An Intent-Driven Orchestration of Cognitive Autonomous Networks for RAN management

Anubhab Banerjee*+, Stephen S. Mwanje*, and Georg Carle +
*Nokia Bell Labs, Munich, Germany
+Dept. of Informatics, Technical University of Munich, Germany
Email:*anubhab.banerjee@tum.de, stephen.mwanje@nokia.com, carle@net.in.tum.de*

Abstract—**Intent Based Networks (IBNs) are mainly used to transform a user's intent into network configuration, operation, and maintenance strategies. In this paper we propose an a generic end-to-end design of an intent based network management system which we further customize to use in RAN control parameter and KPI management utilizing an existing technology (Cognitive Autonomous Network (CAN)). We introduce three new concepts in intent based network management: intent specification platform (ISP), formal intent and Intent Fulfillment System (IFS), which are not only relevant for this specific use case but can also be used by other network components. We discuss how our proposed solution can be implemented in Python as a standalone module so that it can be used with suitable networking simulators. Along with these, we also provide an overview of standardization impact of our research to show that it conforms with the worldwide mobile network management standardization efforts.**

Index Terms—**CAN, IBN, IDN, SON**

I. Introduction

Intent Based Networking (IBN) has attracted a lot of attention in recent years as a promising automation tool for next generation networks. From the perspective of a human mobile network operator (MNO), an intent expresses the expectation of MNO from the network regarding service operations. An intent is ideally expressed declaratively, i.e., as a utility-level goal that describes the properties of a satisfactory outcome rather than prescribing specific ways to achieve that goal. This presents the system with the opportunity to explore various solution options before landing on the optimal one. Intents are gaining popularity in the field of network management and automation because it makes the job of MNO easier. In radio access networks (RAN), intents have already been used for network slicing and resource management [1], [2]. However, no focus has been given on managing radio network control parameters (like, cell transmit power (TXP), remote electrical tilt (RET)) through an intent based system. Introduction of AI replaces rule-based self-organizing network (SON) with Cognitive Autonomous Networks (CAN) [3], which replaces rule-based SON Functions (SFs) with learning based cognitive functions (CFs). Although CAN provides significant improvement over SON in terms of operational complexity and maintenance [3], [4], CAN has not been designed for interaction purposes with MNO.

To enable MNO interact with CAN in real time, we provide the design of an end-to-end system for managing control parameters and KPIs through an intent based system. The intents are translated into some commands or instructions which are then executed by CAN. Through this proposed design, MNO also gets an overview of the possible outcomes when these intents are executed by CAN, which enables MNO to modify her intents if necessary. This type of intent based management provides flexibility in network operations and makes the job of MNO easier. Along with the end-to-end system design, in this paper our contributions are the following:

- We propose a generic intent-driven network architecture useful for any type of network and customize it further to interact with CAN. It is important to note here that the proposed design works equally with SON and CAN.
- To the best of our knowledge, we are the first ones to present an end-to-end architecture for an intent driven management of RAN control parameters and KPIs, and, also propose interactions with CAN (or SON).
- We propose three new network components: intent specification platform (ISP), intent fulfillment system (IFS) and intent-driven network automation function orchestrator (IDNAFO),each of which plays an important role in intent driven network management. The idea of ISP, IFS and formal intent is not only relevant for RAN but for other types of networks as well.
- We implement our proposed design in Python 3.7 which shows our proposed schematic can be implemented and used in a real life scenario.
- We also discuss the standardization impact of our proposed work and show that our proposed work covers crucial standard aspects and it conforms with the worldwide mobile network management standardization efforts.

II. Related Works

A. Intent Based Networking (IBN)

In this Section, we present a brief overview of how IBN was generated and developed over the last five years. CISCO introduced an IBN system in 2017 which comprises of three fundamental building blocks: translation, activa-

978-3-903176-36-2 ©2021 IFIP

tion and assurance [5]. Following that, Huawei introduced a similar system called Intent-Driven Network (IDN) [6], aiming to build a next generation network. Apart from CISCO and Huawei, several others defined and developed IBN over the past few years ([1], [2], [7]). Since we discuss intent based RAN control parameters management in this paper, our focus is on existing intent driven solutions for RAN management.

B. Intent Based Resource Allocation in RAN

There exist a lot of research works proposing intent based slicing mechanisms in RAN. An open source orchestrator named OSM, proposed by ETSI, is designed for slicing networks based on an NFV specifications [8]. The Linux Foundation also provides an open source E2E network orchestrator of virtual appliances named ONAP [9]. Apart from the orchestrators, [10] described the network slicing framework and the basic design challenges faced for performing RAN slicing in 5G.

The closest research work, to the problem addressed in this paper, is the solution for slicing RAN resources with FlexRAN controller [11]. FlexRAN runs in a gNB, allocating resources to different slices in a dynamic environment. However, in this paper, we propose an intent based management of RAN parameters using CAN whereas FlexRAN is designed for RAN slicing. To the best of our knowledge, we are the first ones to propose an intent driven CAN (and SON).

C. Existing Standards

The 3rd Generation Partnership Project (3GPP), European Telecommunications Standards Institute (ETSI), Open Networking Foundation (ONF) and International Telecommunication Union (ITU) have all developed their own study groups on intent-based networking. The work on TR:28.812-"Study on scenarios for Intent driven management services for mobile networks" in Release 16 in the scope of SA5 is 3GPP's effort on intent-based network management is still an ongoing effort [12]. This asserts the concept of IDM where an Intent Driven Management Service (IDMS) is provided to consumers to manage 5G network and services. Utilization of intent driven management service is envisioned to originate from communication service providers and network operators in the considered scenarios. Some of the considered scenarios are related to intent driven service deployment, network provisioning, network optimization, coverage and capacity management - which are exactly the scenarios discussed in this paper.

ETSI has initiated the Zero-touch network and Service Management (ZSM) working group [13] for describing means for network automation in 2018. The document provides details on automation in network management and also concentrates on policy-driven automation, intent-based automation as well as intent-based service orchestration. ITU-T Study Group 13 explores intent as a declarative mechanism (written in ML meta-language) where technology agnostic ML use case can be deployed by operators inside their focus group on Machine Learning for Future Network including 5G (FG-ML5G). In that sense, intents are used as high level ML pipeline components. However, building this meta-language is also foreseen as one of the main challenges in future implementation of intent-based networking.

III. PROBLEM DESCRIPTION AND SOLUTION FRAMEWORK

A. Problem Description

In a cell there are several adjustable control parameters (e.g., cell transmit power (TXP), Remote Electrical Tilt (RET), Cell Individual Offset (CIO)) and observable key performance indicators (KPIs), (e.g., downlink throughput, radio link failures (RLF)). A KPI value can change if one or multiple control parameters are changed, for example, downlink throughput can be modified by changing TXP and RET. On the other hand, changing a single control parameter might affect several KPIs simultaneously, for example, changing CIO might affect both RLF and cell load. This kind of mutual dependency among control parameters and KPIs makes the management of the network difficult.

It is important to note that all the KPIs are crucial in determining the quality of service (QoS) of the network. If MNO wants to make multiple simultaneous changes to certain control parameters, MNO might not be able to predict accurately how these changes might affect the KPIs. Also, if MNO wants to achieve certain KPI related targets, she might not possess the complete knowledge about the changes required in the control parameters. So, we see that if MNO wants to customize the network to meet certain service requirements, she has to learn the mutual dependencies among the control parameters and KPIs.

The job of MNO becomes easier if there is a layer of abstraction between MNO and RAN where she specifies her intentions and the layer of abstraction generates the appropriate actions to be executed in the network and provides her with the possible outcomes. This layer of abstraction gives more control to MNO over the network - before taking any actions on the network, MNO can get an overview of possible outcomes of her action. However, designing such a layer of abstraction is not an easy task - not only the layer has to know the details of mutual dependency among the control parameters and KPIs, but also needs to have the ability to predict network performance under certain conditions in advance to provide an overview of possible consequences of actions by MNO. In this paper we provide an end-to-end design of such a system for efficient management of RAN parameters.

B. Solution Framework

In this paper, we propose the generic design of an intent based management, which can be used for any type of network and further modify our proposed design for managing RAN control parameters and KPIs. It is to note

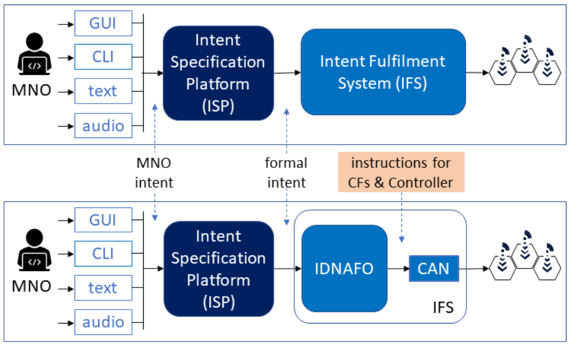

Fig. 1: Architectural overview of a generic intent-driven management and orchestration of CAN

that, the generic design can always be modified to develop intent based management system for any kind of network.

Our proposed design consists of two principal functional blocks: intent specification platform (ISP) and intent fulfillment system (IFS).

ISP: An intent from MNO can be inputted via any form: audio input, command line input (CLI), text, or, a graphical user interface (GUI) command. Main functionality of ISP is to convert an intent from MNO into a format understandable by concerning network components.

IFS: Main responsibility of IFS is to determine how an intent can be executed in the network. The output of ISP, which is understandable by different network components, is called a **formal intent**, which is used as input to IFS. Structures of intents from MNO may not be defined, whereas structure of a formal intent is predefined: it contains identification of the target objects and specific actions on these objects. IFS acts as a bridge between ISP and the network, and if necessary, IFS can further be divided into several functional blocks. In the context of this paper, IFS consists of two functional blocks: intent-driven network automation function orchestrator (IDNAFO) and CAN, both of which are covered in the next Section in detail.

IV. IMPLEMENTATION

In this Section, at first we give an overview of CAN and IDNAFO and discuss how the combination can be used for intent fulfillment purposes, then, we provide details regarding Python implementation of our proposed design.

A. CAN

In CAN, for each individual KPI, there is one closed-loop CF having the main responsibility of learning how the KPI changes when the control parameters are changed [4]. The control parameters, on which the KPI is dependent, are called input control parameters (ICPs) of the CF and the KPI is called its output. Based on the learning, the CF can determine the configuration (values of the ICPs) for which its output is optimal in a certain network state [14].

It is often found that multiple KPIs are influenced by a single control parameter. As each CF works independently, if each CF starts changing a certain control parameter according to its own will, it might affect the other CFs sharing the same parameter and stability of the whole system. So we assume a coordination mechanism on top of these CFs to adjust the control parameters and resolve conflicts among the CFs and throughout the rest of the paper we refer to it as the Controller. If a CF determines a new configuration to optimize its output, it conveys its intention to the Controller which determines a value optimal for the combined interest of all the CFs [4], [14], [15]. Without any intricate details, we can abstract a CAN in where the Controller lies between the CFs and the network as shown in Fig. 2 where F_1 and F_2 are two CFs with a shared ICP: p.

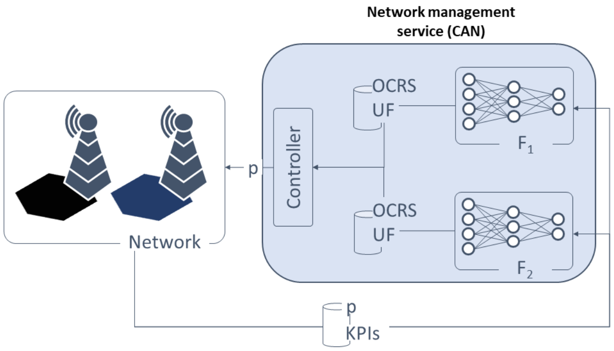

Fig. 2: CAN abstraction with the CFs and the Controller

From the overview of CAN, we see that CAN works in a closed-loop way - in a changing environment CAN always adjusts the network parameters to reach an equilibrium. Although CAN excels in managing network control parameters and KPIs, CAN does not provide an interface for interaction with the MNO for customization. To overcome this, we propose IDNAFO, through which MNO can interact with CAN to customize the network as needed.

B. IDNAFO

IDNAFO takes a formal intent as its input and generates appropriate actions for the Controller and CFs. Workflow of IDNAFO consists of three sequential steps, based on which we introduce three different functionality blocks (Fig. 3):

- After receiving a formal intent, its first task is to check if the intent can be executed by the Controller or CFs. We call this block Intent Identifier (II).
- After the II identifies that an intent is valid, the intent is then classified based on its content. This task is done by Intent Classifier (IC).
- Then, based on the type of the intent, IDNAFO sends specific commands to the Controller or CFs or both. This is done by Intent Decision Maker (IDM).

1) Intent Identifier (II): Only the intents, which deal with KPIs and/or control parameters, are relevant for CAN. An intent like "switch cell X off/on" is beyond the operational capability of CAN. The relevance of the intent is checked by II using the logic: if all the actions are defined

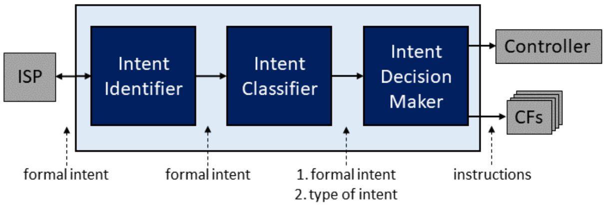

Fig. 3: IDNAFO functionality blocks

on KPIs and/or control parameters, then only the intent can be executed by CAN, else it is discarded.. If II finds the intent to be valid, it passes to IC for further processing, otherwise, II returns the intent back to ISP marking the intent as invalid.

2) Intent Classifier (IC): Based on by which an intent is to be executed, IC classifies a formal intent into three distinct categories following the three rules:

Rule 1: If all the actions are defined on network control parameters, then the intent is Type 1 intent.

Rule 2: If all the actions are defined on KPIs, then the intent is a Type 2 intent.

Rule 3: Rest all are Type 3 intents. Here actions are defined on both network control parameters and KPIs.

The corresponding categories may be grouped as described in Table I.

3) Intent Decision Maker (IDM): Based on the category of the intent, IDM decides what instructions are to be sent to the Controller and/or CFs.

Type 1: As it contains only control parameters, it is executed only by the Controller.

Type 2: To execute it, IDM identifies the CFs responsible for managing the KPIs and send instructions to those CFs to take specific actions on the KPIs. At the same time, instructions are also sent to the Controller to make all the changes proposed by those particular CFs.

Type 3: It contains both KPIs and network control parameters, so instructions are given to both CFs and the Controller. Instructions are sent to the specific set of CFs to take specific actions on their managed KPIs. At the same time, two types of instructions are sent to the Controller: (i) take instructed actions on control parameters, and, (ii) make all the changes in the network as proposed by those particular CFs.

C. Implementation of end-to-end design

To show that our proposed architecture can be used in a real life scenario, we implement the end-to-end design in a simulation environment and provide the implementation of each component in detail.

1) ISP implementation: ISP can be implemented in one of these two ways: (i) An intent from MNO does not necessarily follow a predefined structure, it can be just a wish of MNO expressed in valid English [1]. NLP can be used for parsing the language, for example, the algorithm proposed in [16] is a suitable candidate for use in this purpose. However, our focus in this paper is to provide an end-to-end design for intent driven CAN, so we do not concentrate on NLP based ISP or propose any new algorithm for implementation, rather we focus on how a formal intent can be executed by CAN. (ii) MNO expresses her intent in a predefined structure. In this case a rule based parsing algorithm can be used in ISP to convert the intent into formal intent. Both these methods of ISP implementation have some advantages and disadvantages. Method 1 gives more flexibility to MNO, but it is difficult to implement whereas Method 2 is easy to implement but provides less flexibility to MNO. For this paper we implement ISP using the second way and as a future work, we plan to implement ISP in the first way.

2) IDNAFO implementation: We use Python 3.7 to implement IDNAFO in a simulation environment. Using the logic mentioned in Section IV-B1 II can determine if an intent is relevant for CAN. After implementing II, we implement IC following the Table I. Corresponding to each type of intent, we initialize a set of commands which are to be sent to the Controller and/or CFs. These commands have predefined structures and dynamically generated from the formal intent. This standalone module does not have any simulator specific dependency and can be integrated on top of suitable networking simulators for demonstration.

3) CAN implementation: We already discussed implementation details of CAN in our earlier research papers [4], [14] and we use the same simulation scenario used in [14]. We deploy two CFs - mobility load balancing (MLB), and, capacity and coverage optimization (CCO) together with a Controller [17]. The environment of the simulations is an authentic recreation of a small part of the city of Hamburg using a Nokia Bell Labs internal simulator which has already been used by researchers extensively [14], [18], [19]. Other details regarding simulator, CFs and CAN deployment can be found in [14] and we did not reproduce the same text here.

4) Observation: The simulator automatically updates the changes done by the CAN module and reconfigures the radio processes, and thus, in this way the whole system remains up and running all the time just like a real life network. Since no intent database for RAN management is currently available, we create a database with 20 intents by ourselves, test and observe that all of them can be executed using our proposed design. Average execution time is 0.12 ms, which proves that our proposed design works fast and worthy for real life use. If an intent cannot be fulfilled by a CF, we also implement a direct feedback mechanism so that MNO can modify the intent accordingly.

V. Impacts on Standardization

All ongoing standardization activities take intents as goals specified at a high level without any specifications on how to execute the concrete actions. In ETSI ZSM and ETSI ENI, intents can also be expressed by human language to be later translated into models that can be

TABLE I: Categorization of intents

Type	Description	Example intents
Type 1	actions defined on only network control parameters	increase cell X TXP, make cell X CIO remain constant, etc.
Type 2	actions defined only on KPIs	reduce cell X interference, avoid increase in cell X interference, etc.
Type 3	actions defined on both control parameters and KPIs	make cell X TXP constant and reduce interference, etc.

interpreted by machines, similar to the function of the ISP. A major challenge in standardization documents is the representation of intent in terms of language and model specifications, which still remains an open question [7]. Our proposal avoids this challenge and instead raises new standardization requirements.

It is desirable that any real-life IDNAFO deployment supports multi-vendor integration. To support this, input and output interfaces of the IDNAFO need to be specified in network management specifications e.g. in 3GPP SA5 or ETSI ZSM specifications. Although 3GPP SA5 already provides means for controlling the behavior of NAFs by configuring the goals, it does not provide means for configuring any such controller, so the 3GPP network resource model needs to be extended with models for CAN control functionality and the means to configure such functionality.

On the other hand, the IDNAFO takes a formal intent based on which it determines if and how to fulfill the intent. Such a formal intent may be generated and compiled by an ISP coming from any non-telco vendor like a machine-learning audio processing startup. To allow for integration between these non-telco-centric ISPs and the telco-centric IDNAFO, the intent specification interface and specifically the structure of formal intent need to be standardized. This is actually an in-extensive extension to existing standards since the formal intent's attributes and their values are already speechified in the existing specifications. For example, managed objects, control parameters and metrics are already specified in the 3GPP network resource models. Additionally, the intent specification interface needs to be extended with failure and success events in the messages indicating that 1. "Intent not in CAN scope" and 2. "Report for execution success / failure".

VI. Conclusion and Future Direction

IBN or IDN plays an important role in network management automation in next generation networks. Although there have been quite a few research papers published in IBN and IDN, majority of them provide an abstract overview instead of any implementation details. We are the first ones to present an end-to-end architecture for a generic intent driven management system and its modification for CAN (SON) orchestration. We implemented our proposed design in Python to show it can be used in a real life scenario. We also discussed the standardization impact of our proposed work and showed it covered crucial standard aspects. As a next step, we plan to study the conflict scenarios arising in an intent driven network management.

References

[1] L. Pang, C. Yang, D. Chen, Y. Song, and M. Guizani. A survey on intent-driven networks. *IEEE Access*, 8:22862–22873, 2020.

[2] Y. Wei, M. Peng, and Y. Liu. Intent-based networks for 6g: Insights and challenges. *Digital Communications and Networks*, 6(3):270–280, 2020.

[3] S. S. Mwanje and C. Mannweiler. *Towards Cognitive Autonomous Networks: Network Management Automation for 5G and Beyond*. John Wiley & Sons, 2020.

[4] A. Banerjee, S. S. Mwanje, and G. Carle. Game theoretic conflict resolution mechanism in cognitive autonomous networks. In *2020 International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS)*, pages 1–8. IEEE, 2020.

[5] T. Szigeti, D. Zacks, M. Falkner, and S. Arena. *Cisco Digital Network Architecture: Intent-based Networking for the Enterprise*. Cisco Press, 2018.

[6] Huawei. Intent-driven network (idn). Available online: https://carrier.huawei.com/~/media/CNBG/Downloads/Spotlight/all-cloud-network-towards-5g/idn-en.pdf.

[7] E. Zeydan and Y. Turk. Recent advances in intent-based networking: A survey. In *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*, pages 1–5. IEEE, 2020.

[8] OSM. Open source mano. Available online: https://osm-download.etsi.org/ftp/Documentation/201902-osm-scope-white-paper/#!02-osm-scope-and-functionality.md.

[9] ONAP. Onap: Open networking automation platform. Available online: https://www.onap.org/.

[10] R. Ferrus, O. Sallent, J. Pérez-Romero, and R. Agusti. On 5g radio access network slicing: Radio interface protocol features and configuration. *IEEE Communications Magazine*, 56(5):184–192, 2018.

[11] Mosaic5G. Network sharing using flexran. Available online: https://mosaic5g.io/flexran/.

[12] 3GPP Technical Specification Group Services and System Aspects. Study on scenarios for intent driven management services for mobile networks, telecommunication management. https://bit.ly/2vJ2aOr, 2019.

[13] ETSI. Zero touch network and service management (zsm) means of automation. https://bit.ly/2THSfRl, 2018.

[14] A. Banerjee, S. S. Mwanje, and G. Carle. Optimal configuration determination in cognitive autonomous networks. In *2021 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, pages 494–500, 2021.

[15] A. Banerjee, S. S. Mwanje, and G. Carle. On the necessity and design of coordination mechanism for cognitive autonomous networks. arXiv:2001.07031, 2020.

[16] J. Bogojeska, D. Lanyi, M. Botezatu, and D. Wiesmann. eznl2sql: A system for network devices management with a natural language interface for databases. 2021.

[17] Anubhab Banerjee, Stephen S. Mwanje, and Georg Carle. Towards control and coordination in cognitive autonomous networks. *IEEE Transactions on Network and Service Management*, pages 1–1, 2021. doi: 10.1109/TNSM.2021.3116308.

[18] A. Banerjee, S. S. Mwanje, and G. Carle. On the implementation of cognitive autonomous networks. In *Internet Technology Letters. 2021;e317*, 2021. doi: 10.1002/itl2.317.

[19] A. Banerjee, S. S. Mwanje, and G. Carle. On detection of manipulative cognitive functions in cognitive autonomous networks. In *2021 17th International Conference on Network and Service Management (CNSM)*, 2021.