Using BPEL Workflow Processing for Cross-Layer Orchestrations in IP-over-Optical Networks: A Proof of Concept

Mohit Chamania, Edip Demirbilek, Admela Jukan
Technische Universität Carolo-Wilhelmina zu Braunschweig
Email: {chamania, demirbilek, jukan}@ida.ing.tu-bs.de

X. Masip-Bruin, Marcelo Yannuzzi
Advanced Network Architectures Lab (CRAAX), Spain
Email: {xmasip, yannuzzi}@ac.upc.edu

***Abstract*—The rapid growth of network services is forcing operators to use of inter-layer coordination between the IP and the transport network management systems in order to optimize resource utilization within the carriers network ecosystem. However, the diversity of technology and the related management systems within the IP and transport networks poses significant challenges in developing a single unified management ecosystem for both IP and transport networks. In this work, we present a basic prototype of a Service Oriented Architecture (SOA) based framework proposed in the EU project ONE [1] for programmable orchestration of IP and transport management processes. The proof-of-concept prototype uses web services and BPEL to facilitate programmable inter-layer coordination, and was tested on a number of use cases. We measured the performance penalty of using the SOA framework against the traditional JAVA based provisioning approaches used in integrated multi-layer management, and show that the penalty incurred in execution times were sufficiently small while bringing several significant advantages during the software design, development, integration and maintenance, which is simple and highly effective.**

I. Introduction

The diversity in management and operation of IP and optical transport networks has driven the operational and administrative separation seen in current carrier organizations. Today, optical transport networks are managed using Transport Network Management Systems (T-NMS), which have been designed for reliable provisioning and management of a relatively small number of network services and support standard interfaces such as MTOSI [2]. On the other hand, IP networks are designed to be flexible in order to support quick introduction of new network services. As a result, the management of a typical IP network is performed using custom configuration scripts and the networks are managed by a small number of highly skilled network operators.

While the different management paradigms have helped IP and transport networks to evolve quickly, the operational separation between them has led to the lack of inter-layer coordination: for example, basic operations requiring inter-layer coordination such as provisioning of a new IP link can take as long as one week in carrier organizations, critical operations such as protection are duplicated in both layers and both networks are planned independently with utilization restricted to 40% [3] and planning cycles spanning 1-2 years. As a consequence, the capital investment required for keeping up with growing Internet traffic is very high, so operators are exploring mechanisms to drive down cost by facilitating automated coordination between the IP and transport networks.

978-1-4673-0269-2/12/$31.00 © 2012 IEEE

Initial attempts to address inter-layer coordination focused mainly on developing a single unified *control plane* to support both IP and transport networks, with the two primary solutions as the ITU-Ts ASON framework [4] and the IETFs GMPLS control plane suite [5]. However, the use of the same in an IP-over-transport network environment implies a significant paradigm shift towards integrated control and management, and while control planes can address automation of functions such as service provisioning, recovery, and routing information dissemination, other facets of network management such as inventory management, failure correlation and detection, network monitoring, etc., have not yet been addressed successfully. Given the diversity of the current management practices, it is unlikely that a unified management framework for both IP and transport networks will be feasible in the near future.

It is the authors opinion that a more successful approach may be to utilize tried-and-tested management practices for the IP and transport networks separately, and focus on automating the current processes of coordination for specific operations in the management ecosystem. Within a carrier organization, inter-layer operations are described as workflows, defining the execution flow of atomic operations as well as the information exchange required between them. In this scenario, this paper presents a proof-of-concept prototype of a Service Oriented Architecture (SOA) [6] based framework proposed in the EU project ONE [1] for facilitating programmable cross-layer orchestrations of management functions. The architecture proposes abstraction of basic network management operations available within the IP and Transport management ecosystem as *web services*, while the Business Process Execution Language (BPEL) [7] is used to *program* existing operation workflows within the ONE framework. The support for programmable cross-layer coordination implies that the ONE framework can easily automate existing workflows and can also facilitate introduction of external third-party management

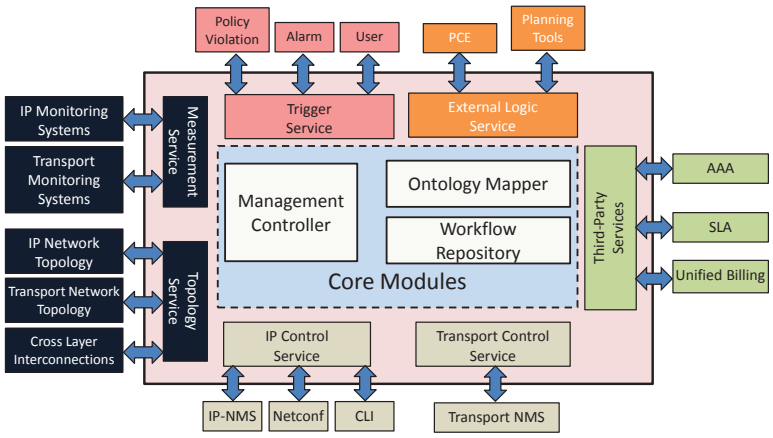

Fig. 1. The ONE framework for an IP-over-optical multi-layer architecture. [1]

subsystems such as the Path Computation Element (PCE) and the Authentication Authorization and Accounting (AAA) systems. [8]. In the proof-of-concept implementation, we developed a limited set of management services, including a user-interface to initiate operations, IP and transport configuration services to perform configuration operations in the IP and the transport networks, respectively, and a topology service to provide multi-layer topology information and demonstrate coordination between the same to provision an IP link and perform IP offloading. The rest of the paper is organized as follows. In Section II, we present an overview of the SOA based framework proposed in ONE and the proof-of-concept implementation presented in this paper. Section III provides the performance results while Section IV concludes the paper.

II. Overview of the ONE Framework for Inter-Layer Management Coordination

In this section, we first present the overview of the ONE framework, and then go on to outline the proof-of-concept implementation developed in this work. The ONE framework is presented in Fig. 1, and consists of three core modules responsible for orchestrating workflows for inter-layer coordination as well as a number of auxiliary services responsible for implementing atomic network management functions. The framework is designed on the Service Oriented Architecture principle, where all operations are abstracted as services.

The ONE framework is *reactive*, i.e., all operations are triggered by events (automated or user-generated), which are received by the *Trigger Module*. As shown in the figure, an event can be a policy violation, an alarm or can be generated by the operator. The trigger module processes incoming event notification(s) in order to generate a *trigger*. Each trigger is associated with a specific multi-layer operation and the corresponding workflow definition is available in the Workflow Repository. The other auxiliary services include web services to perform configuration in the IP and the transport networks, namely the IP and Transport Control services, which interact with the respective NMSs or can use other interfaces such as the Comand Line Interface (CLI) or Netconf [9] to configure network elements. Apart from configuration services, other auxiliary services provide interfaces to other external sub-systems to gather information necessary for inter-layer coordination: for example, the Topology service provides information about the multi-layer topology, the measurement service provides interfaces to measurement systems in the IP and transport networks, while the External Logic Service provides interfaces to components such as the PCE which can be used for inter-layer path computation.

These services provide the necessary set of atomic management functions and the Management Controller orchestrates co-ordination between these services using workflow definitions to execute inter-layer coordinated operations. An advantage of using SOA is that third party systems can also be integrated in the architecture with ease.

A major challenge in the use of this architecture remains in its capability to adapt and evolve with the changes in the external subsystems. For example, the configuration of IP elements can change significantly based on the type of vendor and the operating system used (e.g., version of IOS), and to this end the ONE framework proposes the use of ontology based translation within ONE. Using ontology definitions, the Ontology Mapper can perform request/response translations so that workflow definitions within ONE remain unaffected even as the interfaces to the external subsystems change.

It is clear that the proposed framework is not intended as a unified Network Management System. The objective of the ONE framework is to facilitate coordination for a limited set of operations in multi-layer/multi-vendor network settings. In the ONE project, the effectiveness of the framework to perform coordination is evaluated based on three use cases, namely: 1) Automated IP link Provisioning, 2) Automated IP traffic Offloading, and 3) Coordinated self-healing [8]. The use cases are designed to highlight the capability of the ONE framework to respond to a wide variety of events observed in carrier networks, providing efficient inter-layer coordination and drastically reducing the time (as compared to manual coordination) for inter-layer operations.

In our proof-of-concept implementation, we developed a

limited set of functionalities proposed within the ONE framework in order to demonstrate the capability of existing SOA technologies to facilitate coordination. We are in the process of constructing a multi-layer test-bed, but in this implementation, due to the lack of actual network hardware, we only present the implementation for two use cases, namely, IP link provisioning and IP offloading. We implemented services with limited capability for configuring IP networks and also developed a Topology service to provide multi-layer topology information as well as access information (IP addresses, authentication credentials) required for configuring IP routers. Orchestration of workflows was facilitated using BPEL, which is designed for composition, orchestration and coordination of business processes [7] and is ideal for orchestration of web services.

We describe the workflows and the services developed in this proof-of-concept based on the example workflow for IP offloading as shown in Fig. 2. The top level indicates the different mechanisms used to trigger the IP offloading workflow, and the next two layers indicate the BPEL processes. The advantage of BPEL is that a process defined in BPEL is exposed as a web service and can therefore be used by other workflows, thus helping reduce complexity of large workflows.

We implemented services with limited capability for configuring IP networks and also developed a Topology service to provide multi-layer topology information. We demonstrated how different events such as a command via a UI and a link overloading event obtained via observing IP link traffic using an SNMP [10] Listener can trigger the same workflow. The lower layers in Fig. 2 consist of implementations of the auxiliary services indicated in the ONE architecture.

The IP NMS service (termed IP Control Service in Fig. 1) consists of a limited set of functions capable of configuring IP interfaces and static routing rules on IP routers. In the implementation, we did not implement the Ontology based translation capabilities, and instead developed services using the Factory Design Pattern to generate vendor specific configuration objects for Cisco and Juniper routers. The implemented functions consist of configuring IP interfaces using IP address and subnet information as well as configuring static routing rules on IP routers using the command line interface.

Based on the requirements of the two use cases, namely IP Link Configuration and IP Offloading, the primary capability required from the Transport Configuration service is to create a circuit. Owing to the lack of actual hardware, in our implementation, we emulate the creation of a circuit in the transport network by setting up GRE tunnels [11] between IP routers deployed via the IP NMS web service.

The Topology service is responsible for retrieving topology information from the Topology database. The Topology database maintains multi-layer topology information and provides information about available (free) IP interfaces on routers that may be used during multi-layer operations. In the case of our emulation, the free interfaces can also be indicated as Tunnel interfaces and relevant information used for creating GRE tunnels. The Topology service is also responsible for providing access information and authentication credentials to routers in order to configure the same, which is required by the IP NMS web service in order to access the routers.

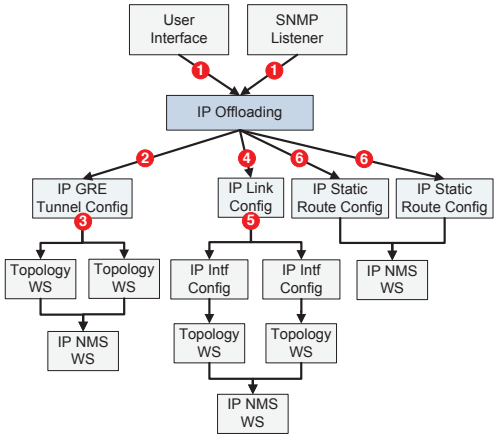

Fig. 2. IP Offloading Functional Flow.

We now describe the workflows indicated in Fig. 2:

1) As mentioned above, the IP Offloading use case is initiated by: A) the operator using the user interface; or B) the SNMP Listener which observes traffic on IP links and initiates offloading when it surpasses a specified threshold. The inputs include the source and destination routers between which an IP link should be created for offloading, and the necessary routing rules to be configured on the routers to offload traffic onto the new link. In the first scenario, all inputs necessary for offloading are provided by the operator, while they are pre-configured in the second scenario.

2-3) The IP Offloading process invokes the IP GRE Tunnel Config BPEL process to create a GRE tunnel between the source and destination routers as specified in the input to the IP Offloading process. This process first invokes the Topology service which provides the tunnel interface ID as well as the IP addresses to be used for creating the GRE tunnel between the source and destination routers, while the IP NMS service is used to configure the tunnel interfaces on the routers.

4-5) The IP Offloading process invokes the IP Link Config BPEL process, which in turn invokes the IP Interface Config BPEL process for interface configuration on the source and destination routers. The process uses the topology service to obtain router access information and then uses the IP NMS service to configure the IP interfaces on the source and destination routers using the IP addresses provided in the input.

6) The IP Offloading process invokes the IP Static Rule Config BPEL in order to configure static routing on the source and destination routers.

III. Test Bed Setup and Performance Evaluation

In current systems, the required time-scales for inter-layer coordination is in the order of days due to manual interactions,

so an automated system to mimic the same in the order of seconds is a significant improvement. However, if performance penalties (execution time) with BPEL are high, our solution may not be suitable for time-critical inter-layer operations such as coordinated self healing. In this section we attempt to highlight that the performance penalties with BPEL are small enough in order to be usable for time-critical inter-layer coordination operations.

In our tests, we implemented two test cases which were developed using: 1) BPEL and web services and 2) an integrated JAVA application. These test-cases are described below. In both test cases, IP configuration was performed on Cisco 7200 routers emulated in GNS3 [12].

In the first test case, one IP interface configuration is applied to one network node repeatedly, and measurements are made while varying the number of repetitions made. In this test case, a telnet connection is made to the router, after which a single interface is configured on the router. After successful configuration, the connection is terminated.

In the second test case, an IP link between two routers is configured on a given network topology. The BPEL implementation first calls the Topology web service twice in order to retrieve IP interface configurations for the two end routers, and then these configurations are applied to the network nodes by passing them to the IP NMS web service.

As the web services were implemented using JAVA, we duplicated the source code used for the web services and the JAVA implementation in order to ensure that there was no performance mismatch due to code optimizations.

Figure 3 shows the comparative average execution times for a single IP interface configuration during the test case 1 runs, while Fig. 4 presents the average execution times for each network node configuration during the IP link creation test case. The horizontal legend shows the repeat count for both implementations. The values shown in the figures are the total execution time divided by the repeat count.

From the figures, it is clear that there is a penalty when using BPEL, which is primarily caused by the invocation of web services in BPEL rather than using direct function calls as in JAVA. This can be seen based on the increased gap between the execution times in test case 2 where the number of web-service calls are higher. While it is clear that the performance gap will increase with the complexity of the workflows (i.e. more web-service calls) the performance gap is observed to be of the order of 10s of milliseconds which

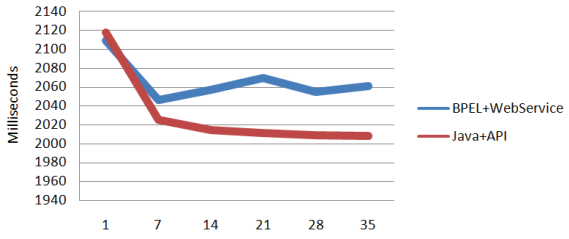

Fig. 3. Case1: IP Interface Configuration.

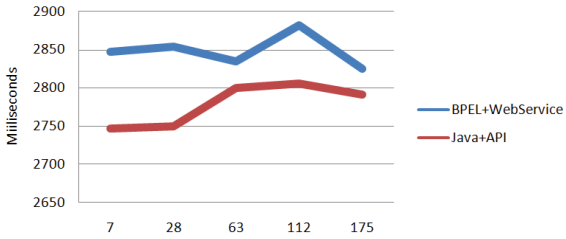

Fig. 4. Case2: IP Link Configuration.

is an acceptable overhead when we take into consideration the ease of maintaining and developing BPEL workflows over dedicated JAVA applications.

IV. Conclusion

This paper presents the framework for multi-layer/multi-vendor coordination proposed in the EU project ONE [1], and outlines a proof-of-concept implementation developed using web services and BPEL. The implementation presents two multi-layer coordination operations, namely IP link configuration and IP offloading and provides an overview of the implementation of the services as well as the BPEL processes used to facilitate this coordination. We also present a basic performance study to show that the framework proposed in ONE, using BPEL and SOA, can achieve acceptable performance in terms of execution times as compared to traditional programming approaches such as JAVA.

Acknowledgment

This work was supported by the European Commission through the ONE project in the Seventh Framework Program (FP7), contract number INFSO-ICT-258300. CRAAX authors also acknowledge the support received by the Spanish Ministry of Science and Innovation (contract TEC2009-07041) and the Catalan Research Council (contract 2009 SGR1508).

References

[1] FP7 EU Project ONE, http://www.ict-one.eu.
[2] Multi-Technology Operations System Interface (MTOSI), release 2.0, TeleManagement Forum, http://www.tmforum.org.
[3] *Internet2 Headroom Practice*. [Online]. Available from: https://wiki.internet2.edu/confluence/download/attachments/17383/Internet2+Headroom+Practice+8-14-08.pdf.
[4] OTN ITU-T Recommendations on ASTN/ASON Control Plane. Available from: http://www.itu.int/ITU-T/.
[5] E. Mannie, "Generalized Multi-Protocol Label Switching (GMPLS) Architecture," IETF RFC 3945, October 2004.
[6] T. Erl, "Service-Oriented Architecture: Concepts, Technology, and Design", Prentice Hall PTR, (2005).
[7] Web Services Business Process Execution Language Version 2.0. OASIS Standard. http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.pdf. 2007.
[8] M. Yannuzzi et al., "Bridging the Interoperability Gap Between the Internet and Optical Network Management Systems," NOCOC I 2011, Newcastle upon Tyne, UK, July 2011.
[9] R. Enns, "NETCONF Configuration Protocol," IETF RFC 4741, December 2006.
[10] D. Harrington, R. Presuhn, and B. Wijnen, "An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks," IETF RFC 3411, December 2002.
[11] Generic Routing Encapsulation (GRE), RFC2784. Available from: http://www.faqs.org/rfcs/rfc2784.html.
[12] GNS3 graphical network simulator. http://www.gns3.net/.