CLARA: Closed Loop-based Zero-touch Network Management Framework

Nathan Franklin Saraiva de Sousa
University of Campinas (UNICAMP)
Sao Paulo, Brazil
nsaraiva@dca.fee.unicamp.br

Christian Esteve Rothenberg
(Supervisor)
University of Campinas (UNICAMP)
Sao Paulo, Brazil
chesteve@dca.fee.unicamp.br

***Abstract*—Evolving 5G systems are adopting strategic technologies such as SDN, NFV, Network Slicing, and Intent-based Networking to provide flexibility, programmability, and more advanced services. Such environments are characterized by heterogeneous infrastructures, dynamic behavior, and growing demand for new users, along with increasing management complexity of networks and services. Accordingly, automation is crucial for reducing management complexity while minimizing the need for human intervention. Towards a fully automated end-to-end network and service management approach, key enablers include intents for representing the user needs, and policy-based closed control loops (CCL) to enable self-x properties. This work contributes to specific advances in using policy-based management to govern the CCL behavior and fulfill user requirements in a multi-domain environment. Specifically, we cover service management from network service requests to the enforcement of policies across management domains. To embed all, we propose a CCL-based Zero-touch Network Management Framework named CLARA. Implementation of prototypes and functional evaluation presented in previous works demonstrate the feasibility of the proposed components.**

I. Introduction

Traditionally, end-to-end (E2E) services management consists of manual and long processes resulting in prolonged lead times until effective service delivery. Relevant technologies such as Software Defined Networking (SDN), Network Function Virtualization (NFV), Intent-based Networking (IBN), and Network Slicing bring advances in virtualization and network softwarization field leveraging flexible, more sophisticated services offering. Such technologies compose the foundation to the 5G and beyond networks providing flexibility and programmability in a software-centric paradigm [10].

However, the network softwarization landscape increases management complexity since the network environments have heterogeneous infrastructure composed of physical and virtual elements, dynamic behavior with constant and frequent changes, as well as ever-growing connectivity demand [3]. Besides, the emerging services demand diversified and sophisticated requirements hard to fulfill them through manual processes. Therefore, the network automation capabilities become essential to provide efficiency in the management process reducing error-prone configurations and raising the agility of service deployment [13]. The automation process needs to go beyond simply automating well-known tasks but provide a fully automated E2E network environment, so-called zero-touch networks, with minimal human intervention and massive use of data analytics and AI-based methods.

In this context, network management performs a essential role in controlling the lifecycle of E2E services. After the network service (or network slice) is deployed, the major challenge is continuously monitoring its performance across all its execution whilst fulfilling SLA requirements. This way, the closed control loops (CCLs) [26] emerge as key enablers for management automation, providing capabilities of monitoring, analyzing, planning, and executing on managed entities [20]. CCLs can deliver the self-x properties such as self-healing, self-configuration, and self-optimization. In multi-domain environments, where the network services span different domains, network management becomes highly complex and even more challenging. Toward reducing the management complexity, IBN employs network abstractions (*i.e.*, intent) automating the management process. In a high-level way, these abstractions or intents represent the user needs and business requirements afterward converted into network policies in line with expressed goals. In general, policies can detect and react to changes in the context in a self-managed manner when inserted in a closed control loop approach [17]. The behavior of control loops can be controlled using a policy-based workflow where managed entities are adjusted to a specific goal in an automated way.

In order to enable a fully automated E2E network and service management, a zero-touch architecture should include a set of key features: *(i)* use of intent to represent the user's need or business requirements, *(ii)* conversion of intent into declarative policies, *i.e.*, high-level policies, *(iii)* enforcement of policies into CCL, *(iv)* use of machine intelligence to make decisions and optimize resources, and *(v)* support to multi-domain scenarios. In a multi-domain environment, each administrative domain can have one or more CCLs with specific goals that should work collaboratively. Figure 1 illustrates an overview of the E2E multi-domain scenario composed of four domains: Cloud, Transport, Core, and RAN. E2E CCL coordinates the interaction between different CCLs to achieve a desired state.

To address the identified requirements and challenges [26], ETSI established the Zero-Touch Network and Service Man-

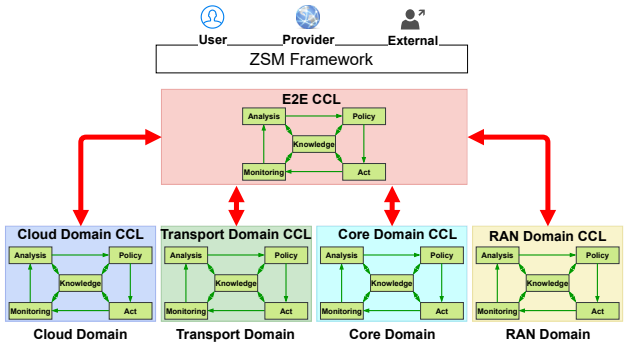

Fig. 1: Overview of an E2E architecture composed of multiple autonomous CCLs organized hierarchically.

agement (ZSM) Industry Specification Group (ZSM ISG) in 2017. The ETSI ZSM ISG targets to specify a scalable, extensible, and resilient reference architecture where cross-domain CCL and Artificial Intelligence (AI) techniques are mainstays to provide full-automation of E2E management operations [19].

In this work, we propose **CLARA** (**C**ontrol-**L**oop based Zero-touch Network M**A**nagement f**RA**mework), a policy-driven network management system that enables automated fault management from collaborating distributed CCLs in a multi-domain environment. Our work pursues the following objectives: *(i)* reduce the network and service management complexity, *i.e.*, implement service assurance through the implementation of self-healing property, *(ii)* enable network automation through CCL along with AI-based methods since the definition of user requirements until the execution of the control loops in each domain, and *(iii)* enforce policies in a hierarchic ZSM-based architecture to govern service behavior.

To address such objectives, we achieve a number of contributions that include:

- survey of multiple CCL solutions, including the implementation of a modular CCL platform;
- translation of service intents into both E2E monitoring models and E2E adaptive policies that extend the ETSI ZSM architecture;
- decomposition of policies into domain-specific imperative policies with actions defined in run-time from knowledge base;
- prototype implementation of the CLARA system;

The rest of the paper is organized as follows. Section II presents the main research problem and motivation. Section III discusses the relevant related works in the context of our research proposal. We point out main goals and contributions in Section IV before concluding the paper in Section V.

II. Motivation & Problem Statement

Typical network management systems are not suitable for software-based environments and lack adequate enablers for operational sustainability [24]. Apart from this, the emerging vertical industry applications with extreme requirements pose new challenges on management systems that trigger the need for novel approaches to how network operators and service providers design and operate their networks and services. The next generation of networks (*e.g.*, 5G, 6G, and beyond) depends on novel methods that automatically integrate virtualization and network softwarization technologies.

An ideal management system should deliver three main features: *(i)* full automation in operation and maintenance of the network and services, *(ii)* assurance of service quality in case of failure or SLA violation, and *(iii)* advanced intelligence in order to incorporate knowledge to support suitable decisions and root cause analysis. In this context, different Standards Developing Organizations (SDOs) have worked in the definition of end-to-end architectures and solutions, including initiatives from ETSI (ENI, ZSM), TM Forum (SID, DEN-ng, ZOOM), MEF (MEF 3.0), GSMA/3GPP (GST) and IRTF (NMRG). Common open questions relate to adequate high-level policies-driven management of multiple closed loops. The ETSI ZSM, for example, has been working definition of an autonomous end-to-end solution leaving open the implementation of the requirements proposed by its reference architecture. These observations lead us to raise the following problem statements.

Problem Statement #1: How to conceive a proper closed control loop mechanism for different use cases fed by a network policy, along with the definition of methods to extract valuable metrics to enforce such policy?

Advances in the state-of-the-art in autonomic network management have indicated the usage of novel feedback control loops, commonly defined by monitor, analyze, planning, and act mechanisms, as enabler solutions. In that regard, we lead a consistent review of the wide variety of solutions in designing and implementing CCL. From that, we propose and implement a CCL platform [11](see Fig. 1). It embodies characteristics like knowledge, modularity, and programmability leveraging best-of-breed open-source tools as well as introduces an abstraction of the network service through a graph-based approach. Knowledge element plays a fundamental role in storing and retrieving data shared between the elements within a CCL and between different CCLs. That platform is agnostic enough to be used in different domains such as cloud, transport, and access networks.

Problem Statement #2: How to build a management model based high-level policies holistic enough to enable a higher-level abstraction of the managed entity, decoupling the service management from the service deployment?

To the best of our knowledge, the translation from intent-based services requirements into KPIs and CCL configurations for automated management is not yet a well-consolidated approach in the academy and SDOs. To address that, we propose a methodology [12] that automatically captures the user's requirements (intents) and convert them into a service management model. This model is use case-independent, *i.e.*, no including details of the network infrastructure, which facilitates its portability and reusability. We utilize an ontology-based schema to create the abstract-level model focused on managing the service performance. That model should be used as input for the generation of both E2E policy and E2E

monitoring templates.

Problem Statement #3: How to add flexibility to policy-based CCL automation solutions?

The management of the multi-domain, multi-technology environment illustrated in Figure 1 is highly challenging and complex since each domain has its own constraints and scope. Zero-touch networks demand automated decisions for management actions. With policy and rules pre-set, traditional PBM solutions are inadequate for softwarized environments due to a lack of flexibility. However, that flexibility can be achieved when adding novel features to policy-based CCL automation (CCLA) solutions: *(i)* IBN to abstract the user requirements, *(ii)* adaptive policies composed of several states, and *(iii)* autonomic definition of actions. Besides, the management should be formed by policies and monitoring actions organized hierarchically (*i.e.*, E2E and local perspectives) and derived from intents. E2E policies must be enforced in the E2E layer and then decomposed to several domain-specific policies. Each domain-specific policy controls the behavior of its control loops and can make local real-time, closed-loop decisions. Regarding monitoring, the E2E monitoring model also must be decomposed into monitoring templates for each domain. Instead of individually monitoring all metrics and elements that compose a service, the model specifies KPIs to be evaluated that better represent the behavior of the service to be monitored.

We aim to design a network management system supported by well-known concepts such as Policy-based Management (PBM) and CCL, and promising technologies such as IBN and ZSM. How to handle the network service provisioning is out of the scope of our research.

III. Background & Related Works

A. Closed Control Loop Automation

IBM proposed, in 2004, an architecture for autonomic computing [6] using MAPE (Monitor, Analyse, Plan, and Execute) control loop over the managed environment. Since then, other works have proposed new solutions based on the MAPE loop, such as FOCALE [22], COMPA [9], C-MAPE [16], and MAPE-K (MAPE-Knowledge) [21]. Besides relevant projects including ONAP[1], OSM[2], and SELFNET[3] have used that approach to enable self-management of their services and networks.

CCL automation (CCLA) creates autonomous systems that fulfill all user requirements without any human intervention. It provides self-x properties as self-optimization and self-healing. Closed loops continuously monitor managed entities, analyze the data, and provide actions to meet the desired target. A myriad of initiatives deals with multiple aspects related to closed-loop automation. Initiatives as 3GPP SA5 and O-RAN define characteristics at the local domain level, while groups such as ETSI ZSM and TMF ANP specify inter-domain-level enablers and architectures. Most organizations mainly focus on deploying the service dynamically, but most important is continuous service management by means CCLs, fulfilling service quality requirements. In this aspect, we can highlight the ETSI ZSM [19] that uses closed loops in different levels inside single- and multi-domain scenarios.

B. Intent-based Networking

Currently, the service providers have considered using intent to express users' requirements, simplifying and abstracting the underlying details. Nevertheless, since intent-based service requesting until its effective deployment and management, there are many challenges to be overcome. Some efforts, in terms of standardization and open source solutions (ONOS Intent Framework[4], NEMO), address that gap but generally are fragmented proposals limited to solve part of the problem. For instance, IETF NMRG[5] discusses general aspects such as the relation between intent and policy and the use of control loops.

A set of academic works have focused on different approaches to IBN. In the SDN field, DISMI [14] presents an Intent-based NBIs for network controllers and iNDIRA [2] proposes high-level descriptive language based on RDF graphs. Although IBN is closely associated with SDN, recent works explore the scope of IBN beyond simple matters of connectivity. The work in [7], for example, proposes automated 5G Network Slice Lifecycle Management based on an intent-based approach. The authors in [13], in turn, use intents to allow communications and coordination among different CCL represented by an RDF ontology in an E2E multi-domain scenario.

C. Autonomic Network Management

PBM manages a set of orchestrated actions to assure the service requirements, *i.e.*, the mechanism monitors the resources and services and changes and maintains the state of managed entities according to desired goals. However, that is very difficult to comply with the 5G network since the management relies on actions execution in different domains and heterogeneous infrastructures. To address that, concepts such as control loop automation and network slicing along with IBN provide autonomic management in complex network environments.

An intent can be accomplished through a set of policies that can be used to control behavior of control loops. This intent refinement process involves translating intent into one or more policies and subsequently decomposing them into low-level policies or configurations. In this context, Jacobs et al. [1] extracts intent from natural language through AI-based assistant and translates it into network configurations. The authors in [15] propose a method for policy refinement that translates high-level policy into low-level rules. The work in [7] presents an automated slice lifecycle management mechanism that

[1]https://www.onap.org
[2]https://osm.etsi.org/
[3]https://selfnet-5g.eu/
[4]https://wiki.onosproject.org/display/ONOS/Intent+Framework
[5]http://www.ietf.org/internet-drafts/draft-irtf-nmrg-ibn-concepts-definitions-01.txt

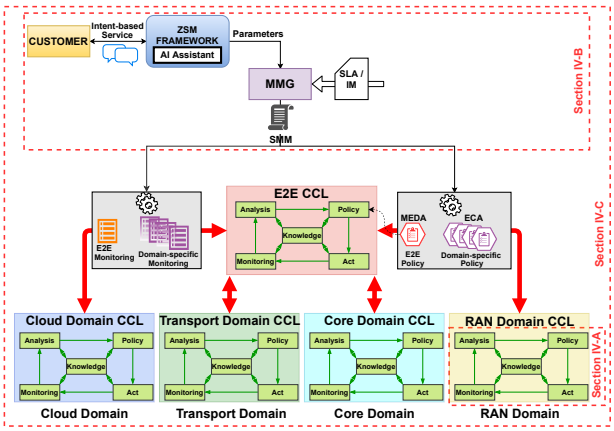

Fig. 2: Complete flow from the definition of user's requirements to multi-domain management.

handles the run time resource scalability and assurance. In a multi-domain landscape managed by multiples CCLS, once refined the intent, the output policy needs to be decomposed into domain-specific policy, in general, imperative policies. Relevant initiatives (ETSI ENI [18]) and works [23][25][5] approach distinctive aspects regarding policy modeling and management.

IV. Approaches & Contributions

Our work is based on a set of approaches to address the mentioned-above gaps through service monitoring and management in multi-domain environments where each domain has one or more policy-driven CCLs with a specific goal. Such approaches are illustrated in Figure 2. Details of each contribution are discussed in the following subsections.

A. Closed Control Loop Platform

Firstly we evaluate diverse implementations of CCL based on problem statement #1. A variety of solutions were found addressing different aspects of CCL but limited to specific use cases. Thus, we propose an autonomic CCL platform controlled by policies. That platform implements four main features: *modularity*, use of REST API to allow to add and remove components; *adaptive policy*, where different states provide more flexibility in the definition of policies; *infrastructure abstraction*, high-level view of underlying infrastructure through graph-based approach; and finally, *knowledge component*, that enables the usage of ML techniques in the network and service management, *e.g.*, build complex metrics, root cause analysis. Besides, our CCL platform provides an environment for testing in programmable networks by creating flexible means towards fast experimentation and agile innovation in lab environments. Further information and results can be found in [11] and [4].

B. Generate an Service Management Model for Zero-touch networks

In the process of service deployment, the first step is to specify the network service through a well-defined information model (*e.g.*, NSD, NEST) or an IBN-based interface. In terms of IBN, one of the most significant challenges is avoiding ambiguity in user intent specification. Generally, that part involves non-technical users that require well-done intent interfaces to express your intent. Our system covers the intent refinement to the service management process, including self-x properties and service assurance. The initial part consists of capturing the user's requirements and extracting the useful features. To address that and reduce ambiguity, we develop an AI assistant that steers the users to specify the main components of intent. The three main elements of the assistant are Natural Language Understanding (NLU) *interpreter*, which extracts the intent, entities, and any other information; *core* responsible for choosing the best message to send to the user; and *middleware* that connects the chatbot to any other external applications. We use the open-source platform Rasa[6] for implementing the virtual assistant. The code is available in our GitHub directory [7].

The next step is to send the parameters of the services to Managing Model Generator (MMG) component. It is in charge of parse these parameters and automatically constructs templates for service management. MMG implements a novel methodology where service parameters and standard information models (IM), obtained from main SDOs such as 3GPP and ETSI, are used as inputs to generate a high-level management template, called Service Management Model (SMM), see Fig. 3a. The process is built upon an ontology-based schema using a Resource Description Framework (RDF) vocabulary. The proof of concept implementation along with functional validation of this methodology are presented in [12]. Such methodology answer to the research question #2.

C. Closed-Loop based Zero-touch Network Management for multi-domain scenarios

Extended ETSI ZSM Architecture. ETSI ZSM defines a reference architecture to provide zero-touch automated network and service management in next-generation networks. ZSM architecture is based on a set of principles [19] such as CCL management automation, Intent-based interfaces, and cross-domain collaboration aiming to achieve a specific target, *e.g.*, service assurance, self-healing. That architecture is aligned to what we have developed in terms of the CCL, IBN, and management model. So, we extend the ZSM architecture to include the management model building method proposed in the previous section. The steps of method execution occur in a previous moment to service deployment process, *i.e.*, in design time. The SMM provides features necessary to specify the E2E metrics to be monitored (E2E monitoring), and to identify SLA violation or performance degradation, *i.e.*, the main part of E2E policy (see Fig. 3a).

Automated APEX Policy Model Generation. For achieving full automated management, the policy needs to be flexible and adaptive, allowing autonomous decision-making. For that, we

[6]https://rasa.com/

[7]https://github.com/nfss83/intent-assistant

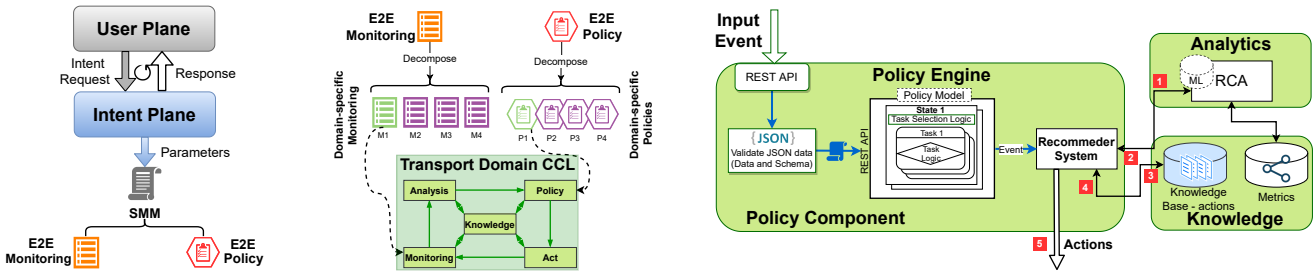

Fig. 3: Details of our main approaches: (a) building of the service management model, (b) decomposing of high-level models into specific-domain one, and (c) main parts of policy component.

use the APEX (Adaptive Policy Execution) [8] policy engine as core part of *policy* component. APEX is a flexible policy engine and can adopt various roles in our system: make real-time decisions considering target KPIs and monitoring information or make preventive decisions based on historical information. The APEX policy model has three main parts: state, event, and task. One or more states form an APEX policy. The events are a collection of fields that can trigger the policy execution (incoming events) or provide its execution result (output events). Tasks specify the information consumed and produced by policy through task logic, *i.e.*, the logic executes the work of the task. APEX also stores state information and data used by policies. In this context, we develop a mechanism to build a policy based on the APEX model from SMM. Algorithm 1 describes the generic workflow for build the APEX model.

Algorithm 1: Generate E2E policy

Data: SMM (format=turtle)
Result: P_{e2e} (initEvent, policyModel, confPolicyEngine)

$inputFields \leftarrow []$;
$metricList \leftarrow extract(SMM)$/* Extract metrics, values, units, and conditions */
def *checkMetrics(metricList)*:
    /* Check if metrics are supported by policy engine */
    for $eachMetric \in metricList$ **do**
        if *existingMetric*$(eachMetric)$ **then**
            $inputFields \leftarrow eachMetric$;
    return $inputFields$;
$inputFields \leftarrow checkMetrics(metricList)$;
$initEvent \leftarrow genInputEvent(inputFields, source = Analytics|Knowledge, target = policyEngine)$;
/* Event that initializes the E2E policy */
$policyModel \leftarrow genPolicyModel(name = MgmtPolicyModel, inputFields)$;
/* Define the states, tasks and logic */
$confPolicyEngine \leftarrow genConfEngine(initEvent, policyModel, inputFields)$;
/* Specify engine parameters as executor, port, protocol, carrier technology */
return $initEvent, policyModel, confPolicyEngine$;

Recommendation Method for Action Selection. A big challenge in automated policy generation is defining adequate actions when incidents occur (*e.g.*, SLA violation, network or service failure). In general, those actions are previously defined what becomes the policy inflexible. Our approach uses a recommender system to propose more suitable actions to certain network or service issues. Information from diverse sources, including vendor manual, network administrator experience, CCL feedback, and other management systems, create a knowledge base correlating issues with existing solutions. Consider knowledge graph as a directed graph $G = (V, E)$ whereby V is the set of vertices representing the incidents and actions, and E is the set of directed edges representing the priority relation between incident and action. Edges have a numeric value that reflects the priority that one action has over others. From CCL feedback, if the action solves the issue successfully, its priority is increased in the knowledge base; otherwise, it is decreased. The knowledge base can inference new correlations, although these relationships were not recorded in the input data. This process is orchestrated by the *Policy* component permitting more flexibility and adaptation to network and service behavior.

The *Policy* component is presented in Figure 3c, depicting the main parts, interfaces, and interaction with other loop components. The policy is triggered from an input event (*e.g.*, messages from *Analytics*, *Monitoring*, or polling into the knowledge base). The event, in JSON format, is validated and sent to the policy engine. It performs the policy logic and, if there is a violation, an output event is sent to the recommender system to define the action to fix the issue. At this point, the recommender requests from *Analytics* a root cause analysis to find the faulty cause (step 01). The *Analytics* can query the information of service performance metrics in a graph-based database stored into *Knowledge* component or use a Machine Learning model (out of our scope) to discover the root cause. The RCA output is sent back to the recommender. Then, the recommender compiles a list of actions ordered by priority from the knowledge base and selects the action with the highest priority (steps 03 and 04). Finally, the *Policy* sends the action to the *Actuator* executes it (step 05). After execution, the violation condition is re-evaluated. Case the executed action fixes the problem, the recommender prioritizes it; otherwise, the recommender penalizes it and can select the next action from the ordered list (steps not shown in Fig. 3c).

SMM decomposition into domain-specific models. The E2E service spans multiple domains where each domain has one or more CCLs organized in hierarchic architecture as

illustrated in Figure 1. We develop a procedure to decompose SMM into domain-specific models (see Fig. 3b). Algorithm 2 presents the decomposition of SMM into domain-specific policies. The function *selectDomainMetrics* is responsible first for identifying the specific-domain metrics (lines 5–6); for example, Packet Error Rate (PER) is specific for RAN domain, and, after, decomposing (lines 8–9) E2E metrics into typically domain-specific values and conditions. In this process, cumulative metrics as latency and packet loss have their values split for each domain based on the real-time measurements collected from the *Knowledge* component. If the maximum E2E latency of service is 50 ms, so the method should divide that threshold value for each domain, for example, 10 ms to RAN, 15 ms to Core, and 25 ms to Transport, based on average values monitored from each domain. In non-cumulative metrics as throughput, the method selects the same values and conditions to be applied in each domain. The procedure of monitoring templates generation per domain involves selecting domain-specific metrics to be monitored.

Algorithm 2: Generate domain-specific policy

```
   Data: SMM (format=turtle), domain (Cloud, Transport, Core, RAN)
   Result: P_ds (initEvent, policyModel, confPolicyEngine)
1  inputFields ← [];
2  metricList ← extract(SMM)/* Extract metrics, values,
       units, and conditions                                  */
3  def selectDomainMetrics(metricList, domain):
       /* Select domain-specific metrics.                    */
4      for eachMetric ∈ metricList do
5          if isDomainMetric(eachMetric, domain) then
6              domainMetrics ← eachMetric;
7          else
8              if isE2eMetric(eachMetric) then
9                  domainMetrics ←
                       decompose(eachMetric, domain);
10     return domainMetrics;
11 inputFields ← selectDomainMetrics(metricList, domain);
12 initEvent ← genInputEvent(inputFields, source =
       Analytics|Knowledge, target = policyEngine);
13 policyModel ← genPolicyModel(name =
       domainPolicyModel, inputFields);
14 confPolicyEngine ← genConfEngine(initEvent,
       policyModel, inputFields);
15 return initEvent, policyModel, confPolicyEngine;
```

V. Final Remarks

With emerging new increasingly demanding services, operators require network management that provides automation and agility and incorporates required intelligence to ensure the end-to-end quality of network services. Recent initiatives have pointed to policy-driven CCL and IBN as key enablers for zero-touch networks. The driving theme of this work is the potential of policy-driven CCL automation for service assurance. We propose CLARA, Closed-Loop based Management Framework for zero-touch networks, to address CCL automation aspects, including intent refinement, implementation of CCL instances, and policy enforcement in multi-domain environments.

Acknowledgment

Work supported by the Innovation Center, Ericsson S.A., Brazil, grants UNI.64 and UNI.67. This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

References

[1] A. Jacobs et al. Refining network intents for self-driving networks. *ACM SIGCOMM Computer Communication Review*, 48(5):55–63, jan 2019.

[2] A. Mercian et al. Indira: 'application intent' network assistant to configure sdn-based high performance scientific networks. In *OFC*, pages 1–3, 2017.

[3] B. Dutta et al. The Challenge of Zero Touch and Explainable AI. *JICT Standardization*, 9:147–158, jun 2021.

[4] C. Rothenberg et al. Intent-based control loop for dash video service assurance using ml-based edge qoe estimation. In *NetSoft*, pages 353–355, 2020.

[5] E. Rutten et al. Feedback Control as MAPE-K Loop in Autonomic Computing. In *Lecture Notes in Computer Science*, volume 9640 LNCS, pages 349–373. Springer Verlag, 2017.

[6] J. Hanson et al. An architectural approach to autonomic computing. In *Proceedings of the First International Conference on Autonomic Computing*, ICAC '04, pages 2–9, Washington, DC, USA, 2004. IEEE Computer Society.

[7] K. Abbas et al. Network Slice Lifecycle Management for 5G Mobile Networks: An Intent-Based Networking Approach. *IEEE Access*, 9:80128–80146, 2021.

[8] L. Fallon et al. Apex: An engine for dynamic adaptive policy execution. In *NOMS 2016*, number Noms, pages 699–702. IEEE, apr 2016.

[9] L. Fallon et al. Using the COMPA autonomous architecture for mobile network security. In *2017 IFIP/IEEE IM*, pages 747–753. IEEE, may 2017.

[10] N. Saraiva et al. Network service orchestration: A survey. *Computer Communications*, 142:69–94, 2019.

[11] N. Saraiva et al. Policy-Driven Network Traffic Rerouting Through Intent-Based Control Loops. In *SBRC-WGRS*, May 2019.

[12] N. Saraiva et al. End-to-End Service Monitoring for Zero-Touch Networks. *JICT Standardization*, 9:91–112, 2021.

[13] P. Gomes et al. Intent-driven Closed Loops for Autonomous Networks. *JICT Standardization*, pages 1–30, jun 2021.

[14] P. Sköldström et al. Dismi - an intent interface for application-centric transport network services. In *ICTON*, pages 1–4, 2017.

[15] R. Craven et al. Policy refinement: Decomposition and operationalization for dynamic domains. In *2011 7th International Conference on Network and Service Management*, pages 1–9, 2011.

[16] S. Ayoubi et al. Machine Learning for Cognitive Network Management. *IEEE Communications Magazine*, 56(1):158–165, 2018.

[17] ETSI. Improved operator experience through Experiential Networked Intelligence (ENI). Technical Report 22, ETSI, 2017.

[18] ETSI. Experiential networked intelligence (ENI); Context-Aware Policy Management Gap Analysis Disclaimer. Technical report, ETSI, 2018.

[19] ETSI GS ZSM. GS ZSM 002 - V1.1.1 - Zero-touch network and Service Management (ZSM); Reference Architecture. 1:1–80, 2019.

[20] ETSI GS ZSM. ETSI GS ZSM 009-1: Zero-Touch Network and Service Management (ZSM); Closed-loop automation; Enablers. Technical report, ETSI, 2021.

[21] J.O. Kephart and D.M. Chess. The vision of autonomic computing. *Computer*, 36(1):41–50, 2003.

[22] Qusay Mahmoud. *Cognitive Networks: Towards Self-Aware Networks*. John Wiley & Sons, Ltd, Chichester, UK, jul 2007.

[23] J.D. Moffett and M.S. Sloman. Policy hierarchies for distributed systems management. *IEEE Journal on Selected Areas in Communications*, 11(9):1404–1414, 1993.

[24] NGMN Alliance. Perspectives on Vertical Industries and Implications for 5G. *White Paper*, pages 1–29, 2016.

[25] J. Strassner. Den-ng: achieving business-driven network management. In *NOMS*, pages 753–766, 2002.

[26] Ishan Vaishnavi and Laurent Ciavaglia. Challenges towards automation of live telco network management: Closed control loops. In *CNSM*, 2020.