System Architecture of Intelligent Monitoring in Multi-Domain Orchestration

Wint Yi Poe, Ishan Vaishnavi
Huawei Technologies Duesseldorf GmbH,
Munich, Germany.
{wint.yi.poe,ishan.vaishnavi}@huawei.com

Francesco Tusa
University College London,
London, UK.
francesco.tusa@ucl.ac.uk

Javier Melián, Aurora Ramos
ATOS,
Spain SA.
{javier.melian, aurora.ramos}@atos.net

***Abstract*— Orchestrating a service or resources across multiple administrative domains requires three main high level steps of operation: i) capability detection of other domains ii) placement of the service request across the domains iii) assurance that the service is functioning within the acceptable bounds of the service level agreement (SLA). This paper focuses on the assurance step; in particular, we present a novel architecture and preliminary implementation that supports monitoring of KPIs across multiple administrative domains. The challenges towards realising such an architecture are: i) coordinated monitoring with *no* direct access to the other domain's infrastructure ii) monitoring over an abstracted (instead of actual) topology that each administrative domain may expose to other domains iii) different domains have different systems for monitoring with different KPIs as well as different ways of measuring those KPI. Our architecture, referred to as IMoS addresses these challenges to provide an end-to-end monitoring as a fundamental functionality for supporting assurance and SLA management for services orchestration across multi-administrative-domains. In addition, the preliminary results are provided from the first proof of concept implementation of an IMoS. The work done in this paper has been developed within the H2020 ICT14 project 5G Exchange.**

Keywords — monitoring; assurance; multi-domain orchestration; 5G architecture

Introduction

The next generation of telecommunication networks, labelled 5G, will see collaborations among multiple administrative domains (i.e., providers) in order to support the deployment of complex services. An administrative-domain refers to a collection of systems and networks each operated by a single organization or administrative authority, such as an operator [6]. For example, an administrative domain can be an infrastructure domain that provides virtualised infrastructure resources such as compute and storage via a service abstraction, to other external administrative domains. In this scenario, cross domain operation becomes a key feature that enables customers buying one stop services at a single administration. The process of creating cross-domain services in our architecture is carried out by one or more multi-domain orchestrators (MdOs), each capable of managing both the local (typically technological) domains and interacting with other MdOs to deploy multi-administrative-domain services.

For assuring the performance of the running services, monitoring is a fundamental functionality of the MdOs enabling reporting on the performance of a service (i.e., the related Key Performance Indicators [KPIs]). In a multi administrative scenario, monitoring faces a number of additional challenges owing to the lack of control and visibility of one MdO into another operator's infrastructure.

In previous work [8], we defined a very high level abstract architecture of how such an IMoS would function within an MdO. In this paper, we dive into the details of the implemented architecture focusing on how enabling multi-administrative domain monitoring for supporting assurance and SLA management and orchestration of services' and resources. The next Section presents the reference MdO architecture and how it deals with the multi-administrative-domain orchestration. Section II covers the detailed aspects of the multi-administrative-domain monitoring architecture and components. Section III then presents the monitoring workflow and implementation. In Section IV we study the related work on this topic and the shortcoming thereof.

I. Reference MdO Framework

A. MdO Architecture

The architecture we have considered for a possible MdO implementation is shown in Figure 1. Figure 1 presents the core components within the MdO that are responsible for the three main steps of the deployment, i.e., i) Each MdO gathers information of the resource (via TADS) and service (via CMS) capabilities of other administrative domains.

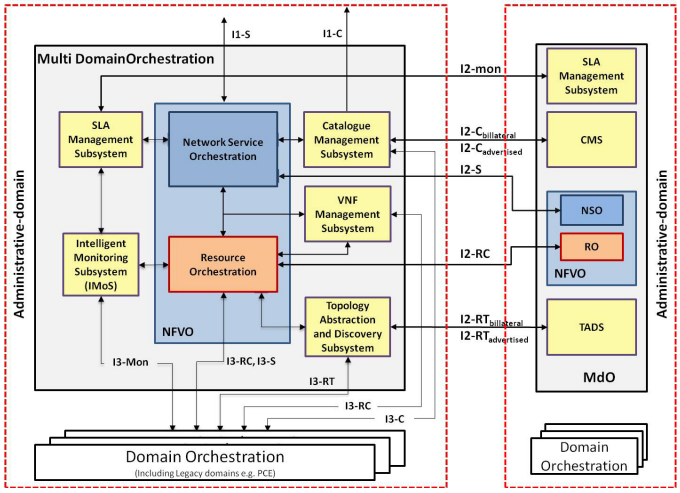

Figure 1: The reference MdO architecture.

This information may, however, be abstracted by the other domain's MdO. For example, a node may expose an entire

domain as an abstracted single node [10]; ii) on receipt of a service request and based on the abstracted topology of other domains (combined with the topology of its own domain), a MdO performs service (via NSO) and/or resource orchestration (via RO) by executing specific algorithms that will embed either service or resource subcomponents over the entire topology [10]. The placement decision is then executed by the Network Function Virtualization Orchestration (NFVO) instantiating the resources and deploying the services across the various domains.

In parallel to the deployment of the service in step iii) the MdO must also start the monitoring to ensure the service level objectives composing the SLA agreement are being met while the service is running. This usually requires collection of KPI values from different domains over abstracted topologies, which in our architecture is done by the IMoS. The in-depth architecture and procedures designed and implemented to achieve this monitoring information across the MdOs by the IMoS is the main contribution of this paper.

B. *Scenario and Service Allocation*

A provider A receives a request to instantiate a service S as shown in Figure 2. The provider A collects the abstracted topology information from provider B (i.e., nodes B1 and B2) and provider C (i.e., nodes C1, C2, and C3) via TADS. TADS exposes only the abstract view of the entire domain, and provider A has no access or information on the actual topology supporting the abstracted one. For example, the node B1 represents the abstracted topology of the DO1 of the provider B. Based on the steps in operation ii, the MdO in provider A decides to host the service request by decomposing it into three parts SA, SB, SC.

The request receiving MdO (RR-MdO) of operator A is responsible for splitting the service request graph and report on the performance of the entire service to the user. To do so, the service embedding engine in the NSO of RR-MdO breaks up the service request graph into multiple VNF sub-graphs that are in turn forwarded to the MdOs (including also the host MdO). In each MdO the VNF graph is converted into a resource graph which is again split and deployed across the underlying domains by the resource embedding engine in the RO. In this paper, we refer to both the splitting processes as *service allocation*. As shown in Figure 2, the service S is decomposed into the three sub-services. While the sub-service, SA is allocated in the provider A, the sub-services SB and SC are forwarded and realised by the MdO in the operator B and C, respectively.

To report about the performance of the entire service to both the customer and other supporting components, the IMoS in RR-MdO will be responsible for the end-to-end monitoring of the service S. More specifically, it will orchestrate the deployment of probes across the different domains to collect KPI values. For the purposes of this work, *a probe is defined as an abstract entity referring to the request for collecting data typically related to a KPI for the probe deploying entity, in this case an IMoS.* The need for such a definition shall become clear when we describe the end-to-end monitoring workflow of the scenario in Section III.

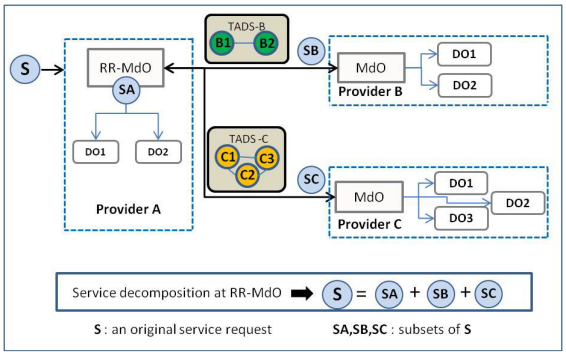

Figure 2: An example scenario and service allocation across provider A, B, and C.

II. Intelligent Monitoring System

Considering the aforementioned MdO architecture and service instantiation process, in this section we focus on describing the architecture and functionalities implemented by the Intelligent Monitoring Subsystem (IMoS) of Figure 1.

A. *IMoS Architecture and components*

IMoS was designed to provide intelligent, coordinated monitoring functionalities when dealing with both services deployed within a single administration premises and multi-administrative-domain services (through the coordination of different IMoS instances). The intelligent here refers to the optimization in probe selection and deployment of the end-to-end service monitoring process. In the case of single administration, IMoS is responsible to perform the monitoring of (sub-)services potentially deployed across different technological domains. Let's assume that the MdO in the administrative domain A receives a service request and the service is allocated across multi-administrative-domain environment (B and C) as presented in Figure 2.

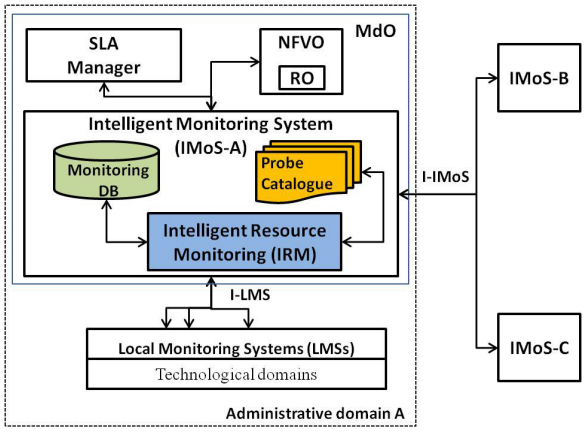

Figure 3: The IMoS architecture.

The IMoS in the administrative domain A enforces the monitoring of the sub-service (i.e., SA) that is realised on resources belonging to the underlying technological domains by coordinating the collection of measurement associated local monitoring systems (LMSs) as shown in Figure 3. In parallel, the IMoS-A (i.e., RR-IMoS) coordinates with IMoS B and

IMoS-C for the sub-services SB and SC, accordingly. The monitoring of a service in a local administration consists selecting the appropriate probes to collect measurements related to the KPIs of (sub-) service instantiation requests and sending related probe activation commands to the LMSs. The monitoring of a service in coordination with other IMoS includes (but is not limited to) the aggregation of the performance of sub-services, a global view of probe selection, and probe deployment for sub-services monitoring. The IMoS in the external administrative domains orchestrates the probe selection and deployment locally.

Note that different technological domains in each administration rely on different LMSs and policies. In a multi-administrative-domain service monitoring, the lack of coordination enforced by a central IMoS entity would lead to a probe selection in each domain that does not represent the optimal solution. We propose overcoming this issue by enabling the coordination of different IMoS instances to guarantee a global optimal monitoring probe deployment in both inter and intra administrative domains. The functionalities of IMoS is managed by three main components.

Intelligent Resource Monitoring (IRM): The *IRM* is mainly responsible for the implementation of the following functionalities:

- To create a probe catalogue, by collecting the set of probes (and related KPIs) from LMSs in different technological domains.
- To select the appropriate probes for each (sub-) service.
- To create a DB entry in monitoring DB for each (sub-) service.
- To instantiate the selected probes in LMS collect monitoring data from probes, and store the data in the monitoring DB.
- To collaborate the probe selection and deployment of sub-services involved in the execution of the original service instance which are realised in different administrative domains.
- To interact with the SLA management subsystem to support service assurance and life cycle management.
- To map the abstracted probe requests from IMoS instances belonging to other domain on the abstracted topology to the real probe requests to the LMS in its domain over the real topology. This mapping from virtual probes in the abstracted topology to probe requests to LMS of the real topology is completely hidden from the IMoS instances in other domains. This responsibility is explained further in the workflow in Section III.

Probe Catalogue: A probe catalogue is used by IMoS to store all the probes, related to the respective topology, available in the underlying technological domains to be used during the process of probes selection and activation. We would like to clarify here again that *a probe is defined as an abstract entity referring to the request for collecting data typically related to a KPI by the probe deploying entity, in this case an IMoS.* This essentially means that a probe is simply a way of collecting values typically for a KPI. In order to create a probe catalogue, IMoS needs to gather that information from each LMS of each local resource domain. In this way each IMoS instance will have a centralized view of all the available probes and will use the probe catalogue to support intelligent monitoring functions across MdOs. The catalogue can be used for instance to create the correct mapping between KPIs and probes to be activated in each resource domain. Together with the definition of probes this catalogue essentially stores an index of the different ways of collecting monitoring information of a KPI from the administrative domain w.r.t to the abstracted topology exposed by that domain.

Monitoring Database: The main functionalities of the *Monitoring DB* are:

- To allow storing measurements coming from LMS in technological domains and IMoS from other MdOs.
- To provide real-time measurements availability to the SLA Management Subsystem. A measurement can be either a raw or an aggregated piece of data (to avoid requesting and calculating the same monitoring information for each of sub-service).

B. Interfaces

Common interfaces are required to allow the exchange of probe catalogues and probe related information as well as data reporting with both multiple LMS and other IMoS instances as shown in Figure 3.

- I-LMS: The interface between IMoS and a LMS used by IMoS to exchange probe related information, to guarantee (sub-)service monitoring by sending the probe activation commands to the LMS, and to collect monitoring data from a LMS.
- I-IMoS: The interface between multiple IMoS instances used to exchange KPIs from a probe catalogue, abstract view of probe catalogue, probes, and probe related information to guarantee end-to-end service monitoring and security related information.

The exchange of information, via the interface I-LMS, includes but is not limited to probe name, parameters, probe deployment costs including both financial and resource related costs, probe dependencies, etc. Then IMoS decides the correct probe selection and deployment by performing the intelligent algorithms. The selected probe instantiation request is send to the LMS and collects the measurement via I-LMS.

The interface I-IMoS is used while aggregating monitoring data from different MdOs. Additionally, the probe related and security information is exchanged between multiple IMoS over this interface. The probe related information can be, for example, a probe which is unavailable in one MdO but is required for the end-to-end monitoring. Another example is the case of sharing information between multiple IMoS, such as the available probes from the probe catalogue to optimize the end-to-end probe deployment. Note that the exchange of probe information between MdOs is limited to the probes that can be instantiated on the abstracted topology; in general we assume that the different administrations are not willing to share the detailed information.

III. Workflow and Implementation

A. *Monitoring Workflow*

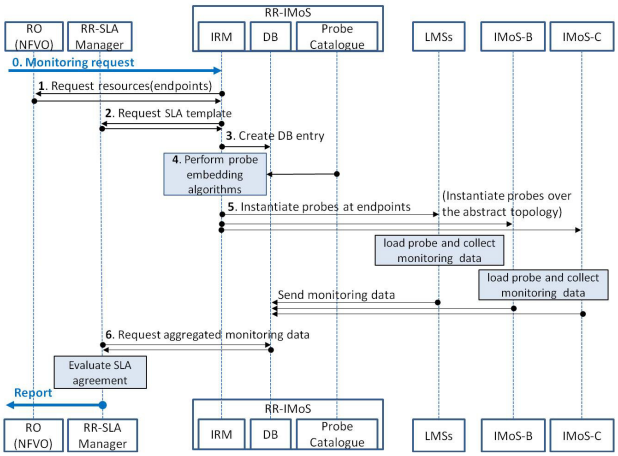

Figure 3: The end-to-end monitoring workflow.

Referring to the scenario in Figure 2 and to the components represented in Figure 3, once a network service is deployed, the customer's request will enter into the assurance phase. The SLA management functionality in the RR-MdO, (RR-SLA manager) will be responsible for reporting about the performance of the service to the customer detecting any breaches on the mutually agreed SLAs and to the MdO for potential reconfiguration reactions. In other words, this process will imply having a functionality (per MdO) that receives both the specific monitoring requirements (i.e. metrics, KPIs) and resource allocation information for each service being deployed, and activates the appropriate probes where/when needed. The end-to-end monitoring workflow in multi-administrative-domain environment required to implement the service assurance is presented in Figure 3. The flow considers from the perspective of RR-IMoS in the administrative domain (e.g., domain A) receiving a service request, (e.g., S). The service is decomposed into multiple sub-services (e.g., SA, SB and SC) across the administrative domains (A, B and C). The RR-IMoS has knowledge of the LMSs from the local domains and the IMoS from different administrative domains for the probe selection and deployment. The information is exchanged via the interfaces I-LMS and I-IMoS.

1. Information describing the resources associated to the service (sub-graph) instance is provided to IMoS by the RO in the NFVO.
2. The relevant SLA parameters associated to a new service instance are passed on from the SLA Manager to the local IMoS as a SLA template.
3. IMoS interacts with the local monitoring DB to perform initialization of the data structures needed to store monitoring data for the new service instance.
4. IMoS calculates the near[1] optimal placement of the probes according to the received inputs. The concept of probe deployment optimality refers to minimizing the probe effect: the resources used by the probes themselves. For example if a KPI could be measured in domain A it does not need to be measured elsewhere or that the same probe could measure KPIs for multiple hosted services. For the concept of optimality of placing probes see [8].
5. According to the previous point, IMoS interacts (through I-LMS) with the LMSs involved in the execution of the current service instance that have been selected for the deployment of probes in order to instantiate/activate the required probes. As soon as probes are instantiated on the relevant resources in the local domains, monitoring data will be sent to the local monitoring DB. Similarly, IMoS interacts (through I-IMoS) with the IMoS from the domains involved in the execution of the original service instance to instantiate the probe(s) to collect the performance measurement. Note that IMoS decides the probe selection and deployment using the abstract view of topology via TADS from the involved administrative domains. The IMoS in the *non-RR domain* must translate the probe deployment request over the abstracted topology to that of deploying local probes over the real topology. It then periodically collects the information from the local toplogy, aggregates it to represent the KPI over the abstract topology and reports the aggregated values for the abstract topology to the RR-IMoS' central DB via I-IMoS. The algorithms for doing so are currently being studied within the project. Note that this step essentially enables recursion in IMoS reporting, enabling domains to be recursively stacked over other domains.
6. Finally, the RR-SLA Manager will retrieve (periodically and/or on demand) the measurements from either the local Monitoring DB or on an external SLA Manager. Based on the collected measurements, the SLA Manager will check/calculate whether the SLA requirements initially set for that service are satisfied and will report the results to the user and/or other MdO management entities, e.g., for reconfiguration.

A possible alternative approach considers the SLA Proxy/Aggregator (submodule of the SLA Manager) as the module in charge of aggregating monitoring data from different administrative domains. In a multi-administrative-domain environment, the service instantiation request is split recursively, delegating to the appropriate MdO the instantiation of a part of the original service. Each SLA Manager keeps a track of the location of the metrics relevant for the local domain so it is easy to know which MdO (or local database) to query to obtain the right monitoring information. The aggregation and storage of the monitoring information is mainly performed by each MdO for the metrics belonging to the instances of this domain. In this way we avoid having redundant flows of monitoring information travelling between domains, however, in this approach a near optimal deployment of probes is more difficult as there is no centralized coordination. This implies that the same KPI could be measured in different domains multiple times increasing the probe effect in the end-to-end monitoring process.

[1] The placement of probes is NP hard problem

B. Implementation

We implemented a preliminary version of IMoS for a single administration using Java. For the LMS we considered *Lattice* monitoring framework [9].We implemented a testbed with two virtual machines (VMs) where IMoS and Lattice are allocated. In the same VM with IMoS, a time series database InfluxDB **Error! Reference source not found.** is configured to store monitoring data. The monitoring request of the sub-service is deployed in the Lattice VM. Initially, the IRM in IMoS exchanged the required information with LMS to create the probe catalogue. When a monitoring request for a sub-service arrived, IMoS performs the steps 1-4 of the workflow in Section III. In this preliminary version, we consider a basic probe selection and deployment algorithm. Using the information in the probe catalogue, IMoS creates a mapping of KPIs and the correct probes where the probes are selected for the specific KPIs of a service request. In order to allow IMoS to perform the multi-technological monitoring orchestration, the I-LMS interface northbound has to exist to translate the commands sent through the I-LMS interface into actions specific for a given LMS. To perform the steps 5 we designed and implemented an adaptation module in Lattice exposing the I-LMS interface northbound and translating the related commands into actual domain level monitoring operations. The exemplary results are show in Figure 4.

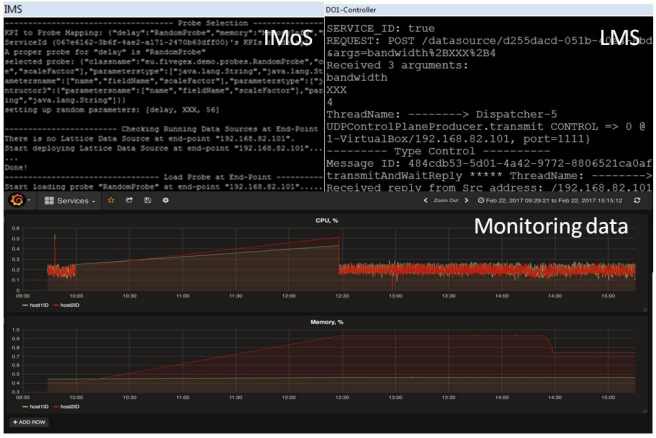

Figure 4. The preliminary implementation of IMoS.

IV. Related Work

Most of the considered monitoring solutions are based on client-server architecture and allow the user to write custom probes mainly using scripting languages (i.e., Nagios [2], Opsview [3] and Zabbix [4]). Some of them (i.e., Opsview and Zabbix) also provide an API to control some of the functionalities of the monitoring server. However, we realized that mechanisms for controlling dynamically the behaviour of the monitoring agents (i.e., the entities running in the resources to be monitored) were missing, together with the possibility of instantiating and configuring probes on the fly (e.g., loading a probe on a particular agent or changing the data sending rate of the probes). Last but not least, we also realised that the communication protocols implemented by the considered monitoring solutions for the transmission of the measurements were not optimized from a data transmission point of view: monitoring data from the agents are not sent to the central server using lightweight serialization mechanisms, and metadata are included in each measurement, leading to a potential network traffic overload. Service Assurance and SLA management in NFV context as part of 5G is still an open research issue not been fully addressed as it is explained here by ETSI NFV [7]. Works in that direction have been performed also by TMForum ZOOM project starting from SLAs in cloud services [11] and by T-NOVA project [12] where specific automatic SLA management was implemented for VNFaaS cases, however not in a multi-administrative-domains context involving the complexity driven by SLAs aggregation as the work presented here.

V. Conclusions

In this paper we proposed a high level architecture of monitoring component, IMoS to perform intelligent, coordinated monitoring for the end-to-end monitoring in management and orchestration across multiple administrative domains for 5G networks. The challenges addressed by the architecture include working over abstracted topologies exposed by other domains while centrally coordinating probe deployment in order to minimize the probe effect. The proposed monitoring architecture named IMoS is a first proof of concept developed within the 5G Exchange project [1] towards the development and testing of a prototype for assurance and SLA management in multi-administrative-domain environment. Future work includes research into probe placement algorithms for the RR-IMoS and abstracted probe to real probe translation and aggregation algorithm for the non-RR IMoS.

Acknowledgment

This work has been supported by the European Community through the 5GEx project (grant no. 671636) within the H2020 program.

References

[1] H2020 ICT14 5G Exchange (5GEx): https://www.5gex.eu/
[2] Nagios : https://www.nagios.org/documentation/
[3] Opsview: https://docs.opsview.com/doku.php
[4] Zabbix: https://www.zabbix.com/documentation/
[5] InfluxDB: https://docs.influxdata.com/influxdb/
[6] ETSI GS NFV-MAN 001 V1.1.1, “Network Functions Virtualisation (NFV); Management and Orchestration”, 2014.
[7] ETSI NFV IFA-012 https://portal.etsi.org/webapp/WorkProgram/Report_WorkItem.asp?WKI_ID=44576
[8] I.Vaishnavi, R. Guerzoni and S. Beker, "An architecture for co-ordinated monitoring for multi-provider cloud platforms," *2014 IEEE Globecom Workshops (GC Wkshps)*, Austin, TX, 2014, pp. 1-6.
[9] S. Clayman, A. Galis, and L. Mamatas, “Monitoring Virtual Networks with Lattice”, *IEEE/IFIP NOMS Wksps*, 2010, pp. 239-246.
[10] I. Vaishnavi, R. Guerzoni and R. Trivisonno, "Recursive, hierarchical embedding of virtual infrastructure in multi-domain substrates". In *2015 1st IEEE Conference on Network Softwarization (NetSoft)*. IEEE. April, 2015, pp. 1-9.
[11] G1127 End-to-end Virtualization Management: Impact on E2E Service Assurance and SLA Management for Hybrid Networks. April 2015.
[12] FP7 T-NOVA – Network function as a Service over Virtualized Infrastructures. www.t-nova.eu- Deliverable D6.4 – SLAs and billing.