# An Analysis of Service Trading Architectures

Manuel Resinas, Pablo Fernandez, and Rafael Corchuelo

ETS Ingenieria Informatica
Universidad de Sevilla, Spain
http://www.tdg-seville.info

**Abstract.** Automating the creation and management of SLAs in electronic commerce scenarios brings many advantages, such as increasing the speed in the contracting process or allowing providers to deploy an automated provision of services based on those SLAs. We focus on the service trading process, which is the process of locating, selecting, negotiating, and creating SLAs. This process can be applied to a variety of scenarios and, hence, their requirements are also very different. Despite some service trading architectures have been proposed, currently there is no analysis about which one fits better in each scenario. In this paper, we define a set of properties for abstract service trading architectures based on an analysis of several practical scenarios. Then, we use it to analyse and compare the most relevant abstract architectures for service trading. In so doing, the main contribution of this article is a first approach to settle the basis for a qualitative selection of the best architecture for similar trading scenarios.

## 1 Introduction

Service level agreements (SLAs) are used by many different service industries to grant guarantees about how a service will be provided or consumed by establishing both functional and non-functional requirements that must be fulfilled by both parties during the service development. The application of Internet-based technologies for electronic commerce to establish and manage these SLAs offers significant advantages to the traditional use of SLAs [1]. Specifically, automating the creation and management of SLAs, so that the human participation in the process is reduced to the minimum, brings benefits such as cutting down the cost of reaching an agreement, increasing the speed in the contracting process and allowing providers to deploy an automated provision of services based on the SLAs agreed with their customers [2].

We define *service trading process* as the process of locating, selecting, negotiating, and creating SLAs. Therefore, the service trading process is a subprocess that covers the information and negotiation phases of the more general contracting process [3]. The characteristics of the service trading process depend on the particular scenario where it is developed. These scenarios are very diverse and can range from a traditional supply chain to dynamically selecting the best VoIP (Voice over IP) provider or contracting or renegotiating a contract with an ISP

(see Section 2 for more information). As the scenarios are very different, the requirements for each of them are also diverse. Therefore, we argue that there is no one unique solution for service trading but we must choose the most appropriate option for each situation.

We focus on abstract architectures for service trading, which are specifications that define a set of elements for service trading. These abstract architectures can be later implemented by using different technologies and applied for different problem domains. Our goal in this paper is to define a set of properties for these abstract service trading architectures based on an analysis of several service trading scenarios. These properties enable the analysis and comparison of those abstract architectures, which is a necessary step to select the most appropriate architecture in each scenario.

The paper is structured as follows. First, in Section 2, we present four service trading scenarios that serve as the basis for the set of properties for abstract service trading architectures presented in Section 3. In Section 4, these properties are used to analyse and compare the most relevant abstract architectures for service trading. Finally, we conclude in Section 5.

## 2 Scenarios

In this section, we describe a set of scenarios that correspond to different cases of service trading that have been selected based on the diversity of features, stakeholders and domains.

***Scenario 1. Service consumer looking for ISPs.*** This case relate to a mid-large size company that looks for an ISP (Internet Service Provider). In this scenario, a company publishes its demands and wait for the ISPs to make their offers. In so doing, the company has a passive role while the providers act as active organisations searching for customers. In this domain and from the point of view of the company, it is appealing to have a periodic renegotiation of the service. Furthermore, a high level of automation in the service trading enables that every renegotiation is open to different ISPs in order to select the best possible in each case; in this way, it is boosted a dynamic market where each provider look forward competitive offers adjusted at their capabilities in each moment. An additional issue is the strict temporal sequencing of the service trading process. The trading process should coherently encompass the stages to fit the temporal constraints for the company to avoid problems such as a lack of the service due to the change of ISP.

In this scenario, the QoS terms during the SLA establishment are a key factor. In this way, an interesting feature is to be able to automatically negotiate such features. Concerning the decision-making process, the information known about providers is the most important element; i.e. the reputation of provider or the historic knowledge based on previous trading process. Lastly, participant organisations should have the guarantee that the agreements reached by the system are legitimate.

***Scenario 2. Computing services provider.*** In this case, a company offers computing services to other organisations. In particular, this case is becoming increasingly popular in research fields with intensive computational needs such as bioinformatics. In a concrete manner, the company in this scenario can be described as a computing service provider that receives demands from other organisations in terms of computing jobs to be developed.

In this scenario it should be allowed for the company to specify offers that optimise the usage of its resources. Specifically, in a computing company, unused (or low used) resources means a decrease in the recovery of the initial investment. In so doing, offers should vary based on the resource usage and the set of SLAs the company has reached with its customers. Closely related with those ideas, from the perspective of the customer, a negotiation of the terms of the SLA is an interesting issue to be addressed. Moreover, this negotiation process can be used by the provider to slightly adapt the final SLA and make concessions or restrictions in order to optimise the current usage level of its resources.

Additionally, as those offers are tightly adjusted to the resource status in each moment, the decision making infrastructure should also take into account this information as a first level element before establishing new commitments with a customer. Finally, it is interesting to remark that, unlike the previous case, the reputation information is not an important issue from the perspective of the computing provider.

***Scenario 3. Company delegating to a trader specialized in VoIP.*** This case describes a company that delegates its telephony needs to a trader that handle its requests and locate the best possible VoIP(Telephony through Internet) provider for each call. This trader represents an organisation that makes profit acting as a facilitator between end-user companies and VoIP providers. These providers offers different services characteristics (e.g. guaranteed bandwidth) or restrictions (e.g. Some of them could only operate between certain countries). In so doing, this trader offers a service of calls management by creating concrete agreements for each call (or set of calls) with the telephony provider taking into account the preferences of the company: e.g. minimising cost.

The information about the telephony service providers can be divided into two sets: First, a set of information describing the capabilities of the provider in general terms such as the operational countries or time slots classification (*Peak hours*, *reduced cost hours*, etc.); this set can be used to create a selection of potential providers. Second, in each moment, when handling an specific call, the trader can ask about *last minute* offers from the providers; this offers would be based on the resource status of provider (as in the previous scenario). In so doing, based on the information harvested, the trader can construct the specific SLA proposal the most appealing provider and, finally, if it agrees, the final SLA is established and the call can be carried out.

***Scenario 4. A generic service trader in a supply-chain.*** One of the scenarios where service trading fits better is supply-chains, where each organisation create added value by composing services from different providers. In this

case, a trader of services represents organisations that create high-level services based on the composition of lower-level services. Many examples can be found in the literature from telecommunications domain [4] to the transport domain [5]. This idea of supply-chain can be isolated from a specific domain and, hence, the elements and requirements expressed in this scenario are mostly valid for the majority of supply-chains independently of the nature of the services supplied.

In this scenario, the key point to be addressed is an efficient composition of services that adds value to the customer whose the trader sells its services. To achieve its goals, an aspect to be addressed is the adaptability to different markets. To a service trader the ability to understand different ontologies is important because it allows him to communicate with different markets or providers. Closely related with the previous ideas, information harvested about different providers is necessary for an efficient decision making process of selection of services. For each potential service provider, the trader should ask to several sources: the provider itself for information about the capabilities it claims to have, and third parties that can also supply important information about the reputation of a certain provider.

In order to construct a composed service, the trader should agree several SLAs with providers for each of the services that will be composed. In this way, during the establishment of every SLA, three processes are relevant: (i) a classification of the proposals (coming from the providers); (ii) a selection of the most promising proposals; (iii) a decision about the handling process for each of the selected proposals: e.g. whether we negotiate them or not.

## 3 Properties of Service Trading Architectures

In this section, we present a set of properties describing features of abstract architectures for service trading. These properties are derived from the four scenarios described in Section 2. For each property a reference to the related scenario is supplied; concretely, this references are based in the special importance of the property for the specified scenario.

1. *External discovery (S.1, S.2, S.3 and S.4)*: We say an abstract architecture has an external discovery process if it uses an external infrastructure (e.g. an external registry) to obtain the list of potential parties that demand (or supply) a service that other party provides (or needs). Alternatively, the process is internal if no external infrastructure is used, for example, if the list of potential parties is directly provided by the user.
2. *Knowledge adaptation (S.4)*: An abstract architecture has knowledge adaptation [6] [7] if it provides elements to adapt the local knowledge model to the appropriate discovery infrastructure, making independent the characteristics of the market modelled by the discovery service to the rest of architecture.
3. *Market observation(S.3 and S.4)*: An abstract architecture with market observation monitors the changes in the market through observation of the information provided by external discovery infrastructures and informs about these changes to the elements of the architecture.

4. *Symmetric architecture for providers and consumers (S.4)*: An abstract architecture is symmetric if both service provider and consumer can start the service trading process and there is no commitment as to which party advertises and which party queries to the discovery service. Alternatively, an abstract architecture is asymmetric if only one consumer or provider can start the service trading process.
5. *Information query (S.3 and S.4)*: An information query is an inquiry made by one party to another to obtain more detailed information about it or about the service it provides or demands. Therefore, for an abstract architecture to support information queries, it must have mechanisms to query services or to respond to those queries.
6. *World model (S.1, S.3 and S.4)*: An abstract architecture builds a world model if it analyses previous interactions with the elements external to the architecture, such as other parties or the discovery services, and uses the results to make better decisions [8] during the service trading process.
7. *Third party information (S.1, S.3 and S.4)*: This property represent that a third party is explicitly queried to obtain information related to another. For instance, to obtain information about its reputation or its geographical location. In this case, a protocol to carry out this query as well as a shared taxonomy of terms must be supported by the architecture.
8. *Information managed about the parties (S.1, S.2, S.3 and S.4)*: There are three types of information that can be managed about the parties: *service information*, that is, information about the functional and non-functional characteristics of the service; *trading information* or information about the features of the trading process followed by the party; and *party information*, i.e. information about the party that provides or demands a service, such as its reputation or its geographical situation.
9. *Proposals preselection (S.3 and S.4)*: An abstract architecture has a proposals preselection process if, before starting an agreement creation process or a negotiation, it ranks and/or filters the proposals that it has received or built based on criteria previously specified.
10. *Agreement creation mechanisms (S.1, S.2 and S.4)*: An abstract architecture has multiple agreement creation mechanisms if it supports different protocols to reach to an agreement. These mechanisms can range from a take-it-or-leave-it protocol [9] to a bilateral negotiation or an auction protocol [8].
11. *Notary (S.1 and S.4)*: An abstract architecture has this property if it provides any mechanism to guarantee that the agreement created between the two parties is reliable and non-repudiable. We say an agreement is reliable if both parties are signing and accepting the same previously agreed document.
12. *Decommitment from previously established agreements (S.1 and S.4)*: An abstract architecture supports the decommitment [10] from previously established agreements if it can revoke previous agreements before the execution of the service, possibly by paying some compensation to the other party. This implies the implementation of any decommit protocol and the mechanisms to decide when a decommit is profitable for it.

13. *Capacity estimator (S.2 and S.3)*: An abstract architecture may make use of a capacity estimator to determine whether the provider can provision a certain agreement enabling a finer control about its resources and the implications of the agreements created [2].
14. *Trading protocols (S.1, S.2, S.3 and S.4)*: A trading protocol is a set of stages (e.g. *advertisement, proposal submission, negotiation, resolution, etcetera.*) cross-linked in accordance to some temporal constraints and bounded to some choreographies. The temporal restrictions specify a set of constraints about the life-cycle of the trading process. These restrictions can vary from simple fixed temporal points (e.g. *End by 14:00 of 14th, March*) to complex relationships amongst the durations of some stages (e.g. *Information stage starts in the middle of the discovery stage*). Therefore, an abstract architecture that supports different trading protocols must be able to deal with different temporal constraints on the stages of the service trading process.
15. *Creation of agreements for composed services (S.4)*: A composed service [11] is a service whose implementation is based on the execution of other services that may be provided by external entities and, hence, there may exist agreements regulating that execution. The support for creating agreements for composed services can vary significantly, from simple dependencies between the services such as "*I want either an agreement on all different services or no agreement at all*" to taking into account the service level properties desired for the composed service.
16. *Cooperative or non-cooperative agreement creation (S.1, S.2, S.3 and S.4)*: An abstract architecture supports non-cooperative agreement creation when it acts as a self-interested party reaching agreements with other self-interested parties. Alternatively, an abstract architecture supports cooperative agreement creation when it can reach agreements with other parties trying to maximise only the social welfare.
17. *Consumer or provider orientation (S.1, S.3 and S.4)*: An abstract architecture is consumer-oriented if it carefully describes the behaviour of the consumer (or the party acting on his behalf) in the service trading process. Alternatively, it is provider-oriented if it carefully describes the behaviour of the provider (or the party acting on his behalf). Note that an abstract architecture may be both consumer and provider-oriented.
18. *Deployment options*: An abstract architecture may present several deployment options depending on their characteristics. Some examples of deployment are to integrate the architecture in the service provider or to implement an independent trader of services offering its trading services to several service providers or consumers.
19. *Assessment mechanisms (S.1, S.3 and S.4)*: The assessment mechanisms of an abstract architecture is the kind of information used in the architecture to evaluate the goodness of a proposal or agreement in relation to some criteria provided by the user [12]. For instance, the most usual assessment mechanism in service trading is utility functions.
20. *Forms of expressing information and preferences (S.1, S.2, S.3 and S.4)*: The preferences and the information managed about the service and the

parties can be expressed in different ways. Each abstract architecture may have their own way to express them, however, the most commonly used are to express them as constraints or as rules.

## 4 Analysis of Service Trading Architectures

Our goal is to apply the set of properties defined in the previous section to the most relevant abstract architectures. In this context, an abstract architecture is a specification that defines a set of elements (subsystems, components, interfaces, data types, or collaborations) for service trading and that can be applied for different domains and implemented with different technologies. Therefore, it is not the goal of this paper to analyse concrete architectures such as CREMONA [13].

*Open Grid Services Architecture* [5] is an abstract architecture for Grid systems and applications. The analysis of OGSA is based in [5] and other specifications developed by GGF (Global Grid Forum), which detail some aspects not fully described in that document, such as WS-Agreement [9]. The *discovery* employed is external and it is carried out by the so called *information services*. The architecture is *symmetric* for service consumer and provider as both of them may act as agreement initiators in WS-Agreement. There is no specific element to deal with *knowledge adaptation* during discovery, although the use of semantic-enabled discovery services could solve that problem. Concerning the *market observation*, it is achieved by using a subscription mechanism specified in the Grid Monitoring Architecture. Like discovery, both the *information query* and the *third party information* is developed by using the information services. The *agreement creation mechanism* employed in OGSA is the WS-Agreement protocol, although a negotiation protocol is also being developed and agreements for composed services can be created by using the *Execution Planning Services*. Concerning the *deployment*, OGSA is conceived to be deployed as independent services that are later used by higher-level applications. Finally, elements that support the decision-making such as the creation of a *world model* and the *types of information managed about the parties* together with the *assessment mechanisms* and the *forms of expressing information and preferences* are not in the scope of the architecture. This is also the case of the *proposals preselection*, although *Candidate Set Generator* could develop that function.

*Semantic Web Service Architecture* [14] describes an abstract reference architecture for semantic web service interoperability. In this architecture, the *discovery* issues are addressed from the perspective of semantic registries (*Matchmakers*).The *knowledge* management is a key point in this architecture expressed in the specification of different ontologies. In this context, despite the idea of market can be induced from this architecture, there is not an explicit element that actively reacts to different changes in the market (*Market observation*). This architecture is not *symmetric* due it is highly focused in the organisation that acts as service consumer and leaves the service provider as a comparatively simple systems that remain passive during the service trading process. The *information query* mechanism can be developed during the *engagement phase* in the *contract*

*preliminaries* interaction. However, there is not specified an interaction with *third parties*.The interaction mechanisms related with *agreement creation* are based on abstract protocols described in FIPA Conversational Language. There is an explicit requirement for non-repudiation mechanisms during the *enactment* phase. Finally, though it is not specifically stated, this architecture is oriented toward non-cooperative scenarios.

*Web Services Modelling Ontology Full* [15] presents an abstract conceptual architecture for semantic web services and it is oriented to cross-organisational scenarios. It uses an *external discovery* based in the Web Service Architecture and it is symmetric for consumer and provider. It also supports *knowledge adaptation* by using semantic-based service descriptions. However, there is not explicit information about how to carry out a *market observation*, the *information query* nor *third party information*. Regarding the *information managed* about the parties, it uses service and trading information (e.g. supported choreographies) but it is not stated whether it can use information related to the parties. Like OGSA, the mechanisms to support decision-making are out of the scope of WSMO-Full. Therefore, neither the *world model*, the *capacity estimator* nor the *assessment mechanisms* are covered. The *agreement creation mechanisms* supported are specified through the so called *contract agreement choreographies*. WSMO-Full includes partial support for *decommitment* in the post-agreement choreography but the mechanism is not fully defined. It also partially supports *trading protocols* through contract agreement and post-agreement choreographies but it does not consider the specification of temporal constraints on them. However, WSMO-Full does not include any support for complex service trading elements such as a *notary* or the *creation of agreements for composed services*. The architecture seems conceived to operate in a non-cooperative agreement creation, although there is no explicit limitation in using it in a cooperative environment. Finally, as it is a conceptual architecture, it does not consider any *deployment* options.

*Adaptive Service Grid* [4] has been developed as an intent to create service providers that quickly adapt to business changes. In particular, the main goal is to achieve an efficient way of composing services to create more complex services with an added value. In the case of *symmetric* property, the elements that implement the provider-part and consumer-part of the system in ASG are not symmetric. The *discovery* is handled by the so called *DDBQuery* in a centred way through a semantic registry (with reasoning capabilities). In this way, though different ontologies handling are considered as part of the registry there is not an explicit *market* that is observed. Concerning the *world model*, this architecture specifies an element called *ServiceProfiling* that stores information about historic interactions with providers creating a relative model of the provider that is taken into account for the optimisation of the negotiation and selection of services. The *information managed* about parties is highly oriented to service; neither provider nor trading information are described in any of the processes. This approach leaves open the specific negotiation protocol used to establish the SLA for each service composed. However, WS-Agreement standard is specified as an implementation option. ASG can be applied to either cooperative

**Table 1.** Comparison of abstract architectures

| | OGSA | SWSA | WSMO-Full | ASG |
|---|---|---|---|---|
| (1) | Yes (distributed) | Yes | Yes | No |
| (2) | No | Yes (ontologies) | Yes (ontologies) | Yes (ontologies) |
| (3) | Yes | No | No | No |
| (4) | Yes | No | Yes | No |
| (5) | Yes | Yes | No | No |
| (6) | Out of scope | Out of scope | Out of scope | Yes |
| (7) | Yes | No | No | No |
| (8) | Out of scope | Service, party | Service, trading | Service |
| (9) | Partial | Out of scope | No | No |
| (10) | WS-Ag | Yes, FIPA-CL based | Yes, through chor. | Yes (e.g. WS-Ag) |
| (11) | No | Yes | No | No |
| (12) | No | No | Partial | No |
| (13) | Yes | No | Out of scope | No |
| (14) | No | No | Lacks temporal constr. | No |
| (15) | Yes | No | No | Yes |
| (16) | Seems coop. | Seems non-coop | Seems non-coop | Both |
| (17) | Both | Consumer | Both | Chiefly Consumer |
| (18) | Independent services | Out of scope | Out of scope | Technologies |
| (19) | Out of scope | Out of scope | Out of scope | Out of scope |
| (20) | Out of scope | Semantic info | Semantic info | Semantic Info |

or non-cooperative scenarios. Despite ASG describes the architecture of a composed services provider, from an architectural point of view this case is chiefly *service consumer oriented* because it just looks for atomic service providers to be composed. In ASG, the *deployment* possibilities are specified in terms of different development technologies and by identifying subsets of elements that are mandatory and other that can be optional.

## 5 Conclusions

From the analysis developed in Section 4, we can extract several conclusions: (i) The discovery process is well supported and most abstract architectures provides knowledge adaptation; (ii) Most abstract architectures do not cover elements to support the decision-making such as the world model; (iii) There is little support for the most advanced features of service trading such as the notary, the decommitment from established agreements and the trading protocols. Due to these lacks, some complex service trading scenarios cannot be completely achieved. Therefore, it may be interesting to develop new abstracts architecture to deal with those scenarios taking the set of properties obtained in this article as a starting point.

In summary, the contributions of this paper are: first, we obtain a set of properties of abstract service trading architectures based on an analysis of several service trading scenarios, and second, we use these properties to analyse and

compare the most relevant abstract architectures for service trading. In so doing, we set the basis for the development of a method to select the service trading architecture most appropriate to the scenario where it is applied.

The future work is twofold. On the one hand, analysing additional service trading scenarios to identify the properties that an abstract architecture for service trading must have to successfully operate in them in order to define a method to select the architecture for each of them. On the other hand, we intend to extend the work to lower-level properties of non-abstract architectures so that they cover concrete technologies, protocols and algorithms.

## References


1. Molina-Jiménez, C., Pruyne, J., van Moorsel, A.P.A.: The role of agreements in it management software. In: Architecting Dependable Systems III. (2004) 36–58
2. Ludwig, H., Gimpel, H., Dan, A., Kearney, R.: Template-Based Automated Service Provisioning - Supporting the Agreement-Driven Service Life-Cycle. In: ICSOC. (2005) 283–295
3. Ludwig, H.: A Conceptual Framework For Building E-Contracting Infraestructure. In Corchuelo, R., Wrembel, R., Ruiz-Cortes, A., eds.: Technologies Supporting Business Solutions. Nova Publishing (2003)
4. Laures, G., Jank, K.: Adaptive Service Grid Reference Architecture. `http://www.asg-platform.org` (2005)
5. Global Grid Forum, .: Open Grid Service Architecture. `http://www.ggf.org/documents/GFD.30.pdf` (2005)
6. Lee, J., Goodwin, R.: Ontology Management for Large-Scale E-Commerce Applications. In: DEEC. (2005) 7–15
7. Giunchiglia, F., Yatskevich, M., Giunchiglia, E.: Efficient Semantic Matching. In: ESWC. (2005) 272–289
8. Jennings, N.R., Faratin, P., Lomuscio, A.R., Parsons, S., Wooldridge, M., Sierra, C.: Automated Negotiation: Prospects, Methods and Challenges. Group Decision and Negotiation **10** (2001) 199–215
9. Andrieux, A., Czajkowski, K., Dan, A., Keahey, K., Ludwig, H., Nakata, T., Pruyne, J., Rofrano, J., Tuecke, S., Xu, M.: WS-Agreement Specification (2004)
10. Sandholm, T., Lesser, V.: Leveled commitment contracts and strategic breach. Games and Economic Behavior **35**(1) (2001) 212–270
11. Chung, J.Y., Bichler, M.: Service-oriented enterprise applications and Web service composition. Inf. Syst. E-Business Management **3**(2) (2005) 101–102
12. Wang, Y., Tan, K.L., Ren, J.: Towards autonomous and automatic evaluation and negotiation in agent-mediated internet marketplaces. Electronic Commerce Research **5**(3 - 4) (2005) 343–365
13. Ludwig, H., Dan, A., Kearney, R.: Cremona: An Architecture and Library For Creation and Monitoring of WS-Agreements. In: Proc. of the 2nd International Conference On Service Oriented Computing, ACM Press (2004)
14. Burstein, M., Bussler, C., Zaremba, M., Finin, T., Huhns, M.N., Paolucci, M., Sheth, A.P., Williams, S.: A Semantic Web Services Architecture. IEEE Internet Computing **9**(5) (2005) 72–81
15. Preist, C.: Agent Mediated Electronic Commerce Research At Hewlett Packard Labs, Bristol. SIGecom Exch. **2**(3) (2001) 18–28