Intent-Driven Management for Multi-Vertical End-to-End Network Slicing Services

Min Xie*, Pedro Henrique Gomes†, Jörg Niemöller†, Jens Patrick Waldemar*
* Telenor Research, Telenor, Norway; † Ericsson Research, Ericsson, Sweden

***Abstract*—Moving forward from 5G to 6G, one critical open challenge is customer-driven and autonomous networks management. To enable autonomy in a large-scale, it is important to separate the customer-centric requirements from the detailed network- and resource-centric operations. *Intent* has been proposed as a means to abstract the expectations of services and hide the implementation details from customers. In a multi-vendor, multi-tenant, and multi-domain 6G network with end-to-end (E2E) network slicing, intents can convey requirements across different management layers to facilitate network autonomy. In this paper, we propose intent-driven management (IDM) architecture that complies with the TM Forum autonomous network framework. Different from previously defined intents, the intent is classified into three types, *business intent*, *service intent*, and *resource intent*, which is useful to address the high complexity caused by assuring E2E network slicing services for multiple verticals. A complex E2E use case is presented to demonstrate how the IDM architecture and principles are practically applied to enable the intent-driven autonomous management for *multi-vertical* and *cross-operator* service provision and assurance.**

I. Introduction

6G is expected to be featured with technology heterogeneity, customer diversity, and environment dynamicity. As the mobile network evolves from 5G to 6G, two open questions continue to challenge the network operators: *automation* and *close interactions with vertical industries*. The management and operations in 6G will be significantly more complex and thus mandate automation. Furthermore, with a deep engagement in the overall service provisioning and assurance loop, the verticals expect their requirements to be explicitly addressed. However, due to the lack of expertise in networking, it is difficult for them to specify their service requirements in a format complying with how the network service is provisioned. A essential step towards network autonomy is to simplify the service specification.

Compared to *policy*, which is a set of rules (*e.g.*, in the format of an "event-condition-action" statement) that specify **how** a goal is achieved and often interpreted into executable configurations for specific network components, *intent* focuses on **what** is the goal, without specifying how the goal should be achieved. Therefore, intent is more abstract and agnostic to the underlying network components or technologies. It is a means to represent and communicate service *expectations* of different parties to enable a higher level of autonomy and abstraction in the management of the entire network [1]. Intent-based networking (IBN) was proposed in 2015 to specify north-bound interfaces (NBIs) of software-defined network (SDN) controllers [2]. It aimed to remove the complexity of service requests and simplify the integration between SDN and other systems. Over the last years, intent has been explored in academia, industry, open-source projects and standardization organizations, positioned as one key feature of the 6G architecture principles and an enabler for B5G/6G service management and orchestration [3].

The definition of intent evolved from *high-level policy* [4] to *goal-centric requirements* [1]. Currently, as agreed in standardization bodies of TM Forum, 3GPP SA5, and ETSI ZSM, intent is defined as *"the formal specification of all expectations, including requirements, goals, and constraints given to a technical system"* [1]. An intent is a set of distinct expectations, *e.g.*, a service or product delivered to customers with speciific functional and non-functional requirements. The requirements could be *quantitative* (key performance indicators (KPIs), service level agreement (SLA) metrics) or *qualitative* (legal compliance, security, system behaviour), *high-level business-oriented* or *low-level technology-oriented*. Intent excludes mandating implementations like policies or workflows. Instead, it aims to *abstract* the products, services, and resources provided by different parties. The details of management and operations are encapsulated in a way that only *outcomes* are exposed to the intent owner. Accordingly, the requirements and implementation details are explicitly separated. Network technologies like virtualization, softwarization, and cloudification are capabilities that facilitate abstraction and are key to realizing *intent-driven management* (IDM). Their advancement in 5G makes IDM more realistic in 6G research.

In this paper, we *propose an IDM architecture* to automate the management of E2E network slicing services, complying with the TM Forum guidelines [1]. Considering a three-layer service provisioning model, intent is defined at three layers, *business intent*, *service intent*, and *resource intent*, to distinguish the expectations from different parties. At each layer, intent is handled by intent management functions (IMF) through an iterative *five-step cognitive loop*, which is showcased by a proof of concept (PoC) for the adaptive virtual network function (NF) placement. A practical use case (UC), co-created with multiple vertical customers in EU projects, is presented to demonstrate the application of the IDM architecture and intent modeling to a broader scope of E2E and cross-operator services.

The paper is organized as follows. Section II presents related work on the topic of IDM. Section III introduces the IDM architecture, including information modeling and intent management interfaces. Section IV provides an implementation approach of the core component of the IDM architecture. A UC is described in Section V, with detailed intent objects. Section VI concludes with open challenges and future work.

II. Related work

The research on intents for networks is in an early stage, focused on the intent representation and deployment, including modeling and translation [5]. Benefiting from the rapid advances in natural language processing (NLP) and natural language understanding propelled by IT and cloud giants, these two areas have been explored from various perspectives, *e.g.*, construct semantics graphics through NLP to understand, interact, and create the required network service as intent [6], eliminate ambiguity, convert intents (primitive verb composition, sequence-to-sequence learning), and translate intents into policy descriptions. However, in most of the literature, intent is still policy-centric, *e.g.*, a set of policy rules guiding the service behaviour, analytics, and closed loop events for elastic service management in [7]; or a high-level security policy that does not demand expertise in special network security functions [8]. Intent translation usually targets direct mapping from customer-facing business intent to a policy description of various management and operations processes, such as network [7] or device configuration (network intent composition (NIC) project under OpenDaylight), service configuration (service composition mapping, service chain configuration, or VNF placement), or traffic forwarding rules [9]. Such translation is specific to certain processes in certain network domains (*e.g.*, network function virtualization (NFV) and SDN). In the end-to-end (E2E) network, the existing direct intent translation solutions are not sufficient to deal with the high complexity caused by multiple vendors and multiple network domains.

For intent handling, both [10] and [11] proposed to create a dedicated platform in their IBN architecture. The former inserted an intent management layer between the business and network layers to receive business intents from customers, translate them into network operation policies, and assure their fulfilment. The latter used a centralized intent controller to decide the policies for intent fulfilment whereas other management entities (*e.g.*, slice orchestration) only need to execute the recommended policies. In the E2E service management, [7] proposed an E2E service orchestration solution in a *bottom-up* manner, where each network domain (core network (CN) and transport network (TN)) produces an abstracted representation of its performance and behaviour, which are correlated with the intent.

At present, intent is mainly applied to the NFV and SDN network although it could be potentially expanded to the E2E network. As concluded in [5], there is a need to develop a unified framework with a clear definition of intent and its management. Moreover, as 5G (and 6G) networks operate in a multi-vendor, multi-tenant, and heterogeneous network environment, intent is needed to interface not only with customers, but also across various network domains and layers.

III. IDM Architecture

A. Architecture

Following the TM Forum generic reference framework for autonomous networks within a delimited management scope (a.k.a. autonomous domain) in [1], we propose an IDM architecture specific to E2E network slicing services in Fig. 1. A network slice can be seen as a logical network, composed of transmission, network functions, and applications [12]. Typically, an E2E network slice can be decomposed into multiple network services, offered by individual network domains, such as radio access network (RAN), transport network (TN), and core network (CN). An intent is associated with two parties, *intent owner* that creates and manages the intent, as a holder of the intent requirements; and *intent handler* that fulfills the requirements within a autonomous domain. Aligned with the three-layer provisioning model of E2E slicing services, the IDM architecture consists of three layers, each of which defines its own intent: *business intent* from customers, *service intent* towards the network service operation layer, and *resource intent* for resource operations of individual network domains, including RAN, CN, and TN. The intent-driven operations vary with the intent type. The high-level business intent contains customer-specific requirements that are too abstract to map into the domain-specific resource operations. They are usually *translated into service intents* which are closer to the resource operations. The lower-level resource intent is detailed and could be *directly mapped* to specific network configuration policies. To fulfill the *requirements* received in the intent, the intent handler contains an intent management function (a.k.a. *intent manager* [1]) that coordinates the intent-driven operations and assures the requirements. Intent manager is the core component of the IDM architecture. It is updated with the current system state through *measurements*, including different data collection and analytic systems. Based on the updated state, the intent manager either produces *actuation* (*e.g.*, configuration policies) to be executed by certain management functions or translates the requirements into a new *intent* to be handled by another autonomous domain. The status/outcome of the fulfillment process will be *reported* back to the intent owner. Therefore an intent manager could be both *intent owner* and *intent handler*.

B. Intent Modeling

The content of an intent is characterized by *intent models*, which create information objects directly consumed by machines. Intent modeling has formal semantics allowing machine reasoning and logic inference. Since intents are exchanged across multiple layers and domains, intent models should provide *domain-specific expressiveness*. For

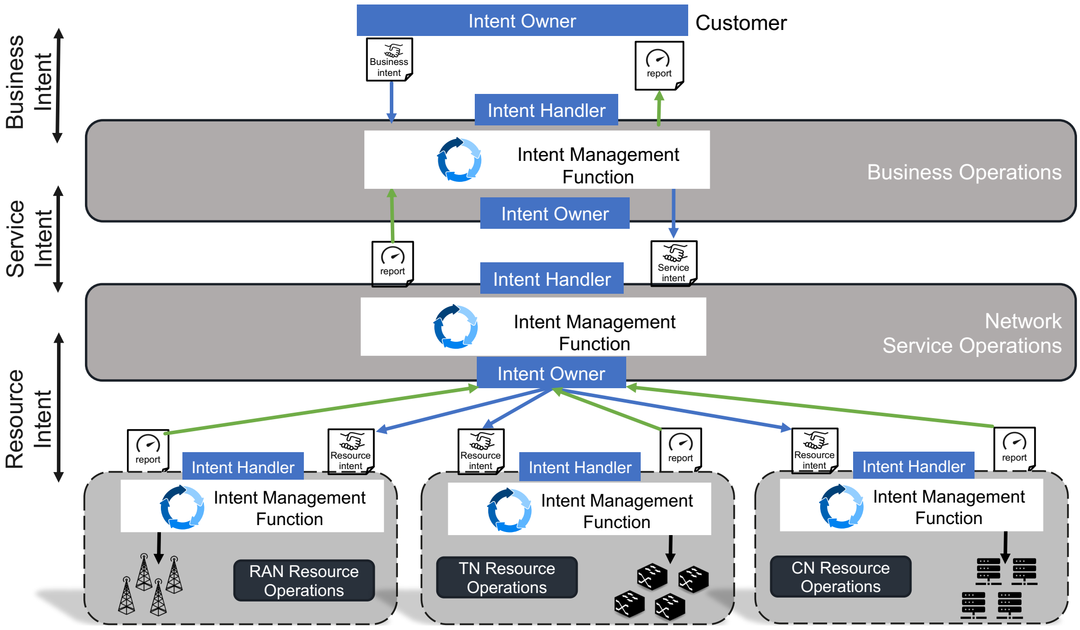

Figure 1: Intent-driven management architecture for 5G network slicing

instance, business intents focus on user requirements and financial results whereas service and resource intents are more into the technology-specific requirements, such as coverage, QoS, availability, slice configurations, etc. Besides, the contents of these intents are highly domain-specific (*e.g.*, KPIs/metrics). On the other hand, incompatible models and interfaces should be avoided as they require huge efforts on intent translations. Consequently, intent modeling targets at combining *common expressiveness* applicable across different management domains, and *domain-specific details*. A pragmatic information modelling approach is proposed by TM Forum [1], known as *intent model federation*. It consists of two parts: a *common intent model* and *domain-specific extensions*. The former provides the constructs for intent and intent report expressions that contain generic and domain-independent aspects. For example, all intents are formulated as a collection of objects about requirements and goals. The latter adds vocabulary and semantics as extensions of the common model, which can be independently developed by other SDOs or organizations. This model federation approach can be used to develop intent objects for both E2E network slices and specific network domains of RAN, TN, and CN, as exemplified in Section V. This approach, which is standardized by TM Forum and can be further extended by other standardization bodies and companies, allow an intent created by an intent owner to be understood by intent handlers developed independently by different vendors or different network operators.

Intent model federation plays an important role in enabling cross-vertical integration. The common intent model can be extended from telecommunications to other industries and technologies domains. Accordingly, intent can communicate and distribute requirements across vertical domains. As a result, it is potentially a universal base for autonomous cross-vertical operations.

C. *Intent handling interface*

A common intent handling interface is fundamental to support intent-based interactions across layers or domains. Unlike conventional communication interfaces, which are usually used to invoke processes, the intent handling interface is primarily used for the lifecycle management (LCM) of intents and related reports (see Fig. 1).

More importantly, the interface should be independent of the intent handling scope and can be reused by all intent manager [1]. It supports various operations between the intent owner and the intent handler, both *mandatory* and *optional*. The mandatory operations take care of the intent LCM, based on the CRUD (Create, Read, Update, Delete) pattern that can be mapped to RESTful API method, which is commonly adopted by modern management systems and standardization. The life cycle of the intents includes *detection* of the need to create/change/remove an intent, *investigation* of feasible intents, *definition* of intents, *distribution* of intents from the owner to the handler, and *operation* of domain-specific responsibilities according to the given intent. The optional operations address advanced features such as intent negotiation and owner-handler collaboration, which require advanced capabilities of intent managers like predictive and speculative models (*e.g.*, predict the consequence of executing an intent and evaluate the predicted results, then decide whether this intent could be accepted or rejected). Currently, a standardized intent management interface is under development by TM Forum in specification TMF921A [13].

IV. Implementation of Intent Manager

As the core component of the IDM architecture, the intent manager should be implemented flexibly, mainly depending on the capabilities of the corresponding management layers or domains. One typical way of implementing intent managers is rule- and policy-based, which is ideal for simple domains with relatively stable environments. However, for complex networks like 6G, such implementation is not sufficient due to the lack of adaptivity to dynamic scenarios. Therefore, we propose a more adaptive implementation approach for networks with a dynamic environment and diverse services.

Leveraging cognitive processes, the intent manager can be implemented in an iterative five-stage cognitive loop, as shown in Fig. 2: 1) measurement, 2) issues, 3) solutions, 4) evaluation, and 5) actuation. At each stage, a set of tasks is executed by software-based autonomous agents, which can be simple tools available in management systems, or advanced AI/ML-based agents.

Measurement is the first stage, through which the intent manager captures the system state with respect to the metrics expressed in the intent requirements. The measurement results are used to report and evaluate if the intent is fulfilled or corrective actions are needed. Measurement is realized by the *measurement agents*, which are adapted to the intent changes over time, *e.g.*, by reconfiguring existing measurement agents or deploying new measurement agents.

Issues are raised when the measurement results indicate that the intent is not fulfilled, *e.g.*, a required performance or quality of experience metric is not reached, or a required service is not available. Then corrective actions are needed. Various *service assurance agents* are employed to analyze the measurement results, detect and raise issues, then diagnose the issues to understand the root causes, and prioritize issues based on the severity level.

Solutions are proposed as a response to the detected issue. They determine the available strategies and action plans to correct the issues. Different strategies can be selected to determine the solutions and then implemented by specialized agents. For example, a solution could be either manually designed policies and rules captured from human expert solutions; or machine-learned policies derived from historical data. Multiple solutions may be proposed to solve the same issue, by deploying multiple agents in parallel.

Evaluation decides which available solutions are the most preferential and deliver the most positive effect on all valid and active intents. Based on an assumption that a solution cannot be trusted until it is independently checked as a fit, the intent manager predicts the consequences of executing the proposed solutions through a set of *evaluation agents*. Therefore *evaluation* is an important sanity-check process and could potentially prevent the network from carrying out risky actions that may lead to service degradation or disruptions. Predictive machine learning techniques and digital twins could be applied at the evaluation stage to find the hypothetical system state and predict the outcomes of every available solution via a virtual execution of the actions. Evaluation also includes conflict detection when an action fulfills one intent but breaks another intent.

Actuation executes the approved solutions after the evaluation stage. A series of specialized *actuation agents* (*e.g.*, orchestrator, SDN controller) are instructed to take actions, such as reconfigure network/resources directly if the responsible domain can act, or create new intents and pass to the subordinate domain if the responsible domain cannot act.

Through this cognitive implementation process, the intent is fulfilled by the intent manager in an autonomous way. One PoC has been conducted to *translate intents* across the three layers in a CN.

From business intents to service intents, the translation is based on the service blueprints from onboarded service definitions. The translation is contextualized by interpreting and transforming the abstract business requirements (*e.g.*, service locations for individual verticals) into more concrete goals.

At the service operation layer, the intent manager delivers a service instance that fulfills the given intent, *e.g.*, by distributing applications and NF components over the network and cloud infrastructure. This process requires a description of the full topology of these artifacts. The context considered for the service intent could be implicit to specific verticals, or explicitly expressed as additional operator requirements, such as resource utilization, energy consumption, and cost, etc. To execute the solution, a service orchestrator coordinates the actions distributed in the CN to instantiate all required NFs.

In the PoC, we focus on cloud-native NFs, which are handled by the intent manager at the resource operation layer. The resource-layer proposal agents determine solutions, mainly identifying all atomic deployment function artifacts and allocating them to the best data centers. Deployment function candidates are technology agnostic, but it is assumed that there is a direct mapping from the atomic function to some deployment artifacts understandable by the resource orchestrator. Note that the deployment function candidates may contain other functional dependencies, *e.g.*, on an operation and maintenance function. These dependencies could be recursively resolved until a set of atomic functions is identified to satisfy all dependencies and their connectivity requirements. The process also involves functional decomposition, *e.g.*, a gateway function is decomposed into user plane and control plane components.

Ultimately, the business intent is fulfilled by a service instance that is broken down to locations (such as data centers and transport paths), deployment artifacts (*e.g.*, represented by Helm charts), and initial resource allocation concerning bandwidth, QoS, or vCPU, memory, storage, etc. The results of this PoC can be transferred to UCs with a wider scope of E2E services.

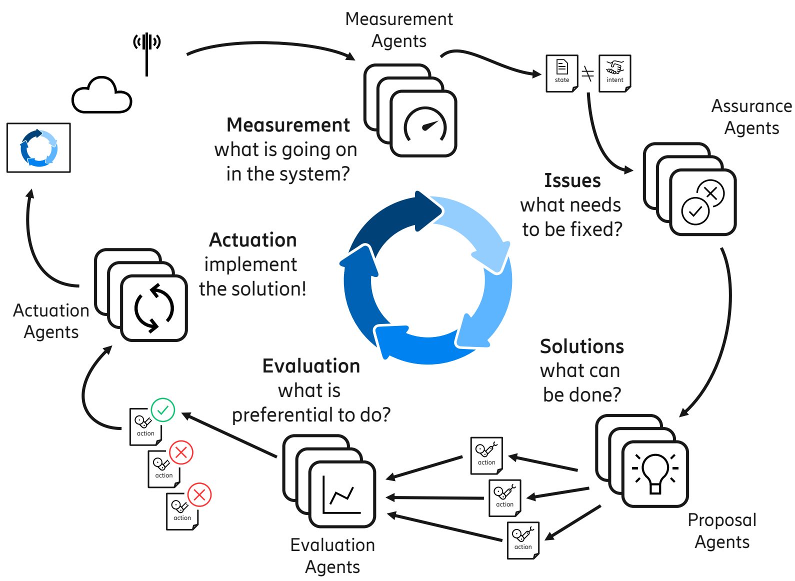

Figure 2: Implementation of an intent management function based on a cognitive loop

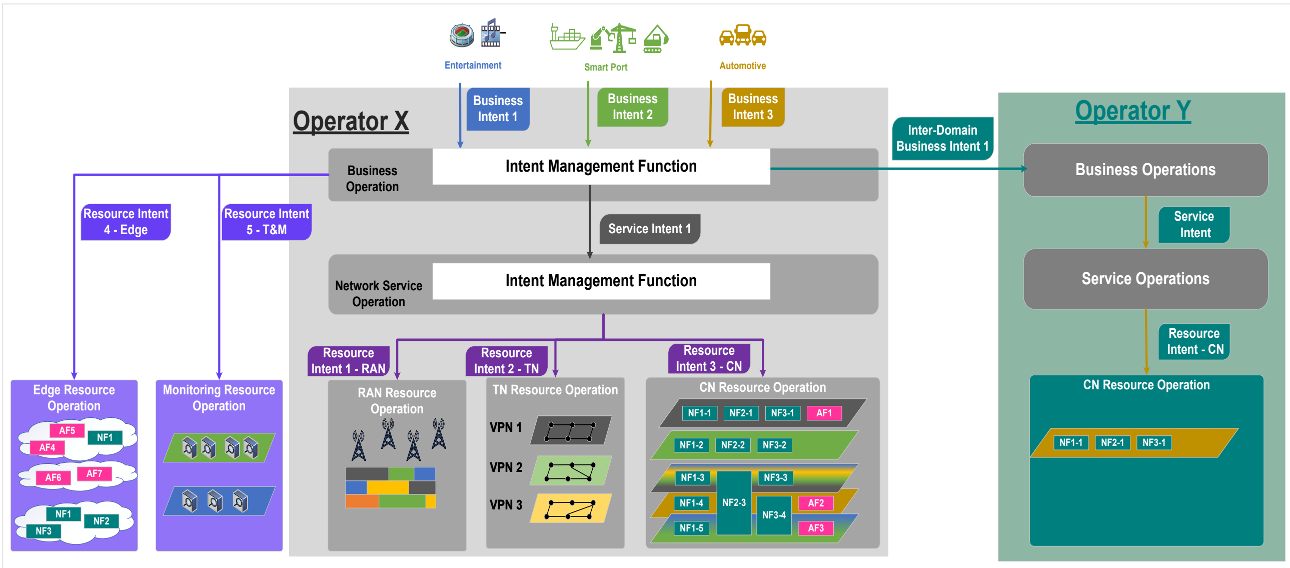

Figure 3: Intent-driven management for provisioning and assuring multiple vertical UCs

V. Use Cases

In several EU-H2020 projects (*e.g.*, 5G-SOLUTIONS and TeraFlow), we co-create multiple UCs with verticals to test the 5G network capabilities and validate the vertical services KPIs. One open issue is how to effectively interact with these UCs and design network services to meet their KPIs *concurrently*. Applying the proposed IDM architecture, we see the potential of improving and automating the service provisioning and assurance process for multiple verticals.

As shown in Fig. 3, Operator X receives multiple business intents created by multiple vertical consumers, including entertainment, smart port, and automotive, among others. Each vertical customer requires different SLAs and thus different intent expectations on *E2E services*. The *multimedia UC* produces content from multiple media sources and broadcasts it to the audience. Normally it shares an E2E network slice with other customers. If special events (concerts or sports) take place and the user experience (measured by the quality of experience (QoE)) is degraded, the UC dynamically demands a dedicated slice. The *smart port UC* aims to build a zero-emission and safe environment for smart production and delivery across multiple locations. It operates multiple scenarios, including i) real-time status tracking of a large number of containers between the main location and support locations; ii) remote control of autonomous vehicles via the surveillance cameras and positioning sensors; iii) high-speed data transfer from an unmanned autonomous ship to a public cloud; iv) port safety by using acoustic sensors on the machines and remote analytics. Besides, an edge site with distributed user plane function (UPF) is demanded to process high-volume data locally. The *automotive UC* not only requires high reliability and low latency but also spans across two countries, involving Operators X and Y.

The three business intents are represented in a **declarative** way, focusing on the expected outcomes like the business KPIs (QoE, time to market), or the contexts (cross-country, edge sites). At the Business Operation layer, the intent manager adds multiple business intents **additively**, analyzes and then creates four new intents: *Service Intent 1*, which is passed to the service operation layer of Operator X; *Inter-Domain Business Intent 1*, which goes towards the peer Operator Y; and two resources intents, namely *Resource Intent 4 - Edge* and *Resource Intent 5 - T&M*, which go to external service providers. The local service intent combines the requirements of all business intents and maps to six network slices with specific KPIs. The remote inter-domain business intent responds to the *automotive UC* by explicitly requiring a URLLC slice from Operator Y, which is treated as sub-slice B (the right side of Fig. 3) of a cross-operator E2E slice while the local service intent requires one URLLC slice as sub-slice A of the E2E slice. The multimedia UC requires an on-demand slice, which relies on real-time monitoring. The *monitoring resource intent* is sent to the monitoring management domain that controls and manages the deployment of monitoring tools, scheduling of test cases, and retrieval of monitoring data. The smart port UC requires an edge, which triggers the creation of *edge resource intent* to the edge resource operation that deploys the edge infrastructure and onboards the required NFs and/or applications.

The service operation layer of Operator X further analyzes the received service intent and flexibly prioritizes and optimizes the intent fulfillment. Although the service intent requires six slices, it is not cost-efficient to create six slice instances. In practice, network operators optimize the resource utilization, *e.g.*, by exploring resource sharing inherited in network slicing. As a result, the service intent manager decides to provision four slices with two shared slices (dark purple intents in Fig. 3) that can meet the requirement of the three business intents.

Since the four slices are provisioned by underlying network domains via resource intent, the service intent is **decomposed** into three resource intents to be processed by the resource operation layer of RAN, CN, and TN, respectively. The resource intent is **technology-agnostic** that allows each network domain chooses its way to fulfil the intent. For example, TN slicing can be implemented by VPN or traffic engineering. In the CN domain, the orchestration system can further explore resource sharing and select a subset of NFs to be shared by multiple slice instances [14]. Note that all the implementation details are **encapsulated** by the intent handler and hidden from the intent owner.

This UC shows flexible intent exchanges between autonomous domains in a hierarchical architecture. Applying *intent model federation*, Fig. 4 illustrates the excerpts of specific intent objects for the UC. Given a broad range of applications in knowledge management, Resource Description Framework (RDF) and the related RDFS (RDF Schema) and OWL (Web Ontology Language) are used as a modeling framework and Turtle as the serialization language, as recommended by TM Forum [1].

In *Business Intent* 1, Lines 1 to 3 point at all existing data models and assign different namespaces, which is the basis for combining multiple information models in the model federation approach. These prefix lines start with a globally unique identifier, *e.g.*,"*icm*", "*ent*", "*tel*". All elements starting with "*icm*" use the common intent model of TM Forum [1], describing the expectations and the associated targets (*e.g.*, in services or resources) and parameters. All elements starting with "*ent*" (defined in Line 2 as an unique identifier of Business intent 1) represent models created by the entertainment vertical. "*tel*" represents models created and governed by SDO1, which has a general scope of telecommunications. These models extend the common model "*icm*", *e.g.*, by providing specific KPIs definitions in the expectations. The business intent object starts with Line 4 (which identifies *Business intent 1* as "*EntertainmentIntent001*") and uses RDF triples. Overall, this intent has 3 expectations, namely $E1$, $E2$ and $E3$ (line 5). $E1$ requires the delivery of an on-demand streaming service referred to "entertainmentService" (Line 11). $E2$ expects the business KPIs of QoE and time to market (Lines $19 - 20$) whereas $E3$ specifies technical KPIs like throughput, latency, and UE density (Lines $26 - 29$). Given the defined three expectations, all business requirements can be interpreted by the intent manager at the business layer, which are translated into service intent 1 or resource intents $1 - 5$ (Fig.4).

VI. Conclusions

In this paper, we propose an IDM architecture to meet the demands of multi-vendor, multi-tenant, and multi-domain next generation (6G) mobile networks. The IDM architecture provide guidance for communicating and handling service requirements from vertical UCs, especially in the context of *cross-operator* E2E services with multiple concurrent verticals. A PoC and a complex UC are provided to verify the application of the IDM architecture in current 5G networks for the intent manager and intent modeling, respectively. However, since the study on intent is in an early stage, there are many open challenges to be addressed for effectively deploying intents in 6G.

First, *how multiple intents from different verticals can be optimally combined and translated into network service intents*. This question requires optimization mechanisms to optimize *intent addition* and *intent decomposition* while avoiding conflicts and maximize the resource utilization. The related research topics include slice design, slice sharing, and slice isolation. Second, *how the IDM architecture can be implemented, with standardized or open interfaces*. Integrating the IDM architecture with the legacy and 5G/6G

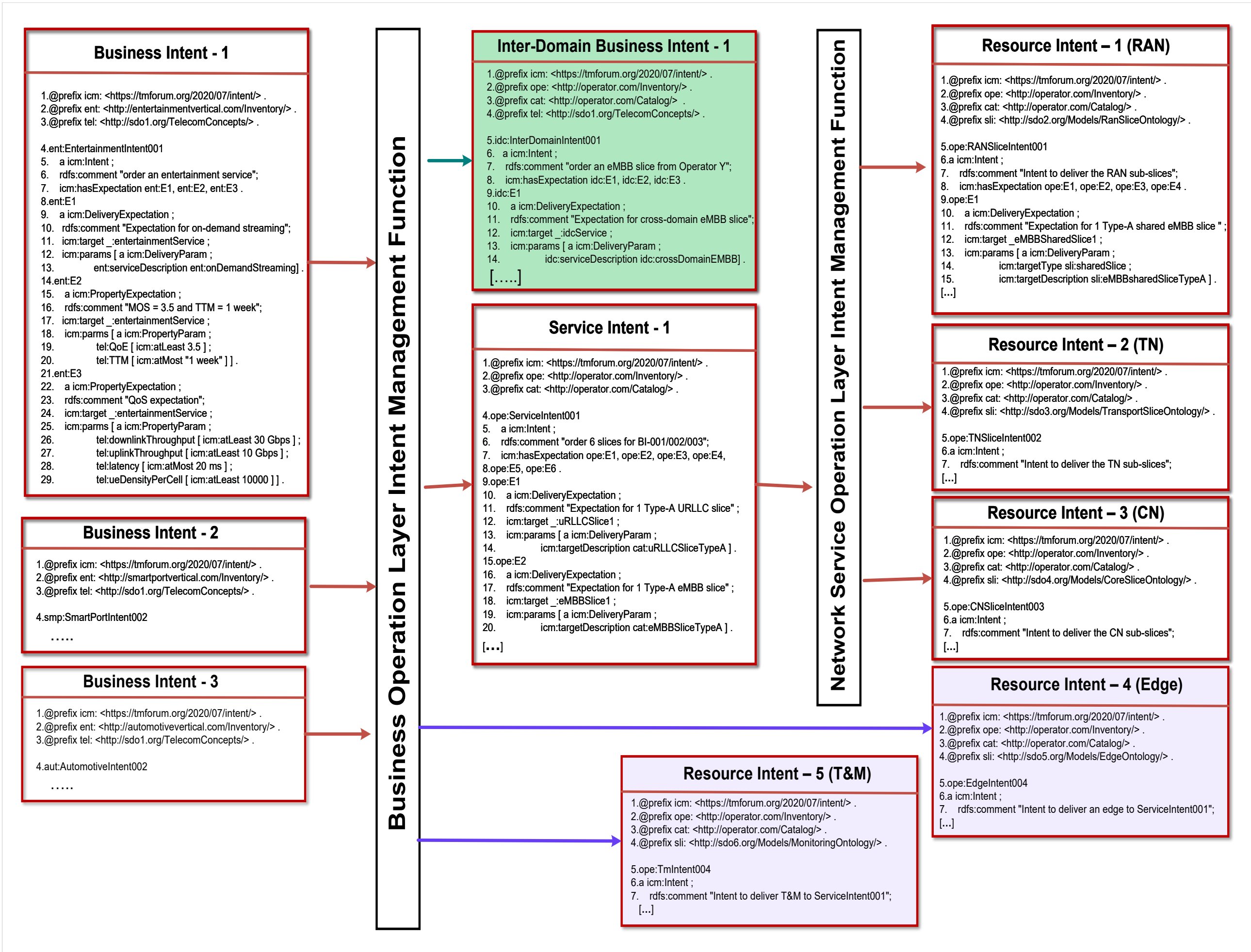

Figure 4: Examples of intent objects in the use case

networks poses critical challenges from an implementation point of view, as we have experienced in the 5G study. Third, *intent handling and management* is complicated and demands highly close cooperation between AI/ML and networking, which has not been practiced in large-scale platforms. These questions will be investigated in the future work through research projects, standardization contributions, and PoC experimentation.

Acknowledgment

This work has been supported by the European Community through the H2020-ICT-19-2018 5G-SOLUTIONS project (grant 856691) and the H2020-ICT-52-2020 TeraFlow project (grant 101015857).

References

[1] T. Forum, "Intent in Autonomous Networks," TM Forum, Tech. Rep. IG 1253, 06 2021, version 1.1.0.

[2] Open Networking Foundation, "TR-523: Intent NBI – Definition and Principles," Oct. 2016.

[3] Hexa-X, "D5.1 - Initial 6G Architectural Components and Enablers," Dec 2021.

[4] IETF, "Autonomic Networking: Definitions and Design Goals (RFC7575)," 2015.

[5] L. Pang, C. Yang, D. Chen, Y. Song, and M. Guizani, "A Survey on Intent-Driven Networks," *IEEE Access*, vol. 8, pp. 22 862 – 22 873, 2020.

[6] M. Kiran, E. Pouyoul, A. Mercian, B. Tierney, C. Guok, and I. Monga, "Enabling Intent to Configure Scientific Networks for High Performance Demands," *Future Generation Computer Systems*, vol. 79, pp. 205 – 214, 2018.

[7] L. Velasco *et al.*, "End-to-End Intent-Based Networking," *IEEE Communications Magazine*, pp. 106 – 112, Oct. 2021.

[8] J. Kim *et al.*, "IBCS: Intent-Based Cloud Services for Security Applications," *IEEE Communications Magazine*, pp. 45 – 51, Apr. 2020.

[9] A. Leivadeas and M. Falkner, "VNF Placement Problem: A Multi-Tenant Intent-Based Networking Approach," in *24th Conference on Innovation in Clouds, Internet and Networks (ICIN2021)*, 2021.

[10] E. Zeydan and Y. Turk, "Recent Advances in Intent-Based Networking: A Survey," in *IEEE Vehicular Technology Conference (VTC)*, May 2020.

[11] Y. Wei, M. Peng, and Y. Liu, "Intent-based Networks for 6G: Insights and Challenges," *Digital Communications and Networks*, vol. 6, pp. 270 – 280, 2020.

[12] J. Ordonez-Lucena *et al.*, "Network Slicing for 5G with SDN/NFV: Concepts, Architectures and Challenges," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 80–87, May 2017.

[13] TM Forum, "TMF921A: Intent Management API Profile."

[14] C. Mei, J. Liu, J. Li, L. Zhang, and M. Shao, "5G Network Slices Embedding with Sharable Virtual Network Functions," *Journal of Communications and Networks*, vol. 22, no. 5, pp. 415 – 427, 2020.