualisation, the management operations can be split between the orchestration (MANO) and classical management functions, where management does not have to cope with hardware. However, the network slicing paradigm can lead to serious management scalability problems. As a network slice can be seen as a network instance, the management has to cope not with one but with the unknown a priori number of networks to be managed. So far, in the MANO approach, a single Operations Support System (OSS) is in charge of that that raises the issue of scalability and lack of separation of management plane of slices. Moreover, in such an approach, the slice tenants may have a problem with the management of their slices.

The MANO orchestration is automated per se, but some scalability issues pertain. No multi-orchestrator solutions are natively supported, and no individual blocks of the orchestrator have their performance monitored. Moreover, there is a limited impact on the behaviour of the MANO orchestrator, e.g. it is not possible to have an impact on the way the VNF placement is carried out (except for the Network Service Descriptor (NSD) flavour), and in general, automated run-time resource and other reconfiguration options are rather limited, inflexible, and monolithic (e.g. based on preconfigured rules). Proactive resource scaling has to be accomplished using iterative interactions between OSS/BSS and NFVO.

From an implementation point of view, choosing a centralised approach may have the advantages of a low delay in the setup of the slice, helped by the fact that the decision is taken by a single entity with a global view of the network. In order to allow a centralised entity to optimally perform its orchestration tasks, it is also important to define and design a complex monitoring system reporting updated information about the different domain resources availability, consumption and congestion levels. Such information is crucial to reduce the probability of orchestration errors and enable efficient admission and control mechanisms. However, in complex and wide networks as those expected to support the future 5G ecosystem, it is practically unfeasible to transmit, with the adequate level of time granularity, the set of massive KPIs and monitoring metrics related to the high number of slices without introducing significant communication overhead, which will impact on data-plane transmissions, therefore, reducing the overall network efficiency.

On the one hand, the centralised decision entity represents a single-point-of-failure in the network architecture, which, in case of failure, can affect all the network management operations and lead to critical results. On the other hand, a centralised approach may introduce communication security and confidentiality drawbacks, as each domain would need to provide the decision entity, in a continuous way, with sensible monitoring information over the network.

MonB5G aims to provide solutions to the aforementioned scalability challenges and lack of autonomic management and orchestration mechanisms. MonB5G proposes to automate network management and orchestration by using **AI-based algorithms and distributed automated operations**. We do not assume that the solution is flat but instead propose a **hierarchical approach**, which allows for flexible distribution of management tasks among entities at different levels of the hierarchy, while supporting **different levels of centralisation by the optimal adaptive delegation of monitoring, analysis, and decision-making tasks**, based on the current operation status.

2.3 Impact of decentralisation on network management

A network slice can be defined as an independent and logically-isolated end-to-end virtual network, dynamically instantiated over shared physical infrastructure. From a network slicing management point-of-view, ad-hoc resource allocation schemes should span all the different networking domains, including the radio access, transport, core and data-centre segments, in order to orchestrate networking, computing and/or storage resources aiming at the satisfaction of 3rd party industry verticals' business requirements. The need for enhanced network management schemes to overcome the scalability issues described in sections 2.1 and 2.2 enticed the proliferation of decentralised approaches.

In a distributed management system, the absence of a single point of failure increases **reliability**, **scalability** and **security.** This is because individual management entities are not reliant on a single central server to handle all their processes but can, upon domain-specific limitations and boundaries, autonomously enforce orchestration decisions, therefore, decreasing the risk of a bottleneck in the network, and hence increasing network **reliability.** This design is also inherently more **secure**, as there is no central server for attackers to target, where hypothetically, attackers would then need to gain access to a large number of networked computers in order to compromise the network. Decentralised networks are also much **easier to scale**, as additional computing power can be easily added, in a horizontal way, to match the growing complexity of today's deployments. A decentralised orchestration approach would also allow each technological domain to play a role in the slice provisioning and definition. Finally, decentralised network architectures offer compelling **cost advantages** in comparison to an enterprise assuming full responsibility for network design, deployment and management. However, this shift from centralised to distributed management also introduces novel technological challenges that need to be addressed.

- **Distribution** This poses severe challenges in defining the set of actions that each decision agent may (or may not) perform, and under which operational conditions. Therefore, a distributed management plane will need to go hand-in-hand with the deployment of data-driven mechanisms based on Artificial Intelligence (AI) algorithms.
- **Synchronisation** Local decisions taken in one technological domain can affect the performance obtained into another and influence the resource allocation scheme to be performed. Since network slices span different networking domains, feedback loops control should be adopted to synchronise the decision-making process of distributed MANO elements into the network and avoid continuous and costly reconfiguration efforts.
- **Security** Distributed systems still rely on the exchange among the different architectural elements of network-management data and monitoring information which requires novel trust-based mechanisms that not only deal with the authentication of reliable monitoring and collection of data but also verify the trustworthiness of slice composition and deployment.

2.4 Benefits of AI for network slices management and orchestration

5G introduces the use of virtualisation technology as a means to offer customised communication service capabilities over the same infrastructure by partitioning it into slices [1][2].In this way, it is possible to satisfy the service requirements of different vertical industries [3]. The slices consist of a set of Virtual Network Functions (VNFs) that encapsulate specific sub-services that the slice needs to provide the service functionalities it was designed for. The VNFs are mapped to physical nodes of the infrastructure, while the virtual links of the slice are mapped to physical links.

The slicing functionality is achieved by leveraging SDN and NFV techniques [2][4][5][6][7]. A slice represents a virtual subset of the physical resources of the infrastructure that have been assigned to a tenant, which is the entity that reserves and pays for the resources of the slice. In this setting, the number of slices deployed simultaneously is expected to be very large, and coordinating their deployment over the infrastructure will be extremely difficult for a human being, if not impossible. Since manual management and coordination are unfeasible, automation tools need to be deployed in order to achieve these tasks efficiently.

AI is a natural candidate to automate the slice management tasks [8] since state-of-the-art research on AI has demonstrated the benefits it can achieve in this context. Some of the sub-problems of the slice management tasks include slice admission control [7][9][10], which in turn includes slice scheduling and slice collocation problems. The fundamental issue in all these tasks is to ensure that the VNFs of the slice and the virtual links between them can be mapped efficiently to physical resources in the infrastructure, making it a resource allocation problem [6]. The allocation of resources to the network slice in a timely and optimal manner is to ensure that the performance constraints of the slice, defined in its SLA specifications, will be met. But mapping the performance constraints to SLAs and then to actual resource allocations may not be very straightforward, depending on the SLA specifications and the definition of the performance constraints. AI can also be useful in order to generate SLA specifications that better represent the performance needs of the slice, and can also be used to drive resource allocation mechanisms [11].

In addition to slice admission control, it is necessary to also dynamically manage the slices, by readjusting their resource allocation to match their current demand, in order to optimise the resource allocation of the infrastructure [1][5][6][12][13][14], thus truly bringing forward the benefits of network slicing. Proactive management of resource utilisation based on AI prediction methods is also a useful aspect. Other aspects of the slice management problem can also benefit from AI, such as monitoring, data acquisition, channel state prediction [15], security and fault detection, which all also need to be considered for 5G deployments [8].

State-of-the-art research has documented many solutions for the problems related to slices management. For example, AI can be used for monitoring and data acquisition, since it can be used to infer the operation status of the components of the infrastructure [8][15]. Another common use of AI in 5G networks is for traffic forecasting [4][9][13][16][17][18], which is useful in the design of slice admission control policies and dynamic resource management mechanisms. In [4], the authors employ a recurrent neural network (RNN) for traffic forecasting in order to drive the dimensioning and positioning of edge data centres of the 5G infrastructure. In [16], the authors predict traffic using the Holt-Winters (HW) method but only apply prediction on the subset of cells that are pre-classified using an NB classifier. This pre-classification is used to establish whether the traffic in the cell is, in some way, predictable. Similarly, an HW predictor was used in [9] and [10]. In the latter the authors also employed a Reinforcement Learning (RL) approach to modify the parameters of their traffic forecasting predictor in order to reduce SLA violations. In [17], the authors used an LSTM-based traffic predictor to drive bandwidth reconfiguration among the slices. They modelled the bandwidth reconfiguration problem as a fractional knapsack problem [19].

AI can also be beneficial for improving the users' experience according to the services they access. Given the current regulations for data privacy, it is not possible for service providers to accumulate users' data. Thus, a new AI technique emerged that trains agents for optimisation tasks without directly accessing users' data. This technique, called Federated Learning (FL) [20], works as follows: in the point where the users' data is being generated, an agent learns using the local data it has access to, but the data never leaves the users' device. Once it learns, it sends its model parameters to a central agent that aggregates the models of multiple logical agents. This is done to increase the accuracy of the learned models. A lot of state-of-the-art research has been focusing on how to make FL more resource-efficient and more accurate [21][22][23][24][25], findings that are very relevant for the context of MonB5G as well. Moreover, FL has been used for many applications in mobile networks that improve users' experience, namely cyber-attack detection, edge caching and computation offloading, base station association [26], and predict users application demands [27].

RL has shown vast successes in many applications, such as natural language processing and robotics [28]. It has proved to be a viable tool when tackling real-time dynamic decision-making problems.

RL is a promising tool for solving resource management and other optimisation challenges in 5G networks characterised by temporal variation and stochasticity of service and resource availability. Distributed variants of RL, such as Multi-Agent RL, can benefit the MonB5G architecture by introducing automation in MANO tasks in a decentralised fashion. Some of the benefits RL can provide are listed below.

- Intra-slice reconfiguration. Well-studied RL algorithms could efficiently be used to reconfigure the VNF placement and chaining inside a slice dynamically [29]. Based on the capacity and load of connecting links and servers, the congestion level of local and alternative computing resources, and even KPI predictors from the distributed AEs proposed in the MonB5G, RL could attempt to identify feasible local reconfigurations, affecting only a part of the chain. These reconfigurations could apply a specific policy, without the need for global reconfiguration or migrating the entire chain. This reconfiguration could also utilise the proposed KPI estimates.
- Inter-slice Reconfiguration. RL implementations could be leveraged to simultaneously reconfigure slices. Inter-slice reconfiguration could benefit the performance of the network in cases of non-feasible reconfigurations of a single slice, or resource utilisation across slices, based on the available shared resources [30]. The large numbers of resources, when already fully allocated, could pose multiple challenges for reconfiguring one slice because it can affect all the others and their respective SLAs. Also, different slices could overlap in their placement and compete for only a subset of common resources. That phenomenon creates complicated dependencies between slices. By using novel methods from the recent literature, RL methods could be developed to automatically learn and extract the key dependencies between slices. These dependencies could be performed both locally and globally to efficiently allocate radio, transport, computation, and storage resources between a substantial number of slices. RL could also offer a multi-objective approach to minimise the number of reconfigurations and SLA violations, maintaining almost optimal multiplexing gains, and optimising objectives such as cost, profit, or energy, as indicated by the service provider intent policies.
- Power consumption reduction. RL could introduce the concept of energy slicing, attempting to guarantee the required energy supply in different network domains and resources to satisfy various levels of SLAs [29]. It could be trained to maximise the energy efficiency in the network, by placing the VNFs in the same physical machine when possible and switching off the idle servers.
- Resource Allocation. RL algorithm could be leveraged to predict the optimal computational or network resource ratio between slices and dynamically adjust the performance of each one [31]. As proposed, predicted KPIs could enable proactive resource allocation.
- VNF and slice life-cycle management (instantiation, scaling, termination). RL could be used to decide when to scale in or out, instantiate, terminate or even duplicate a specific VNF, VNF chain, or a slice,

according to its traffic at the moment. KPIs could be used to proactively perform life-cycle management operations and orchestration.

- Security and reliability. RL could prevent specific types of attacks by detecting and acting rapidly, ensuring slice security and isolation. Decentralised data-driven management mechanisms could also be leveraged to adapt to the distributed architecture of MonB5G.
- Prevent performance and service quality degradation. RL could dynamically take decisions to maintain performance, either by migrating VNFs or changing the slice resource ratio to optimise the policy set by the operator.

These enhancements could consider multi-agent RL implementations that can be distributed across the entire proposed architecture of MonB5G, acting in all layers.

3 Related work

In this section, an overview of network management concepts and their evolution is presented. It also includes relevant research activities of standardisation Working Groups and EU-founded projects.

3.1 Remarks on classical network management

3.1.1 NETWORK AND SERVICE MANAGEMENT BY ITU

ITU-T has defined two main management approaches, namely FCAPS and eTOM. FCAPS functions have been defined by ITU-T in the 1990s as the model for network management, consisting of five management services categories, namely, Fault, Configuration, Accounting, Performance and Security. All management functions are classified under one of these categories [32]. The FCAPS model, however, is being challenged in the context of the very dynamic 5G networks. For example, a fault outside the context of the current configuration (which might be very dynamic in the case of SDN-based networks) will make no sense. Similarly, the performance of a Virtual Network Function (VNF) will be linked to the deployment or the elastic parameters that are set [33].

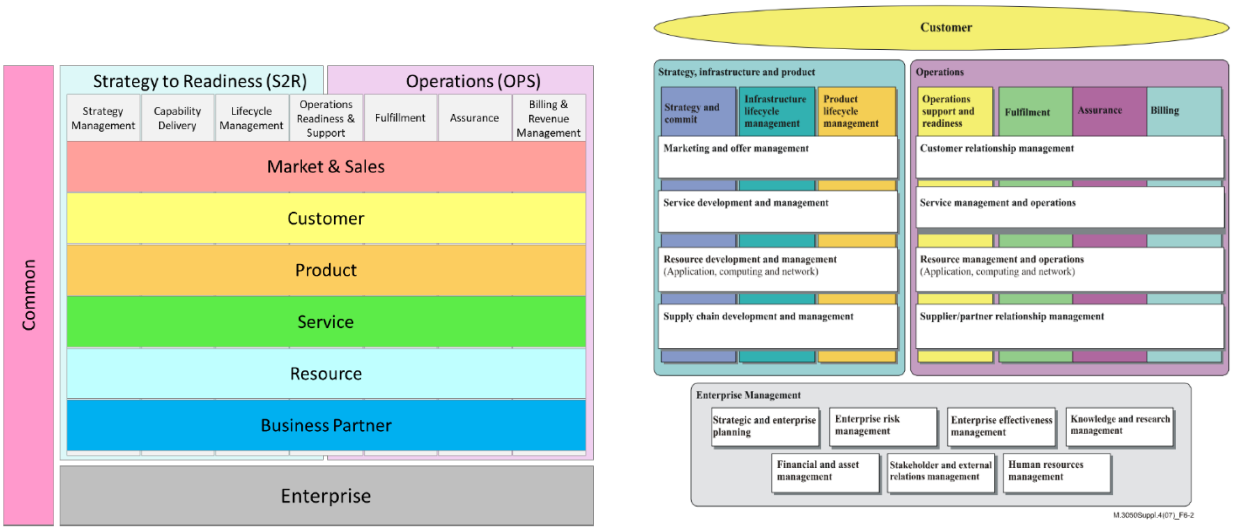

Figure 1. High level overview of eTOM framework proposed by TMF – domains and context verticals[35] (on the left) and ITU-T model adaptation (right)

Enhanced Telecom Operations MAP (eTOM)[34][35] is a framework enabling the categorisation of all the business activities of a service provider into different levels of detail depending on their significance and priority for the business. In a way, eTOM can be considered as a template for standardising business processes as well as operations support systems (OSS) and business support systems (BSS). TMF eTOM and the early version of the model adopted by ITU-T have been presented in Figure 1. From the point of view of suppliers, the eTOM framework outlines potential boundaries of software components and the required functions, inputs, and outputs that need to be supported by-products using the common language of the service providers. The high-level overview of eTOM model consists of three major areas (Level 0 processes) containing a set of functions (vertical) and functional processes (horizontal) that relate to a specific branch of business management. Each area can be further decomposed into components of Level 1, Level 2.

3.1.2 AUTONOMIC NETWORK MANAGEMENT

Automation of network management is based on a set of closed-loop automation (CLA) that meet certain intent or policies, including dimensions, objectives and constraints. To reach these expectations as closely as possible, the CLA should be performed on each layer, including the service layer, the network function layer, and the infrastructure layer. CLA can leverage an autonomic computing framework such as IBM's architectural blueprint for autonomic computing MAPE-K [36] or Boyd's OODA control loop approach for decision making. The MAPE concept has also been adopted by many EU projects related to autonomic and cognitive (i.e. with learning abilities) network management (AutoI [37], Bionets [38], EFIPSANS [39], UniverSelf [40], SEMAFOUR [41]).

The main goal of the Self Organizing Network (SON) concept (defined by 3GPP) is 4G, and 5G RAN management automation [42]. The concept is an implementation of the Autonomic Network Management (ANM) paradigm; therefore, for its implementation, it is necessary to monitor RAN and collect information that is used for near real-time management. The list of SON functions has been defined for LTE, while for 5G RAN (NR), the work is still in progress and there is no detailed architecture of SON provided by 3GPP. Generally, SON is assumed to be a part of the operations, administration and management (OAM) that provides to SON relevant measurements, information about alerts and network events and allows the SON mechanisms to reconfigure network nodes or functions [42]. In 4G the Nm-Centralised SON is implemented as a part of the network management system (i.e. OSS/BSS), in EM-Centralized SON, the SON algorithms are executed at the Element Management level. In case of 5G SON algorithms can operate on different levels of the network: (i) in the Cross-Domain Layer (ii) in the Domain Layer and (iii) at the Network Function Layer. Accordingly, three types of SON are distinguished: Cross Domain-Centralized SON (C-SON) and Domain-Centralized SON that both execute in the management system and the Distributed SON (D-SON) located in the Network Function layer [43]. As the additional source of data, SON can use the Management Data Analytics Service (MDAS) further described in [43][44]. In 5G it is also expected that SON will operate in the Core part of the network and address the NS related operations (resource allocation optimisation, collecting slice relevant data, solving inter-slice issues, etc.). The work regarding this topic is, however, still in progress.

Despite the extensive standardisation efforts, the deployed SON solutions are vendor-specific and are not interoperable. One of the issues with SON is the lack of detailed implementation architecture and interfaces. Especially, no monitoring database that could be used by SON to elaborate its reconfigurations is defined. Moreover, the SON concept does not use the NFV paradigm and orchestration of SON functions so far.

3.1.3 DISTRIBUTED CONCEPTS IN NETWORK AND SERVICE MANAGEMENT

Distributed Network Management is a network management paradigm consisting of performing management functions (FCAPS), network monitoring and control throughout the network in a decentralised way [45]. This paradigm provides more scalability, security and adaptability that are essential with the increasing size of modern networks, especially with the emergence of 5G with a high number of connected devices and new usages, with a variety of QoS requirements and an imperative for data privacy and security.

Existing categories of decentralised management architectures are well summarised in [46]. First, hierarchical architectures are similar to centralised architectures, but make use of a hierarchy of sub-managers to delegate some management tasks. Second, Peer-to-peer architectures, where all the sub-managers communicate directly to accomplish network-wide tasks. Third, fully Distributed architectures, where management services are distributed throughout the network and can be accessed by management applications anywhere in the network. Finally, dispersed architectures contain no

discernible management component – instead, small management agents move about the network and use inter-agent communication and group intelligence to optimise network performance.

A new management paradigm for the Future Internet was developed within the 4WARD project, driven by a European consortium under the FP7 research program. The proposed "In-Network Management" (INM) paradigm leverages the high integration of management functions with the network components: management functions are seen as embedded capabilities, which differ radically from the traditional design and deployment of management functions as add-on features. The benefits range from increased network autonomy to reduced cost of integration [47].

Generic network and service management still cannot ensure a trade-off between centralisation and distribution. This mixed approach preserves the centralised benefits while ensuring distributed control, where management can be performed locally for each, e.g. technological domain and escalated to the central entity in case of failure or need of a higher degree of coordination. This ensures the trade-off between optimal centralised decisions and overhead/delay. Indeed, one of the disadvantages of distributed AI-based networking is that incomplete local information may lead to inaccurate estimation, especially in highly dynamic environments. On the other hand, centralised control depends on periodically collected information, which causes high signalling and computing overhead, especially in the 5G large-scaled network. Furthermore, the end-to-end delay of integrated data and control messages may lead to unexpected control latency and synchronisation problems, especially in the 5G/6G large-scaled network. For 5G ultra-low latency networks, the trade-off between centralised global accuracy and high overhead should be carefully considered. Therefore, In the presence of cloud and dynamic edge computing in 5G architecture, the advantages of decentralised and centralised AI algorithms should be combined in network controllers (multi-agent), thereby trading off complexity, latency, and reliability. This requires integration and further development of methods for data fusion, compression, and distributed decision making. In the distributed setting, there is also the need to develop solutions capable of learning the relationships between the network entities and their time evolution. Since dynamic network inference is a complex task in general, scalable solutions are required.

3.1.4 POLICY-BASED MANAGEMENT

Policy-Based Management is a management paradigm that separates the rules that govern the behaviour of a system from the functionality of the system. It promises to reduce maintenance costs of information and communication systems while improving flexibility and runtime adaptability. This makes policy-based management suitable to implement autonomic behaviour that can exhibit self-management properties, including self-configuration, self-healing, self-optimisation, and self-protection [48].

Two important categories of policies are Event Condition Action (ECA) policies and Intent-Based policies. An ECA Policy is a policy that explicitly specifies an action or set of actions that should be taken when a certain event occurs. The idea is to compile rationality and human or computer knowledge into the policy. Its rule is read as: when an event occurs in a situation where a condition is true, then the action is executed. Therefore, an ECA Policy comprises three key elements: event, associated conditions, and associated actions. An event triggers the evaluation of the condition. A Condition specifies a state or set of states. An Action defines what is required to transition to this state [49].

An Intent-Based policy is defined as an abstract, high-level policy used to operate a network in the context of Autonomic Networks. More specifically, the intent is a declaration of operational goals that a network should meet and outcomes that the network is supposed to deliver, without specifying how to achieve them. Those goals and outcomes should be rendered by the network itself, i.e. translated into device-specific rules and courses of action. Ideally, intent should be orchestrated and broken down by the network devices themselves using a combination of distributed algorithms and local device abstraction. This facilitates management since it obviates the need for a higher-layer system to break down and

decompose higher-level intent and because there is no need to even discover and maintain an inventory of the network to be able to manage it [48].

3.2 Standardisation of network slices management and orchestration

3.2.1 NGMN

The NGMN network slicing framework presented in [50][51] includes multiple logical layers characterised by different operational and business requirements. For this reason, as shown in Figure 2, the management plane defined by NGMN includes a closed-loop AI-enabled feedback control system, namely Knowledge Plane (KP), in charge of automating the operation and performance optimisation of the end-to-end system (multi-domain performance optimisation), and accommodates the requirements of multiple Network Slice Instances, over shared computing, storage, and network resources in the system. Therefore, distribution and decentralisation to optimise the system performance and the user experience can be achieved through closed-loop feedback control mechanisms with system-wide scope and awareness and possible inter-layer interactions with flexible open APIs. At each layer, the KP-enabled autonomic networking provides system-wide capabilities to manage, configure, activate, orchestrate, instantiate, monitor, and decommission the resources associated with the establishment of an end-to-end network slice.

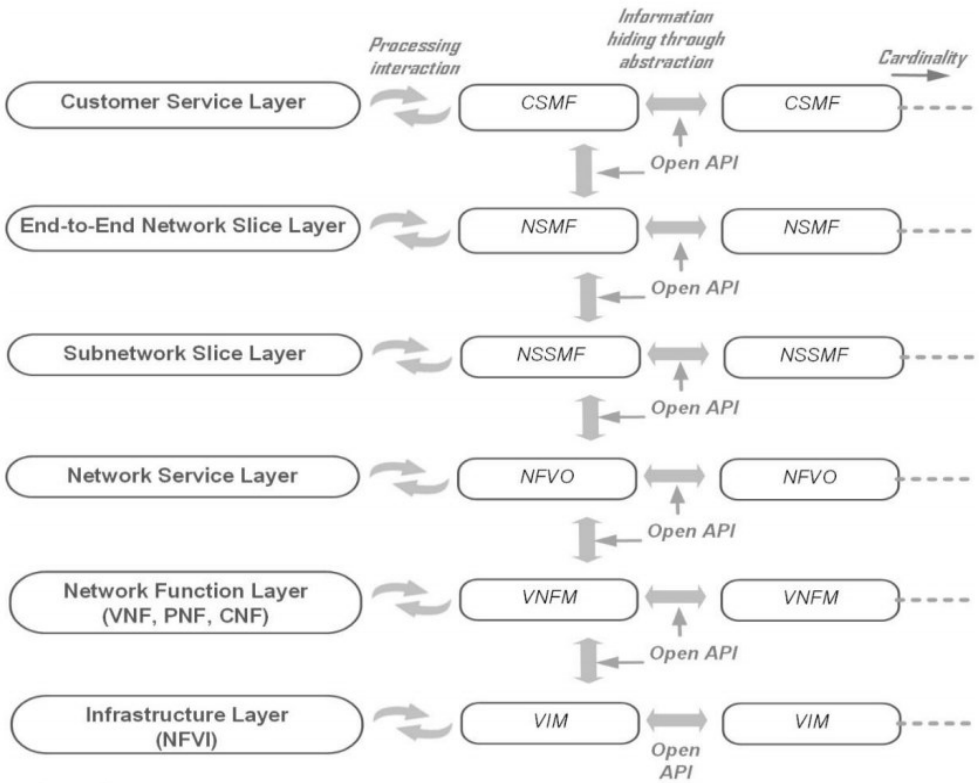

Figure 2. Logical layers from an end-to-end network perspective, taken from [51].

3.2.2 3GPP

Network slicing in 5G is used in the context of service-based architecture (SBA) to create dedicated and tailored logical networks [52] upon a common infrastructure. Four types of Standard Slice / Service Types (SST) have been specified by 3GPP in [53], which are in line with the four major 5G-supported service types (SST = 1 for service type eMBB, 2 for URLLC, 3 for mMTC, 4 for V2X). The mobile network operators must be able to create and manage slices tailored for the different types of services, run them in parallel

and in isolation, provide service-specific security assurance per slice, support network slice elasticity in terms of capacity with no impact on the current slice or other running slices, support creation of a new slice/removal and update of a slice with minimal impact on the ongoing services in other slices, manage resources in an end-to-end fashion for a network slice. The comparison of the 3GPP structure of Network Slicing and the one adopted by ETSI has been presented in Figure 3.

A realisation of a network slice is a network slice instance (NSI) [53] that is composed of Network Functions (NFs – either Virtual Network Functions VNFs or Physical Network Functions PNFs), where the connectivity between different NFs is described by the Network Slice Template (NST) [54]. In this respect, the lifecycle of an NSI includes the stages for the preparation phase, commissioning/decommissioning of a slice, as well as the operation phase (including activation/de-activation, supervision, reporting, and modification stages) as depicted in Figure 4.

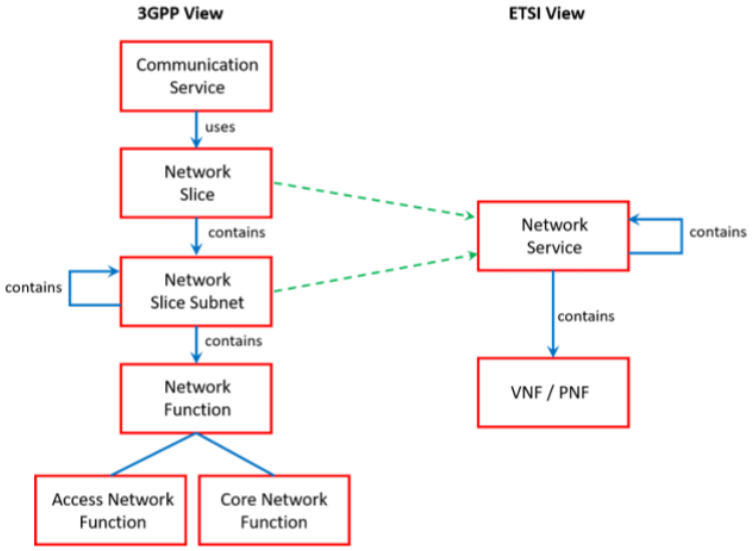

Figure 3. 3GPP Network Slicing – comparison with ETSI terminology

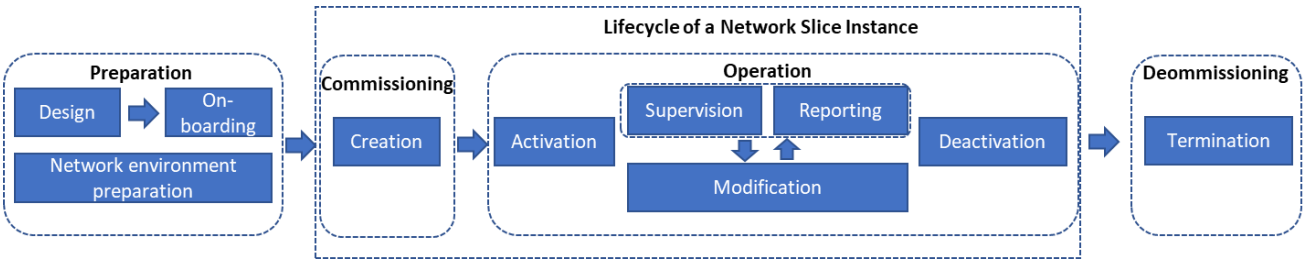

Figure 4. 3GPP Slice lifecycle management

The 3GPP network orchestration and management concept is the most advanced one; it is, however centralised, all the slice-related functions are part of OSS/BSS with the exception of element managers. The 3GPP concept allows for delegation of management operations to slice tenant using the publish/subscribe paradigm. The Network Slice Subnet is managed by Network Slice Subnet Management Function (NSSMF), whereas Network Slice is managed by the Network Slice Management Functions (NSMF). Both functions belong to the Network Management Layer [44]. The functions are responsible for both the lifecycle and runtime management of a slice and reside in the operator's OSS/BSS. The CSMF (Communication Service Management Function) acts as the user interface for slice management and also is a part of OSS/BSS. NSSMF is directly linked with an orchestrator.

3.2.3 ITU-T SG 13

The ITU-T has started its effort devoted to the incorporation of AI support into international standardisation. Within the Study Group 13 (Future networks, with focus on IMT-2020, cloud computing and trusted network infrastructures), two Questions (Q20 and Q21, i.e. "IMT-2020: Network requirements and functional architecture" and "Network softwarization including software-defined networking, network slicing and orchestration") work on AI-related documents, which include the following topics:

- AI integrated cross-domain network architecture,
- The self-organising core network,
- Machine learning function orchestrator,
- ML models in future networks,
- Machine learning sandbox,
- Machine learning-based end-to-end multi-domain network slice management and orchestration,
- Machine learning-enabled network slicing management (including input from verticals),
- Traffic classification support by AI,
- Intelligence capability for network slice management and orchestration,
- Data linkage between AI-based network slice management and orchestration and network slice customers.

The mentioned works are in-line with the scope of the MonB5G project, and most of them are still at the very early stage.

3.2.4 ETSI MANO

The 5G vision proposes a flexible infrastructure, where a pool[1] of white servers (i.e. data centres) provide virtual compute, network, and storage resources to be provisioned on-demand in isolated partitions referred to as Network Slices. Such slices host Network Services in the form of Virtual Network Functions (VNFs), which are connected together via Software Defined Network (SDN) overlays. The ETSI Network Functions Virtualization (NFV) Architectural Framework [55] is illustrated in Figure 5, where the NFV MANO block is in charge of determining the availability of virtual resources in the data centres – referred to as NFV Infrastructure (NFVI) – for the orchestration of a network slice, as well as taking care of the lifecycle management of each VNF, provide telemetry information on the state of the NFVI and VNFs, termination of slices and release of the virtual resources. NFV MANO is composed of the NFV Orchestrator (NFVO), VNF Manager (VNFM) and Virtualized Infrastructure Manager (VIM).

[1] Or geographically distributed pools, e.g. data centers in different locations, or the multi-tiered cloud model.

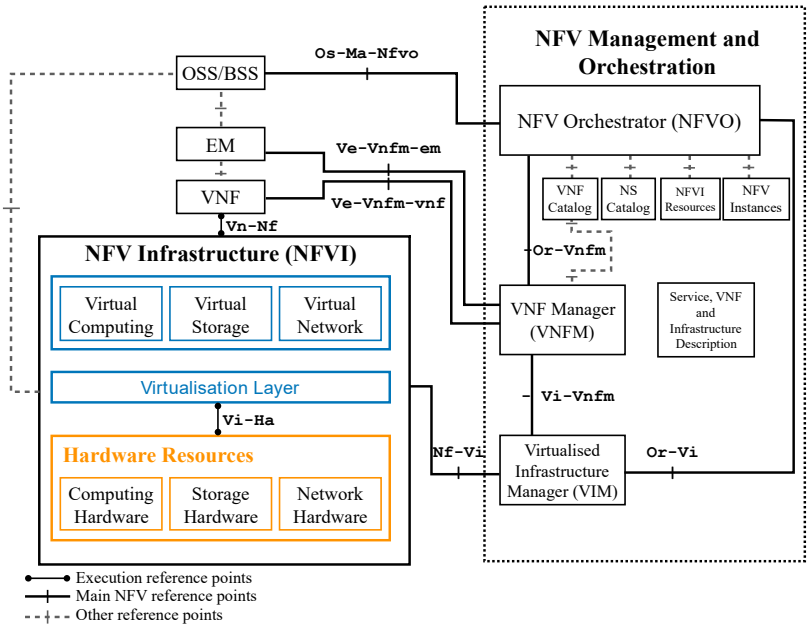

Figure 5. ETSI NFVI MANO Reference Architecture

A brief description of their role in the NFV framework is provided below.

- **Virtualized Infrastructure Manager (VIM) –** responsible for the control and management of the interaction between VNFs and the NFVI hardware resources, such as compute, storage, and network, as well as their virtualisation. It takes care of exposing a pool of virtualised resources derived from the NFVI, as well as the allocation of such resources to each VNF.
- **VNF Manager (VNF-M) -**Takes care of VNF lifecycle management. That implies the instantiation, scaling, and termination of one or several VNFs.
- **NFV Orchestrator (NFVO) -**NFVO is able to gather information about the NFVI from one or several VIMs through standardised reference points or APIs (see Figure 3), and then determine the suitable place in the NFVI to instantiate a VNF. As services are often provided via network slices, NFVO is in charge of satisfying all the slice's VNFs requirements prior to orchestration. All in all, NFVO works as an automation tool for instantiating and terminating network slices from a centralised control position. Furthermore, it enables unprecedented infrastructure re-utilisation by allowing scaling-out VNFs at runtime (e.g. for preserving KPIs) or freeing resources at low-demand periods for energy savings. Scaling-out refers to the replication of an existing VNF; conversely, scale-in eliminates such replicas. Scaling operations are triggered by the NFVO according to a set of user-defined policies. These policies, in turn, are based on telemetry information (e.g. percentage of CPU usage, available memory, etc.) of the VNFs. Reference VIMs, such as OpenStack, provide telemetry services, which in turn publish time series to consumers, such as Prometheus [56], or Gnocchi [57].

The NFV-MANO framework can become autonomous. ETSI NFV is working on such extensions by adding as much closed-loop automation as the scenarios require, each loop to have management data analysis (MDA) modules that using the data collected from NFV-MANO (e.g. Performance metrics of NFV objects using NFV IFA-027 interfaces, Faults, alarms using NFV-IFA 005, NFV-IFA 006, NFV-IFA-007, NFV IFA-013, Runtime information of NFV objects: NFVinfo, VNFinfo) characterise the situation of the observed environment. Depending on the result of the analysis, the decision module can act on all layers of NFV-MANO and above. In addition, the MDA provides services to the NFVO without being aware of the administrative domain. The NFVO of the nested NS should obtain analysis results and report them to the

NFVO of the composite NS as presented in Figure 6. Finally, intent-based management decouples the client and the server and simplifies their interaction, intent as information object can be used between entities of the closed-loop automation (CLA) or at the reference point Os-Ma-Nfvo. The autonomous MANO is of great interest for the MonB5G that applies a similar philosophy to that one proposed by ETSI MANO.

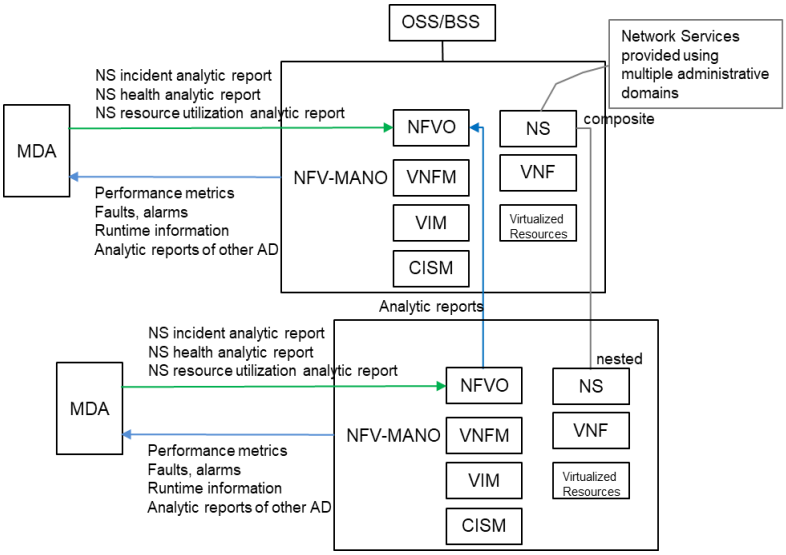

Figure 6. Overview of the interactions between MDA and NFVO across multiple administrative domains

The management system has to be secure, and the same requirement concerns the orchestration solution as well. To that end, it is worth recalling the ETSI NFV SEC sub-group activities concerning security in NFV systems. The security aspects addressed by the group cover hardware and software issues, identity management, authentication, authorisation, and monitoring as well as appropriate measures for operational efficiency and features to support regulatory requirements, e.g. Lawful Intercept, Privacy and Data Protection. In this regard, security in MonB5G architecture will be aligned to the working progress in ETSI NFV, namely:

- **Security Specification for MANO Components and Reference points [58]:** including a simplified threat analysis for NFV-MANO functional blocks (NFVO, VNFM, VIM) and reference points Or-Vnfm, Vi-Vnfm, Or-Vi based on the guidance given in [59]. The objective of this document is to define security requirements on the interfaces between MANO and a Security controller.
- **Security Management and Monitoring specification [60]:** specifying functional and security requirements for automated, dynamic security policy management and security function lifecycle management, as well as Security Monitoring of NFV systems.
- **Report on Certificate Management [61]:** providing guidance to NFV on the use of certificates and certificate authorities. It looks at various certificate deployment scenarios and describes certificate-specific use cases, threats to the certificate management structure, and resulting requirements for NFV.
- **NFV Security; Security and Trust Guidance [62]:** describing the security and trust guidance that is unique to NFV development, architecture and operation. The guidance consists of items to consider that may be unique to the environment or deployment and is based on use cases defined in this document and derived from [63].

Of the MonB5G project interest, there is also a concept of MANO-orchestrated PaaS. This idea is described in [64] document that introduces Platform as a Service (PaaS) as VNF (or Network Slice) in the form of Dedicated or Shared Services. In the management context, a collection of stateless management functions

(i.e. a cloud-native management system) is scalable, in a way that federated management approaches are enabled (e.g. hierarchical management). Such kind of management functions (systems) can be orchestrated within NFV MANO objects (i.e. VNFs, or Network Slice Instances) to provide management services to NFVI or other Network Slice Instances.

3.3 ETSI ZSM

The ETSI Zero Touch Network and Service Management (ZSM) Industry Specification Group (ISG) has been formed in response to the radical change in the way of network and service management caused by the specificity of the network slicing concept. The ZSM ISG is focused on the creation of a framework that enables addressing those needs. ETSI ZSM tries to establish the architecture that enables autonomous networks that can be driven by high-level policies and rules and poses the capabilities of self-monitoring, self-optimisation, self-configuration and self-healing. The goal is a design a horizontal and vertical end-to-end architecture framework incorporating closed-loop automation and optimised for data-driven machine learning and AI algorithms. The ZSM ISG has already established, on the foundation of the adopted set of potential ZSM use cases, a list of requirements that the framework has to fulfil to provide the desired capabilities [65]. On that basis, the reference architecture has been devised.

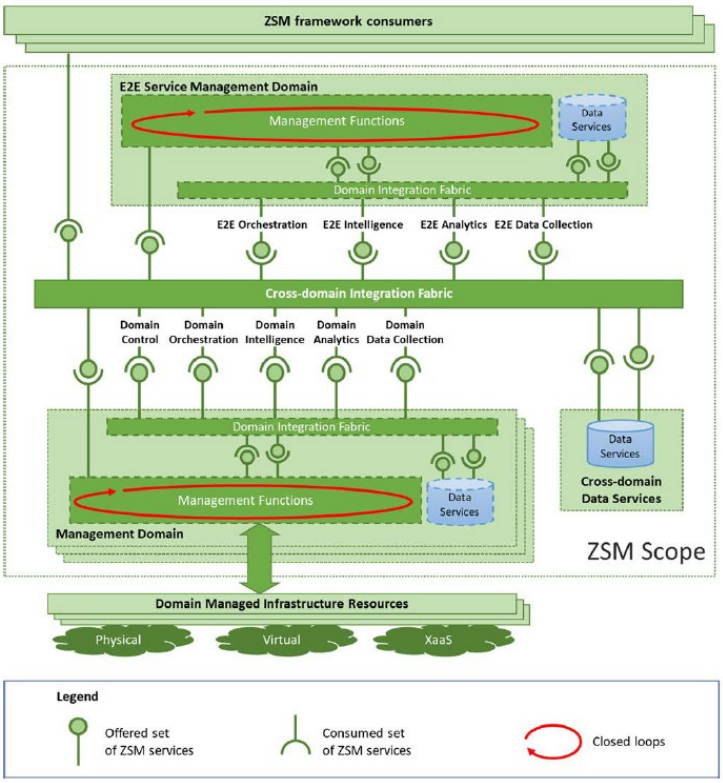

Figure 7. ZSM framework reference architecture [66]

ETSI ZSM promotes stateless management functions that can work in a distributed manner by:

- leveraging virtualisation and 5G Core APIs for building intelligence (e.g. NWDAF) and enabling challenging scenarios (e.g. 3GPP SA6)
- Decentralising network management (e.g. per-slice) leveraging stateless functions running on top of NFV MANO objects, such as Slice Dedicated Service.

The ETSI ZSM ISG focus is currently laid on defining generic enablers, closed-loop enhancements and operations for next-generation closed-loop. The documents that respectively address those topics, i.e. GS ZSM 009-1, GS ZSM 009-2 and GR ZSM 009-3, however, have not been published yet.

The MonB5G project has similar goals to ETSI ZSM however, the project is more focused on the distribution of management functions and the use of AI. It has been, however, decided to describe the MonB5G architecture functionalities in terms of ETSI ZSM requirements (see Section 5 for details). Moreover, the concept of several management domains is used in the MonB5G architecture. The domain-internal functions are in MonB5G architecture more distributed than in the ETSI ZSM case, while keeping a central element in charge of inter-domain management issues, such as cross-domain orchestration placement policies, overall energy consumption optimisation, etc.

3.4 ETSI ENI

The ETSI Experiential Networked Intelligence Industry Specification Group (ENI ISG) is defining a Cognitive Network Management architecture, using Artificial Intelligence (AI) techniques and context-aware policies to adjust offered services based on changes in user needs, environmental conditions and business goals [67]. ENI specifies an architecture to enable closed-loop network operations and management leveraging AI. The need for close-loop operations at any domain of the network and cross-domains requires ENI to be deployed and operate at, e.g. one domain of the network, and/or cooperatively across different domains.

ENI can be deployed as an external AI/ML entity, outside of an existing “Assisted System”. In that context, ENI can be configured to operate in two modes:

- Recommendation mode, to provide insights and advice for the operator or Assisted System, and
- Direct Control Management mode, coupled to the Assisted System control loop to participate in its management, based on gathered data, Knowledge, policies and Context and Situational awareness.

ENI is a closed-loop policy-driven AI/ML specification, designed to employ existing and emerging technologies, such as big data analysis, analytics, and artificial intelligence mechanisms. The Assisted System, depicted in purple above, may use its own native data format for conveying State or Policy. ENI is working on defining, in its current Release 2, an Information Model and internal representation of the data/state gathered and manipulated. For systems that use other formats, an API Broker may need to be deployed outside of the ENI system to provide a translation of its formats/models/data to those supported by ENI. A brief overview of the ENI architecture is provided in Figure 8.

This is a simplified view of the main processing components of an ENI System. The arrows represent the directionality of data and information using any of the External Reference Points. In this respect, [68] specifies the internal interaction and functionality of ENI’s Functional Blocks and External Reference Points, which, per our analysis:

- Does not yet support network slicing, which is treated as a use case, but not as a system trait.
- Covers “hierarchical”, “distributed” and “federated” options, in terms of administrative domains and policy management. The “federated” learning option is also listed as a type of machine learning.
- States that “a primary goal of ENI is to provide a robust, distributed platform that uses modelling, policy management, and AI”. But then it clarifies that “the ENI System is, therefore, a distributed system, in which its Functional Blocks share information and work together, collectively, to manage the state of the Assisted System” or that “ENI may use a distributed or hybrid agent architecture”.

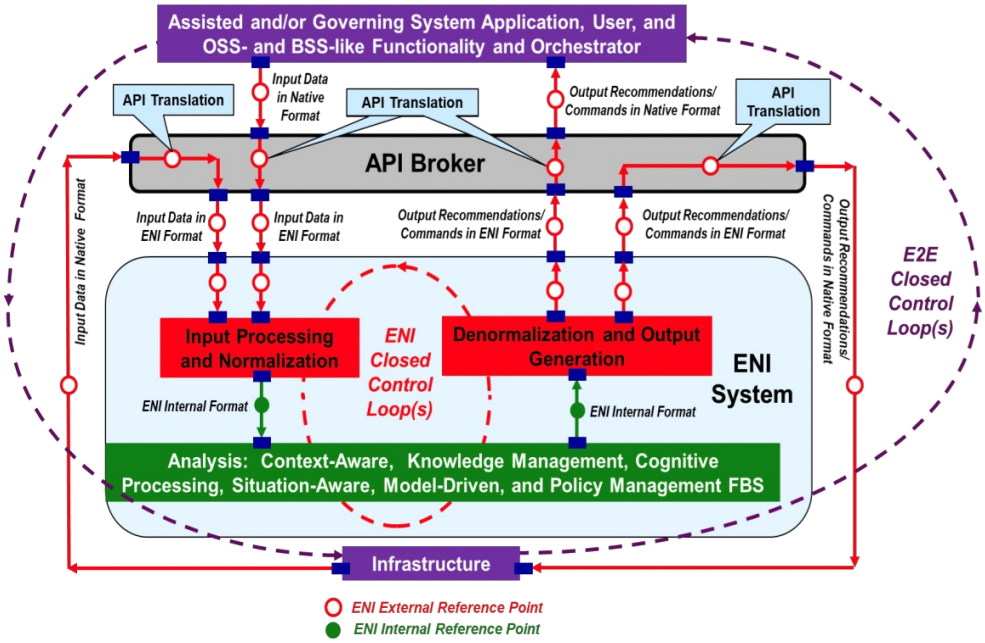

Figure 8. ENI Architecture [68]

MonB5G, on the other hand, has distributed monitoring, analytics and decisions in the core architecture. As elaborated by [69], AI integration and network autonomicity is an important objective of ENI, which is also the case for MonB5G.

3.5 ETSI GANA

The ETSI NTECH AFI group has developed: (I) A reference model for GANA, (II) an implementation guide for the GANA reference model, (III) Autonomics-enabled implementation-oriented architectures that use the different reference architectures defined by the standardisation organisations, and (IV) Proof-of-Concepts (PoCs) Framework aimed at encouraging the industry to setup demonstrations. The GANA reference model, depicted in Figure 9, defines the generic functional blocks and their characteristics. These key functional blocks are enablers for autonomics, cognition and self-features, and they represent an abstract architecture that can be instantiated onto specific reference architecture, e.g. 3GPP architecture. This model tries to avoid the limitations of the current architectures, but it is not designed to be constrained by an implementation architecture.

- **GANA MEs:** MEs represent the bottom layer and the fundamental resources hosted in the network node (NE). They consider physical network elements and functional entities such as protocols, applications. In particular, the element that can be employed, orchestrated, configured and adapted to achieve the network goals.
- **GANA DEs:** The GANA model is a blueprint model that defines Decision-making Elements called DEs. Without any human interaction, DE performs self-configuration, self-diagnosing, self-healing, self-repair, self-optimisation, self-protection. In addition to a secure auto-discovery of network objectives specified by the human operator, the DEs configure and collaborate with assigned Managed Entities (MEs).
- **GANA Knowledge Plane (GANA KP):** is a novel smart management & control system on all the vertical levels, the Element Management (EM), Network Management (NM) and Operation and Support System (OSS), either by interworking with them or by providing a replaceable and (re)-loadable DEs at specific abstraction levels of management and control operations.

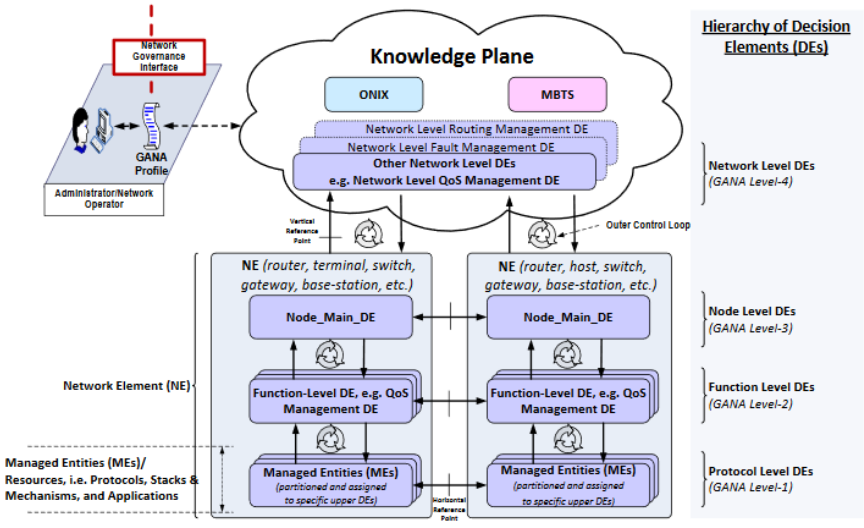

Figure 9. Snapshot of the AFI GANA Reference Mode [70]

The GANA model is about ten years old, and it seems to be very complex and costly to be implemented. The MonB5G architecture reuses some ideas of GANA, but at the same time introduces many separated management subspaces (at slice level or orchestration domain level) that provide a much simpler implementation of the concept.

3.6 Network slice management and orchestration related projects addressing management performance

In this section, we will provide a brief overview of EU-founded projects that had a similar approach, goals or have used similar tools for the management and orchestration of network slices.

3.6.1 SLICENET

The SliceNet project [71]defines management sub-planes within the management architecture of each of the two main actors, namely, Digital Service Provider (DSP), managing the communication services, and the Network Service Provider (NSP), managing the network services. Within the DSP, the management architecture components are related to E2E service-level management capabilities, i.e. E2E Service Orchestration, E2E Service Monitoring, E2E Service Aggregation, E2E Service Cognition, etc. The network slices and resources management components are not required in the DSP domain (since it is not its management responsibility). From the NSP perspective, the required architecture components are related to the management of resources, network slices and network slices as a service exposition – Resource/Slice/Service (NSaaS) Orchestrator, Resource/Slice/Service (NSaaS) Monitoring, etc. Furthermore, SliceNet project considers multi-domain views from NGMN where network services need to be provided across multiple service providers.

Both SliceNet and MonB5G projects focus on cognitive mechanisms to achieve dynamic slice reconfiguration. However, while SliceNet leverages centralised monitoring and AI data-driven control, MonB5G will focus on the hierarchical distribution of AI-driven management functions, in order to significantly reduce the amount of raw data exchanged, enhancing that way the management system scalability.

3.6.2 COGNET

The core of the CogNet [72] architecture, which is presented in Figure 10, is the Machine Learning (ML) block.

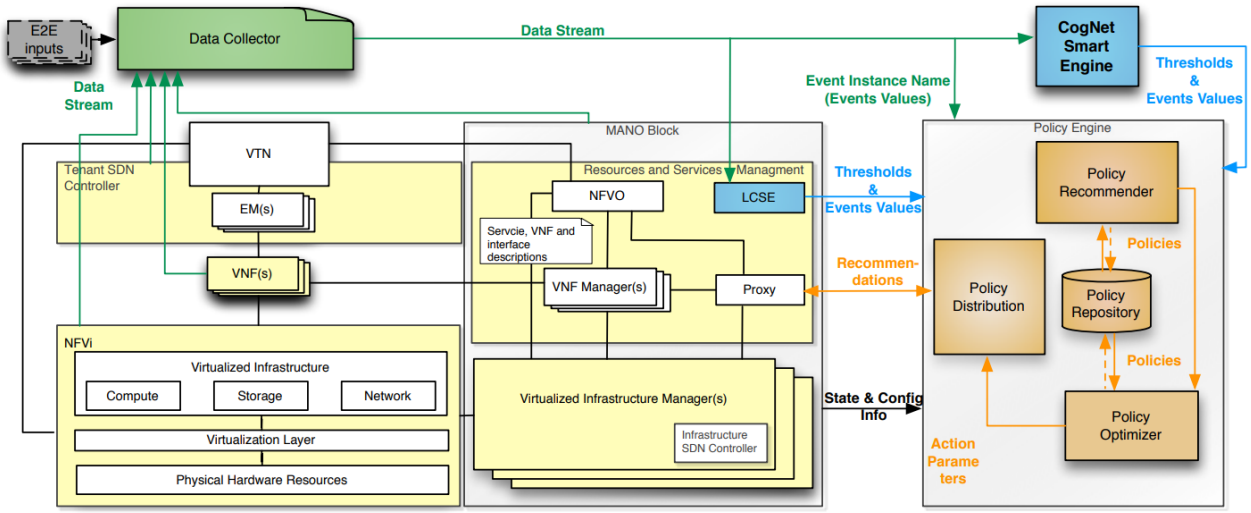

Figure 10. CogNet Architecture [73]

The block consists of (i) CogNet Smart Engine, which receives and pre-processes records, selects algorithms, then applies selected models; (ii) CogNet Optimizer, which transforms the outputs of CSM into optimisation functions; and (iii) Policy Distribution that translates the policies from the Policy Repository and sends them directly to the MANO Block, Tenant Controller and OSS/BSS/VTN. In this regard, CogNet uses predictive, auto-scaling models for VNF scaling to translate policies into actions. Furthermore, they use conventional Machine Learning algorithms capable of forecasting resource demand requirements through usage prediction, recognising error conditions, security conditions MonB5G will build on the CogNet concept, which uses conventional ML algorithms such as Support Vector Machines (SVMs) but leveraging beyond state-of-the-art AI algorithms (e.g. Deep Reinforcement Learning, Federated Learning, Generative Adversarial Networks); moreover, we will also go towards distributed AI-approaches with federated learning.

3.6.3 NORMA

The 5G NORMA [74] project has defined an adaptable network slicing-oriented architecture according to a modular approach with four layers, namely (i) Management including Element Managers (EM), Network Management (NM), and selected OSS functions; (ii) NFV MANO architecture including NFVO, VIM, and VNFM; (iii) cross-slice resource allocations made by the Inter-slice Resource Broker (ISRB) and iv) service management intervening between the service layer and ISRB. The control layer hosts two controllers, Software-Defined Mobile Coordinator & Controller (SDM-X & SDM-C), the first for shared network functions (NFs) and the second for dedicated NFs. Finally, the Data layer hosts the VNFs and PNFs for user data processing.

Compared to MonB5G, NORMA does not offer the same level of scalability as MonB5G. While NORMA's distributed functions are specified only on the control layer, MonB5G tackles the scalability issues by providing automatic management and orchestration mechanisms, by taking advantage of AI-based algorithms and distributed automated operations. Moreover, NORMA did not entertain using AI-based algorithms to solve scalability issues. The MonB5G architecture will reuse, however, the concept of shared and dedicated network functions.

3.6.4 5GEX

5GEx [75] has designed and validated a multi-layer architecture for multi-domain orchestration. In particular, the project has developed multi-domain orchestrators (MdO), which handle the orchestration of resources and services from different providers, coordinating resource and/or service orchestration at multi-domain level, where a multi-domain may refer to multi-technology or multi-provider. The MdO interface with Domain Orchestrators that are responsible for Virtualization Service Orchestration and/or Resource Orchestration exploiting the abstractions exposed by the lower Resource Domains. The 5GEx orchestration framework supports several operations, such as VNF lifecycle management, resource management and control, as well as SLA monitoring. However, 5Gex did not explore the scalable management and orchestration of Network Slices, neither the usage of AI for this purpose.

3.6.5 5G-MONARCH

The MoNArch [76] architecture includes slice-specific and slice-common functions, multi-tenancy-capable management and orchestration, inter-slice resource management and integration of RAN control applications. The management approach is two-dimensional, both inter-domain and inter-slice, and features cross-layer, unified service-based communication between functional instances and cross-layer/cross-domain integrated data analytics framework to aid per-slice context-aware and/or machine learning-based optimisation as well as management and orchestration. MoNArch also addresses the security of network slicing by such mechanisms as anomaly detection for identifying abnormal traffic behaviour and triggering security alerts, the concept of security trust zones or investigation of mutual impacts and synergies of fault and security management approaches for specific use cases. In total, 14 mechanisms and algorithms have been designed, including fast and fine-grained AI/ML-based algorithms, and fully integrated into the overall 5G architecture in order to achieve an optimised utilisation of cloud resources in the complex 5G network, while providing desired SLA under network slicing.

5G-MONARCH focuses on intra-slice and cross-slice controllers to allow re-programmability and functional reconfiguration of slices. MonB5G goes beyond this concept with autonomous dynamic slice management and reconfiguration mechanisms with a hierarchy of local and centralized AE, DE and MS in multiple technological domains.

3.6.6 MATILDA

MATILDA 5G architecture [77] is based on the “separation-of-concerns” concept, which allows individuals and vertical industries to orchestrate their applications and service through a Vertical Application Orchestrator (VAO). MATILDA follows a layered architectural design of four layers, (i) Applications Layer that covers the application components, the virtual network functions, and a set of network resources that attaches the network functions to achieve a complete network service package, (ii) Orchestration Layer that incorporates a set of intelligent mechanisms for optimal deployment, strategic placement, runtime policies enforcement, data mining and analysis and context awareness support for 5G end-to-end network service. This layer takes advantage of a high-level software module that contributes to network awareness and builds an intelligent and innovative orchestration service system; (iii) Network Functions and Resource Management Layer provides the resource management features and the lifecycle management of VNFs, this layer uses both PNFs and VNFs to deliver application-aware network slices; Finally, iv) Infrastructure Layer consists of user data traversing cloud computing and other resources such as Edge/fog computing, edge network, etc.

MonB5G will use AI for network management and take advantage of the network-wide statistics and data to improve the overall network performance. Moreover, MonB5G offers a higher level of flexibility and scalability with distributed management.

3.6.7 5G!PAGODA

In terms of management of Network Slices, 5G!Pagoda [78] has addressed several challenges, including the scalable management of a high number of running slices, where a new concept, namely In-Slice Management, has been introduced (ISM) [79]. In-Slice Management consists of onboarding inside slices a dedicated management function (in the form of VNF), which exposes interfaces to the Orchestrator as well as to the slice owner allowing the latter to execute and run its own management functions. Through this concept, 5G!Pagoda can scale with a high number of running slices, by keeping several heavy management procedures inside the slices (see Figure 11), such as monitoring slice behaviour and update its configuration locally without referring or exchanging information with the centralized Orchestrator. However, 5G!Pagoda did not explore the usage of AI to improve management.

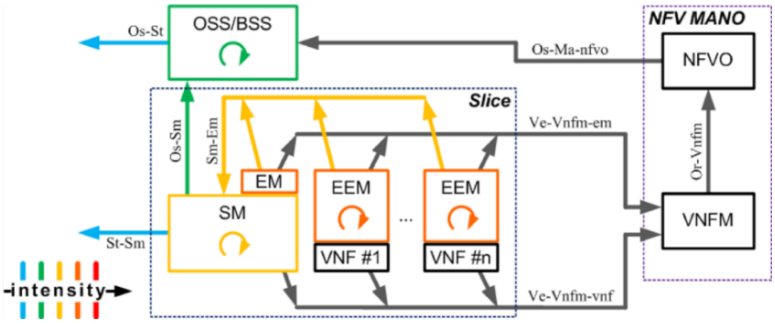

Figure 11. Pagoda Deliverable D4.1: Scalability-driven management system

The MonB5G architecture reuses the ISM concept that is enhanced by AI capabilities. Moreover, the MonB5G approaches contribute to distributed orchestration, an issue that has been neglected in 5G!Pagoda.

3.6.8 5G-TRANSFORMER

5G-Transformer [80] architecture consists of three layers: Vertical Slicer (VS), Slice Orchestrator (SO) and Mobile Transport Platform (MTP). The VS is the entity that allows a vertical to describe a blueprint in order to deploy its service in the form of a Network Slice. The VS provides a web portal to that purpose and translates the blueprint into an NSD. The VS uses the SO API to deploy the Network Slice. The SO is an abstraction layer that exposes API to the VS to deploy a Network Slice on top of MTP. The latter is responsible for the computing and transport resources and constituted by several NFVI. In addition, to deploy a Network Slice, the SO may federate resources from another SO in case of multiple domain deployments.

5G-Transformer focused mainly on how to abstract the resources to vertical and did not dig on management process nor on using AI. However, VS is one of the key innovations of the project, as it includes many components that facilitate the deployment of vertical services.

3.6.9 5G-ESSENCE

5G ESSENCE [81] proposed a two-tier architecture with a distributed tier aimed at providing low latency services, and a second centralized tier for providing high processing power for compute-intensive network applications. The 5G ESSENCE Project provides a flexible and scalable platform to support different and evolving verticals aiming to create a neutral host market and reducing operational costs. The project exploited end-to-end (E2E) network slicing mechanisms to share the infrastructure among multiple operators/vertical industries and customize its capabilities on a per-tenant basis. The versatility and scalability of the architecture are achieved through high-performance virtualization techniques for data isolation, latency reduction, and resource efficiency, and by orchestrating lightweight virtual resources

enabling efficient Virtualized Network Function (VNF) placement and live migration. The coordination of the infrastructure is carried out by a central manager that is responsible for coordinating and supervising the use, the performance, and the delivery of both radio resources and services. The central manager is comprised of mainly three entities that are in charge of the interactions between the infrastructure and the tenants and handles related SLAs, slice management, service chain deployment, telemetry and data analytics functionality. The issue of distribution and scalability of management and orchestration is not addressed by the project.

3.6.10 SUPERFLUIDITY

Superfluidity [82] is focused on achieving what was referred to as a "superfluid" mobile network, i.e. the ability to instantiate and provision network functions and services on-the-fly, run them anywhere in the network (core, aggregation, edge), shift them transparently to different locations, and make them portable across multiple hardware (computing and networking) platforms. The architecture relies on the concept of Reusable Functional Blocks (RFBs), the elementary primitives or building blocks that result from the decomposition of network services and functions, and which can operate at different layers of the protocol stack. At least two 'levels' of RFBs were envisaged: 'higher level' functions, such as those in the scope of ETSI NFV, and which are chained together to build complex services, and 'lower level' primitives that are embedded in domain-specific network devices (such as OpenFlow-type actions for core network devices, signal processing blocks in SDR/C-RAN, etc.) and are covered by relevant standardisation bodies. The project uses centralized orchestration with the ManageIQ platform at the top. In opposition to MonB5G, the project has used classical optimization algorithms for service chain placement and load balancing.

3.7 The most popular management platforms

3.7.1 ONAP

The ONAP project is often regarded as one of the main open-source solutions capable of addressing the rising need for automation in the management and orchestration of telecommunication and cloud-based solutions. ONAP exploits SDN and NFV technologies to improve service deployments and provisioning and provides a unified framework for monitoring solutions that is able to inspect and verify end-to-end service level agreements (SLAs) and KPIs. ONAP also features a scalable and, to a certain extent, a distributed approach in the management of multi-site and multi-VIM resources. All the different blocks composing the architecture communicate through REST APIs, and this offers a high level of interoperability, favouring the integration of new blocks in the architecture as well as the evolution of the whole platform. The definition and enforcement of all the resource management policies pass through the so-called "closed control loop automation" process, provided by Data Collection, Analytics and Events (DCAE) module presented in Figure 12. DCAE is in charge of collecting and storing granular data in real-time streaming and batch mode from multiple underlying cloud infrastructures to monitor network services and level condition by means of performance surveillance and visualization tools. In large deployments, DCAE components can be geographically distributed in multiple sites and be hierarchically connected to each other. In this case, the so-called edge DCAE sites will be deployed physically closer to the network function and services to be monitored, providing faster communication latency and reducing the amount of data to be transmitted towards the central DCAE site. As a drawback, edge DCAE sites often present limited computational capacity and communication resources with respect to centralized DCAE deployments and are not as well connected towards other ONAP components.

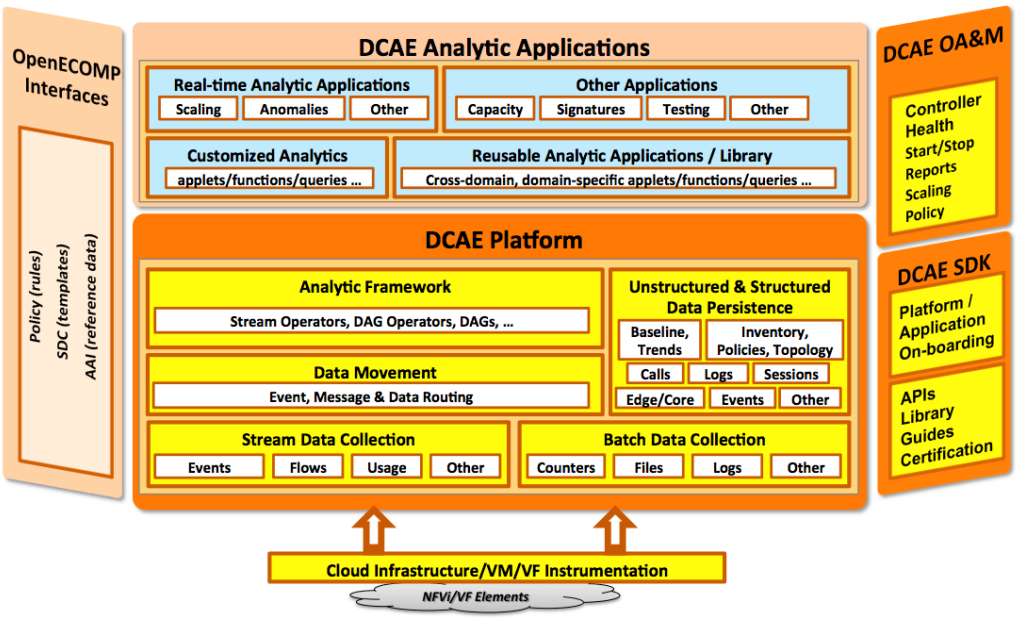

Figure 12. Data Collection, Analytics and Events (DCAE) Architecture [83]

Monitoring data are collected at different levels (hardware, infrastructure, Linux OS, Hypervisor and Kubelet) and aggregated according to on-demand policies. This is usually done by means of monitoring services based on open-source software like Prometheus, which is able to distribute its functionalities among geographically separated location sites. Synchronization messages and the results of local processing are exchanged among different sites of the distributed architecture by means of the Data Movement (also known as DMaaP) sub-module. In particular, DMaaP supports both file-based and message-based data transport among different system entities by means of publish & subscribe paradigm, where software clients can register to consume or find feeds for the data they require. The platform implements a Data Router (DR) and Message Router (MR) to manage the overall registration and communications exchanges, which are made available as networking services by means of RESTful HTTP APIs. These, together with state-of-the-art Kafka-based messaging schemes, enable efficient and horizontal scalability in the system. To reduce the amount of data exchange, data pre-processing is in place enforcing data analytics and compression functions at the edge of the network. At the same time, to increase the overall transport efficiency and introduce fault-tolerance across data centres, DMaaP performs data transformation and aggregation, finally enabling both batch and near real-time data consumption. Such distributed and hierarchical architecture is well aligned with the MonB5G point of view.

While this subsystem allows gathering a global view of performance and resource usage from the underlying infrastructure, it also provides a framework for the development of ad-hoc analytics applications. This is particularly helpful to overcome standard monitoring schemes and fault detection algorithms towards the definition of AI-based algorithms able to allow fast reconfiguration of multi-domain slices in response to unexpected network dynamics. In the 5G context, the monitoring of physical and virtual functionalities related to RAN will be particularly challenging due to highly variable radio channel conditions and mobility aspects traditionally alien to cloud-based ONAP use-cases. In these regards, novel and more dynamic monitoring solutions should be designed to ensure robust functional placement over the multi-domain resource pool as well as limit the traffic overhead generated by monitoring tasks.

For the purpose of ML integration with ONAP, the DCAE has been enhanced by a new software component, namely the Acumos-DCAE Adapter, which provides a client interface to receive ML models

from an Acumos catalogue to DCAE and is presented in Figure 13. This adapter will generate the required metadata artefacts (microservice component specification, data-format) to on-board them directly into the DCAE CLI-DB. The Acumos-DCAE Adapter transforms an Acumos ML model into an ONAP compatible DCAE microservice in order for it to run in an ONAP DCAE environment.

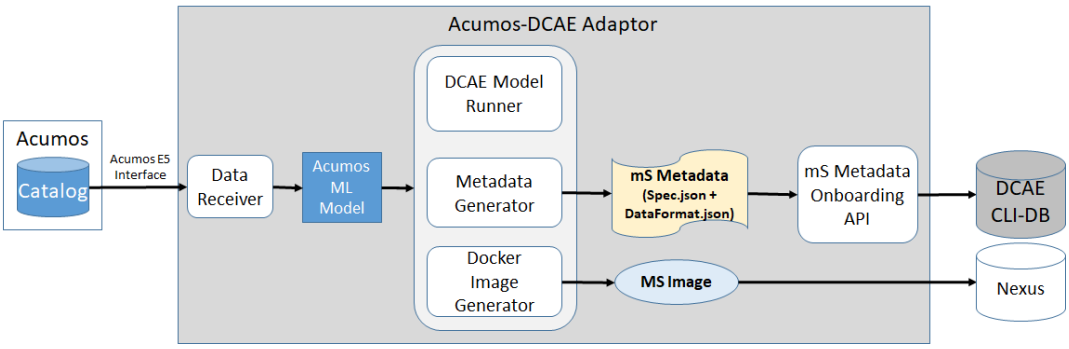

Figure 13. The architecture of the Acumos – DCAE adaptor

The ONAP platform, despite some decentralization efforts, is still very centralized in comparison to the MonB5G approach described in the next section. Moreover, in order to implement it, huge infrastructure resources have to be devoted.

3.7.2 OPEN SOURCE MANO

Open Source MANO (OSM) [84] is a collaborative open source project hosted by ETSI to develop an NFV Management and Orchestration (MANO) stack aligned with ETSI NFV Information Models and APIs. In terms of scope, it focuses on the Network Service Orchestrator (NSO) part of ETSI MANO NFVO (Figure 14). OSM has produced nine releases so far (each named after the respective number in capital letters), the last five introducing capabilities that may be of interest for MonB5G.

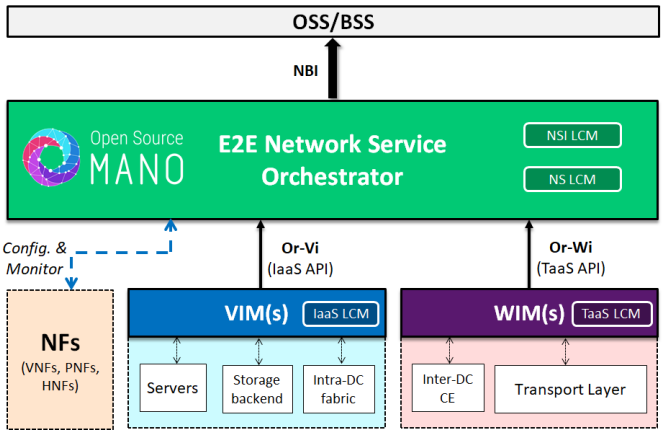

Figure 14. The overall look of OSM role in the management and orchestration ecosystem

The list below describes the features of the last five releases of OSM:

- Release FIVE introduced support of network slices, as well as extended monitoring capabilities, including VNF metrics collection. In the context of the network, slices support, OSM was extended to support lifecycle management of Network Slice Instances (NSI), i.e. compositions of several Network Services (NS) that can be treated as a single entity, which is described through Network Slice Template (NST) descriptors.

In the same context, OSM can operate in two modes of deployment and management, Full E2E Management (Integrated Modelling) and Standalone Management (Vanilla NFV/3GPP) as presented in Figure 15. The Full E2E approach has been reused in MonB5G.

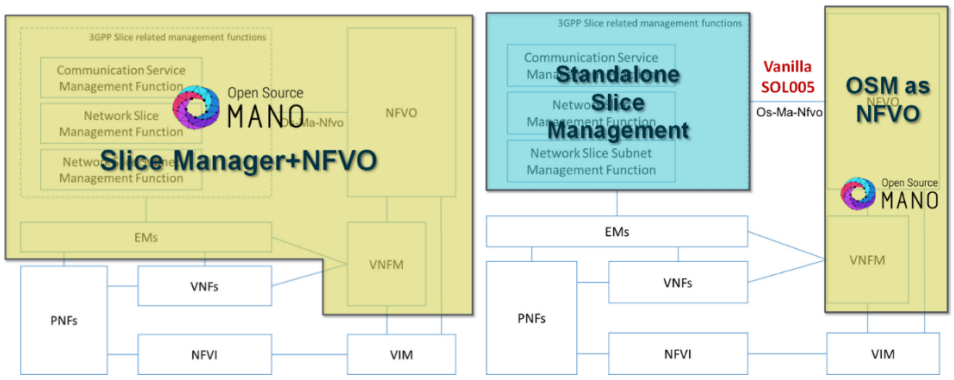

Figure 15. OSM modes of operation.

OSM supports metrics collection from both the infrastructure (VIM) and directly from VNFs. This was made possible by the new architecture of the OSM monitoring framework (OSM MON).

- Release SIX introduced a revamped version of the Service Assurance (SA) framework, which can control and react to a wider set of events and conditions in the context of running Network Services and Slices. This resulted in further improvements in OSM Performance Management (MON), as a continuation of the Release FIVE work [85].
- Release SEVEN introduced the ability to orchestrate Containerised Network Functions (CNFs), leveraging Kubernetes as the underlying infrastructure, allowing the combination within the same Network Service of cloud-native applications with “traditional” VNFs and PNFs, and all the required advanced networking required to build such complex E2E services. This added the “KDU”, to the already supported VDU & PDU.
- Release EIGHT is able to operate hybrid scenarios that include entire “Kubernetes Network Functions” (KNFs) through a uniform mechanism. The community leveraged this capability in the most recent Hackfest [86] to demonstrate deploying the Facebook Magma EPC stack [87] as a Helm Chart on top of Kubernetes. Release EIGHT also introduces “ultra-scalable” service assurance capabilities, including a new framework for the real-time gathering of metrics and alerts, and makes use of SNMP (Simple Network Management Protocol) monitoring. Important to note is that distributed monitoring tasks are now executed on Kubernetes clusters (thus, the “ultra-scalable” claim).
- Release NINE further evolved Kubernetes integration, making OSM installation on Kubernetes the default, deploying VCA (Juju) on the same Kubernetes cluster as the rest of OSM, adding support for the Helm 3 package format, as well as the capability to operate distributed applications in multiple Edge locations through distributed proxy charms. With regards to Performance Management, Release NINE introduced changes in the Tenant Manager and Dashboard components of MON to unify and align the user lifecycle and dashboard access in Grafana with the RBAC and multi-tenancy capabilities of OSM RBAC Controller.

During a relevant session of the OSM9 Hackfest [86] Ecosystem Day it has been presented how the MON and POL components of OSM could be integrated respectively with machine learning models and reward scoring functions for implementing intelligent closed-loop automation. The proposed concept is presented in Figure 16. The OSM community elaborated further on this concept/vision at the ETSI ENI ISG technical workshop “on the relation of ENI policy management to network intelligence" (Sept. 25, 2020).

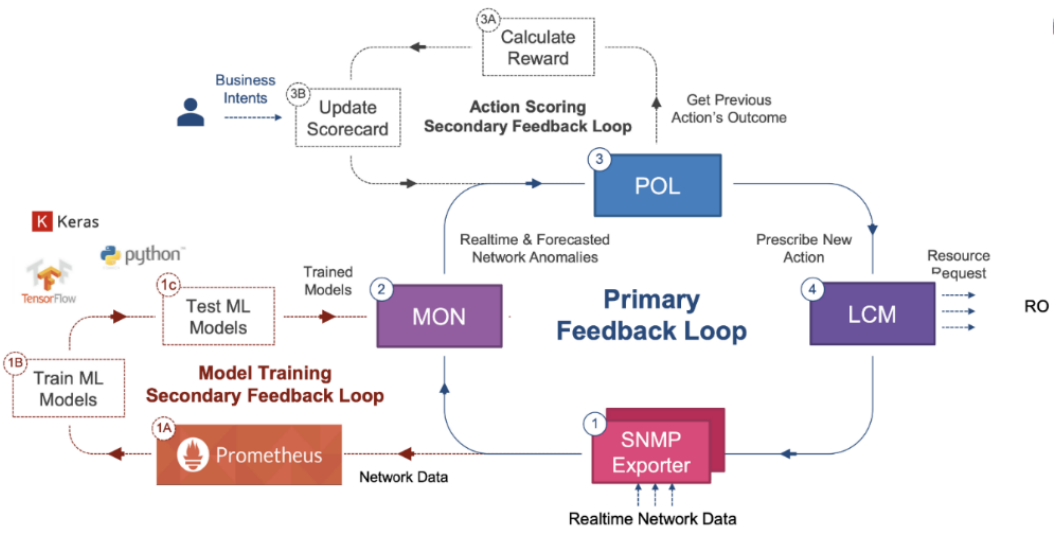

Figure 16. Implementing Closed-Loop Automation with the OSM Service Assurance Components

It has to be noted that this concept, as well as the new capabilities of OSM Release NINE, are of full interest to the MonB5G project. RIFT's RIFT.ware [88], Canonical's Charmed OSM [89] and Whitestack's WhiteNFV [90] are the three most well-known commercially supported OSM distributions.

4 MonB5G initial architecture

In the previous sections of this deliverable, an overview of network slicing related and general network and service management concepts as well as standardization efforts, have been described. As we have already outlined, a vast number of 5G radio access nodes deployed to satisfy ubiquitous coverage requirements together with increasing end-users' demand for low latency and high bandwidth envisioned for the 5G systems exacerbate the network management complexity with respect to previous mobile network generations, making human-centric managing solutions unfeasible and challenging the capabilities of other standard approaches. The network slicing concept further intensifies the problem. The overview that we made shows that there are not many activities related to the distribution of management and orchestration for network slicing. Also, the programmability of the network slicing management has been ignored. We have noticed that most activities follow the ETSI MANO/3GPP approach in which slice management is centralized and common for all slices. As each network slice instance can be treated as an independent and isolated network, the approach brings the complexity of the overall management to a totally new level.

The aim of the MonB5G architecture is to address these issues and provide a solution for the concurrent provisioning of high numbers of network slices as envisioned in 5G and beyond with maintaining the relatively simple structure of the management system. The main goal of the MonB5G approach is to achieve scalable and automated management of multiple network slices. In this section, the initial outline of the concept will be presented.

4.1 MonB5G architecture Principles

The MonB5G framework uses the management system decomposition that follows the ITU-T [91] and the MAPE (Monitor-Analyse-Plan-Execute) paradigm [36] as the basis. In our case, the MAPE concept is implemented in a distributed way by means of multiple AI-driven operations. Moreover, the runtime management of slices is distributed and programmable. We have also modified the MANO approach slightly by distributing some of the orchestration functions.

The key features of the proposed MonB5G framework are the following:

- **A strong separation of concerns**. In MonB5G, the OSS/BSS of each orchestration domain is focused on the lifecycle management (LCM) of slices and on resource management in this domain, but it is agnostic to slices (i.e. it is not involved in slice runtime management). In MonB5G each single- or cross-domain slice can be seen as a service with its own management platform (called embedded or In-Slice Management, ISM [79]), which is separated from the domain(s) OSS/BSS. The ISM is a part of the slice template and is responsible for the fault, configuration, accounting, performance, and security management (FCAPS) of a slice. That approach provides benefits like isolation of management planes of slices (feature not provided by ETSI NFV MANO [92] nor 3GPP). Using the approach, the deployment of a slice requires marginal modification of OSS/BSS in order to support each slice management. In the case of multi-domains slices, a special inter-domain component (ISM) is added to the end-to-end slice template. It interacts with domain-level ISMs to achieve the end-to-end management of the slice.
- **Distribution of management operations**. The management operations are AI-driven and pursue different goals. The embedded management concerns nodes, slices, orchestration domains and end-to-end slice. Using distributed AI allows for local management information processing, thus reducing the exchange of management information between entities. The AI-driven approach also enables the use of intent-based interfaces.
- **Hierarchical, end-to-end slice orchestration.** In MonB5G, there are multiple domain-level orchestrators (they can be domain-specific) and one master orchestrator. This implies the use of domain-specific slice templates. The use of multiple orchestrators contributes to orchestration scalability.
- **ISM capability of orchestration.** The ISM of each slice may act as a service orchestrator, i.e. it may use the Os-Ma-nfvo-like interface [92] to request slice template modifications, and such action is no longer

executed by the domain-level OSS/BSS. The request is typically based on the ISM analysis, and the action is related to slice topology update.

- **Scalable and programmable slice management**. Since the ISM is part of a slice, and it is implemented as a set of VNFs, the resource scaling mechanism can contribute to ISM (i.e. slice management) performance. Moreover, all the FCAPS functionalities can be dynamically deployed or updated during slice lifetime using the orchestration capabilities of ISM.
- **Enhanced security of slices.** The use of the ISM concepts provides isolation of the management spaces of different slices, therefore, contributing to enhanced slices security. It also limits information exchange between slices and OSS/BSS of each orchestration domain.
- **Support for Management as a Service (MaaS).** MonB5G allows the creation of a "management slice" that can be used for runtime management of multiple slice instances of the same template. In such a case, a new business player, called Slice Management Provider, can be involved in slice management.
- **Programmable, energy-aware infrastructure management.** The infrastructure management system proposed by MonB5G can use the architecture to deploy its services dynamically, in a similar way in which slices are deployed. The framework provides extensions to include energy-aware operations on infrastructure resources by the use of modified, energy consumption aware orchestration. For that purpose, the interface between the orchestrator and the infrastructure is provided.

The abovementioned features are in line with several ETSI ZSM requirements [65] – the list of the essential requirements of ZSM that are satisfied by the MonB5G management and orchestration framework is provided in Section 5.

In MonB5G, the well-known autonomic network management provided by a feedback-loop based is used. The solution faces problems related to response times (associated with the round-trip time between network elements) and system stability (the managed system is a nonlinear one. Therefore, the feedback-based control may lead to instabilities and chaotic behaviour). To this end, we propose a hierarchical control scheme with *fast local* control loops and *slow wider-scope* ones. Leveraging time-scale decomposition at different levels of the proposed system, we achieve to limit the interference among different feedback-based decisions. We also assume a rich multi-objective environment, where various goals (e.g. energy consumption, statistical multiplexing, slice isolation, etc. vs. performance) may have different weights, and the proposed algorithms should be able to automatically learn to prioritize accordingly. We have also introduced the architecture components that are responsible for providing the feedback loop control stability evaluation and restoring.

MonB5G introduces logical entities for monitoring, analytics, and decision making that are decomposed into **distributed, interacting components** executed at various levels: at the OSS/BBS level, inside the virtualized infrastructure, and embedded in slices. By local data processing and decisions, our design aims to (a) minimize the exchange of (big) data between components to keep management scalable, and (b) significantly reduce the reaction time of data-driven management decisions that could be handled locally. **Reducing the monitoring load** is critical for **carrier-grade performance** for a sliced beyond 5G network, a challenge that has not yet been adequately addressed. This approach requires, however, proper coordination of the 'local' management subsystems. The **design philosophy** of MonB5G is to provide hierarchical, feedback-loop-based control for fault, configuration, accounting, performance, and security (FCAPS) management, and slice orchestration, featuring **different control loops with different scopes, goals, and timescales**, at the following levels:

- **Global OSS/BSS level**. At this level resides a centralized component with full-scope slice management and orchestration decision capabilities and takes global actions for network-wide, cross-slice, and cross-domain optimizations. The functions at this level are implementation-dependent – in more distributed approaches, the OSS/BSS becomes simpler.
- **Technological/Orchestration Domain level:** Each technological domain (e.g. cloud infrastructure, edge, RAN, etc.) may operate its own instances of monitoring, analysing and decision functions. Undertaken AI-driven decisions aim at managing specific domain resources in the presence of

coexisting slices, as well as carrying out optimizations to save on energy consumption without degradation of slices performance.

- **Slice level:** For each slice (called further Slice Functional Layer – SFL), we have proposed embedded AI-driven management, and advanced implementation of ISM, called Slice MonB5G Layer – SML. It can be a part of a slice template, or it can be provided in the form of a MaaS. The Slice MonB5G Layer is logically decomposed into Monitoring System Sublayer (MSS), Analytic Engines Sublayer (AES) and the Decision Engines Sublayer (DES). Such decomposition enables the independent design of components of each of the mentioned sublayers that offer their services to other sublayers. All the sublayers may have AI-driven behaviour.
- **Node (VNF/PNF/CNF) level:** At this level, the control loops are implemented as part of modified Element Manager (EM), called Embedded Element Manager (EEM) that belong to SFL. The EEM can take some node/function-focused, AI-driven decisions based on node-level monitoring. The EEMs interact with SML as slaves, e.g. SML functions may leverage EEM endpoints for gathering data or actuation. The EEM information processing contributes to the reduction of the monitoring traffic and may also provide mechanisms for efficient and secure actuation.

The proposed framework assumes that the Infrastructure may also need programmable management. To that end, we have proposed a separate infrastructure OSS/BSS (called **Infrastructure Domain Manager** – **IDM**) that manages the supporting infrastructure. Domain Manager and Orchestrators (DMO) on the request of IDM can dynamically deploy management functions that cooperate with IDM to achieve efficient infrastructure management in terms of energy-saving and slice cost.

4.2 Architecture outline

The MonB5G architecture is composed of static and dynamically deployed components. Altogether they provide support for operations related to slicing orchestration, fault management (self-healing), self-configuration, performance optimization (including energy-saving) and security-related operations of slices. The AI-driven In-Slice Management approach provides separations of management functions of each slice and gives the ability to manage slice(s) to the slice tenant in a simplified way. Moreover, the MonB5G management services can be deployed dynamically; therefore, they can also be orchestrated in a similar way to slices.

The MonB5G framework is composed of three layers:

- **Business Layer.** The layer consists of the business entities that operate the framework, provide slice management services to slice tenants, or own a slice (slice tenants).
- **Management and Orchestration Layer.** The layer consists of the core functions of the framework responsible for management and orchestration of slices, slice LCM and exposure of management interfaces to specific business entities.
- **Infrastructure Layer.** This layer consists of the infrastructure, infrastructure providers and functions enabling communication with Management and Orchestration Layer and enabling optimization of usage of infrastructural resources.

The forthcoming sections provide a detailed description of each static and dynamic component of the framework belonging to Management and Orchestration and Infrastructure Layers. The description of business entities and their interactions is presented from a high-level perspective. A more detailed description can be found in [93].

4.3 Static components of the architecture

The static components of the architecture together with the business entities are presented in Figure 17, where for the sake of clearness, slices are omitted. In the following section, a description of the static components and their main functionalities will be provided.

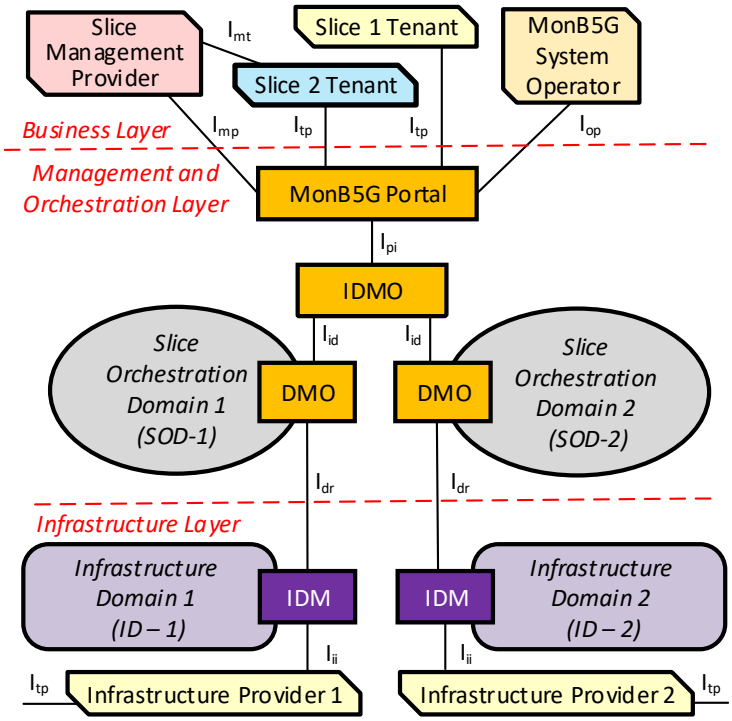

Figure 17. Static components of the MonB5G architecture

4.3.1 MONB5G PORTAL

The MonB5G portal is used by Slice Tenants, Slice Management Providers and Infrastructure Providers to request operations regarding slice LCM, i.e. slice deployment, slice modification and slice termination. It also exposes the capabilities offered by the MonB5G framework (available slice templates, etc.) and partakes in negotiations related to the business dimension of the contract. The portal is also used to pass all the accounting and billing-related information. The internal structure of MonB5G Portal is presented in Figure 18.

The MonB5G Portal is composed of the following components:

- Access Management – entity responsible for policy enforcement regarding users' access to MonB5G framework features, policy management and users' authorization.
- System Health Monitoring – a component responsible for providing real-time high-level monitoring data showing the current state of the network for the MonB5G Operator. In case of critical failures or instabilities of the system, the MonB5G Operator, based on the accumulated monitoring data, can bring the framework back to stable conditions manually.
- MonB5G Subscribers Database – the database containing information of all entities having rights to access functions provided by the MonB5G system.
- IDMO Connector – the component responsible for communication with Inter-Domain Manager and Orchestrator (IDMO) described in section 4.3.2. The exchanged information includes, among others, slice LCM-related requests, contract negotiation, and high-level system monitoring data.

MonB5G Portal exposes three northbound interfaces that expose the MonB5G framework capabilities to MonB5G System Operator (I_{op}), Slice Tenants (I_{tp}) and Slice Management Providers (I_{mp}).

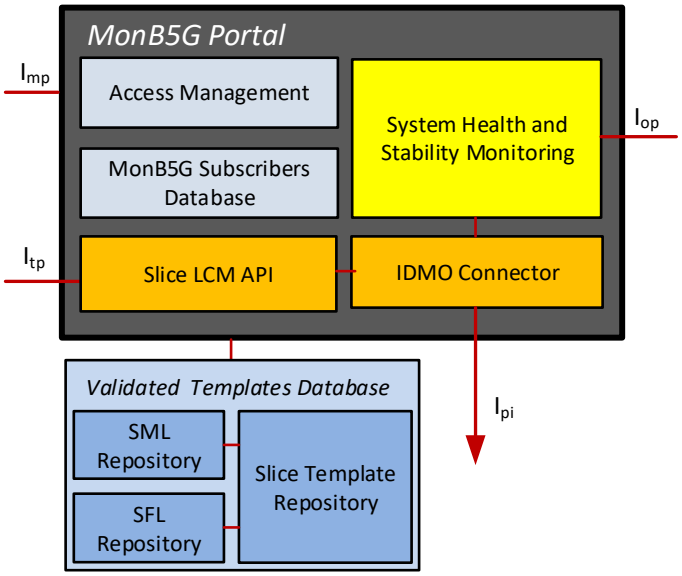

Figure 18. The internal structure of MonB5G Portal

The MonB5G Operator can interact with the MonB5G system via the MonB5G portal using I_{op} management interface. It provides monitoring capabilities as well as configuration capabilities of the overall framework. The MonB5G System Operator is responsible for controlling the health, security and stability of the operation of the whole network. He can access the highest level KPIs that reflect the quality of operation and act accordingly to the obtained measures. All the operations are done via the I_{op} interface. The I_{tp} interface, which will be typically implemented as a web-based interface, enables Slice Tenants to select and request Slice related operations. It can also be used by the Infrastructure Providers to ask for orchestration of infrastructure-oriented management functions. The procedure is performed in a similar way as slice-related requests of Slice Tenants.

Slice Management Providers may use MaaS platform, called MonB5G Layer as a Service (MLaaS), to offer management of multiple instances of slices based on the same template (as the slice runtime management is slice-specific). LCM of MLaaS is done via the I_{mp} interface. This approach will be described in details in section 4.4.4.

To perform negotiations related to the business dimension of the contracts, MonB5G Portal interacts with IDMO via the southbound, I_{pi} interface. The exchanged information concerns aspects like availability of resources, existing policies, the resource demand and other data that enables allocation of a certain amount of resources to the requester. After the successful establishment of the contract, the I_{pi} interface is used for LCM of negotiated slices.

4.3.2 IDMO

The Inter-Domain Manager and Orchestrator (IDMO) is at the heart of the system. This entity plays a crucial role in slice preparation and deployment phases by negotiation of deployment policy with a slice requester (Slice Tenants, Slice Management Providers or Infrastructure Provides). A schematic picture of IDMO with its internal components is presented in Figure 19.

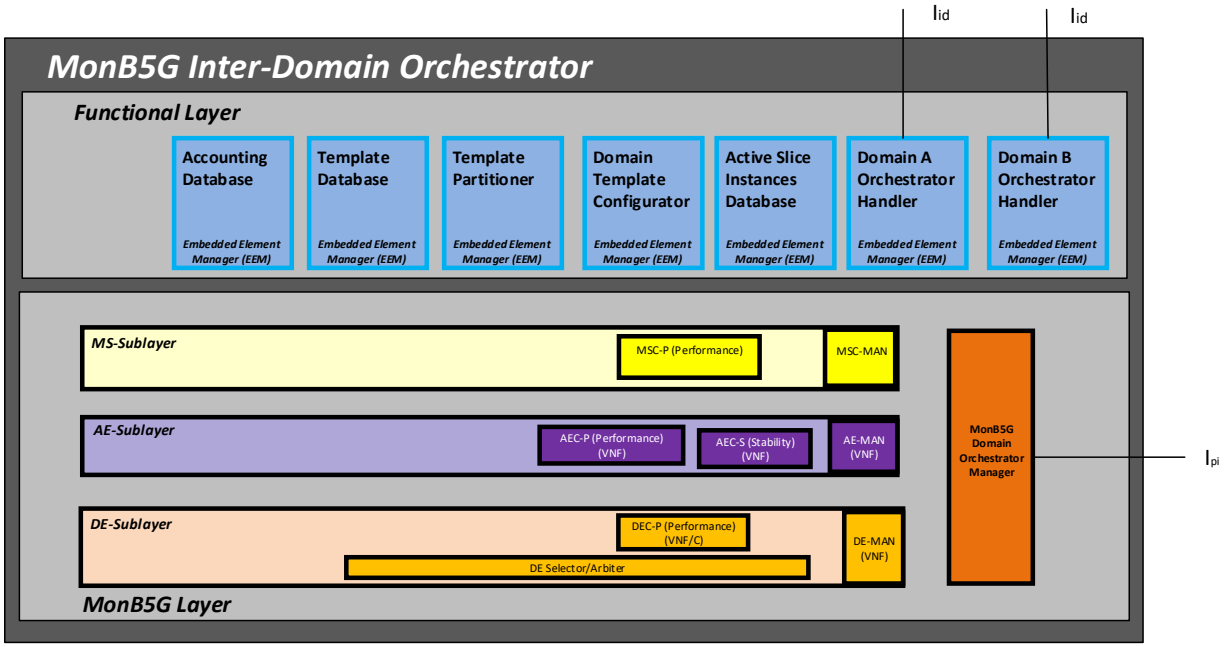

Figure 19. IDMO internal structure

The IDMO structure is decomposed into Functional and MonB5G Layers. The MonB5G Layer (management of IDMO) is AI-driven and uses sets of Monitoring System (MS)/Analytic Engine (AE)/Decision Engine (DE)/Actuator (ACT) as well as other components of the management architecture. The approach is described in detail in section 4.4.2.

The Functional Layer of IDMO provides capabilities of storage of data related to accounting, available slice templates and currently active slice instances. It also enables the configuration of domain templates as well as template partitioning described further on. IDMO has knowledge about all deployed domains and about the status of resources in all the infrastructure domains. It also collects information coming from DMOs, slices or infrastructure.

IDMO interacts with DMOs (see further) via I_{id} interface by using domain handlers to deploy the end-to-end slices based on the information obtained from DMOs. It is responsible for the modification of the end-to-end slice template before its deployment according to the negotiated contract. The modified slice template includes mechanisms that were added to slice template by IDMO for slice stitching in order to obtain the end-to-end slice as well as proper modification of the end-to-end slice management plane (correlation of events and KPIs from different domains that are used for slice deployments). It can be seen as an end-to-end orchestrator (umbrella orchestrator, according to [94]).

If the infrastructure has multiple owners, IDMO may decide how to split the end-to-end slice template dynamically to a new one, which supports inter-domain interaction of slice components located in different orchestration domains. The split may be shaped by various factors, e.g. price, performance or energy efficiency. For that purpose, the IDMO is aware of all infrastructure domains involved in the system and the status of their resources. IDMO also keeps a permanent Accounting Database, as the ISM component is not a permanent one. IDMO may also interact with the IDM (via DMO) in order to decide how to deploy slice instances, considering the price, performance or other important factors such as energy efficiency. Eventually, it can perform a dynamic split of slice template and provide a new end-to-end slice template that includes components responsible for the interaction of E2E slice components deployed in different orchestration domains. In the “legacy” implementation of network slice orchestration and management, the IDMO may play a role of CSMF and NSMF (see section 3.2.2) for MonB5G compliant slice templates IDMO is an orchestration part of NSMF only.

4.3.3 DMO

Domain Manager and Orchestrator (DMO) is responsible for orchestration and management of each of Slice Orchestration Domain (SOD) slices. The internals of DMO are presented in Figure 20. The DMO can be seen as a combination of resource-oriented OSS/BSS and an orchestrator (a MANO orchestrator is shown in the picture – in other technological domains other orchestrators can be used). The behaviour of both components is optimized by the use of AI. The OSS/BSS is tailored to cope with domain-specific management. It is focused on slice lifecycle management (including slice admission control) and resource management (FCAPS of resources). The OSS/BSS of DMO can be used for all external interfaces of DMO. It has to be noted that IDMO does not interact directly with the orchestrator but with OSS/BSS of each SOD. Therefore, the IDM-IDMO interface can be defined in a similar way for different orchestration technologies. The DMO is focused on SOD operations concerning resources (resource allocation to slices, slice LCM, resources FCAPS) and is agnostic to slices, i.e. it is not involved in slice run-time management, including initial slice configuration. Therefore, DMO generally deals with the software dimension of slices (LCM, resource scaling) or allocation of PNFs to slices. In contrast, the run-time management is handled by the management components embedded in slices (i.e. ISM). Similarly to IDMO, all DMO operations are AI-driven. Therefore, the internal structure of DMO is also composed of Functional and MonB5G Layers. The operations related to resource management as well as the exchange of infrastructure-related data (e.g. about energy consumption) are done via I_{dr} interface.

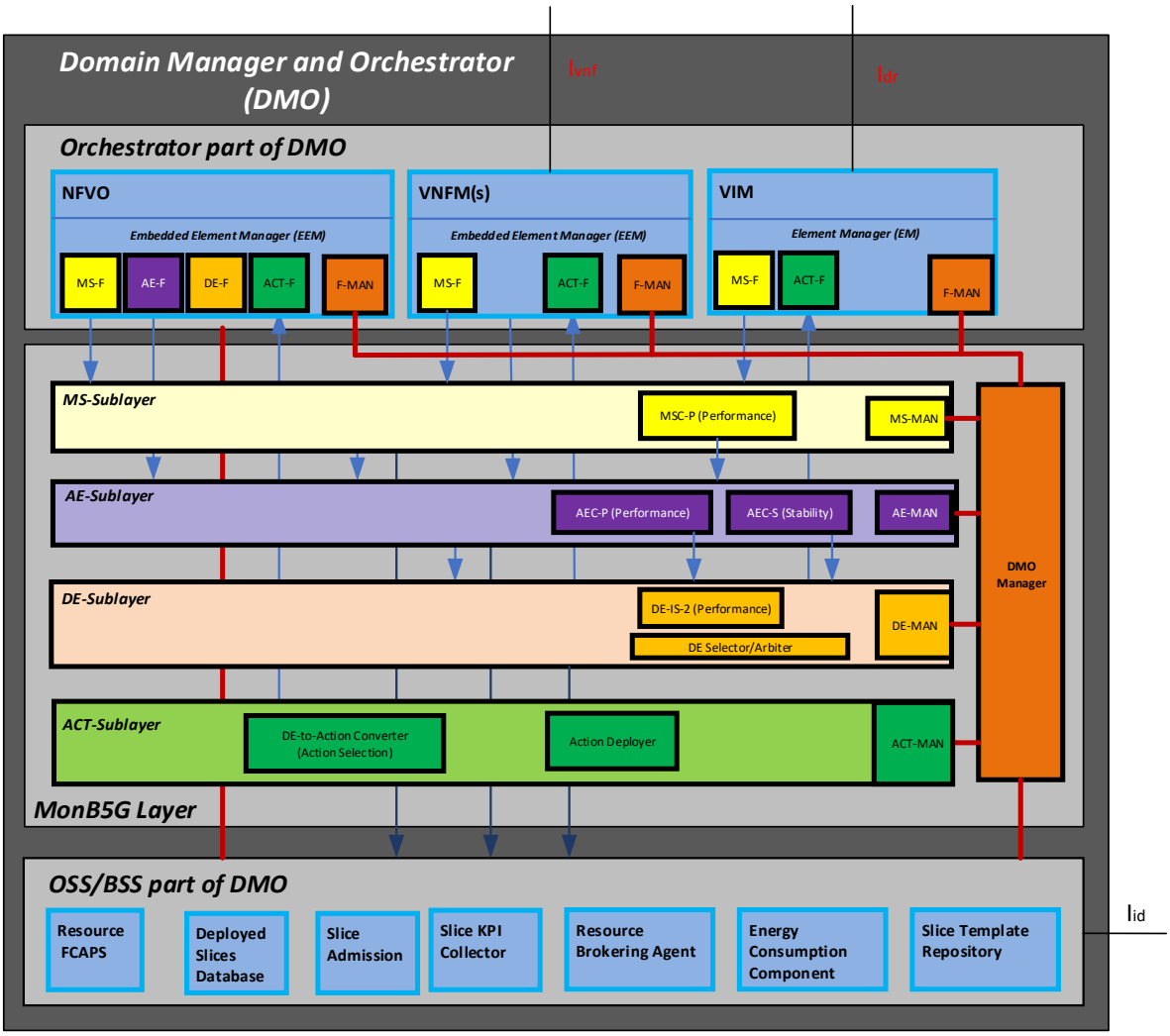

Figure 20. The internal architecture of the Domain Manager and its interactions (MANO case).

In contrast to the 3GPP approach in which management of all slices is part of the OSS/BSS, the DMO is not involved in individual slice management. In MonB5G, it has been decided to keep the orchestration part agnostic to slices; therefore, the runtime management of slices is performed by different components than the orchestrator. The DMO keeps the repository of slice templates that can be deployed in its domain.

The IDMO-DMO-IDM interaction can be used by IDMO for resource brokering decisions to split the slice and deploy it on the infrastructure belonging to multiple owners.

In the "legacy" implementation of network slice orchestration and management, the DMO may play a role of NSSMF (see section 3.2.2). For MonB5G compliant slices, it is the orchestration part of NSSMF only, not involved in slice run-time management.

4.3.4 IDM

The proposed framework assumes that the Infrastructure may also need management. To that end, we have proposed a separate management entity called **Infrastructure Domain Manager** – **IDM** presented in Figure 21. The IDM provides the overall management of the Infrastructure. Its interface to DMO allows for the allocation of resources (NFVI agent), exchange of the information related to the energy consumption of resources, exchange of the information related to the cost of resources that can be used by IDMO for resource brokering. The framework enables programmable infrastructure management. The DMO orchestrator can dynamically deploy management functions that cooperate with IDM to achieve infrastructure management. The IDM has an interface to the Infrastructure Provider, who can use the MonB5G portal asking for the deployment of additional infrastructure management functions, called IOMFs (see further). The functions are orchestrated in a similar way to slices, and LCM requests are sent by the Infrastructure Provider to the MonB5G Portal.

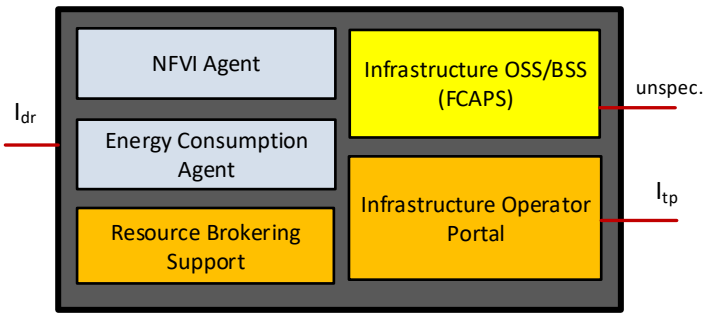

Figure 21. Internal Structure of IDM (an example)

The details of the IDM are out of the scope of the MonB5G framework; however, in case of MANO, the IDM should include the NFVI Agent, the Energy Consumption Agent, Resource Brokering Support, Infrastructure Operator Portal and Infrastructure oriented OSS/BSS.

4.4 Dynamic components of the architecture

The dynamic components of the architecture are slices that are defined in a different way than NGMN has defined them. In MonB5G, a slice is a set of functions that implement a specific goal – it does not have to be a network. It can be a set of functions that implement a specific goal, for example, network management, implementation of services or accelerators that support certain operations of multiple slices. Interactions between such slices and classical slices can be implemented using the PaaS approach. The approach is sometimes called vertical stitching of slices in opposition to the horizontal stitches of single-domain slices in the case of a multi-domain slice. The ways in which the MonB5G framework is using PaaS and the benefits of the approach will be described later.

The MonB5G follows the ISM concept in which the slice management plane is a part of the slice template. Therefore, it is (with one exception that will be described later) implemented as other components of the slices, for example, a set of VNFs. Such an approach leads to the creation of self-managed slices and reduced information exchange between slices and external management components of the architecture (i.e. DMO and IDMO). The usage of PaaS and the need for the creation of slices that span multiple domains leads to different options of slices deployment, as presented in Figure 22.

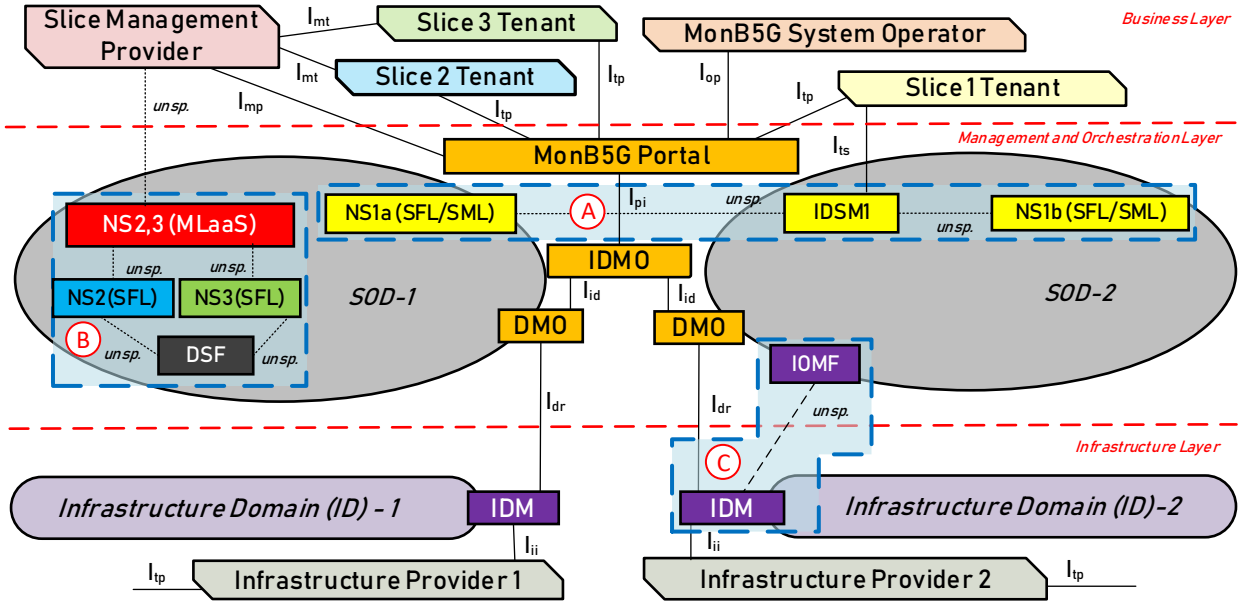

Figure 22. The overall MonB5G management and orchestration framework

The Option A of Figure 22 concerns the deployment of a self-managed multidomain slice. Such a type of slice requires a special component that is responsible for end-to-end slice management – it is worth noting that this component is implemented as a part of the end-to-end-slice template, not as a part of DMO. Option B shows the deployment of slices that use the PaaS approach, i.e. shared functions that implement the Management as a Service (MaaS) paradigm. Another set of shared functions, DSF, is in this option exploited by the functional part of the slice. Option C shows infrastructure management-oriented and orchestrated by DMO functions that are created in a similar way to slices, on the request of the Infrastructure Provider. The components that are deployed within each option are described in detail in the forthcoming sections.

4.4.1 MONB5G SLICE STRUCTURE AND FUNCTIONS

The generic structure of the MonB5G slice is presented in Figure 23. In MonB5G slice structure, two separate layers can be distinguished – the slice management part called Slice MonB5G Layer (SML) and slice main part called the Slice Functional Layer (SFL).

The SML performs FCAPS at the slice-level and can be considered as an embedded slice-level *OSS/BSS*, with interfaces to the Element Managers (EMs) of the slice's VNFs/PNFs or CNFs, and to the DMO. The SML is a part of the slice template or is deployed independently using PaaS/MaaS paradigm. In such a case, the SML is implemented as an independent slice that is able to manage multiple instances of SLFs of the same template.

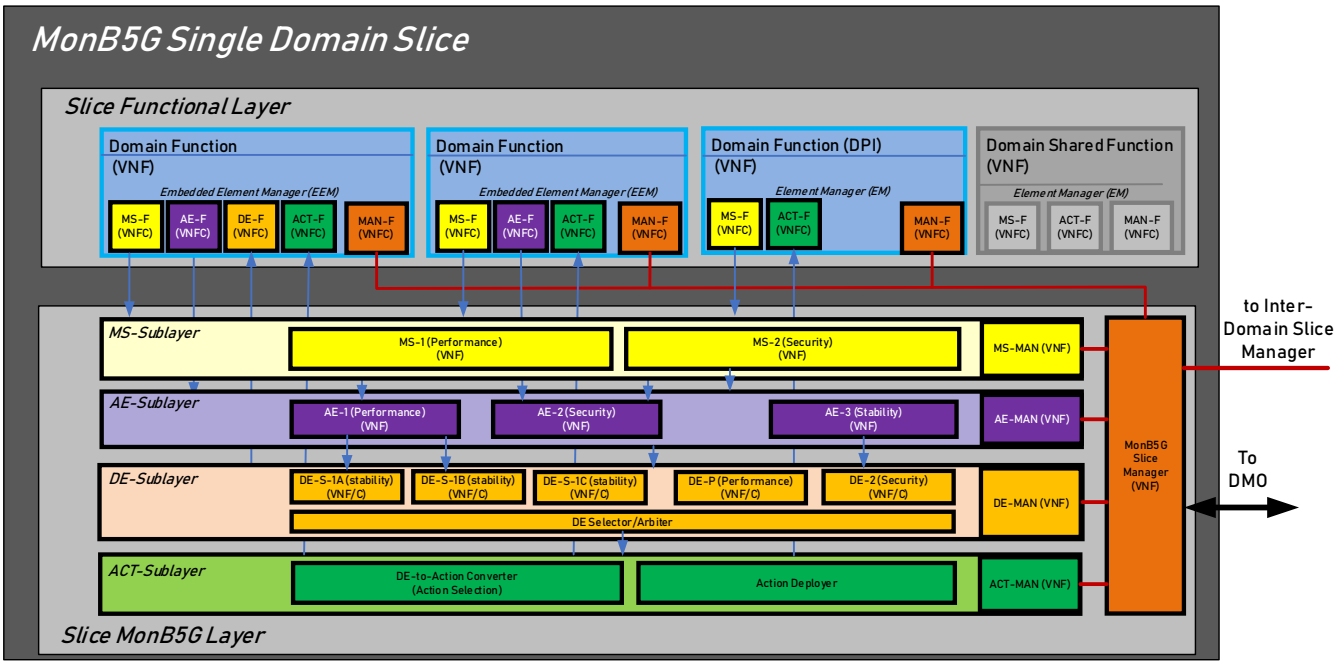

Figure 23. A generic structure of MonB5G slice

4.4.1.1 SLICE FUNCTIONAL LAYER

The Slice Functional Layer (SFL) contains a set of virtual functions that form the network slice to be deployed. The SFL part is composed of virtual functions that are dedicated solely to a slice (they are included in the slice template). However, SFL can also use functions that are shared functions available in a SOD. Such functions may be used by all or some slices. The functions are called Domain Shared Functions (DSFs), which can be implemented as PNFs/VNFs or CNFs. In fact, this is a PaaS approach used for SFL. The use of DSFs provides a reduced footprint of the deployed slices improving that way also the slices deployment time. DSFs are grouped (i.e. form a slice) for the purpose of their management. They are managed by the DMO.

According to the ITU-T and ETSI MANO model, each of the functions (i.e. VNF) should have an element manager that typically interacts with OSS/BSS. In the MonB5G case, the EM is replaced by MAPE-based Embedded Element Managers (EEMs) that are implemented as components of the functional entities (VNF-C), as presented in Figure 24.

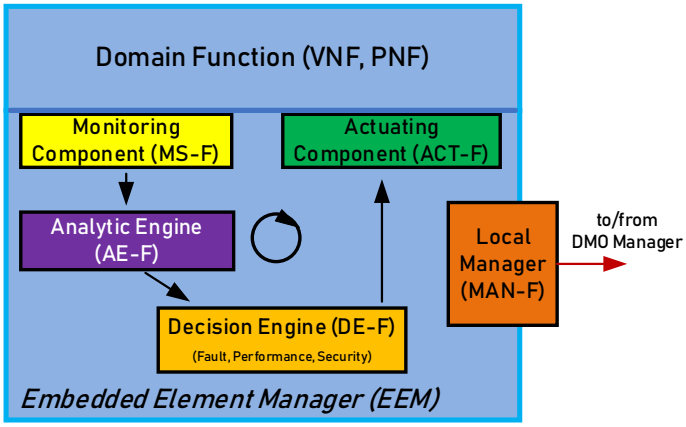

Figure 24. Typical interactions between EEM components

The EEM is internally split into components responsible for its VNF monitoring (MS-F), anomaly detection (AE-F), decision engine (DE-F) and actuating component (ACT-F). The MS-F is used for the monitoring of the function/node behaviour, the AE-F is looking for potential anomalies in the function/node behaviour, the DE-F is taking decisions concerning function/node reconfiguration, and finally, the ACT-F component converts the DE-F decision into a set of low-level commands. All the entities cooperate together to achieve MAPE behaviour at the node level. The EEM also includes the management component (MAN-F) that is used for the external management of the component, in order to validate function/node MAPE decisions or to override them. For the backward compatibility, it is assumed that some network functions/nodes may not have EEM; therefore, the EM in such case has to be used.

The usage of EEM reduces the management related traffic significantly and introduces self-managed functions/nodes in the management architecture hierarchy. The usage of AI for AE-F or DE-F is dependent on implementation; the EEM footprint has, however, to be kept small. The monitoring information, pre-processed by EEM, is fed to the SML part of the slice. Depending on the specific of SOD, the SOD functions/nodes allocated to slices can use different orchestration approaches. In a non-virtual environment, the EEMs (i.e. RAN slicing in 3GPP Release 15) have to be implemented as a part of SML in a virtual environment. In general, EEMs are the links between the SFL and SML parts of a slice. It is expected that the interaction can use a message bus or APIs.

4.4.2 SLICE MANAGEMENT LAYER

The SML is an implementation of the ISM concept having in mind the AI-based MAPE management. The SML is, therefore, split into (see Figure 23): Monitoring System Sublayer (MS-S), Analytic Engines Sublayer (AE-S), Decision Engines Sublayer (DE-S) and the Actuating Functions Sublayer (ACT-S). Each component of the MS-S, AE-S, DE-S and ACT-S sublayer can be managed from the SM level by individual managers MS-MAN, AE-MAN, DE-MAN, ACT-MAN, respectively. The SML, depending on slice type (multi-or single domain), may play the role of CSMF, NSMF and NSSMF (single domain slice) or the run-time part of NSSMF in the case of multi-domain slices (see section 3.2.2 for details on NSMF, NSSMF and CSMF).

4.4.2.1 MS SUBLAYER

It is assumed that **MS Sublayer** provides generic, reusable monitoring that is consumed by AEs, DEs and other entities of the SML, as presented in Figure 25.

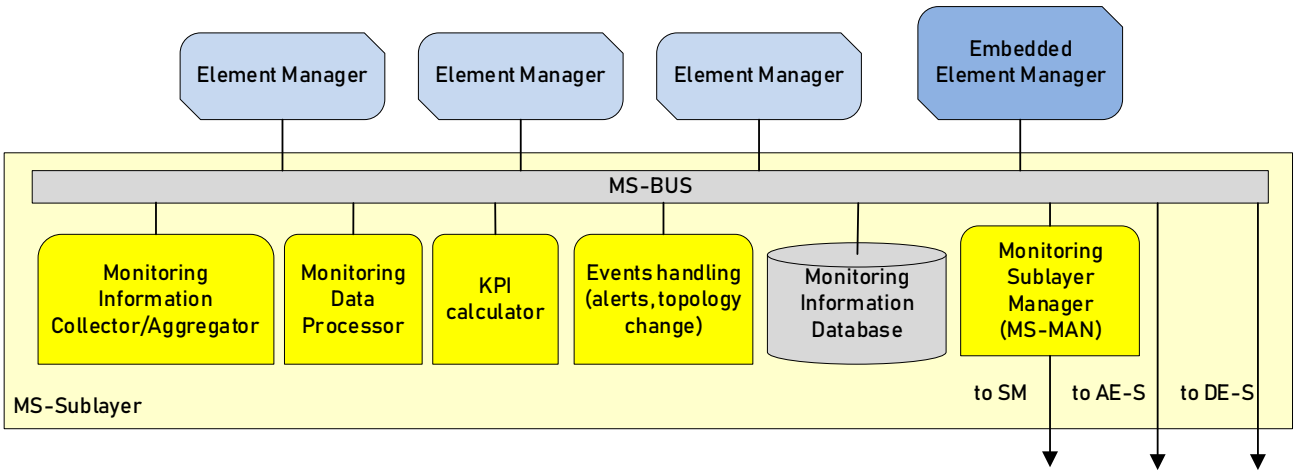

Figure 25. Monitoring System Sublayer internal components

The MS should contain:

- Monitoring Information Collector/Aggregator which interacts with the EEMs of SML;
- Monitoring Information Database – a database in which collected monitoring data are stored in a raw and pre-processed format;
- Monitoring Information Processor – an entity that is responsible for filtering, interpolation and prediction of the monitoring data;
- Slice KPI calculator – an entity that is used for the calculation and prediction of slice specific KPIs
- Monitoring Sublayer Manager is an entity that allows remote configuration of MS sublayer operations.

The output of the MS sublayer is accessible to other components of SML via a message bus. Therefore the publish/subscribe paradigm is in use. The MS has to interact with EEMs that are VNF specific, but most of the MS operations are generic. Therefore, many of the internal components of MS can be reused for multiple slice templates. The protocols for efficient communication between the EEMs and the MS and the adaptability of the monitoring (adaptive sample rate or resolution, using gossiping protocols, etc.) are out of the scope of the deliverable. However, the architecture allows for the use of such mechanisms.

4.4.2.2 AE SUBLAYER

The **AE Sublayer** includes a set of AEs and the AE Sublayer Manager that is used to configure the AEs remotely, as depicted in Figure 26.

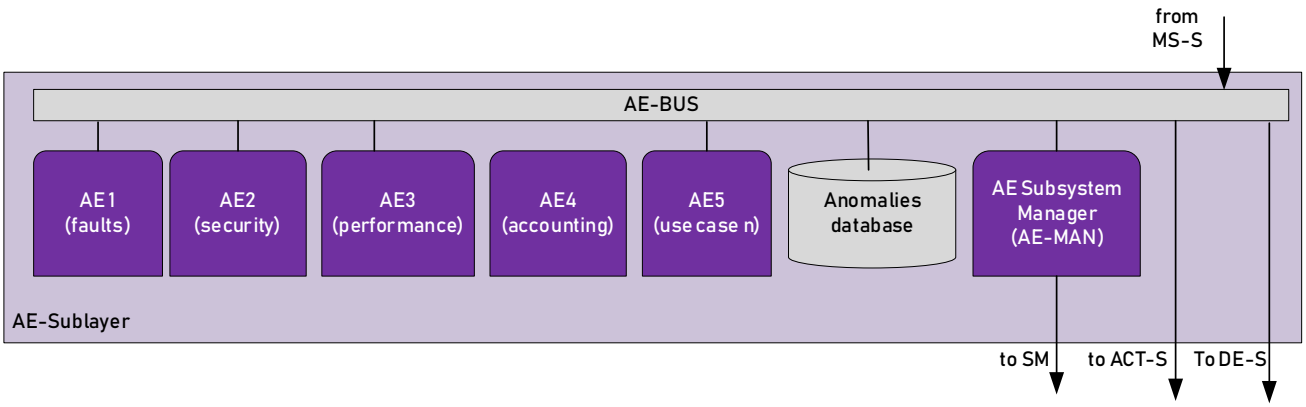

Figure 26. Analytic Engine Sublayer internal components

Each of the AEs has a specific goal, i.e. it may analyse the monitoring traffic for a specific security threat, fault or performance degradation. The internal specification of AE is algorithm dependent and cannot be provided *a priori*; it is however, possible to create a library of AEs that with a relatively small adaptation, can be used for different slice templates. In general, it is assumed that between AE and DE there is a one-to-one mapping, but the architecture allows to use of multiple AEs for the same DE. It is worth mentioning the MS sublayer provides some kind of abstraction of the monitored data that positively contributes to the reusability of AEs.

4.4.2.3 DE SUBLAYER

The DEs are the entities that are responsible for the reconfiguration of SFL or SML. It is assumed that the DE sublayer is composed of multiple DEs, as presented in Figure 27. Each of them is trying to reach a specific goal regarding, for example, performance optimization according to KPIs, fault handling, security or enforcing energy-efficient operations of the slice. The input to the DE sublayer is the output of AE and ME sublayers. Each of the FCAPS functions may require multiple DEs. The existence of multiple "selfish" DEs implies the need for their decisions arbitrage. For this problem, the DE Selector/Arbiter component is implemented in the DE sublayer. The component can be AI-driven as one of the implementation options is the possibility of the implementation of several DEs that use different algorithms for the same goal. In

such a case, the Coordinator/Arbiter is creating a ranking of DEs. As the stability of the feedback-loop based management can be an issue, a special entity called Stability Observer is introduced in the DE sublayer. It is used in order to avoid the chaotic behaviour of the system or the ping-pong effect. The Stability Observer uses Reconfiguration History Database to restore to the last stable configuration. This database stores recent reconfiguration decisions together with the input values that were used by DEs to take the decision reconfiguration. The DE sublayer, as other sublayers of SML, have a DE Sublayer Manager that can be used for the change of the configuration of its components or their policies.

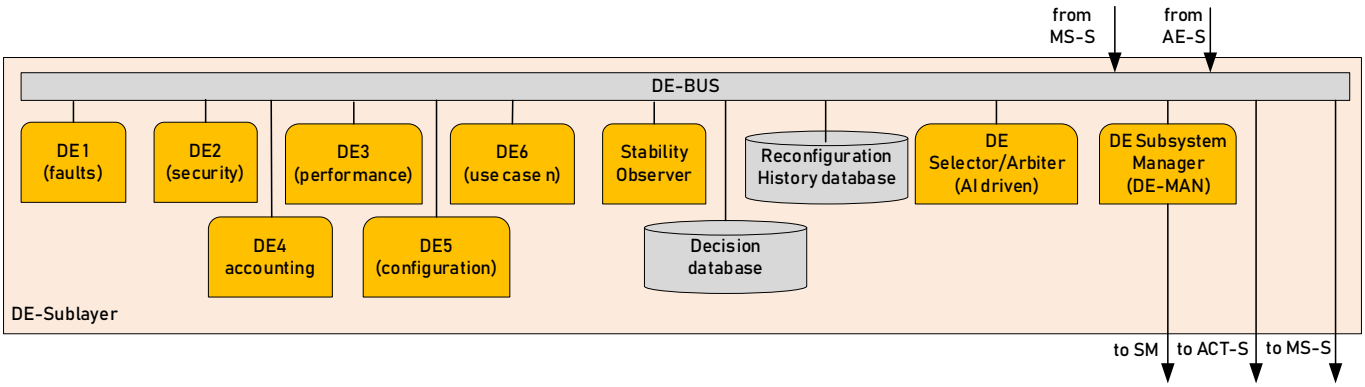

Figure 27. Decision Engine Sublayer Internal components

The DE decision can be used for the reconfiguration of SFL or SML. In both cases, there are three possible reconfiguration operation types:

- Reconfiguration of functions/nodes of SFL/SML;
- Change of resource allocation to SFL/SML components (including transport). Dependent on the implementation, it can be done directly or by the interaction with DMO;
- Modification of the SFL/SML by the upgrade of the slice template. In this case, the SML will interact with DMO requesting deployment or removal of a specific function or a node.

It is noteworthy that the modification of resource allocation to SFL or SML by SML can be proactive instead of the reactive approach provided by the MANO orchestration. Moreover, in the case of SFL, resource allocation can be driven by slice QoE. The modification of SFL template that is driven by SML may be used for cloning some slice functions in order to optimize slice traffic or to add additional components like DPIs or firewalls. The same mechanism can be used for the programmability of SML providing that way programmability of the slice management plane. Using the mechanism during SML runtime, new components like AEs or DEs can be added. This feature is very important for the evaluation of different AE and DE algorithms, but the programmability is also important in real implementations.

4.4.2.4 ACT SUBLAYER

The ACT sublayer's role is to convert high-level (intent) reconfiguration commands obtained from the DE sublayer into a set of atomic reconfiguration commands, as shown in Figure 28. Due to the existence of ACT sublayer, the DEs do not have to deal with details of reconfiguration. Therefore, they can be designed in a more generic way. The ACT sublayer can be seen as a set of device-specific (i.e. node/functions) drivers. The ACT typically interacts with EMs/EEMs of SFL, but they may also interact with DMO requesting orchestration related action (adding or removing a VNF).

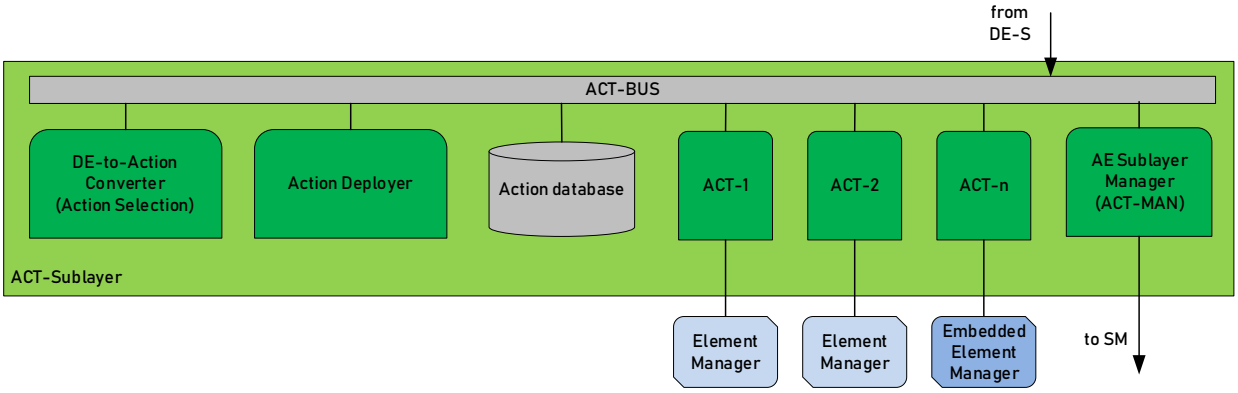

Figure 28. Actuator Sublayer Internal components

4.4.2.5 SLICE MANAGER

Slice Manager, presented in Figure 29, is an entity of SML that provides interactions with DMO and Inter-Domain Slice Manager (IDSM, described in this section). It can also be used for the manual management of SFL or to implement Policy-Based Management. It interacts with EEMs, MS, AEs and DEs. The component is responsible for sending to DMO and, if applicable, to IDSM, slice related synthetic information (KPIs). The SML provides direct, intent-based management to the Slice Tenant. This a perfect way of providing slices management isolation. For that purpose, the Slice Manager has a tenant portal and a set of tools that enable simple and comfortable slice management by slice tenant. A *conditio sine qua non* for such management is the embedded intelligence of the management that is in our case provided by AI algorithms. The management interface is created after slice deployment, and the slice tenant can use it for the lifetime of a slice. For accounting and historical reasons, the accounting data combined with slice resource consumptions and KPIs are transferred to IDMO Accounting database before termination of the slice.

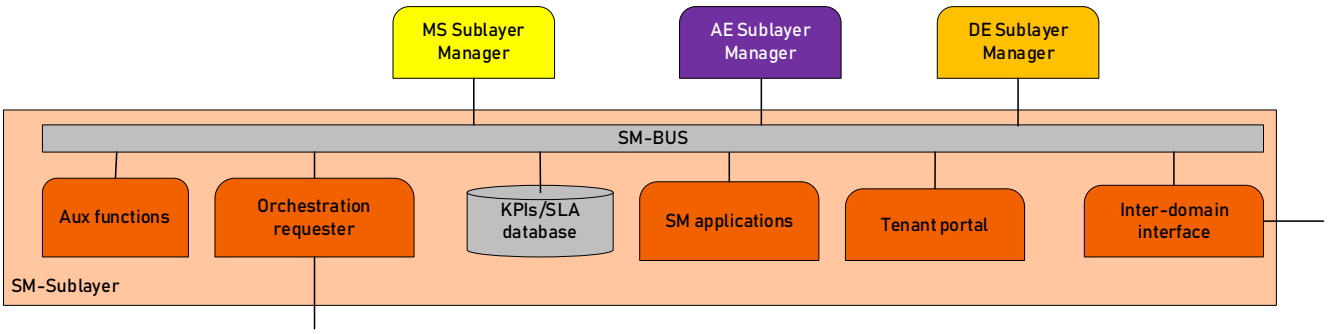

Figure 29. Slice Manager internal components

4.4.3 IDSM

The proposed SML-based slice management approach can also be used for end-to-end slice management when slices are implemented across multiple domains (SODs) as shown in Figure 30. In such a case, the entity called Inter-Domain Slice Manager (IDSM) is responsible for the end-to-end slice management. It interacts with SMLs of all domain slices that compose the end-to-end slice. The IDSM is a part of slice template (a set of VNFs), and in some cases, it can be generated automatically by the IDMO (if IDMO is responsible for slice template split between multiple SODs). When IDSM is in use, it provides to the Slice Tenant the management interface. The IDSM is also responsible for the calculation of slice-related KPIs. Figure 30 shows a multi-domain slice, with the IDSM is deployed in one of SODs.

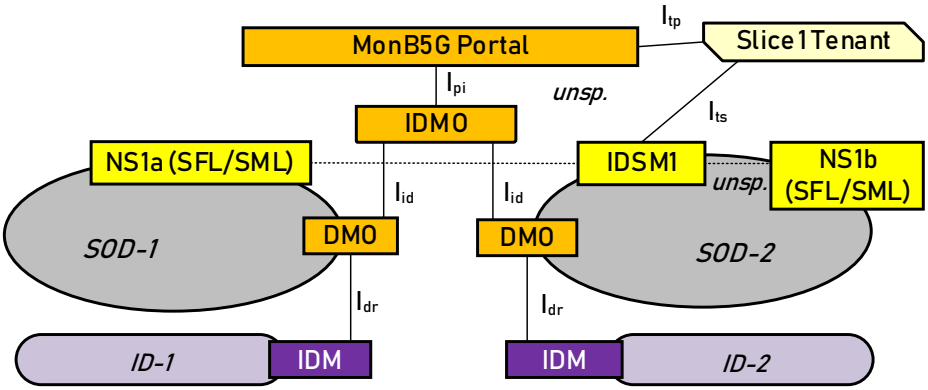

Figure 30. Multi-domain slice - the IDSM is deployed in one of SODs

The Inter-Domain Slice Manager (IDSM) can be seen as an instance of SML. The generic structure of IDSM and its usage has been presented in Figure 31.

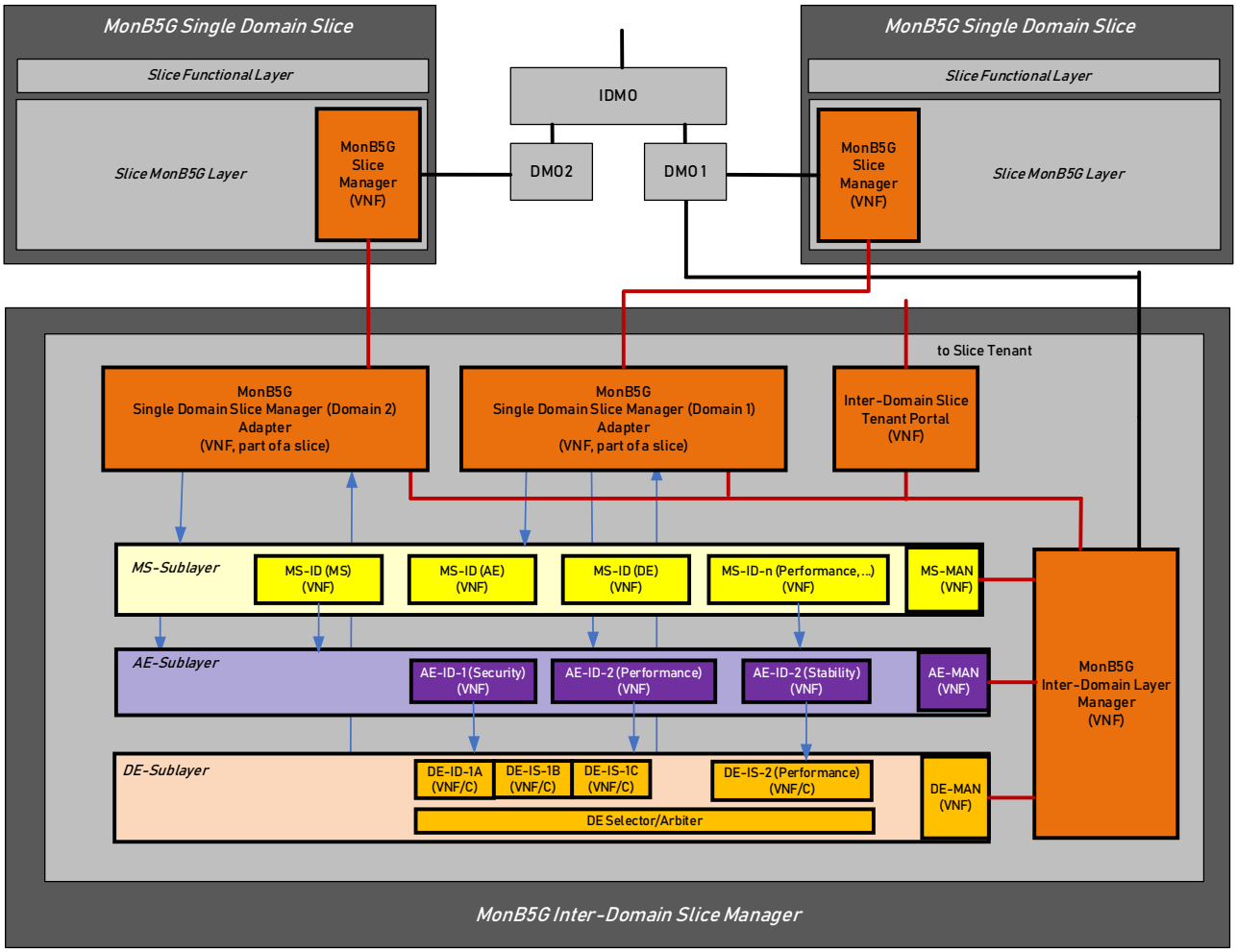

Figure 31. An example of usage of IDSM

The IDSM implements the runtime part of CSMF and NSMF of 3GPP (see section 3.2.2 for details on NSMF and CSMF).

4.4.4 MLAAS

The addition of SML to SFL undoubtedly increases the slice footprint, implying also longer slice deployment times. Moreover, sometimes the Slice Tenant is not interested in slice management. To solve the mentioned issues, MonB5G proposes the use of the Management as a Service (MaaS/PaaS) paradigm, as described in [64]. In this case, SML is an independent slice capable of managing multiple SFL instances of the same template. Such split requires the implementation of additional functions in SML related to the creation of secure partitions for the managed SFLs, as well as dynamic adaptation in case of deployment of a new SFL, or termination of the existing one. The MaaS platform (called MonB5G Layer as a Service – MLaaS) can be operated by a business entity called Slice Management Provider. LCM of MLaaS is done via I_{mp} interface. This case for a single SOD is marked as Option B in Figure 22 where SFLs of the MLaaS-managed slices also use services provided by DSF for reduction of SFL footprint.

In order to implement the approach, MonB5G architecture will leverage the concepts proposed in [64] in order to provide PaaS services to network services. These PaaS may also be requested by consumers via formal APIs, which will define the type of PaaS required (e.g. there exist many Container Infrastructure Services, such as Kubernetes and OpenStack Zun) and other design considerations to be defined. MonB5G components (i.e. MS, AE, DE and actuation component) may be deployed inside PaaS alongside each managed function (e.g. technological domain). These components are indeed managed by the infrastructure owners. On the contrary, PaaS requested by consumers is managed as a single entity by the operator, leaving LCM tasks associated with the services residing therein to the consumer.

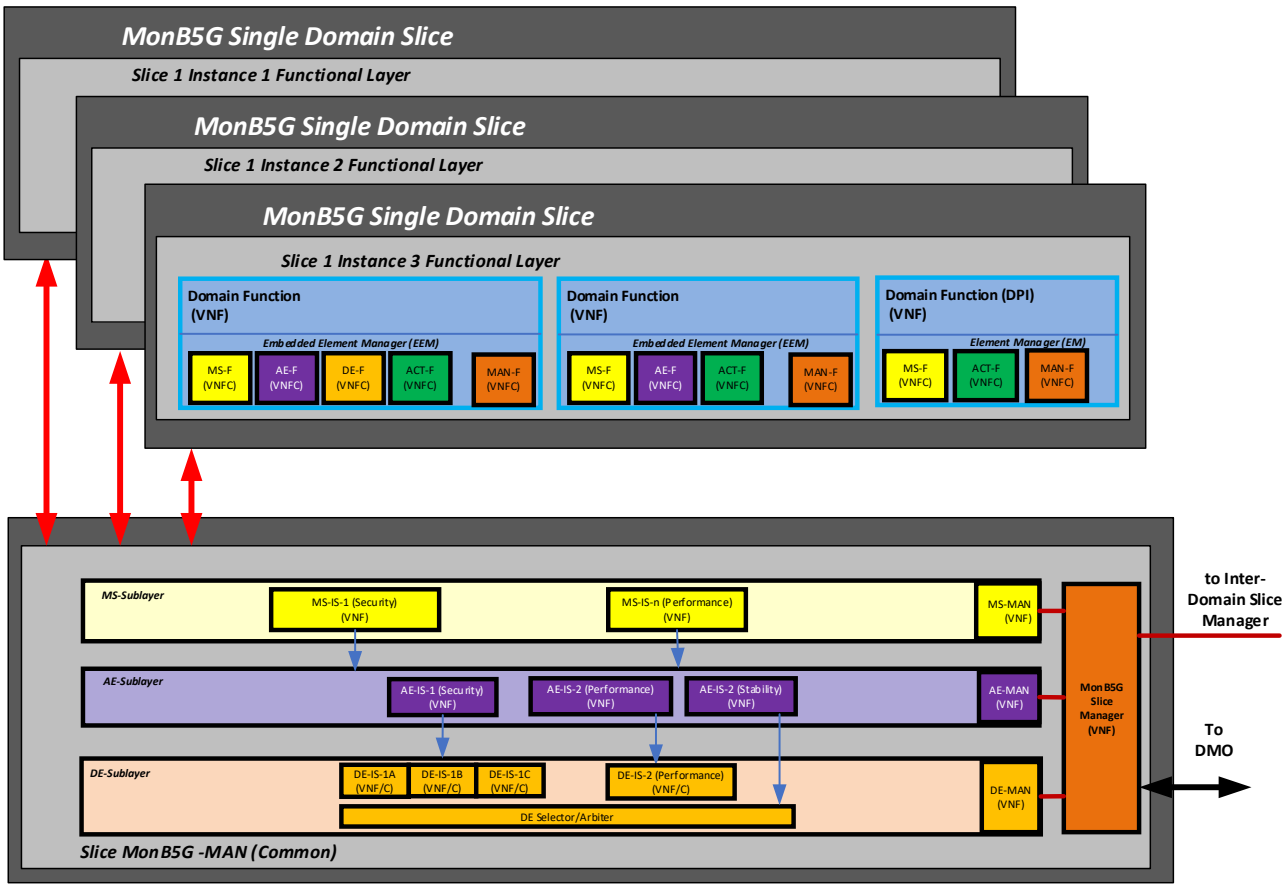

Figure 32. An example of usage of MLaaS (security, multitenancy to be added)

4.4.5 DSF

The Domain Shared Functions (DSF) are a set of shared functions (VNFs) that can be implemented as PNF/VNF or CNF and can be reused by SFLs of multiple slices. The approach provides a reduced footprint of the deployed slice. It is assumed that the DSF is dedicated to a specific SFL – it is a complementary part of it. The operation can be seen as vertical stitching of slices in which the DSF can be seen as a PaaS. In the PaaS case, both vertically stitched slices have SFL and SML layers implemented, unlike the MLaaS case. For the management of the multiple slices that use DSF, a special kind, dedicated IDSM is used. The DSM can be used in combination with MLaaS to further reducing slice footprint. In such a case, the slice contains only the SLF part, whereas the DSF consists of shared SFL, its SML and shared SML for SFL of slices that use DSF services.

4.4.6 IOMF

The **Infrastructure Orchestrated Management Functions** (IOMF) are specific functions that support infrastructure management. They can be orchestrated by the IDMO upon request of an Infrastructure Provider via the MonB5G Portal as described in section 4.3.1 and presented in Figure 33.

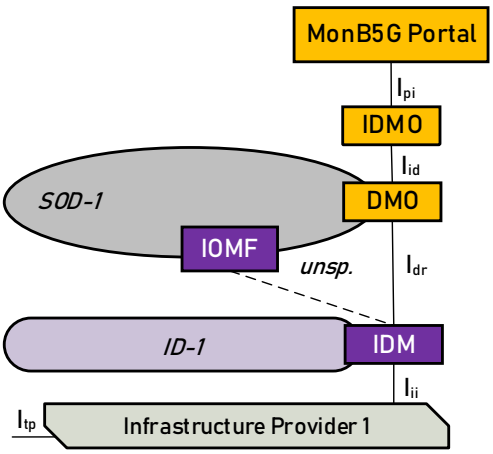

Figure 33. Deployment of IOMF function

The IOMF cooperates with IDM to achieve their specific goals. The IOMF functions can cover a variety of functions that can improve the effectiveness of infrastructure utilization and contribute to the overall quality of infrastructure management. The most prospective use cases include the deployment of IOMF for predictions of resource consumption and resource groupings to optimize utilization and as a result, achieve energy-saving goals. The IOMF functions are orchestrated in a similar way to slices, and LCM requests are sent by the Infrastructure Provider to the MonB5G Portal. The IOMF functions are specific for the virtualization technology used in the infrastructure and tools; therefore, they have to be customized for each IDM type separately.

4.5 Security components of the architecture

The security orchestration in MonB5G is based on the methodology recommended by security frameworks to implement and reinforce network cybersecurity. The National Institute of Standards and Technology (NIST) cybersecurity framework [95], for example, is divided into five main functions: identify, protect, detect, respond, and recover that comprise the security lifecycle. From the network model designed by the manager (NSMF or NSSMF), the security orchestrator makes an inventory of assets, captures the security needs of the intent object and then analyses the vulnerabilities, threats and risks to determine the set of security objectives that need to be achieved. It uses the AE to understand the structure of the network to help the DE manage security controls. Then the protective measures can be identified and deployed together with the network instance to protect it against cyber-attacks. Identity

and access control, the security of data and networks are examples of measures. However, security measures may have breaches as threats are constantly evolving; it is necessary to monitor services and resources in order to detect new security problems by anomaly or artefact. The events emitted by the detection system are then handled by the response system, which will determine, on the basis of its model and rules, the actions to be taken: for example, hypothesizing a known attack and then verifying it on the basis of the observed data, or activating existing mechanisms or deploying additional ones to limit the effects of the incident and eradicate it. This should be implemented by closed-loop automation involving MS, AE, and DE components. The security framework also recommends a post-incident analysis to learn from the incidents handled. The speed of remediation, the inadequacies of the data collected and the overall cost of an incident are all pieces of information that help to identify new vulnerabilities and risks, strengthen defensive measures, broaden the scope of analysis and improve response plans.

The MonB5G architecture is implemented at multiple levels of the management and orchestration hierarchy. The global security is managed using the E2E Security Orchestrator located in IDMO, besides, the security-related management loops. However, at the domain level, the security management is hosted in the DMO where we have a Domain (local) Security Orchestrator managing the security closed loops and the security enablers (VSFs). In the case of multidomain network slices, the NSMF (located in IDMO), and in the case of single-domain slices, the NSSMF (located in DMO), delegates part of the management of security goals or intents to slice management level, i.e., to SML.

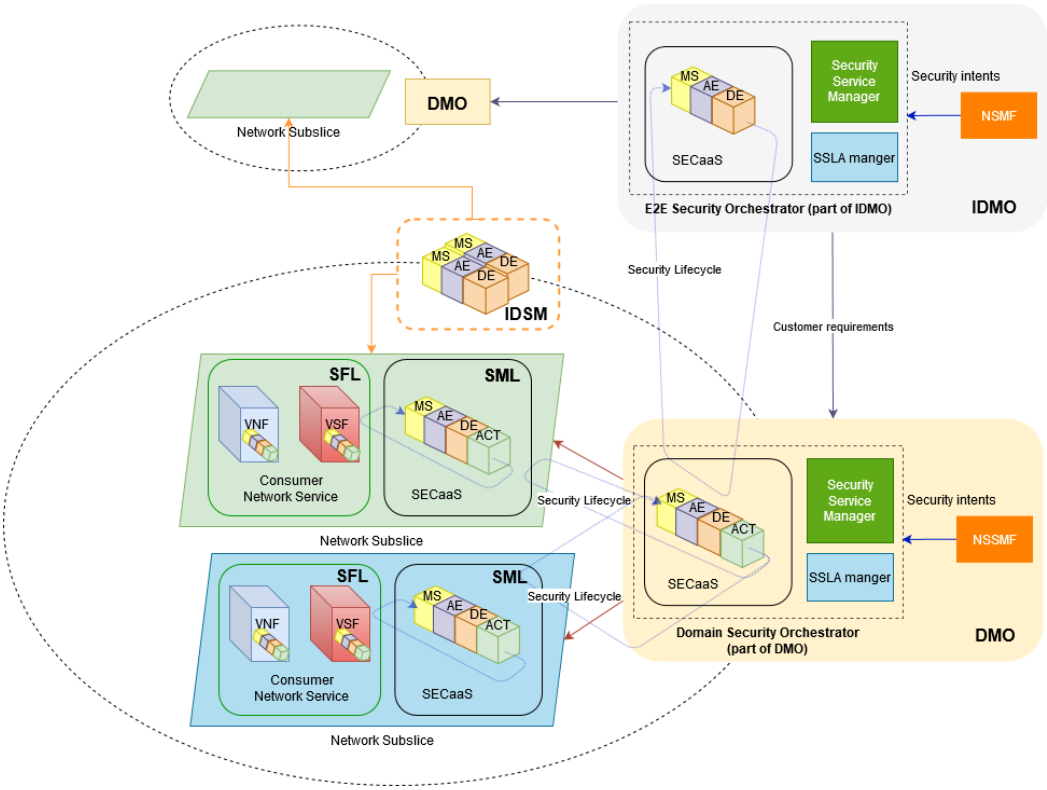

Figure 34 Security components of the architecture

Security control functions and the core functions of the cybersecurity framework are provided by a Security Service Platform (SECaaS). As a set of VNFs, the SECaaS can be dedicated to the safeguard of a network slice, or it can be shared. The application of the security framework at this layer requires the presence of a security platform in DSO and IDSO. Indeed, the SECaaS of IDMO manages the security lifecycle globally, from the framework core function and identify the framework core function respond/recover. The SECaaS of DSO can monitor multiple slice instances and amplify the signal to detect a threat that is transparent to every individual network slice. Another valuable point is the sharing of

information between the network slice to reinforce the protection. In addition, a Security Platform as a Service (SecPaaS, and implementation of PaaS) can be dedicated to a group of network slices according to some criteria: per tenant, vertical, slice type, security level. That isolation between SECaaS with the security orchestrator allows the possibility of employing multiple strategies and policies to manage the security life cycle.

At the inter-domain level, that of network slice, which aggregates several slice subnets to offer end-to-end services, a SECaaS is also in place to manage the security life cycle of network slice instances. From the customer security intent, the SECaaS of the NSMF can extract and distribute goals to each NSSMF. As for the detection, it can also collect security reports and detect a threat thanks to its view over the network segments. As a response, for instance, a change of security requirement is sent to the NSMF of a network slice subnet instance.

4.5.1 SECURITY ORCHESTRATION

The security functionalities are provided by the security orchestrators, namely the E2E Security Orchestrator and Domain Security Orchestrators. Although the security is distributed with the slice instances using the closed loops as SECaaS components (of slice SML), the security orchestrator controls the closed loops and the security enablers within a single domain slice (subslice). The overview of the executed security status in the domain and the slice requirements allow the DSO to efficiently manage the SECaaS. We provide security orchestration on two levels:

4.5.1.1 E2E SECURITY ORCHESTRATOR

The End-to-End Security Orchestrator (E2ESO, part of IDMO) has a global view of all the slices from end-to-end perspective. When it receives a slice template and SSLA requirements, it confirms or refuses the security agreement if the platform cannot ensure the required security. This security orchestrator will have the ability to identify the required security feature based on the blueprint before instantiating the slice. It is responsible for the high-level security reactions, i.e. Security Decision policies coming from the IDMO closed loops like migrating subslice to a new domain. The security manager manages those IDMO closed loops and might coordinate and assist the local security orchestrator with selecting the SECaaS to be deployed with each subslice. The E2E Security Orchestrator stores the policies enforced from his level into the conflict manager that permits to check any conflicts.

4.5.1.2 DOMAIN SECURITY ORCHESTRATOR

Domain Security Orchestrator (DSO) that is local to every domain ensures the local security management, first by instantiating the closed loops when deploying a sub-slice, thus providing a Security-as-a-Service functionality. As it is aware of the hosted slices and their related SECaaS, it can assist reusability by sharing the existing SECaaS or sub-components (MS, AE, DE), taking into consideration the isolation level. The domain security orchestrator has an overview of all the executed actions by SECaaS since the DEs save policies into its storage (conflict manager), which is exposed for the monitoring systems to avoid conflicts with the installed security policies. Also, the actuation sends notifications after enforcing security policies to update the stored policies state from "Enforcing" to either "Success" if all actions are correctly executed or "Failure" in case the Actuation component could not enforce the policy. By then, the DSO can check that the security issue is corrected without undesired side-effects locally or from the end-to-end perspective. Otherwise, it reacts to correct the failure.

Similarly to the E2ESO, the DSO has a Security SLA (SSLA) manager that interprets the SSLA requirements into security actions and KPIs while deploying a slice to monitor and ensure that the tenant security requirements are met. The security manager is responsible for instantiating and deploying the SECaaS. In addition to the conflict manager that stores the domain enforced policies and prevent the conflicts.

4.5.1.3 SECURITY AS A SERVICE COMPONENTS

Our architecture is strongly based on the triplet components Monitoring System (MS), Analytic Engine (AE), and Decision Engine (DE). At certain levels, a component called actuator (ACT) is added (see details in section 4.4.2.4). These components are providing security as a service (SECaaS) and can be shared between multiple slices. As we expect to have different security and management components, we use the layering concept for reusability and simple communication. To clarify, all MSs are connected in a layer (Monitoring Sublayer), and similarly, we have an Analytic Sublayer and Decision Sublayer. A vertical channel connects the three sublayers; for example, an Analytic can request additional data from other MSs. Also, this channel opens an interface for the other hierarchical management layers so any management element can have access to any services.

- The MS collects real-time security-relevant data and provides information to the AE. The data sources are identified depending on the hierarchical level and the final objective of the DE. The MS pre-processes the collected data.
- The AE processes the security-related monitoring data in order to provide high-level security information and events. Analysing the collected KPIs, network flows and resources status will help in diagnosing the node and the network to detect or predict attacks and security issues.
- The Decision Engine is the mastermind that has the ability to tell the system what to do as a reaction or prevention to protect the network against security threats. The Security DEs' role is crucial in detecting attacks and deciding on dynamic security policy per slice and per attack episode. The decision can configure an existing security enabler in the slice or deploy a new one; however, these decisions are described in an abstract model rather than vendor-specific. Among the DEs distributed horizontally and vertically, each can be related to an application/VNF, sub slice, or a slice scope. The DEs are atomic elements that have a specific autonomic security function. For instance, at the VNF level, the DE embedded is responsible only for a unique security threat that may target its hosting VNF. In this case, the security vulnerability or the potential attack should be known earlier (before deploying the DE) based on the application running or the protocols which may differ from a VNF to another. At the higher level, the subslice's attached DEs depend on the subslice threats and characteristics. Similarly, for slice DEs deployed in the IDSM. This distribution will greatly simplify the autonomic threat detection and fast, local response.
- The Actuation component (ACT) is in charge of executing security policies. First, it performs a translation from the high level into vendor-specific configuration according to the targeted enablers. The ACT can trigger deploying a specific VSF through the MANO or update the configuration on an existing VNF/VSF.

4.6 Interfaces of the MonB5G framework

The MonB5G architecture is composed of static and dynamic components. Similar to ETSI MANO architecture, some management-specific interfaces are not defined as they are not general and dependent on the slice template and its functions. A similar problem concerns the Infrastructure management that is virtualization platform dependent. Lack of specification of interfaces concerns mainly interfaces of the dynamic components of the architecture. It is worth noting that embedded slice management reduces external interfaces and makes them more abstracted. The most important interfaces are the interfaces between the Management Portal and IDMO, between the IDMO and DMO and between the DMO and IDM. In the MonB5G concept, the hierarchical orchestration approach has been proposed. Despite it, the orchestrators of multiple levels do not interact directly. For the interaction between them, an OSS integrated with the orchestrator is used. Such an approach makes the possibility of more abstracted, i.e. orchestrator type independent orchestration interface definition. In comparison to MANO interfaces, the MonB5G interfaces are enhanced in order to provide an exchange of information related to energy-aware operations and resource brokering. As the MonB5G architecture is a reference one the interfaces are not defined in a detailed way – their definition is implementation-dependent. The

outline of the specification of the interfaces of the static components of the architecture is provided in Table 1.

Table 1. Interfaces of the MonB5G framework

Interface	Type	Description
Iop	Web interface	The interface is used by the MonB5G System Operator in order to manage the whole MonB5G system.
Itp	Web interface	This interface is used by Slice Tenants and Infrastructure Providers for the purpose of slice LCM.
Imt	Web interface	This interface is used by Slice Management provider for the communication with Slice Tenant for the purpose of runtime slice management.
Its	Web interface	This interface is used by Slice Tenant for the purpose of runtime slice management, can be for interaction with IDSM or SML.
Imp	Web interface	This interface is used by Slice Management Provider for the purpose of MLaaS LCM.
Ipi	Web interface	The interface is used by the MonB5G Portal in order to interact with IDMO. The interactions involve the negotiation of the slice deployment policies and are used for the LCM of multi-domain slices.
Iid	Os-Ma-Nfvo-like interface	This interface is used by the IDMO for the purpose of LCM management of slices implemented by DMO. It can be seen as MANO Os-Ma-Nfvo extended interface. It may provide LCM abstractions and provides data and management capabilities of the DMO to IDMO.
Idr	DMO-IDM interface	This interface is used by DMO in order to allocate, update or deallocate resources. This interface is used also used to exchange between DMO and IDM additional information about the infrastructure, for example, about energy consumption and cost. It can be seen as extended VIM – NFVI interface.
Iii	Web interface	The interface is used by the Infrastructure Provider in order to manage its infrastructure domain.

5 Key management and orchestration functionalities supported by the MonB5G architecture

The MonB5G architecture presented in section 4 is distributed, AI-Driven and programmable. The mentioned features make the proposed approach scalable and flexible. It is rather hard to define all the functionalities that are supported by the architecture. It is, however, possible to list the key requirements of the ETSI ZSM group that the MonB5G architecture fulfils. The ETSI ZSM requirements list contains over 170 different topics related to autonomic management. Some of the requirements concerns procedures, like testing or software upgrade, therefore they are not related directly to the architecture. However, we have decided to extract a relatively small but important subset of the ZSM requirements that the MonB5G supports. The ETSI ZSM requirements list is flat, without any grouping. For the purpose of the section, it has been decided to split them into several categories related to the specific aspects of autonomic service management. The categories include monitoring and data analytics, management actions, management operations, control loops and other important requirements not belonging to previous categories.

5.1 Monitoring and Data Analytics Functionalities

The Monitoring and Data Analytics category includes the requirements related to the collection of the performance data, their aggregation and ways of data usage to fuel analytic engines (AE). In particular, MonB5G framework supports:

1. Collecting performance data and fault data for a network instance at different granularities and aggregation of VNF/PNF raw measurements to calculate, e.g., slice-level KPIs;
2. Performing data analytics for predicting KPI changes and failure conditions;
3. Storage of historical data needed for the prediction and its exposure to the analytics;
4. KPIs measurement;
5. Analysis of the collected data to detect the undesired states and derive the root cause. The past, current and future states of the managed entities can be modelled to help to detect the undesired states and move the state of managed entities to the desired state;
6. Prediction of the growth or reduction of traffic volume for managed resources for a Customer-Facing Service (CFS) and over a given time period;
7. The capability to predictively detect abnormal behaviours of the managed networks and services.
8. The ability to demand forecast for capacity planning;
9. Monitoring of managed services (network as a service including network slicing as a service) originating from different network/infrastructure domains including but not limited to NFVI, IP/SDN networks, fronthaul, and Radio;
10. The capability to analyse conditions to detect root causes.

The requirements mentioned above can be satisfied due to incorporation of hierarchical control loops and sets of MS, AE, DE, ACT components that perform optimization and management-related tasks on each level. The performed operations concern both service-related events (SML, IDSM) as well as resource utilization (DMO, IDMO, IDM, IOMF).

5.2 Management Actions

Management Actions category consists of requirements related to network maintenance, coordination of management, recovery actions etc. These include:

1. The capability of taking actions to perform predictive maintenance of a network instance;
2. The management coordination across different technical domains, including at least Core network domain, RAN network domain, transport network domain and virtualization part to support network slicing management;
3. The capability to perform recovery actions based on the KPIs of the managed networks and services;

4. The capability of zero-touch, E2E management and orchestration of 5G networks and services covering network slicing and edge computing;
5. Managing the complete lifecycle of the network services/capabilities exposed per management domain, and shall provide an interface that hides internal details (such as the resource layer).

The MonB5G framework enables management and orchestration in different administrative and technological domains (RAN, CN, Edge etc.) by introducing abstractions and entities enabling concatenation of slices deployed in different domains (IDMO, IDSM) into an end-to-end slice. The framework also exposes capabilities for a framework operator (MonB5G Operator) to intervene in case of critical failures and perform recovery actions.

5.3 Management Operations

Management Operations relate to access to network slicing management services, LCM, management data policies, etc. In particular, the MonB5G framework supports:

1. Access to network slicing management services exposed by the framework for authorized vertical industry customers;
2. The capability to provide the interfaces' exposures for the automated management of the services;
3. Data availability inside management domains and outside of them so that it can be exposed to any authorized consumer within the framework belonging to one operator;
4. The capability of automatic installation of management software;
5. Automatic configuration of management software parameters;
6. Automated detection of management services offered by a management domain;
7. Automated lifecycle management of the framework functional components.

The access to the MonB5G framework capabilities is provided via portal interfaces described in section 4.3.1.

5.4 Control Loops Support

This group of requirements is devoted to the operation of control loops, disabling the control loops in terms of faulty operation etc. Particularly, they include:

1. Capability to allow different sets of collected data to be used in different closed loops inside a domain and cross-domain;
2. Detection and conflict resolution between different closed loops inside a domain and in different domains;
3. Nested closed loops;
4. Reconfiguration of any domain services as required, e.g. in support of closed-loop assurance.
5. The use of automated decision loops, with different characteristics and scope, as a means to perform network and service management;
6. Provision of an interface for the purpose of bringing decision criteria to the decision loops, i.e. triggers, policies;
7. The collection of all available, relevant data and contextual information for a specific decision loop.
8. The ability of the network owner to disable any automation function in case of malfunction.

MonB5G implements control loops on VNF-level, slice level, domain level and inter-domain level. The slice level (SML Sublayer) has mechanisms enabling detection and conflict resolution between different loops. Each SML sublayer provides the interface enabling modification of properties of each separate sublayer by the Slice Manager. Fully manual management in case of the control-loops malfunction is supported.

5.5 Uncategorized functionalities

Other important functionalities of ETSI ZSM that can be implemented in the MonB5G framework include:

1. Scaling of network slice instances within available network resources;
2. Configuration of network slice instances during runtime without disruption;
3. Status monitoring of all the network slice instances and identification of the network slice instances causing high utilization of network resource(s);
4. Automatic performance of FCAPS management for compute, storage and network resources, NFs, slices and services;
5. Automatic configuration of physical and virtualized network function parameters;
6. Taking decisions regarding actions to take, time of their execution, and the execution itself based on the analytics results;
7. Security capabilities when delivering automated network and service management.

The mentioned functionalities are supported by IDM, SML, AI-driven AEs and the security orchestrator.

6 Remarks on the implementation of the MonB5G architecture

6.1 Tools to be used for the implementation of MonB5G architecture

In this section, we will provide a mapping between the reference architecture described earlier and a set of existing tools for future implementation. The architecture depicted in Figure 35 represents three technological domains hosted in different infrastructure virtualization solutions (AWS, OpenStack[2], Docker[3]), which represent the infrastructure layer.

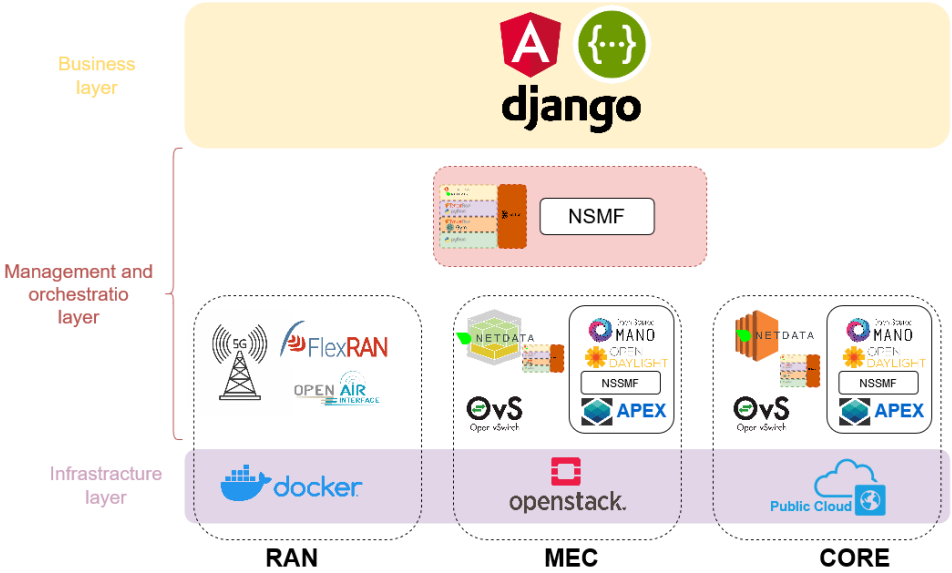

Figure 35. Mapping of MonB5G reference architecture and existing tools

For the Management and Orchestration Layer in the Core (Cloud) and MEC (Edge), we have mainly ETSI OSM[4] for VNF management and orchestration in addition to the Open Daylight (ODL)[5], an SDN controller that controls the links between different nodes using Open vSwitches[6]. Additionally, APEX[7] for policy management and execution. The RAN domain in our approach uses the FlexRAN[8] on top of OAI[9] for controlling and slicing the Radio domain. The NSMF and NSSMF contain the slice management functions that are supposed to be developed since, to the best of our knowledge; there is not an open-source tool that can provide the required functions. The Business Layer has web interfaces and APIs with a database to store authentication credentials. Many tools are available on the web market Django[10]/Angular[11] and Swagger[12] are among the most popular stable tools. MonB5G is heavily based on the AI-driven management (underpinned by MS, AE, and DE components) described earlier for autonomous management. For the implementation, the Netdata[13] tool can be used to gather resources-related and

[2] Openstack, [Online]. Available: https://www.openstack.org/
[3] Docker, [Online]. Available: https://www.docker.com/
[4] ETSI Open Source MANO, [Online]. Available: https://osm.etsi.org/
[5] Open Daylight, [Online]. Available: https://www.opendaylight.org/
[6] Open vSwitch, [Online]. Available: https://www.openvswitch.org/
[7] APEX, [Online]. Available: https://docs.onap.org/en/dublin/submodules/policy/apex-pdp.git/docs/APEX-Introduction.html
[8] FlexRAN, [Online]. Available: https://mosaic5g.io/flexran/
[9] Open Air Interface, [Online]. Available: https://openairinterface.org/
[10] Django, [Online]. Available: https://www.djangoproject.com/
[11] Angular, [Online]. Available: https://angular.io/
[12] Swagger, [Online]. Available: https://swagger.io/
[13] Netdata, [Online]. Available: https://www.netdata.cloud/

network information of the deployed VNF instances. Prometheus might gather information from several customized agents to collect slice specific KPIs. The AEs and DEs are mostly AI-based that are planned to be developed using the ML frameworks such as Python TensorFlow[14], Pytorch[15] and Open-AI gym[16], as presented in Figure 36. The communication between layers MS, AE and DE can be based on publish/subscribe tools like Kafka[17].

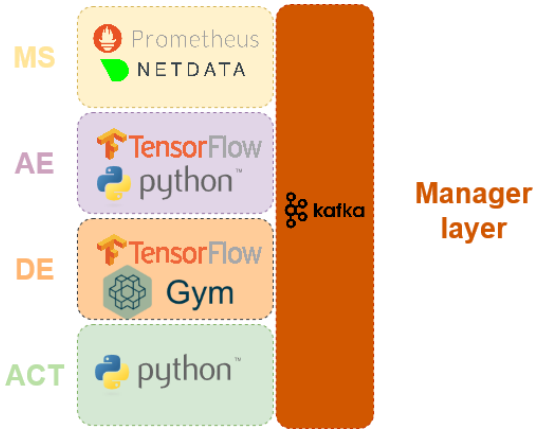

Figure 36. MLaaS instantiation (example)

6.2 Implementation of MS/AE/DE functions

In this section, we will provide more information about the implementation and interactions between different MS/AE/DE components of the architecture. The description is just exemplary as many of the MS/AE/DE components are part of a slice or they are implemented in PaaS style. There are, however, some recommendations for the implementation of the components. We will show an example implementation of the components on selected MonB5G Use-Cases (UC).

6.2.1 MONITORING SYSTEM ROLE AT DIFFERENT LEVELS OF MANAGEMENT HIERARCHY

In MonB5G, the management system is distributed for scalability reasons. It especially concerns the monitoring part of the management. The proposed separation of concerns splits the slice-specific monitoring from resource-oriented monitoring.

6.2.1.1 SLICE-RELATED MONITORING

For the slice part, the producers of the monitoring information are typically the virtual functions of the slice (i.e. EMs or EEMs); moreover, the aggregated and correlated monitoring information of the virtual function of a slice contributes to the information about a slice collected by the Slice Manager of SML. In case of multi-domain slices, the monitoring information about (sub)slices that compose the end-to-end slices is sent from respective Slice Managers to IDSM. It has to be noted that at each level of the autonomic slice management hierarchy, i.e. the node/function level, slice level and inter-domain level, there are also consumers of the monitoring information and in some cases, the monitoring information concerning a specific slice is not exchanged between the entities into a primary (raw) format. The embedded intelligence implemented at different levels of the management hierarchy allows for the use of the concept of intent-based interfaces. The monitoring information at the node/function level can be consumed by AE/DE of the node/function in order to deploy, for example, the plug-and-play behaviour of

[14] Tensorflow, [Online]. Available: https://www.tensorflow.org/
[15] Pytorch, [Online]. Available: https://pytorch.org/
[16] Open-AI gym, [Online]. Available: https://gym.openai.com/
[17] Kafka, [Online]. Available: https://kafka.apache.org/

the node. At the slice level, the monitoring information is consumed after pre-processing for the purpose of the self-management of a slice that based on multiple AEs and DEs of SML of a slice. As envisioned in MonB5G, the principal consumer of MS information are AEs. Indeed, AEs are in charge of triggering the monitoring of needed information from MS. The latter starts the monitoring process by connecting to the appropriate source, i.e. infrastructure and Function. Accordingly, MS exposes two types of APIs: the Control API and the Data Collection API. The MS Control API may be used by AE to request specific metrics to monitor, the periodicity, resolution the duration, etc., while the Data Collection API is the interface from which data are provided to AE as requested through the control API. The control API also indicates how data are provided, i.e. publish/subscribe, request/response, the data format, etc. Besides collecting monitoring data and providing API for AE to control and consume monitored data, MS implements functions to treat the collected data as described in Section 4.4.2.1. MS may perform data transformation by adding semantic and context information, such as the timestamp, source of data (e.g. MDO, function, type, etc.), metric name, and value. Moreover, MS may use persistent storage to store monitored data, their interpolation and extrapolation for future requests.

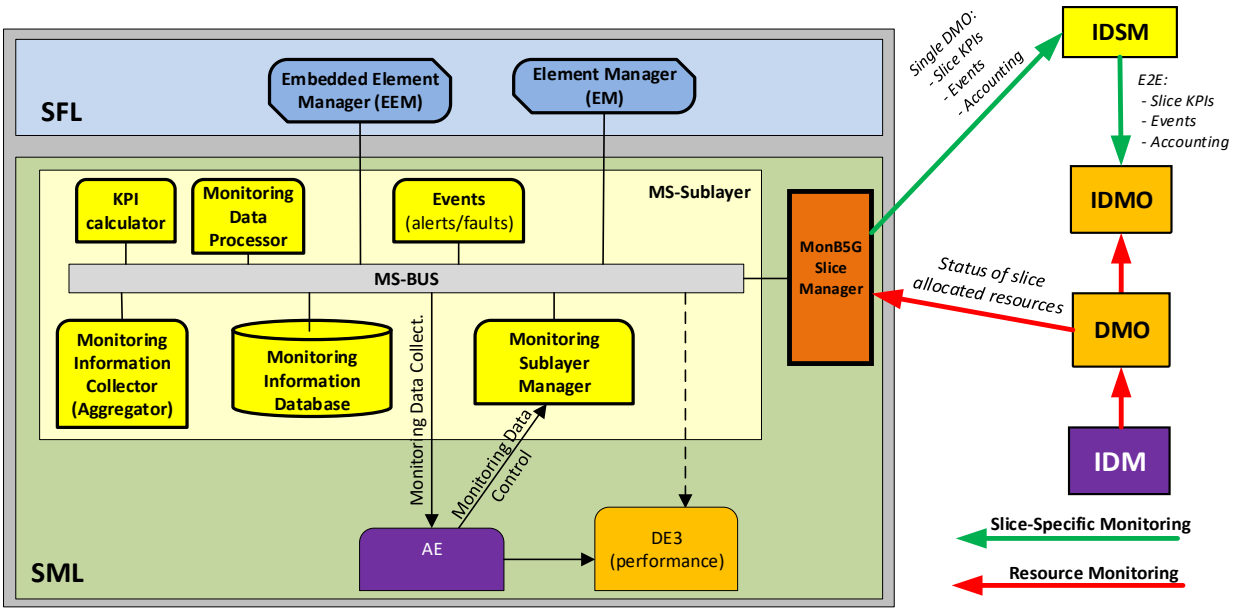

Figure 37. Overall end-to-end monitoring information exchange in case of a multi-domain slice (an example)

Part of the pre-processed MS information of a slice goes out of a slice, and it concerns IDSM, DMO, via DMO the IDMO and the Slice Tenant Management Interface (or Slice Management Provider Interface). The external SML information exchange is mostly used for the exchange of slice-related KPIs and accounting related information, security incidents and for passing information about faults. The slice-related KPIs are calculated by MS and transferred externally via Slice Manager of SML. The Slice Manager is also a consumer of the fault-related from EEMs/EEMs (direct faults), respective AEs of the SML (faults identified by AEs) or from DMO (concerns only infrastructure faults linked with resources allocated to a slice. Faults and security-related incidents are in a direct form are passed through all levels of MS hierarchy. In cases when proposer mitigation has been taken at any level, this information is also exchanged between different levels of the monitoring hierarchy. It has been assumed that in case of resources, the mitigation actions are taken first by IDM, next by DMO and if DMO cannot handle the fault, it sends the information to SML, which is trying to find fault mitigation at the slice level. If all the mechanisms cannot solve the problem, the information about the faults is passed to the IDO, which is in charge of finding a solution by the deployment of the slice or sub-slices in a different domain or to terminate it if such operation is not feasible. The information exchange between MS components of

different levels of the MS hierarchy in the case of a single slice is presented in Figure 37. In the presented case, a multi-domain slice has been shown. In the case of a single domain slice, the IDMO does not exist, and the information from Slice Manager goes directly to IDMO

It has to be noted that a significant part of the MS is included in a slice template; therefore, the MS-related internal to SML interactions do not have to use standardized interfaces. However, some of the MS functions can be pretty common for different types of slices; therefore, some implementation guidelines can be given. The form of the information exchange between SMLs, DMOs and IDMO can be, to a certain extent, universal. The issue is left for further study and will be solved during the MonB5G architecture implementation.

Table 2 shows some examples of monitored KPIs by MS of SML for selected MonB5G use-cases. We organize this table according to the technological domain from which it has been generated and where it will be demonstrated. We recall:

- UC1Sen1: Zero-Touch multi-domain service management with end-to-end SLAs;
- UC1Sen2: Elastic end-to-end slice management;
- UC2Sen1: Attack identification and mitigation;
- UC2Sen2: Robustness of learning algorithms in the face of attacks.

Table 2. Slice-related KPI examples

Source	Technological Domain	Metric/KPI	MonB5G UC
	RAN, Edge, Cloud	Latency	UC1Sen1 and UC1Sen2
	RAN, Edge	Bit rate	UC1Sen1 and UC1Sen2
	RAN	Packet Loss Rate	UC1Sen1 and UC1Sen2
	VNF (MME/AMF)	Number of UE attach	UC2Sen1
	Cloud or Edge NFVI	CPU and memory consumption of used VNFs	UC2Sen2
	MEC Platform	User access	UC2Sen1
	RAN, Edge, Cloud	SLA Violations	UC1Sen1 and UC1Sen2
	RAN, Edge, Cloud	Reaction time to NS malfunction	UC1Sen1
	Cloud or Edge NFVI	Network Energy Efficiency	UC1Sen2
Function	–	Video Quality (QoE)	UC1Sen1 and UC1Sen2
	–	Latency	UC1Sen1 and UC1Sen2
	–	Service response time	UC1Sen1 and UC1Sen2

6.2.1.2 RESOURCE-RELATED MONITORING

The resource monitoring is separated from slice monitoring; however, some information between the two monitoring areas is exchanged. The resources monitoring at the Infrastructure level is done for the purpose of availability of resources, consumption of resources and for detecting resources related faults. Moreover, the information about the energy consumption by slice allocated resources is fed from IDM to DMO. The Infrastructure monitoring, via the interactions between the IDM and DMO, allows the DMO to collect information on:

- NFVI: such as computing platforms and hardware;
- Physical Network Function (PNF) running network functions on dedicated hardware: such as eNB/gNB, router, and UPF;
- VNFs are running common virtualized network functions: such as Core Network (CN) functions or DNS (including DFS).

The collected information is used by DMO for LCM operations on slices like admission control and resources scaling.

6.2.2 *ANALYTICAL ENGINES*

Analytical Engines (AEs) in MonB5G are distributed in the different parts of the management system. They perform a key role in SML, looking for anomalies related to a single slice operation. In DMO AEs look for a resource and orchestrated anomalies, while in IDMO, the AEs look into end-to-end related anomalies. The AEs of different management components (SML, DMO, IDMO, IDM) do not cooperate; they typically prepare data for DEs.

In the case of SML, as opposed to MS, AE does not store but processes data gathered from the same or lower-level MS or AE; and exposes the result to any requester (i.e. DE or another AE) in an on-demand or periodic fashion. AE to AE communication is possible, mainly to build a learning model using federated learning techniques.

Generally, the main functions of AE are: (i) identify performance degradation or a fault of a network slice; (ii) optimize the performance of a network slice or the DMO resources; (iii) react to security threats. To this aim, AE subscribes to data types to which it is interested, using the control API exposed by the MS. The data type will be determined according to the logic of the LCM application runs. Then, AE starts receiving the stream of data or uses a request/response mechanism, depending on the purpose of the analysis. AE may adapt the monitoring data rate or stop the request and request for other related monitoring information. AE is able to complete an inference task locally, extract features, and analyse these features and send alerts and notifications to DE. AE may collaborate with other AEs to build distributed learning (based on federated learning) model to realize the analysis and notify the DE accordingly. For the adaptation of MS, an AE has to interact with the Slice Manager of SML.

Table 3 shows examples of AE considering the MonB5G use-cases:

Table 3. AE examples

Feature	PoC of MonB5G
Prediction of SLA violation	UC1Sc1, UC1Sc2 and UC2Sc1
Prediction of NS Faults	UC1Sc1 and UC1Sc2
Attack Identification	UC2Sc1
Anomaly detection	UC2Sc1
Prediction of service migration	UC2Sc2

6.2.3 DECISION ENGINES

Decision Engines (DEs) exist in different layers of the architecture, moreover in the same layer than can be several DEs that realize competing goals. In the MonB5G architecture, the conflicts between DEs have been solved by several mechanisms. The first one is a Decision Arbiter that, according to active policy, selects the DE output to be enforced. Another separation is provided by the range of the DE-related reconfiguration. Some DEs operate on separate slices (i.e. SFLs), some may also operate on their SML, some DEs are oriented towards resources (IDM, DMO) some are oriented towards slice orchestration (DMO), some may be used in slice deployment preparation phase to provide an optimal split of a slice template into multiple domains. The third kind of separation between DEs is a time scale. It is expected that the fastest control-loops will reside inside EEMs, slower ones inside SML, yet slower reconfigurations will be done by DMO. That way, somehow interfering reconfigurations will be separated, even if their impact is global.

As depicted in Figure 38, DE is the decision-making element of the MonB5G architecture. It analyses alerts and notifications from AE(s) and considers a decision to take. The decisions are either derived using a local ML algorithm, based mainly on Reinforcement Learning (RL), or a predefined policy enforced by the Tenant or DMO through Intent, or a combination of both. DE may collect notification from several AEs, which may interact with MS monitoring different Technological Domains (TDs) to consider a global decision on the end-to-end NS. Global decisions are mainly considered at the IDMO level.

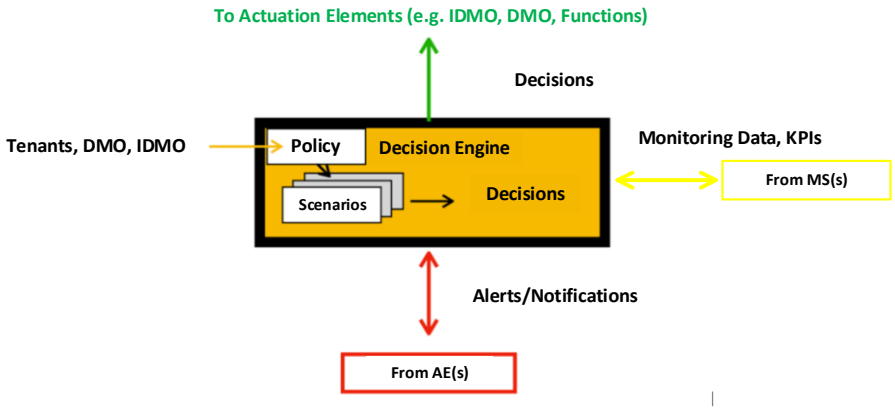

Figure 38. DE interactions

DE interacts with actuation elements (function, DMO, or IDMO) to enforce the considered decisions. For local decision, DE interacts with DMO and function; while for global decisions, the DE has to interact with IDMO. Table 4 shows examples of DE decisions in relation to MonB5G use cases. We classified the decision according to their scopes, i.e. local or global.

Table 4. DE examples

Decision	Type	MonB5G UC
Scale VNF	Local	UC1Sc1
Energy optimization	Global	UC1Sc1
Increase the RAN resources for an NS	Local	UC2Sc2
Block a UE connection	Global	UC2Sc1
Service migration	Local	UC2Sc2
VNF Placement	Global	UC1Sc2

6.3 The use of PaaS in MonB5G

MonB5G leverages the concepts proposed in [64], particularly the definition of Platform as a Service (PaaS) as a new NFV Object. The concept is used for the SFL of a slice and also for SML. In both cases, it contributes to the reduction of the slice footprint that contributes to faster and higher efficiency (in terms of resource consumption and cost). The VNF Common/Dedicated Services Instances that can be used by the SFL part of the MonB5G slice are called Dynamically Shared Functions (DSF), and the VNFs of the SML are called Management Layer as a Service (MLaaS). Both concepts use the same implementation mechanisms and can be used simultaneously to the same slice. However, it has to be noted that in most cases, the DSF and MLaaS are not generic i.e., they are SFL template specific. DSFs need an operator that cannot be a single slice tenant, not the DMO operator (we want to keep the DMO slice agnostic). Therefore, in both cases, there is a need to use the Slice Management Provider as an operator. The VNF Common/Dedicated Service is working on a VNF level; however, from the management point of view, it has been decided to group DSF and MLaaS VNFs and treats them as slices. Therefore, their usage can be seen as vertical stitching of slices. The VNF Common Service Instances (MLaaS, DSF to be deployed before a slice tenant that wants to use it. In the MonB5G case, such checking and eventual deployment are provided by IDMO. ETSI IFA 029 provides support for container-based services. In general, most of the mechanisms described in detail in ETSI IFA 029 are applicable in the mentioned use cases of PaaS, especially those related to security (Sec 8.4 of ETSI IFA 029). The usage of PaaS comes with the mentioned above benefits, but the main negative feature of the approach is reduced security. In order to increase the isolation and security of slices, the 3GPP recommendations included in [96] can be reused. To that end, Network Function (NF) discovery and registration shall be authorized and support confidentiality, integrity, and replay protection of data. The mutual authentication between NF Service Consumer and NF Service Producer shall be supported.

The deployment of some functions in NFVI, seen in a similar way as PNF, especially if they use hardware acceleration, has to be also considered. The usage of PaaS as NFVI resources is left for further study.

6.3.1 DSF IMPLEMENTATION

The DSF implementation can be seen as vertical stitching of slices. Both slices have SFL and SML parts. The management of such a combination of slices should be made by Slice Management Provider. The interactions are typically API-based. A DSF is not a generic PaaS, but it is linked with a specific slice template; therefore, there is no need to discover its functions. All the security mechanisms described above should be used.

6.3.2 MONB5G MANAGEMENT AS A SERVICE IMPLEMENTATION

A tenet of MonB5G reference architecture is the concept of the fast local control loop and increasingly slower ones for wider-scope administrative domains (e.g. slice-level, tenant-level, etc.). Leveraging the proposed concept of MLaaS, it is evident that many software components (i.e. MS-C, AE-C and DE-C) need to be able to interact in order to realise such control loops. Moreover, IDMO/IDSM may require changes in its administrative components (e.g. lifecycle management operations, reconfiguration of parameters) in order to comply with new objectives or ensuring support for an increasing number of managed SFLs.

Figure 39 shows a simplified version of a slice, according to MonB5G reference architecture. It includes a Slice Functional Layer (SFL) and MonB5G Layer as a Service (MLaaS). The former is composed of the tenant's VNF and corresponding EEMs/EMs that allow MLaaS components to gather metrics and/or actuate upon such resources. MLaaS is in turn composed of VNFs providing the Container Infrastructure System (CIS) (i.e. Container Infrastructure Service Instance and Managers, CISI and CISM, respectively), and the PaaS itself (as a COE) which effectively acts as a runtime environment for MonB5G administrative elements and components. MLaaS management services are therefore conceived as cloud-native

applications build-out of interaction among several MonB5G components of SML (i.e. MS-C, AE-C, DE-C, Slice Manager). This is exemplified by NFV MANO and Infrastructure Optimisations in the figure.

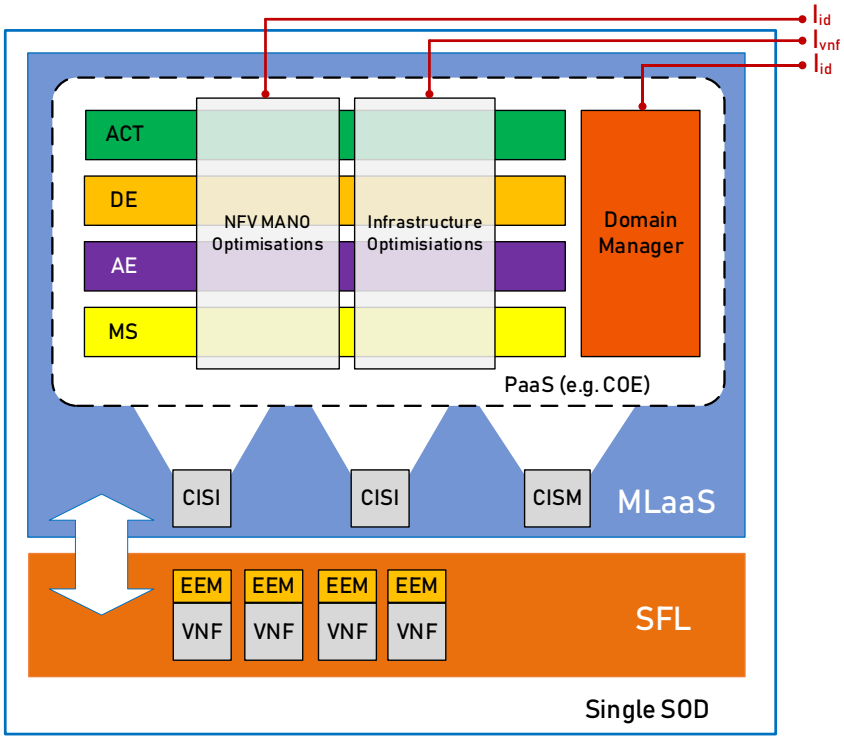

Figure 39. Single Slice Orchestration Domain DMO instance

Leveraging COE for hosting MLaaS allows dynamic reconfiguration of components, custom resource scaling mechanisms for supporting an increasing number of SFLs, and compatibility with ETSI NFV via the models proposed in [64].

The case of IDSM/IDMO, therefore, implies the instantiation of MLaaS at different technological domains (e.g. Edge, Cloud). Resource-wise, this is achieved via IDMO's NFVO, while a centralized provision of MonB5G software components is achieved leveraging CIS federation (e.g. Kubernetes KubFed API[18]). Figure 40 further elaborates by providing an instantiation of an end-to-end MonB5G Network Service, that is, containing MLaaS several technological domains (e.g. Cloud, Edge/MEC and RAN). The figure highlights the existence of two end-to-end network slices, denoted as SFL-1 and SFL-2. These Network Slice Instances have components (i.e. SFL) at each technological domain, therefore are also accompanied by a corresponding management layer. In the case of the figure, each technological domain hosts a MLaaS managing both SFLs (i.e. NSSIs). Relevantly, IDMO in the Cloud domain holds a so-called Umbrella PaaS Controller, which is used to provision MLaaS across technological domains, respectively. Also, it is worth mentioning that DSF is independent NFV Objects and is included at SFL in each technological domain just to highlight the services offered to such layer.

[18] https://github.com/kubernetes-sigs/kubefed

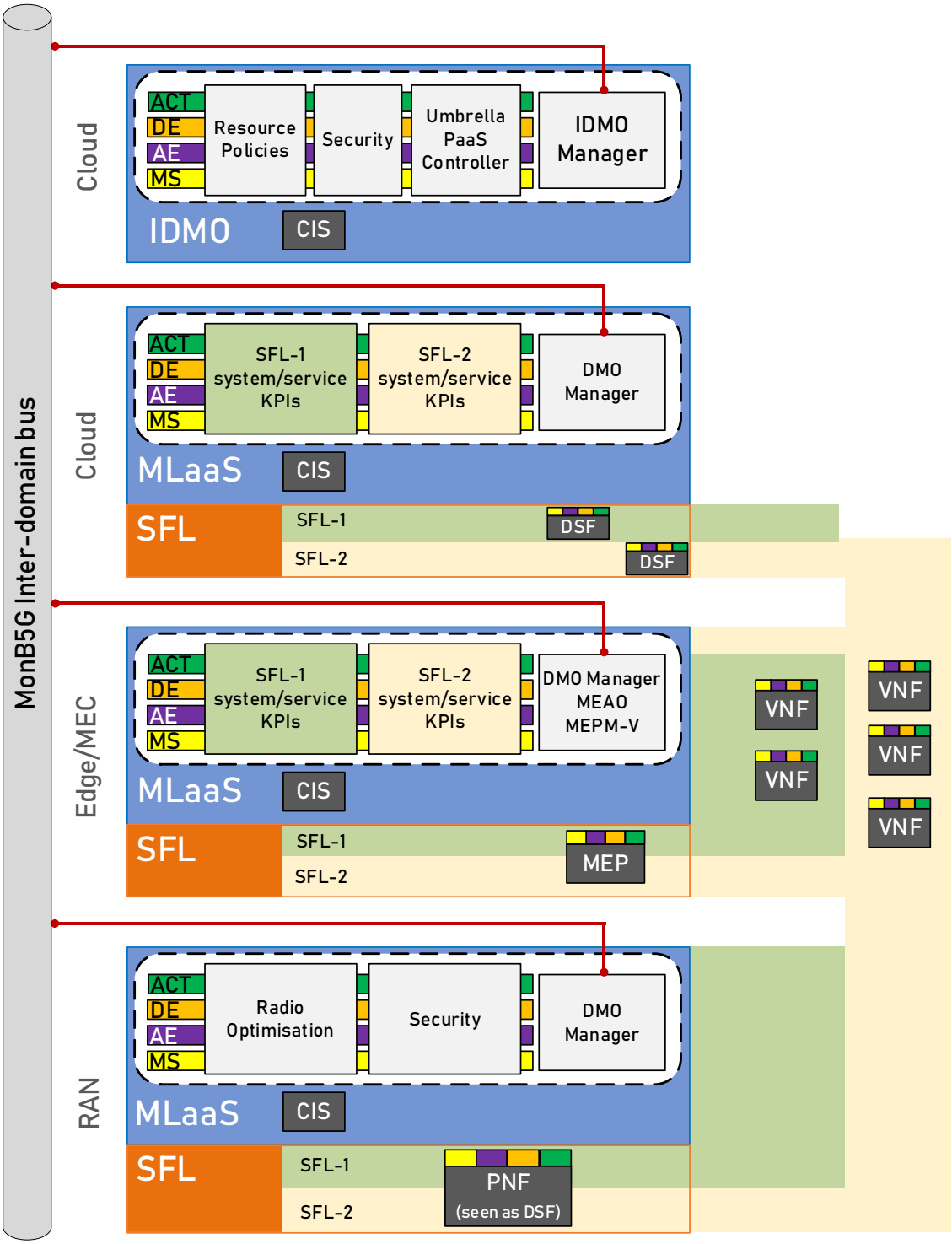

Figure 40 An end-to-end network slice with MLaaS used at several levels

The shown instantiation allows fast control loops at the VNF level by leveraging MLaaS and EEM, while wider-scoped and therefore slower control loops are exercised at the Slice (RAN, Edge and Cloud technological domains) and OSS/BSS (multi-slice) level.

6.4 Energy-aware service dynamics

As a consequence of the advent of virtualization into the telco world, a big part of 5G and beyond 5G communication infrastructure will be composed of data centres spanning the Cloud and Edge technological domains. Either for hosting VNFs, enabling URLLC slices, or hosting virtual RAN functions, the capabilities brought by virtualization define the road forward.

Energy-efficiency is a primary key performance indicator for the sustainability of beyond 5G networks. In this regard, network resource management algorithms should achieve the best quality of service (QoS) with minimum energy consumption. This involves network-level and slice-level strategies that require architectures with more flexibility/programmability in resource placement and allocation.

In this context, both SDN and NFV technologies are envisioned as the appropriate platform to deploy optimization models (based either on AI or heuristic approaches) and management functions enabling energy-aware network slicing operations. Based on energy consumption estimations or network parameters information (e.g. traffic load, radio coverage, equipment activation intervals, or active users), the SDN/NFV architectural framework can carry out actions such as optimized routing of traffic flows or allocation of physical (networking, computing, and storage) and/or dynamically scale-in/out Virtual Network Functions (VNFs) to meet the desired energy savings for each slice [97]. Having said that, a set of conceptual requirements should be met:

- Priority order for services within a network slice in the case the available energy at a certain time is insufficient to meet all demand. Based on this level of priority, the infrastructure provider must be able to differentiate the access to the energy resource and the application of management strategies for each service [52][103][104],
- Services belonging to network slices must be able to work in energy-saving states (e.g. low energy consumption) or suspend their execution in case of low activity (i.e. enter into sleep or idle mode) according to the availability [52][105][106],
- Measurement of metrics or indicators to assess the energy efficiency achieved with the proposed energy management model [107]. This includes the available power (managed by the Infrastructure provider) to be fed to the AI models for making decisions.
- Energy-aware decisions (e.g. VNF placement, scaling, idle modes) taken by AI algorithms (generally Deep Reinforcement Learning schemes) should be enforced via MANO API calls.

In this framework, several solutions have emerged recently. Specifically, in the context of a cooperative multi-operator, 5G network based on virtualized radio access and core, a sleep-mode and spectrum-sharing strategy to minimize the base station (BS) power consumption has been presented [98]. The proposed dynamic inter-operator spectrum–sharing formulation is cognizant of inter-RAN traffic volume to motivate mobile network operators (MNOs) to cooperate to achieve energy efficiency in their RANs. In this intent, the inter-operator joint optimization problem is formulated to obtain power-efficient intra- and inter-RAN beamforming vectors for supplementary energy gains and improved UE signal reception. Meanwhile, a distributed Q-learning algorithm has been introduced [100], which chooses how deep a BS can sleep according to the best switch-off sleep mode (SM) level policy that maximizes the trade-off between energy savings and system delay and may reach an energy saving of 90% when users are delay tolerant. Moreover, as cell load impacts its energy-efficiency, a multi-agent online reinforcement learning-based traffic offloading algorithm has been introduced [101], which benefits from the awareness about other macro-cells offloading strategies to improve the quality of the selected traffic offloading action without explicit information exchange. This yields a 14% improvement in network energy-efficiency. In the same direction, a joint energy-aware deep Q-network (DQN) traffic offloading and demand forecasting strategy has been presented [102], which leverages an open dataset from a major telecom operator to train BSs' control model leading to 5% energy-efficiency gain compared to native Q-learning.

Leveraging Network Function Virtualization (NFV), energy-efficient Integer Linear Programming (ILP)-based dynamic network functions placement has been proposed [99], which can adapt the joint locations of DU/CU and MEC to the actual distribution of network processing and transport resources so that to aggregate DUs/CUs into fewer cloud servers, resulting thereby in 20% energy saving. Further research into the Edge domain tends to favour small form-factors due to their lower energy consumption (mostly due to the physical restriction imposed by the deployment locations). These are mainly the reasons why Multi-access Edge Computing (MEC) platforms are coming from the OS container (e.g. Docker, Container Orchestration Engines for the Edge like KubeEdge, etc.) direction, instead of that from the traditional Virtual Machines (VM), whose LCM is delegated to big software, such as OpenStack.

It is then in the Cloud Technological Domain where the potential for energy-savings strategies could thrive, mainly due to the wide availably of (i) tools inherited from decades of data centre management (e.g. VIM plugins for energy quotas), (ii) protocols (e.g. Preboot Execution Environment (PXE)), and (iii) the possibility to redistribute allocated resources conditioned by service-level KPIs via NFV MANO reference points and Objects (e.g. PaaS).

MonB5G Architecture essentially generalizes the elements contained within its Functional Layer. That is, MonB5G management layer can be tuned for new models of infrastructure management. Leveraging policies dictated at the Inter-Domain level, the entire collection of Network Services (NS) on a determined Infrastructure Domain may be subject to MonB5G Administrative Elements; the latter being configured towards, e.g. efficient resource placement and compatible data centre energy management strategies.

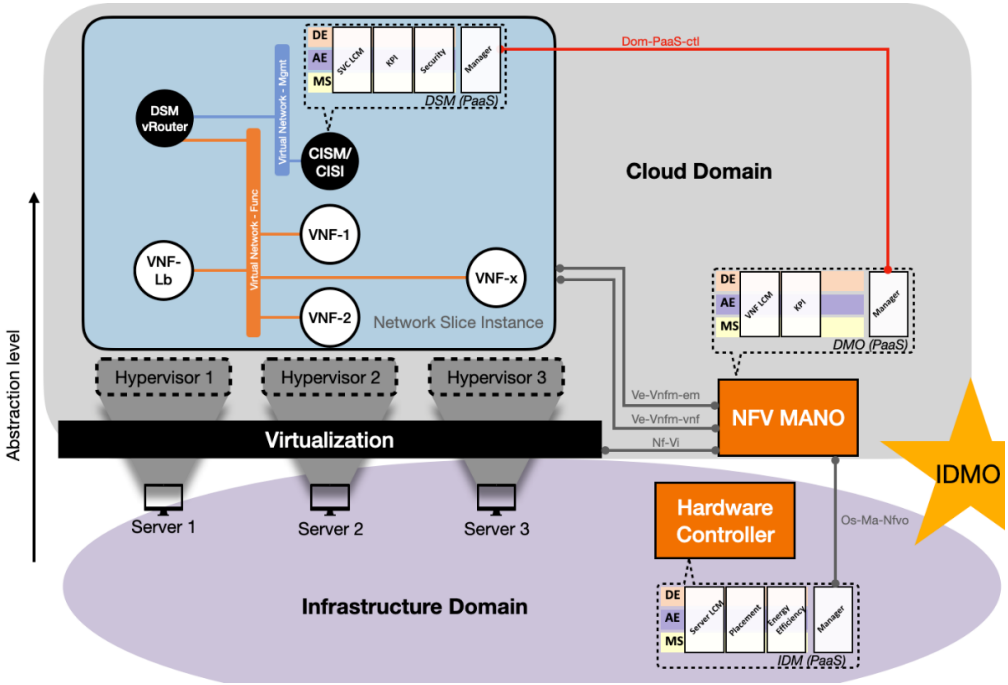

Figure 41. DE interactions

Figure 41 serves as a first step towards the implementation of a data-driven energy management system leveraging MonB5G Architecture, i.e. the orchestration capability of IDM. This solution will leverage APIs and operations provided by data centre management tools, e.g. MaaS, FOG Project or OpenStack Ironic to control as well as extract relevant metrics from a pool of servers (i.e. Hardware Domain in the figure), i.e. Hardware Controller. Furthermore, MonB5G Administrative Elements from the Infrastructure Domain Manager (IDM) can then extract relevant metrics from Hardware Controller, while enabling very valuable inter-domain optimizations such as dynamic Network Functions Virtualization Infrastructures (NFVI).

Figure 41 exemplifies this by showing Cloud and Infrastructure Domains compatible with MonB5G Architecture, while the corresponding SML deals with service-level optimizations and life cycle management. In turn, DMO leverages SML for resource-oriented optimizations or policy enforcement operations (e.g. VNF migration, VNF placement, NS admission control, etc.). Relevantly, Infrastructure Domain Manager (IDM) may predict valuable overall energy savings if alternative placement directives or resources migration is executed and therefore allow for the powering off of Servers at the Infrastructure Domain. The converse would also be true: failing to admit a Network Slice due to resource insufficiency, IDM may be instructed by other Technological Domain's DEs (e.g. DMO) to increase the pools of Servers in the Infrastructure Domain, and consequently the Virtualization capacity of the NFVI.

7 Conclusions

In this deliverable, the initial MonB5G zero-touch slice management and orchestration framework that facilitates the deployment of the massive amount of slices in different administrative and technological domains is presented. In order to achieve the scalability goal, we have used a distribution of AI-driven management functions at any level of the management hierarchy. The distributed management reduces the amount of the information exchanged for management purposes and taking some decisions locally reduces the management system response time. The use of distributed components with embedded intelligence has made it possible to use the intent-based interfaces that also contribute to the reduction of the information exchange between management functions and subsystems. Moreover, we have used a multi-domain orchestration and separation of each slice's management from domain resource management. The use of the in-slice management concept has reduced the number of slice external interfaces and provides a perfect separation of the slice management plane that cannot be achieved in the 3GPP approach to network slicing management. The implementation of slice management as a part of a slice (i.e. a set of VNFs) provides higher scalability of slice performance and allows for the programmability of slice management services on the fly. That includes slice-related security services. The in-slice management can play the role of its slice modification request, typically based on slice-specific analysis, enabling that way proactive management operations and contributing to the agnostics of the slice orchestrator. In MonB5G, the slice orchestrator is mostly focused on domain resources and is linked with OSS/BSS that performs appropriate functions. The domain-based approach reduced the overall management traffic. In order to reduce it more, we have used the well-known, KPI-based approach to exchange for monitoring information between domains. In order to include resource brokering and energy-efficient operations, we have modified the existing interfaces between the infrastructure and other components of the architecture. The usage of PaaS is another novelty of the MonB5G architecture. The MLaaS approach is an excellent bridge between the existing concepts, like 3GPP one, and the self-managed slice as proposed by MonB5G.

The deliverable describes the initial version of the architecture. It will be further updated by the inclusion of the work of work packages devoted to the monitoring system (MS), analytic engines (AE) and the decision engines (DE). In the near future, we will also collect experience related to the implementation of the architecture. The updated version of the architecture will be included in D2.4.

8 List of Figures

9 References

[1] O. U. Akgul, I. Malanchini, and A. Capone, "Anticipatory resource allocation and trading in a sliced network", in ICC 2019 - 2019 IEEE International Conference on Communications (ICC), pp. 1–7, 2019.

[2] L. Sanabria-Russo, L. Righi, D. Pubill, J. Serra, F. Granelli, and C. Verikoukis, "LTE as a service: Leveraging NFV for realising dynamic 5G network slicing", in 2019 IEEE Global Communications Conference (GLOBECOM), pp. 1–6, 2019.

[3] J. Ordonez-Lucena, P. Ameigeiras, D. Lopez, J. J. Ramos-Munoz, J. Lorca, and J. Folgueira, "Network slicing for 5G with SDN/NFV: Concepts, architectures, and challenges", IEEE Communications Magazine, vol. 55, no. 5, pp. 80–87, 2017.

[4] U. Paul, J. Liu, S. Troia, O. Falowo, and G. Maier, "Traffic-profile and machine learning based regional data center design and operation for 5G network", Journal of Communications and Networks, vol. 21, no. 6, pp. 569–583, 2019.

[5] M. R. Raza, M. Fiorani, A. Rostami, P.¨Ohlen, L. Wosinska, and P. Monti, "Dynamic slicing approach for multi-tenant 5G transport networks [invited]", IEEE/OSA Journal of Optical Communications and Networking, vol. 10, no. 1, pp. A77–A90, 2018.

[6] C. Marquez, M. Gramaglia, M. Fiore, A. Banchs, and X. Costa-Pérez, "Resource sharing efficiency in network slicing", IEEE Transactions on Network and Service Management, vol. 16, no. 3, pp. 909–923, 2019.

[7] B. Han, V. Sciancalepore, D. Feng, X. Costa-Perez, and H. D. Schotten, "A utility-driven multi-queue admission control solution for network slicing", in IEEE INFO-COM 2019 - IEEE Conference on Computer Communications, pp. 55–63, 2019.

[8] V. P. Kafle, P. Martinez-Julia, and T. Miyazawa, "Automation of 5G network slice control functions with machine learning", IEEE Communications Standards Magazine, vol. 3, no. 3, pp. 54–62, 2019.

[9] V. Sciancalepore, K. Samdanis, X. Costa-Perez, D. Bega, M. Gramaglia, and A. Banchs," Mobile traffic forecasting for maximizing 5G network slicing resource utilization", in IEEE INFOCOM 2017 - IEEE Conference on Computer Communications, pp. 1–9, 2017.

[10] V. Sciancalepore, X. Costa-Perez, and A. Banchs, "RL-NSB: Reinforcement learning-based 5G network slice broker", IEEE/ACM Transactions on Networking, vol. 27, no. 4, pp. 1543–1557, 2019.

[11] E. Kapassa, M. Touloupou, and D. Kyriazis, "SLAs in 5G: A complete framework facilitating VNF- and NS- tailored slas management", in 2018 32nd International Conference on Advanced Information Networking and Applications Workshops (WAINA), pp. 469–474,2018.

[12] S. Zheng, Z. Gao, X. Shan, W. Zhou, and Y. Wang, "Tail latency optimized resource allocation in fog based 5G networks", in 2018 IEEE Symposium on Computers and Communications (ISCC), pp. 00249–00254, 2018.

[13] C. Gutterman, E. Grinshpun, S. Sharma, and G. Zussman, "Ran resource usage prediction for a 5G slice broker", Proceedings of the Twentieth ACM International Symposium on Mobile Ad Hoc Networking and Computing, 2019.

[14] M. Maule, P. Mekikis, K. Ramantas, J. Vardakas, and C. Verikoukis, "Real-time dynamic network slicing for the 5G radio access network", in 2019 IEEE Global Communications Conference (GLOBECOM), pp. 1–6, 2019.

[15] C. Luo, J. Ji, Q. Wang, X. Chen, and P. Li, "Channel state information prediction for 5G wireless communications: A deep learning approach", IEEE Transactions on Network Science and Engineering, vol. 7, no. 1, pp. 227–236, 2020.

[16] D. Clemente, G. Soares, D. Fernandes, R. Cortesao, P. Sebastiao, and L. S. Ferreira, "Traffic forecast in mobile networks: Classification system using machine learning", in 2019 IEEE 90th Vehicular Technology Conference (VTC2019-Fall), pp. 1–5, 2019.

[17] S. Xiao and W. Chen, "Dynamic allocation of 5G transport network slice bandwidth based on LSTM traffic prediction", in 2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS), pp. 735–739, 2018.

[18] X. Shen, J. Gao, W. Wu, K. Lyu, M. Li, W. Zhuang, X. Li, and J. Rao, "Ai-assisted network-slicing based next-generation wireless networks", IEEE Open Journal of Vehicular Technology, vol. 1, pp. 45–66, 2020.

[19] N. Ferdosian, M. Othman, K. Y. Lun, and B. M. Ali, "Optimal solution to the fractional knapsack problem for LTE overload-state scheduling", in 2016 IEEE 3rd International Symposium on Telecommunication Technologies (ISTT), pp. 97–102, 2016.

[20] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications", ACM Trans. Intell. Syst. Technol., vol. 10, Jan. 2019.

[21] J. Konecny, H. B. McMahan, F. X. Yu, P. Richtarik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency", in NIPS Workshop on Private Multi-Party Machine Learning, 2016.
[22] M. Mohri, G. Sivek, and A. T. Suresh, "Agnostic federated learning", in Proceedings of the 36th International Conference on Machine Learning (K. Chaudhuri and R. Salakhutdi-nov, eds.), vol. 97 of Proceedings of Machine Learning Research, (Long Beach, California,USA), pp. 4615–4625, PMLR, 09–15 Jun 2019.
[23] M. Duan, D. Liu, X. Chen, Y. Tan, J. Ren, L. Qiao, and L. Liang, "Astraea: Self-balancing federated learning for improving classification accuracy of mobile deep learning applications", in 2019 IEEE 37th International Conference on Computer Design (ICCD), pp. 246–254, 2019.
[24] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions", IEEE Signal Processing Magazine, vol. 37, no. 3, pp. 50–60, 2020.
[25] X. Yao, T. Huang, C. Wu, R. Zhang, and L. Sun, "Towards faster and better federated learning: A feature fusion approach", in 2019 IEEE International Conference on Image Processing (ICIP), pp. 175–179, 2019.
[26] W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y. Liang, Q. Yang, D. Niyato, and C. Miao, "Federated learning in mobile edge networks: A comprehensive survey", IEEE Communications Surveys Tutorials, pp. 1–1, 2020.
[27] R. Fantacci and B. Picano, "Federated learning framework for mobile edge computing networks", CAAI Trans. Intell. Technol., vol. 5, pp. 15–21, 2020.
[28] Y. Xiao, Q. Zhang, F. Liu, J. Wang, M. Zhao, Z. Zhang, and J. Zhang, "NFVdeep: Adaptive online service function chain deployment with deep reinforcement learning", in 2019 IEEE/ACM 27th International Symposium on Quality of Service (IWQoS), pp. 1–10,2019.
[29] N. C. Luong, D. T. Hoang, S. Gong, D. Niyato, P. Wang, Y. Liang, and D. I. Kim, "Applications of deep reinforcement learning in communications and networking: A survey", IEEE Communications Surveys Tutorials, vol. 21, no. 4, pp. 3133–3174, 2019.
[30] A. Othman and N. A. Nayan, "Efficient admission control and resource allocation mechanisms for public safety communications over 5G network slice", Telecommunication Systems, vol. 72, pp. 595–607, Dec 2019.
[31] Z. Luo, C. Wu, Z. Li, and W. Zhou, "Scaling geo-distributed network function chains: A prediction and learning framework", IEEE Journal on Selected Areas in Communications, vol. 37, no. 8, pp. 1838–1850, 2019.
[32] ITU-T, "TMN management functions", ITU-T Recommendation M.3400, Feb. 2000. [Online]. Available: https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-M.3400-200002-I!!PDF-E&type=items (Accessed: 08.02.2021)
[33] A. Bosneag and M. X. Wang, "Intelligent network management mechanisms as a step towards SG," 2017 8th International Conference on the Network of the Future (NOF), London, 2017, pp. 52-57, doi: 10.1109/NOF.2017.8251220.
[34] ITU-T, "Supplement 4: An eTOM primer", ITU-T Recommendation M.3050, Feb. 2007.
[35] TM Forum, "Core Frameworks Concepts and Principles, Business Process, Information and Application Frameworks", TMF GB991 V19.5.1, Dec. 2019.
[36] IBM, "Autonomic Computing White Paper: An architectural blueprint for autonomic computing", 3rd edition, 2006.
[37] AutoI, "Autonomic Internet", [Online]. Available: http://www.autoi.ics.ece.upatras.gr/ (Accessed 31.01.2021)
[38] BIONETS, "Bio-inspired Service Evolution for the Pervasive Age", [Online]. Available: https://www.bionets.eu/ (Accessed 31.01.2021)
[39] EFIPSANS, "Exposing the features in IP version six protocols that can be exploited/extended for the purposes of designing/building autonomic networks and services", https://secan-lab.uni.lu/efipsans-web/index.php.html (Accessed 31.01.2021)
[40] UniverSelf, [Online]. Available: http://www.univerself-project.eu/ (Accessed 31.01.2021)
[41] SEMAFOUR, "Self-Management for Unified Heterogeneous Radio Access Networks", [Online]. Available: http://www.fp7-semafour.eu/ (Accessed 31.01.2021)
[42] 3GPP, "Self-Organizing Networks (SON); Concepts and requirements", 3GPP TS 32.500 V16.0.0, Jul. 2020.
[43] 3GPP, "Self-Organizing Networks (SON) for 5G networks", 3GPP TS 28.313 V17.0.0, Dec. 2020.
[44] 3GPP, "Management and orchestration; Architecture framework", 3GPP TS 28.533, v17.0.0, Dec. 2020.
[45] Saini, A., Mishra, A., Sharma, A. K., Distributed Network Management Architectures: A Review. International Journal of Computer Applications, 68(3). 2013.
[46] Kidston, D., (2000). Distributed Network Management. DEFENCE RESEARCH ESTABLISHMENT OTTAWA (ONTARIO). Avaliable at https://apps.dtic.mil/sti/pdfs/ADA386172.pdf [Verified 28 January 2021].

[47] F. Christopher et al., "A framework for in-network management in heterogeneous future communication networks", IEEE International Workshop on Modelling Autonomic Communications Environments. Springer, Berlin, Heidelberg, 2008. p. 14-25.
[48] Clemm, A., Ciavaglia, L., Granville, L., & Tantsura, J. (2019). Intent-based networking-concepts and overview. Internet Engineering Task Force, Internet-Draft.
[49] Wang, M. et al. (2019). A YANG Data model for ECA Policy Management. Internet Engineering Task Force, Internet-Draft.
[50] NGMN, "Description of Network Slicing Concept", NGMN 5G Whitepaper v1.0, January 2016.
[51] NGMN, "5G End-to-End Architecture Framework", NGMN 5G Whitepaper v4.3.1, November 2020.
[52] 3GPP, "Service requirements for the 5G system", 3GPP TS 22.261 V18.1.1, Jan. 2021.
[53] 3GPP, "System architecture for the 5G System", 3GPP TS 23.501 V16.7.0, Dec. 2020.
[54] GSMA, "Generic Network Slice Template", Version 1.0, May 2019, [Online]. Available: https://www.gsma.com/newsroom/wp-content/uploads/NG.116-v1.0-4.pdf.
[55] ETSI, "Architectural Framework", ETSI GS NFV 002 V1.2.1, Dec. 2014.
[56] Prometheus, [Online]. Available: https://prometheus.io.
[57] Gnocchi, [Online]. Available: https://gnocchi.xyz.
[58] ETSI, "Security Specification for MANO Components and Reference points", GS NFV-SEC 014 V3.1.1 Apr. 2018.
[59] ETSI, "Report on Security Aspects and Regulatory Concerns", GS NFV-SEC 006 V1.1.1, Apr. 2016.
[60] ETSI, "Security Management and Monitoring specification", ETSI GS NFV-SEC 013 V3.1.1, Feb. 2017.
[61] ETSI, "Trust; Report on Certificate Management", GS NFV-SEC 005 V1.1.1, Jan. 2019.
[62] ETSI, "NFV Security;. Security and Trust Guidance", ETSI GS NFV-SEC 003 V1.1.1, Dec. 2014.
[63] ETSI, "NFV Security; Problem Statement", ETSI GS NFV-SEC 001 V1.1.1 Oct. 2014.
[64] ETSI, "Report on the Enhancements of the NFV architecture towards "Cloud-native" and "PaaS"", GS NFV-IFA 029 V3.3.1, Nov. 2019.
[65] ETSI, "Requirements based on documented scenarios", ETSI GS ZSM 001 v1.1.1, Oct. 2019.
[66] ETSI, "Reference Architecture", ETSI GS ZSM 002 v1.1.1, Aug. 2019.
[67] ETSI, "Experiential Networked Intelligence (ENI)", [Online]. Available: https://www.etsi.org/technologies/experiential-networked-intelligence.
[68] ETSI, "System Architecture", GS ENI 005 V1.1.1 Sep. 2019.
[69] ETSI, "ENI Definition of Categories for AI Application to Networks", GR ENI 007 V1.1.1, Nov. 2019.
[70] ETSI, "Generic Autonomic Network Architecture (An Architectural Reference Model for Autonomic Networking, Cognitive Networking and Self-Management)", GS AFI 002 V1.1.1., Apr. 2013.
[71] Q. Wang. et al., "SliceNet: End-to-End Cognitive Network Slicing and Slice Management Framework in Virtualised Multi-Domain, Multi-Tenant 5G Networks", pp. 1-5, 10.1109/BMSB.2018.8436800 (2018).
[72] L. Xu et al., "CogNet: A network management architecture featuring cognitive capabilities", pp. 325-329. 10.1109/EuCNC.2016.7561056.
[73] CogNet, "Deliverable D2.2: Cognet final requirements, scenarios and architecture", Apr. 2017, [Online]. Available:]https://ec.europa.eu/research/participants/documents/downloadPublic?documentIds=080166e5bfb37f81&appId=PPGMS
[74] 5G NormA, "Deliverable D1.4: Final project report and evaluation", [Online]. Available: http://www.it.uc3m.es/wnl/5gnorma/pdf/5g_norma_d1-4.pdf
[75] C. J. Bernardos, , B. P. Gerö, M. Di Girolamo, A. Kern, B. Martini, and I. Vaishnavi, "5GEx: realising a Europe-wide multi-domain framework for software-defined infrastructures". Emerging Tel. Tech., 27: 1271– 1280 (2016). doi: 10.1002/ett.3085.
[76] 5G-Monarch, "Project Summary", [Online]. Available: https://5g-monarch.eu/wp-content/uploads/2019/09/5G-MoNArch_761445_Final_Project_Report_v1.0_clean.pdf
[77] Matilda, "Deliverable D8.2: Publishable Final Report", [Online]. Available: https://www.matilda-5g.eu/index.php/outcomes.
[78] S. Kukliński *et al.*, "A reference architecture for network slicing", 2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft), Montreal, QC, 2018, pp. 217-221, doi: 10.1109/NETSOFT.2018.8460057.

[79] S. Kukliński and L. Tomaszewski, "DASMO: A scalable approach to network slices management and orchestration", NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium, Taipei, 2018, pp. 1-6, doi: 10.1109/NOMS.2018.8406279.
[80] A. de la Oliva et al., "5G-TRANSFORMER: Slicing and Orchestrating Transport Networks for Industry Verticals", in IEEE Communications Magazine, vol. 56, no. 8, pp. 78-84, August 2018, doi: 10.1109/MCOM.2018.1700990.
[81] 5G-Essence, "Embedded Network Services for 5G Experiences", [Online]. Available: https://www.5g-essence-h2020.eu/.
[82] G. Bianchi et al. "Superfluidity: a Flexible Functional Architecture for 5G Networks". Transactions on Emerging Telecommunications Technologies 27, (2016). 1178-1186. 10.1002/ett.3082.
[83] ONAP Wiki, "Data Collection, Analytics and Events (DCAE)", [Online]. Available: https://wiki.onap.org/pages/viewpage.action?pageId=1015831.
[84] OSM, "Open Source Mano", [Online]. Available: https://osm.etsi.org.
[85] OSM, "OSM Performance Management", [Online]. Available: https://osm.etsi.org/wikipub/index.php/OSM_Performance_Management.
[86] OSM, "OSM9 Hackfest", [Online]. Available: https://osm.etsi.org/wikipub/index.php/OSM9_Hackfest
[87] Magma, [Online]. Available: https://github.com/magma/magma.
[88] Rift, "Rift.Ware Overview", [Online]. Available: https://riftio.com/riftware/.
[89] Canonical, "Charmed OSM", [Online]. Available: https://charmed-osm.com.
[90] Whitestack, "WhiteNFV", [Online]. Available: https://www.whitestack.com/products/whitenfv/.
[91] ITU-T, "Overview of TMN Recommendations", ITU-T M.3000 (02/00), Feb. 2000.
[92] ETSI, "Management and Orchestration", ETSI GS NFV-MAN 001 V1.1.1, Dec. 2014.
[93] MonB5G, "Deliverable D2.2: Techno-economic analysis of the beyond 5G environment, use case requirements and KPIs", Dec. 2020.
[94] ETSI, "Report on Architectural Options", ETSI GS NFV-IFA 009 V1.1.1, Jul. 2016.
[95] NIST, Framework for Improving Critical Infrastructure Cybersecurity", [Online]. Available: https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.04162018.pdf
[96] 3GPP, "Security architecture and procedures for 5G System", 3GPP TS 33.501 V17.0.0, Dec. 2020.
[97] A.N. Al-Quzweeni *et al.,* "Optimized energy aware 5G network function virtualization", in *IEEE Access* , vol. 7, pp. 44939–44958, 2019.
[98] J. Opadere, Q. Liu, T. Han, and N. Ansari, "Energy-Efficient Virtual Radio Access Networks for Multi-Operators Cooperative Cellular Networks", in IEEE Transactions on Green Communications and Networking, vol. 3, no. 3, pp. 603-614, Sept. 2019.
[99] Y. Xiao, J. Zhang, and Y. Ji, "Energy efficient Placement of Baseband Functions and Mobile Edge Computing in 5G Networks", 2018 Asia Communications and Photonics Conference (ACP), Hangzhou, 2018, pp. 1-3.
[100] A. El-Amine, M. Iturralde, H. A. Haj Hassan and L. Nuaymi, "A Distributed Q-Learning Approach for Adaptive Sleep Modes in 5G Networks", 2019 IEEE Wireless Communications and Networking Conference (WCNC), Marrakesh, Morocco, 2019, pp. 1-6.
[101] I. AlQerm and B. Shihada, "Energy Efficient Traffic Offloading in Multi-Tier Heterogeneous 5G Networks Using Intuitive Online Reinforcement Learning", in IEEE Transactions on Green Communications and Networking, vol. 3, no. 3, pp. 691-702, Sept. 2019.
[102] C.-W. Huang and P.-C. Chen, "Mobile Traffic Offloading with Forecasting using Deep Reinforcement Learning", [Online]. Available: https://arxiv.org/abs/1911.07452 (Accessed: 15 Apr. 2020).
[103] 3GPP, "Management and orchestration; Concepts, use cases and requirements", 3GPP TS 28.530 V17.0.0, Dec. 2020.
[104] ETSI, "Report on Network Slicing Support with ETSI NFV Architecture Framework", ETSI NFV-EVE 012 V3.1.1, Dec. 2017.
[105] 3GPP, "Management and orchestration; Energy efficiency of 5G", 3GPP TS 28.310 V16.3.0, Dec. 2020.
[106] ITU-T, "Energy efficiency metrics and measurement methods for telecommunication equipment", ITU-T L.1310, Sep. 2020.
[107] 3GPP, "Telecommunication management; Study on system and functional aspects of energy efficiency in 5G networks", 3GPP TR 32.972 V16.1.0, Sep. 2019.