SLAWs: Towards a conceptual architecture for SLA enforcement

Jose Antonio Parejo, Pablo Fernandez, Antonio Ruiz-Cortés, José María García
University of Seville
Sevilla, Spain
Emails: {japarejo, pablofm, aruiz, josemgarcia}@us.es

Abstract

Current technologies in Service Oriented Computing (SOC) provide a solid framework to drive the interaction of organizations from a functional point of view. In order to introduce non-functional properties in this scenario, the concept of Service Level Agreement (SLAs) comes into play as a key element. SLAs can be seen as containers of the functional and non-functional properties that both parties (the service consumer and the service provider) agree specifying its rights and obligations during the interaction. However this SLAs represent an additional responsibility for the service provider since it motivates the need of a SLA-Enforcement process in its infrastructure. A proper SLA-enforcement implies optimized resource usage that meet SLAs established with consumer, making it possible to the provider afford a larger number of customers to maximize its benefits.

Current approaches to SLA enforcement are domain-specific approaches and/or based on monolithic platforms (from a SOC point of view).In this paper we propose a conceptual architecture (SLAWs) for SLA enforcement. The main goal of the proposed architecture is to be used as a conceptual framework to build a flexible SLA enforcement layer. This layer could be integrated in a seamless way in the pre-existing provider infrastructure when is based upon the service oriented architecture principles.

1. Introduction

In the last years Service Oriented Computing (SOC) [16]has evolved as a prominent paradigm in system integration. The set of technologies (such as WSDL or SOAP) developed around this paradigm is a solid support for the integration from a functional point of view but the non-functional properties are still a challenge. Typically, in real industrial scenarios, the relationship between organizations is guided by a contractual context where rights and obligations of each party are stated. However, since most of the integrations correspond to intraorganizational nature (EAI), non-functional properties has currently been marginally addressed by real scenarios in industry. On the contrary, the promising interorganizational (B2B) integration scenario should be a motivating horizon to fully address non-functional features; normally, these properties are a explicit or implicit part of the contractual context.
The minimal scenario of interorganizational integration in SOC involves two organizations: the Service Consumer and the Service Provider. In order to materialize the contractual context amongst parties, the concept of Service Level Agreement appear as a explicit element to specify the functional and non-functional properties that are guaranteed by each party.

The usage of SLA is, in principle, attractive for both consumer and provider: on the one hand, the consumer would get guarantees and reliability from the provider; on the other hand, provider could achieve a desired fidelity and reputation by meeting the SLA expectations with an appropriate SLA enforcement. The enforcement process refers to the constraining mechanisms to fulfill (and to force a fulfillment of) the terms that compose the SLA for each party (i.e. consumer and provider); from an operational point of view, it implies a management of the SLA object (i.e. a mean to specify and obtain the SLA). As a consequence, on the one hand from customerѕ perspective, enforcement implies monitoring of properties associated with the terms stated on the SLA, and validating their values according to those terms. On the other hand, from providers perspective, enforcement process implies, as well, monitoring and validation of service invocations by customer, configuring also the infrastructure to provide an appropriate service. This last element is a key point of success in terms of business goals. In this context, the automation of the enforcement process, would bring important benefits for the provider since the infrastructure could act autonomously to optimize the usage of its resources without a violation of the stated SLAs.

Currently, some important steps have been taken towards to obtain and express the SLA: In [1], an analysis of the gap between legal contracts and operational SLAs is discussed.

A language to specify and manage machine-processable SLAs is proposed in [15]. [7] outlines the architectural foundation for building platforms to create and negotiate SLAs in an automated way. A method to determine the mismatch between two QoS offers (SLAs) is proposed in [18], and in [14] temporal concerns are introduced to complete the QoS specification and offer evaluation. Concerning the SLA enforcement itself, proposals can be divided in two sets: (i) Specific application-domain approaches (such as [10]) ; these works are only intended for a particular business (e. g. from automotive parts production and design to computation power trading) that are usually based on monolithic platforms (from a SOC point of view) [5][4]. Additionally, this application-domain focus, usually means a high adaption cost to other industrial scenarios (ii) A manual approach to the configuration of the infrastructure performed by an human actor [20]. These approaches to the SLA enforcement bring important drawbacks: On the one hand, to create an SLA enforcing infrastructure for a new domain is a cumbersome task since there is a lack of a general SLA enforcement architecture. Additionally, it represents the dependence of a usually intrusive platform for the service oriented architecture of providers. On the other hand, a manual approach does not take advantage of an automated optimization of resources when having a machine processable SLA.

In this paper, a conceptual architecture (SLAWs) for automated SLA enforcement is presented. This architecture, is designed upon the principles of SOC outlining the elements and its relationships as services and taking into consideration the main standard of interface definition (WSDL [21]); in particular, it is used as the key element to wrap the services to perform the functional logic and the control of non functional properties. The main goal of the proposed architecture is to provide a conceptual framework to build a flexible and automated SLA enforcement layer. In so doing, this layer could be seamlessly integrated in a pre-existing provider infrastructure when it corresponds to a service oriented architecture.

In the following section, a discussion of the SLA enforcement is presented: First, a conceptual background is provided to identify the key element of the enforcement in the SLA, then an example scenario is outlined; later on the section, a comparison for the example scenario is discussed between a typical aggregated enforcement and the proposed SLAWs-based enforcement. In section 3, an analysis of related work is presented. Finally, on section 4, a concluding summary and future work are discussed.

2. SLA Enforcement in SOC

Current proposals to express SLAs in SOC [15] [6], provide a way to specify the set of functional and non functional properties that must be guaranteed during the service transaction. In these SLA languages, functional properties are expressed using WSDL (i. e. they specify the interface of invocation of services) while there is a lack of standard language for expressing non-functional properties. Several research efforts have been made towards this goal (such as [17]) but currently ad-hoc languages [8] (with sufficient vocabulary for the concrete domain) remain as the most used solution to express non-functional properties. In particular, according to [12], two kinds of non functional properties can be identified:

- **Controllable properties**: Providers can control the value that properties present, performing actions to change it (e. g. the length of the key used in the encryption algorithm specified in the SLA for a secure transaction). It is important to point out that the value of this property can be continuous (not discrete), or dynamic but in the context of the transaction (with a valid SLA) the provider guarantees a concrete bounded value.

- **Non-controllable properties**: Providers can not modify the property value (e. g. the nationality of the provider). These properties are part of the very own nature of the provider but are expressed in the SLA since they are important for the consumer (e.g. a consumer only looks for providers in its country).

According to our vision, a proper management of controllable properties is key point from the SLA-enforcement point of view: First, it implies control capability of SLA violations. Secondly, an intelligent management of controllable properties means an optimized resource usage that meet SLAs established with consumer. This optimization process allows the provider to afford a larger number of customers maximizing its benefits. In the context of a service provider implementing a Service Oriented Architecture (SOA), two kinds of services are required to provide an adequate SLA enforcement:

- Services that implement functional properties. It usually correspond to a specific implementation of the functionality requested by the consumer.

- Services used to configure the provider infrastructure to give support to the controllability of properties. If those facilities are not provided natively as services by their infrastructure, usually an Enterprise Service Bus (ESB), provider should be able to encapsulate it as new services.

In order to illustrate the key elements for SLA enforcement, we propose an example scenario (depicted in figure 1), inspired in [15]. In our scenario computing services are

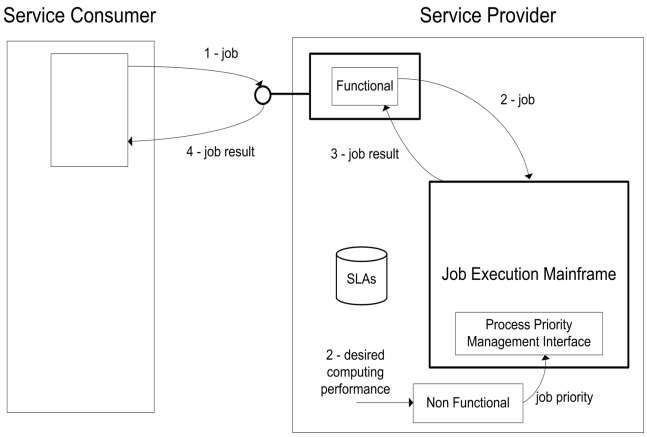

Figure 1. Example Scenario

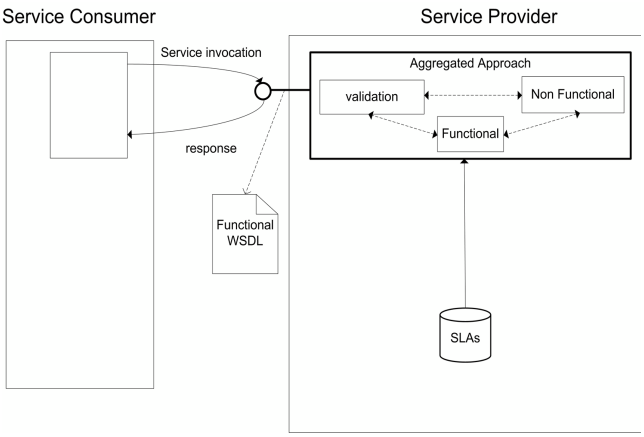

Figure 2. Aggregated Approach

provided according to previously established SLAs. This scenario could be useful in research fields that require intensive computation, like bio-informatics and complex systems simulation. A service accepts jobs that are submitted to a mainframe for their execution, and returns result obtained to customers. There is a registry that provides access to the SLAs agreed with consumers where are expressed the requirements and obligations for completing a job, concretely in this scenario, provider guarantee a specific computing performance (e.g. expressed in floating point operations per second (Flops)).

The functional part is represented by a service that launches the process in the mainframe job execution system. The non functional service logic is a component that using a model of the computing performance, it can identify and set the right values of process priority using the mainframe process priority management interface. In the following sections we will describe how providers could implement SLA enforcement for this scenario using two different approaches, an aggregated solution, and a generic solution using our proposed conceptual architecture (SLAWs based enforcement).

2.1. Aggregated Enforcement

The aggregated approach in a SOC context is characterized as a single web service representing both the SLA enforcement logic and the functional part. It is important to highlight that the majority of platforms with this model usually have a modular structure but, from a SOC perspective an unique service is published. In this context, the internal components are not exposed as services, and only the functional service is deployed.

Figure 2 shows this general scenario. Main elements of this approach are: (i) the WSDL port exposing the functional interface. (ii) The implementation of the functional logic. (iii) the SLA validation logic (iv) the non functional element that can actuate over the platform to configuring its parameters; its goal is to modify the functional service deployment to guarantee the non-functional controllable properties expressed in the SLA.

In this scenario consumer access the service through its functional interface (as expressed on the WSDL). This enforcement process have three stages: First, a location of the SLA terms (usually in a SLA registry); then, a validation of the service invocation with the terms stated on the SLA and, finally, the invocation of the functional logic joint with the control logic to guarantee the controllable properties.

The main drawback of this approach is the strong coupling amongst the different elements. In particular the SLAs structure joint with the validation, configuration and functional logic. In this way a change on any of those components usually implies the need of modification on others. Moreover, most of the approaches with this aggregated structure, are exposed as a black box that is difficult to modify and integrate in a pre-existing service oriented architecture.

Figure 3 shows an aggregated enforcement for the job submission example proposed. The sequence of interactions would be as following:

1. Consumer invokes the service.
2. Enforcer reads SLA Data.
3. Job is submitted to the mainframe (using functional logic block).
4. The priority of the process is configured (using non-functional logic block).
5. Mainframe returns job result to enforcer.

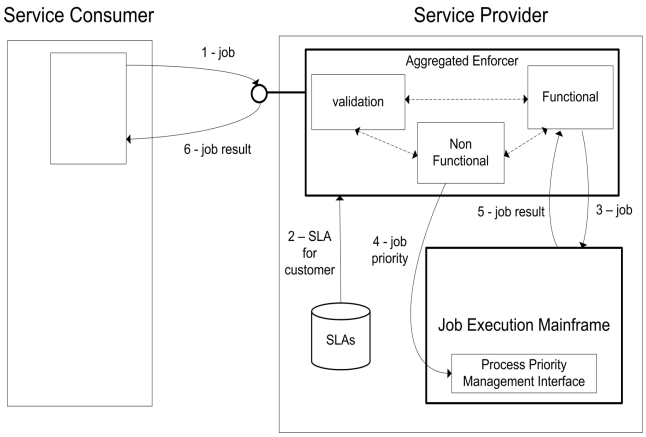

Figure 3. Aggregated Approach applied to Job Execution Scenario

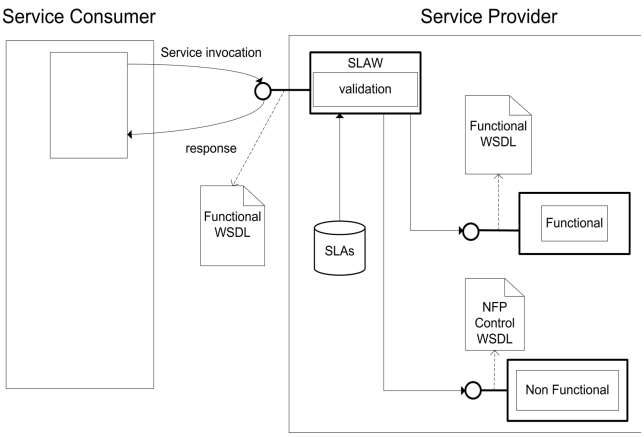

Figure 4. SLAW Approach

6. Consumer receives the results obtained.

2.2. SLAW-Based Enforcement

In this approach we promote the idea of wrapping the validation logic in a new service called the Service Level Agreement Wrapper (SLAW) and force a SOA structure in the remaining elements. The key principles in a SLAW-Based enforcement are threefold: First, the SLAW is exposed to the consumer request. Second, the functional implementation service remains intact hidden to the consumer as a back-end isolated service. Third, the control logic to actuate over the non functional properties is also wrapped and exposed as an internal service. Figure 4 shows the conceptual architecture of the SLAW approach in a general scenario.

Concerning the design, main idea behind the approach is the usage of the proxy and adapter design patterns [9] following the SOC principles: on the one hand, from a consumer point of view, a proxy behavior is performed having the same WSDL both functional and SLAW service. On the other hand, from a provider point of view, an adaption (or wrapping) is made to perform the SLA-enforcement based on two services providing the functional and the non-functional logic.

With the SLAW-Based enforcement a decoupling based in the idea of SOA is promoted. In this approach, the functional and non-functional logic are decoupled from an infrastructure point of view. In doing so, the adaption of additional non-functional logic can be done by new services deployed over the SOA infrastructure.

Moreover, this approach can be less intrusive in pre-existing SOA environments (without SLA-Enforcement) in its transition to a SLA-driven scenario. In this context, the pre-existing functional services remain intact and SLAWs can be created only in areas where an SLA is desirable; e.g. keep EAI interactions without SLA while enforce the B2B transactions with SLA.

In figure 5 we present an application of the SLAW-Based enforcement architecture to the job-execution example presented.

The sequence of interactions in figure 5 are as following:

1. Customer invokes the service through the SLAW.
2. The SLAW reads the data from the SLA and validates the invocation.
3. If invocation is invalid the SLAW returns a fault, else it determines the control service to apply to this service invocation based on the SLA.
4. The SLAW invokes the functional service that, submits the Job to the mainframe
5. The SLAW invokes the non-functional service that implements the enforcement logic (providing the agreed computing performance) so it can change the process priority if necessary.
6. Mainframe returns job result to enforcer.
7. Consumer receives the results obtained.

3. Related Work

In the last years, several research efforts try to combine SLA and SOC research lines. In general, those related work approaches can be grouped in two main sets: SLA creation

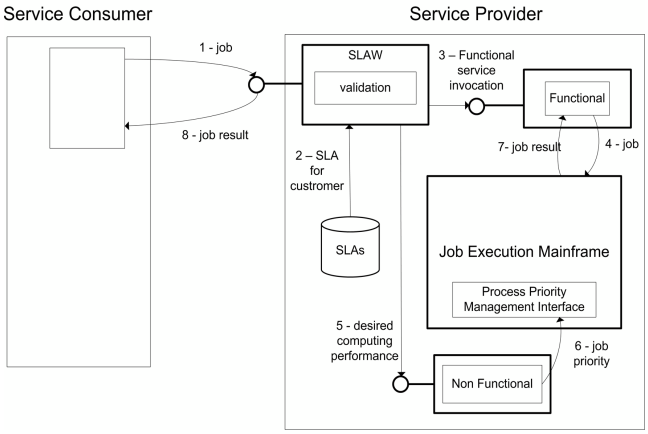

Figure 5. SLAW Approach applied to the Job Execution Scenario

and SLA enforcement. On the one hand, the procurement process is centered in the way the SLA is obtained: how it is expressed, negotiated, stored or exchanged are challenges involved in this research field. On the other hand, once the SLA is available, a mechanism to enforce the terms of it should be established; in this context, important issues to be addressed would be: how services can parametrize its behavior to meet the SLA or how violations over the SLA are detected. Concerning the SLA procurement, some important advances have been made; in particular, after a long period of incubation WS-Agreement specification has recently be promoted as a final recommendation by the OGF [15] as a logical extension to the basic SOA Stack to introduce the idea of SLA. More concretely, WS-Agreement provide a basic framework to express the SLA structure and provide a minimum operational support for the procurement. However, in order to deal with an specific scenario, this specification already (and intentionally) require a high degree of refinement in term of DSL (Domain Specific Languages) for the specific scenario (such as [8]); additionally, to address non-basic scenarios, some important extensions must be done [14]. To this end, some efforts have been made, in particular, authors propose FAST [7]; this architecture propose a general framework to automate the trading of agreements: i.e. the process of reaching an agreement amongst different parties in complex-scenarios by creating a structured infrastructure to extend and reuse DSLs, standards and patterns in a coherent way.

SLA enforcement has been a prominent field of research in the last years: from a theoretical standing point, in [13] authors propose a protocol for contract monitoring and enforcement based on protocol model checking; in this context, a more general approach of behavioral model in cross-enterprise interaction is proposed in [11]. Complementary, in order to deal with real scenarios, some infrastructures have been proposed; this infrastructures can be decomposed in two main groups depending on the side they are concern with: customer or provider side. On the one hand, customer side infrastructures are focused on the way the consumer of service deal with a pre-established SLA (or some QoS properties): in [19] some service tooling and systems are presented to create transactions where SLA is monitored and negotiated by service consumers to detect violations and possible switchings amongst providers; in a similar approach [2] and [3] projects develops a GRID-oriented infrastructure where composed process can be reconfigured to assure a certain SLA. On the other hand, provider-side proposals deal with the infrastructure to make an SLA-aware enforcement of services; following in this section, we analyze a set of four relevant proposals and compare it with our approach in table 1.

In this study, we expose a comparative analysis of the different SLA-provision and SLA-enforcement approaches with our proposal according to the following properties:

- Automated Enforcement. This property refers whether the SLA enforcement of the service is performed in an automated way or base on a human actor.
- NFP Enforcement mechanism. This property relates to the way that is used to enforce the NFP agreed in the SLA. In some approaches, there is a application-specific system, in other approaches, there is an external element that is responsible of enforcing the NFP.
- SLA Vocabulary. This property corresponds to the nature of the information expressed in the SLA, whether it is an open vocabulary that can be extended, or a fixed set of primitives that are available.
- Application Domain adaption. In order to apply each of the proposals to different scenarios, some work must to be done. In this property we analyze the possibilities of application domain adaption; in this context, a way to extend the core vocabulary and enforcing elements should be provided.
- Enforcement Policies. This property establishes if the enforcement mechanism can be parameterized to allow a degree of variability; in so doing, the enforcement could be open, or fixed in terms of enforcement behavior.
- Layered Adaptation. In this case, the property indicates if the architecture of the proposal make it possible to create a set of different abstraction layers to create a multi-stage enforcement mechanisms.

	Autom. Enforc.	NFP Enf. Mechanism	SLA Vocabulary	Domain Ext.	Enf. Policies	Layered Adaptation
SAM	full automated	ad-hoc	unk.	unk.	open	multi-layer
FRESCO	Based on DSL (SCOL)	external	open	SLA Vocabulary and ad-hoc	open	single
GRIA	full automated	ad hoc	fixed	API	fixed	single
WSMX	none	external	n/a	ad-hoc	none	n/a
Our proposal (SLAWs)	Proxy base implement.	external	open	Through SLA Vocabulary	open	multi-layer

Table 1. Feature comparison

In SAM (SLA Action manager) [4], authors propose a generic SLA management framework and an integrated set of advanced service level management technologies; in particular non functional properties can be managed manually, or using automated or semiautomated processes. The result of executing the service-quality management task is usually a modification of the overall configuration for the managed utilities and processes. Each SLA is linked with a SMO (SLA Management Object), that essentially transforms the associated SLA contract into an active computing entity in the environment that performs evaluations according to the business logic, raise quality alerts and associates SLM (Service level management) processes to the alerts. An interesting aspect of this approach is a Cross-SLA Quality Alert Manager that gathers quality alert data form all of the SMOs, and maintains one or more ordered list of quality alerts in terms of business impact. Those lists can be distributed to other service management agents and personnel; in addition, they provide an optimizing scheduler for SLM process execution that can be customized based on these alerts. Finally, it is important to highlight that this approach outlines a duality of concerns between humans/automatic processes for validation/management/actuation of Services and therefore, in some scenarios, it is not clear the degree of automation of certain tasks.

The Fresco approach [20], describes a framework to create SLA management systems especially focused towards the SOC paradigm. In particular, this proposal elaborates the idea of external systems to monitor and enforce SLAs, called ”Services Monitoring System (SMS)” and ”SLA Management System (SLAMS)” respectively; however only allows to configure and state the mapping between data obtained from the SMS to the data expressed in the SLA, and configure the actions triggered in the SLAMS. In this context, it is important to highlight that Fresco framework delegates the functional part of the service to be fulfilled by the underlying implementation services layer.

GRIA System [10](Grid Resources for Industrial Applications) establishes an integral infrastructure for SLA management focused on GRID environments. More concretely, this proposal establishes a set of building blocks to deploy specific services according to a particular SLA. The available infrastructure is oriented towards a basic set of application services of data processing and job distribution. In this context, though GRIA approach has a very mature infrastructure for dealing with real scenarios in terms of security or reliability issues, it is highly domain oriented for the initial package of services provided; in doing so, it lacks of extensibility specially in terms of SLA properties to be observed and dynamic changes of SLA enforcing mechanisms.

WSMX [22](Web Service Modeling eXecution environment) is part of a wide initiative for dealing with semantic web services. This infrastructure gives support for the execution of semantically-annotated web services taking advantage of the semantic technologies to perform complex reasoning in operations such as matchmaking or dynamic service composition.

4. Conclusions

In this paper we analyze the Service Level Agreement (SLAs) enforcement problem in providers with a Service-oriented Architecture.

During this analysis we have identified the ”controllability” of Non-functional properties as a key point for this problem since it allows a better resource usage management. A conceptual architecture for SLA enforcement around the idea of SLA Wrappers (SLAWs) has been presented. SLAWs services are designed to encapsulate the SLA validation logic and act as dispatchers to the appropriate functional and non-functional control services. The SLAW-Based enforcement boost the decoupling of functional implementation services and non functional properties using a Service Oriented Architecture (SOA) approach;

as a consequence, a flexible applicability is described.

Future work includes the development of a prototype deployable on an Enterprise Service Bus (ESB) in a non intrusive way. The use Java Business Integration (JBI) standard for an automated deployment and component model is currently being studied. Additionally, the application to other industrial scenarios will be developed to validate the current approach.

Acknowledgment

This work has been partially supported by the European Commission (FEDER) and Spanish Government under CICYT project Web-Factories (TIN2006-00472) 00472), and Andalusian Government project ISABEL (TIC-2533).

References

[1] A. Arenas, M. Wilson, S. Crompton, D. Cojocarasu, T. Mahler, and L. Schubert. Bridging the gap between legal and technical contracts. *Internet Computing, IEEE*, 12(2):13–19, March-April 2008.

[2] ASG. Adaptive services grid. Integrated Project supported by the European Commission. Web site: http://asg-platform.org.

[3] AssessGrid. Advanced risk assessment & management for trustable grids. Project funded by the EU. Web site: http://www.assessgrid.eu.

[4] M. J. Buco, R. N. Chang, L. Z. Luan, C. Ward, J. L. Wolf, and P. S. Yu. Utility computing sla management based upon business objectives. *IBM Syst. J.*, 43(1):159–178, 2004.

[5] C. Dumitrescu and I. Foster. GRUBER: A Grid Resource Usage SLA BrokER. *Proc. of 11th Intl. Euro-Par Conference, Portugal*, 2005.

[6] ebXML. ebxml. www.ebxml.org.

[7] P. Fernandez, M. Resinas, and R. Corchuelo. A conceptual framework for automated service trading. In J. C. Riquelme and P. Botella, editors, *Proceedings of JISBD06*, volume 1, pages 273–282. Jornadas de Ingeniera del Software y Bases de Datos 2006, 2006.

[8] G. G. Forum. Job submission description language (jsdl) specification. http://www.gridforum.org/documents/GFD.56.pdf, December 2005.

[9] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns*. Addison-Wesley Professional, January 1995.

[10] GRIA. Service oriented collaborations for industry and commerce. Project funded by EC, December. Web site: http://www.gria.org/.

[11] P. F. Linington, Z. Milosevic, J. Cole, S. Gibson, S. Kulkarni, and S. Neal. A unified behavioural model and a contract language for extended enterprise. *Data and Knowledge Engineering*, 51(1):5–29, October 2004.

[12] O. Martín-Díaz. *Automatic matching of web services using constraing programming*. PhD thesis, Dept. of Cumputer Languages and Systems. University of Seville, 2007.

[13] C. Molina-Jimenez, S. Shrivastava, and J. Warne. A method for specifying contract mediated interactions. In *EDOC '05: Proceedings of the Ninth IEEE International EDOC Enterprise Computing Conference (EDOC'05)*, pages 106–118, Washington, DC, USA, 2005. IEEE Computer Society.

[14] C. Müller, O. Martín-Díaz, A. R. Cortés, M. Resinas, and P. Fernandez. Improving temporal-awareness of ws-agreement. In *ICSOC*, pages 193–206, 2007.

[15] OGF. Grid resource allocation agreement protocol wg (graap-wg): Web services agreement specification (ws-agreement) (v. gfd.107) (2007).

[16] M. Papazoglou. Service-oriented computing: concepts, characteristics and directions. *Web Information Systems Engineering, 2003. WISE 2003. Proceedings of the Fourth International Conference on*, pages 3–12, 10-12 Dec. 2003.

[17] A. Ruiz Cortés. *A semi-cualitative approach to the automated management of Quality Requirements*. PhD thesis, Dept. of Cumputer Languages and Systems. University of Seville, 2002.

[18] A. Ruiz-Cortés, O. Martín-Díaz, A. Durán Toro, and M. Toro. Improving the automatic procurement of web services using constraint programming. *Int. J. Cooperative Inf. Syst.*, 14(4):439–468, 2005.

[19] SeCSE. Service centric systems engineering. 17 partners from 8 countries, September. Web site: http://secse.eng.it.

[20] C. Ward, M. Buco, R. Chang, L. Luan, and E. So. Fresco: a Web services based framework for configuring extensible SLA management systems. *ICWS 2005. Proceedings. 2005 IEEE International Conference on Web Services*, pages 237–245, 2005.

[21] WSDL. Web services description language (wsdl) 1.1. W3C Starndard., March. Document available at: http://www.w3.org/TR/wsdl.

[22] WSMX. Web service modelling execution environment. Open source reference implementation of WSMO (Web Service Modelling Ontology). Web site: http://www.wsmx.org/.