A Logic-based Approach for IP Network Services Management and Configuration

Pedro Alípio, José Neves and Paulo Carvalho
Department of Informatics, University of Minho, Campus Gualtar
4710-057 Braga, Portugal
Email: {pma,jneves,pmc}@di.uminho.pt

***Abstract*— Most of the network service management systems rely on informal specifications, hard-coded programming and relational databases to store and manage network services. As a result, such systems may not be correct facing their requirements and they may not be flexible enough to perform network service management efficiently. This paper presents ongoing work toward an innovative approach, based on knowledge representation, to formally specify the contractual, administrative and technical contents of Service Level Agreements, and the network service management processes and their orchestration promoting network service autonomic management and configuration. By using a knowledge based formal framework and an inference engine capable of reasoning over concepts, relations and changes of state, it is possible to create a more flexible and robust ground for specifying and implementing autonomic and adaptive management tasks.**

I. Introduction

Frequently, customer network services are expressed through Service Level Agreements (SLAs), where a technical part, called Service Level Specification (SLS) is defined. While the SLA includes legal, administrative, and economic information, the SLS includes edge-to-edge IP level information about the offered services' quality. SLA and SLS contents are often specified through XML or Database Schemas. Service Management Systems are usually applications which rely on human operators to add, modify or delete information about services and customers, and to verify if the service requirements are being assured. In case of performance degradation, caused by a link failure, or by any abnormal situation in the network operation, it is up to the operator to decide which actions should be taken.

Consider a huge ISP running a multiservice network with thousands of SLAs with different Quality of Service (QoS) demands. Upon the occurrence of a link failure, it is impossible for a human operator to handle, in a proper way, tens or maybe hundreds of alarms alerting for SLA QoS violation. A human operator will probably take a long time to reconfigure the network in order to reallocate resources to those services. Additionally, the time spent on trying to solve the problem may itself violate the agreed service availability levels. A convenient solution would require a system which automatically performs an action, according to the ISP policies in place, whenever an SLA QoS is violated. In fact, unless the action requires operations such as a node or a link replacement, management actions may be triggered and performed automatically without human intervention. Those actions may be classified as Automated and Interactive actions. Actions which take short time periods to execute, i.e., seconds or a few minutes, and may be performed without any user intervention, i.e., do not require interactivity, are considered Automated Actions. Actions which require interactivity, e.g., filling out fields in web forms, changing the service QoS requirements or price, consequently taking time to be carried out, maybe hours or days, are considered Interactive Actions. Operations carried out not frequently, which may affect a wide region of the network domain, e.g., changing the routing policies of the network domain, are also considered Interactive Actions.

A formal specification of network services management semantics is required as the building blocks to create the reasoning mechanisms to allow the development of Self-managed ISPs. The explicit or formal characterization of atomic entities (concepts) in a domain and relations that may be established among them is called an ontology [1], i.e., an ontology defines a common vocabulary for information interchange in a knowledge domain. It includes machine-interpretable definitions of basic concepts in the domain and relations among them.

This paper reports ongoing work toward a new approach to SLAs specification and management based on a high-level ontology. The present proposal was formally specified using the Flora-2 system [2] which includes, among other first-order logics extensions, the F-logic (FL) [3], higher-order and meta-programming (HiLog) [4] and Transaction Logic (TR) [5]. These frameworks include several valuable features for both specification and implementation of a network service management engine. On the one hand, TR allows the separation of the service management processes flow logics from their implementation. On the other hand, FL allows the development of frame based knowledge specifications including the concepts and relations necessary to reason about network services instances. Furthermore, through meta-programming is possible to include meta-predicates to check the consistency of service specifications and the system's correctness.

This paper is structured as follows: Section II debates related work and state-of-the art in network services specification and management, and in knowledge representation; Section III points out the requirements for SLC/SLA speci-

fication; Section IV explains the SLA management process and Section V presents the conclusions and future work.

II. Related Work

Many of the network services management research groups have been more committed to SLS definition and management [6], [7], [8], [9] rather than obtaining more extensive approaches as toward SLS, SLA and SLC autonomic management.

Usually, pure XML is the preferred network services specification language. However, XML has well known limitations, namely in creating non-hierarquical relations between elements. Lately, ontologies are being mostly used to bring semantics to the World-Wide Web (WWW). The WWW Consortium (W3C) is developing the Resource Description Framework (RDF) [10], a language for encoding knowledge on Web pages to make it understandable to electronic agents searching for information. The Defense Advanced Research Projects Agency (DARPA), in conjunction with the W3C, is developing DARPA Agent Markup Language (DAML) by extending RDF with more expressive constructs aimed at facilitating agent interaction on the Web [11]. More recently, the W3C Web Ontology Working Group is developing OWL (Web Ontology Language) [12] based on description logic, maintaining as much compatibility as possible with the existing languages, including RDF and DAML.

The Service Oriented Architecture (SOA) community is also using ontology based languages to specify semantic web services, such as DAML-S [13], OWL-S [14], Semantic Web Services Language (SWSL) [15], Web Service Modeling Language (WSML) [16], focusing on web services discovery, composition, choreography and orchestration. Relevant work regarding Web SLA specification and management has also been developed [17], [18], [19].

Most of the ontology specification languages rely on XML and RDF only as underneath platform [11], [20], [12]. As a result, these ontologies may be validated, parsed or transformed with regular XML tools. Nevertheless, reasoning (queries, verification and taxonomical inference) is often performed by knowledge based systems using other formalisms. Several of these tools and formalisms, such as Flora-2 based on the FL and TR frameworks, integrates frames, rules, inheritance, and transactions, consisting of far more powerful languages than those exclusively designed for the Semantic Web or for the Semantic Web Services. The main drawback of these languages is interoperability, i.e, exchanging information with other systems or software components. Nevertheless, efforts are being made to develop a FL XML Schema and tools to transform XML documents into FL [21] which may be used to overcome this problem. A Java package is also being developed for Flora-2 to allow using it as a reason engine for knowledge based desktop or web applications. Furthermore, the Web Service Modeling Language (WSML), which is based on the FL and TR, and the Web Service Modelling eXecution environment (WSMX) [22] are also in progress.

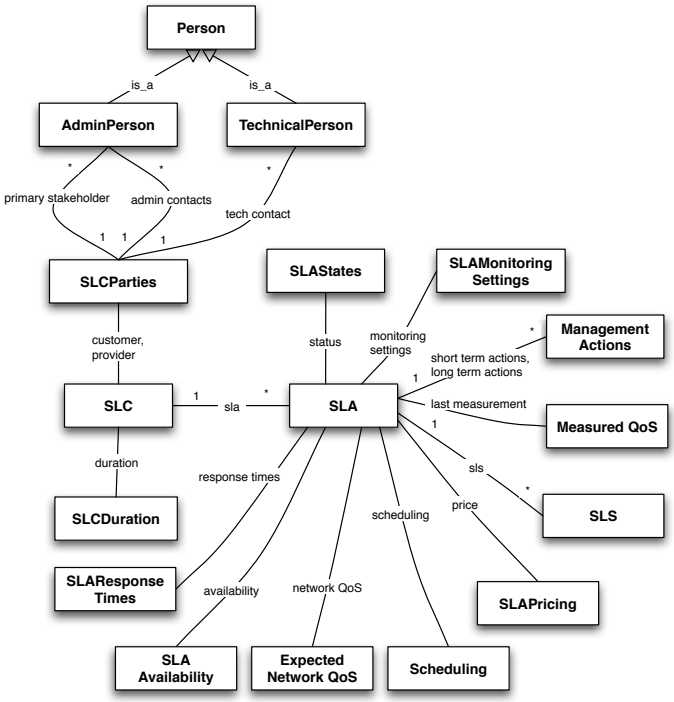

Fig. 1. SLC/SLA Ontology.

As WSML and WSMX still in progress, the present proposal follows a logic based approach, including the FL and TR formalisms to specify SLCs, SLAs and SLS, which may be implemented in the Flora-2 system, including features that allow reasoning over concepts, relations and changes of state. This will provide a flexible and robust ground for specifying and implementing autonomic and adaptive management tasks in multiservice environments.

III. Network Services Specifications

In the present proposal, network service specifications include three different abstraction layers: (i) the contract information layer; (ii) service agreement layer; and (iii) technical specification layer. Each of these sub-specifications has its own context and requirements, i.e, each layer includes a different perspective over network services including a different knowledge base and operations. For example, a Service Level Contract (SLC) includes information about the parties involved and the contract duration, a Service Level Agreement (SLA) includes which type of service the customer requires and what are his expectation concerning matters such as the availability or pricing, finally, the Service Level Specification (SLS), includes technical service requirements. SLCs, SLAs and SLSs are interrelated as Fig. 1 illustrates.

A. Service Level Contracts

An SLC includes information about the parties involved in the service negotiation, the contract validity period and the services enclosed in the contract by including one or several

SLAs. The parties involved are a service provider, usually an ISP, and a customer, which eventually may be another ISP. In Fig. 1, both entities, provider and customer, are represented by one or more persons given in the specification by the relation *primary stakeholder*. Administrative and technical personnel are also referred to be responsible for further negotiations or technical problems during the validity of the contract, respectively. The contract validity period is given by the relation *duration* including the start date and the end date for the contract. In addition, a contract may include several SLAs, as a customer may want, for example, a telephony service, e.g., VoIP, and high throughput data service, e.g, for database related traffic. By following this approach, it is not necessary to create two contracts, one for each service, for the same customer and validity period.

B. Service Level Agreements

SLAs include information about which network service(s) are subscribed by the customer and which are his expectations including service availability, network QoS, response times and pricing. In Figure 1, the relation *response times* includes information about the Maximum Setup Time, i.e, the maximum time spent in network configurations to start offering the service with the agreed QoS and the Mean and Maximum Time to Repair, i.e., the mean and maximum time to repair the service in case of any SLA violation. The relation *availability* includes information about the Unavailable Time Limit and the Mean Down Time, i.e, the maximum and mean time, within the service scheduling, in which, the service is not available. The relation *Network QoS* expresses the level of network QoS agreed with the customer. The relation *last measurement* is used to obtain the network performance metric values, in order to verify if the expected QoS is being somehow violated. In case of violation of any of the agreed parameters, two types of actions may be performed, Automated Actions or Interactive Actions. As explained in Section I, Automated Actions are actions which do not require user interactivity, therefore they may be executed automatically. On the other hand, Interactive Actions, require user intervention, usually consisting of changing a set of parameters which require the customer or the ISP approval. The relation *price* expresses the service price which depends on the agreed QoS, service scheduling, and other SLA parameters negotiated with the customer. Finally, SLAs may contain one or several SLSs, as some services may be divided, in technical terms, into more than one different service types. For example, a VoIP service may be divided into a telephony service and a signaling service.

C. Service Level Specifications

SLSs include technical information regarding the service scope, the QoS requirements, and the implementation of the services, i.e, the required configurations for the components of a network with QoS support such as classifiers, traffic conditioners, traffic shapers and active queue managers. SLS is not the main focus of this paper as it is more concerned

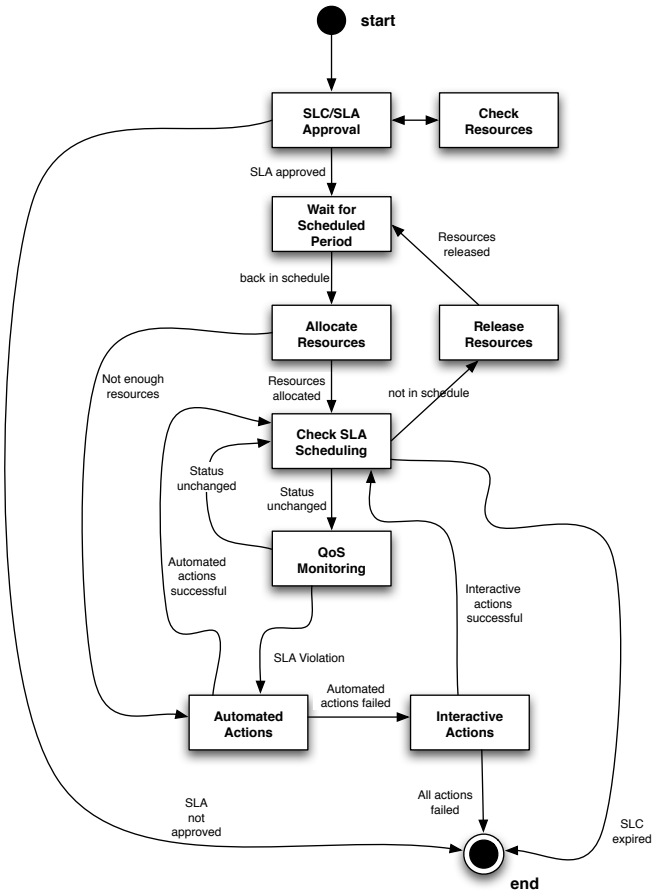

Fig. 2. SLA Management Process.

in debating a high-level specification for automated service level management of network services considering the highest abstraction layers of those services (SLCs and SLAs). For information about a similar approach focused on SLSs we recommend further reading [23].

IV. SLA Management

The SLA goes through several operations and changes of state during its lifetime, therefore it is specified as a workflow. Figure 2 illustrates the SLA processing phase through a graph, where the arrows stand for conditions and the nodes stand for processes. The SLA management includes the following steps:

- initially, the SLA is validated and required for both ISP and customer approval. This operation consists of validation, i.e, type and cardinality checking, and consistency checking verifying if the new SLA instance meets the specification requirements, e.g., verifying if the SLA scheduling matches the SLC duration. If the SLA is valid and consistent, the system proceeds by checking if there are enough network resources for the services included in the SLA, otherwise the SLA cannot be approved;
- the SLA processing proceeds by checking if the current time is included in the SLA scheduling, then all the

network services included in the SLA, given by the set of SLSs, get network resources. In an implementation scenario this is achieved by mapping SLSs into network configurations, e.g., using a network management solution based on COPS or SNMP.

- after this stage is completed, the SLA is monitored in order to check if the network services included in the SLA are being offered with the agreed QoS. If an SLA violation occurs, two kinds of defined actions may be applied to recover from such a state: Automated and Interactive Actions. Automated Actions are attempted in the first place as they may solve the problem much faster. If these actions fail to recover from the violation state, Interactive Actions are then executed. For example, a router queue reconfiguration is considered to be an Automated Action as it does not need customer approval. On the other side, a QoS downgrade or a price adjustment is considered a Interactive Action as it requires the service subscriber approval. This actions are crucial to keep providing the network services, maximizing the network usage and ISP profit;
- if any of these actions are able to solve the problem, then the SLA management engine loops back to the scheduling checking step. If the SLA is out of the scheduling period, then the allocated network resources are released and the SLA processing is suspended. As soon as it is checked to be back in schedule the network resources are reallocated and the SLA processing is resumed. When none of the SLA violation recovery operations is successful the SLC is rescinded. It is also rescinded when the contract expires.

V. Conclusion

FL and TR include the necessary syntax and semantics to be used as a formal ground to specify SLCs, SLAs, and SLSs. Moreover, it allows to specify processes as flows of operations which may include non-determinist execution and backtrackable updates. Thus, this ongoing work, besides proposing a new approach in SLA management and a formal specification for a network service ontology, it also includes the necessary reasoning processes to implement an autonomous decision mechanism, capable of recovering from SLA violations. Work is in progress to create interfaces for managing the SLC, SLA and SLS instances and external process connections, namely, for network QoS configuration and monitoring. Specification and implementation details can be obtained directly from the authors.

Acknowledgments

A PhD grant provided by Fundação para a Ciência e Tecnologia (SFRH/BD/17579/2004) is gratefully acknowledged.

References

[1] T. R. Gruber. Towards Principles for the Design of Ontologies Used for Knowledge Sharing. In N. Guarino and R. Poli, editors, *Formal Ontology in Conceptual Analysis and Knowledge Representation*, Deventer, The Netherlands, 1993. Kluwer Academic Publishers.

[2] Guizhen Yang and Michael Kifer. FLORA: Implementing an Efficient DOOD System Using a Tabling Logic Engine. In *CL '00: Proceedings of the First International Conference on Computational Logic*, pages 1078–1093, London, UK, 2000. Springer-Verlag.

[3] Michael Kifer, Georg Lausen, and James Wu. Logical Foundations of Object-Oriented and Frame-Based Languages. *J. ACM*, 42(4):741–843, 1995.

[4] Weidong Chen, Michael Kifer, and David Scott Warren. Hilog: A foundation for higher-order logic programming. *J. Log. Program.*, 15(3):187–230, 1993.

[5] Anthony J. Bonner and Michael Kifer. Transaction Logic Programming. In *ICLP*, pages 257–279, 1993.

[6] P. Morand, M. Boucadair, R. Egan P. Levis, H. Asgari, D. Griffin, J. Griem, J. Spencer, P. Trimintzios, M. Howarth, N. Wang, P. Flegkas, K. Ho, S. Georgoulas, G. Pavlou, P. Georgatsos, and T. Damilatis. Mescal D1.3 - Final Specification of Protocols and Algorithms for Inter-domain SLS Management and Traffic Engineering for QoS-based IP Service Delivery and their Test Requirements . Mescal Project IST-2001-37961, Jan 2005.

[7] A. Diaconescu, S. Antonio, M. Esposito, S. Romano, and M. Potts. Cadenus D2.3 - Resource Management in SLA Networks. Cadenus Project IST-1999-11017, May 2003.

[8] Danny Goderis, Yves T'joens, Chistian Jacquenet, George Memenious, George Pavlou, Richard Egan, David Griffin, Panos Georgatsos, Leonidas Georgiadis, and Pim Van Heuven. Service Level Specification Semantics, Parameters, and Negotiation Requirements. Internet-Draft, drafttequila-sls-03.txt (work in progress), Oct 2003.

[9] Pedro Alipio, Solange Lima, and Paulo Carvalho. XML Service Level Specification and Validation. In *10th IEEE Symposium on Computers and Communications (ISCC'05)*, pages 975–980, 2005.

[10] Dan Brickley and R.V. Guha. Resource Description Framework (RDF) Schema Specification. http://www.w3.org/TR/rdf-schema, W3C, 1999.

[11] J. Hendler and D. McGuinness. DARPA Agent Markup Language. *IEEE Intelligent Systems*, 15(6), 2000.

[12] Sean Bechhofer, Frank van Harmelen, Jim Hendler, Ian Horrocks, Deborah L. McGuinness, Peter F. Patel-Schneider, , and Lynn Andrea Stein. OWL Web Ontology Language Reference. W3C, 2004.

[13] Anupriya Ankolekar, Mark Burstein, Jerry R. Hobbs, Ora Lassila, David L. Martin, Sheila A. McIlraith, Srini Narayanan, Massimo Paolucci, Terence Payne, Katia Sycara, and Honglei Zeng. Daml-s: Semantic markup for web services. In *Proceedings of the International Semantic Web Workshop*, 2001.

[14] D. Martin, M. Burstein, G. Denker, J. Hobbs, L. Kagal, O. Lassila, D. McDermott, S. McIlraith, M. Paolucci, B. Parsia, T. Payne, M. Sabou, E. Sirin, M. Solanki, N. Srinivasan, and K. Sycaran. OWL-S 1.1 Release. http://www.daml.org/services/owl-s/1.1/, 2004.

[15] Benjamin N. Grosof, Michael Kifer, and David L. Martin. Rules in the Semantic Web Services Language (SWSL): An Overview for Standardization Directions. In *Rule Languages for Interoperability*, 2005.

[16] Holger Lausen, Jos de Bruijn, Axel Polleres, and Dieter Fensel. Wsml - a language framework for semantic web services. In *Rule Languages for Interoperability*, 2005.

[17] IBM Research. WSLA - Web Services Level Agreement Project. http://www.research.ibm.com/wsla, April 2003.

[18] Vijay Machiraju Akhil Sahai, Anna Durante. Towards Automated SLA Management for Web Services. HP Tech.Report, July 2002.

[19] A. Dan, D. Davis, R. Kearney, A. Keller, R. King, D. Kuebler, H. Ludwig, M. Polan, M. Spreitzer, and A. Youssef. Web services on demand: WSLA-driven automated management. *IBM Syst. J.*, 43(1):136–158, 2004.

[20] Dan Connolly, Frank van Harmelen, Ian Horrocks, Deborah L. McGuinness, Peter F. Patel-Schneider, and Lynn Andrea Stein. DAML+OIL Reference Description. W3C, 2001.

[21] Jos de Bruijn and Michael Kifer. F-logic/XML - An XML Syntax for F-logic. WSMO Working Draft. http://www.wsmo.org/2004/d16/d16.2/v0.1/20040324, March 2004.

[22] Armin Haller, Emilia Cimpian, Adrian Mocan, Eyal Oren, and Christoph Bussler. Wsmx - a semantic service-oriented architecture. In *ICWS*, pages 321–328, 2005.

[23] Pedro Alípio, José Neves, and Paulo Carvalho. An Ontology for Network Services. In *International Conference on Computational Science (3)*, Lecture Notes in Computer Science, pages 240–243. Springer, 2006.