Capturing adaptive B2B Service Relationships Management through a Generalized SLA Information Model

Issam Aib[13], Nazim Agoulmine[2], Guy Pujolle[1]
[1]*LIP6 Computing Laboratory, University of Paris 6, France*
[2]*University of Evry Computer Laboratory, France*
[3]*Institut Supérieur d'Electronique de Paris (ISEP), France*
Contact (issam.aib@lip6.fr)

Abstract

Service Level Agreements for service quality assurances are being a hot and complex research topic in both the network management and eCommerce fields. Research in IT Networks and systems management is evolving from device and system component centered management towards a service oriented systems management paradigm. The new entity that is gaining management research focus is now the "service" in all its possible forms and interactions. Service contracts are specified through Service Level Agreements (SLAs), which represent an essential building block that needs to be conceived and managed in a way that catches up the complexity of such a network of services and service interactions. In this paper, we address the central issue of service management by proposing the GSLA, a role-based multi-party SLA model that is intended to catch up the complex nature of service interactivity within a pervasive service driven IT environment. Examples accompanying the concepts we introduce would illustrate the utility and the choices behind each component of the GSLA model.

Keywords: Service Level Agreement, Service Relationship Management, Policy Based Management, Role Based Management.

Introduction

The Information Technology (IT) market represents for the time being a strong and fast growing business field that is driving worldwide business towards a universal IT-based business interactivity model. The advantages of this are numerous and equally so are the driving motivations. In this concern, the models and tools that form the building blocks and components of such an infrastructure occupy a specific importance. Current research in IT systems management is fundamentally gearing from *device* and *system component centered management* towards a *service oriented systems management* paradigm. Instead of the traditional device centered system management, the new virtual entity that requires management research focus is now the "service" in all its possible forms and interactions.

A *service instance* is defined through the specification of its different components and their parameters as well as the intended quality of these parameters which is defined into a Service Level Agreement (SLA). SLA modeling is being a hot research topic in networks and overall systems management, as it constitutes the key starting point of any strong management infrastructure. Such a modeling should take into account current facts in the business market but equally well and perhaps more important the future tendency of such a market so as the modeling would be rightly flexible. Traditionally, an SLA is defined as a contract between a service provider and a customer that defines all aspects of the service to be provided. An SLA generally covers availability, performance, customer service details, and often contains a judicial formal component. A service centric modeling would take into consideration not only the specification of the details of service parameters and service quality parameters. Such a modeling of services independent of the network of service dependencies in which they'll take action does not capture the interactivity between services, service relationships in real scale business environments.

We propose a UML based information model for capturing a generic flexible specification of services and Service Level Agreements. We investigate the role of dynamic Business-to-Business SLA negotiation and deployment in the enhancement of cross-organizational service relationship integration. The relationships that take place in such an environment are not always of the Client/Server type. B2B relationships might equally well consist of peer-to-peer expertise exchange relationships. By peer, we mean that each party is both a service provider and a service consumer (this is to differentiate it from p2p file sharing). In addition, the information model formalizes also service delegation relationships (fig.2) so that to allow for better controlled federated services management. It takes advantage of a role-based paradigm in order to formalize intra and inter organizational service relationships. This modeling of services would represent the initial step in the development of an integrated service specification and management solution in a business-to-business IT environment.

The paper is structured as follows. First, we motivate the need for a Generalized SLA (GSLA) specification and management in the future pervasive IT market. We follow by proposing a UML-based information model that would capture the identified conceptual components that are necessary to be present in the GSLA. Finally, we study a wireless connectivity GSLA case where the need for multi-party SLAs is illustrated.

1. Requirements for a Generalized Multi-Party SLA (GSLA) Specification

As well-defined Service Level Agreements (SLA) should govern service interactions, the modeling of SLAs represents a major building block of any sound Service Management Framework. SLAs are hence required in service interactions regardless of the underlying communication medium whether it is wired or wireless. Because of this, the SLA modeling we propose hereby holds also for wired environments. We also try to identify not only the elements necessary for the specification of the SLA, but also those necessary to take into consideration to be able to manage efficiently those SLAs.

Adapted from [16][4], an SLA (Service Level Agreement) is recognized to be a contract between a Service Provider (SP) and a Service Customer (SC) which is designed to create a clear measurable common understanding of the minimal expectations and obligations about what the customer is requesting and what the provider has committed to provide and at which constraints. The constraints may be of any type and normally include contract scope (temporal, geographical, etc.), the agreed upon billing policy, as well as the behavior in case of abnormal service operation. Hence, an SLA constitutes a legal foundation over which both parties can rely in order to plan their relative businesses and future growth.

We believe however that this view of SLAs as concerning only single client-server relationships is restrained and is unable to catch up all real life situations where complex business relationships involving more than two parties take place and where SLAs are required to fix-up the rules for the well conduct of the business relationships. This situation exists particularly in wireless environments where by nature many parties might be involved into a same service relationship, such as a video conference, a live scene multicast, a multi-player wireless game, and so on. Also, we need to take into consideration that many service relationships start and end up in wireless environments, while the end-to-end connectivity is assured through a combination of heterogeneous wireless and wired networks as is the case in the Internet.

To differentiate the classic Client-Server view of an SLA that is predominant in the literature from the view we introduce here, we will call our Multi-Party SLA model the GSLA (for Generalized SLA). Hence, when we employ the term SLA we mean the usual classic Client-Server SLA, however keeping in mind that an SLA is not but a special case of a GSLA.

The simple form of a GSLA occurs when two parties, say A and B, agree upon a given exchange of services. For example, A renders some service(s) to B and B renders other service(s) to A according to specified constraints. A typical example is that when A and B are physically neighbor Network Operators and the exchanged service concerns bandwidth trunks linking the two adjacent operator domains. In this case, it is both useful and more natural to catch up the service relationship between the two operators into one semantic and structural unit. In this case, the SLA between A and B can contain rules that might specify some actions related to the trunk from A to B if ever the B-to-A service experiences unexpected irregular behavior. Finally, One notices that the most degenerate form of a two-parties SLA is when A's service to B is null. Hence, A becomes a *Pure Service Customer* and B a *Pure Service Provider*.

To be able to manage GSLAs, we need to see how they can be related in real business situations. In the case of an e2e (end-to-end) GSLA, such as a Service Customer (SC) invoking a VoD Service Provider (SP), a chain of GSLAs (Figure 1) spanning many ISPs may intervene so that the e2e VoD service will function properly.

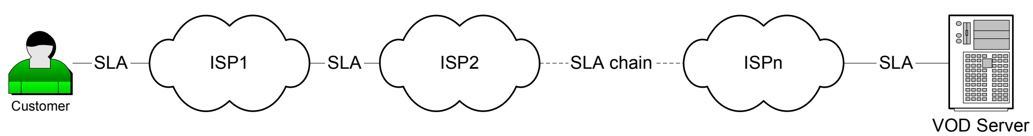

Figure 1: An E2E SLA value chain

This linear SLA relationship is not the unique possible interaction model in a real business situation. A set of SC-SP (Service Customer-Service Provider) SLAs might equally have a hierarchical structure in which a whole SLA references, or is composed up of, a set of other "smaller" SLAs. This is presently the trend of many SPs of various specializations. These SPs are evolving to focus on the growth of their core competency in a cost effective manner. Hence, mitigating the serious worldwide shortage of IT skills and improving the rapid innovation in a value chain or hierarchy of IT Services.

This linear SLA relationship is not the unique possible model of SLA-dependency we could find in a real business situation. A set of SC-SP SLAs might equally have a hierarchical structure in which an SLA references, or is composed-up of, a set of other supporting SLAs.

The last GSLA form we could encounter is that when more than two parties participate into the contract with more or less complex dependencies between all the involved services. This concerns for example situations where multiple SPs tightly cooperate in order to deliver strong e2e QoS assurances. Real cases of such a form of SLA are found for example in MANETS, where all parties (Ad Hoc nodes) participate in the overall QoS assurance policy. A GSLA model should then capture this complex case.

To summarize, a GSLA definition in a universal ubiquitous service-driven Market is governed by various business needs which differ according to the position of a service contractor whether it plays the role of a SC, a SP, a Third Party (ThP), or simply a Party (i.e. plays both SC and SP roles):

- If he is a SC, then he awaits for a clear and valid specification of the GSLA with sufficient guaranties and monitoring sensors that guide the appropriate pricing policy in return to the offered service.
- In the case of a SP, the emphasis is much on expenditure minimization and profit maximization along with clever service offering. For this to happen, standard formal GSLA specification and fulfillment mechanisms along with small risk evaluation results are key differentiation features in a highly competitive IT market.
- For a ThP contractor role, he is generally present for trust purposes. It may take the role of monitoring the delivered service level, or simply as an official contract witness for judicial validation.
- In the Multi-Party GSLA case, the contracting parties are here for a kind of expertise exchange as both are offering a service in return of another service but through clearly specified service guaranties and pricing policies.

After motivating the different requirements for a generalized SLA specification, we now consider the challenge of designing a GSLA information model which best supports the above requirements.

2. A GSLA Information Model

SLA models proposed in the literature [2][6][8][10][16] reveal the same overall structural components. This sustains the idea of coming up with a unified information model for SLAs.

First, we need to differentiate between SLA modeling at a business level and current or future network management technologies; as a universal SLA modeling solution should be able to handle both of them and in a seamless way. This implies that the specification needs to be as open as possible to modular design and evolution. Besides, in modeling our GSLA, we did understand the importance of standardization in empowering universal SLA management solutions. This helped us profit from the work that has been made in the DMTF and IETF for the standardization of system management components. We have referenced such components whenever this seemed plausible as a choice.

A GSLA is defined as a *contract signed between two or more parties relating to a service relationship and that is designed to create a clear measurable common understanding of the role each party plays in the GSLA*. A party *role* represents a set of *rules* which define the minimal service level expectations and service level obligations it has with other roles and at which constraints. The constraints might be of any type and normally include contract scope (temporal, geographical, etc.), the agreed upon billing policies, as well as the expected behavior in case of abnormal service operation.

We identify in Figure 2 the top-level components of the GSLA. A GSLA comprises a set of *parties* joined according to a certain *schedule* in order to realize the contract by playing each one or more *roles*. During the GSLA life cycle, a required behavior or constraint related to the GSLA is captured in the model by the abstract *GSLAPolicy* component. A *GSLARole* inherits indirectly from the GSLAPolicy (see Figure 6). This is to catch the idea that a role is modeled at first approximation by a set of GSLA Rules, and as it participates in defining the behavior of the system, it is derived from the GSLAPolicy component. A *Schedule* component represents the temporal scope during which a GSLA component is valid. A Schedule class is a specialization of a Constraint. A *Constraint* is an abstract class intended to capture any type of logical predicates over GSLA components parameters. A Constraint is always attached to one or more GSLA components. For example, if a party Role is valid only for a determined time intervals, a schedule object attached to this role should be able to carry out this constraint. Finally, as a GSLA is here to guarantee the quality of offered services, the SLA object comprises one or more *Service Packages* to each of which is associated a *Service Package Objective* that some GSLA party is required to guarantee as is specified in its attached role(s).

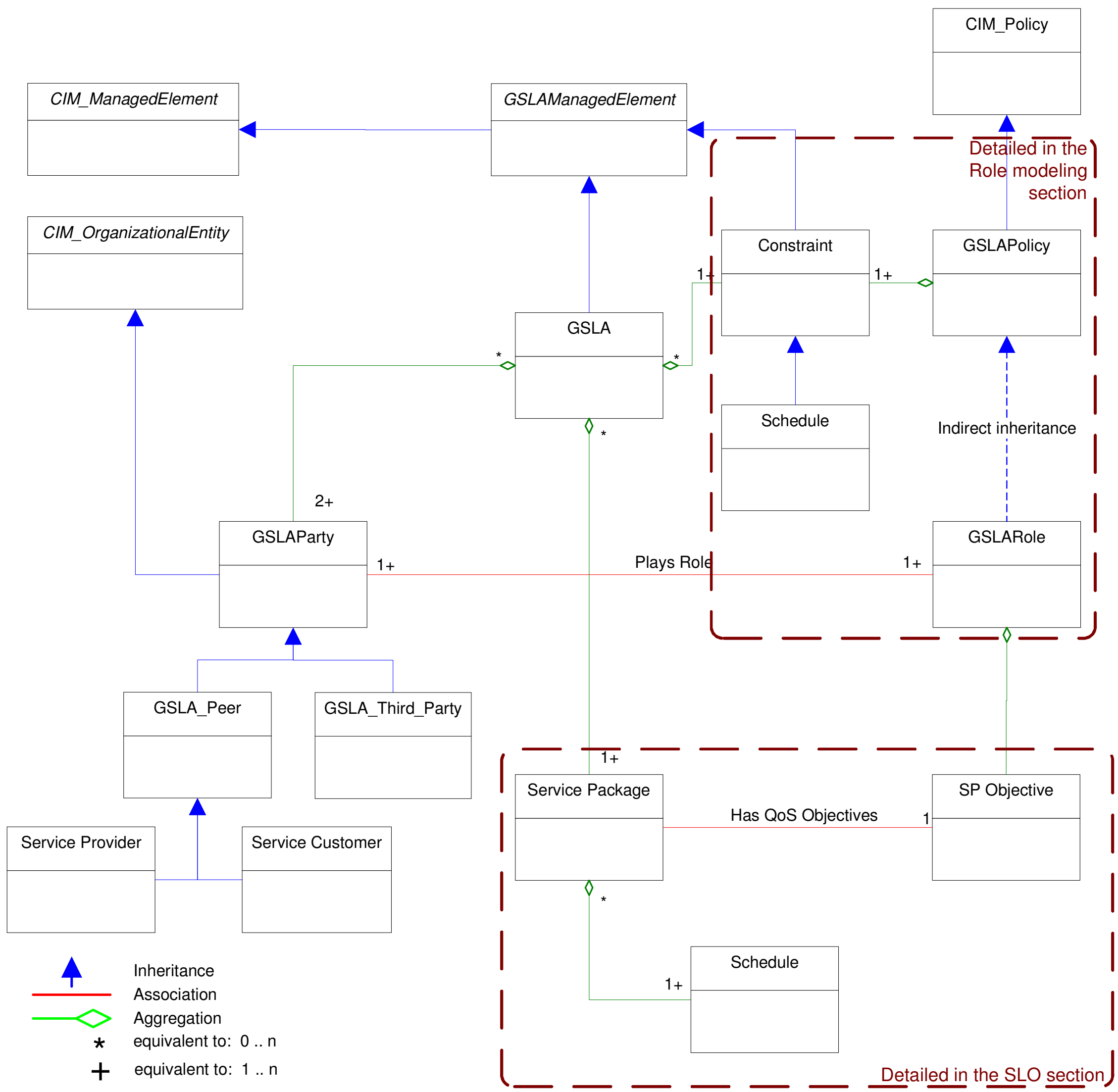

Figure 2: GSLA Top Level Components Hierarchy

We identify in this figure the top-level components of the generalized Multi-Party Service Level Agreement Information Model. This model comprises a set of *parties* joined according to a certain *schedule* in order to realize the contract by playing each one or more *roles*. During the GSLA life cycle, a required behavior or constraint related to the GSLA, is captured in the model by the abstract G*SLAPolicy* class. A G*SLARole* captures in a formal way any behavior that a GSLA party would accept to play.

A GSLA contractor *party* can be either a *Third Party (ThP)* or a *Peer Party*. A ThP is not directly considered in the GSLA. It serves for trust purposes for the validation of the contract and to ensure impartiality. A peer, on the other hand, represents a major player in the GSLA. A degenerate form of a peer concerns either a pure *Service Customer (SC)* or a pure *Service Provider (SP)*. A Service Customer ensures no Service Level Objective (SLO) but is the subject (consumer) of offered services.

Since the differentiation between SC, SP, Peer, or ThP concerns the contents of the role they play, there is no need to have them as separate entities, they do all hold within the GSLAParty entity. This means that the inheritance links from GSLAParty and GSLAPeer represent a specialization rather than an extension to GSLAParty capabilities.

A GSLA Service Package (SPg) is composed of a set of *Service Elements* each of which is related to one or more *Service Resources*. A Service Element (SE) [16] typically represents a single technology-specific service capability, such as an IP connectivity, or an operational capability, such as a help desk support. An SPg represents a group of related SEs that need to be instantiated and managed as a whole. A simple example concerns an Internet web access service, which requires at least an HTTP protocol support and an IP connectivity SEs. An SPg concerns a group of services that are generally offered altogether, such as a web browsing and a mail service and/or a web hosting service. In this case, we notice that service elements can have *requires* relationships that are needed for their operation. An SE can be directly related to a physical service resource. A service resource is intended to be transparent to the customer and represents a basic provider resource, such as an email server, a network element, a processing server, a database, or a stockpile.

Modeling Service Level Objectives

The next building block of the GSLA information model concerns Quality of Service (QoS) modeling. Overall Service Quality is a broad concept covering many performance aspects and numerous measures. It may be defined as [16] "*the collective effect of service performances that determine the degree of satisfaction of the user of the service*". Our GSLA information model captures all aspects related to service quality starting from the SPg Objective class.

A Service Package Objective (SPO), as its name suggests, defines Service Level Objectives for one or more SPgs. Basically, an SPO is a constraint and it can be defined mainly through two different ways. First, it can be defined as a set of predicates or logical expressions over one or more SPg Parameters. This represents a high level way of defining QoS objectives based on direct calculus made over high-level service parameters that are synthesized up (Figure 3) from the basic System Metrics up through System Resource (SR) parameters and System Element (SE) Parameters. The other way around is to calculate the objectives based on QoS appreciations coming from subordinate Service Element Objectives. This represents the high-level compilation of low-level QoS appreciations. This second approach in appreciating the overall Service Quality reflects better the way users appreciates a given service infrastructure, i.e, by giving a final appreciation based on separate 'sub' appreciations over the different service components.

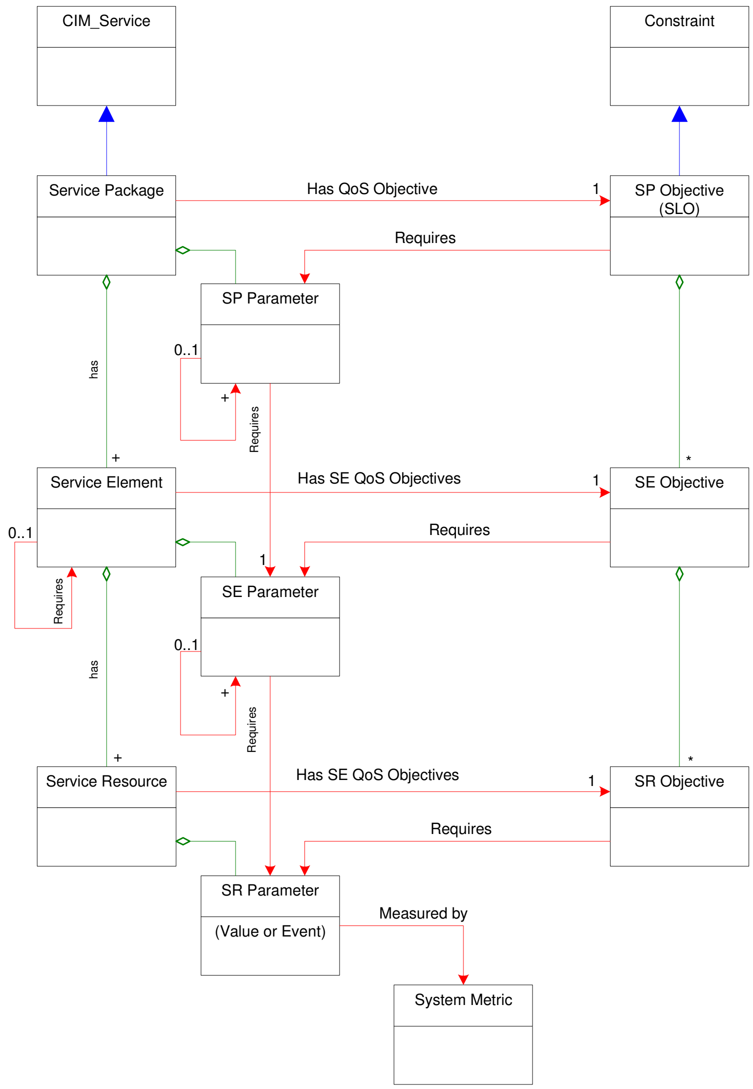

Figure 3: Services, parameters, and Service Objectives structure in the GSLA

Suppose for example that the offered SPg is composed up of a web browsing SE, an email SE, an FTP SE, and a VoD SE. A SPg parameter SPgMeanPerf may be defined equal to Mean(SPgPerfTimeSeries). SPgPerfTimeSeries being a series of SPgPerfElement values. A SPgPerfElement parameter is calculated through a function f(MailSE.Perf, FTPSE.Perf, WBSE.Perf, VoDSE.Perf) of the performance of each constituent SE of the SPg. The value of SPgMeanPerf may then be used by certain SPO, say SPgPerfO, to appreciate whether the overall SPg performance is acceptable or not based on some contracted thresholds. The other way around is to define an SPO that constructs the overall SPg performance appreciation based on the results of the SE Objectives of each SE of the package. In this case, the SPgPerfO might contain a set of logical expressions such as "IF (at least 3 of (WBSEPerfO, WBSEPerfO, WBSEPerfO, WBSEPerfO)) == good THEN SPgPerfO🡐good)".

Role Based SLA modeling and the policy-based approach

We reach now the third and final building block of our GSLA model. Here, we consider the modeling of the behavior each party of the GSLA is required to observe during the GSLA life cycle. As each party plays a role or a set of roles in the GSLA, role modeling should be able to capture all aspects of behavior a party (SC, SP, Peer, or ThP) should have. For this, we will take advantage of the study we did in [14] about policy specification and policy based management systems. The reader is encouraged to refer to this work for more details about the different choices that guided our policy and role related modeling in the GSLA information model.

A system behavior at the lowest view is modeled as a policy. By system, we mean any component participating during the overall GSLA life cycle, be it a person, a software component, a network element, or an organization. Research made in RBAC (Role Based Access Control) systems and security management systems shows that policies are mainly of two kinds: action policies and authorization policies. Conceptually, an *authorization policy* defines conditions for limiting access to or actions over some system components or operation. Authorization policies are subdivided into permissions and prohibitions. An *action policy* on the other hand, defines conditions that need to be met in order to execute some system operations. Conceptually, an action policy is made up of two main components: a (set of) condition(s) implying the execution of a (set of) action(s)

ACTION POLICY = CONDITION(s) 🡒 ACTION(s)

A Condition is a generic term. It can be a temporal condition, a condition over existing system parameters, or a condition concerning specific system states. Because of the special importance temporal conditions and system state conditions possess, further decompositions of a Condition into temporal condition, event condition, and other conditions has been largely accepted within both the networking management and the security management communities. PDL [14] formulates action policies using the Event-Condition-Action (ECA) rule paradigm and extends it by providing a rich event sublanguage allowing only un-interpreted concurrent actions. Events represent a clever mechanism to transport useful local status information to portions of the system that need it. With events we become able to mark history that we want to record and hence have a better understanding of the system “state” information.

We identify three main policy types: permissions, prohibitions, and duties. A *Permission* specifies an authorization to execute a certain action, such as accessing a client's profile data. It can also be a *delegation* for another role to execute an action. A SP can delegate the monitoring of some GSLA parameters to a ThP that the customer is unaware of. In this case the delegation policy represents a formal way to capture the role of that ThP. A policy can also represent a *Prohibition*, that is, a negative permission to access some system components or executing some system actions. Finally, a *Duty* policy represents actions that a role is required to take during system run time. Duties represent the key to QoS policy specification in our model. A duty specifies the actions that must be performed by a role object within the system when specific events occur and a set of conditions are met. The syntax for a duty policy using follows:

<policy type=duty name=policy-name**>**

{**<subject>** domain-scope-expression **</subject>**}

{**<eventlist>** events **</eventlist>**}

<actionlist> actions **</actionlist>**

{**<constraintlist>** conditions **</constraintlist>**}

</policy>

Figure 4: Syntax of a duty policy

A *Role* is modeled through the set of duty, permission, and prohibition policies having their subject domain the party or the group of parties that play that role, as well as the set of Service Level Objectives (SLOs) that it is required to ensure as part of its responsibilities in the SLA.

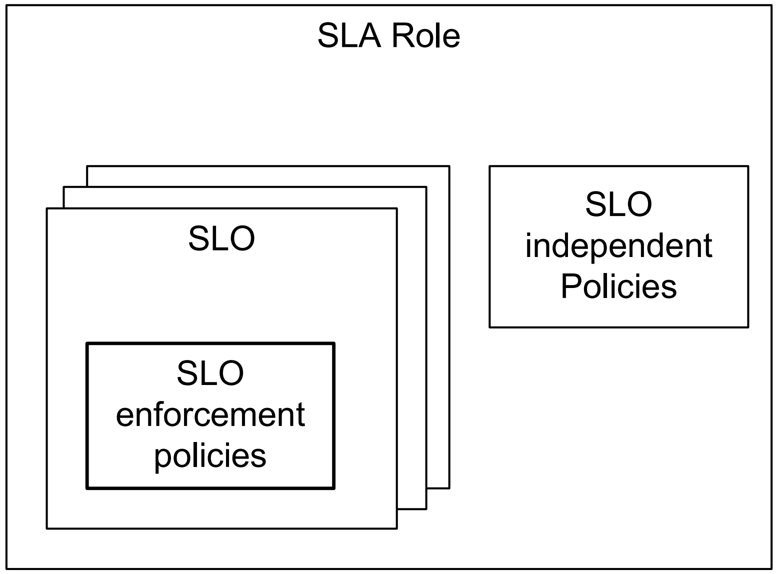

Figure 5: SLO enforcement Policies and Role intrinsic policies

Finally, a *Role-Relationship* is a type of policy set which contains rules defining the rights and duties of roles towards each other. For example, if the SP is required to send a monthly report about a given service quality parameter to the client, then a Role-Relationship object will contain this specification. A Role-Relationship can also include policies related to resources that are shared by the roles. It thus provides an abstraction for defining policies that are not part of the role specifications, but are part of the interaction between the roles.

We give hereby the resulting overall structure of the policy model of the GSLA:

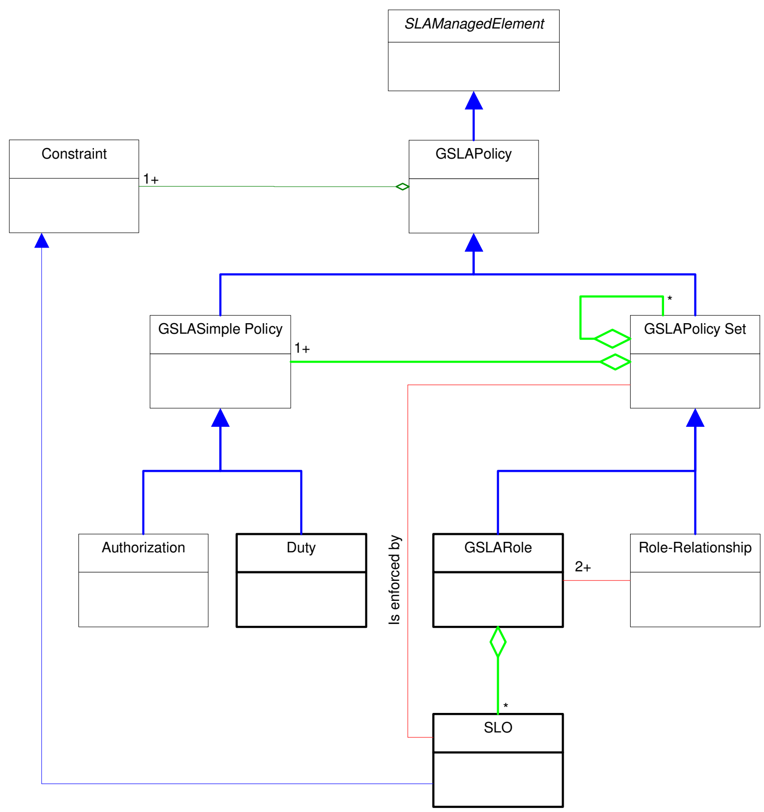

Figure 6: Modeling Roles and Policies in the GSLA

3. Use case: Outsourced Video on Demand Service

We will illustrate how service relationships are captured within the GSLA model through an example of a set of VoD (Video on Demand) SPs which collaborate to offer as much content as possible to their customers and at the same time maximizing their individual benefits. The example might equally hold for internet flight reservation agencies, hotel reservation agencies, and the like.

We suppose that we have 5 VoD Service Providers, which are located respectively in Paris, London, Berlin, Rome, and Madrid. Each VoD-SP serves customers worldwide (over the Internet) and offers a variety of VoD content but specifically specializes in serving local language content. Since, no VoD can contain all the existing Video contents with all existing languages, the VoD SPs would agree to delegate some VoD requests to the VoD-SP which better provides the service. The agreements are not only required for just content shortage, they are also required to meet QoS assurances. The inability to meet a QoS assurance can have multiple sources. If a VoD-SP is congested or the network path from the SC to the VoD-SP is congested at the time of a new SC request session, then it might delegate the task of serving the SC session to another VoD-SP which is able to meet the customer requirements. The delegation is operated on the fly and is managed through specific delegation policies that the VoD-SP incorporates within the SP Role it plays in the GSLA it contracted with the SC.

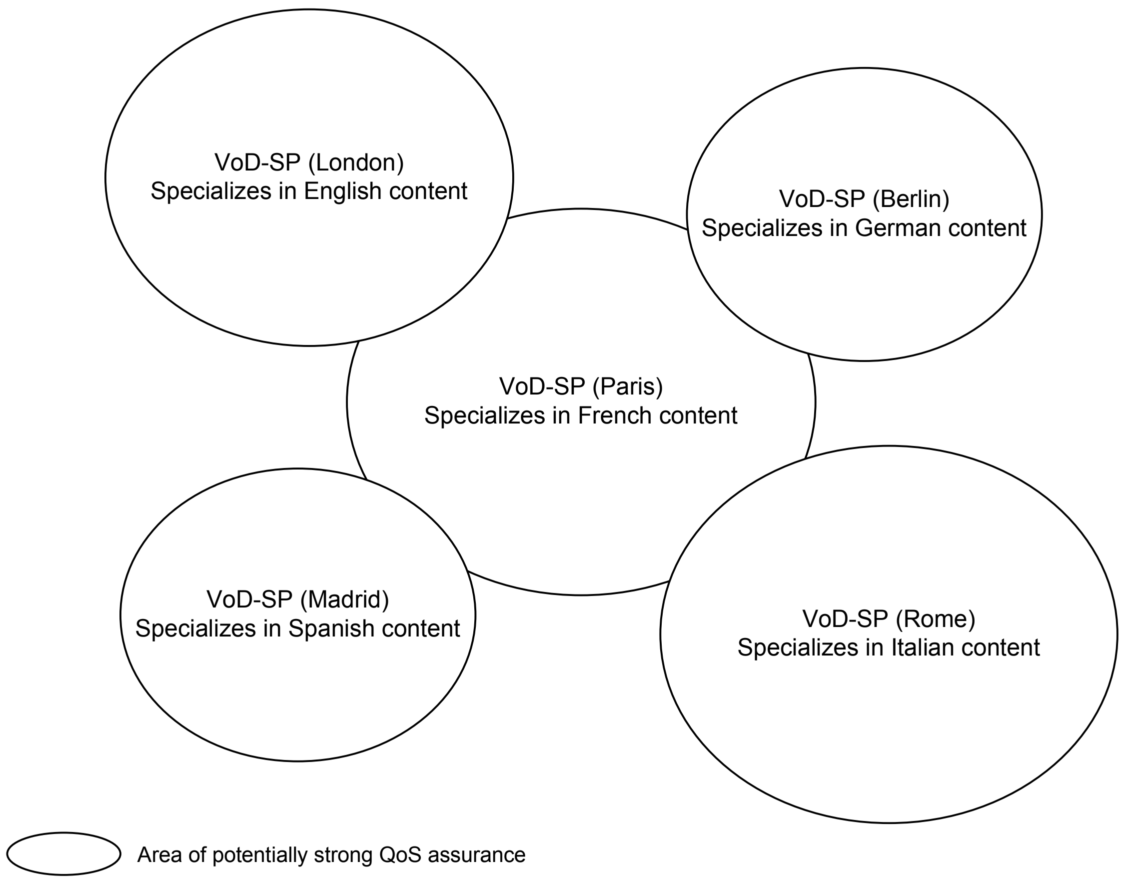

Figure 7: Service delegation between a set of geographically distant VoD-SPs

4. Related work

In the literature, several SLA information models are proposed. The main feature of those SLAs is that they focus on the client/server framework, with an emphasis on isolated individual view of SLAs, in which the network wide view of services and SLAs interactions is nearly absent. [6][18] defines a Web Services based SLA framework. In this work, an SLA is a bilateral contract made up of two parties, a Customer and a Provider, and models party behavior as a set of obligations. Our model enhances this approach by considering a wider definition for the concept of "party" and by bringing forth a modeling of all possible types of behaviors. [20] considers Client/Server SLAs and introduces the notions of client responsibilities, server responsibilities, and mutual responsibilities with respect to non-functional properties. The GSLARole seamlessly captures all those cases. Moreover, it extends it by allowing roles to be attached to more than one party, hence bringing forth a multi-party responsibilities object. [19] addresses the network wide SLA view by proposing a Web Services based overlay network for SLA management. However, their SLA model does not support policies for behavior modeling. Finally, [21] introduces within a client/server SLA model, the notion of service-centric, client centric, and server centric views of an SLA. It proposes an SLA model that supports policies in a basic form. Views represent an important concept and we consider them as a second step in the refinement of the GSLA model. Hence, each GSLA party would have its own local view of the overall logical GSLA. At a transversal plan, refinement of SLOs and high-level behavioral rules represents a second way in defining views over the GSLA components.

5. Conclusion and future work

In this paper, we considered the modeling of Service Level Agreements from the futurist vision of a network of interacting services governed by SLAs that do take into consideration this high-level view within their basic constructs. The main contribution of this work was hence the proposal of the GSLA, a new information model for SLA specification, that is intended to catch up the complexity of future SLA-driven managed networks and systems. We believe the future of the IT business market to evolve towards pervasive multi-party service interactivity. Our model handles the resulting complex multiparty service relationships and considers all cases of service contracts between clients, servers, peers, and third parties. GSLA party behavior is captured into a unique and strong semantic and structural component modeling the role that the party plays. The model defines also a first approximation about how service parameters are specified and monitored. We illustrated the usefulness of our model through. Future work would consider the definition of a complete framework for the management of the GSLA life cycle as well as the development of a test-bed for the validation of the introduced concepts.

6. References

[1] Dinesh C. Verma, New Riders, *Policy Based Networking, Architecture and Algorithms*, New Riders, Nov 2000.

[2] Ron Sprenkels, Aliko Pras, *Service Level Agreements*, Internet NG, April 2001, Deliverable D2.7.

[3] Dinesh C. Verma, Simplifying Network Administration using Policy based Management.

[4] Emmanuel Marilly et al., Alcatel CIT, *Requirements for Service Level Agreement Management*, ©2002 IEEE.

[5] Thanassis Tiropanis, Automating SLA Negotiation a Policy based approach, TM World, Nice, May 2001.

[6] Alexander Keller, Heiko Ludwig, IBM Research Division, The WSLA Framework: Specifying and Monitoring Service Level Agreements for Web Services, JNSM, Vol .11, No. 1, March 2003.

[7] Alexander Keller, Heiko Ludwig, IBM Research Division, *Web Service Level Agreement (WSLA) Language Specification*, Version 1.0, Revision wsla-2003/01/28.

[8] Jim Martin, Arne Nilsson, *On Service Level Agreements for IP Networks*, IEEE INFOCOM 2002.

[9] Akhil Sahai et al., Towards Automated SLA Management for Web Services, HP July 2002.

[10] Li-jie Jin et al., Analysis on Service Level Agreement for Web Services, HP June 2002.

[11] Melissa Buco et al., Managing eBusiness on Demand SLA Contracts in Business Terms Using the Cross-SLA Execution Manager SAM, IBM T.J Watson Research Center, IEEE ISADS, 2003.

[12] Nicodemos Damianou, *A Policy Framework for Management of Distributed Systems*, PhD Thesis, Imperial College of Science, April 2002.

[13] B. Moore et al., Policy Core Information Model, Version 1 Specification, IETF, RFC 3090, February 2001.

[14] Issam Aib et al, Analysis of Policy Management Models and Specification Languages, IEEE Net'Con, October 2003.

[15] DMTF DSP0108, *CIM Core Policy Model*, DMTF, May 12 2000.

[16] TMF, *SLA Management Handbook*, Public Evaluation 1.5, June 2001.

[17] DMTF, *CIM Core Model White Paper*, Version 2.4, DMTF, 2000.

[18] Vijay Machiraju, et al. *Automated SLA Monitoring for Web Services*, IFIP/IEEE DSOM 2002.

[19] Vijay Machiraju, et al. Web Services Management Network, An Overlay Network for Federated Service Management, HP Technical Report, HPL-2002-234.

[20] D.Davide Lamanna, James Skene and Wolfgang Emmerich, *SLAng: A Language for Defining Service Level Agreements*, IEEE FTDS'03.

[21] Gabi Dreo Rodosek, A Generic Model for Services and Service Management, IFIP/IEEE IM 2003.