

Framework for evaluation of services

Research Area: EDEN – Ecosystem Design and Evolution
Project Title: EDEN WP4 – User-driven Service Ecosystem Evolution
Document Type: I (Internal)

Document Identifier: FS-EDEN-D413
Document Title: **Deliverable 4.1.3: Framework for evaluation of services**
Editor: Seppo Törmä
Authors: Seppo Törmä
Status / Issue: Version 1.1
Date Last Change: 12/17/2009 1:53:00 PM
File: FS-EDEN-D413.doc
Delivery Date: 31.05.2009

Document History:

12.02.2009	Document created
13.02.2009	Version 0.1
27.02.2009	Version 0.2
20.05.2009	Version 0.3
26.05.2009	Version 0.4
28.05.2009	Version 0.5
31.05.2009	Version 1.0
28.08.2009	Version 1.1

Abstract

Services need to be evaluated when deciding which ones to use or recommend for other people to use, and how to develop or improve a service. The evaluation depends on perspective: users, providers, and developers of complementary services all have their own criteria. In this report we survey the properties of services to establish a basis for determining relevant evaluation criteria for different situations. The properties are divided into short-term properties that are relevant in individual service deliveries, and to longer-term properties that affect the success of a service in the long run. Short-term properties are classified into functionality, behavior, and non-functional properties. Functionality is the reason for a requestor to invoke a service; behavior relates to the manner in which service is delivered, and non-functional properties act as constraints on the functionality of the service. The longer-term properties are discussed from the three different perspectives: user, provider, and ecosystem. When a service uses and is used by other services and simultaneously needs to compete from limited resources in a service ecosystem, many other properties in addition to the short-term ones need to be taken into account. These range from user experience to loose coupling and existence of viable business models for service providers.

Table of contents

1	Introduction	5
2	Short-term criteria	6
2.1	Overview	6
2.2	Functionality	7
2.3	Behavior	9
2.4	Non-functional properties	10
3	Longer-term criteria	12
3.1	User perspective	12
3.2	Ecosystem perspective	12
3.3	Provider perspective	13
4	Discussion	14
5	Bibliography	15

1 Introduction

Services need to be evaluated in different situations to decide which services to use, which services to recommend to other people, and how to develop or improve a service.

At the first glance it appears that the *amount of use* would be the ultimate definition for the success of a service and therefore also a sufficient evaluation criterion. However, this notion is not without problems. The amount of use can only be measured post hoc and does not therefore provide proper guidance in the development phase of a service. It depends on other issues besides the properties of a service – for instance, marketing efforts for the service – and it may vary over the time even if the service remains the same. Moreover, even large amount of use does not automatically mean success for the service provider; there needs to be a viable business model too. A broader understanding of the properties of services is required, and hopefully the properties help to understand also why some services are more successful than others.

The evaluation of services depends on perspective: users, providers, developers of complementary services, and even regulators all have their own criteria. In this report we survey the properties of services to establish a basis for determining relevant evaluation criteria for different situations. The properties are divided into short-term properties that are relevant in individual service deliveries, and to longer-term properties that affect the success of a service in the long run.

Short-term properties are classified into functionality, behavior, and non-functional properties. Functionality is the reason for a requestor to invoke a service; behavior relates to the manner in which service is delivered, and non-functional properties act as constraints on the functionality of the service. This discussion is based on Törmä (2008).

The longer-term properties are discussed from the three different perspectives: user, provider, and ecosystem. When a service uses and is used by other services and simultaneously needs to compete from limited resources in a service ecosystem, many other properties in addition to the short-term ones need to be taken into account. These range from user experience to loose coupling and existence of viable business models for service providers.

This report is the deliverable D4.1.3 of EDEN WP4 Tools for user-driven service design. It is result of the task WP4.1.3 the contents of which deal with the evaluation criteria of services. The criteria were targeted to the task WP4.1.4 to support a set of experiments with available service composition tools to implement requirements collected from Flexible Services use cases. The criteria defined in this report will serve as the basis for the evaluation of the services created, as well as the service creation process and the tools that were used. We assume that the criteria are also relevant to work in task WP4.2 Monitoring and measuring services.

2 Short-term

In this section we will give an overview of the properties of services as they appear in a single service delivery interaction. The approach is to review the ways that a service can be described. In any description, some of its properties of a service need to be specified; otherwise it is not sensible to talk about a description at all. Different description approaches address different kinds of properties, and review of them gives an overview of possibly interesting properties of services. The properties are again close to the potential evaluation criteria of services.

2.1 Overview of a service interaction process

The two principal participants in an occurrence of a service are (1) the *service requestor* that can also be called a user, client, or customer, and (2) the *service provider*, sometimes also called the server. Both requestor and provider are *relations* of a party to a particular service; they are not classes of agents, since one agent can well be the requestor of one service and the provider of another. There may be other participants acting in mediating roles in the service discovery and provision process. Some examples are *matchmakers* that attempt to discover services matching the demands of a requestor, *brokers* that interact with a service provider on behalf of a service requestor, and *registries* that store information about services to provide it to requestors.

A *service interaction process* consists of the actions that the service requestor and service provider take to fulfill the goals of the requestor. In Burstein (2005) the interaction is seen as driven both by the goals of the service provider – creating a service provision process – and the goals of the service requestor – creating a goal achievement process. The combined interaction process can be divided into the following phases:

1. *Discovery*. The requestor finds candidate services either through a *referral* such as link or recommendation, or through a *lookup* from a service registry where the provider has previously registered a service. To query available services, requestors need to use abstract characterizations of desired services.
2. *Engagement*. The requestor negotiates with the providers of the candidate services until it comes to an agreement about one of the services. The requestor and provider need to talk about the service in much more concrete and detailed manner than what was necessary in the discovery phase. The result is an implicit or explicit service agreement between the requestor and the provider.
3. *Enactment*. The initiation of the execution – that is, delivery and consumption – of the agreed on service. This may include monitoring of the execution. If the contracts objectives are not accomplished, a compensation protocol can be used restore financial equity.

Invocation is the term that denotes the process that begins with the formation of binding service agreement between requestor and provider (O’Sullivan, 2002). The agreement does not need to be explicit. In addition, when a service provider – for example, a Web server – has declared a general commitment to provide a service to any requestor, the formation of a service agreement is reduced to a *call* by the requestor.

In the service interaction process the participants need to *describe services to each other*: the provider describes its service offerings, and the requestor the services desired. The descriptions exchanged in the service interaction process are typically partial, and during a successful process the overall description of the service should become more refined and comprehensive. Throughout the process, the service description framework should provide the means of mapping between the provider’s description of its offerings and the requestor’s description of its needs and goals (Turner, 2003).

The participants need to agree on three aspects of a service: (1) *functional*, (2) *behavioral*, and (3) *non-functional* (Toma, 2006). Functional aspect describes what a service can do. Examples of simple functional

descriptions are the following: “deliver a message to recipient”, “print a document”, or “translate text”. Behavioral aspect describes how the functionality can be achieved both in terms of the externally visible interaction and the internal decomposition into smaller functional pieces. For example, “call a call center, give your location, wait at the location until service personnel arrive, instruct the personnel, accept the service, pay”. Finally, the non-functional properties capture constraints over the previous two aspects (Toma, 2006), such as availability, costs, quality, and so on.

2.2 Functionality

Functionality is arguably the most essential aspect of a service: the desired functionality is the reason for an agent to invoke a service. Non-functional properties, such as price or availability, are secondary in the sense that they only constrain the selection of a service, once functionally suitable candidates have been identified.

It is complicated to specify functionalities in a general and comprehensive manner. This is especially true for services dealing with communication and entertainment. However, for simpler functionalities descriptions the following approaches have been identified in the literature (McIlraith, 2001, Martin, 2004, Gil, 2000):

1. *Capability category* – the similarity with the members of a known category. Each sufficiently homogeneous functionality type is represented as a class in a functionality hierarchy. For example, there can be a class of travel service that has subclasses transportation service and accommodation service. Transportation service can have further subclasses flight service, bus service, and so on and accommodation service subclasses hotel accommodation service, bed-and-breakfast service, and so on.
2. *Structural capability description* – a many-faceted representation of the properties of a service. In (Gil, 2000) capabilities are represented as verb clauses using a case-grammar style of formalism. Each capability consists of a verb, that specifies what is to be done, and a number of roles, or slots, which specify the parameters to be used in the action. For example, the estimation of the closure date of a particular transportation movement could be specified as: estimate OBJ closure-date OF transportation-1.
3. *State transformation* – the relation between the state before and state after a service. For example, a particular flight service may specify that in the state before the execution a passenger is in London and in the state after the execution in Helsinki. In the usual terminology, the conditions that must hold before a service can be started are called preconditions and the conditions that will hold in the end are called effects. This representational approach is needed with world-altering services (McIlraith, 2001), for instance, flight booking service, or electronic commerce services.
4. *Input-output function* – the relation between the inputs and the outputs of a service. For example, a flight passenger could use a service to query flight schedules by providing as inputs the source and destination of a desired flight and the time period of planned trip. The output of the service would be a list of possible flights. This representation is especially suitable for information-providing services (McIlraith, 2001) such as local weather forecasts, flight information provision, and cameras.

If compared to procedure call in a programming language, *inputs* and *outputs* correspond to the *parameters* and *return values* while *preconditions* and *effects* correspond to *assertions* and *side effects*. It should be emphasized that functionality means the relation between these and the body of a procedure implements this relation. Capability categories are not commonly used in programming, since it is difficult to develop sufficiently expressive and precise classifications.

Capability categories have, however, been used in all kinds of business catalogues and yellow pages targeted for human use. There are several existing industrial classification systems (UNSPSC, ISIC, NAICS, etc.) that could form a basis for service classifications.

Classifications can be used in service discovery as a fast way to identify candidate services. At the same time it should be stressed that the knowledge of the class of a service gives only a rough picture of what it actually does. Consequently, it is likely in many domains that many of the candidates identified just on the basis of a category are not actually applicable. Moreover, in strictly hierarchical classifications schemes a service cannot be placed in more than one category, which means that any single category would not necessarily contain all applicable services. In short, a simple category-based discovery mechanism is likely to return both too many and too few services.

Classification is also problematic as a general solution for representing functionalities of services. Services are artifacts – i.e. man-made things – and there are no natural classifications for artifacts. Any hierarchical classification scheme can – and often will – be broken by new classes that combine properties of two or more existing classes (e.g., SUVs that combine properties of cars and small trucks).

Structured capability descriptions make it possible to specify much more information about a service. They are more flexible in the sense that they can be used to specify the relevant aspects of a particular type of service. In addition, different kinds of services can be described in different level of detail. The EXPECT reasoning system (Gil, 2000) uses structured descriptions of capabilities based on description logics. The capabilities can be described as union of capability classes, the values of their properties can be constrained, and so on. The automatic classifier can match the sought out capabilities with the capabilities offered by existing services.

The state-transformation and input/output-function approaches together define so-called IOPE -properties of a service, namely inputs, outputs, preconditions, and effects. This is the common way to specify the functionality of services in Semantic Web Services frameworks.

The following is an example of IOPE-properties presented in W3C OWL-S ontology language specification (2004) (with relevant parts emphasized).

To complete the sale, a book-selling service requires as **input** a *credit card number* and *expiration date*, but also the **precondition** that the *credit card exists* and *is not overdrawn*. The result of the sale is the **output** of a *receipt* that confirms the proper execution of the transaction, and as **effect** the *transfer of ownership* and the *physical transfer of the book from the warehouse of the seller to the address of the buyer*.

The state transformation approach has been used in the classical AI planning research to compose action sequences to satisfy given goals. However, in that area there has been a proposals to extends the simple precondition-effects -models in various ways:

- *Conditional effects*. It is apparent that practical activities do not always produce the intended effects. They can fail for various reasons or produce alternative effects in some circumstances. There is thus an element of non-determinism present and a consequent need to represent multiple different effects qualified by conditions in which they will result.
- *Maintenance goals*. Unlike the achievement goals that define conditions that need to be achieved – that is, hold in the end of the execution – there can also be conditions that must remain true all the way during the execution of a service. For example, the maintenance of some security or integrity conditions. In agent systems maintenance goals are typically managed in a reactive manner, i.e., acted on only during execution if such condition becomes untrue. There are proactive approaches that attempt to anticipate possible failures and plan to prevent conditions to become untrue (Duff, 2006).
- *Temporally extended goals*. A condition can be associated with the time period in which it holds. The period can be specified in absolute time or as depending on other conditions. Extended goals define desired behaviors not only in term of final states to be reached, but also in terms of conditions on the service execution paths.

The fact that a need for these kinds of extensions has been identified in different domains suggests that the simple state transformation model based on preconditions and effects may be insufficient as a generic solution for service modeling.

2.3 Behavior

The behavior of a service describes how the service achieves its functionality. For example, the behavior of a service whose function is to ship goods might be described as: receive the cargo manifest, load the goods, check the mounting, ask for permission to leave, move to target location, find the contact person, unload the goods, and get signature.

There are two aspects of the behavior of a service (Toma, 2006):

1. *Externally visible behavior* refers to the aspects of service occurrence that involve interaction between the service provider, the service requestor, and possible other agents related to the service. This includes the visible changes that the service provider will make to the state of the world, and the communicative behavior specifying what kinds of messages will be sent, what are the message layouts, when will the messages be sent, what kinds of interactions can result from a sent message (message exchange patterns), and so on.

2. *Internal behavior* that describes the process – or the workflow – of a service. How the functionality of a service is achieved by aggregating other services? There are basically two aspects of the process:

- *Constituent services*. From what services are this service aggregated from?
- *Constraints and conditions*. How are the constituent services combined?

The constituent services typically have constraints on their execution order: the execution of a service can be started only when other services have been completed. There can be different kinds of reasons behind an ordering constraint. Some artifacts or information can flow from one service to another. The services may use shared resources whose limited capacity does not allow parallel execution, resulting in a disjunctive ordering constraint: either before or after. One service may destroy the desirable state changes (e.g., order, or cleanliness) established by another.

The ordering constraints can be approximated either with explicit links or precedence relations between services – in a graph-like manner as in project networks – or with control structures such as sequence, parallel execution, and so on – in a block structured manner as in programming languages. The execution of some constituent services may be conditional as they may depend on unknown and uncontrollable external conditions. The process may also contain loops where certain services are iterated until a desired condition holds. Services may also encounter exceptional situations and terminate in abnormal way, which requires reactive measures from the service provider.

If the service allows for alternative internal processes, it is important to understand who can make the decision between the alternatives. If the service provider makes the decisions, the process specification can be thought of as a program or execution specification for the service. If the decision is made by external agent, e.g. service requestor, the process is a specification of an interaction between provider and requestor.

The externally visible behavior is naturally a result of the internal process of a service. However, there can be different internal processes that lead to the same external behavior.

2.4 Non-functional properties

Non-functional properties¹ can be regarded as constraints on the functionality and behavior of a service. They play an important role in service selection, negotiation and monitoring processes, as well as in service contracts or service level agreements if such are formulated in an explicit manner.

The research reported in O’Sullivan (2002) – and elaborated with detailed ORM (Object-Role Modeling) diagrams in O’Sullivan (2005) – identified the following kinds of non-functional properties based on a review of existing commercial services:

1. *Availability-related properties*. The availability of a service can be subject to different kinds of constraints. Specific properties are temporal availability that tells when, and spatial availability that tells where the service is available. The representations of these properties can be complex, especially if mobility must be taken into account.
2. *Channels*. Service interaction can be regarded as occurring through channels that are characterized by the endpoints, the information being transmitted, and the interaction pattern that occurs. A service can be accessible through multiple channels with different properties.
3. *Cost-related properties*. The definition of the compensation provided by the service requestor to the service provider has a number of aspects.

Charging style specifies the way that the price is determined; for example, fixed amount per service delivery (e.g., fixed price local telephone call), based on a unit of measure (length of a long distance call), or on a percentage on some aspect of a service (by commission). The different styles can be mixed and can also be redirected to external parties (e.g., to advertisers).

Settlement model defines the process of fulfillment of the mutual obligations of service provider and service requestor. Basic settlement models are transactional and rental. The settlement can be specified in an explicit form in a settlement contract.

Payment-properties relate to the schedule – the number and timing – of monetary transfers. The timing can depend on the occurrence of other events, for instance, on the end time of a successful service delivery. Also relevant are constraints on the payment instruments such as cash, cards, checks, or vouchers. Payment instruments are characterized by a number of properties such as usability over the network, acceptability to receiving party, traceability, refutability, liquidity, security, expiration, transferability, immediacy, and negotiability.

Discounts and penalties are properties closely related to the costs of a service. A service provider may use discounts in various ways to attract new customers or to reward loyal customers. Penalties relate to effects of non-compliance with some obligation of service provision.

4. *Quality-related properties*. A measure of the difference between expected and actual service provision. For example, the customer perception of the quality can be measured through five dimension: *reliability* means the dependability and accuracy of the service, *responsiveness* means the promptness and willingness of the service provider to assist, *assurance* deals with trust conveyed by attributes related to capabilities of

¹ It should be noted that the term “non-functional properties” is sometimes used in a sloppy or confusing manner. In WSMO - Web Services Modeling Ontology (Feier, 2005), for instance, the term is applied to properties of the *description* of a service, not properties of the service itself. Examples of such description-related properties include creator, date, format, language, owner, publisher, and version. These could be called “properties of service descriptions” instead of non-functional properties of services.

provider, *empathy* for personalized attention and caring provided, and *tangibles* for concrete aspects of service (e.g. orderliness).

A service provider can be committed to a certain level of quality, which can be specified in an explicit manner in a *Service Level Agreement (SLA)*. There are two related concepts worth mentioning in this context: *Quality of Service (QoS)* means quality properties of a service that can be objectively measured, and *Quality of Experience (QoE)* refers to subjective evaluations by the service requestors regarding how satisfactory the service delivery was.

Usability is an important quality-related attribute in digital services meant for end users. Usability measures how effective, efficient, and satisfactory a service is in its intended context of use.

5. *Security and trust*. Security alleviates concerns relating to identity, privacy, alteration and repudiation between parties. With composite services, there are a lot of difficult questions about security management: What is the relationship of authentication of a service and authentication of its subservices? Should service compositions have security certificates?

What happens if some subservices require security and others not? Do all subservices obey the same policy with respect to handling of the client's information?

Trust becomes an important issue in an open environment characterized by lots of interactions among strangers: people, companies, and intermediaries. Trust deals with the management of interactions at the application level (Singh, 2005): while security is about authenticating another party or authorizing actions, trust is about a party acting in the best interest of another and choosing the right actions from those authorized. Trust relate both to the intentions and the competence of an agent (O'Sullivan, 2002). Singh (2005) identifies the following ingredients of trust: capability, sincerity, helpfulness, duty, predictability, and understanding. One approach to manage trust are reputation mechanisms.

6. *Ownership and rights*. As a rule, provision of a service does not create ownership. However, service requestors typically have following kinds of rights. The *right to comprehend* means that a requestor has the right to question the service provider about the service to better understand it. The *right to retract* means that the requestor does not need to request a service refined into a negotiated contract. The *right of premature termination* means that requestors may have the right to terminate the delivery of a service before it has been completed. The *rights of suspension and resumption* mean that the delivery of a service can be temporarily interrupted.

3 Long-term

The long term success and survival of a service will be affected by a number of properties in addition to those outlined in the previous chapter. These longer-term properties concern questions such as the following. Will the user come back and use the service again? How will a service benefit other services or benefit from other services? How can it adapt to changes in the ecosystem? Will all the parties that are required for the services to operate receive more than they need to give? For example, is there a viable business models for the service providers?

The parties that are interested of a service – its stakeholders - include users, providers, competitors, collaborators, regulators, and so on. Below the longer-term evaluation criteria of a service are discussed from three different viewpoints: the users of a service, the providers of a service, and the service ecosystem which includes the parties competing and collaborating with the service.

3.1 User perspective

The longer-term interest of a user to a service is based on the following characteristics:

- *User experience* – The emotional response that the use of a service evokes in the user. This emotional response is affected by the service itself, the context of its use, as well as the previous experiences and opinions of the properties of service and its provider (Sinkkonen, 2009). The user experience determines whether the user will want to come back and use a service again.
- *Ability to support variety of practices* – The ability of a service to be misused or used for other purposes that originally intended. The actual use of service combines the capabilities of the service with the needs and practices of user. The ability to adapt to these practices and to form a platform for the development of new shared practices can greatly expand the usage scenarios of the service.
- *Adaptation to the context of the user* – The service should be able to adapt to the social and physical context of the user. The social context of a user is composed of his or her family, relatives, friends, work mates, and so on. In many situation the service would naturally extend to some part of users social network: sharing photographs with friends, planning future events such as parties, managing chores within the family, organizing basketball exercises, and so on.

All services in a service ecosystem are not directly used by people. A service can belong to the infrastructure or act in supporting roles for other services. Examples deal with service discovery, automatic composition or substitution of services, routing of messages, monitoring of other services, and so on. The user perspective is not applicable for these services.

3.2 Ecosystem perspective

Services are increasingly part of service ecosystems where they use the capabilities of other services and offer their own capabilities to other services. Simultaneously they need compete with other similar services and face a pressure to evolve over the time. To succeed in an ecosystem it is an advantage, on the one hand, if a service is – as a good citizen – capable of supporting wide range of mutually beneficial collaborative relationships within the system. On the other hand, the service should be flexible and adaptive in the face of competitive pressures and to rapidly evolve over the time.

Both of the collaborative and competitive aspects are reflected in the principles of Flexible Services Architecture defined in the EDEN Deliverable 1.2:

- *Modularity* – Service is a clearly separable component of a service ecosystem and can be understood and developed independently.
- *Transparency* – The interfaces exposed by a service and the operations exposed in the interfaces are clear, well defined, and preferably machine-inspectable.
- *Loose coupling* – There are no unnecessary dependencies between services. While some of the dependencies are real and useful – for example, one service accessing the information about social network maintained in another service – in practice many of the dependencies are artificial and unnecessary; for example, the need to use same language or platform, particular API, synchronous communication when asynchronous would suffice, and so on.
- *Composability* – The services are recombinant and can be selected and assembled in different configurations to satisfy specific external requirements. Composability deals with the way connections between services are established and specified.
- *Late binding* – The creation of connections between services is postponed to run time, which makes it possible to use services available in particular situations, and make the binding decision with maximum information.
- *Reflection* – Services can inspect the structure and state of the system, and modify the composition and the state at run time through generic interfaces.

3.3 Provider perspective

It is essential to the long-term health of a service that no party necessary for its provision needs to give more than it receives. In general, this means that there should be a viable business model – or other incentive mechanism – for each party needed to support the service.

- *Existence of a business model* – The mechanism that motivates each party to continue the service provision. This should apply to all parties necessary for the operation of a service, including the parties that provide the supporting services.

4 Discussion

In this report we have surveyed the properties of services in order to provide a basis for determining the evaluation criteria for them. The properties are divided into short-term properties that are relevant in individual service deliveries, and to longer-term properties that affect the success of a service in the long run.

Short-term properties are classified into functionality, behavior, and non-functional properties. Functionality is the reason for a requestor to invoke a service; behavior relates to the manner in which service is delivered, and non-functional properties act as constraints on the functionality of the service. In this sense the functionality is the most essential aspect of a service. Unfortunately it is also the most difficult to represent, especially for services that deal with communication or entertainment. While non-functional properties like quality or security are secondary in their importance, they may gain most of the attention due to the difficulty to conceptualize the essential properties of services.

The longer-term properties concern the long-term relationships to a service from different parties of a service ecosystem. These were discussed from three different perspectives: user, provider, and ecosystem. When a service uses and is used by other services and simultaneously needs to compete from limited resources in a service ecosystem, many other properties in addition to the short-term ones need to be taken into account. These range from user experience and potential to adapt to alternative uses to loose coupling and existence of viable business models for service providers.

There is no one way to create evaluation criteria for services as a combination of the properties outlined above. The evaluation is always from a specific perspective and with specific goals. The understanding of the range of properties may help to avoid too narrow or one-sided approach.

5 Bibliography

M. Burstein, C. Bussler, T. Finin, M.N. Huhns, M. Paolucci, A.P. Sheth, S. Williams, and M. Zaremba. A semantic Web services architecture. *IEEE Internet Computing*, 9(5):72–81, 2005. URL: http://ebiquity.umbc.edu/file_directory/papers/208.pdf.

Simon Duff, James Harland, and John Thangarajah. On proactivity and maintenance goals. In *AAMAS '06: Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, pages 1033–1040, New York, NY, USA, 2006. ACM Press.

Cristina Feier and John Domingue. D3.1v0.1 WSMO Primer. DERI, 2005. URL: <http://www.wsmo.org/TR/d3/d3.1/v0.1/>.

Y. Gil and J. Blythe. How Can a Structured Representation of Capabilities Help in Planning. *Proceedings of the AAAI–Workshop on Representational Issues for Real-world Planning Systems*, 2000.

David Martin, Massimo Paolucci, Sheila McIlraith, Mark Burstein, Drew McDermott, Deborah McGuinness, Bijan Parsia, Terry Payne, Marta Sabou, Monika Solanki, Naveen Srinivasan, and Katia Sycara. Bringing Semantics to Web Services: The OWL-S Approach. In *Proceedings of the First International Workshop on Semantic Web Services and Web Process Composition (SWSWPC 2004)*, San Diego, California, USA, 2004. URL: <http://www.daml.org/services/owl-s/OWL-S-SWSWPC2004-CameraReady.doc>.

Sheila McIlraith, Tran Cao Son, and Honglei Zeng. Semantic Web Services. *IEEE Intelligent Systems*, 16(2):46–53, 2001. URL: <http://dx.doi.org/10.1109/5254.920599>.

Justin O’Sullivan, David Edmond, and Arthur Ter Hofstede. What’s in a Service? Towards accurate description of non-functional service properties. *Distributed and Parallel Databases*, 12(2-3):117–133, 2002. URL: <http://citeseer.ist.psu.edu/663154.html>.

Justin O’Sullivan, David Edmond, and Arthur Ter Hofstede. Formal description of non-functional service properties. Technical report. Queensland University of Technology, 2005. URL: <http://www.bpm.fit.qut.edu.au/about/docs/non-functional.jsp>

M.P. Singh and M.N. Huhns. *Service-oriented Computing: Semantics, Processes, Agents*. Wiley, West Sussex, England, 2005.

Irmeli Sinkkonen, Esko Nuutila, and Seppo Törmä. *Helppokäyttöisen verkkopalvelun suunnittelu*. Tietosanoma, 2009.

Ioan Toma and Douglas Foxvog. Non-functional properties in Web services. Technical report, DERI, 2006. URL: <http://www.wsmo.org/TR/d28/d28.4/v0.1/>.

M. Turner, D. Budgen, and P. Brereton. Turning software into a service. *Computer*, 36(10):38–44, 2003.

Seppo Törmä, Jukka Villstedt, Ville Lehtinen, Ian Oliver, Vesa Luukkala. *Semantic Web Services – A Survey*. Technical report TKK-TKO-B158, Helsinki University of Technology, 2008.

W3C. OWL-S: Semantic Markup for Web Services, 2004. URL: <http://www.w3.org/Submission/OWL-S/>.