A PATTERN APPROACH TO MODELING THE PROVIDER SELECTION PROBLEM

José Javier Durán
Centro para las Tecnologías Inteligentes de la Información y sus Aplicaciones (CETINIA)
Universidad Rey Juan Carlos, C/ Tulipan s/n, Mostoles, Spain

Carlos A. Iglesias*
Departmento de Ingenería de Sistemas Telemáticos, Universidad Politécnica de Madrid
Ciudad Universitaria s/n, Madrid, Spain

Keywords: Service oriented computing, Software engineering, Web-services.

Abstract: This article introduces the notion of agreement patterns, which provide a framework for modelling reusable problem solution descriptions for agreement fulfilment. In particular, the Provider Selection pattern has been identified for modelling the common problem of selecting a provider by a service consumer. The article presents the pattern structure as well as the reusable domain model and cognitive structures. Agreement patterns aim at providing reusable patterns useful for developers in multidisciplinary areas, such as Agent Technology and Service Oriented Computing.

1 INTRODUCTION

Throw the years, different computation paradigms have had a predominant factor in the software development, from the begging when a centralized, an monolithic, architecture conformed any application, to a more relaxed one, in which clients presented more capability to work in the system, and converging to an architecture in which each element should be as important in the system, and also totally replaceable, been that view the Internet of services, also known as the cloud computing paradigm.

That vision of an Internet of services comes from the arise of the *Service Oriented Computing* (SOC), in which a system divided in components is built using different services offered by third-parties, or using that abstraction to decentralizing the system itself. This vision has the goal to simplify the development of complex systems, and to cut down costs of the final product. As a result of that behaviour, it has appeared a market of different service providers, offering reusable services, which as a result make possible to create more solutions based in the SOC vision of software development.

A menace of SOC, is a problem that always has been in the software development, how different components are used together, which in fact it is performed taken care of the developer documentation, which in this case is the contract of each service. This contracts, which are described using standards like WS-Agreement (Andrieux et al., 2007), may lack of some information, or could have errors in it, some case it is possible that a contract is unreal, in those cases it is necessary to enrich contracts with trust and reputation information of the developer, in most cases using social information, or performing it another party.

Another problem of SOC, because been an emerging development paradigm, is the fault of information about the good practices in that development. Other paradigms, like the object-oriented paradigm, establish different patterns of how to solve a generic problem, performing a defined set of steps, in our case, the objective of such kind of patterns is the way which different agreements are treated, and how a system performs different maintenance task for its *service oriented architecture* (SOA). The objective of those patterns is to help developers with a huge catalogue of SOC good practices in common problems, and how

*The second author has been partially supported by Germinus XXI (Grupo Gesfor) under the project RESULTA.

Javier Durán J. and A. Iglesias C. (2010).
A PATTERN APPROACH TO MODELING THE PROVIDER SELECTION PROBLEM.
In *Proceedings of the 2nd International Conference on Agents and Artificial Intelligence - Agents*, pages 38-45
DOI: 10.5220/0002726400380045

they are solved.

The goal of this article is presenting how agreement technologies, developed within the agent research community, can be modelled as software patterns, providing a framework for capturing best practices for recurring problems, a common vocabulary and reusable solution descriptions. This article is focused on describing how agreement patterns can describe the recurring Provider Selection problem.

The rest of the article is organized as follows. First, section 2 presents the Provider Selection Problem, and the main approaches to solve it. Then, section 3 gives an overview of the notion of agreement patterns, their classification scheme an how provider selection patterns are classified. Section 4 presents the ProviderSelection pattern, describing its structure, usage and examples. Finally, section 5 draws out the main conclusions and future works of this work.

2 THE PROVIDER SELECTION PROBLEM

Selection of the right parties to interact with is a fundamental problem in open and dynamic environments (Şensoy, 2008). This problem is recurrent in SOC environments, where dynamic service selection (Maximilien and Singh, 2004) allows to combine available services to user needs. It is applied as a need to find and select an specific service for the purpose of the system, first needing to identify the service properties, which is called *quality-of-service* (QOS), and the resources that could be spent by that need, and after that, it is necessary to find a provider that offers that service, which in fact sometimes is not possible, and it is necessary to switch to the service provider that offers the more suitable service. Because of those properties, this problem is pillar for SOC.

This problem is present in different environments, which some of them are listed next:

- **Travel Agency** (Billhardt et al., 2007)**.** An user requires an offer of a travel, which is formed of the flight and an hotel, selecting the one that suits it's preferences, like flight company or arrival time, and that has the lower cost.
- **e-Commerce** (Aydoğan, 2008)**.** Different providers offer a similar product, but with different qualities, and costs, and the user (or a mediator agent) should select one provider among them, based on similarity to its preferences, and cost restrictions.
- **WiFi Roaming** (Merino et al., 2005)**.** The user must select among different access points, those that have a reliable security, and a good quality of signal.

Each of these domains represents a problem, in which confidence of bidders is necessary, and in which solutions taken are similar.

First, it is imperative to determine which roles are conforms this problem, in fact which are the parts that interact between them to advert and negotiate service agreements.

- The main role is the consumer, or user, role, which suits with the system that needs the service to be implemented in, and that will be responsible to find, establish, and maintain different agreements with service providers to use a service.
- The other main role in this interaction is the service provider one, which is responsible of keep offering their services and helping to establish agreements to use those services.
- A secondary role of this interaction is performed by the marketplace, in which different providers are registered to help consumers to list them, and ask them about a service. The marketplace role could be implemented by the consumer, as a predefined list of providers, by a service provider offering that service, or throw a social collaborative network of provider's directory.

Also it is necessary to define the main aspects of this problem, in this case, what are the aspects of the interaction, which will grant a better quality to the solution adopted:

- **Providers Market-place.** It is a component that knows which providers are able to offer an specific service, in this case retrieving different offers as agreements. This component should be a predefined list of providers accessible to the user, another service that offers a provider search engine, or a collaboratively created directory of providers.
- **Service Offer Evaluation.** Each received offer is evaluated, so, they are comparable between them, using different techniques, that is more detailed in Section 2.1.
- **Agreement Negotiation.** Once one offer has been selected, it is done a negotiation process, as stated in Section 2.2.

2.1 Service Offer Evaluation

The main problem around service selection is how to select the most appropriate one, taking care of different factors (Singh and Huhns, 2005):

- **Quality of Service**: determined by similarity with required specifications, cost, and availability. It is necessary to remark that agreements' information could be difficult to understand by the system, because there should be different kinds of properties. The main properties are the functional ones, which could be measured easily, like final cost or minimal bandwidth offered. In counterpart, non-functional properties are more obscured, and difficult to treat, like the feedback from user.
- **Trust of the Provider** (Yang et al., 2006): the service offered by a provider depends on the trust that is assigned to that provider. This information is totally non-functional, but in some cases should be treated like functional, e.g. mean times a services has failed, or what was the perceived quality of the service, for example, using a rating from 0 to 5.

Taking care of that aspects, there are several methods for rating a service. They can be distinguished the following approaches (Singh and Huhns, 2005):

- **Rating** (Şensoy, 2008)**.** This technique use social collaboration to create a score based board, also known as "'Social service selection"', in which different user scores each provider, or also services of each provider, with a value used to create a feedback about user feel of QOS. This technique is useful when a service is obscure, for example non guaranteed inversions, but lacks of subjectivity of users, been manipulable by users. Also it is richer that other techniques, in the fact that there exists information about service feedback that could be used to enrich the selection process.
- **Ranking** (Maximilien and Singh, 2004)**.** This other, instead, use heuristics to determine how the QOS of a service is near to the needed service, also known as "'Semantic service selection"', an treating semantic distance to required one as the evaluation metric, and *ranking* different offers to select the one that suits better. It is necessary to extract semantic information in which the service could be measured to understand how near it is to the required service.
- **Economic Service Selection.** Another technique to select a provider, simpler but the most appropriate in some cases, is to select those ones that are offering a service with a lower cost, in which only quantitative properties of the QOS are taken into account, for example time taken to treat a petition, or availability of the service in a determined time slot.

Selection of which technique to use is not trivial, for example, in a trusted network of providers it is better to select economic techniques, or ranking if the service definition is ambiguous, as it is expected that providers are trustworthy, but in no trusted networks, it is preferred to use rating ones. Also, it is possible to fuse different techniques, so the information of the service offer is richer, and in that way, the service will be selected wisely, but it is necessary to weight how each technique depends on the environment, to assure that the system is not highly restrictive with no trusted providers, and is cost-balanced.

An advantage of ranking techniques is that it is prepared to sustain any heuristic used to match offers with required service, been able to use anyone in a *trust-based component*. The advantage of abstracting this knowledge to a different component is that it is possible to fit the system requirements in this component, and reuse the rest of provider selection interaction.

2.2 Agreement Negotiation

Once a provider is chosen to establish an agreement with, it is done a negotiation process, in which consumer and provider offers each one an offer for that agreement, defining which valuables will be exchanged, in this case they are QOS and cost for service renting. For this process, there are different techniques (Bromuri et al., 2009), which assure that an agreement will be establish, but a problem of this negotiation is that it presents a Nash equilibrium problem, in which two parties have interest in conflict with each other.

Actually, this negotiation is performed after a provider is selected, but a more wide vision of the provider selection problem should be able to establish a negotiation process between the consumer and all the providers, been able to select the one that accepts first an offer from the consumer. This interaction should be treated as a bidding process, and presents more complexity that the provider selection problem, and would be suitable for a future agreement pattern.

3 AGREEMENT PATTERNS

When working with a new technology, or beginning to work with one that previously existed, it is useful to have access to a collection of rules, examples, and descriptions, of the principal aspects of that technology. In software development there are programming languages as rules, source code examples, and software patterns as descriptions of good approaches taken to produce a specific piece of software. Those software patterns are intended to offer new developers

a view of how to approach to a solution that is proven to have good properties, and also offering information that contrast that. Also, different works have proven that patterns are really usable as experience representation and distribution (Oluyomi et al., 2006).

Agreement technologies (Jennings, 2005) (AT) is a recent discipline which collects this multidisciplinary research and can be defined as *the technologies for the practical application of knowledge to the automated fulfilment of agreements*. Agreement technologies do not dictate the underlying technologies (objects, components, agents, services, ...), but are focused on the formalization of knowledge structures, protocols, algorithms and expertise that contribute to the establishing of agreements in an open dynamic environment.

Based on this previous research, **Agreement patterns** (Iglesias et al., 2009) are defined as *software patterns which helps software components coordination through the fulfilment of agreements*. Agreements patterns include all kind of agreements, both explicit ones (e.g. negotiation) and tacit ones (e.g. organization).

The main need of the pattern engineering is to define a pattern template that reflects the needs from the domain in which it will be used. In the case of agreement oriented services there is important to determine those ones:

- Participants, or Roles.
- Trigger
- Purpose

Taking those aspects in account, it should be possible to define the pattern template. In this case we propose the use of the canonical form (also called Alexandrian) for software patterns informal description(Buschmann et al., 1996), which it has been enrich with the elements listed before:

- **Name:** a meaningful name that provides a vocabulary for discussing.
- **Alias:** an alternative name to the pattern.
- **Participants:** who are the main participants in the interaction, and their roles.
- **Trigger:** why the interaction process begins, and with which interactions is related. It will describe when it is used.
- **Purpose:** what is the problem that it solves.
- **Problem:** a statement of the problem and the goals it wants to reach.
- **Context:** the preconditions under which the problem and its solution seem to recur.
- **Forces:** a description of the relevant forces and constraints and how they interact with one another and with the goals. Considerations to be taken into account to select a solution for a problem.
- **Solution:** static relationships and dynamic rules describing how to realize the desired outcome. It should be described using pseudo-code, class diagrams, reasoning diagrams, or any model that helps to understand the solution.
- **Examples:** one or more sample applications of the pattern which illustrate its application. Known occurrences of the pattern, which help in verifying that the pattern is a proven solution to a recurring problem.
- **Resulting Context:** the state or configuration of the system after the pattern has been applied.
- **Rationale:** a justification of the pattern, explaining how and why it works, and why it is "good".
- **Related Patterns:** compatible patterns which can be combined with the described pattern.

All this points forms a good batch of questions about the pattern itself, helping to determine the inner of the pattern, as assuring that it is a true pattern widely useful, and not an anti-pattern(Rising, 1998) of bad manners in software development, that don't offer and extensible and reusable interaction process for service oriented computing.

In order to classify agreement patterns, a classification scheme has been proposed (Iglesias et al., 2009), which has identified the following dimensions:

- **Duration.** Is the agreement established temporally, short term, or permanently, long term?
- **Normative Context.** The pattern is strict as a established norm, or flexible?
- **Topic.** Which is the main purpose of the pattern? E.g.: Service offering, Service negotiation, or Service bidding.
- **Phase.** What moment of the agreement life-cycle it represents? E.g.: negotiation, conclusion or selection.
- **Decision Making.** How selection process are performed? E.g.: In provider selection it should be social-collaborative, but in agreement portability it should be rule based.

4 THE PROVIDER SELECTION PATTERN

Using the provider selection problem as example of how the agreement patterns are applied, they are go-

ing to be defined a series of steps and models to be used as formalization of it. The purpose of those models is to define the interaction process that is part of this problem.

4.1 Problem Description

The pattern template previously showed in section 3 would be applied to the provider selection problem, which presents a good number of factors to consider it the basis of service oriented computing:

- It is use when it is required to create an adaptable system.
- Provider selection trust is required to assure system's assurance.
- It will dynamically establish agreements as required to manage different offers, and select the best that suits requirements, and cost factors.

The main purpose of the agreement is to unify all the information about different approaches taken in this scenario to be able to present it in a formal way, accessible by different developers, to take care of the pattern whenever a system requires its capabilities.

Those aspects assures the need for a formal description of how a solution is obtained, as presenting the problem enough complexities, and been widely used and generic.

4.2 Description of the Agreement Pattern

Based on the pattern structure described in section 3, the solution to the Provider Selection problem can be described as follows.

- **Name.** Provider selection.
- **Alias.** Service selection
- **Duration.** Variable. Based on system and purpose of service.
- **Normative Context.** Flexible.
- **Topic.** Service provider selection.
- **Phase.** Agreement selection.
- **Decision Making.** Based on trust/reputation mechanism, mainly social-collaborative.
- **Participants.** *User*, that requires a service; *Service provider*, that offers an agreement for a service to be used by the *User*; and the *Market place*, which list the different *Service provider* that are offering services. Relations between roles are present in Figure 1.

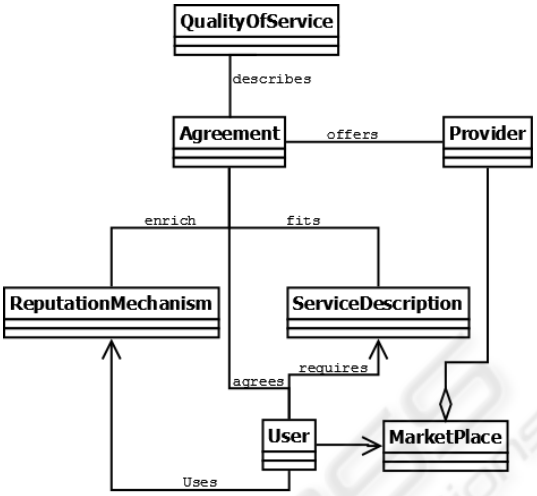

Figure 1: Participant classes in the provider selection problem.

- **Trigger.** A system requires a service, commonly with some restrictions or preferences on its non functional properties (QoS, price, ...), and there is more than one service provider that fits in that description.
- **Purpose.** Retrieve the best provider, and establish a service usage agreement with it.
- **Problem.** A user requires an agreement with a provider, that must offer a service with required properties, like quality of the service, cost or trust in that provider, and as a result, an agreement is done with the most appropriate provider.
- **Context.** The user has access to a market of offered services, and a trust system.
- **Forces.** Trust and reputation techniques to enrich bidders information.
- **Solution.** See Algorithm 1, for a pseudo-code description of the solution.

  1. The user asks bidders in a service providers' market, for a service offer, with an specific properties.
  2. Each provider offers a different proposed agreement, including non functional properties, such as costs or QOS information.
  3. The user enriches the information in each agreement with trust information, using a trust-based component, like a collaborative reputation system, self-experience, or heuristics for service matching.
  4. Agreements are evaluated, based on their non functional properties as well as based on the

trust and reputation of the provider, using a specific evaluation function based on the system purpose, for example, a high security system will evaluate poorly any system without good trust information.

5. If there is almost one provider with an acceptable evaluation, it is realized an agreement with it. A threshold must be defined to don't establish an agreement if all providers are offering invalid agreements. Once the target provider is chosen, it begins a negotiation process, in which the consumer tries to establish the agreement with highest utility.

Algorithm 1. Pattern solution pseudo-code.

```
Require: RequiredServiceDescription
Require: ReputationMechanism
Require: ProvidersMarketPlace
  OfferedAgreements = ProvidersMarket-
  Place.askFor( RequieredServiceDescription
  )
  while Agreement a in OfferedAgreements do
    ReputationMechanism.enrich( a )
    if RequiredServiceDescription.isBetter( a, best )
    then
      best = a
    end if
  end while
  if best.assures( RequiredServiceDescription ) then
    ServiceAgreement = best.AgreeProposal( )
  end if
Ensure: ServiceAgreement.assures( RequiredSer-
  viceDescription )
```

- **Examples.**
  - Broadband access negotiation (Merino et al., 2005), in which users selects the provider that fits with its needs, and use other users feedback to select the most appropriate.
  - Ad-Hoc service negotiation (Song, 2008), in which a provider offers different services, in which the QOS changes, but fits better with the required service as an increase of cost instead. In this case the user tries to find equilibrium between service matching and cost assumed.
  - E-Commerce (Aydoğan, 2008), in which several providers are offering the same good but with different important aspects, mainly shipping method and final cost, and the user measures the offering, with other users feedback, like comments in the provider web, and feedback about trustworthy of the provider.
- **Resulting Context.** The user establish an agreement, if an acceptable one is offered among the providers.
- **Rationale.** It defines the interaction basis in the search for an agreement when it is necessary to compare different offers, and enrich them with trust systems.
- **Related Patterns.** Agreement Portability.

4.3 Reasoning Cognitive Pattern

To help to understand how this problem could be driven, it is possible to divide it in different tasks, which should be threatened independently, except of how they are interconnected. This tasks interconnection is described in figure 2. The purpose of those tasks is as follows:

- **Estimate.** Enriches providers offers with trust information from the trust/reputation knowledge.
- **Assess.** Selects a provider that fits the user requirements of the QOS. In this task it is measured how the service proposal is similar to the required service, and the trust information of the provider.

Those tasks treat different information from different knowledge sources:

- **Service Provider Offer.** This information is obtained from asking to the market place about service providers that fit a required need for a service. It should be described using an agreement definition language, like WS-Agreement.
- **Trust/Reputation.** This information should be obtained from a service of service providers reputation, the provider itself, or from a social-collaborative source. It should be a quantification of average number of service losses, a measure of principal properties, like average bandwidth in a WiFi access point; or feedback of other users, which requires to apply a new trust filtering to that information.
- **Provider Agreement.** This is the final product of the problem, in which the system, after selecting a service provider, creates an agreement, in which the consumer ask for access to the specified service. This agreement can be represented by an agreement language such as WS-Agreement.

5 RELATED WORK AND CONCLUSIONS

This article has presented *agreement patterns* as an instrument for modelling reusable solutions.

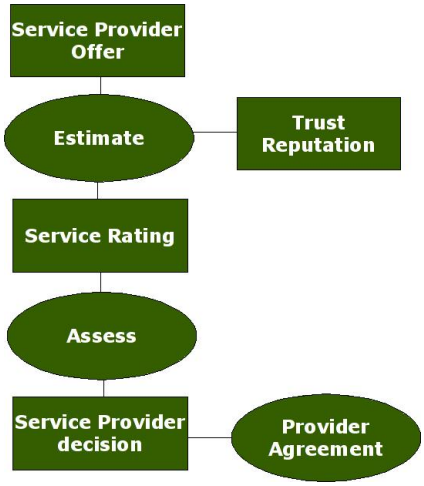

Figure 2: Provider Selection reasoning diagram.

There are related works for defining design patterns in the areas of multi-agent systems and Service Oriented Computing (SOC).

Agent-Oriented Patterns have been defined for sharing multi-agent system development experiences. Oluyomi (Oluyomi et al., 2007; Oluyomi, 2006) presents an agent pattern classification scheme based on two dimensions: stages of the agent-oriented software development and tasks in each stage of development. At each stage or level of development (analysis, multi-agent architecture, agent architecture, multi-agent implementation), the framework identifies the attributes of that level of abstraction, in order to classify these patterns. In addition, Oluyomi proposes to refine the canonical pattern form for defining an Agent-Oriented Pattern Template Structure, which adds more granularities depending on the pattern type (agent internal architecture structural, interactional or strategic patterns, etc.). Some of the patterns identified by Oluyomi, whose classification scheme includes other approaches, can be considered agreement patterns. The main differences between her classification and the one proposed in this article is that Oluyomi's classification is agent oriented, and it is hard to use if it is not implemented with agents (agent oriented development phase, agent architecture, etc.), while the one proposed here is independent of the technology to be used, although implementation examples can be presented with different technologies. In addition, agreements are not a key concept in Oluyomi's classification scheme as in our proposal. Future work will provide a mapping of the agreement related patterns classified by Oluyomi onto our classification scheme.

In the area of Service Oriented Architecture (SOA), *SOA patterns* have been defined (Erl, 2008; Rotem Gal Oz, 2009; Zdun et al., 2006). For example, Erl (Erl, 2008) classifies patterns for architecture services, service compositions, service inventories and service oriented enterprise. Rotem-Gal-Oz (Rotem Gal Oz, 2009) describes patterns for Message Exchange, Service Interaction, Service Composition, Structural, Security and Management. SOA patterns (Zdun et al., 2006) provide high level architectural patterns, which do not detail yet agreement issues.

Inside the SOC community, the GRAAP Working Group (Grid Resource Allocation and Agreement Protocol WG) has defined the specification Web Services Agreement (Andrieux et al., 2007), which is particularly interesting for this research. The purpose of the specification is the definition of a Web Services protocol for establishing agreements defined in XML. The specification covers the specification of agreement schemas, agreement template schemas and a set of port types and operations for managing the agreement life cycle. This specification defines an agreement as *an agreement between a service consumer and a service provider specifies one or more service level objectives both as expressions of requirements of the service consumer and assurances by the service provider on the availability of resources and/or service qualities. An agreement defines a dynamically-established and dynamically-managed relationship between parties. The object of this relationship is the delivery of a service by one of the parties within the context of the agreement. The management of this delivery is achieved by agreeing on the respective roles, rights and obligations of the parties*. An agreement is characterized by its name, context and terms.

The OASIS Reference Architecture for SOA (McCabe, 2008) is *an abstract realization of SOA, focusing on the elements and their relationships needed to enable SOA-based systems to be used, realized and owned*. The reference architecture defines three primary viewpoints: *business via services* that captures what SOA means for people using it to conduct business, *realizing service oriented architectures* deals with the requirements for constructing a SOA; and *owning service oriented architectures* addresses issues involved in owning and managing a SOA. The notion of agreement is included in several ways in the architecture, as an organizational concept (constitution) or as a formalization of a relationship (business agreement and contract).

These two initiatives, OASIS RA and WS-Agreement are compatible and complementary of our proposal, since they provide a modelling reference architecture as well as a language for describing the

identified patterns boiling down to the implementation level. This integration will be include in future publications.

The pattern ProviderSelection described within this article illustrates how agreement patterns can help in providing a common vocabulary as well as a collection of best practices for engineering agreement-based distributed applications.

Future works will validate this model with other problems, and represent a compendium of agreement based problems, and their solutions, which they purpose is to help developers to take each pattern an assemble a system capable of interact with service providers, without needing to know how the interaction must be done, instead knowing the required elements to be implemented.

REFERENCES

Andrieux, A., Czajkowski, K., Dan, A., Keahey, K., Ludwig, H., Nakata, T., Pruyne, J., Rofrano, J., Tuecke, S., and Xu, M. (2007). Web services agreement specification (WS-Agreement). Technical report, Grid Resource Allocation Agreement Protocol (GRAAP) Working Group.

Aydoğan, R. (2008). Content-oriented composite service negotiation with complex preferences. In *AAMAS '08: Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems*, pages 1725–1726, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

Billhardt, H., Hermoso, R., Ossowski, S., and Centeno, R. (2007). Trust-based service provider selection in open environments. In *SAC '07: Proceedings of the 2007 ACM symposium on Applied computing*, pages 1375–1380, New York, NY, USA. ACM.

Bromuri, S., Urovi, V., Morge, M., Stathis, K., and Toni, F. (2009). A multi-agent system for service discovery, selection and negotiation. In *AAMAS '09: Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems*, pages 1395–1396, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

Buschmann, F., Meunier, R., Rohnert, H., Sommerlad, P., and Stal, M. (1996). *Pattern-oriented software architecture: a system of patterns*. John Wiley & Sons, Inc., New York, NY, USA.

Şensoy, M. (2008). *A Flexible Approach For Context-Aware Service Selection In Agent-Mediated E-Commerce*. PhD thesis, Boğaziçi University.

Erl, T. (2008). *SOA Design Patterns*. Prentice-Hall.

Iglesias, C. A., Garijo, M., Fernandez-Villamor, J. I., and Durán, J. J. (2009). Agreement patterns.

Jennings, N. (2005). Agreement technologies. *Intelligent Agent Technology, IEEE / WIC / ACM International Conference on*, 0:17.

Maximilien, E. M. and Singh, M. P. (2004). A framework and ontology for dynamic web services selection. *IEEE Internet Computing*, 8(5):84–93.

McCabe, F. G. (2008). Reference architecture for service oriented architecture. Technical report, OASIS.

Merino, A. S., Matsunaga, Y., Shah, M., Suzuki, T., and Katz, R. H. (2005). Secure authentication system for public wlan roaming. *Mob. Netw. Appl.*, 10(3):355–370.

Oluyomi, A., Karunasekera, S., and Sterling, L. (2006). Design of agent-oriented pattern templates. In *ASWEC '06: Proceedings of the Australian Software Engineering Conference*, pages 113–121, Washington, DC, USA. IEEE Computer Society.

Oluyomi, A., Karunasekera, S., and Sterling, L. (2007). A comprehensive view of agent-oriented patterns. *Autonomous Agents and Multi-Agent Systems*, 15(3):337–377.

Oluyomi, A. O. (2006). *Patterns and Protocols for Agent-Oriented Software Development*. PhD thesis, Faculty of Engineering. University of Melbourne, Australia.

Rising, L., editor (1998). *The patterns handbooks: techniques, strategies, and applications*. Cambridge University Press, New York, NY, USA.

Rotem Gal Oz, A. (2009). *SOA Patterns*. Manning.

Singh, M. P. and Huhns (2005). M.n.: Service-oriented computing: Semantics, processes, agents. *J. Wiley and Sons*.

Song, W. (2008). *Building dependable service-oriented application via dynamic reconfiguration and fault-tolerant reconfiguration collaboration protocol*. PhD thesis, Tempe, AZ, USA.

Yang, S. J. H., Hsieh, J. S. F., Lan, B. C. W., and Chung, J. (2006). Composition and evaluation of trustworthy web services. *Int. J. Web Grid Serv.*, 2(1):5–24.

Zdun, U., Hentrich, C., and Aalst, W. M. P. V. D. (2006). A survey of patterns for service oriented architectures. *Int. J. Internet Protoc. Technol.*, 1(3):132–143.