

White Paper

Understanding Service Contract in a SO Ecosystem

WP0111 | October 2013



Michael Poulin

Michael Poulin is Head of Business and Technology Architecture at BuTechCon, a consulting firm based in the UK. He has built up a wealth of experience in business, enterprise and solutions architecture in both the UK and United States. His work focuses on bridging the gap between business architecture and modern technology.

Michael provides professional coaching and knowledge exchange and is a member of BCS and IASA.

Michael is actively engaged in the Enterprise Architecture realm having actively contributed in OASIS SOA standards. Michael has also authored various publications including "Ladder to SOE" and "Architects Know What Managers Don't", written multiple articles and blogs and presented at conferences.

Every one of us likes to be served, especially if we receive a good service. All businesses are created by people who have brought service into play as the fundamental element of any organization. An enterprise is not an exception – inside the enterprise we also service each other every day even though the servicing may be masqueraded by processes and procedures. According to the Value Network theory [1], people adhere to services in the work place and in an interpersonal relationship regardless of any processes and policies.

A consumer of a service is interested in only two things about the service: if the service has certain capabilities, and if the results of these capability executions suits the needs of the consumer. Every consumer may become a service provider at the same time. This is why the old saying "What goes around, comes around" has a literal meaning in the ecosystem of services or Service-Oriented (SO) Ecosystem [2]. If an enterprise not only works in a SO Ecosystem but also consciously preserves SO Ecosystem inside itself, we have a Service-Oriented Enterprise [3].

Access our **free**, extensive library at
www.orbussoftware.com/community

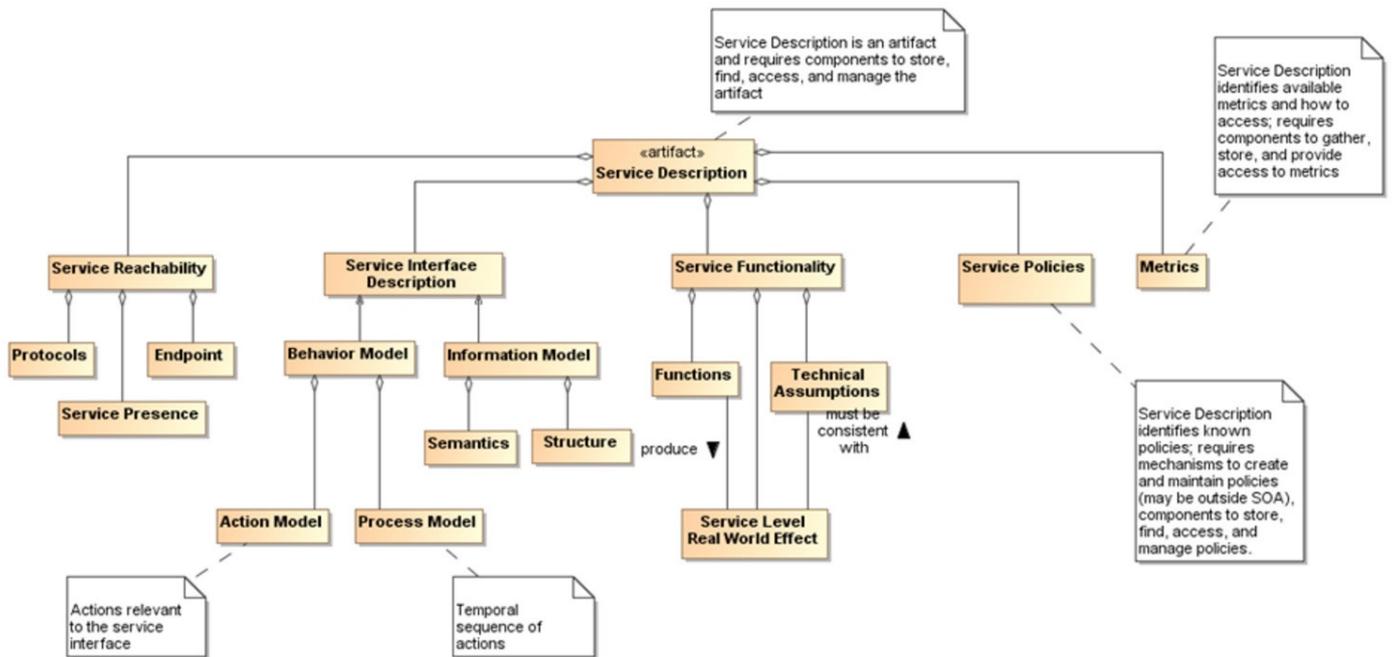


Figure 3. A Model of Service Description Defined in SOA RAF [2]

How Do We Know Which Service is Good for Us?

Every service consumer has common and specific requirements for their desired service. Consumers commonly ask whether the service performs particular business functionality and provides for certain results (RWE). Also, information about the provider (reputation) is important. In many cases, business law and regulations dictate the consumer's options in choosing providers. All these are the parts of the Service Description. The SOA RAF defines Service Description as it is shown in Figure 3.

A Service Description has to contain all information that a potential consumer might require to make a decision on whether a particular service is the one that can satisfy the consumer's needs. The major elements of Service Description are:

- Service Functionality
- Service Policies
- Description of Service's Interfaces
- Service Reachability
- Operational Metrics

While many major elements are relatively well known in the industry, the specifics of the distributed nature of SO Ecosystem, and especially its part covering enterprise business, make Service Policies a very distinct but mainly overlooked element; it represents so-called Business and Technical Execution Context (EC). In contrast with usual programmatic context related to the transaction or session and user's identity, EC deals with policies – rules, regulations and laws.

For instance, such policies may include:

- a) A rule that prohibits a service to work with clients deployed on the SW platform lower than a particular version
- b) A law that prohibits serving consumers from certain countries or whose names are on the Sanctions Lists
- c) A regulation that requires the use of a particular formula to calculate for an exposure of credit risk.

We deliberately mentioned policies from different categories to demonstrate that EC is much wider and more important than just a programmatic communication context. As an example, a USA regulation for mutual fund pricing is different from the corresponding UK regulation, i.e. if a client in the US uses a UK service for this price calculation, they may face problems with the American financial audit. Altogether, this means that the Service Contract has to be very clear about agreed EC – the same service can behave differently depending on the EC.

The industry is still uncertain on how to choose necessary service features. Here is a receipt: a Service Description may be used in service development as a source of requirements for design and implementation. As the reader can see, Service Description is consumer-centric and driven by an intent of satisfying the needs of a category of consumers; it is much more than a service interface. Besides available service interfaces, a potential consumer can find information about service functionality, metrics for SLA and policies in the Service Description. The latter represents the primary source for defining Service Contracts.

Service Contract in a SO Ecosystem

SOA RAF specifies: “When a consumer prepares to interact with a service, the consumer and the service provider must come to an agreement on the service features and characteristics that will be provided by the service and made available to the consumer. This agreement is known as a service contract.

Service Contract

An implicit or explicit documented agreement between the service consumer and service provider about the use of the service based on:

- the commitment by a service provider to provide service functionality and results consistent with identified real world effects and
- the commitment by a service consumer to interact with the service per specific means and per specified policies, where both consumer and provider actions are in the manner described in the service description” [1].

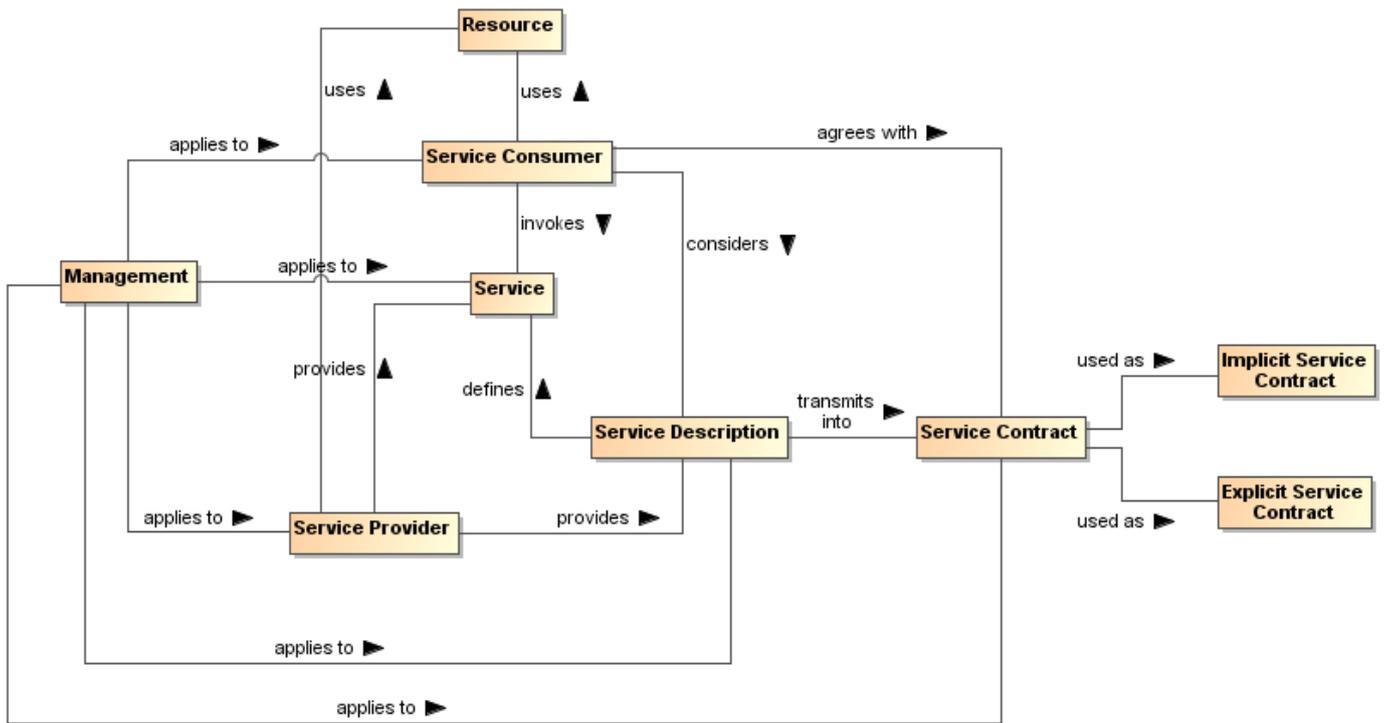


Figure 4. The Positioning of the Service Contract in the SO Ecosystem [2]

A Service Contract is derived from the Service Description. With regard to technology, a Service Contract incorporates all information about agreed programmatic interfaces of the service as well as all applicable policies of service invocation. A Service Contract defines all obligations – operational and executional – that the service accepts toward its consumer, as well as all limitations which the consumer accepts toward its rights and opportunities regarding the service. Figure 4 illustrates the relationship of a Service Contract with other elements of the SO Ecosystem.

The concept of a Service Contract in the SO ecosystem is the basis for all service interactions, and the proper management of Service Contracts is very important for any organization. If IT has intentions to become a partner and enabler of business, it has to refine its terminology and attribute a “service contract” to an agreement instead of to a connectivity interface as it is still exclusively used in technology. The scope of the use of technical “service contracts” is limited by development, while at the level of the Solution or Enterprise Architecture it starts either misleading or yielding the way up to the “relationship agreement”.

A particular programmatic interface of a service is not even a full contract between the programmatic modules of the consumer and service because these modules may have more than one connectivity interface. Moreover, policies do not have formally defined places in the service interfaces, while a notion of the execution context (expressed via policies) is extremely important for services and consumers. In other words, service execution context considers local policies of both consumer and

service, thus making the service outcome dependent on the context. If policies are omitted in the Service Contract, the consumer may be heading towards serious problems when using the service.

A Service Contract does not necessarily need to contain all information in its text; some information, like policy definitions and WSDL, may be referenced. One service may have several contracts with the same consumer, just as a consumer may have Service Contracts with several services.

Consequently a Service Contract includes, as a minimum, a description of all agreed service interfaces, related SLAs, service communication and operation policies and policies of the execution context. Service interfaces listed in the Service Contract are the only ones that a particular consumer may use for a given service, despite other interfaces that might be publicly available. A violation of the Service Contract can lead to a potential breakdown of the relationship between consumer and service up to a legal level.

How a Service Contract may be Used

SOA RAF identifies two types of Service Contracts:

1. Explicit Service Contracts that require certain negotiations between the consumer and the service before the agreement is reached and the service may be used
2. Implicit Service Contracts that are based on the consumer's acceptance with no negotiations of the service conditions and constraints.

Here is an example of a Service Contract: A Service Description offers three public interfaces A, B and C, and the use of each of them is associated with a certain fee. The service policy declares that if a consumer signs an explicit contract, the fee may be discounted. A consumer enters into negotiation with the service and they agree that this consumer will only use interface C. This results in some savings for the consumer while public interfaces A and B become irrelevant.

In another example, a service offers consumers the chance to choose among several SLAs for the same interface. This means that the service requires an explicit contract with everyone; there is not a default SLA bound to this interface. All explicit contracts are considered confidential. That is, an explicit Service Contract may not be referred to in another explicit Service Contract, even between the same consumer and service.

An implicit Service Contract is an agreement that assumes that a consumer accepts a service's constraints "as is". Usually, implicit Service Contracts appear in the form of public Service Descriptions though in some cases, an implicit Service Contract may be published separately. As such, implicit Service Contracts may be referred to in other implicit or explicit Service Contracts.

An example of an implicit Service Contract in the form of Service Description takes place when we shop in a store. When we book something or purchase goods, we agree with the store's rules such as merchandise price, return policy, opening hours and alike. A Service Contract defines obligations that the service and consumer take on themselves in regards to each other. No Service Contract, no obligation, and, therefore, no service.

If an enterprise consciously enters into the SO Ecosystem for its external and internal market, Service Contracts must have preconditions for all cases of service use [3]. Even if services are used in the same LOB or Business Unit, every service provider should know its obligations and every consumer should know where its rights for the service start and end. In many cases, when the service is created in a LOB, the same people develop both consumer and service implementation. At a glance this makes the creation of a Service Contract seem excessive. However, the life of service (if it is designed properly) can last for a long time because of an uncontrolled consumer base that requires a long-time service support period. This means that eventually the consumers and the services can end up under different business ownership and even the IT department/staff may be split accordingly, e.g. between the in-house and Cloud teams. If the time for creating a strong Service Contract at inception is missed, adding it later on becomes a difficult and expensive task [4].

While in the case of explicit contracts it is well defined who the consumers are and how to manage service versions and the service life-cycle, therefore the cases of implicit contracts can be more difficult to manage. This is why the best practice of service management recommends a mandatory registration of the service consumers with the service regardless of the type of Service Contract applied. In a shop, a payment check realizes the function of registration. This registration may be articulated as one of the policies for the service interaction, and consequently every consumer who utilizes the implicit Service Contract automatically agrees with said registration.

It is important to note that the Service Description may set mandatory aspects of a service that must be reflected in the Service Contract, such as security means and acceptable alternatives, financial regulations, or information access by 3rd Parties. The USA Patriot Act 2001 affects all American services offered in other countries, e.g., in the EU, where consumers usually prefer to avoid such services since they

allow a foreign (American) government to gain access to the personal information of EU citizens.

As an example of alternatives, the Service Description may identify a few versions of a terminology that may be recognized, and the specifics of the contract are satisfied when a consumer chooses one of the alternatives. At the same time, it is also acceptable if a consumer identifies a policy it requires to be preserved by the service, and the provider may be prepared to accept this policy as a part of the Service Contract.

Any form of explicit contract couples the service consumer and provider. While explicit contracts may be necessary or desirable in some cases such as supply chain management, a commerce often uses a mix of implicit and explicit contracts, and a service provider might offer (via Service Description) a conditional shift from implicit to explicit contract. For example, a company offers an implicit contract on the use of its APIs to any application with a limit on the amount of service invocations. If the application has to use more invocations, it has to enter into the explicit fee-based contract with the provider. Thus, the same service may have several unique Service Contracts with special SLAs for different consumers.

Service Contract, Interface and Relationship

A service, especially Business Service, may have as many interfaces as needed, e.g. one interface for each of the types of the service's customer base. Service interfaces are driven by the services, not the other way around. Service interfaces cannot exist without the service, i.e. without the service 'body'; otherwise, it is not a service interface. All available service interfaces are enumerated in the Service Description. Service Contracts may specify any sub-sets of the described service interfaces for the particular consumer. For example, a Business Service may expose a programmatic interface in a form of a Web Service. However, if we deal with just a Web Service, we cannot conclude if it represents a service or acts as a standardized integration between two applications, none of which is a service. A Web Service does not make an application a service.

Let's assume we have a service provider who announces a public Business Service with an explicit Service Contract only. There may be many reasons for doing this (vs. an implicit contract). For instance, a service provider wants to stay in control over its consumer base and explicitly manage its relationships with each consumer regarding replacement of service versions over time. One of the consequences of an existence of a Service Contract is that the service is dealing only with the consumers who agree with it, explicitly or implicitly.

This leads to the so-called Knight Rules of Ownership in the SO Ecosystem [5] that the services follow in their relationships:

- A Consumer of my Consumer is not my Consumer
- A Service of my Server is not my Service
- A Partner of my Partner is not my Partner
- A Supplier of my Supplier is not my Supplier.

Thus, Service Contracts define the boundaries in the service relationships. The significance of such boundaries is high.

Let's consider a popular security solution such as the propagation of an end-user's identity through the chained invocations of different applications, or modules, or components to the final "working" element. IT security and audit originally used this propagation as a means of tracing the requester in order to verify its rights to access each component on the way. Some even created tokens that contained access rights of particular users regarding each component in the environment. Apparently, this model works only if all those components share the same security realm.

However, if each component has its independent ownership, the existence of a single security realm is not guaranteed. SOA RAF says, "In a SOA ecosystem there may not be any single person or organization that is really 'in control' or 'in charge' of the whole" [2]. For example, company A works with a Cloud service provider and serves its own consumer, company B. All of them have their own security realms. The Cloud service provider delivers a special software system and has a contract with company A. This Cloud service provider knows nothing about B. That is, if company A and the Cloud service provider resolve security issues among them and the Cloud service serves company A, the identities of company B's users have no meaning to the Cloud service (recall, "A Consumer of my Consumer, is not my Consumer"). Even if company A propagates a B-user's identity to the Cloud service, the B-user will have no access rights in the Cloud service and the latter will simply ignore this information.

SO Ecosystem assumes that each service is responsible for setting a contractual trust with its consumers. This trust is enough for a service to perform an action upon the request from the trusted consumer regardless of whose behalf this request was issued. No Service Contract, no service.

Thus, when we work within services, do we really need to propagate the end-user identity? This is the example where contractual business relationships between consumers and services require the changing of existing technology practice [4] and Service Contracts play one of the fundamental roles in this reformation.

Conclusion

This white paper introduces and discusses the notion of a Service Contract as it is defined in the OASIS SOA RAF specification. Following this specification, we reviewed motivations that lead to the interactions between the consumer and the service. We have also formalized an introduction of the service to the potential consumer via a Service Description. The latter is the source for deriving a Service Contract that regulates relationships and physical connectivity between the service and its consumer.

A Service Contract in its implicit and explicit forms is the glue that connects otherwise isolated services. A service may have many Service Contracts with regular consumers or with other services that appear as consumers or suppliers. This white paper reflects on the service-specific aspect of an independent ownership of services (the Knight Rules) and points to the ways in which Service Contracts demarcate ownership and inter-obligatory boundaries. Overall, Service Contracts represent a mechanism of formalizing co-operation and collaboration between services and consumers. This mechanism can heavily impact existing accustomed technical methods that may become inefficient in the SO Ecosystem, thus calling for new solutions.

Resources

- [1] Allee, V. 2011 “A Baker’s Dozen Principles of Value Networks” [online] January. Available at: <<http://internettime.posterous.com/a-bakers-dozen-principles-of-value-networks-v>>
- [2] OASIS Reference Architecture Foundation for Service Oriented Architecture Version 1.0 Committee Specification 01 December 2012 Available at: <<http://docs.oasis-open.org/soa-rm/soa-ra/v1.0/soa-ra.html>>
- [3] Poulin, M. 2009 “Ladder to SOE” in How to Create Resourceful and Efficient Solutions for Market Changes within Business and Technology [ebook] Troubador Publishing Ltd. Available at: <<http://www.mpoulin.com/ladder-to-soe/>>
- [4] Poulin, M. 2013 “Architects Know What Managers Don’t” in Business Architecture for Dynamic Market [ebook] BuTechCon Ltd. - Troubador Publishing Ltd. Available at: <<http://www.mpoulin.com/architects-know-what-manager-dont/>>
- [5] Poulin, M. 2012 “Knight Rules of Ownership in Service-Oriented Ecosystem” eBizQ [blog] Available at: <http://www.ebizq.net/blogs/service_oriented/2012/06/knight_rules_of_ownership_in_service-oriented_ecosystem.php>

© Copyright 2013 Orbus Software. All rights reserved.

No part of this publication may be reproduced, resold, stored in a retrieval system, or distributed in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior permission of the copyright owner.

Such requests for permission or any other comments relating to the material contained in this document may be submitted to: marketing@orbussoftware.com

Orbus Software

3rd Floor
111 Buckingham Palace Road
London
SW1W 0SR
United Kingdom

+44 (0) 870 991 1851
enquiries@orbussoftware.com
www.orbussoftware.com

