2007

Employing Intelligent Agents to Automate SLA Creation

Halina Kaminski
University of Western Ontario, hkaminsk@csd.uwo.ca

Mark Perry
University of Western Ontario, mperry@uwo.ca

Employing Intelligent Agents to Automate SLA Creation

Halina Kaminski, Mark Perry
Department of Computer Science, University of Western Ontario, Canada
{hkaminsk | markp@csd.uwo.ca}

Abstract

Service Level Agreements (SLAs) are commonly prepared and signed agreements that form the contracts between a service provider and its customers, defining the obligations and liabilities of the parties. Naturally, SLAs should reflect the business needs of both customer and supplier. SLAs are usually formed through either the adoption of a boilerplate agreement from the provider, or through a mediation/negotiation process between the parties. With the increasing adoption of software supply being implemented as a network service, such schemes are rigid or slow and costly, This paper proposes a system that the parties can use to facilitate both fast and flexible agreements. It proposes automation of SLA creation from a set of Service Level Objectives (SLOs), making use of software agents and adopting a social order function by incorporating it into the decision process.

Keywords: Service Level Agreements, Service Level Objectives, Web Service, Negotiation Manager, Software Agents, Software Service Provision

1. Introduction

One of the many benefits offered by high speed and reliable large scale network services has been the opportunity for software vendors to move rapidly into providing web services, and treating software delivery as a service. This movement away from traditional packaged software requires a different type of agreement between the providers of such software and their customers, which was previously managed by simple licensing agreements, shrink wrap licenses and the like, or, for larger systems, by negotiated licenses. In the service provision environment, the relationship between the provider and customer is typically embodied in Service Level Agreements (SLAs). These are commonly prepared and signed contracts between a service provider and its customers, defining the obligations and liabilities of the parties. Depending on the nature of the agreement, it may take the form of adopting a boilerplate contract from the provider, or for larger scale agreements, a fully negotiated contract. Although the former may satisfy many aspects desired by the customer, it is likely that there are many issues that do not fully meet the customer's needs. Fully negotiated agreements will avoid the inclusion of such non-satisfactory terms, but will require the intervention of personnel who can bring technical, business needs and legal perspectives to the negotiations [1]. It is crucial for both parties to ensure that the terms of the agreement are realistic and meet their requirements, as the financial consequences of failure can be fatal to the business. For example, many service recipients do not require service availability to be guaranteed for 99.99% of the time, as this would be very expensive, and a provider guaranteeing a service that it cannot support may find itself subject to penalties.

This paper proposes the automation of SLA creation from a set of Service Level Objectives (SLOs), employing software agents and adopting a social order function by incorporating it into the decision process. By adopting this system, the service provider can form SLAs and satisfy the need for fast and flexible agreements. Earlier work in SLA management has focused on a bottom up approach, looking to capture managed SLA data [2]. However, the present study concentrates on automatic SLA creation that integrates an effective negotiation process, removing the need for the service provider to engage highly qualified personnel at the time of SLA adoption by the customer. One area in which companies are seeing increased cost is support personnel for their system offerings. Where a company's business is primarily (software) service provision, such costs are critical to contain. In such an environment there is a need to automate with the result of reducing support and management costs [3]. This environment make it very desirable to automate the monitoring, selection, and decision making processes, leaving the service provider more resources to focus on the provision of better services. Generally, most of the business decisions are based on resource prioritization. In this paper by a *resource* we mean any service that is quantifiable, such as application, server, CPU usage, disk space, license etc. Such automation can be achieved by building a

software system that embodies high level decisions and which possesses the properties of autonomy, social ability, reactivity and pro-activeness. Intelligent agents can provide this type of functionality, and an SLA real-time negotiation system that utilizes these features will prove to be a great asset to service provision enterprises.

2. Service Level Agreements

Most SLAs are formed by the provider of services, although it is possible that a customer may come up with a totally original SLA in extraordinary circumstance. Here, we focus on the provision of SLAs from the provider side, but this does not preclude the development of customer originating agreements. Naturally, the provider's perspective is for the SLA to reflect the business goals of the company. It is likely that this will also include the maximization of the customer satisfaction in addition to the limitation of provider liability for problems such as non-performance or failure to meet the quality goals. Rather than simply an end issue, the development of SLAs must be considered a vital step in the business process. Although static, preformed SLAs, which are basically monolithic agreements, may continue to have a role to play in the future, it is desirable to enable clients to select elements of an SLA, or the overall type of SLA, that can meet the requirements of their own situation. Our aim is to provide methods for dynamic, automated SLA creation. As well as benefiting the service provider with automation, such a flexible, dynamic system will allow customers to choose the type of SLA scheme that they want and, consequently, exercise control over the policies for which they have the most concern.

An SLA is not created in isolation, simply to meet the technical needs of the parties, although these need to be considered. The total business strategy of the service provider must be integral to the process. Generally, every SLA should include:

a) the specification and availability of the service to the customer,
b) the performance goals of various components of the customer's workloads,
c) the bounds of guaranteed performance and availability,
d) the measurement and reporting mechanisms,
e) the cost of the service,
f) priorities if service can not be delivered,
g) penalties if the customer exceeds the load,
h) penalties if the provider does not provide service as agreed,
i) schedules for follow-up meetings and interface [3].

SLAs become more complex when the provider offers multiple services such as networking, online databases and end user direct support [4]. Usually, the services provided by such businesses vary both in diversity and intricacy. Many organizations are now utilizing service level objectives (SLOs) as a means of expressing the aims of the company, and to establish parameters for the tracking of the effectiveness of their service infrastructure.

3. Service Level Objectives

A business in the highly competitive and growing online, on demand, service environment must have a clear business plan and define service levels that can be attained. Every resource that is offered to a customer should have an indication what its business levels are and what performance is acceptable to the end-user. These will include performance requirements for applications offered as services, and, in addition, more general business objectives that need to be attained by the system. It has been suggested [5] that SLOs must be realistic, quantifiable (measurable), clear and meaningful, manageable, cost effective and mutually acceptable. The target goals of SLOs have to reflect reality and should be attainable. They also should include the metric definition which contain how the values are measured and reported to the managing authority. Each SLO has to have a meaningful description of the service level such that it can be easily understood by a customer. For example, expressing service performance in packets dropped or server congestion may not be of significance to the end-user. Most importantly, SLOs have to be cost effective. There is a belief that the best SLOs are impractical because they are too expensive to be measured. Simply having the objectives by themselves is not sufficient to provide a high quality service.

A wide variety of service offerings poses another difficulty: to create the best possible SLA from a selection of SLOs from an option pool requires careful consideration and quantification of resource dependencies and the connections between resources wherever possible. As an example, by having two servers that are each capable of handling ten thousand transactions per second does not necessarily mean that we can provide a service of twenty thousand transactions per second to a customer. Both servers could be using a secondary resource that is limited to a lower capacity (a common router for example). Thus the overall performance of the entire business system is unlikely to be a simple summation of the resources available. Many objectives can be embodied in a single SLA, and within the parts of the SLA; for example, with a

network service provision agreement there may be ones dealing with availability, network latency, packet delivery and even reporting. This will clearly differ between clients and so there will be a different, though similar, set of objectives associated with each client.

As an example, a partial SLO set for a resource (SellSolution application) is shown in Table 1.

Application name = SellSolution				..
Service Level	Platinum	Gold	Silver	…
Number of transactions	unlimited	1000	500	…
Initial Response Time	10 sec	12 sec	15 sec	…
Transaction Processing Time	2 μs	3 μs	5 μs	…
Monthly Availability	98%	97%	95%	…
Validity Time Start/End	To be filled at the SLA creation time	To be filled at the SLA creation time	To be filled at the SLA creation time	…
Cost	$500.-	$ 150.00	$ 80.00	…

Table 1. SLOs for a specified resource

It is our goal to be able to set service levels for the resource (service) in such a way that they are not custom made, but predefined and reusable. Ideally there should be many levels for the same resource and the levels would differ in QoS and the cost for flexible offerings. Levels of service can be predefined for the resources of the same type, and the same level of service can be used by many customers. SLOs also express a commitment to maintain a particular state of the service in a predefined period of time. For example, (SLO) gold in Table 1 indicates that the SellSolution will start within 12 seconds from the initial request and every transaction will be processed in less than 3 μs. The customer is limited to perform 1000 transactions. In this service level the application will be available to the user 97% of time and the cost for this type of service is $150.00. The validation time period has to be specified during the negotiation phase i.e. when the customer and the service provider agree to the specific service terms. We will return to this example in section 6.4.

The flexibility of having a pool of SLOs available will result in the existence of a range of service levels and performance metrics for each resource: for each service there will be multiple SLOs on the basis of which SLAs will be offered.

4. Intelligent Agents

A negotiation model is an abstract representation of the structure, activities, processes, information, resources, people, behaviour, goals, rules and the constraints of a computing service environment. From the operational perspective, the negotiation model supplies the information and knowledge necessary to support the SLA creation process. There is a wide variety of information systems that participate in business processes and they are aimed at fulfilling different business requirements. Consequently in business, there are widely varying viewpoints and assumptions regarding what is essentially the same subject. A negotiation framework should have a very carefully "engineered" translation of such different reasoning. To deal with the complex representation issue the system should support the appropriate ontology. The purpose is to provide a shared and common understanding of a domain that can be communicated to people, application systems, and businesses giving some specification of the meaning of semantics of the terminology within the vocabulary [6]. The basic concepts of ontology have also been established in works on intelligent agents and knowledge sharing, such as Knowledge Interchange Format (KIF) and Ontolingua languages [7, 8].

The automation of a negotiation process can advantageously adopt the intelligent agent paradigm. The system can contain one super agent that gets its knowledge from other agents: there can be an agent assigned to each sub-domain, such as a business rules agent, a price agent, an obligations agent, and a resource discovery agent. All of the secondary agents would be reporting to the super agent and only the super agent will engage in the decision making and outer interactions. Figure 1 depicts a Negotiation Model Agent assignment.

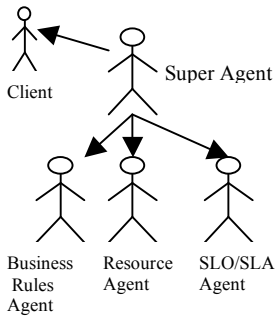

Figure 1. Intelligent Agent Assignments

The Negotiation Manager system is based on a multiple agent framework. There should be one agent per every issue that needs an agreement such as resources, price and business policies. Our model is based on a sequential decision making (i.e. as each party presents an offer, a counteroffer or a decision to accept or decline is made in sequence).

5. Negotiations

To date, most research in service provision has concentrated on how to manage SLA compliance as well as tracking performance for planning purposes. The existence of a variety of measuring tools allows the service managers to measure and track performance of service levels based on the actual service usage. At the same time the results obtained from such metrics can be used in planning corrective actions.

Automated contract creation enables service providers and their clients to make use of technology to create SLAs within pre-planned and pre-approved parameters. Our goal is to use intelligent agents to provide automation of SLA development and creation, (i.e. the creation of the electronic contracts for computing services), which in addition to giving flexibility to the contracting system will optimize the provider's profits. At the same time it will maximize the customer's satisfaction and the ability to be flexible. We are developing a negotiating tool (SLA Negotiation Manager) described hereafter along with the process of negotiation and creation of a SLA from existing business objectives. The Negotiation Manager is a truth based system and it has a system-wide objective of computing an efficient cost-gain relation. Our goal is to provide an interactive negotiation system that would help a service provider to formulate and evaluate an offer, and then send that offer to the client.

The main module of our system will be dedicated to automate processes on behalf of service provider. The overall negotiation process will be modeled as exchanging proposals and counter-proposals between the provider and the customer. Figure 2 presents a state diagram for a negotiation process.

Each negotiation starts with the customer choosing one service offer from a pool of predefined service packs. Usually such offer depends on service price, delivery, quality etc. The initial offers can be pre-defined and stored in a repository or they can be automatically generated by using existing SLOs and current system's state.

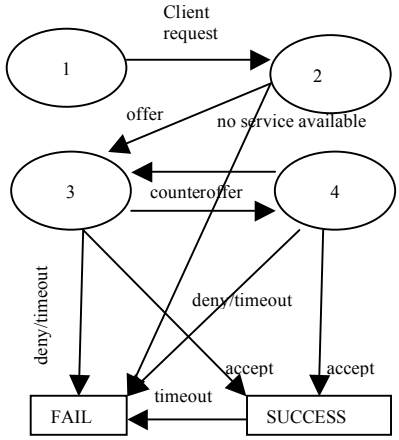

Figure 2. Negotiation Process State Diagram

The provider takes all factors into account and calculates the expected pay-off value function associated with possible offers, and selects the offer that maximizes its payoff. When satisfied with an offer, the customer (client) just sends an acceptance message to the provider and a SLA is finalized. In Figure 2, the transition:

1 –> 2 –> 3 –> SUCCESS

presents such process. If not accepting the first offer, then the client can either abort the negotiations:

1 –> 2 –> 3 –> FAIL

or can send a counter - proposal:

1 –> 2 –> 3 –> {4 –>3}

At this point the service provider evaluates an offer and updates its knowledge about the customer. If the offer is acceptable the Negotiation Manager creates an SLA, otherwise provider sends counter-proposal. Exchange of counter-proposals continues until one of the parties decides to accept an offer or quit. The state SUCCESS or FAIL has to be reached. The essential work in creating SLOs takes place in the business/marketing department. SLOs should aim at achieving the best performance possible, but representing true and real values at all times.

6. Implementation

In our system resource specific knowledge inclusion should eliminate many of the inefficiencies in SLA creation. By using templates and SLO libraries SLA Negotiation Manager will ease the contract creation. Our system makes the use of the widely approved contract language Web Service Level Agreement (WSLA). It also provides a user friendly interface for the client to see and choose requested services as well as enabling the exchange of counter-offers. It is anticipated that the contract creation time will be reduced significantly as a result

of the usage of templates and pre-approved clauses. By using our system the service provider will be able to ensure consistency and compliance with company's standards. Storing all SLAs in a single repository will provide an additional benefit to the service planning and management tools, so that it is required to search for a contract in only one place. In the SLA creation process, a client is presented with the services that are offered by the provider. Based on the customer's choice the Negotiation Manager aggregates and combines these choices into various SLA parameters, chooses service levels (SLO) for every SLA parameter. Every SLA has to be checked for the resource availability because it defines the agreed level of performance for a particular service. This process is also known as compliance monitoring. It has been our attempt to *teach* the SLA Negotiation Manager the business knowledge, goals, and policies of the party it belongs to. Such knowledge enables the system to choose and combine the set of SLOs that should be specified in the SLA in order to ensure compliance with the business goals.

In [7] it is shown that there are five main components of an enterprise Contract Lifecycle Management strategy:

- automated contract creation,
- secure contract negotiation,
- electronic contract repository,
- automatic upload of relevant contract data to back-end systems,
- generation of proactive management reports and alerts to encourage compliance to committed contract terms and conditions.

It is our goal to provide first four out of the above five directives in the SLA Negotiation Manager. Our system will automate contract creation through a secure negotiation with the customer, then newly created SLA will be stored in a central repository and the back-end system logs will be updated for the usage of resources that are specified in the contract. As for the last component, we leave the generation of relevant reports to the service management tools.

6.1 System dependencies

Every SLA consists of at least two signatory parties: the service provider and the customer (client). Both service provider and a client can have multiple SLAs in their internal company's repository. Each SLA can consist of multiple SLOs. There is at least one SLO for each service offered.

As an illustration of these type of situations, hereafter is a typical scenario of a retail store that needs a front end billing transactions handled.

A customer finds a service description and relative URL in the business directory (e.g. UDDI). Then it connects to the company that offers the service. Upon such connection an SLA Negotiation Manager is started. The customer wants to subscribe to a particular service (for example: store customers' billing system). The customer knows that to be successful it needs to have an access to software that can handle 10,000 transactions per day, with an initial transaction response time lower than 5 seconds and the average transaction time not longer than 60 seconds.

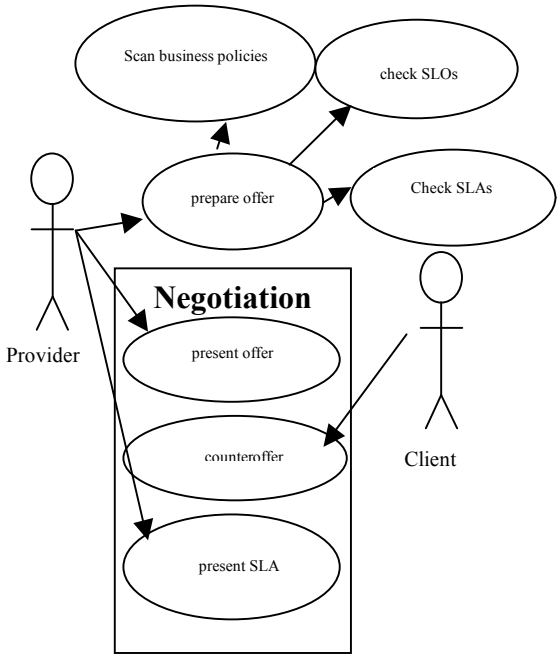

Figure 3: Use case diagram for negotiation scenario

The customer is willing to pay $800/month for such service. The SLA Negotiation Manager by examining existing SLOs and existing SLAs checks if such service is available (checking of the existing SLAs is done in order to avoid over-commitment). If the provider's company can provide a service required then a SLA is created accordingly and presented to the customer for an acceptation.

Upon customer's acceptance, the SLA is stored into the repository and the service is made available to the client. It is anticipated that at this point a SLO defining a service of renting a hardware capable of performing 10,000 transactions per day would have to be removed from a resource pool to avoid over-commitment.

This is the best case scenario. Often, the service provider can not commit to the requested service and then the SLA Negotiation Manager would come up

with the next best offer. Such decision making might be based on asking customer how much money it is willing to spend or how many transactions its store must absolutely have and based on that and on knowledge of the system the Negotiation Manager can propose a number of options to choose from. The offer can also depend on other parameters as well. Maybe the provider can commit to 10,000 transactions, but the upper limit on the average transaction time will be 90 seconds. One option might be an offer of 8,000 transactions per day with the initial response time lower than 10 seconds and an average transaction time of less than 60 seconds for $650.00/month and/or another offer could be 12,000 transactions per day with the initial response time lower than 5 seconds and the average transaction time of 3 minutes for $1,000.00/month. Ideally the customer chooses one of the offers and a SLA is created. If the customer does not agree to the proposed service then negotiation continues.

6.2 Negotiation Manager Model

An Automated Negotiation Manager model is a 7-tuple: {R, K, Z, P, Q, F, M} where:
R is a set of participants,
K is a set of all possible agreements (SLAs),
Z is a set of business rules,
P is a set of all SLOs,
Q is a set of all negotiation sequences,
F is a utility function,
M is a set of all possible offers.

1. R is a set of participants. This set contains all parties that can be involved in the contract. The customer, service provider and all supporting parties belong to this set. At least two elements of this set (service provider and customer) must participate in any SLA negotiation process $q_n \in Q$.
2. K is a set of all possible agreements (SLAs). Every existing SLA agreement that is stored in a data base belongs to the set K. It also contains all the possible agreements that can be created as a result of any successful negotiation process.
3. Z is a set of business rules (also called business knowledge). A business rule that a service can not cost less than $0.07 per transaction might be an example of $z_i \in Z$. Set Z represents corporate preferences and aligns business strategies of a service provider.
4. P is a set of all SLOs. Every SLA contains at least one SLO for the agreed service.
5. Q is a set of all sequences *s*, such that every *s* $=q_1,q_2,q_3 \ldots q_n$ where q_i is an action (an offer, a counteroffer, accept or decline). Each *s* illustrates a negotiation process and every successful negotiation is a finite sequence s. Here, by successful negotiation we mean any negotiation process that resulted in either accept or decline. Sequence s can also serve as a history log when stored in a repository. The past negotiation procedure can be recreated from such sequence.
6. F is a utility function. This function is customized according to the negotiating party needs and business preferences. For example it might be widely known that the customer offers 10% less for the service than it is really willing to pay. Function f might be used to calculate next offer: f = current offer - 10%.
7. M is a set of all possible offers. Every permutation of elements of P belongs to M. In addition M contains any combination of an offer that has been modified according to one or more business rules from set Z.

There have been many mathematical models developed for negotiations, typically on direct e-commerce negotiations, and often employing game theory algorithms [8,9]. Although these are not directly applicable to the SLA environment where there are a great deal more factors to consider above the product and price, they are useful for further development of the negotiation system.

A key factor for a Negotiation Manager is the ability to operate in an open environment where the preferences of a client are not known and we can only assume using a common knowledge that client's goal is to get more of a service for less money. This comes from the fact that customer's needs may go beyond specialized capabilities of any single service offerings. Moreover, the participating parties' legacy environments have to be incorporated seamlessly into the system. The Negotiation Manager design will follow the framework of a computational mechanism design which is an aggregation of a game theory, artificial intelligence and algorithmic theory. Mechanism design problem is to implement a system wide solution to a decentralized optimization problem with an intelligent agent representing the service provider and a customer who has private information about its preferences for different outcomes.

6.3 Negotiation Mechanism

A negotiation mechanism design is to define the possible strategies and a method used to select an outcome based on client's type and preferences. A negotiation mechanism:

$$\mathbf{M = (\sum_1, \ldots \sum_n, g(.))}$$

defines a set of strategies $\sum_i$ available to the negotiation agent, and an outcome rule:

$g:\sum_1 x \sum_2 \ldots \sum_n \rightarrow O$, such that $g(\delta)$ is the outcome implemented by mechanism for strategy profile $\delta = (\delta_1,\ldots\delta_n)$.

All of the SLA's components and SLA itself has to be translated into the machine readable format. There are several such specifications resulting from ongoing research at the large software companies such as HP, Sun Microsystems and IBM [10,11]. For our model we have chosen WSLA expressions. WSLA is based on Extensible Markup Language (XML), and it has the ability to define and describe computing services along with quality of service and service performance parameters. In addition XML is a very flexible text format that was originally designed to meet the challenges of large-scale electronic publishing, and it can be easily extended to meet one's needs. WSLA is defined as an XML schema therefore the resulting SLOs can be easily translated into system-level configuration and stored in the machine readable format to be used by various system services such as SLA Negotiation Manager. We do not discuss SLOs creation in this paper as this is research topic of its own, and the scope of this paper does not allow for an elaboration on this process. Here we assume that SLOs are developed by the Business/Marketing department and have already been defined in WSLA.

In our scenario there are two sides of the negotiations. One side, a service provider, has a repository of SLOs that define limits of the resources offered and the cost for each service, and on the other side there is a customer, who also has to define thresholds for acceptable service performance and the price that it is willing to pay.

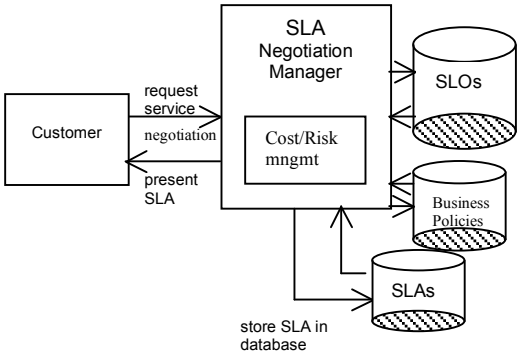

Figure 4. Process of creating an SLA

In our automated SLA Negotiation Manager the system will provide the compliance monitoring according to the customers choices. A base framework for SLA negotiation model is presented in Figure 4.

6.4 Service Process Explained

It is very common that the service providers list their service offers in some business directory such as UDDI. A potential customer can find such listing on the web and locate the service. For the clarity of this paper we will continue with our retail store customer who needs hardware and necessary network connections to provide a store front sale billing functionality. Upon the client's choice of a specific vendor (or a specific service) the SLA negotiation manager will be executed. Figure 6 shows a sequence diagram for the SLA creation scenario. Let the application SellSolution serve as an example here.

A financial institution, offers a Web service to private and corporate store owners to perform a number of different types of store transactions (such as bank account transfers, credit card payments, returns, store credit option) and generate the statements needed for tax related and bookkeeping purposes. It is a web service on demand (also called utility service) where the customers can be billed for services used. The computing resource is SellSolution that allows for billing transactions on demand. A potential customer might be a large corporation that has a variety of different types of transactions; a medium size store that uses store credit card charges; or a single private store owner who only wants to use bank account debit charges.

The billing rate might be based on number of transactions, transaction time and/or availability to the customer. In our example the SellSolution has four SLOs specified for different performance levels: platinum, gold, silver and bronze.(Shown in Table 1) Every level depends on a number of transactions being performed. The platinum level has an unlimited number of transactions, but instead is bounded by the response time and transaction time.

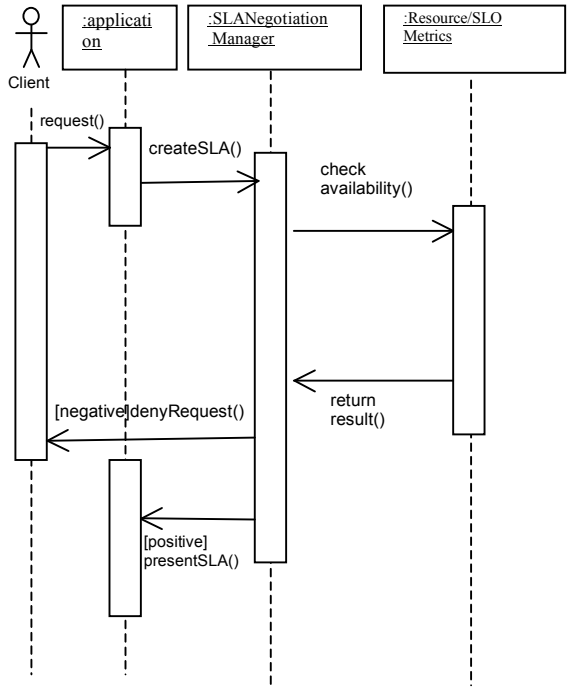

Figure 4. Sequence diagram for SLA creation process

In our model, every customer no matter how small or how large of an enterprise will be able to take advantage of an automatic SLA creation through our SLA Negotiation Manager. The resulting SLA will be based on the SLOs of the business, and created according to WSLA specifications, which in turn will make them readable for other system utilities such as performance manager or service level manager.

7. Conclusion

Even though the software has been around for decades, with passage of time, the complexity of it simply increases. The latest studies show that computing services in combination with software on demand might provide solution for an enterprise level architecture.

Our paper presents a unique approach to the creation of Service Level Agreements. In practice constructing an SLA requires planning and care. While the process can vary among companies, it is often a politically oriented topic. SLAs are known to be used to find blame instead of being a driving force towards a positive change. There is a lot more to SLA Management tools than XML schemas and standards. The combination of information and contract negotiation procedure plays an important role. The system presented in this paper will provide an automated way to create and document SLAs which in turn will increase web service provider's profits, maximize customer satisfaction, and it will open up the way to more flexible service provision.

References:

[1] Christopher Ward, Melissa J. Buco, Rong N. Chang, Laura Z. Luan, Edward So, Chunqiang Tang "Fresco: A Web Services based Framework for Configuring Extensible SLA Management Systems" Proceedings of the IEEE International Conference on Web Services (ICWS'05) 11-15 July 2005 Page(s):237 - 245 vol.1

[2] Buco M.JU., Chang R.N., Luan L.Z., Ward C., Wolf JL., Yu P.S. "Utility computing SLA management based upon business objectives" IBM Systems Journal Vol 43 No.1 2004 p.159.

[3] Suh, Bob. "Avoiding an Austerity Trap" Outlook Journal, February 2004 Retrieved from: http://www.accenture.com/Global/Research_and_Insights/By_Subject/High_Performance_Business/AvoidingtheAusterityTrap.htm on Dec 12, 2005

[4] Leopoldi, R. "IT Services Management, A Description of Service Level Agreements", White Paper, RL Consulting, 2002 Retrieved from: http://www.itsm.info/SLA%20description.pdf on June 22, 2005

[5] Sturm, Richard. "Service Level Objectives", Network Word Fusion, 2002 Enterprise Management Associates, Inc. Retrieved from: http://www.slminfo.com/articles/slobjectives.htm on Dec 12, 2005

[6] Gualtieri Andrea, Ruffolo Massimo, "An Ontology-Based Framework for Representing Organizational Knowledge", Proceedings of I-KNOW '05 Graz, Austria, June 29 - July 1, 2005

[7] Weintraub Allan, "Contract Management – A Strategic Asset" CRM Today website , Retrieved from: http://www.crm2day.com/highlights/EEplVVVFlpFCMLrUcN.php On June 22, 2005

[8] Zeng, D., and Sycara, K. "Bayesian Learning in Negotiation" Working Notes of the AAAI 1996 Stanford Spring Symposium Series on Adaptation, Co-evolution and Learning in Multiagent Systems

[9] Oprea M., "An Adaptive Negotiation Model for Agent-Based Electronic Commerce", Studies in Informatics and Control, Vol.11, No 3, September 2002

[10] Dan, A., Ludwig, H., Pacifici, G., "Web Service Differentiation With Service Level Agreements", White Paper, IBM Corporation, March 2003, Retrieved from: http://www-106.ibm.com/developerworks/library/ws-slafram/ on Feb 02, 2005

[11] Sun Microsystems, "Using the Sun ONE Application Server 7 to Enable Collaborative B2B Transactions" Informit Network Website, Retrieved from: http://www.informit.com/articles/article.asp?p=100664&seqNum=2&rl=1 on Feb 22, 2005