



Reference Architecture Foundation for Service Oriented Architecture Version 1.0

Committee Specification 01

04 December 2012

Specification URIs

This version:

<http://docs.oasis-open.org/soa-rm/soa-ra/v1.0/cs01/soa-ra-v1.0-cs01.pdf> (Authoritative)
<http://docs.oasis-open.org/soa-rm/soa-ra/v1.0/cs01/soa-ra-v1.0-cs01.html>
<http://docs.oasis-open.org/soa-rm/soa-ra/v1.0/cs01/soa-ra-v1.0-cs01.doc>

Previous version:

<https://www.oasis-open.org/committees/download.php/46922/soa-ra-v1.0-csprd03.zip>

Latest version:

<http://docs.oasis-open.org/soa-rm/soa-ra/v1.0/soa-ra.pdf> (Authoritative)
<http://docs.oasis-open.org/soa-rm/soa-ra/v1.0/soa-ra.html>
<http://docs.oasis-open.org/soa-rm/soa-ra/v1.0/soa-ra.doc>

Technical Committee:

OASIS Service Oriented Architecture Reference Model TC

Chair:

Ken Laskey (klaskey@mitre.org), MITRE Corporation

Editors:

Peter Brown (peter@peterfbrown.com), Individual Member
Jeff A. Estefan (jeffrey.a.estefan@jpl.nasa.gov), Jet Propulsion Laboratory
Ken Laskey (klaskey@mitre.org), MITRE Corporation
Francis G. McCabe (fmccabe@gmail.com), Individual Member
Danny Thornton (danny.thornton@ngc.com), Northrop Grumman

Related work:

This specification is related to:

- *Reference Model for Service Oriented Architecture 1.0*. 12 October 2006. OASIS Standard.
<http://docs.oasis-open.org/soa-rm/v1.0/soa-rm.html>.

Abstract:

This document specifies the OASIS Reference Architecture Foundation for Service Oriented Architecture (SOA-RAF). It follows from the concepts and relationships defined in the OASIS Reference Model for Service Oriented Architecture as well as work conducted in other organizations. While it remains abstract in nature, the current document describes the foundation upon which specific SOA concrete architectures can be built.

The focus of the SOA-RAF is on an approach to integrating business with the information technology needed to support it. These issues are always present but are all the more important when business integration involves crossing ownership boundaries.

The SOA-RAF follows the recommended practice of describing architecture in terms of models, views, and viewpoints, as prescribed in the ANSI/IEEE 1471-2000.

It has three main views: the *Participation in a SOA Ecosystem* view which focuses on the way that participants are part of a Service Oriented Architecture ecosystem; the *Realization of a SOA Ecosystem* view which addresses the requirements for constructing a SOA-based system in a SOA ecosystem; and the *Ownership in a SOA Ecosystem* view which focuses on what is meant to own a SOA-based system.

The SOA-RAF is of value to Enterprise Architects, Business and IT Architects as well as CIOs and other senior executives involved in strategic business and IT planning.

Status:

This document was last revised or approved by the OASIS Service Oriented Architecture Reference Model TC on the above date. The level of approval is also listed above. Check the "Latest version" location noted above for possible later revisions of this document.

Technical Committee members should send comments on this specification to the Technical Committee's email list. Others should send comments to the Technical Committee by using the "Send A Comment" button on the Technical Committee's web page at <http://www.oasis-open.org/committees/soa-rm/>.

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the Technical Committee web page (<https://www.oasis-open.org/committees/soa-rm/ipr.php>).

Citation format:

When referencing this specification the following citation format should be used:

[SOA-RAF]

Reference Architecture Foundation for Service Oriented Architecture Version 1.0. 04 December 2012. OASIS Committee Specification 01.

<http://docs.oasis-open.org/soa-rm/soa-ra/v1.0/cs01/soa-ra-v1.0-cs01.html>.

Notices

Copyright © OASIS Open 2012. All Rights Reserved.

All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property Rights Policy (the "OASIS IPR Policy"). The full [Policy](#) may be found at the OASIS website.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to OASIS, except as needed for the purpose of developing any document or deliverable produced by an OASIS Technical Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must be followed) or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

OASIS requests that any OASIS Party or any other party that believes it has patent claims that would necessarily be infringed by implementations of this OASIS Committee Specification or OASIS Standard, to notify OASIS TC Administrator and provide an indication of its willingness to grant patent licenses to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification.

OASIS invites any party to contact the OASIS TC Administrator if it is aware of a claim of ownership of any patent claims that would necessarily be infringed by implementations of this specification by a patent holder that is not willing to provide a license to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification. OASIS may include such claims on its website, but disclaims any obligation to do so.

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS' procedures with respect to rights in any document or deliverable produced by an OASIS Technical Committee can be found on the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this OASIS Committee Specification or OASIS Standard, can be obtained from the OASIS TC Administrator. OASIS makes no representation that any information or list of intellectual property rights will at any time be complete, or that any claims in such list are, in fact, Essential Claims.

The name "OASIS" is a trademark of [OASIS](#), the owner and developer of this specification, and should be used only to refer to the organization and its official outputs. OASIS welcomes reference to, and implementation and use of, specifications, while reserving the right to enforce its marks against misleading uses. Please see <http://www.oasis-open.org/policies-guidelines/trademark> for above guidance.

Table of Contents

1	Introduction.....	9
1.1	Context for Reference Architecture for SOA	9
1.1.1	What is a Reference Architecture?	9
1.1.2	What is this Reference Architecture?	10
1.1.3	Relationship to the OASIS Reference Model for SOA	10
1.1.4	Relationship to other Reference Architectures.....	10
1.1.5	Expectations set by this Reference Architecture Foundation	11
1.2	Service Oriented Architecture – An Ecosystems Perspective	11
1.3	Viewpoints, Views and Models	11
1.3.1	ANSI/IEEE 1471-2000 and ISO/IEC/IEEE 42010:2011	11
1.3.2	UML Modeling Notation.....	13
1.4	SOA-RAF Viewpoints	13
1.4.1	Participation in a SOA Ecosystem Viewpoint.....	14
1.4.2	Realization of a SOA Ecosystem Viewpoint.....	14
1.4.3	Ownership in a SOA Ecosystem Viewpoint	14
1.5	Terminology	14
1.6	References.....	15
1.6.1	Normative References.....	15
1.6.2	Non-Normative References.....	15
2	Architectural Goals and Principles	17
2.1	Goals and Critical Success Factors of the Reference Architecture Foundation	17
2.1.1	Goals	17
2.1.2	Critical Success Factors.....	18
2.2	Principles of this Reference Architecture Foundation	18
3	Participation in a SOA Ecosystem View.....	20
3.1	SOA Ecosystem Model.....	21
3.2	Social Structure in a SOA Ecosystem Model	22
3.2.1	Stakeholders, Participants, Actors and Delegates	24
3.2.2	Social Structures and Roles	26
3.2.3	Needs, Requirements and Capabilities.....	29
3.2.4	Resource and Ownership.....	31
3.2.5	Establishing Execution Context.....	32
3.3	Action in a SOA Ecosystem Model.....	36
3.3.1	Services Reflecting Business	37
3.3.2	Activity, Action, and Joint Action	38
3.3.3	State and Shared State	40
3.4	Architectural Implications.....	40
3.4.1	Social structures.....	40
3.4.2	Resource and Ownership.....	40
3.4.3	Policies and Contracts.....	41
3.4.4	Semantics.....	41
3.4.5	Trust and Risk	41
3.4.6	Needs, Requirements and Capabilities.....	41

3.4.7	The Importance of Action	41
4	Realization of a SOA Ecosystem view	43
4.1	Service Description Model	43
4.1.1	The Model for Service Description	44
4.1.2	Use of Service Description	52
4.1.3	Relationship to Other Description Models	57
4.1.4	Architectural Implications	58
4.2	Service Visibility Model	59
4.2.1	Visibility to Business	60
4.2.2	Visibility	60
4.2.3	Architectural Implications	64
4.3	Interacting with Services Model	64
4.3.1	Interaction Dependencies	64
4.3.2	Actions and Events	65
4.3.3	Message Exchange	66
4.3.4	Composition of Services	68
4.3.5	Implementing Service Composition	69
4.3.6	Architectural Implications of Interacting with Services	72
4.4	Policies and Contracts Model	73
4.4.1	Policy and Contract Representation	73
4.4.2	Policy and Contract Enforcement	74
4.4.3	Architectural Implications	75
5	Ownership in a SOA Ecosystem View	76
5.1	Governance Model	76
5.1.1	Understanding Governance	76
5.1.2	A Generic Model for Governance	78
5.1.3	Governance Applied to SOA	82
5.1.4	Architectural Implications of SOA Governance	85
5.2	Security Model	86
5.2.1	Secure Interaction Concepts	87
5.2.2	Where SOA Security is Different	89
5.2.3	Security Threats	89
5.2.4	Security Responses	90
5.2.5	Access Control	92
5.2.6	Architectural Implications of SOA Security	95
5.3	Management Model	95
5.3.1	Management	95
5.3.2	Management Means and Relationships	99
5.3.3	Management and Governance	100
5.3.4	Management and Contracts	100
5.3.5	Management for Monitoring and Reporting	104
5.3.6	Management for Infrastructure	104
5.3.7	Architectural Implication of SOA Management	105
5.4	SOA Testing Model	105
5.4.1	Traditional Software Testing as Basis for SOA Testing	105

5.4.2 Testing and the SOA Ecosystem	106
5.4.3 Elements of SOA Testing	107
5.4.4 Testing SOA Services	109
5.4.5 Architectural Implications for SOA Testing.....	110
6 Conformance	112
6.1 Conformance Targets	112
6.2 Conformance and Architectural Implications	112
6.3 Conformance Summary.....	112
Appendix A. Acknowledgements	113
Appendix B. Index of Defined Terms.....	114
Appendix C. Relationship to other SOA Open Standards	115
C.1 Navigating the SOA Open Standards Landscape Around Architecture	115
C.2 The Service-Aware Interoperability Framework: Canonical.....	116
C.3 IEEE Reference Architecture	117
C.4 RM-ODP	117

Table of Figures

Figure 1 - Model elements described in the Participation in a SOA Ecosystem view	20
Figure 2 - SOA Ecosystem Model.....	21
Figure 3 - Social Structure Model	23
Figure 4 – Stakeholders, Actors, Participants and Delegates	25
Figure 5 - Social Structures, Roles and Action	27
Figure 6 - Roles in a Service.....	29
Figure 7 - Cycle of Needs, Requirements, and Fulfillment	30
Figure 8 - Resources.....	31
Figure 9 - Willingness and Trust	33
Figure 10 – Policies, Contracts and Constraints.....	34
Figure 11: An Activity, expressed informally as a graph of Actions	38
Figure 12: Activity involving Actions across an ownership boundary	39
Figure 13 - Model Elements Described in the Realization of a SOA Ecosystem view	43
Figure 14 - General Description.....	45
Figure 15 - Representation of a Description	46
Figure 16 - Service Description.....	48
Figure 17 - Service Interface Description.....	49
Figure 18 - Service Functionality	50
Figure 19 - Model for Policies and Contracts as related to Service Participants.....	51
Figure 20 - Policies and Contracts, Metrics, and Compliance Records	52
Figure 21 - Relationship between Action and Components of Service Description Model	53
Figure 22 - Execution Context.....	56
Figure 23 - Interaction Description.....	57
Figure 24 - Visibility to Business	60
Figure 25 - Awareness in a SOA Ecosystem.....	62
Figure 26 - Service Reachability	63
Figure 27 - Interaction dependencies	65
Figure 28 - A 'message' denotes either an action or an event.....	65
Figure 29 - Fundamental SOA message exchange patterns (MEPs).....	67
Figure 30 - Simple model of service composition	69
Figure 31 - Abstract example of a simple business process exposed as a service	70
Figure 32 - Abstract example of a more complex composition that relies on collaboration	71
Figure 33 - Policies and Contracts.....	73
Figure 34 - Model Elements Described in the Ownership in a SOA Ecosystem View	76
Figure 35 - Motivating Governance.....	78
Figure 36 - Setting Up Governance	79
Figure 37 - Carrying Out Governance.....	80
Figure 38 - Ensuring Governance Compliance.....	81
Figure 39 - Relationship Among Types of Governance.....	83
Figure 40 - Authorization.....	88

Figure 41 - Management model in SOA ecosystem 97
Figure 42 - Management Means and Relationships in a SOA ecosystem 99
Figure 43 - Management of the service interaction 102
Figure 44 - SOA Reference Architecture Positioning 116

1 Introduction

Service Oriented Architecture (SOA) is an architectural paradigm that has gained significant attention within the information technology (IT) and business communities. The SOA ecosystem described in this document bridges the area between business and IT. It is neither wholly IT nor wholly business, but is of both worlds. Neither business nor IT completely own, govern and manage this SOA ecosystem. Both sets of concerns must be accommodated for the SOA ecosystem to fulfill its purposes.¹

The OASIS Reference Model for SOA [**SOA-RM**] provides a common language for understanding the important features of SOA but does not address the issues involved in constructing, using or owning a SOA-based system. This document focuses on these aspects of SOA.

The intended audiences of this document and expected benefits to be realized include non-exhaustively:

- Enterprise Architects - will gain a better understanding when planning and designing enterprise systems of the principles that underlie Service Oriented Architecture;
- Standards Architects and Analysts - will be able to better position specific specifications in relation to each other in order to support the goals of SOA;
- Decision Makers - will be better informed as to the technology and resource implications of commissioning and living with a SOA-based system; in particular, the implications following from multiple ownership domains; and
- Stakeholders/Developers - will gain a better understanding of what is involved in participating in a SOA-based system.

1.1 Context for Reference Architecture for SOA

1.1.1 What is a Reference Architecture?

A reference architecture models the abstract architectural elements in the domain of interest independent of the technologies, protocols, and products that are used to implement a specific solution for the domain. It differs from a reference model in that a reference model describes the important concepts and relationships in the domain focusing on what distinguishes the elements of the domain; a reference architecture elaborates further on the model to show a more complete picture that includes showing what is involved in realizing the modeled entities, while staying independent of any particular solution but instead applies to a class of solutions.

It is possible to define reference architectures at many levels of detail or abstraction, and for many different purposes. A reference architecture is not a concrete architecture; i.e., depending on the requirements being addressed by the reference architecture, it generally will not completely specify all the technologies, components and their relationships in sufficient detail to enable direct implementation.

¹ By *business* we refer to any activity that people are engaged in. We do not restrict the scope of SOA ecosystems to commercial applications.

33 1.1.2 What is this Reference Architecture?

34 There is a continuum of architectures, from the most abstract to the most detailed. As a Committee, we
35 have liaised and worked with other groups and organizations working in this space to ensure that our
36 efforts overlap as little as possible. We look at some of these other works in Appendix C. The result is that
37 this Reference Architecture is an abstract realization of SOA, focusing on the elements and their
38 relationships needed to enable SOA-based systems to be used, realized and owned while avoiding
39 reliance on specific concrete technologies. This positions the work at the more abstract end of the
40 continuum, and constitutes what is described in [TOGAF v9] as a 'foundation architecture'. It is
41 nonetheless a *reference* architecture as it remains solution-independent and is therefore characterized as
42 a *Reference Architecture Foundation* because it takes a first principles approach to architectural modeling
43 of SOA-based systems.

44 While requirements are addressed more fully in Section 2, the SOA-RAF makes key assumptions that
45 SOA-based systems involve:

- 46 • use of resources that are distributed across ownership boundaries;
- 47 • people and systems interacting with each other, also across ownership boundaries;
- 48 • security, management and governance that are similarly distributed across ownership
49 boundaries; and
- 50 • interaction between people and systems that is primarily through the exchange of messages with
51 reliability that is appropriate for the intended uses and purposes.

52 Even in apparently homogenous structures, such as within a single organization, different groups and
53 departments nonetheless often have ownership boundaries between them. This reflects organizational
54 reality as well as the real motivations and desires of the people running those organizations.

55 Such an environment as described above is an *ecosystem* and, specifically in the context of SOA-based
56 systems, is a **SOA ecosystem**. This concept of an ecosystem perspective of SOA is elaborated further in
57 Section 1.2.

58 This SOA-RAF shows how Service Oriented Architecture fits into the life of actors and stakeholders, how
59 SOA-based systems may be realized effectively, and what is involved in owning and managing them.
60 This serves two purposes: to ensure that SOA-based systems take account of the specific constraints of
61 a SOA ecosystem, and to allow the audience to focus on the high-level issues without becoming over-
62 burdened with details of a particular implementation technology.

63 1.1.3 Relationship to the OASIS Reference Model for SOA

64 The OASIS Reference Model for Service Oriented Architecture identifies the key characteristics of SOA
65 and defines many of the important concepts needed to understand what SOA is and what makes it
66 important. The Reference Architecture Foundation takes the Reference Model as its starting point, in
67 particular the vocabulary and definition of important terms and concepts.

68 The SOA-RAF goes further in that it shows how SOA-based systems can be realized – albeit in an
69 abstract way. As noted above, SOA-based systems are better thought of as dynamic systems rather than
70 stand-alone software products. Consequently, how they are used and managed is at least as important
71 architecturally as how they are constructed.

72 1.1.4 Relationship to other Reference Architectures

73 Other SOA reference architectures have emerged in the industry, both from the analyst community and
74 the vendor/solution provider community. Some of these reference architectures are quite abstract in
75 relation to specific implementation technologies, while others are based on a solution or technology stack.
76 Still others use middleware technology such as an Enterprise Service Bus (ESB) as their architectural
77 foundation.

78 As with the Reference Model, this Reference Architecture is primarily focused on large-scale distributed
79 IT systems where the participants may be legally separate entities. It is quite possible for many aspects of
80 this Reference Architecture to be realized on quite different platforms.

81 In addition, this Reference Architecture Foundation, as the title illustrates, is intended to provide
82 foundational models on which to build other reference architectures and eventual concrete architectures.

83 The relationship to several other industry reference architectures for SOA and related SOA open
84 standards is described in Appendix C.

85 **1.1.5 Expectations set by this Reference Architecture Foundation**

86 This Reference Architecture Foundation is not a complete blueprint for realizing SOA-based systems. Nor
87 is it a technology map identifying all the technologies needed to realize SOA-based systems. It does
88 identify many of the key aspects and components that will be present in any well designed SOA-based
89 system. In order to actually use, construct and manage SOA-based systems, many additional design
90 decisions and technology choices will need to be made.

91 **1.2 Service Oriented Architecture – An Ecosystems Perspective**

92 Many systems cannot be completely understood by a simple decomposition into parts and subsystems –
93 in particular when many autonomous parts of the system are governing interactions. We need also to
94 understand the context within which the system functions and the participants involved in making it
95 function. This is the **ecosystem**. For example, a biological ecosystem is a self-sustaining and dynamic
96 association of plants, animals, and the physical environment in which they live. Understanding an
97 ecosystem often requires a holistic perspective that considers the relationships between the elements of
98 the system and their environment at least as important as the individual parts of the system.

99 This Reference Architecture Foundation views the SOA architectural paradigm from an ecosystems
100 perspective: whereas a system will be a **capability** developed to fulfill a defined set of needs, a **SOA**
101 **ecosystem** is a space in which people, processes and machines act together to deliver those capabilities
102 as services.

103 Viewed as whole, a SOA ecosystem is a network of discrete processes and machines that, together with
104 a community of people, creates, uses, and governs specific services as well as external suppliers of
105 resources required by those services.

106 In a SOA ecosystem there may not be any single person or organization that is really ‘in control’ or ‘in
107 charge’ of the whole although there are identifiable stakeholders who have influence within the
108 community and control over aspects of the overall system.

109 The three key principles that inform our approach to a SOA ecosystem are:

- 110 • a SOA is a paradigm for *exchange of value* between independently acting *participants*;
- 111 • participants (and stakeholders in general) have legitimate claims to *ownership* of resources that
112 are made available within the SOA ecosystem; and
- 113 • the behavior and performance of the participants are subject to *rules of engagement* which are
114 captured in a series of policies and contracts.

115 **1.3 Viewpoints, Views and Models**

116 **1.3.1 ANSI/IEEE 1471-2000 and ISO/IEC/IEEE 42010:2011**

117 The SOA-RAF structures its analysis based on the concepts defined in IEEE “Recommended Practice for
118 Architectural Description of Software-Intensive Systems” [ANSI/IEEE 1471]. ANSI/IEEE 1471 was later
119 approved as ISO/IEC 42010-2007 and subsequently superseded by ISO/IEC/IEEE 42010:2011
120 [ISO/IEC/IEEE 42010]. Although the more recent standard modifies some of its original definitions and
121 introduces new material, the modifications and additions were not found to significantly impact the SOA-
122 RAF analysis. As such, the SOA-RAF follows the definitions and structure of the original standard.

123 An architectural description conforming to [ANSI/IEEE 1471] and [ISO/IEC/IEEE 42010] must include the
124 following six (6) elements:

- 125 1. Architectural description identification, version, and overview information
- 126 2. Identification of the system stakeholders and their concerns judged to be relevant to the
127 architecture
- 128 3. Specifications of each viewpoint that has been selected to organize the representation of the
129 architecture and the rationale for those selections
- 130 4. One or more architectural views

- 131 5. A record of all known inconsistencies among the architectural description's required constituents
132 6. A rationale for selection of the architecture (in particular, showing how the architecture supports
133 the identified stakeholders' concerns).

134 The standard defines the following terms²:

135 **Architecture**

136 The fundamental organization of a system embodied in its components, their relationships to
137 each other, and to the environment, and the principles guiding its design and evolution.

138 **Architectural Description**

139 A collection of products that document the architecture.

140 **System**

141 A collection of components organized to accomplish a specific function or set of functions.

142 **System Stakeholder**

143 An individual, team, or organization (or classes thereof) with interests in, or concerns relative to, a
144 system.

145 A stakeholder's concern should not be confused with either a need or a formal requirement. A concern,
146 as understood here, is an area or topic of interest. Within that concern, system stakeholders may have
147 many different requirements. In other words, something that is of interest or importance is not the same
148 as something that is obligatory or of necessity [TOGAF v9].

149 When describing architectures, it is important to identify stakeholder concerns and associate them with
150 viewpoints to insure that those concerns are addressed in some manner by the models that comprise the
151 views on the architecture. The standard defines views and viewpoints as follows:

152 **View**

153 A representation of the whole system from the perspective of a related set of concerns.

154 **Viewpoint**

155 A specification of the conventions for constructing and using a view. A pattern or template from
156 which to develop individual views by establishing the purposes and audience for a view and the
157 techniques for its creation and analysis.

158 In other words, a view is what the stakeholders see whereas the viewpoint defines the perspective from
159 which the view is taken and the methods for, and constraints upon, modeling that view.

160 It is important to note that viewpoints are independent of a particular system (or solutions). In this way,
161 the architect can select a set of candidate viewpoints first, or create new viewpoints, and then use those
162 viewpoints to construct specific views that will be used to organize the architectural description. A view,
163 on the other hand, is specific to a particular system. Therefore, the practice of creating an architectural
164 description involves first selecting the viewpoints and then using those viewpoints to construct specific
165 views for a particular system or subsystem. Note that the standard requires that each view corresponds to
166 exactly one viewpoint. This helps maintain consistency among architectural views which is a normative
167 requirement of the standard.

168 A view is comprised of one or more architectural models, where model is defined as:

² See <http://www.iso-architecture.org/42010/cm/> for a diagram of the updated standard's Conceptual Framework

169 **Model**

170 An abstraction or representation of some aspect of a thing (in this case, a system)

171 All architectural models used in a particular view are developed using the methods established by the
172 architectural viewpoint associated with that view. An architectural model may participate in more than one
173 view but a view must conform to a single viewpoint.

174 **1.3.2 UML Modeling Notation**

175 An open standard modeling language is used to help visualize structural and behavioral architectural
176 concepts. Although many architecture description languages exist, we have adopted the Unified Modeling
177 Language™ 2 (UML® 2) [UML 2] as the main viewpoint modeling language. Normative UML is used
178 unless otherwise stated but it should be noted that it can only partially describe the concepts in each
179 model – it is important to read the text in order to gain a more complete understanding of the concepts
180 being described in each section.

181 The UML presented should not be treated blindly or automatically: the models are intended to formalize
182 the concepts and relationships defined and described in the text but the nature of the RAF means that it
183 still concerns an abstract layer rather than an implementable layer.

184 **1.4 SOA-RAF Viewpoints**

185 The SOA-RAF specifies three views (described in detail in Sections 3, 4, and 5) that conform to three
186 viewpoints: *Participation in a SOA Ecosystem*, *Realization of a SOA Ecosystem*, and *Ownership in a SOA*
187 *Ecosystem*. There is a one-to-one correspondence between viewpoints and views (see Table 1).

Viewpoint Element	Viewpoint		
	Participation in a SOA Ecosystem	Realization of a SOA Ecosystem	Ownership in a SOA Ecosystem
Main concepts covered	Captures what is meant for people to participate in a SOA ecosystem.	Captures what is meant to realize a SOA-based system in a SOA ecosystem.	Captures what is meant to own a SOA-based system in a SOA ecosystem
Stakeholders addressed	All participants in the SOA ecosystem	Those involved in the design, development and deployment of SOA-based systems	Those involved in governing, managing, securing, and testing SOA-based systems
Concerns addressed	Understanding ecosystem constraints and contexts in which business can be conducted predictably and effectively.	Effective construction of SOA-based systems.	Processes to ensure governance, management, security, and testing of SOA-based systems.
Modeling Techniques used	UML class diagrams	UML class, sequence, component, activity, communication, and composite structure diagrams	UML class and communication diagrams

188
189

Table 1 - Viewpoint specifications for the OASIS Reference Architecture Foundation for SOA

190 **1.4.1 Participation in a SOA Ecosystem Viewpoint**

191 This viewpoint captures a SOA ecosystem as an environment for people to conduct their business. We do
192 not limit the applicability of such an ecosystem to commercial and enterprise systems. We use the term
193 business to include any transactional activity between multiple participants.

194 All stakeholders in the ecosystem have concerns addressed by this viewpoint. The primary concern for
195 people is to ensure that they can conduct their business effectively and safely in accordance with the
196 SOA paradigm. The primary concern of decision makers is the relationships between people and
197 organizations using systems for which they, as decision makers, are responsible but which they may not
198 entirely own, and for which they may not own all of the components of the system.

199 Given SOA's value in allowing people to access, manage and provide services across ownership
200 boundaries, we must explicitly identify those boundaries and the implications of crossing them.

201 **1.4.2 Realization of a SOA Ecosystem Viewpoint**

202 This viewpoint focuses on the infrastructure elements that are needed to support the construction of SOA-
203 based systems. From this viewpoint, we are concerned with the application of well-understood
204 technologies available to system architects to realize the SOA vision of managing systems and services
205 that cross ownership boundaries.

206 The stakeholders are essentially anyone involved in designing, constructing and deploying a SOA-based
207 system.

208 **1.4.3 Ownership in a SOA Ecosystem Viewpoint**

209 This viewpoint addresses the concerns involved in owning and managing SOA-based systems within the
210 SOA ecosystem. Many of these concerns are not easily addressed by automation; instead, they often
211 involve people-oriented processes such as governance bodies.

212 Owning a SOA-based system implies being able to manage an evolving system. It involves playing an
213 active role in a wider ecosystem. This viewpoint is concerned with how systems are managed effectively,
214 how decisions are made and promulgated to the required end points; how to ensure that people may use
215 the system effectively; and how the system can be protected against, and recover from consequences of,
216 malicious intent.

217 **1.5 Terminology**

218 The keywords "MUST", "MUST NOT", "REQUIRED" (and by extension, "REQUIRES"), "SHALL", "SHALL
219 NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are
220 to be interpreted as described in **[RFC2119]**.

221 References are surrounded with [square brackets and are in bold text].

222 The terms "SOA-RAF", "this Reference Architecture" and "Reference Architecture Foundation" refer to
223 this document, while "the Reference Model" and "SOA-RM" refer to the OASIS Reference Model for
224 Service Oriented Architecture. **[SOA-RM]**.

225 **Usage of Terms**

226 Certain terms are used in this document (in sections 3 to 6) to denote concepts that are formally defined
227 here and intended to be used with the specific meanings indicated. Where mention is first made of a
228 formally defined concept, or the term is used within the definition of another concept, we use a **bold font**.
229 When this occurrence appears in the text substantially in advance of the formal definition, it is also
230 **hyperlinked** to the definition in the body of the text. A list of all such terms is included in the [Index of](#)
231 [Terms at Appendix B](#).

232 1.6 References

233 1.6.1 Normative References

- 234 [ANSI/IEEE 1471] *IEEE Recommended Practice for Architectural Description of Software-Intensive*
235 *Systems*, American National Standards Institute/Institute for Electrical and
236 Electronics Engineers, September 21, 2000.
- 237 [ISO/IEC 10746-2] *Information Technology – Open Distributed Processing – Reference Model:*
238 *Foundations*, International Organization for Standardization and International
239 Electromechanical Commission, 1999 (Also published as ITU-T
240 recommendation X.902)
- 241 [ISO/IEC IS 19793] *Information Technology – Open Distributed Processing – Use of UML for ODP*
242 *System Specification*, International Organization for Standardization and
243 International Electromechanical Commission, 2008 (Also published as ITU-T
244 recommendation X.906).
- 245 [RFC 2119] *Key words for use in RFCs to Indicate Requirement Levels*, S. Bradner, IETF
246 RFC 2119, March 1997, <http://www.ietf.org/rfc/rfc2119.txt>
- 247 [SOA-RM] *Reference Model for Service Oriented Architecture 1.0*, OASIS Standard,
248 12 October 2006. <http://docs.oasis-open.org/soa-rm/v1.0/soa-rm.pdf>
- 249 [UML 2] *Unified Modeling Language: Superstructure*, Ver. 2.1.1, OMG Adopted
250 Specification, OMG document formal/2007-02-05, Object Management Group,
251 Needham, MA, February 5, 2007.

252 1.6.2 Non-Normative References

- 253 [DCMI] Dublin Core Metadata Initiative, <http://dublincore.org>.
- 254 [HOTLE] *SOA Governance – What You Need to Know*, Matt Hotle, Gartner, 2010
- 255 [IEEE 829] *IEEE Standard for Software Test Documentation*, Institute for Electrical and
256 Electronics Engineers, 16 September 1998
- 257 [ISO 11179] *Information Technology -- Metadata registries (MDR)*, ISO/IEC 11179,
258 <http://metadata-standards.org/11179/>
- 259 [ISO/IEC 27002] *Information technology -- Security techniques – Code of practice for information*
260 *security management*, International Organization for Standardization and
261 International Electrotechnical Commission, 2007
- 262 [ISO/IEC/IEEE 42010] *Systems and software engineering – Architecture description*, 1 December
263 2011.
- 264 [LININGTON] *Building Enterprise Systems with ODP*, Peter Linington, Zoran Milosevic, Akira
265 Tanaka, Antonio Vallecillo, Chapman & Hall / CRC, 2012
- 266 [NEWCOMER/LOMOW] *Understanding SOA with Web Services*, Eric Newcomer and Greg Lomow,
267 Addison-Wesley: Upper Saddle River, NJ, 2005.
- 268 [SMITH] *Mitigating Risks Associated with Transitive Trust in Service Based Identity*
269 *Propagation*, K. Smith, Information Security Journal: A Global Perspective, 21:2,
270 71-78, April 2012)
- 271 [SOA NAV] *Navigating the SOA Open Standards Landscape Around Architecture*,
272 Heather Kreger and Jeff Estefan (Eds.), Joint Paper, The Open Group, OASIS,
273 and OMG, July 2009. [http://www.oasis-](http://www.oasis-open.org/committees/download.php/32911/wp_soa_harmonize_d1.pdf)
274 [open.org/committees/download.php/32911/wp_soa_harmonize_d1.pdf](http://www.oasis-open.org/committees/download.php/32911/wp_soa_harmonize_d1.pdf)
- 275 [TOGAF v9] *The Open Group Architecture Framework (TOGAF)*, Version 9 Enterprise
276 Edition, The Open Group, Doc Number: G091, February 2009.
- 277 [WEILL] *IT Governance: How Top Performers Manage IT Decision Rights for Superior*
278 *Results*, Peter Weill and Jeanne W. Ross, Harvard Business School Press,
279 2004
- 280 [WSA] *Web Services Architecture*, David Booth, et al., W3C Working Group Note,
281 World Wide Web Consortium (W3C) (Massachusetts Institute of Technology,

282
283
284

European Research Consortium for Informatics and Mathematics, Keio University), February, 2004. <http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/>

285 2 Architectural Goals and Principles

286 This section identifies the goals of this Reference Architecture Foundation and the architectural principles
287 that underpin it.

288 2.1 Goals and Critical Success Factors of the Reference Architecture 289 Foundation

290 There are three principal goals:

- 291 1. to show how SOA-based systems can effectively bring participants with needs ('consumers') to
292 interact with participants offering appropriate capabilities as services ('producers');
- 293 2. for participants to have a clearly understood level of confidence as they interact using SOA-based
294 systems; and
- 295 3. for SOA-based systems to be scaled for small or large systems as needed.

296 There are four factors critical to the achievement of these goals:

- 297 1. **Action:** an account of participants' action within the ecosystem;
- 298 2. **Trust:** an account of how participants' internal perceptions of the reliability of others guide their
299 behavior (i.e., the trust that participants may or may not have in others)
- 300 3. **Interaction:** an account of how participants can interact with each other; and
- 301 4. **Control:** an account of how the management and governance of the entire SOA ecosystem can
302 be arranged.

303 These goals and success factors are expanded in the following subsections.

304 2.1.1 Goals

305 2.1.1.1 Effectiveness

306 A primary purpose of the SOA-RAF is to show how SOA-based systems ensure that participants can use
307 the facilities of the system to meet their needs. This does not imply that every need has a SOA solution,
308 but for those needs that can benefit, we look at what is needed to use the SOA paradigm effectively.

309 The key factors that govern effectiveness from a participant's perspective are actions undertaken—
310 especially across ownership boundaries — with other participants in the ecosystem and which lead to
311 measurable results.

312 2.1.1.2 Confidence

313 SOA-based systems should enable service providers and consumers to conduct their business with the
314 appropriate level of confidence in the interaction. Confidence is especially important in situations that are
315 high-risk; this includes situations involving multiple ownership domains as well as situations involving the
316 use of sensitive resources.

317 Confidence has many dimensions: confidence in the successful interactions with other participants,
318 confidence in the assessment of trust, as well as confidence that the ecosystem is properly managed.

319 2.1.1.3 Scalability

320 The third goal of this reference architecture is scalability. In architectural terms, we determine scalability in
321 terms of the smooth growth of complex systems as the number and complexity of services and
322 interactions between participants increases. Another measure of scalability is the ease with which
323 interactions can cross ownership boundaries.

324 2.1.2 Critical Success Factors

325 A critical success factor (CSF) is a property of the intended system, or a sub-goal that directly supports a
326 goal and there is strong belief that without it the goal is unattainable. CSFs are not necessarily
327 measurable in themselves. CSFs can be associated with more than one goal.

328 In many cases, critical success factors are often denoted by adjectives: reliability, trustworthiness, and so
329 on. In our analysis of the SOA paradigm, however, it seems more natural to identify four critical concepts
330 (nouns) that characterize important aspects of SOA:

331 2.1.2.1 Action

332 Participants' principal mode of participation in a SOA ecosystem is action; typically action in the interest of
333 achieving some desired **real world effect**. Understanding how action is related to SOA is thus critical to
334 the paradigm.

335 2.1.2.2 Trust

336 The viability of a SOA ecosystem depends on participants being able to effectively measure the
337 trustworthiness of the system and of participants. Trust is a private assessment of a participant's belief in
338 the integrity and reliability of the SOA ecosystem (see Section 3.2.5.1).

339 Trust can be analyzed in terms of trust in infrastructure facilities (otherwise known as reliability), trust in
340 the relationships and effects that are realized by interactions with services, and trust in the integrity and
341 confidentiality of those interactions particularly with respect to external factors (otherwise known as
342 security).

343 Note that there is a distinction between trust in a SOA-based system and trust in the capabilities
344 accessed via the SOA-based system. The former focuses on the role of SOA-based systems as a
345 *medium* for conducting business, the latter on the trustworthiness of participants in such systems. This
346 architecture focuses on the former, while trying to encourage the latter.

347 2.1.2.3 Interaction

348 In order for a SOA ecosystem to function, it is essential that the means for participants to interact with
349 each other is available throughout the system. Interaction encompasses not only the mechanics and
350 semantics of **communication** but also the means for discovering and offering communication.

351 2.1.2.4 Control

352 Given that a large-scale SOA-based system may be populated with many services, and used by large
353 numbers of people; managing SOA-based systems properly is a critical factor for engendering confidence
354 in them. This involves both managing the services themselves and managing the relationships between
355 people and the SOA-based systems they are utilizing; the latter being more commonly identified with
356 governance.

357 The governance of SOA-based systems requires decision makers to be able to set policies about
358 participants, services, and their relationships. It requires an ability to ensure that policies are effectively
359 described and enforced. It also requires an effective means of measuring the historical and current
360 performances of services and participants.

361 The scope of management of SOA-based systems is constrained by the existence of multiple ownership
362 domains.

363 2.2 Principles of this Reference Architecture Foundation

364 The following principles serve as core tenets that guided the evolution of this reference architecture.

365 Technology Neutrality

366 Statement: Technology neutrality refers to independence from particular technologies.

367 Rationale: We view technology independence as important for three main reasons: technology
368 specific approach risks confusing issues that are technology specific with those that are
369 integrally involved with realizing SOA-based systems; and we believe that the principles
370 that underlie SOA-based systems have the potential to outlive any specific technologies
371 that are used to deliver them. Finally, a great proportion of this architecture is inherently
372 concerned with people, their relationships to services on SOA-based systems and to
373 each other.

374 Implications: The Reference Architecture Foundation must be technology neutral, meaning that we
375 assume that technology will continue to evolve, and that over the lifetime of this
376 architecture that multiple, potentially competing technologies will co-exist. Another
377 immediate implication of technology independence is that greater effort is needed on the
378 part of architects and other decision makers to construct systems based on this
379 architecture.

380 Parsimony

381 Statement: Parsimony refers to economy of design, avoiding complexity where possible and
382 minimizing the number of components and relationships needed.

383 Rationale: The hallmark of good design is parsimony, or “less is better.” It promotes better
384 understandability or comprehension of a domain of discourse by avoiding gratuitous
385 complexity, while being sufficiently rich to meet requirements.

386 Implications: Parsimoniously designed systems tend to have fewer but better targeted features.

387 Distinction of Concerns

388 Statement: Distinction of Concerns refers to the ability to cleanly identify and separate out the
389 concerns of specific stakeholders in such a way that it is possible to create architectural
390 models that reflect those stakeholders’ viewpoint. In this way, an individual stakeholder or
391 a set of stakeholders that share common concerns only see those models that directly
392 address their respective areas of interest.

393 Rationale: As SOA-based systems become more mainstream and increasingly complex, it will be
394 important for the architecture to be able to scale. Trying to maintain a single, monolithic
395 architecture description that incorporates all models to address all possible system
396 stakeholders and their associated concerns will not only rapidly become unmanageable
397 with rising system complexity, but it will become unusable as well.

398 Implications: This is a core tenet that drives this reference architecture to adopt the notion of
399 architectural viewpoints and corresponding views. A viewpoint provides the formalization
400 of the groupings of models representing one set of concerns relative to an architecture,
401 while a view is the actual representation of a particular system. The ability to leverage an
402 industry standard that formalizes this notion of architectural viewpoints and views helps
403 us better ground these concepts for not only the developers of this reference architecture
404 but also for its readers. The IEEE Recommended Practice for Architectural Description of
405 Software-Intensive Systems [ANSI/IEEE 1471] is the standard that serves as the basis
406 for the structure and organization of this document.

407 Applicability

408 Statement: Applicability refers to that which is relevant. Here, an architecture is sought that is
409 relevant to as many facets and applications of SOA-based systems as possible; even
410 those yet unforeseen.

411 Rationale: An architecture that is not relevant to its domain of discourse will not be adopted and thus
412 likely to languish.

413 Implications: The Reference Architecture Foundation needs to be relevant to the problem of matching
414 needs and capabilities under disparate domains of ownership; to the concepts of ‘Intranet
415 SOA’ (SOA within the enterprise) as well as ‘Internet SOA’ (SOA outside the enterprise);
416 to the concept of ‘Extranet SOA’ (SOA within the extended enterprise, i.e., SOA with
417 suppliers and trading partners); and finally, to ‘net-centric SOA’ or ‘Internet-ready SOA.’

3 Participation in a SOA Ecosystem View

No man is an island

*No man is an island entire of itself; every man
is a piece of the continent, a part of the main;
if a clod be washed away by the sea, Europe
is the less, as well as if a promontory were, as
well as any manner of thy friends or of thine
own were; any man's death diminishes me,
because I am involved in mankind.
And therefore never send to know for whom
the bell tolls; it tolls for thee.*

John Donne

The *Participation in a SOA Ecosystem* view in the SOA-RAF focuses on the constraints and context in which people conduct business using a SOA-based system. By business we mean any shared activity whose objective is to satisfy particular **needs** of each participant. To effectively employ the SOA paradigm, the architecture must take into account the fact and implications of different **ownership** domains, and how best to organize and utilize capabilities that are distributed across those different ownership domains. These are the main architectural issues that the *Participation in a SOA Ecosystem* view tries to address.

The subsections below expand on the abstract Reference Model by identifying more fully and with more specificity what challenges need to be addressed in order to successfully apply the SOA paradigm. Although this view does not provide a specific recipe, it does identify the important things that need to be considered and resolved within an ecosystem context.

The main models in this view are:

- the **SOA Ecosystem Model** introduces the main relationships between the social structure and the SOA-based System, as well as the key role played by the hybrid concept of participant in both.
- the **Social Structure in a SOA Ecosystem Model** introduces the key elements that underlie the relationships between participants and that must be considered as pre-conditions in order to effectively bring needs and capabilities together across **ownership boundaries**;
- the **Action in a SOA Ecosystem Model** introduces the key concepts involved in service **actions**, and shows how **joint action** and **real-world effect** are the target outcomes that motivate interacting in a SOA ecosystem.

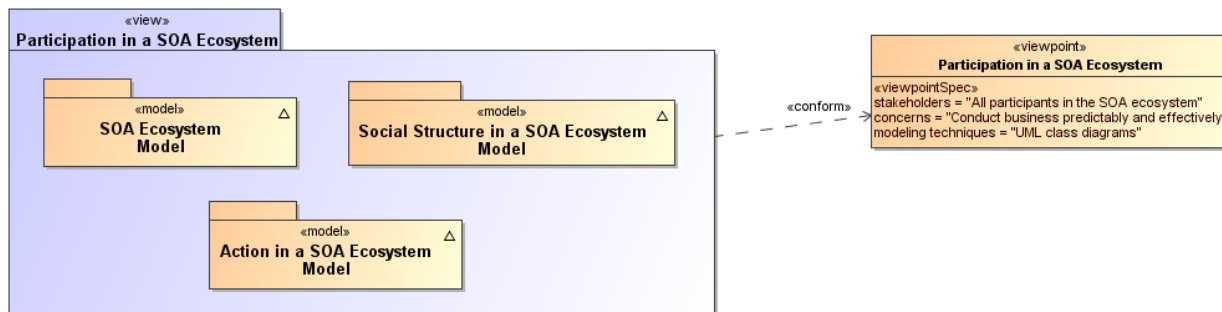


Figure 1 - Model elements described in the *Participation in a SOA Ecosystem* view

Furthermore, this *Participation in a SOA Ecosystem* view helps us understand the importance of execution context – the set of technical and business elements that allow interaction to occur in, and thus business to be conducted using, a SOA-based system.

The dominant mode of **communication** within a SOA ecosystem is electronic, supported by IT resources and artifacts. The **stakeholders** (see next section) are nonetheless people: since there is inherent indirection involved when people and systems interact using electronic means, we lay the foundations for

459 how *communication* can be used to represent and enable action. However, it is important to understand
460 that these communications are usually a means to an end and not the primary interest of the participants
461 of the ecosystem.

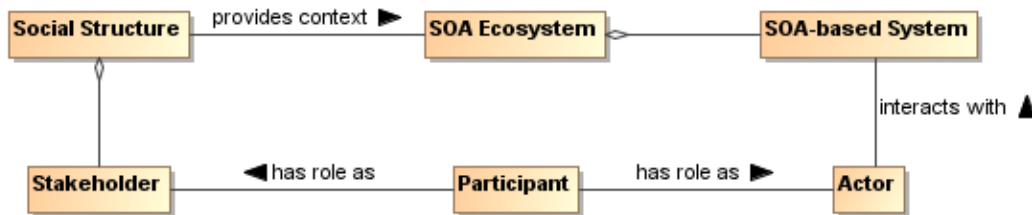
462 3.1 SOA Ecosystem Model

463 The OASIS SOA Reference Model defines *Service Oriented Architecture* (SOA) as “a paradigm for
464 organizing and utilizing distributed capabilities that may be **under the control of different ownership**
465 **domains**” (our emphasis) and *services* as “the mechanism by which needs and capabilities are brought
466 together”. The central focus of SOA is “the task or business function – getting something done.”

467 Together, these ideas describe an environment in which business functions (realized in the form of
468 services) address business needs. Service implementations utilize capabilities to produce specific (real
469 world) effects that fulfill those business needs. Both those using the services, and the capabilities
470 themselves, may be distributed across ownership domains, with different **policies** and conditions of use
471 in force – this environment is referred to as a **SOA Ecosystem** and is modeled in Figure 2.

472 The role of a service in a SOA Ecosystem is to enable effective **business solutions** in this environment.
473 Any technology system created to deliver a service in such an environment is referred to as a **SOA-**
474 **based system**. SOA is thus a paradigm that guides the identification, design, implementation (i.e.,
475 organization), and utilization of such services. SOA-based systems act as technology-based proxies for
476 activity that would otherwise be carried out within and between social structures.

477 A SOA-based system is concerned with how **actors** interact within a system to deliver a specific result -
478 the delivery of a real world effect. The SOA ecosystem is concerned with all potential stakeholders and
479 the roles that they can play; how some stakeholders’ needs are satisfied by other stakeholders’ solutions;
480 how stakeholders assess **risk**; how they relate to each other through policies and **contracts**; and how
481 they communicate and establish relationships of **trust** in the processes leading to the delivery of a
482 specific result.



483
484 Figure 2 - SOA Ecosystem Model

485 SOA Ecosystem

486 An environment encompassing one or more **social structure(s)** and **SOA-based system(s)** that
487 interact together to enable effective **business solutions**

488 SOA-based System

489 A technology system created to deliver a service within a **SOA Ecosystem**

490 Social Structures are defined and described in more detail in the next model, shown in Figure 3.

491 **Stakeholders, Actors, and Participants** are formally defined in Section 3.2.1.

492 Participants (as stakeholders and as actors), SOA-based systems, and the environment (or context)
493 within which they all operate, taken together forms the SOA ecosystem. Participants (or their **delegates**)
494 interact with a SOA-based system - in the role of actors - and are also members of a social structure - in
495 the role of stakeholders. Here we explicitly note that stakeholders and, thus, participants are people³
496 because machines alone cannot truly have a stake in the outcomes of a social structure. Delegates may
497 be human and nonhuman but are not directly stakeholders. Stakeholders, both Participants and **Non-**
498 **participants**, may potentially benefit from the services delivered by the SOA-based system. Again, this is
499 discussed more fully in Section 3.2.1.

500 The SOA ecosystem may reflect the SOA-based activities within a particular enterprise or of a wider
501 network of one or more enterprises and individuals; these are modeled in and discussed with respect to
502 Figure 3. Although a SOA-based system is essentially an IT concern, it is nonetheless a system
503 engineered deliberately to be able to function in a SOA ecosystem. In this context, a service is the
504 mechanism that brings a SOA-based system **capability** together with stakeholder needs in the wider
505 ecosystem.

506 Several interdependent concerns are important in our view of a SOA ecosystem. The ecosystem includes
507 stakeholders who are participants in the development, deployment, **governance** and use of a system and
508 its services; or who may not participate in certain activities but are nonetheless affected by the system.
509 Actors – whether stakeholder **participants** or delegates who act only on behalf of participants (without
510 themselves having any stake in the actions that they have been tasked to perform) – are engaged in
511 **actions** which have an impact on the real world and whose meaning and intent are determined by implied
512 or agreed-to semantics. This is discussed further in relation to the model in Figure 4 and elaborated more
513 fully in Section 3.3.

514 3.2 Social Structure in a SOA Ecosystem Model

515 The Social Structure Model explains the relationships between stakeholders and the social context in
516 which they operate, within and between distinct boundaries. It is also the foundation for understanding
517 **security**, governance and management in the SOA ecosystem.

518 Actions undertaken by people (whether natural or legal persons) are performed in a *social context* that
519 defines the relationships between them. That context is provided by **social structures** existing in society
520 and the roles played by each person as stakeholders in those structures.

521 Whether informal peer groups, communities of practice, associations, enterprises, corporations,
522 government agencies, or entire nations, these structures interact with each other in the world, using
523 treaties, contracts, market rules, handshakes, negotiations and – when necessary – have recourse to
524 arbitration and legislation. They interact because there is a mutual benefit in doing so: one has something
525 that the other can provide. They interact across defined or implicit **ownership boundaries** that define the
526 limits of one structure (and the limits of its **authority**, responsibilities, capabilities, etc.) and the beginning
527 of another.

528 Social structures, together with their **constitution**, their stakeholders, their mission and goals, need
529 therefore to be understood when examining the role that technology plays. Technology systems play an
530 increasing role in carrying out many of the functions performed by such structures and therefore model
531 real-world procedures. The technology systems serve as proxies in digital space for these real-world
532 structures and procedures. The SOA paradigm is particularly concerned with designing, configuring and

³ 'People' and 'person' must be understood as both humans and 'legal persons', such as companies, who have **rights** and **responsibilities** similar to 'natural persons' (humans)

533 managing such systems across ownership boundaries precisely because this mirrors the real-world
 534 interactions between discrete structures and across their ownership boundaries.

535 A stakeholder in a social structure will be involved in many 'actions' that do not involve a SOA-based
 536 system. Although such actions and the roles relating to them are outside the scope of this Reference
 537 Architecture Foundation, they may nonetheless result in constraining or otherwise impacting a given SOA
 538 ecosystem – for example, a new item of legislation that regulates service interactions. The terms **Actor**
 539 and **Action** used throughout the document refer thus only to SOA-based systems.

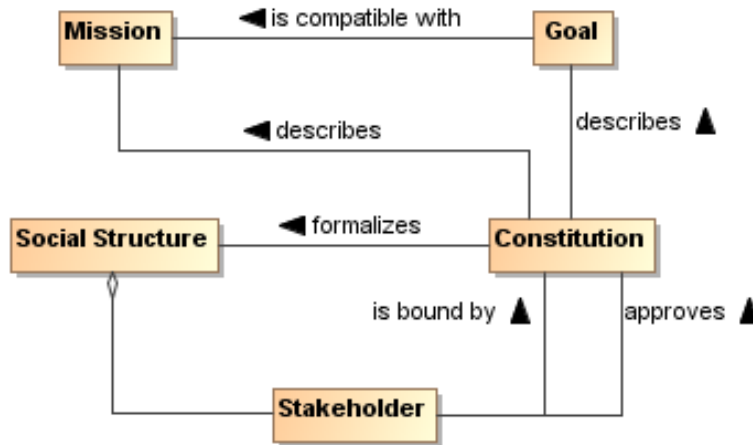


Figure 3 - Social Structure Model

540
 541
 542 **Social Structure**

543 A nexus of relationships amongst people brought together for a specific purpose.

544 The social structure is established with an implied or explicitly defined mission, usually reflected in the
 545 goals laid down in the social structure's constitution or other 'charter'. Although goals are often expressed
 546 in terms of general ambitions for the social structure's work or of desired end states, objectives are
 547 expressed more formally in terms of specific, measurable, and achievable action required to realize those
 548 states. Action in the context of a social structure is discussed in Section 3.3.

549 A social structure may involve any number of persons as stakeholders and a large number of different
 550 relationships may exist among them. The organizing principle for these relationships is the social
 551 structure's mission. Any given person can be a stakeholder in multiple social structures and a social
 552 structure itself can be a stakeholder in its own right as part of a larger one or in another social structure
 553 entirely. These multiple roles can result in disagreements, particularly when the mission or goals of
 554 different social structures do not align.

555 A social structure can take different forms. An enterprise is a common kind of social structure with its
 556 distinct legal personality; an online community group might represent a social structure of peers that is
 557 very loose, albeit with a shared mission. A market represents a social structure of buyers and sellers.
 558 Legislation in different geo-political areas (from local and regional to national or global) provides a
 559 framework in which social structures can operate.

560 A social structure will further its goals in one of two ways:

- 561 • by acting alone, using its own **resources**;
- 562 • interacting with other structures and using their resources.

563 Many interactions take place within social structures. Some interactions may or may not cross ownership
 564 boundaries depending on the scale and internal organization of the structure (an enterprise, for example,
 565 can itself be composed of sub-enterprises). Our focus is on interactions *between* social structures,
 566 particularly as they determine the way that technology systems need to interact. Systems that are
 567 designed to do this are SOA-based systems.

568 The nature and extent of the interactions that take place will reflect, often implicitly, degrees of trust
 569 between people and the very specific circumstances of each person at the time, and over the course of

570 their interactions. It is in the nature of a SOA ecosystem that these relationships are rendered more
571 explicit and are formalized as a central part of what the **[SOA-RM]** refers to as Execution Context.

572 The validity of the interactions between social structures is not always clear and is often determined
573 ultimately by relevant legislation. For example, when a customer buys a book over the Internet, the
574 validity of the transaction may be determined by the place of incorporation of the book vendor, the
575 residence of the buyer, or a combination of both. Such legal jurisdiction qualification is typically buried in
576 the fine print of the service description.

577 **Constitution**

578 A set of **rules**, written or unwritten, that formalize the mission, goals, scope, and functioning of a
579 **social structure**.

580 Every social structure functions according to **rules** by which people interact with each other within the
581 structure. In some cases, this is based on an explicit agreement; in other cases, participants behave as
582 though they agree to the constitution without a formal agreement. In still other cases, participants abide
583 by the rules with some degree of reluctance. In all cases, the constitution may change over time; in those
584 cases of implicit agreement, the change can occur quickly. Section 5.1 contains a detailed discussion of
585 governance and SOA.

586 **3.2.1 Stakeholders, Participants, Actors and Delegates**

587 A social structure represents the interests of a collection of people who have **rights** and **responsibilities**
588 within the structure. People have a 'stake' in such a social structure, and when that social structure is part
589 of a SOA Ecosystem, the people continue to interact through their roles as stakeholders. In addition,
590 people – either directly or through their delegates - interact with SOA-based (technology) systems. Here,
591 the people interact through their roles as actors interacting with specific system-level activity.

592 A person who participates in a social structure as a stakeholder *and* interacts with a SOA-based system
593 as an actor is defined as an ecosystem **Participant**. The concept of participant is particularly important as
594 it reflects a hybrid role of a Stakeholder concerned with expressing needs and seeing those needs fulfilled
595 *and* an Actor directly involved with system-level activity that result in necessary effects.

596 The hybrid role of Participant provides a bridge between social structures within the wider (real-world)
597 ecosystem – in particular the world of the stakeholder – and the more specific (usually technology-
598 focused) system – the world of the actor.

599 The concept of the ecosystem therefore embraces all aspects of the 'real world', human-centered, social
600 structures that are concerned with business interactions together with the technology-centered SOA-
601 based system that deliver services:

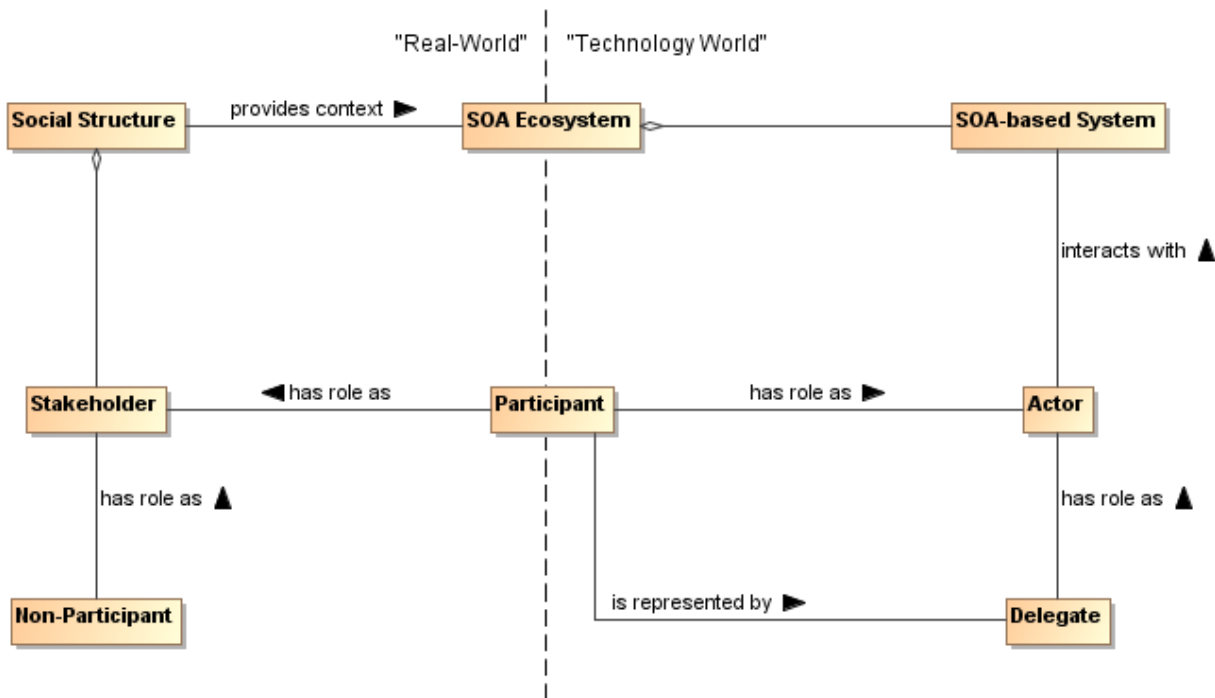


Figure 4 – Stakeholders, Actors, Participants and Delegates

602
603
604

Stakeholder

A person with an interest (a 'stake') in a **social structure**.

606 Not all stakeholders necessarily participate in all activities in the SOA ecosystem; indeed, the interest of
607 non-participant stakeholders may be to realize the benefits of a well-functioning ecosystem and not suffer
608 unwanted consequences. Non-participant stakeholders cannot all or always be identified in advance but
609 due account is often taken of such stakeholder types, including potential customers, beneficiaries, and
610 other affected third parties. A stakeholder may be a participant with respect to some activities and a non-
611 participant with respect to others.

Actor

613 A role played either by a **Participant** or its **Delegate** and that interacts with a **SOA-based**
614 **system**.

Participant

616 A person who plays a role *both* in the **SOA ecosystem** as a **stakeholder** *and* with the **SOA-**
617 **based system** as an **actor** either

- 618 • directly, in the case of a human participant; or
- 619 • indirectly, via a **delegate**.

620 Not all participants are necessarily benign to the social structure: such 'negative stakeholders' might
621 deliberately seek a negative impact on the ecosystem (such as hackers or criminals) and social structures
622 will work to ensure that they are not able to operate as welcome participants.

Non-Participant

624 A person who plays no role as a **participant** in a **social structure's** activities but nonetheless
625 has an interest in, or is affected by, such activities.

Delegate

627 A role played by a human or an automated or semi-automated agent and acting on behalf of a
628 **participant** but not directly sharing the participant's stake in the outcome.

629 Many actors interact with a SOA-based system, including software agents that permit people to offer, and
630 interact with, services; delegates that represent the interests of other participants; or security agents
631 charged with managing the security of the ecosystem. Note that automated agents are *always* delegates,
632 in that they act on behalf of a participant.

633 In the different models of the SOA-RAF, the term actor is used when action is being considered at the
634 level of the SOA-based system and when it is not relevant who is carrying out the action. However, if the
635 actor is acting explicitly *on behalf of* a participant, then we use the term delegate. This underlines the
636 importance of delegation in SOA-based systems, whether the delegation is of work procedures carried
637 out by human agents who have no stake in the actions with which they are tasked but act on behalf of a
638 participant who does; or whether the delegation is performed by technology (automation). On the other
639 hand, if it is important to emphasize that when the actor is also a stakeholder in the ecosystem, then we
640 use the term participant. This also underlines the pivotal role played by a participant, in a unique position
641 between the social structure and the SOA-based system, in the broader ecosystem.

642 The difference between a participant and a delegate is that a delegate acts on behalf of a participant and
643 must have the authority to do so. Because of this, every social structure must clearly define the roles
644 assigned to actors (whether participants or delegates) in carrying out activity within its domain.

645 **3.2.2 Social Structures and Roles**

646 Social structures are abstractions: they cannot directly perform actions with SOA-based systems – only
647 actors can, whether they be participants acting under their own volition or delegates (human or not)
648 simply following the instructions of participants. An actor advances the objectives of a social structure
649 through its interaction with SOA-based systems, influencing actions that deliver results. The specifics of
650 the interaction depend on the roles defined by the social structure that the actor may assume or have
651 conferred and the nature of the relationships between the stakeholders concerned. These relationships
652 can introduce constraints on an actor when engaged in an action. These points are illustrated in Figure 5.

653 A role is not immutable and is often time-bound. An actor can have one or more roles concurrently and
654 may change them over time and in different contexts, even over the course of a particular interaction.

655 **3.2.2.1 Authority, Rights, and Responsibilities**

656 One participant with appropriate authority in the social structure may formally designate a role for a
657 delegate or another participant, with associated rights and responsibilities, and that authority may even
658 qualify a period during which the designated role may be valid. In addition, while many roles are clearly
659 identified, with appropriate names and definitions of responsibilities, it is also possible to separately
660 bestow rights, bestow or assume responsibilities and so on, often in a temporary fashion. For example,
661 when a company president delegates certain responsibilities on another person, this does not imply that
662 the other person has become company president. Likewise, a company president may bestow on
663 someone else her role during a period of time that she is on vacation or otherwise unreachable with the
664 understanding that she will re-assume the role when she returns from vacation.

665 Conversely, someone who exhibits qualification and skill may assume a role without any formal
666 designation. For example, an office administrator who has demonstrated facility with personal computers
667 may be known as (and thus assume to role of) the ‘go to’ person for people who need help with their
668 computers.

669 The social structure is responsible for establishing the authority by which actors carry out actions in line
670 with defined constraints:

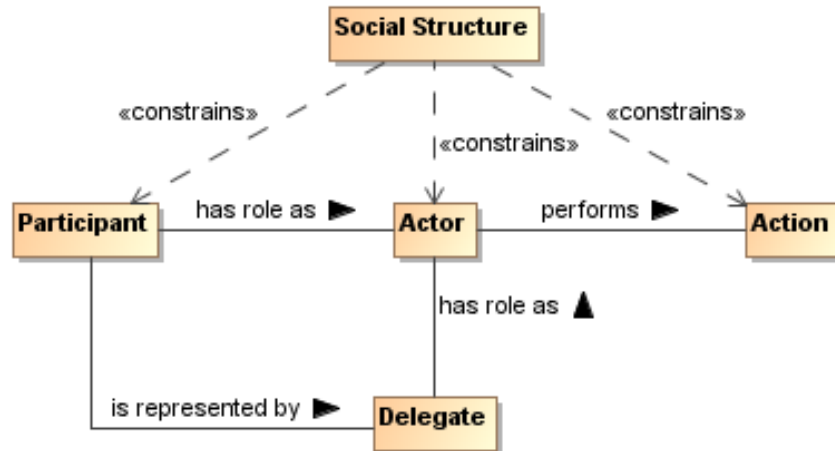


Figure 5 - Social Structures, Roles and Action

671
672

673 Authority

674 A **right** conferred on a **participant** to ensure that **actions** are carried out consistent with the
675 objectives of a **social structure**.

676 Actions are carried out by actors, either participants themselves or delegates acting on their behalf, by
677 interacting with the SOA-based system.

678 Right

679 A predetermined **permission** conferred upon an **actor** to perform some **action** or assume a role
680 in relation to the **social structure**.

681 Rights can be constrained. For example, sellers might have a general right to refuse service to potential
682 customers but this right could be constrained so as to be exercised only when certain criteria are met.

683 Responsibility

684 A predetermined **obligation** on a **participant** to ensure that some **action** is performed or assume
685 a role in relation to other **participants**.

686 Responsibility implies human agency and thus aligns with participants and potentially human delegates
687 but not with non-human delegates. This applies even if the consequences of such responsibility can
688 impact other (human and non-human) actors. Having authority often implies having responsibility.

689 Rights, authorities, responsibilities and roles form the foundation for the security model as well as
690 contributing to the governance model in the **Ownership in a SOA Ecosystem** View of the SOA-RAF.

691 3.2.2.2 Permissions and Obligations

692 People will assume and perform roles according to their actual or perceived rights and responsibilities,
693 with or without explicit authority. In the context of a SOA ecosystem, human abilities and skills are
694 relevant as they equip individuals with knowledge, information and tools that may be necessary to have
695 meaningful and productive interactions with a view to achieving a desired outcome. For example, a
696 person who wants a particular book, and has both the right and responsibility of purchasing the book from
697 a given bookseller, will not have that need met from the online delegate of that bookstore if he does not
698 know how to use a web browser. Equally, just because someone does have the requisite knowledge or
699 skills does not entitle them *per se* to interact with a specific system.

700 Assuming or accepting rights and responsibilities depend on two important types of constraints that are
701 relevant to a SOA ecosystem: Permission and Obligation.

702 Permission

703 A constraint that identifies **actions** that an **actor** is (or is not) allowed to perform and/or the
704 **states** in which the actor is (or is not) permitted.

705 Note that permissions are distinct from ability, which refers to whether an actor has the capacity to
706 perform the action. Permission does not always involve acting on behalf of anyone, nor does it imply or
707 require the capacity to perform the action.

708 **Obligation**

709 A constraint that prescribes the **actions** that an **actor** must (or must not) perform and/or the
710 **states** the actor must (or must not) attain or maintain.

711 An example of obligations is the case where the service **consumer** and **provider** (see below) have
712 entered into an agreement to provide and consume a service such that the consumer is obligated to pay
713 for the service and the provider is obligated to provide the service – based on the terms of the contract.

714 An obligation can also be a **requirement** to maintain a given **state**. This may range from a requirement to
715 maintain a minimum balance on an account to a requirement that a service provider ‘remember’ that a
716 particular service consumer is logged in.

717 Both permissions and obligations can be identified ahead of time, but only permissions can be validated a
718 priori: before the intended action or before entering the constrained state. Obligations can only be
719 validated a posteriori through some form of auditing or verification process.

720 **3.2.2.3 Service Roles**

721 As in roles generally, a participant can play one or more in the SOA ecosystem, depending on the
722 context. A participant may be playing a role of a service provider in one relationship while simultaneously
723 playing the role of a consumer in another. Roles inherent to the SOA paradigm include **Consumer**,
724 **Provider**, **Owner**, and **Mediator**.

725 **Provider**

726 A role assumed by a **participant** who is offering a service.

727 **Consumer**

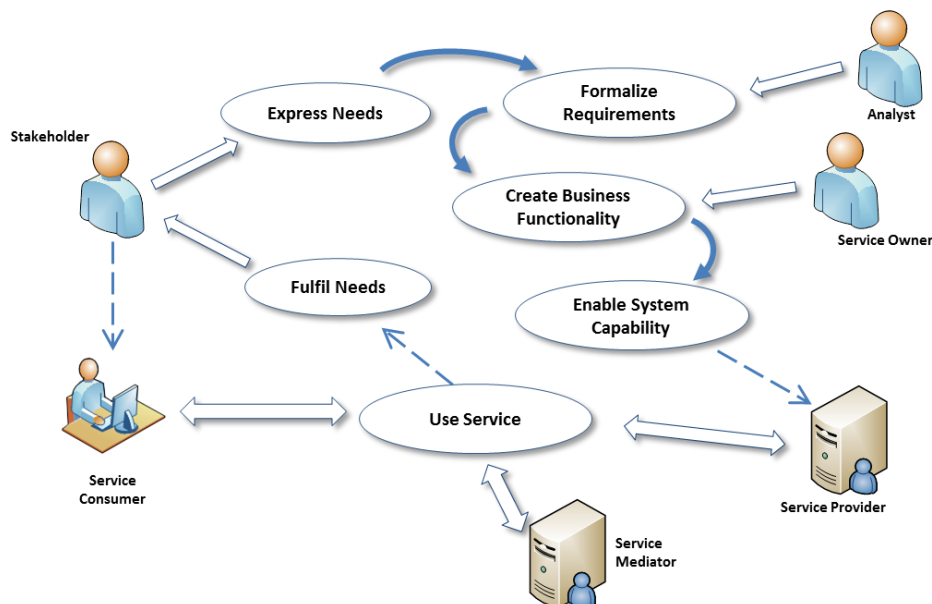
728 A role assumed by a **participant** who is interacting with a service in order to fulfill a **need**.

729 **Mediator**

730 A role assumed by a **participant** to facilitate interaction and connectivity in the offering and use of
731 services.

732 **Owner**

733 A role assumed by a **participant** who is claiming and exercising **ownership** over a service.



734

735

Figure 6 - Roles in a Service

736 Service consumers typically initiate interactions, but this is not necessarily true in all situations.
737 Additionally, several stakeholders may be involved in a service interaction supporting a given consumer.

738 The roles of service provider and service consumer are often seen as symmetrical, which is also not
739 entirely correct. A stakeholder tends to express a **Need** in non-formal terms: "I want to buy that book".
740 The type of need that a service is intended to fulfill has to be formalized and encapsulated by designers
741 and developers as a **Requirement**. This Requirement should then be reflected in the target service, as a
742 **Capability** that, when accessed via a service, delivers a **Real World Effect** to an arbitrary consumer:
743 "The chosen book is ordered for the consumer." It thus fulfills the need that has been defined for an
744 archetypal consumer.

745 Specific and particular customers may not experience a need exactly as captured by the service: "I don't
746 want to pay that much for the book", "I wanted an eBook version", etc. There can therefore be a process
747 of implicit and explicit negotiation between the consumer and the service, aimed at finding a 'best fit'
748 between the consumer's specific need and the capabilities of the service that are available and consistent
749 with the service provider's offering. This process may continue up until the point that the consumer is able
750 to accept what is on offer as being the best fit and finally 'invokes' the service. 'Execution context' has
751 thus been established. Conditions and agreements that contribute to the execution context are discussed
752 throughout this Reference Architecture.

753 Service mediation by a participant can take many forms and may invoke and use other services in order
754 to fulfill such mediation. For example, it might use a service registry in order to identify possible service
755 partners; or, in our book-buying example, it might provide a price comparison service, suggest alternative
756 suppliers, different language editions or delivery options.

757 3.2.3 Needs, Requirements and Capabilities

758 Participants in a SOA ecosystem often need other participants to *do* something, leveraging a **capability**
759 that they do not themselves possess. For example, a customer requiring a book may call upon a service
760 provider to deliver the book. Likewise, the service provider requires the customer to pay for it.

761 There is a reason that participants are engaged: they have different **needs** and have or apply different
762 capabilities for satisfying them. These are core to the concept of a service. The SOA-RM defines a
763 service as "the mechanism by which needs and capabilities are brought together". This idea of services
764 being a mechanism 'between' needs and capabilities was introduced in order to emphasize capability as
765 the notional or existing **business functionality** that would address a well-defined need. Service is
766 therefore the *implementation* of such business functionality *such that it is accessible* through a well-
767 defined interface. A capability that is isolated (i.e., it is inaccessible to potential consumers) is
768 emphatically not a service.

769 Business Functionality

770 A defined set of business-aligned tasks that provide recognizable business value to consumer
771 **stakeholders** and possibly others in the **SOA ecosystem**.

772 The idea of a service in a SOA ecosystem combines business functionality with implementation, including
773 the artifacts needed and made available as IT resources. From the perspective of software developers, a
774 SOA service enables the use of capabilities in an IT context. For the consumer, the service (combining
775 business functionality and implementation) generates intended real world effects. The consumer is not
776 concerned with the underlying artifacts which make that delivery possible.

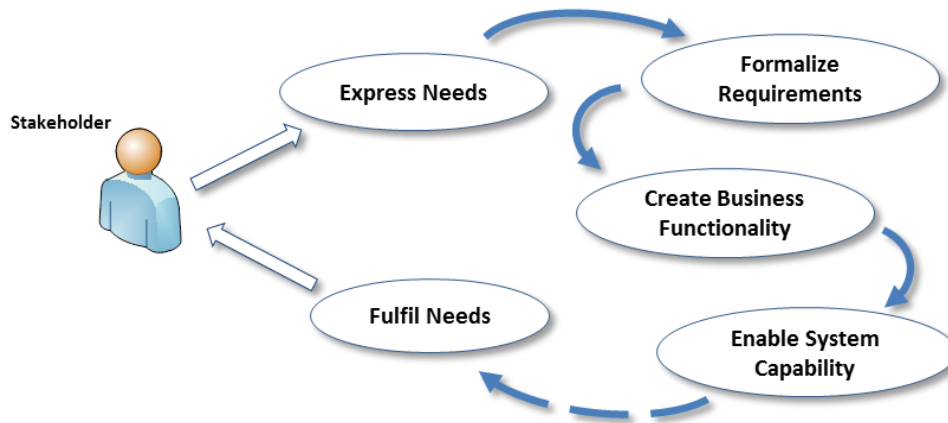


Figure 7 - Cycle of Needs, Requirements, and Fulfillment

777
778
779
780
781
782
783
784
785
786

In a SOA context, the stakeholder expresses a need (for example, the consumer who states “I want to buy a book”) and looks to an appropriate service to fulfill that need and assesses issues such as the trustworthiness, intent and **willingness** of a particular provider. This ecosystem communication continues up to the point when the stakeholder is ready to act. The stakeholder will then interact with a provider by invoking a service (for example, by ordering the book using an online bookseller) and engaging in relevant actions with the system (at this point, in a role as an *actor*, interacting with the system through a browser or mobile device, validating the purchase, submitting billing and delivery details) with a view to achieving the desired real world effect (having the book delivered).

787 **Need**

A general statement expressed by a **stakeholder** of something deemed necessary.

789 A need may be formalized as one or more requirements that must be fulfilled in order to achieve a stated
790 goal.

791 **Requirement**

A formal statement of a desired result (a **real world effect**) that, if achieved, will satisfy a **need**.

793 This requirement can then be used to create a capability that in turn can be brought to bear to satisfy that
794 need. Both the requirement and the capability to fulfill it are expressed in terms of desired real world
795 effect.

796 **Capability**

An ability to deliver a **real world effect**.

798 The Reference Model makes a distinction between a capability (as a *potential* to deliver the real world
799 effect) and the ability of bringing that capability to bear (via a realized service) as the realization of the
800 real world effect.

801 **Real World Effect**

A measurable change to the **shared state** of pertinent entities, relevant to and experienced by
803 specific **stakeholders** of an **ecosystem**.

804 This implies measurable change in the overall state of the SOA ecosystem. In practice, however, it is
805 specific state changes of certain entities that are relevant to particular participants that constitute the real
806 world effect as experienced by those participants.

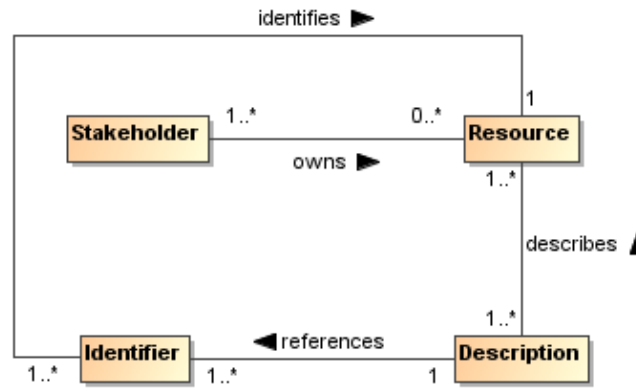
807 Objectives refer to real world effects that participants believe are achievable by a specific action or set of
808 actions that deliver appropriate changes in shared state, as distinct from a more generally stated ‘goal’.
809 For example, someone may wish to have enough light to read a book. In order to satisfy that goal, the
810 reader walks over to flip a light switch. The *objective* is to change the state of the light bulb, by turning on
811 the lamp, whereas the *goal* is to be able to read. The *real world effect* is more light being available to
812 enable the person to read.

813 While an effect is any measurable change resulting from an action, a SOA ecosystem is concerned more
814 specifically with real world effects.

815 3.2.4 Resource and Ownership

816 3.2.4.1 Resource

817 A resource is generally understood as an asset: it has value to someone. Key to this concept in a SOA
818 ecosystem is that a resource must be identifiable.



819
820

Figure 8 - Resources

821 Resource

822 An identifiable entity that has value to a **stakeholder**.

823 A resource may be identifiable by different methods but within a SOA ecosystem a resource must have at
824 least one well-formed identifier that may be unambiguously resolved to the intended resource.

825 Codified (but not *implied*) contracts, policies, obligations, and permissions are all examples of resources,
826 as are capabilities, services, service descriptions, and SOA-based systems. An *implied* policy, contract,
827 obligation or permission would not be a resource, even though it may have value to a stakeholder,
828 because it is not an identifiable entity.

829 Identifier

830 A sequence of characters that unambiguously indicates a particular **resource**.

831 Identifiers are assigned by social structures according to context, policies and procedures considered
832 sufficient for that structure's purposes.

833 For example, a group of otherwise unrelated humans are all, in a given context, employees of a particular
834 company and managed there as human resources. That company's policy is to assign each employee a
835 unique identifier number and has processes in place to do this, including verifying documentary evidence
836 (such as a birth certificate or ID). Each set of policies and procedures will reflect the needs of the social
837 structure for its particular context. Resources are typically used or managed by different stakeholder
838 groups, each of which may need to identify those resources in some particular way. As such, a given
839 resource may have multiple identifiers, each valid for a different context. In a SOA ecosystem, it is good
840 practice to use globally unique identifiers (for example, Internationalized Resource Identifiers, or IRIs)
841 irrespective of any other resource identifier that might be in use for a particular context.

842 The ability to identify a resource is important in interactions to determine such things as rights and
843 authorizations, to understand what functions are being performed and what the results mean, and to
844 ensure repeatability or characterize differences with future interactions. Many interactions within a SOA
845 ecosystem take place across ownership boundaries. Identifiers provide the means for all resources
846 important to a given SOA-based system to be *unambiguously* identifiable at any moment and in any
847 interaction.

848 Resources frequently have descriptions and the descriptions themselves may be considered resources.
849 This is discussed in Section 4.1.1. Resource description may link to other resources and their
850 descriptions; for example, a service description may link to a policy that constrains the conditions of use
851 of the service.

852 3.2.4.2 Ownership

853 Ownership is defined as a relationship between a stakeholder and a resource, where some stakeholder
854 (in a role as owner) has certain claims with respect to the resource.

855 Typically, the ownership relationship is one of control: the owner of a resource can control some aspect of
856 the resource.

857 Ownership

858 A set of claims, expressed as **rights** and **responsibilities** that a **stakeholder** has in relation to a
859 **resource**; it may include the right to transfer that ownership, or some subset of rights and
860 responsibilities, to another entity.

861 To own a resource implies taking responsibility for creating, maintaining and, if it is to be available to
862 others, provisioning the resource. More than one stakeholder may own different rights or responsibilities
863 associated with a given service, such as one stakeholder having the responsibility to deploy a capability
864 as a service, another owning the rights to the profits that result from charging consumers for using the
865 service, and yet another owning the right to use the service. There may also be joint ownership of a
866 resource, where the rights and responsibilities are shared.

867 A stakeholder who owns a resource may delegate some or all of these rights and responsibilities to
868 others, but typically retains the responsibility to see that the delegated rights and responsibilities are
869 exercised as intended

870 A crucial property that distinguishes ownership from a more limited right to use is the right to transfer
871 rights and responsibilities totally and irrevocably to another. When participants use but do not own a
872 resource, they may not be allowed to transfer the right to use the resource to a third participant. The
873 owner of the resource maintains the rights and responsibilities of being able to authorize others to use the
874 owned resource.

875 Ownership is defined in relation to the social structure relative to which the given rights and
876 responsibilities are exercised. For example, there may be constraints on how ownership may be
877 transferred, such as a government may not permit a corporation to transfer assets to a subsidiary in a
878 different jurisdiction.

879 Ownership Boundary

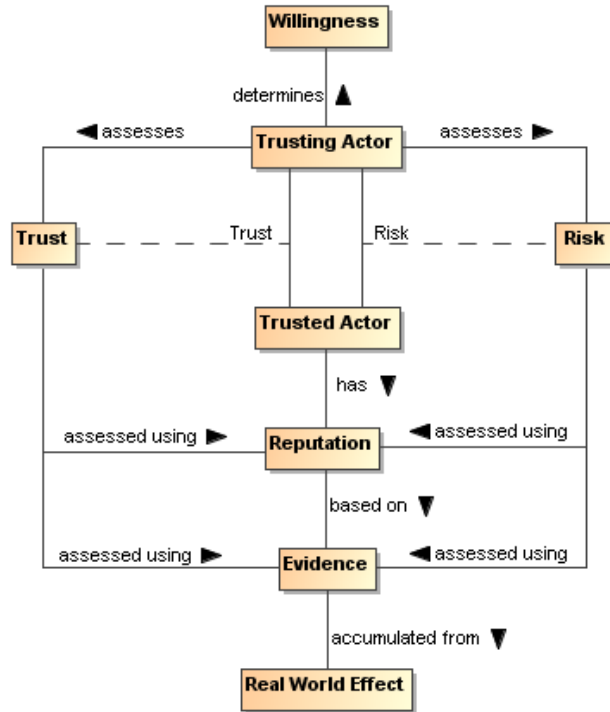
880 The extent of **ownership** asserted by a **stakeholder** or a **social structure** over a set of
881 **resources** and for which **rights** and **responsibilities** are claimed and (usually) recognized by
882 other stakeholders.

883 3.2.5 Establishing Execution Context

884 In a SOA ecosystem, providers and consumers of services may be, or may be acting on behalf of,
885 different owners, and thus the interaction between the provider and the consumer of a given service may
886 necessarily cross an ownership boundary. It is important to identify these ownership boundaries in a SOA
887 ecosystem and successfully crossing them is a key aspect of establishing execution context. This is turn
888 requires that the elements identified in the following sections be addressed.

889 **3.2.5.1 Trust and Risk**

890 For an interaction to occur each actor must be able and **willing** to participate.



891
892 *Figure 9 - Willingness and Trust*

893 **Willingness**

894 The internal commitment of a human **actor** (or of an automated non-human agent acting on a
895 **participant's** behalf) to carry out its part of an interaction.

896 Willingness to interact is not the same as a willingness to perform requested actions. For example, a
897 service provider that rejects all attempts to perform a particular action may still be fully willing and
898 engaged in interacting with the consumer. Important considerations in establishing willingness are both
899 **trust** and **risk**.

900 **Trust**

901 The private assessment or internal perception of one **actor** that another actor will perform
902 **actions** in accordance with an assertion regarding a desired **real world effect**.

903 **Risk**

904 The private assessment or internal perception of the likelihood that certain undesirable **real world**
905 **effects** will result from **actions** taken and the consequences or implications of such.

906 Trust is involved in all interactions and each actor will play a role as either (or alternately) a 'trusting' actor
907 and a 'trusted' actor. These roles are needed in order that all actors can trust all others in any given
908 interaction, at least to the extent required for continuance of the interaction. In traditional systems, the
909 balance between trust and risk is achieved by severely restricting interactions and by controlling the
910 participants of a system. In a SOA ecosystem, the degree and nature of that trust is likely to be different
911 for each actor, most especially when those actors are in different ownership boundaries.

912 An actor perceiving risk may take actions to mitigate that risk. At one extreme this will result in a refusal to
913 interact. Alternately, it may involve adding protection – for example by using encrypted communication
914 and/or anonymization – to reduce the perception of risk. Often, standard procedures are put in place to
915 increase trust and to mitigate risk.

916

917 The assessments of trust and risk are based on evidence available to the *trusting* actor. In general, the
918 trusting actor will seek evidence directly from the *trusted* actor (e.g., via documentation provided via the
919 service description) as well as evidence of the reputation of the trusted actor (e.g., third-party annotations
920 such as consumer feedback).

921 Trust is based on the confidence that the trusting actor has accurately and sufficiently gathered and
922 assessed evidence to the degree appropriate for the situation being assessed.

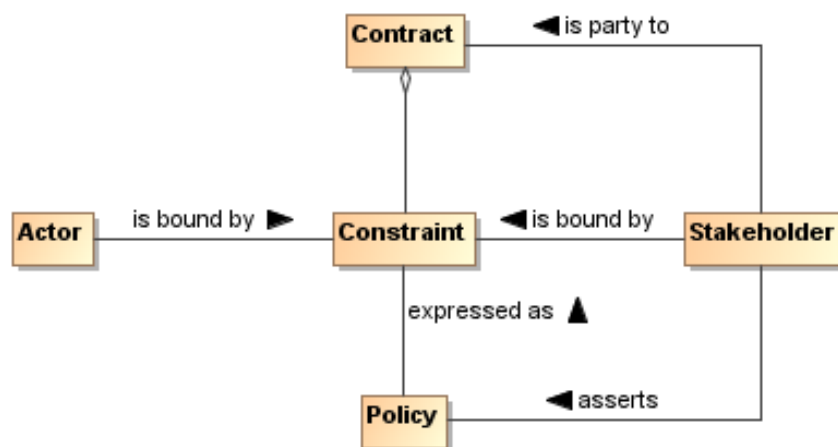
923 Assessment of trust is rarely binary. An actor is not completely trusted or untrusted because there is
924 typically some degree of uncertainty in the accuracy or completeness of the evidence or the assessment.
925 Similarly, there may be uncertainty in the amount and potential consequences of risk.

926 The relevance of trust to interaction depends on the assessment of risk. If there is little or no perceived
927 risk, or the risk can be covered by another party who accepts responsibility for it, then the degree of trust
928 may be less or not relevant in assessing possible actions. For example, most people consider there to be
929 an acceptable level of risk to privacy when using search engines, and submit queries without any sense
930 of trust being considered.

931 As perceived risk increases, the issue of trust becomes more of a consideration. For interactions with a
932 high degree of risk, the trusting actor will typically require stronger or additional evidence when evaluating
933 the balance between risk and trust. An example of high-risk is where a consumer's business is dependent
934 on the provider's service meeting certain availability and security requirements. If the service fails to meet
935 those requirements, the service consumer will go out of business. In this example, the consumer will look
936 for evidence that the likelihood of the service not meeting the performance and security requirements is
937 extremely low.

938 3.2.5.2 Policies and Contracts

939 As noted in the Reference Model, a policy represents some commitment and/or constraint advertised and
940 enforced by a stakeholder and that stakeholder alone. A contract, on the other hand, represents an
941 agreement by two or more participants. Enforcement of contracts may or may not be the responsibility of
942 the parties to the agreement but is usually performed by a stakeholder in the ecosystem (public authority,
943 legal system, etc.).



944
945

Figure 10 – Policies, Contracts and Constraints

946 Policy

947 An expression of constraints made by a **stakeholder** that the stakeholder commits to uphold and,
948 if desired or necessary, enforce. The constraints are usually stated as **permissions** and
949 **obligations** that affect the behavior of stakeholders or of any **actor** acting on their behalf.

950 Policies have an **owner** – the stakeholder who asserts and takes responsibility for the policy. This owner
951 may or may not be the owner of the object of the policy. These constraints may affect the stakeholder
952 asserting the policy or any other stakeholder involved. The constraints themselves represent some
953 measurable limitation on the state or behavior of the object of the policy, or of those who interact with it.

954 **Contract**

955 An agreement made by two or more **participants** (the contracting parties) on a set of conditions
956 (or contractual terms) together with a set of constraints that govern their behavior and/or **state** in
957 fulfilling those conditions.

958 A service provider's policy may become a service provider/consumer contract when a service consumer
959 agrees to the provider's policy. That agreement may be formal, or may be informal. If a consumer's policy
960 and a provider's policy are mutually exclusive, then some form of negotiation (involving human
961 interactions) or mediation must resolve the mutual exclusion before the service consumer/provider
962 interaction can occur. Note that this also applies if the consumer instead of the provider introduces the
963 policy.

964 Both policies and contracts imply a desire to see constraints respected and enforced. Stakeholders are
965 responsible for ensuring that any constraints in the policy or contract are enforced, although the actual
966 enforcement may be delegated to a different mechanism. A contract does not necessarily oblige the
967 contracting parties to act (for example to use a service) but it does constrain how they act if and when the
968 condition covered by the contract occurs (for example, when a service is invoked and used).

969 The realization of policies and contracts is discussed in Section 4.4 and contracts in the context of
970 management are discussed in Section 5.3.4.

971 **3.2.5.3 Communication**

972 **Communication**

973 A process involving the exchange of information between a sender and one or more recipients
974 and that ideally culminates in mutual understanding between them.

975 A communication involves a message, a sender of the message and at least one intended recipient, who
976 must be able to correctly interpret the message – or at least those parts of the message relevant to
977 sender and recipient in the particular context. Each must perform its respective role in order for the
978 communication to be successful and failing which, communication is not effective.

979 A communication may involve any number of recipients. In some situations, the sender may not be aware
980 of the recipient. However, without both a sender and a recipient, there is no communication. A given
981 communication can be a simple one-way transmission and not require a response by the recipient.
982 However, interaction does, necessarily, involve communication.

983 Message interpretation can itself be characterized in terms of **semantic engagement**: the proper
984 understanding of a message in a given context.

985 We can characterize the necessary modes of interpretation in terms of a shared understanding of a
986 common vocabulary (or mediation among vocabularies) and of the purpose of the communication. More
987 formally, we can say that a communication has a combination of message and purpose.

988 In a SOA ecosystem, senders and recipients can be stakeholders, participants or actors, depending on
989 whether execution context is being established or a specific interaction with the SOA-based system is in
990 progress. Communications need not resemble human speech: indeed system-level machine-to-machine
991 communication is typically highly stylized in form. It may take a particular form and involve terms not
992 found in everyday human communication.

993 **3.2.5.4 Semantics and Semantic Engagement**

994 Shared understanding is vital to a trusted and effective ecosystem and is a prerequisite to joint action
995 being carried out as intended. Semantics are therefore pervasive throughout SOA ecosystems and
996 important in communications as described above, as well as a driver for policies and other aspects of the
997 ecosystem.

998 In order to arrive at a shared understanding wherever this is necessary within the ecosystem, a
999 message's recipient must effectively understand and process statements, made in the sender's message,
1000 in a manner appropriate and sufficient to the particular context. Within a SOA-based system, non-human
1001 actors must at least be able to parse a message correctly (syntax) and act on the message's statements
1002 in a manner consistent with the sender's intent.

1003 Understanding and interpreting those assertions in a SOA-based system allows all the actors in any
1004 particular joint action to ‘know’ what may be expected of them. An actor can potentially ‘understand’ an
1005 assertion in a number of ways, but it is specifically the process of arriving at a *shared* understanding that
1006 is important in the ecosystem. This process is semantic engagement and it takes place in different forms
1007 throughout the SOA ecosystem. It can be instantaneous or progressively achieved. Participants – who
1008 play the role both as actors in the SOA-based system and as stakeholders in social structures and the
1009 wider ecosystem – can be pivotal in resolving problems of understanding and determining when there is a
1010 level of engagement appropriate and sufficient to the particular context.

1011 **Semantic Engagement**

1012 The process by which an **actor** engages with a set of assertions based on that actor’s
1013 interpretation and understanding of those assertions.

1014 Different actors have differing capabilities and requirements for understanding assertions. This is true for
1015 both human and non-human actors. For example, a purchase order process does not require that a
1016 message forwarding agent ‘understand’ the purchase order, but a processing agent does need to
1017 ‘understand’ the purchase order in order to know what to do with the order once received.

1018 The impact of any assertion can only be fully understood in terms of specific social contexts that
1019 necessarily include the actors that are involved. For example, a policy statement that governs the actions
1020 relating to a particular resource may have a different impact or purpose for the participant that owns the
1021 resource than for the actor that is trying to access it: the former understands the purpose of the policy as
1022 a statement of enforcement - the latter understands it as a statement of constraint.

1023 **3.3 Action in a SOA Ecosystem Model**

1024 Participants cannot always achieve desired results by leveraging resources in their own ownership
1025 domain. This unfulfilled need leads them to seek and leverage services provided by other participants and
1026 using resources beyond their ownership and control. The participants identify service providers with which
1027 they think they can interact to achieve their objective and engage in joint action with those other actors
1028 (service providers) in order to bring about the desired outcome. The SOA ecosystem provides the
1029 environment in which this happens.

1030 An action model is put forth a-priori by the service provider, and is effectively an undertaking by the
1031 service provider that the actions – identified in the action model and invoked consistent with the process
1032 model – will result in the described real world effect. The action model describes the actions leading to a
1033 real-world effect. A potential service consumer – who is interested in a particular outcome to satisfy their
1034 need – must understand those actions as capable of achieving that desired outcome.

1035 When the consumer ‘invokes’ a service, a joint action is started as identified in the action model,
1036 consistent with the temporal sequence as defined by the process model, and where the consumer and
1037 the provider are the two parties of the joint action. Additionally, the consumer can be assured that the
1038 identified real-world effects will be accomplished through evidence provided via the service description.

1039 Since the service provider does not know about all potential service consumers, the service provider may
1040 also describe what additional constraints are necessary in order for the service consumer to invoke
1041 particular actions, and thus participate in the joint action. These additional constraints, along with others
1042 that might not be listed, are preconditions for the joint action to occur and/or continue (as per the process
1043 model), and are referred to in the SOA-RM as execution context. Execution context goes all the way from
1044 human beings involved in aligning policies, semantics, network connectivity and communication
1045 protocols, to the automated negotiation of security protocols and end-points as the individual actions
1046 proceed through the process model.

1047 Also, it is important to note that both actions and real world effect are recursive in nature, in the sense
1048 that they can often be broken down into more and more granularity depending on how they are examined
1049 and what level of detail is important.

1050 All of these things are important to getting to the core of participants’ concern in a SOA ecosystem: the
1051 ability to leverage resources or capabilities to achieve a desired outcome, and in particular where those
1052 resources or capabilities do not belong to them or are beyond their direct control. i.e., that are outside of
1053 their ownership boundary.

1054 In order to use such resources, participants must be able to identify their own needs; state those needs in
1055 the form of requirements; compose or identify a suitable business solution using resources or capabilities
1056 that will meet their needs; and engage in joint action – the coordinated set of actions that participants
1057 pursue in order to achieve measurable results in furtherance of their goals.

1058 In order to act in a way that is appropriate and consistent, participants must communicate with each other
1059 about their own goals, objectives and policies, and those of others. This is the main concern of Semantic
1060 Engagement.

1061 A key aspect of joint action revolves around the trust that both parties must exhibit in order to participate
1062 in the joint action. The willingness to act and a mutual understanding of both the information exchanged
1063 and the expected results is the particular focus of Sections 3.2.5.1 and 3.2.5.4.

1064 **3.3.1 Services Reflecting Business**

1065 The SOA paradigm often emphasizes the interface through which service interaction is accomplished.
1066 While this enables predictable integration in the sense of traditional software development, the prescribed
1067 interface alone does not guarantee that services will be composable into business solutions.

1068 **Business Solution**

1069 A set of defined interactions that combine implemented or notional **business functionality** in
1070 order to address a set of business needs.

1071 **Composability**

1072 The ability to combine individual services, each providing defined **business functionality**, so as
1073 to provide more complex **business solutions**.

1074 To achieve composability, capabilities must be identified that serve as building blocks for business
1075 solutions. In a SOA ecosystem, these building blocks are captured as services representing well-defined
1076 business functions, operating under well-defined policies and other constraints, and generating well-
1077 defined real world effects. These service building blocks should be relatively stable so as not to force
1078 repeated changes in the compositions that utilize them, but should also embody SOA attributes that
1079 readily support creating compositions that can be varied to reflect changing circumstances.

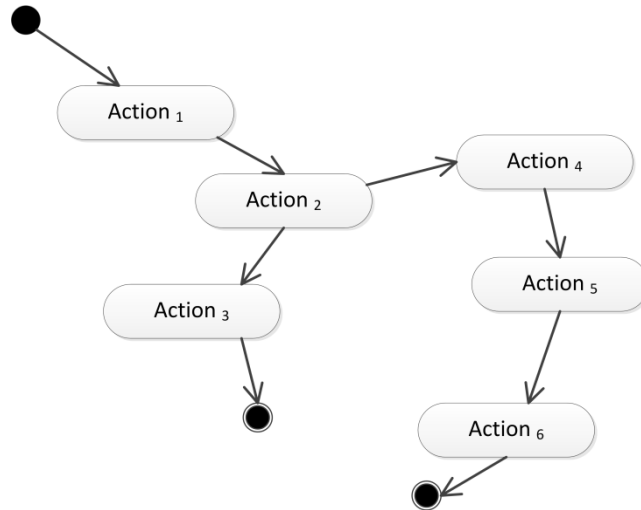
1080 The SOA paradigm emphasizes both composition of services and opacity of how a given service is
1081 implemented. With respect to opacity, the SOA-RM states that the service could carry out its described
1082 functionality through one or more automated and/or manual processes that in turn could invoke other
1083 available services.

1084 Any composition can itself be made available as a service and the details of the business functionality,
1085 conditions of use, and effects are among the information documented in its service description.

1086 Composability is important because many of the benefits of a SOA approach assume multiple uses for
1087 services, and multiple use requires that the service deliver a business function that is reusable in multiple
1088 business solutions. Simply providing a Web Service interface for an existing IT artifact does not, in
1089 general, create opportunities for sharing business functions. Furthermore, the use of tools to auto-
1090 generate service software interfaces will not guarantee services that can effectively be used within
1091 compositions if the underlying code represents programming constructs rather than business functions. In
1092 such cases, services that directly expose the software details will be as brittle to change as the underlying
1093 code and will not exhibit the characteristic of loose coupling.

1094 **3.3.2 Activity, Action, and Joint Action**

1095 In general terms, entities act in order to fulfill particular objectives. More precisely, they generate activity.
1096 An activity is made up of specific Actions (or other Activities) and is formally defined in [ISO/IEC 10746-2]
1097 as “a single-headed directed acyclic graph of actions...”⁴ It is most clearly understood diagrammatically:



1098 *Figure 11: An Activity, expressed informally as a graph of Actions, with a single Start*
1099 *point and alternative End points*
1100

1101 What constitutes an Action or an Activity will be a matter of context. For the SOA-RAF, an Action
1102 represents the smallest and most discrete activity that must be modeled for a given Viewpoint.

1103 The form of Activity that is of most interest within a SOA ecosystem is that involving Actions as defined
1104 below and their interaction across ownership boundaries (and thus involving interaction between more
1105 than one actor) – we call this **joint action**. In Figure 12 below, one line of activity (on the left) can be
1106 completed thru Action₃ without crossing any ownership boundary but the alternative path, starting at
1107 Action₄, can only be completed as a result of joint action across an ownership boundary:

⁴ See [ISO/IEC 10746] Part 2: Foundations

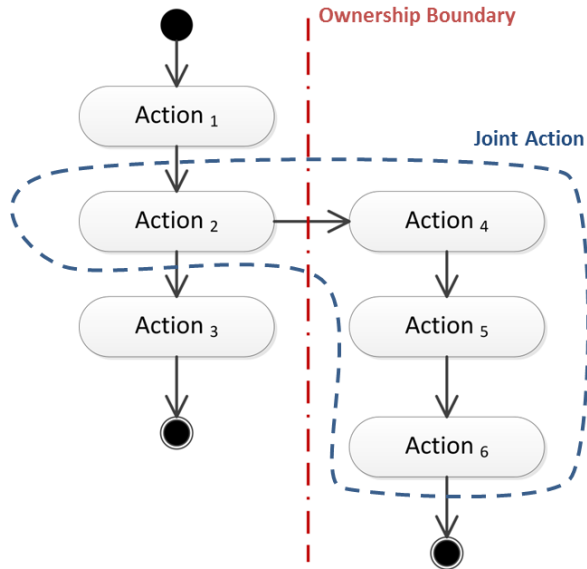


Figure 12: Activity involving Actions across an ownership boundary

1108

1109

1110 **Action**

1111 The application of intent by an **actor** to cause an effect.

1112 The aspect of action that distinguishes it from mere force or accident is that someone *intends* that the
 1113 action achieves a desired objective or effect. This definition of action is very general. In the case of SOA,
 1114 we are mostly concerned with actions that take place within a system and have specific effects on the
 1115 SOA ecosystem – defined in section 3.2.3 as real world effects. The actual real world effect of an action,
 1116 however, may go beyond the intended effect.

1117 In order for multiple actors to participate in a joint action, they must each act according to their role within
 1118 the joint action. This is achieved through communication and messaging.

1119 Communication – the formulation, transmission, receipt and interpretation of messages – is the
 1120 foundation of all joint actions within the SOA ecosystem, given the inherent separation – often across
 1121 ownership boundaries – of actors in the system.

1122 Communication between actors requires that they play the roles of ‘sender’ or ‘receiver’ of messages as
 1123 appropriate to a particular action – although it is not necessarily required that they both be active
 1124 simultaneously.

1125 An actor sends a message in order to communicate with other actors. The communication itself is often
 1126 not intended as part of the desired real world effect but rather includes messages that seek to establish,
 1127 manage, monitor, report on, and guide the joint action throughout its execution.

1128 Like communication, joint action usually involves different actors. However, joint action – resulting from
 1129 the deliberate actions undertaken by different actors – *intentionally* impacts shared state within the
 1130 system leading to real world effects.

1131 **Joint Action**

1132 The coordinated set of **actions** involving the efforts of two or more **actors** to achieve an effect.

1133 Note that the effect of a joint action is *not* always equivalent to one or more effects of the individual
 1134 actions of the actors involved, i.e., it may be more than the sum of the parts.

1135 Different perspectives lead to either communication or joint action as being considered most important.
 1136 For example, from the perspective of ecosystem security, the integrity of the communications may be
 1137 dominant; from the perspective of ecosystem governance, the integrity of the joint action may be
 1138 dominant.

1139 3.3.3 State and Shared State

1140 State

1141 The condition of an entity at a particular time.

1142 State is characterized by a set of facts that is true of the entity. In principle, the total state of an entity (or
1143 the world as a whole) is unbounded. In practice, we are concerned only with a subset of the state of an
1144 entity that is measurable and useful in a given context.

1145 For example, the total state of a light bulb includes the temperature of the filament of the bulb, the
1146 composition of the glass, the dirt that is on the bulb's surface and so on. However, someone needing
1147 more light to read is only interested in whether the bulb is 'on' or 'off' and if it is working properly. That
1148 individual's characterization of the state of the bulb reduces to the fact: "bulb is now on".

1149 In a SOA ecosystem, there is a distinction between the set of facts about an entity that only that entity can
1150 access and the set of facts that may be accessible to others, notably actors in the SOA-based system.

1151 Private State

1152 That part of an entity's **state** that is knowable by, and accessible to, only that entity.

1153 Shared State

1154 That part of an entity's **state** that is knowable by, and may be accessible to, other actors.

1155 Note that shared state does not imply that the state *is* accessible to other actors. It simply refers to that
1156 subset of state that *may* be accessed by other actors. This will principally be the case when actors need
1157 to participate in joint actions.

1158 It is the aggregation of the shared states of pertinent entities that constitutes the desired effect of a joint
1159 action. Thus the change to this shared state is what is experienced in the wider ecosystem as a real world
1160 effect

1161 3.4 Architectural Implications

1162 3.4.1 Social structures

1163 A SOA ecosystem's participants are organized into various forms of social structure. Not all social
1164 structures are hierarchical: a SOA ecosystem **SHOULD** be able to incorporate peer-to-peer forms of
1165 organization as well as hierarchic structures. In addition, it **SHOULD** be possible to identify and access
1166 any constitutional agreements that define the social structures present in a SOA ecosystem.

- 1167 • Different social structures have different rules of engagement but predictable behavior is one of
1168 the underpinnings of trust. Mechanisms **MUST** therefore be available to:
 - 1169 ○ express constitutions and other organizing principles of participants;
 - 1170 ○ inherit rules of engagement from parent to child social structures.
- 1171 • Social structures have roles and members and this impacts who may be authorized to act and in
1172 what circumstances. Mechanisms **MUST** be available to:
 - 1173 ○ identify and manage members of social structures
 - 1174 ○ Identify and manage attributes of the members
 - 1175 ○ describe roles and role adoption
- 1176 • Social structures overlap and interact, giving rise to situations in which rules of engagement may
1177 conflict. In addition, a given actor may be a member of multiple social structures and the social
1178 structures may be associated with different jurisdictions. Mechanisms **MUST** be available to:
 - 1179 ○ identify the social structures that are active during a series of joint actions;
 - 1180 ○ identify and resolve conflicts and inconsistencies.

1181 3.4.2 Resource and Ownership

1182 Communication about and between, visibility into, and leveraging of resources requires the unambiguous
1183 identification of those resources. Mechanisms **MUST** be available for:

- 1184 • Assigning and guaranteeing uniqueness of globally unique identifiers

- 1185 • Identifying the extent of the enterprise over which the identifier must be understandable and
1186 unique
1187 • Ensuring the longevity of identifiers (i.e., they cannot just change arbitrarily)

1188 3.4.3 Policies and Contracts

- 1189 • Policies are expressed as constraints:
 - 1190 ○ Policies **MUST** be expressed
 - 1191 ○ Constraints **MUST** be enforceable
 - 1192 ○ Management of potentially large numbers of policies **MUST** be achievable
- 1193 • Policies have owners:
 - 1194 ○ Policies are established and **MUST** be owned and enforced by stakeholders within social
1195 structures.
- 1196 • Policies may be inconsistent with one another. As such,
 - 1197 ○ Policy conflict resolution techniques **MUST** exist and be accessible
- 1198 • Agreements are accepted constraints:
 - 1199 ○ Contracts **SHOULD** be enforced by mechanisms of the social structure and be
1200 accessible as needed

1201 3.4.4 Semantics

1202 Semantics is pervasive in a SOA ecosystem: apart from communicated content there are mission and
1203 policy statements, goals, objectives, descriptions, agreements, etc. Without clear semantics,
1204 understanding is degraded and semantic engagement difficult to achieve, in turn leading to loss of trust
1205 and willingness to cooperate in the use of ecosystem resources. As such,

- 1206 • messages **MUST** be understood and processed in a manner appropriate and sufficient to a
1207 particular context. In particular, actors **MUST** be able to process content such as
1208 communications, descriptions and policies solely on the basis of semantic engagement between
1209 the actors concerned.
- 1210 • any assertion essential to the operation of the ecosystem (such as policy statements, and
1211 descriptions) **MUST** have carefully chosen written expressions and associated decision
1212 procedures.
- 1213 • While no two actors can fully share their interpretation of elements of vocabularies, they **SHOULD**
1214 be able to understand the intended public meaning of vocabularies' elements.

1215 3.4.5 Trust and Risk

1216 Actors **MUST** be able to explicitly reason about both trust and risk in order to effectively participate in a
1217 SOA ecosystem. The more open and public the SOA ecosystem is, the more important it is for actors to
1218 be able to reason about their participation.

1219 3.4.6 Needs, Requirements and Capabilities

1220 In the process of capturing needs as requirements, the subsequent processes of decomposition and
1221 allocation of requirements **SHOULD** be informed by capabilities that already exist and those capabilities
1222 **MUST** be understood as potential services.

1223 3.4.7 The Importance of Action

1224 People participate in a SOA ecosystem in order to have specific needs met. This involves both individual
1225 and joint actions. As such,

- 1226 • Actions **MUST** be adequately documented so as to make clear what real world effects are
1227 intended as a result of an actor's involvement;
- 1228 • Actions **SHOULD** be designed to provide well-scoped functionality that support composition into
1229 larger actions;

1230
1231
1232

- As with related services, implementation details of an action **SHOULD** be opaque to the consumer; an action resulting from the composition of other actions **SHOULD NOT** expose composition details to the consumer

4 Realization of a SOA Ecosystem view

Make everything as simple as possible but no simpler.
Albert Einstein

The *Realization of a SOA Ecosystem* view focuses on elements that are needed to support the discovery of and interaction with services. The key questions asked are "What are services, what support is needed and how are they realized?"

The models in this view include the Service Description Model, the Service Visibility Model, the Interacting with Services Model, and the Policies and Contracts Model.

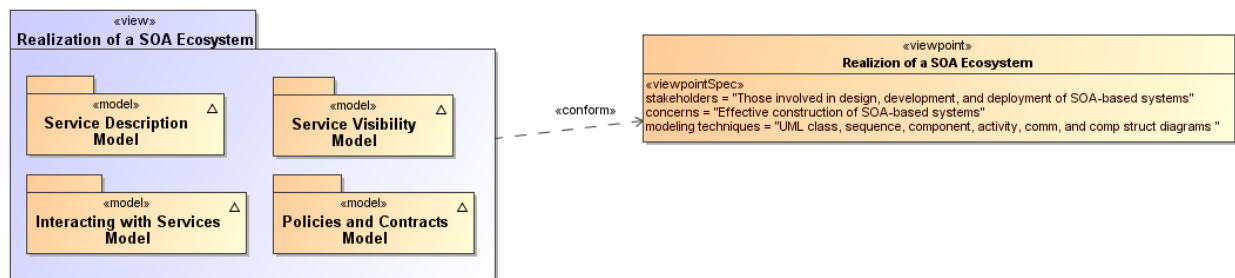


Figure 13 - Model Elements Described in the Realization of a SOA Ecosystem view

The Service Description Model informs the participants of what services exist and the conditions under which they can be used. The Policies and Contracts Model elaborates on the conditions under which service use is prescribed and agreements among participants in the SOA ecosystem.. The information in the service description as augmented by details of policy provides the basis for visibility as defined in the SOA Reference Model and captured in the Service Visibility Model. Finally, the process by which services are used under the defined conditions and agreements is described in the Interacting with Services Model.

4.1 Service Description Model

A service description is an artifact, often document-based, that defines or references the information needed to use, deploy, manage and otherwise control a service. This includes not only the information and behavior models associated with a service that define interaction via the service interface but also includes information needed to decide whether the service is appropriate for the current requirements of the service consumer. Thus, the service description should also include information such as service reachability, service functionality, and the policies associated with a service.

A service description artifact may be a single document or it may be an interlinked set of documents. For the purposes of this model, differences in representation are to be ignored, but the implications of a 'web of documents' are discussed later in this section.

There are several points to note regarding service description:

- The Reference Model states that one of the hallmarks of SOA is the large amount of associated description. The model presented below focuses on the description of services but it is equally important to consider the descriptions of the consumer, other participants, and needed resources other than services.
- Descriptions are inherently incomplete but may be determined as *sufficient* when it is possible for the participants to access and use the described services based only on the descriptions provided. This means that, at one end of the spectrum, a description along the lines of "That service on that machine" may be sufficient for the intended audience. On the other extreme, a service description with a machine-process-able description of the semantics of its **operations** and real world effects may be required for services accessed via automated service discovery and planning systems.

- 1273
- 1274
- 1275
- 1276
- 1277
- 1278
- 1279
- 1280
- 1281
- 1282
- 1283
- 1284
- 1285
- 1286
- 1287
- 1288
- 1289
- 1290
- 1291
- 1292
- Descriptions come with context, i.e. a given description comprises information needed to adequately support the context. For example, a list of items can define a version of a service, but for many contexts an indicated version number is sufficient without the detailed list. The current model focuses on the description needed by a service consumer to understand what the service does, under what conditions the service will do it, how well the service does it, and what steps are needed by the consumer to initiate and complete a service interaction. Such information also enables the service provider to clearly specify what is being provided and the intended conditions of use.
 - Descriptions change over time as, for example, the ingredients and nutrition information for food labeling continues to evolve. A need for transparency of transactions may require additional description for those associated contexts.
 - Description always proceeds from a basis of what is considered ‘common knowledge’. This may be social conventions that are commonly expected or possibly codified in law. It is impossible to describe everything and it can be expected that a mechanism as far reaching as SOA will also connect entities where there is inconsistent ‘common’ knowledge.
 - Descriptions become the collection point of information related to a service or any other resource, but it is not necessarily the originating point or the motivation for generating this information. In particular, given a SOA service as the access to an underlying capability, the service may point to some of the capability’s previously generated description, e.g. a service providing access to a data store may also have access to information indicating the freshness of the data.

1293 These points emphasize that there is no one ‘right’ description for all contexts and for all time. Several descriptions for the same subject may exist at the same time, and this emphasizes the importance of the description referencing source material maintained by that material’s owner rather than having multiple copies that become out of synch and inconsistent.

1297 It may also prove useful for a description assembled for one context to cross-reference description assembled for another context as a way of referencing ancillary information without overburdening any single description. Rather than a single artifact, description can be thought of as a web of documents that enhance the total available description.

1301 This Reference Architecture Foundation uses the term service description for consistency with the concept defined in the Reference Model. Some SOA literature treats the idea of a ‘service contract’ as equivalent to service description. In the SOA-RAF, the term service description is preferred. Replacing the term ‘service description’ with the term ‘service contract’ implies that just one side of the interaction is governing and misses the point that a single set of policies identified by a service description may lead to numerous contracts, i.e. service level agreements, leveraging the same description.

1307 4.1.1 The Model for Service Description

1308 Figure 14 shows Service Description as a subclass of the general Description class. As well as *describing*
1309 a Resource (as we saw in Section 3.2.4.1), a Description is also a subclass of the Resource class. In
1310 addition, each resource is assumed to *have* a description⁵. The following section discusses the
1311 relationships among elements of general description and the subsequent sections focus on service
1312 description. Other descriptions, such as those of participants, are important to SOA but are not
1313 individually elaborated in this document.

⁵ The description itself can have further descriptive data such as its version or last revision. The model emphasizes this point but should not be interpreted too rigorously as allowing endless recursion.

1314 **4.1.1.1 Elements Common to General Description**

1315 The general Description class is composed of a number of elements that are expected to be common
 1316 among all descriptions supporting a service oriented architecture. A registry/repository often contains a
 1317 subset of the description instance, where the chosen subset is identified as that which facilitates
 1318 discovery. Additional information contained in a more complete description may be needed to initiate and
 1319 continue interaction.

1320

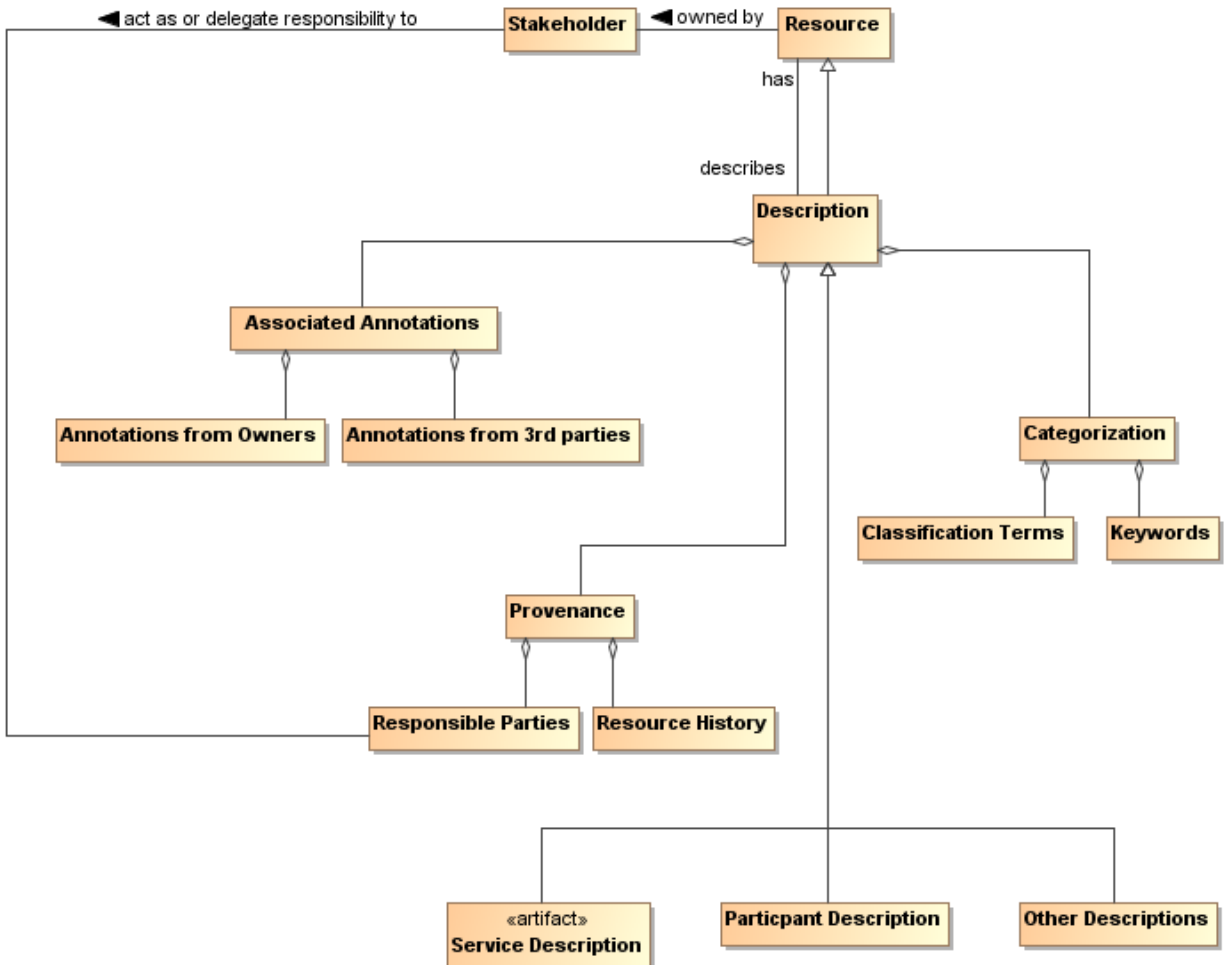


Figure 14 - General Description

1321
1322

1323 **4.1.1.1.1 Provenance**

1324 While the resource Identifier provides the means to know which subject and subject description are being
 1325 considered, Provenance as related to the Description class provides information that reflects on the
 1326 quality or usability of the subject. Provenance specifically identifies the stakeholder (human, defined role,
 1327 organization, etc.) who assumes responsibility for the resource being described and tracks historic
 1328 information that establishes a context for understanding what the resource provides and how it has
 1329 changed over time. Responsibilities may be directly assumed by the stakeholder who owns a resource
 1330 (see Section 3.2.4.2) or the Owner may designate Responsible Parties for the various aspects of
 1331 maintaining the resource and provisioning it for use by others. There may be more than one stakeholder
 1332 identified under Responsible Parties; for example, one stakeholder may be responsible for code
 1333 maintenance while another is responsible for provisioning of the executable code.

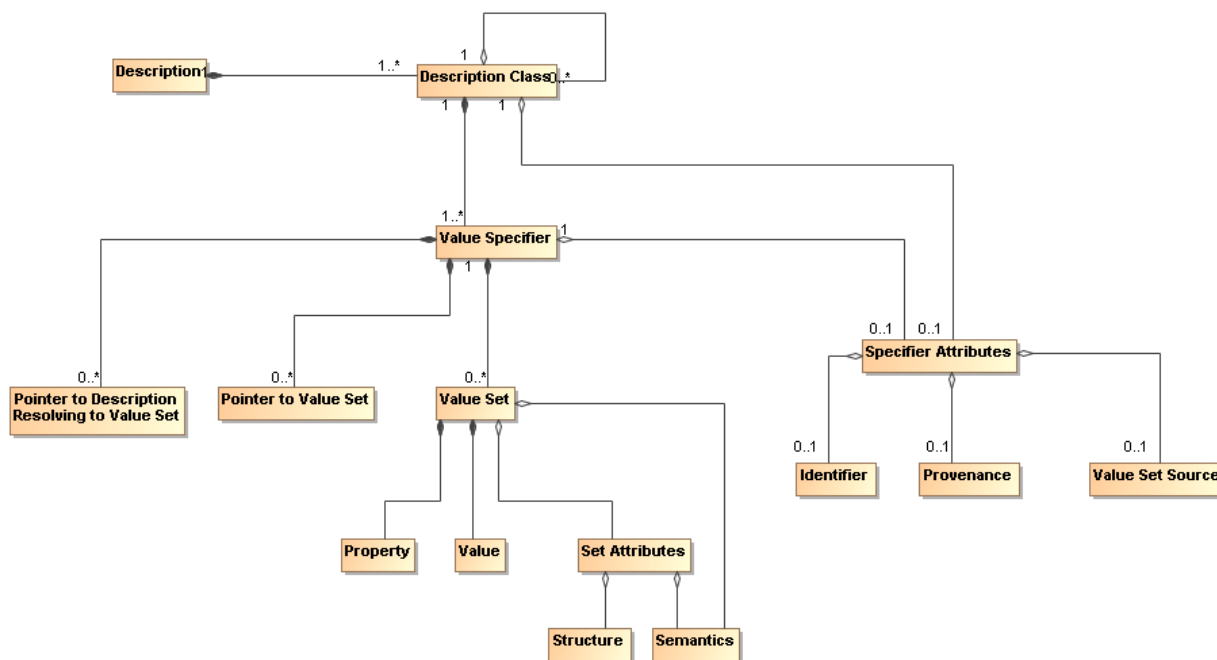
1334 **4.1.1.1.2 Keywords and Classification Terms**

1335 A traditional element of description has been to associate the resource being described with predefined
1336 keywords or classification taxonomies that derive from referenceable formal definitions and vocabularies.
1337 This Reference Architecture Foundation does not prescribe which vocabularies or taxonomies may be
1338 referenced, nor does it limit the number of keywords or classifications that may be associated with the
1339 resource. It does, however, state that a normative definition of any terms or keywords should be
1340 referenced, whether that be a representation in a formal ontology language, a pointer to an online
1341 dictionary, or any other accessible source. See Section 4.1.1.2 for further discussion on associating
1342 semantics with assigned values.

1343 **4.1.1.1.3 Associated Annotations**

1344 The general description instance may also reference associated documentation that is in addition to that
1345 considered necessary in this model. For example, the owner of a service may have documentation on
1346 best practices for using the service. Alternately, a third party may certify a service based on their own
1347 criteria and certification process; this may be vital information to other prospective consumers if they were
1348 willing to accept the certification in lieu of having to perform another certification themselves. Note, while
1349 the examples of Associated Documentation presented here are related to services, the concept applies
1350 equally to description of other entities.

1351 **4.1.1.2 Assigning Values to Description Instances**



1352
1353 *Figure 15 - Representation of a Description*

1354 Figure 14 shows the template for a general description, but individual description instances depend on
1355 the ability to associate meaningful values with the identified elements. Figure 15 shows a model for a
1356 collection of information that provides for value assignment and traceability for both the meaning and the
1357 source of a value. The model is not meant to replace existing or future schema or other structures that
1358 have or will be defined for specific implementations, but it is meant as guidance for the information such
1359 structures need to capture to generate sufficient description. It is expected that tools will be developed to
1360 assist the actor in populating description and auto-filling many of these fields, and in that context, this
1361 model provides guidance to the tool developers.

1362 In Figure 15, each class has an associated value specifier or is made up of components that eventually
1363 resolve to a value specifier. For example, Description has several components, one of which is
1364 Categorization, which would have an associated value specifier.

1365 A value specifier consists of

1366 • a collection of value sets with associated property-value pairs, pointers to such value sets, or

1367 pointers to descriptions that eventually resolve to value sets that describe the component; and

1368 • attributes that qualify the value specifier and the value sets it contains.

1369 The qualifying attributes for the value specifier include

1370 • an optional identifier that would allow the value set to be defined, accessed, and reused

1371 elsewhere;

1372 • provenance information that identifies the person (individual or organization) who has

1373 responsibility for assigning the value sets to any description component;

1374 • an optional source of the value set, if appropriate and meaningful, e.g. if a particular data source

1375 is mandated.

1376 If the value specifier is contained within a higher-level component (such as Service Description containing

1377 Service Functionality), the component may assume values from the attributes of its container.

1378 Note, provenance as a qualifying attribute of a value specifier is different from provenance as part of an

1379 instance of Description. Provenance for a service identifies those who own and are responsible for the

1380 service, as described in Section 3.2.4. Provenance for a value specifier identifies who is responsible for

1381 choosing and assigning values to the value sets that comprise the value specifier. It is assumed that

1382 granularity at the value specifier level is sufficient and provenance is not required for each value set.

1383 The value set also has attributes that define its structure and semantics.

1384 • The semantics of the value set property should be associated with a semantic context conveying

1385 the meaning of the property within the execution context, where the semantic context could vary

1386 from a free text definition to a formal ontology.

1387 • For numeric values, the structure would provide the numeric format of the value and the

1388 ‘semantics’ would be conveyed by a dimensional unit with an identifier to an authoritative source

1389 defining the dimensional unit and preferred mechanisms for its conversion to other dimensional

1390 units of like type.

1391 • For nonnumeric values, the structure would provide the data structure for the value

1392 representation and the semantics would be an associated semantic model.

1393 • For pointers, architectural guidelines would define the preferred addressing scheme.

1394 The value specifier may indicate a default semantic model for its component value sets and the individual

1395 value sets may provide an override.

1396 The property-value pair construct is introduced for the value set to emphasize the need to identify

1397 unambiguously both what is being specified and what is a consistent associated value. The further

1398 qualifying of Structure and Semantics in the Set Attributes allows for flexibility in defining the form of the

1399 associated values.

1400 **4.1.1.3 Model Elements Specific to Service Description**

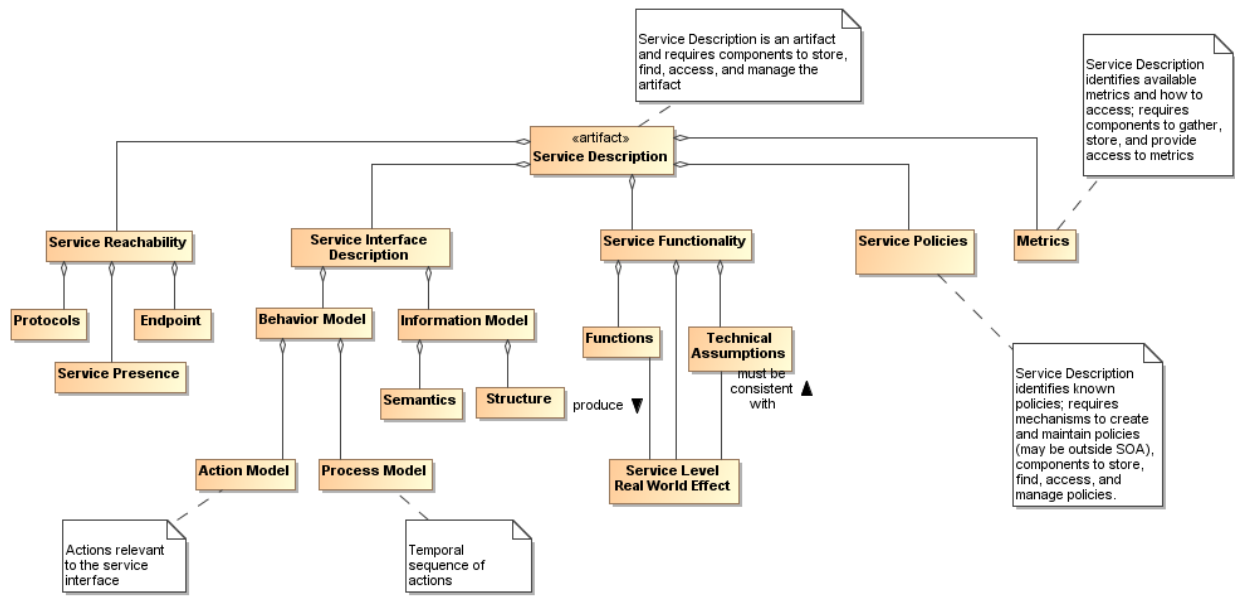


Figure 16 - Service Description

1401
1402

1403 The major elements for the Service Description subclass follow directly from the areas discussed in the
 1404 Reference Model. Here, we discuss the detail shown in Figure 16 and the purpose served by each
 1405 element of service description. For example, Service Policies as included in Figure 16 indicate those
 1406 policies that affect conditions of use of the service; however, while the description may link to detailed
 1407 policy documents, it is not the purpose of description to justify or elaborate on the rationale for the
 1408 policies. Similarly, Service Interface Description as included in Figure 16 captures information about what
 1409 interactions are supported by the service via its Behavior Model and the information exchange needed to
 1410 carry out those interactions in accordance with the service's Information Model; it is not the coded
 1411 interface.

1412 Note, the intent in the subsections that follow is to describe how a particular element, such as the service
 1413 interface description, is reflected in the service description, not to elaborate on the details of that element.

1414 **4.1.1.3.1 Service Interface Description**

1415 As noted in the Reference Model, the service interface is the means for interacting with a service. For the
 1416 SOA-RAF and as shown in Section 4.3 the service interface supports an exchange of messages, where

- 1417 • the message conforms to a referenceable message exchange pattern (MEP, covered below in
- 1418 Section 4.3.3.1),
- 1419 • the message payload conforms to the structure and semantics of the indicated information model,
- 1420 • the messages are used to denote events related to or actions against the service, where the
- 1421 actions are specified in the action model and any required sequencing of actions is specified in
- 1422 the process model.

1423 The Service Interface Description element as shown in Figure 17 includes the information needed to carry
 1424 out this message exchange in order to realize the service behavior described. In addition to the
 1425 Information Model that conveys the Semantics and Structure of the message, the Service Interface
 1426 Description indicates what behavior can be expected through interactions conveyed in the Action and
 1427 Process Models.

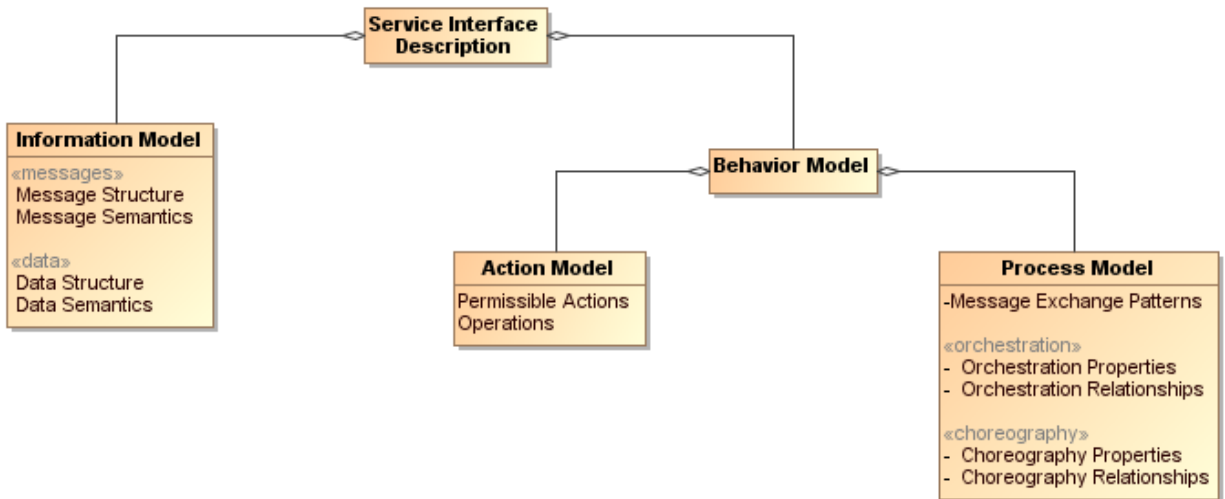


Figure 17 - Service Interface Description

1428
1429

1430 Note we distinguish the structure and semantics of the message from that of the underlying **protocol** that
1431 conveys the message. The message structure may include nested structures that are independently
1432 defined, such as an enclosing envelope structure and an enclosed data structure.

1433 These aspects of messages are discussed in more detail in Section 4.3.2.

1434 4.1.1.3.2 Service Reachability

1435 Service reachability, as modeled in Section 4.2.2.3 enables service participants to locate and interact with
1436 one another. To support service reachability, the service description should indicate the **endpoints** (also
1437 modeled and defined in that section) to which a service consumer can direct messages to invoke actions
1438 and the protocol to be used for message exchange using that endpoint.

1439 As generally applied to an action, the endpoint is the conceptual location where one applies an action;
1440 with respect to service description, it is the actual address where a message is sent.

1441 4.1.1.3.3 Service Functionality

1442 While the service interface and service reachability are concerned with the mechanics of using a service,
1443 service functionality and performance metrics (discussed in Section 4.1.1.3.4) describe what can be
1444 expected as a result of interacting with a service. Service Functionality, shown in Figure 16 as part of the
1445 overall Service Description model and extended in Figure 18, is a clear expression of service function(s)
1446 and the real world effects of invoking the function. The Functions represent business activities in some
1447 domain that produce the desired real world effects.

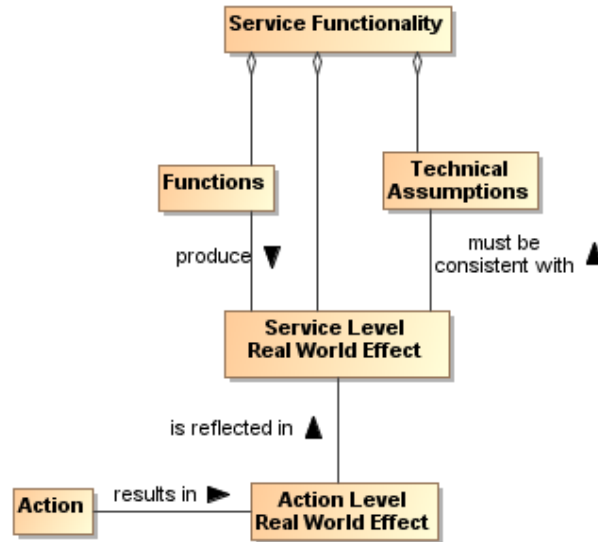


Figure 18 - Service Functionality

1448
1449

1450 The Service Functionality may also be limited by technical assumptions/constraints that underlie the
1451 effects that can result. Technical constraints are defined as domain specific restrictions and may express
1452 underlying physical limitations, such as flow speeds must be below sonic velocity or disk access that
1453 cannot be faster than the maximum for its host drive. Technical constraints are related to the underlying
1454 capability accessed by the service. In any case, the real world effects must be consistent with the
1455 technical assumptions/constraints.

1456 In Figure 16 and Figure 18, we specifically refer to the descriptions of **Service Level** and **Action Level**
1457 **Real World Effects**.

1458 **Service Level Real World Effect**

1459 A specific change in the **state** or the information returned as a result of interacting with a service.

1460 **Action Level Real World Effect**

1461 A specific change in the **state** or the information returned as a result of interacting through a
1462 specific action.

1463 Service description describes the service as a whole while the component aspects should contribute to
1464 that whole. Thus, while individual Actions may contribute to the real world effects to be realized from
1465 interaction with the service, there would be a serious disconnect for Actions to contribute real world
1466 effects that could not consistently be reflected in the Service Level Real World Effects and thus the
1467 Service Functionality. The relationship to Action Level Real World Effects and the implications on defining
1468 the scope of a service are discussed in Section 4.1.2.1.

1469 Elements of Service Functionality may be expressed as natural language text, reference an existing
1470 taxonomy of functions or other formal model.

1471 **4.1.1.3.4 Service Policies, Metrics, and Compliance Records**

1472 Policies prescribe the conditions and constraints for interacting with a service and impact the willingness
1473 to continue visibility with the other participants. Whereas technical constraints are statements of 'physical'
1474 fact, policies are subjective assertions made by the service provider (sometimes as passed on from
1475 higher authorities).

1476 The service description provides a central location for identifying what policies have been asserted by the
1477 service provider. The specific representation of the policy, e.g. in some formal policy language, is outside
1478 of the service description. The service description would reference the normative definition of the policy.

1479 Policies may also be asserted by other participants, as illustrated by the model shown in Figure 19.
1480 Policies that are generally applicable to any interaction with the service are asserted by the service
1481 provider and included in the Service Policies section of the service description.

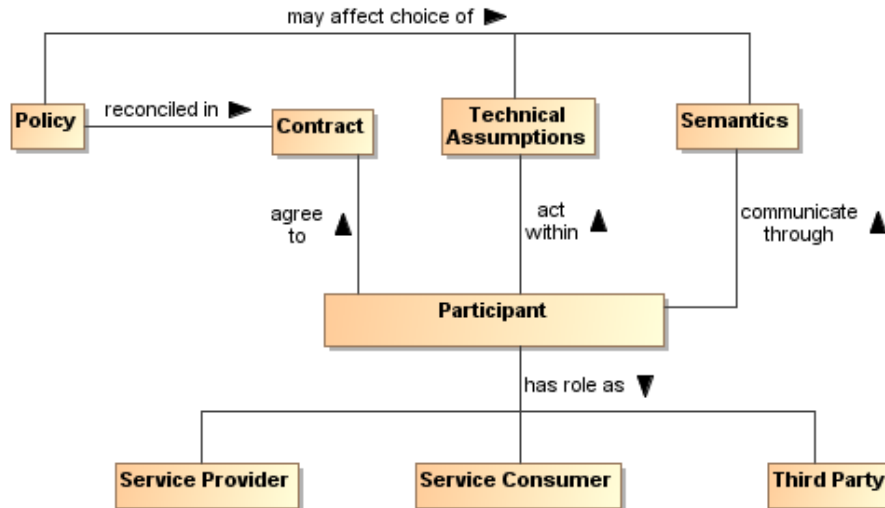


Figure 19 - Model for Policies and Contracts as related to Service Participants

1482
1483

1484 In Figure 19, we specifically refer to policies at the service level. In a similar manner to that discussed for
1485 Service Level vs. Action Level Real World Effects in Section 4.1.1.3.3, individual Actions may have
1486 associated policies stating conditions for performing the action, but these must be reflected in and be
1487 consistent with the policies made visible at the service level and thus the description of the service as a
1488 whole. The relationship to Action Level Policies and the implications on defining the scope of a service
1489 are discussed in Section 4.1.2.1.

1490 As noted in Figure 19, the policies asserted may be reflected as Technical Assumptions/Constraints that
1491 available services or their underlying capabilities must be capable of meeting; it may similarly affect the
1492 semantics that can be used. For example of the former, there may be a policy that specifies the surge
1493 capacity to be accommodated by a server, but a service that is not designed to make use of the larger
1494 server capacity would not satisfy the intent of the policy and would not be appropriate to use. For the
1495 latter, a policy may require that only services that support interaction via a community-sponsored
1496 vocabulary can be used.

1497 Contracts are agreements among the participants. The contract may reconcile inconsistent policies
1498 asserted by the participants or may specify details of the interaction. Service level agreements (SLAs) are
1499 one of the commonly used categories of contracts.

1500 The definition and later enforcement of policies and contracts are predicated on the potential for
1501 measurement; the relationships among the relevant concepts are shown in the model in Figure 20.
1502 Performance Metrics identify quantities that characterize the speed and quality of realizing the real world
1503 effects produced using the SOA service; in addition, policies and contracts may depend on
1504 nonperformance metrics, such as whether a license is in place to use the service. Some of these metrics
1505 may reflect the underlying capability, some metrics may reflect processing of the SOA service, and some
1506 metrics may include expected network overhead. The metrics should be carefully defined to avoid
1507 confusion in exactly what is being reported, for example, a case where the service processing time is
1508 reported as if it were the total time including the capability and network processing but is only measuring
1509 the service processing.

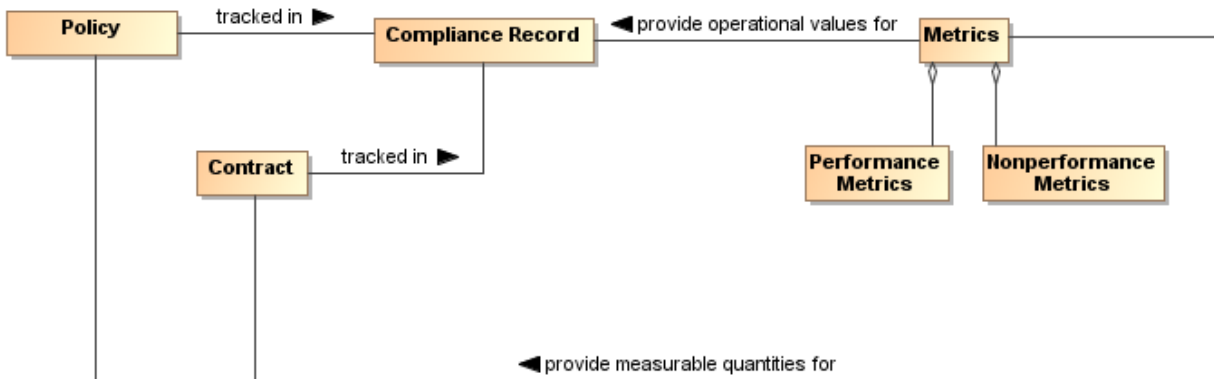


Figure 20 - Policies and Contracts, Metrics, and Compliance Records

1510
1511

1512 As with many quantities, the metrics associated with a service are not themselves defined by this Service
1513 Description Model because it is not known *a priori* which metrics are being collected or otherwise checked
1514 by the services, the SOA infrastructure, or other resources that participate in the SOA interactions.
1515 However, the service description should provide a placeholder (possibly through a link to an externally
1516 compiled list) for identifying which metrics are available and how these can be accessed.

1517 The use of metrics to evaluate compliance and the results of compliance evaluation are maintained in
1518 compliance records and the means to access the compliance records is included in the Service Policies
1519 portion of the service description. For example, the description may be in the form of static information
1520 (e.g. over the first year of operation, this service had a 91% availability), a link to a dynamically generated
1521 metric (e.g. over the past 30 days, the service has had a 93.3% availability), or access to a dynamic
1522 means to check the service for current availability (e.g., a ping). The relationship between service
1523 **presence** and the presence of the individual actions that can be invoked is discussed under Reachability
1524 in Section 4.2.2.3.

1525 Note, even when policies relate to the perspective of a single participant, policy compliance can be
1526 measured and policies may be enforceable without contractual agreement with other participants. While
1527 certain elements of contracts and contract compliance are likely private, public aspects of compliance
1528 should be reflected in the compliance record information referenced in the service description. This
1529 provides input to evidence that supports determining willingness as described in Section 3.2.5.1.

1530 4.1.2 Use of Service Description

1531 4.1.2.1 Service Description in support of Service Interaction

1532 If we assume we have awareness, the service participants must still establish willingness and presence to
1533 ensure full visibility (See Section 4.2) and to interact with the service. Service description provides
1534 necessary information for many aspects of preparing for and carrying through with interaction. Recall the
1535 fundamental definition of a SOA service as a mechanism to access an underlying capability; the service
1536 description describes this mechanism and its use. It lays the groundwork for what can occur, whereas
1537 service interaction comprises the specifics through which real-world effects are realized.

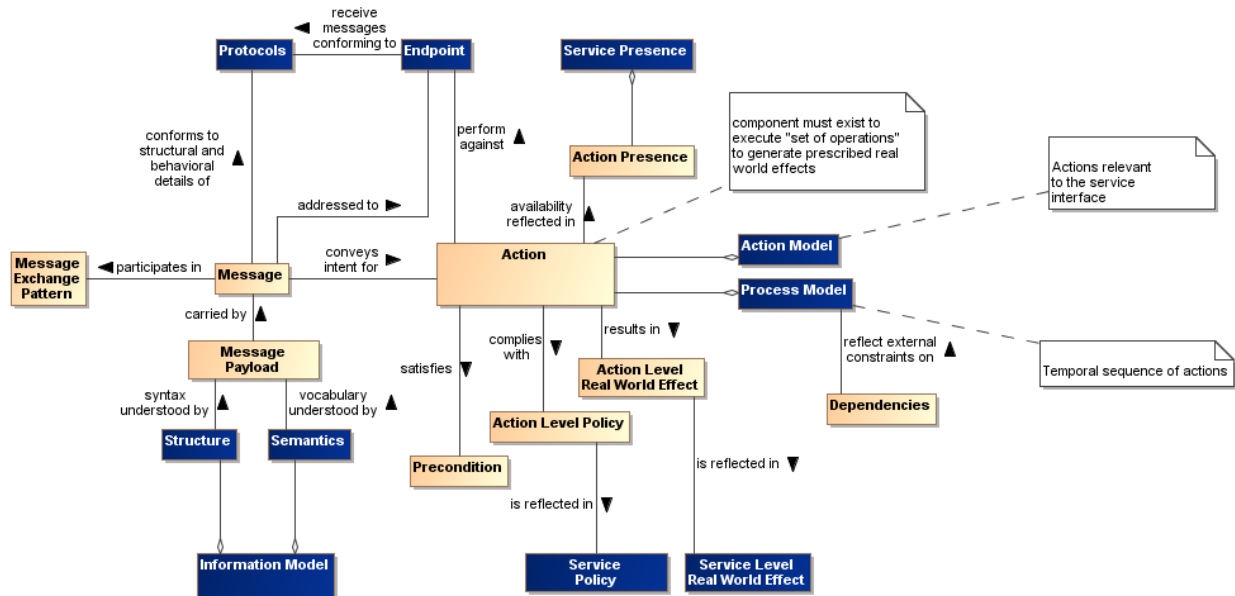


Figure 21 - Relationship between Action and Components of Service Description Model

1538
1539

1540 Figure 21 combines the models in the subsections of Section 4.1.1 to concisely relate action and the
 1541 relevant components of the Service Description model. The purpose of Figure 21 is to demonstrate that
 1542 the components of service description go beyond arbitrary documentation and form the critical set of
 1543 information needed to define the what and how of action. In Figure 21, the leaf nodes from Figure 16 are
 1544 shown in blue.

1545 Action is typically invoked via a Message where the structure and processing details of the message
 1546 conform to an identified Protocol and is directed to the address of the identified endpoint, and the
 1547 message payload conforms to the service Information Model.

1548 The availability of an action is reflected in the Action Presence and each Action Presence contributes to
 1549 the overall Service Presence; this is discussed further in Section 4.2.2.3. Each action has its own
 1550 endpoint and protocols are associated with the endpoint⁶. The endpoint and service presence are also
 1551 part of the service description.

1552 An action may have preconditions where a Precondition is something that must be in place before an
 1553 action can occur, e.g. confirmation of a precursor action. Whether preconditions are satisfied is evaluated
 1554 when an actor tries to perform the action and not before. Presence for an action means an actor can
 1555 initiate it and is independent of whether the preconditions are satisfied. However, the successful
 1556 completion of the action may depend on whether its preconditions were satisfied. The service as a whole
 1557 may provide fallback if a precondition is not met, and the service description may indicate functionality
 1558 without explicitly containing details of how preconditions are satisfied or otherwise mitigated.

1559 Analogous to the relationship between actions and preconditions, the Process Model may imply
 1560 Dependencies for succeeding steps in a process, e.g. that a previous step has successfully completed, or
 1561 may be isolated to a given step. An example of the latter would be a dependency that the host server has

⁶ This is analogous to a WSDL 2.0 interface operation (WSDL 1.1 portType) having one or more defined bindings and the service identifies the endpoints (WSDL 1.1 ports) corresponding to the bindings.

1562 scheduled maintenance and access attempts at these times would fail. Dependencies related to the
1563 process model do not affect the presence of a service although these may affect whether the business
1564 function successfully completes. The service as a whole may provide fallback if a dependency is not met,
1565 and the service description may indicate functionality without explicitly containing details of how
1566 dependencies are satisfied or otherwise mitigated.

1567 The conditions under which an action can be invoked may depend on policies associated with the action.
1568 The Action Level Policies must be reflected in (or subsumed by) the Service Policies because such
1569 policies may be critical to determining whether the conditions for use of the service are consistent with the
1570 policies asserted by the service consumer. For example, if an action requires interaction with another
1571 service and that other service has licensing requirements, then the service with such an action also has
1572 the same requirement. The Service Policies are included in the service description.

1573 Similarly, the result of invoking an action is one or more real world effects, and any Action Level Real
1574 World Effects must be reflected in the Service Level Real World Effect included in the service description.
1575 The unambiguous expression of action level policies and real world effects as service counterparts is
1576 necessary to adequately describe what constitutes the service interaction. For example, if an action
1577 allows for the tracking of customer preferences, then the service with such an action results in the same
1578 real world effect.

1579 An adequate service description must provide a consumer with information needed to determine if the
1580 service policies, the (business) functions, and service-level real world effects are of interest, and there is
1581 nothing in the technical constraints that preclude use of the service.

1582 Note at the service level, the business functions are not concerned with the action or process models.
1583 These models are detailed separately.

1584 The service description is not intended to be isolated documentation but rather an integral part of service
1585 use. Changes in service description should immediately be made known to consumers and potential
1586 consumers.

1587 **4.1.2.2 Description and Invoking Actions Against a Service**

1588 At this point, let us assume the descriptions were sufficient to establish willingness; see Section 4.2.2.2.
1589 Figure 21 indicates the service endpoint establishes where to actually carry out the interaction. This is
1590 where we start considering the action and process models.

1591 The action model identifies the multiple actions an actor can perform against a service and the actor
1592 would perform these in the context of the process model as specified or referenced under the Service
1593 Interface Description portion of Service Description. For a given business function, there is a
1594 corresponding process model, where any process model may involve multiple actions. From the above
1595 discussion of model elements of description we may conclude (1) actions have reachability information,
1596 including endpoint and presence, (2) presence of service is some aggregation of presence of its actions,
1597 (3) action preconditions and service dependencies do not affect presence although these may affect
1598 successful completion.

1599 Having established visibility, the interaction can proceed. Given a business function, the consumer knows
1600 what will be accomplished (the service functionality), the conditions under which interaction will proceed
1601 (service policies), and the process that must be followed (the process model). The remaining question is
1602 how the description information for structure and semantics enable interaction.

1603 We have established the importance of the process model in identifying relevant actions and their
1604 sequence. Interaction proceeds through messages and thus it is the syntax and semantics of the
1605 messages with which we are here concerned. A common approach is to define the structure and
1606 semantics that can appear as part of a message; then assemble the pieces into messages; and,
1607 associate messages with actions. Actions make use of structure and semantics as defined in the
1608 information model to describe its legal messages.

1609 The process model identifies actions to be performed against a service and the sequence for performing
1610 the actions. For a given action, the Reachability portion of description indicates the protocol bindings that
1611 are available, the endpoint corresponding to a binding, and whether there is presence at that endpoint. An
1612 interaction is through the exchange of messages that conform to the structure and semantics defined in
1613 the information model and the message sequence conforming to the action's identified MEP. The result is

1614 some portion of the real world effect that must be assessed and/or processed (e.g. if an error exists, that
1615 part that covers the error processing would be invoked).

1616 **4.1.2.3 The Question of Multiple Business Functions**

1617 Action level effects and policies must be reflected at the service level for service description to support
1618 visibility.

1619 It is assumed that a SOA service represents an identifiable business function to which policies can be
1620 applied and from which desired business effects can be obtained. While contemporary discussions of
1621 SOA services and supporting standards do not constrain what actions or combinations of actions can or
1622 should be defined for a service, the SOA-RAF considers the implications of service description in defining
1623 the range of actions appropriate for an individual SOA service.

1624 Consider the situation if a given SOA service is the mechanism for access to multiple independent (but
1625 loosely related) business functions. These are not multiple effects from a single function but multiple
1626 functions with potentially different sets of effects for each function. A service can have multiple actions
1627 that an actor may perform against it, and this does not change with multiple business functions. As an
1628 individual business function corresponds to a process model, so multiple business functions imply
1629 multiple process models. The same action may be used in multiple process models but the aggregated
1630 service presence would be specific to each business function because the components being aggregated
1631 may be different between process models. In summary, for a service with multiple business functions,
1632 each function has (1) its own process model and dependencies, (2) its own aggregated presence, and (3)
1633 possibly its own list of policies and real world effects.

1634 A common variation on this theme is for a single service to have multiple endpoints for different levels of
1635 quality of service (QoS), e.g. Gold, Silver, and Bronze. Different QoS imply separate statements of policy,
1636 separate endpoints, possibly separate dependencies, and so on. One could say the QoS variation does
1637 not require this because there can be a single QoS policy that encompasses the variations, and all other
1638 aspects of the service would be the same except for the endpoint used for each QoS. However, the
1639 different aspects of policy at the service level would need to be mapped to endpoints, and this introduces
1640 an undesirable level of coupling across the elements of description. In addition, it is obvious that
1641 description at the service level can become very complicated if the number of combinations is allowed to
1642 grow.

1643 One could imagine a service description that is basically a container for action descriptions, where each
1644 action description is self-contained; however, this would lead to duplication of description components
1645 across actions. If common description components are factored, this either is limited to components
1646 common across all actions or requires complicated tagging to capture the components that often but do
1647 not universally apply.

1648 If a provider cannot describe a service as a whole but must describe every action, this leads to the
1649 situation where it may be extremely difficult to construct a clear and concise service description that can
1650 effectively support discovery and use without tedious logic to process the description and assemble the
1651 available permutations. In effect, if adequate description of an action begins to look like description of a
1652 service, it may be best to have it as a separate service.

1653 Recall, more than one service can access the same underlying capability, and this is appropriate if a
1654 different real world effect is to be exposed. Along these lines, one can argue that different QoS are
1655 different services because getting a response in one minute rather than one hour is more than a QoS
1656 difference; it is a fundamental difference in the business function being provided.

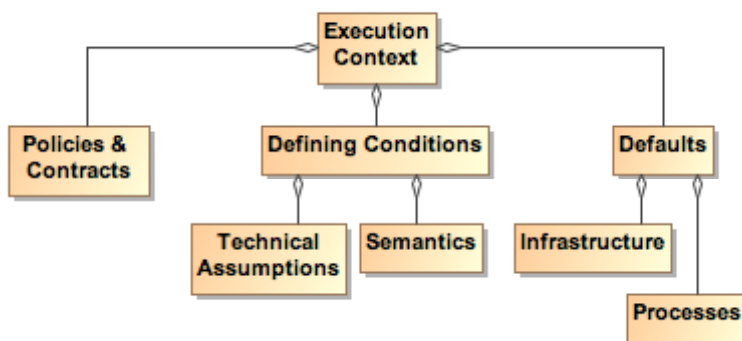
1657 As a best practice, the criterion for whether a service is appropriately scoped may be the ease or difficulty
1658 in creating an unambiguous service description. A consequence of having tightly-scoped services is there
1659 will likely be a greater reliance on combining services, i.e. more fundamental business functions, to create
1660 more advanced business functions. This is consistent with the principles of service oriented architecture
1661 and is the basic position of this Reference Architecture Foundation, although not an absolute
1662 requirement. Combining services increases the reliance on understanding and implementing the concepts
1663 of orchestration, choreography, and other approaches yet to be developed; these are discussed in more
1664 detail in section 4.4 Interacting with Services.

1665 **4.1.2.4 Service Description, Execution Context, and Service Interaction**

1666 The service description must provide sufficient information to support service visibility, including the
1667 willingness of service participants to interact. However, the corresponding descriptions for providers and
1668 consumers may both contain policies, technical assumptions, constraints on semantics, and other
1669 technical and procedural conditions that must be aligned to define the terms of willingness. The
1670 agreements that encapsulate the necessary alignment form the basis upon which interactions may
1671 proceed – in the Reference Model, this collection of agreements and the necessary environmental
1672 support establish the execution context.

1673 To illustrate execution context of a service interaction, consider a Web-based system for timecard entry.
1674 For an employee onsite at an employer facility, the execution context requires a computer connected to
1675 the local network and the employee must enter their network ID and password. Relevant policies include
1676 that the employee must maintain the most recent anti-virus software and virus definitions for any
1677 computer connected to the network.

1678 For the same employee connecting from offsite, the execution context specifies the need for a computer
1679 with installed VPN software and a security token to negotiate the VPN connection. The execution context
1680 also includes proxy settings as needed to connect to the offsite network. The employee must still comply
1681 with the requirements for onsite computers and access, but the offsite execution context includes
1682 additional items before the employee can access the same underlying capability and realize the same
1683 real world effects, i.e. the timecard entries.



1684
1685

Figure 22 - Execution Context

1686 Figure 22 shows a few broad categories found in execution context. These are not meant to be
1687 comprehensive. Other items may need to be included to provide a sufficient description of the interaction
1688 conditions. Any other items not explicitly noted in the model but which are needed to set the environment,
1689 would also be included in the execution context.

1690 While the execution context captures the conditions under which interaction can occur, it does not capture
1691 the specific service invocations that do occur in a specific interaction. A service interaction as modeled in
1692 Figure 23 introduces the concept of an Interaction Description that is composed of both the Execution
1693 Context and an Interaction Log. The execution context specifies the set of conditions under which the
1694 interaction occurs and the interaction log captures the sequence of service interactions that occur within
1695 the execution context. This sequence should follow the Process Model but can include details beyond
1696 those specified there. For example, the Process Model may specify an action that results in identifying a
1697 data source, and the identified source is used in a subsequent action. The Interaction Log would record
1698 the specific data source used.

1699 The execution context can be thought of as a container in which the interaction occurs and the interaction
1700 log captures what happens inside the container. This combination is needed to support auditability and
1701 repeatability of the interactions.

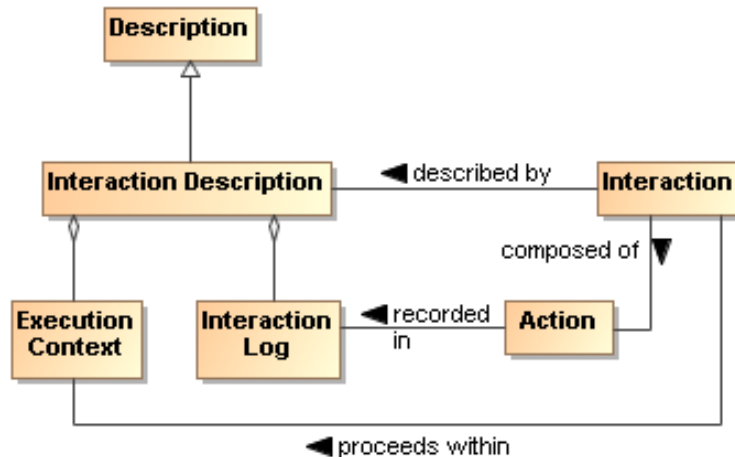


Figure 23 - Interaction Description

1702
1703

1704 SOA allows flexibility to accomplish both repeatability and reusability. In facilitating reusability, a service
1705 can be updated without disrupting the customer experience of the service. So, Google can improve their
1706 ranking algorithm without notifying the customer about the details of the update.

1707 However, it may also be vital for the consumer to be able to recreate past results or to generate
1708 consistent results in the future, and information such as what conditions, which services, and which
1709 versions of those services were used is indispensable in retracing one's path. The interaction log is a
1710 critical part of the resulting real world effects because it defines how the effects were generated and
1711 possibly the meaning of observed effects. This increases in importance as dynamic composability
1712 becomes more feasible. In essence, a result has limited value if one does not know how it was generated.

1713 The interaction log provides a detailed trace for a specific interaction, and its reuse is limited to
1714 duplicating that interaction. An execution context can act as a template for identical or similar interactions.
1715 Any given execution context can define the conditions of future interactions.

1716 Such uses of execution context imply (1) a standardized format for capturing execution context and (2) a
1717 subclass of general description could be defined to support visibility of saved execution contexts. The
1718 specifics of the relevant formats and descriptions are beyond the scope of this document.

1719 A service description is unlikely to track interaction descriptions or the constituent execution contexts or
1720 interaction logs that include mention of the service. However, as appropriate, linking to specific instances
1721 of either of these could be done through associated annotations.

1722 4.1.3 Relationship to Other Description Models

1723 While the representation shown in Figure 15 is derived from considerations related to service description,
1724 it is acknowledged that other metadata standards are relevant and should, as possible, be incorporated
1725 into this work. Two standards of particular relevance are the Dublin Core Metadata Initiative (DCMI)
1726 **[DCMI]** and ISO 11179 **[ISO 11179]**, especially Part 5.

1727 When the service description (or even the general description class) is considered as the DCMI
1728 'resource', Figure 15 aligns nicely with the DCMI resource model. While some differences exist, these are
1729 mostly in areas where DCMI goes into detail that is considered beyond the scope of the current
1730 Reference Architecture Foundation. For example, DCMI defines classes of 'shared semantics' whereas
1731 this Reference Architecture Foundation considers that an identification of relevant semantic models is
1732 sufficient. Likewise, the DCMI Description Model goes into the details of possible syntax encodings
1733 whereas for the Reference Architecture Framework it is sufficient to identify the relevant formats.

1734 With respect to ISO 11179 Part 5, the metadata fields defined in that reference may be used without
1735 prejudice as the properties in Figure 15. Additionally, other defined metadata sets may be used by the
1736 service provider if the other sets are considered more appropriate, i.e. it is fundamental to this reference
1737 architecture to identify the need and the means to make vocabulary declarations explicit but it is beyond
1738 the scope to specify which vocabularies are to be used. In addition, the identification of domain of the

1739 properties and range of the values has not been included in the current Reference Architecture
1740 discussion, but the text of ISO 11179 Part 5 can be used consistently with the model prescribed in this
1741 document.

1742 Description as defined here considers a wide range of applicability and support of the principles of service
1743 oriented architecture. Other metadata models can be used in concert with the model presented here
1744 because most of these focus on a finer level of detail that is outside the present scope, and so provide a
1745 level of implementation guidance that can be applied as appropriate.

1746 4.1.4 Architectural Implications

1747 The definition of service description has numerous architectural implications for the SOA ecosystem:

- 1748 • The real world effects that the service description definition support must be consistent with the
1749 technical assumptions/constraints. In particular, any Action Level Real World Effect **MUST** be
1750 reflected in the Service Level Real World Effect included in the sedcription.
- 1751 • The service description definition changes over time and its contents will reflect changing
1752 requirements and context. The service description definition **MUST** therefore have:
 - 1753 ○ mechanisms to support the storage, referencing, and access to normative definitions of
1754 one or more versioning schemes that may be applied to identify different aggregations of
1755 descriptive information, where the different schemes may be versions of a versioning
1756 scheme itself;
 - 1757 ○ configuration management mechanisms to capture the contents of each aggregation and
1758 apply a unique identifier in a manner consistent with an identified versioning scheme;
 - 1759 ○ one or more mechanisms to support the storage, referencing, and access to conversion
1760 relationships between versioning schemes, and the mechanisms to carry out such
1761 conversions.
- 1762 • Description makes use of defined semantics, where the semantics may be used for
1763 categorization or providing other property and value information for description classes. In such
1764 cases, the service description **MUST** have:
 - 1765 ○ semantic models that provide normative descriptions of the utilized terms, where the
1766 models may range from a simple dictionary of terms to an ontology showing complex
1767 relationships and capable of supporting enhanced reasoning;
 - 1768 ○ mechanisms to support the storage, referencing, and access to these semantic models;
 - 1769 ○ configuration management mechanisms to capture the normative description of each
1770 semantic model and to apply a unique identifier in a manner consistent with an identified
1771 versioning scheme;
 - 1772 ○ one or more mechanisms to support the storage, referencing, and access to conversion
1773 relationships between semantic models, and the mechanisms to carry out such
1774 conversions.
- 1775 • Once awareness exists, the service description **MUST** provide sufficient access to information so
1776 that service participants are able to establish willingness and presence to ensure full visibility
1777 (See Section 4.2).
- 1778 • The Service Description **MUST** provide a consumer with information needed to: determine the
1779 service functionality; the conditions under which interaction can proceed (service policies and
1780 process model); the intended Service Level Real World Effects; any technical constraints that
1781 might preclude use of the service.
- 1782 • Changes in Service Description **SHOULD** be made available immediately to actual and potential
1783 consumers.
- 1784 • Actions **MAY** have associated policies stating conditions for performing the action, but these
1785 **MUST** be reflected in and be consistent with the policies made visible at the service level and
1786 thus the description of the service as a whole.
- 1787 • Policies asserted **MAY** be reflected as Technical Assumptions/Constraints that available services
1788 or their underlying capabilities **MUST** be capable of meeting.
- 1789 • Descriptions include reference to policies defining conditions of use. In this sense, policies are
1790 also resources that need to be visible, discoverable, and accessible. The service description (as
1791 also enumerated under governance) **MUST** have:

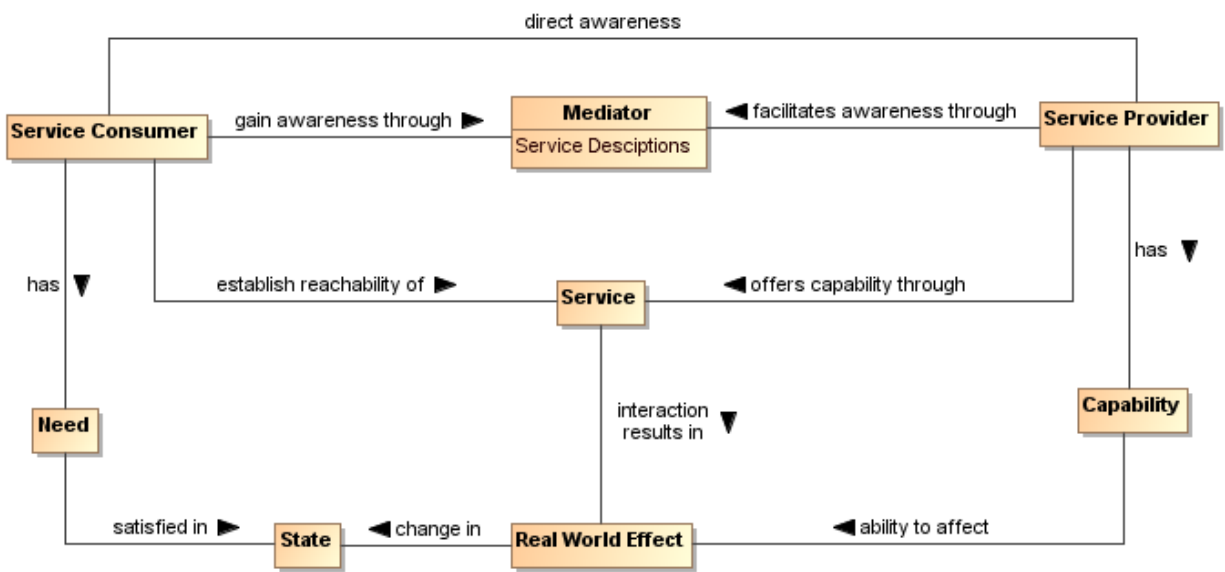
- 1792 ○ description of policies, including a unique identifier for the policy and a sufficient,
1793 preferably machine processable, representation of the meaning of terms used to describe
1794 the policy, its functions, and its effects;
- 1795 ○ a method to enable searching for policies that best meet the search criteria specified by
1796 the service participant; where the discovery mechanism has access to the individual
1797 policy descriptions, possibly through some repository mechanism;
- 1798 ○ accessible storage of policies and policy descriptions, so service participants can access,
1799 examine, and use the policies as defined.
- 1800 • Descriptions include references to metrics that describe the operational characteristics of the
1801 subjects being described. The service description definition (as also partially enumerated under
1802 governance) **MUST** have:
 - 1803 ○ infrastructure monitoring and reporting information on SOA resources;
 - 1804 ○ possible interface requirements to make accessible metrics information generated;
 - 1805 ○ mechanisms to catalog and enable discovery of which metrics are available for a
1806 described resources and information on how these metrics can be accessed;
 - 1807 ○ mechanisms to catalog and enable discovery of compliance records associated with
1808 policies and contracts that are based on these metrics.
- 1809 • Descriptions of the interactions are important for enabling auditability and repeatability, thereby
1810 establishing a context for results and support for understanding observed change in performance
1811 or results. Thus, the service description definition **MUST** have:
 - 1812 ○ one or more mechanisms to capture, describe, store, discover, and retrieve interaction
1813 logs, execution contexts, and the combined interaction descriptions;
 - 1814 ○ one or more mechanisms for attaching to any results the means to identify and retrieve
1815 the interaction description under which the results were generated.
- 1816 • Descriptions may capture very focused information subsets or can be an aggregate of numerous
1817 component descriptions. Service description is an example of an aggregate for which manual
1818 maintenance of the whole would not be feasible. Thus, the service description definition **MUST**
1819 have:
 - 1820 ○ tools to facilitate identifying description elements that are to be aggregated to assemble
1821 the composite description;
 - 1822 ○ tools to facilitate identifying the sources of information to associate with the description
1823 elements;
 - 1824 ○ tools to collect the identified description elements and their associated sources into a
1825 standard, referenceable format that can support general access and understanding;
 - 1826 ○ tools to automatically update the composite description as the component sources
1827 change, and to consistently apply versioning schemes to identify the new description
1828 contents and the type and significance of change that occurred.
- 1829 • The description is the source of vital information in establishing willingness to interact with a
1830 resource, reachability to make interaction possible, and compliance with relevant conditions of
1831 use. Thus, the service description definition **MUST** have:
 - 1832 ○ one or more discovery mechanisms that enable searching for described resources that
1833 best meet the criteria specified by a service participant;
 - 1834 ○ tools to appropriately track use of the descriptions by service participants and notify them
1835 when a new version of the description is available.
- 1836 • The service description **MUST** provide sufficient information to support service visibility, including
1837 the willingness of service participants to interact. However, the corresponding descriptions for
1838 providers and consumers may both contain policies, technical assumptions, constraints on
1839 semantics, and other technical and procedural conditions that must be aligned to define the terms
1840 of willingness

1841 4.2 Service Visibility Model

1842 One of the key requirements for participants interacting with each other in the context of a SOA
1843 ecosystem is achieving visibility: before services can interoperate, the participants have to be visible to
1844 each other using whatever means are appropriate. The Reference Model analyzes visibility in terms of
1845 awareness, willingness, and reachability. In this section, we explore how visibility may be achieved.

1846 **4.2.1 Visibility to Business**

1847 The relationship of visibility to the SOA ecosystem encompasses both human social structures and
 1848 automated IT mechanisms. Figure 24 depicts a business setting that is a basis for visibility as related to
 1849 the Social Structure Model (Figure 3) in the Participation in a SOA Ecosystem view (see Section 3.1).
 1850 Participants acting in the roles of service consumers and service providers may have awareness of each
 1851 other directly or gain such awareness through some third party acting in the role of mediator, what we
 1852 refer to as **mediated awareness**. A consumer's willingness to use a service is reflected by the
 1853 consumer's presumption of satisfying goals and needs as these compare with information provided in the
 1854 service description. Service providers offer capabilities that have real world effects that result in a change
 1855 in state. Reachability of the service by the consumer may lead to interactions that change the state of the
 1856 SOA ecosystem. The consumer can measure the change of state to determine if the claims made by
 1857 description and the real world effects of consuming the service meet the consumer's needs.
 1858 Those acting in the roles of consumers, providers, or mediators may reside within a single ownership
 1859 boundary or interactions between them may cross ownership boundaries.



1860
 1861 *Figure 24 - Visibility to Business*

1862 Visibility and interoperability in a SOA ecosystem requires more than location and interface information. A
 1863 meta-model for this broader view of visibility is depicted in Section 4.1. In addition to providing improved
 1864 awareness of service capabilities through description of information such as reachability, behavior
 1865 models, information models, functionality, and metrics, the service description may identify policies
 1866 valuable for determination of willingness to interact.

1867 A mediator using service descriptions may provide event notifications to both consumers and providers
 1868 about information relating to the descriptions. One example of this is a publish/subscribe model where the
 1869 mediator allows consumers to subscribe to service description version changes made by the provider.
 1870 Likewise, the mediator may provide notifications to the provider of consumers that have subscribed to
 1871 service description updates.

1872 Another important characteristic of a SOA ecosystem is the ability to narrow visibility to trusted members
 1873 within a social structure. Mediators for awareness may provide policy based access to service
 1874 descriptions allowing for the dynamic formation of awareness between trusted members.

1875 **4.2.2 Visibility**

1876 Attaining visibility is described in terms of steps that lead to visibility. Different participant communities can
 1877 bring different contexts for visibility within a single social structure, and the same general steps can be
 1878 applied to each of the contexts to accomplish visibility.

1879 Attaining SOA visibility requires

1880 • service description creation and maintenance,
1881 • processes and mechanisms for achieving awareness of and accessing descriptions,
1882 • processes and mechanisms for establishing willingness of participants,
1883 • processes and mechanisms to determine reachability.
1884 Visibility may occur in stages, i.e. a participant can become aware enough to look or ask for further
1885 description, and with this description, the participant can decide on willingness, possibly requiring
1886 additional description. For example, if a potential consumer has a need for a tree cutting (business)
1887 service, the consumer can use a web search engine to find web sites of providers. The web search
1888 engine (a mediator) gives the consumer links to relevant web pages and the consumer can access those
1889 descriptions. For those prospective providers that satisfy the consumer's criteria, the consumer's
1890 willingness to interact increases. The consumer may contact several tree services to get detailed cost
1891 information (or arrange for an estimate) and may ask for references (further description). The consumer is
1892 likely to establish full visibility and proceed with interaction with the tree service that mutually establishes
1893 visibility.

1894 **4.2.2.1 Awareness**

1895 An important means for one participant to be aware of another is to have access to a description of that
1896 participant and for the description to be sufficiently complete to support the other requirements of visibility.

1897 Awareness can be established without any action on the part of the target participant other than the target
1898 providing appropriate descriptions. Awareness is often discussed in terms of consumer awareness of
1899 providers but the concepts are equally valid for provider awareness of consumers.

1900 Awareness can be decomposed into: creating the descriptions, making them available, and discovering
1901 the descriptions. Discovery can be initiated or it can be by notification.

1902 Achieving awareness in a SOA ecosystem can range from word of mouth to formal service descriptions in
1903 a standards-based registry/repository. Some other examples of achieving awareness in a SOA
1904 ecosystem are the use of a web page containing description information, email notifications of
1905 descriptions, and document based descriptions.

1906 Direct awareness occurs between a consumer and provider without the use of a third party. Mediated
1907 awareness, on the other hand, is provided by a third party participant to one or more providers or
1908 consumers of one or more services. A registry/repository can provide such awareness as can a Web
1909 page displaying similar information.

1910 Direct awareness may be the result of having previously established an execution context or may include
1911 determining the presence of services and then querying the service directly for description. As an
1912 example, a priori visibility of some sensor device may provide the means for interaction or a query for
1913 standardized sensor device metadata may be broadcast to multiple locations. If acknowledged, the
1914 service interface for the device may directly provide description to a consumer so the consumer can
1915 determine willingness to interact.

1916

1917 **4.2.2.1.1 Mediated Awareness**

1918 Mediated awareness ideally promotes simplification of the overall services infrastructure. As an example,
1919 the telephone book is a mediating registry where individuals perform manual searches to locate services
1920 (i.e. the yellow pages). The telephone book is also a mediated registry for solicitors to find and notify
1921 potential customers (i.e. the white pages).

1922 The benefits for large and dynamic numbers of service consumers and service providers, of utilizing the
1923 awareness mediator typically far outweigh the management issues associated with it. Some of the
1924 benefits of mediated service awareness are:

- 1925 • Potential service consumers have a known location for searching thereby eliminating needless
1926 and random searches
- 1927 • Typically a consortium of interested parties (or a sufficiently large corporation) serves as the host
1928 of the mediation facility

- 1929 • Standardized tools and methods can be developed and promulgated to promote interoperability and ease of use.
- 1930
- 1931 However, mediated awareness can have some risks associated with it:
- 1932 • A single point of failure. If the awareness mediator fails then a large number of service providers and consumers are potentially adversely affected.
- 1933
- 1934 • A single point of control. If the awareness mediator is owned by, or controlled by, someone other than the service consumers and/or providers then the latter may be put at a competitive disadvantage based on policies of the discovery provider.
- 1935
- 1936

4.2.2.1.2 Awareness in Complex Social Structures

1937 Awareness applies to one or more social structures where there is at least one description provider and one description consumer. Awareness may occur within the same social structure or across social structures.

1941 In Figure 25, awareness can be between a limited set of consumers and providers within a single social structure. Within a social structure, awareness can be encouraged or restricted through policies and these policies can affect participant willingness. The information about policies should be incorporated in the relevant descriptions. Additionally, the conditions for establishing contracts are governed within a social structure.

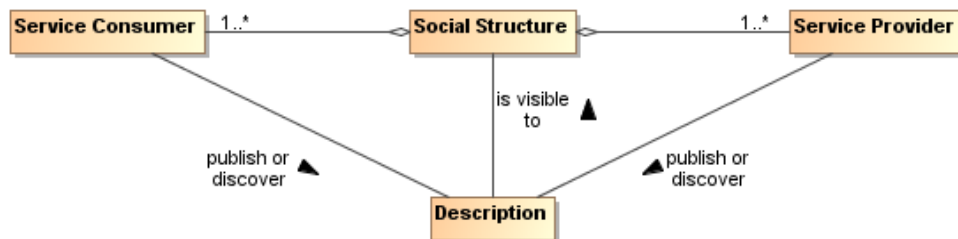


Figure 25 - Awareness in a SOA Ecosystem

1946 IT policy/contract mechanisms can be used by visibility mechanisms to provide awareness between social structures, including trust mechanisms to enable awareness between trusted social structures. For example, government organizations may want to limit awareness of an organization's services to specific communities of interest.

1952 Another common business model for awareness is maximizing awareness to those within the social structure, the traditional market place business model. A centralized awareness-mediator often arises as a provider for this global visibility, a gatekeeper of visibility so to speak. For example, Google is a centralized awareness-mediator for accessing information on the web. As another example, television networks have centralized entities providing a level of awareness to communities that otherwise could not be achieved without going through the television network.

1958 However, mediators have motivations, and they may be selective in which information they choose to make available to potential consumers. For example, in a secure environment, the mediator may enforce security policies and make information selectively available depending on the security clearance of the consumers.

4.2.2.2 Willingness

1963 Having achieved awareness, participants use descriptions to help determine their willingness to interact with another participant. Both awareness and willingness are determined prior to consumer/provider interaction.

1966 By establishing a willingness to interact within a particular social structure (see Section 3.2.5.1), the social structure provides the participant access to capabilities based on conditions the social structure finds appropriate for its context. The participant can use these capabilities to satisfy goals and objectives as specified by the participant's needs.

1970 Information used to determine willingness is provided by Description (see Section 4.1.1). Information referenced by Description may come from many sources. For example, a mediator for descriptions may

1972 provide 3rd party annotations for reputation. Another source for reputation may be a participant's own
 1973 history of interactions with another participant. The contribution of real world effects to providing evidence
 1974 and establishing the reputation of a participant is discussed with relation to Figure 9.

1975 A participant inspects functionality for potential satisfaction of needs. Identity is associated with any
 1976 participant, however, identity may or may not be verified. If available, participant reputation may be a
 1977 deciding factor for willingness to interact. Policies and contracts referenced by the description may be
 1978 particularly important to determine the agreements and commitments required for business interactions.
 1979 Provenance may be used for verification of authenticity of a resource.

1980 Mechanisms that aid in determining willingness make use of the artifacts referenced by descriptions of
 1981 services. Mechanisms for establishing willingness could be as simple as rendering service description
 1982 information for human consumption to automated evaluation of functionality, policies, and contracts by a
 1983 rules engine. The rules engine for determining willingness could operate as a policy decision procedure
 1984 as defined in Section 4.4.

1985 4.2.2.3 Reachability

1986 Reachability involves knowing the endpoint, protocol, and presence of a service. At a minimum,
 1987 reachability requires information about the location of the service and the protocol describing the means
 1988 of communication.

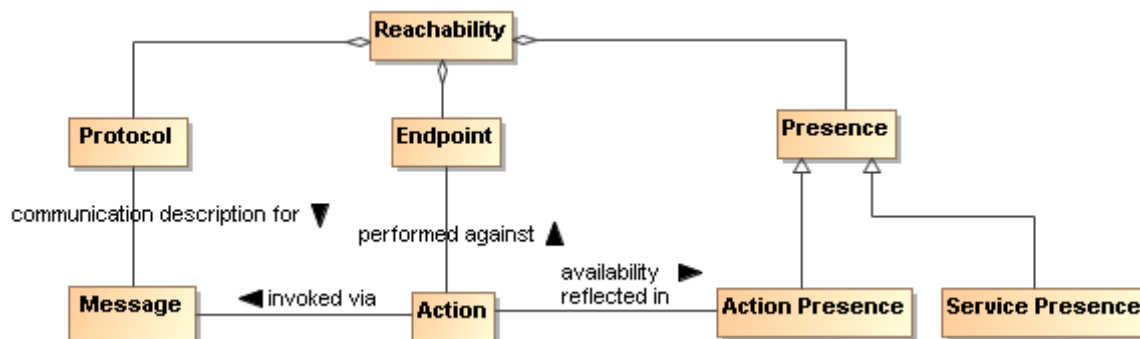


Figure 26 - Service Reachability

1989

1990

1991 **Endpoint**

1992 A reference-able entity, processor or **resource** against which an **action** can be performed.

1993 **Protocol**

1994 A structured means by which details of a service interaction mechanism are defined.

1995 **Presence**

1996 The measurement of reachability of a service at a particular point in time.

1997 A protocol defines a structured method of communication. Presence is determined by interaction through
 1998 a communication protocol. Presence may not be known in many cases until the interaction begins. To
 1999 overcome this problem, IT mechanisms may make use of presence protocols to provide the current
 2000 up/down status of a service.

2001 Service reachability enables service participants to locate and interact with one another. Each action may
 2002 have its own endpoint and also its own protocols associated with the endpoint and whether there is
 2003 presence for the action through that endpoint. Presence of a service is an aggregation of the presence of
 2004 the service's actions, and the service level may aggregate to some degraded or restricted presence if
 2005 some action presence is not confirmed. For example, if error processing actions are not available, the
 2006 service can still provide required functionality if no error processing is needed. This implies reachability
 2007 relates to each action as well as applying to the service/business as a whole.

2008 4.2.3 Architectural Implications

2009 Visibility in a SOA ecosystem has the following architectural implications on mechanisms providing
2010 support for awareness, willingness, and reachability:

- 2011 • Mechanisms providing support for awareness **MUST** have the following minimum capabilities:
 - 2012 ○ creation of Description, preferably conforming to a standard Description format and
 - 2013 structure;
 - 2014 ○ publishing of Description directly to a consumer or through a third party mediator;
 - 2015 ○ discovery of Description, preferably conforming to a standard for Description discovery;
 - 2016 ○ notification of Description updates or notification of the addition of new and relevant
 - 2017 Descriptions;
 - 2018 ○ classification of Description elements according to standardized classification schemes.
- 2019 • In a SOA ecosystem with complex social structures, awareness **MAY** be provided for specific
2020 communities of interest. The architectural mechanisms for providing awareness to communities
2021 of interest **MUST** support:
 - 2022 ○ policies that allow dynamic formation of communities of interest;
 - 2023 ○ trust that awareness can be provided for and only for specific communities of interest.
- 2024 • The architectural mechanisms for determining willingness to interact **MUST** support:
 - 2025 ○ verification of identity and credentials of the provider and/or consumer;
 - 2026 ○ access to and understanding of description;
 - 2027 ○ inspection of functionality and capabilities;
 - 2028 ○ inspection of policies and/or contracts.
- 2029 • The architectural mechanisms for establishing reachability **MUST** support:
 - 2030 ○ the location or address of an endpoint;
 - 2031 ○ verification and use of a service interface by means of a communication protocol;
 - 2032 ○ determination of presence with an endpoint which may only be determined at the point of
 - 2033 interaction but may be further aided by the use of a presence protocol for which the
 - 2034 endpoints actively participate.

2035 4.3 Interacting with Services Model

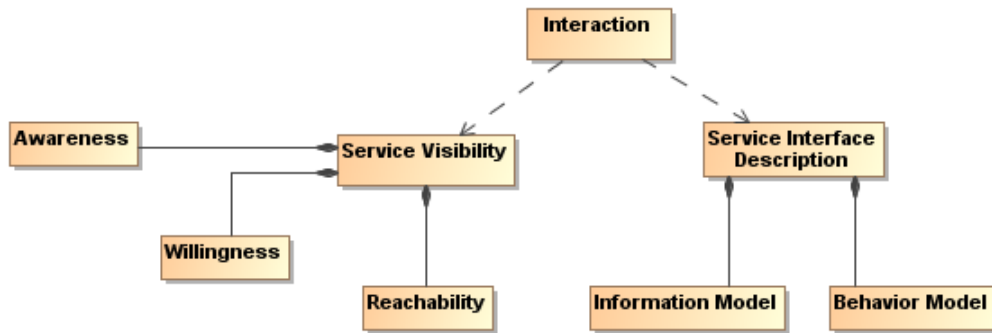
2036 Interaction is the activity involved in using a service to access capability in order to achieve a particular
2037 desired real world effect, where real world effect is the actual result of using a service. An interaction can
2038 be characterized by a sequence of communicative actions. Consequently, interacting with a service, i.e.
2039 participating in joint action with the service—usually accomplished by a series of message exchanges—
2040 involves individual actions performed by both the service and the consumer.⁷ Note that a participant (or
2041 delegate acting on behalf of the participant) can be the sender of a message, the receiver of a message,
2042 or both.

2043 4.3.1 Interaction Dependencies

2044 Recall from the Reference Model that service visibility is the capacity for those with needs and those with
2045 capabilities to be able to interact with each other, and that the service interface is the means by which the
2046 underlying capabilities of a service are accessed. Ideally, the details of the underlying service

⁷ In order for multiple actors to participate in a joint action, they must each act according to their role within the joint action. For SOA-based systems, this is achieved through a message exchange style of communication. The concept of “joint action” is further described in Section 3.3.2.

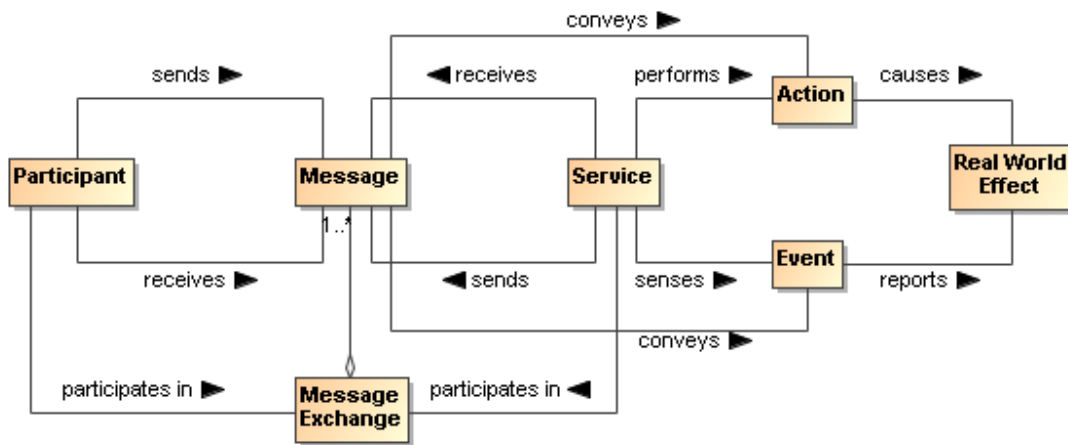
2047 implementation are abstracted away by the service interface. (Service) interaction therefore has a direct
 2048 dependency on the visibility of the service as well as its implementation-neutral interface (see Figure 27).
 2049 Service visibility is composed of awareness, willingness, and reachability, and these are discussed in
 2050 Section 4.2. The information related to the service interface description is discussed in Section 4.1.1.3.1,
 2051 and the specifics of interaction are detailed in the remainder of Section 4.3. Service visibility is modeled in
 2052 Section 4.2.2.



2053
 2054 *Figure 27 - Interaction dependencies*

2055 **4.3.2 Actions and Events**

2056 The SOA-RAF uses message exchange between service participants to denote actions performed
 2057 against and by the service, and to denote events that report on real world effects that are caused by the
 2058 service actions. A visual model of the relationship between these concepts is shown in Figure 28.



2059
 2060 *Figure 28 - A 'message' denotes either an action or an event*

2061 Both actions and events, realized by the SOA services, are denoted by the messages. The Reference
 2062 Model states that the action model characterizes the “permissible set of actions that may be invoked
 2063 against a service.” We extend that notion here to include events and that messages are intended for
 2064 invoking actions or for notification of events.

2065 In Section 3.3.2 we saw that participants interact with each other in order to participate in joint actions. A
2066 joint action is not itself the same thing as the result of the joint action. When a joint action is participated in
2067 with a service, the real world effect that results may be reported in the form of an event notification.

2068 4.3.3 Message Exchange

2069 *Message exchange* is the means by which service participants (or their delegates) interact with each
2070 other. There are two primary modes of interaction: joint actions that cause real world effects and
2071 notification of events that report real world effects⁸.

2072 A message exchange is used to affect an action when the messages contain the appropriately formatted
2073 content, are directed towards a particular action in accordance with the action model, and the delegates
2074 involved interpret the message appropriately.

2075 A message exchange is also used to communicate event notifications. An event is an occurrence that is
2076 of interest to some participant; in our case when some real world effect has occurred. Just as action
2077 messages have formatting requirements, so do event notification messages. In this way, the Information
2078 Model of a service must specify the syntax (structure) and semantics (meaning) of the action messages
2079 and event notification messages as part of a service interface. It must also specify the syntax and
2080 semantics of any data that is carried as part of a payload of the action or event notification message. The
2081 Information Model is described in greater detail in the Service Description Model (see Section 4.1).

2082 In addition to the Information Model that describes the syntax and semantics of the messages and data
2083 payloads, exception conditions and error handling in the event of faults (e.g., network outages, improper
2084 message formats, etc.) must be specified or referenced as part of the Service Description.

2085 When a message is used to invoke an action, the correct interpretation typically requires the receiver to
2086 perform an operation, which itself invokes a set of private, internal actions. These **operations** represent
2087 the sequence of (private) actions a service must perform in order to validly participate in a given joint
2088 action.

2089 Similarly, the correct consequence of realizing a real world effect may be to initiate the reporting of that
2090 real world effect via an event notification.

2091 **Message Exchange**

2092 The means by which **joint action** and event notifications are coordinated by service **participants**
2093 (or **delegates**).

2094 **Operations**

2095 The sequence of **actions** a service must perform in order to validly participate in a given **joint**
2096 **action**.

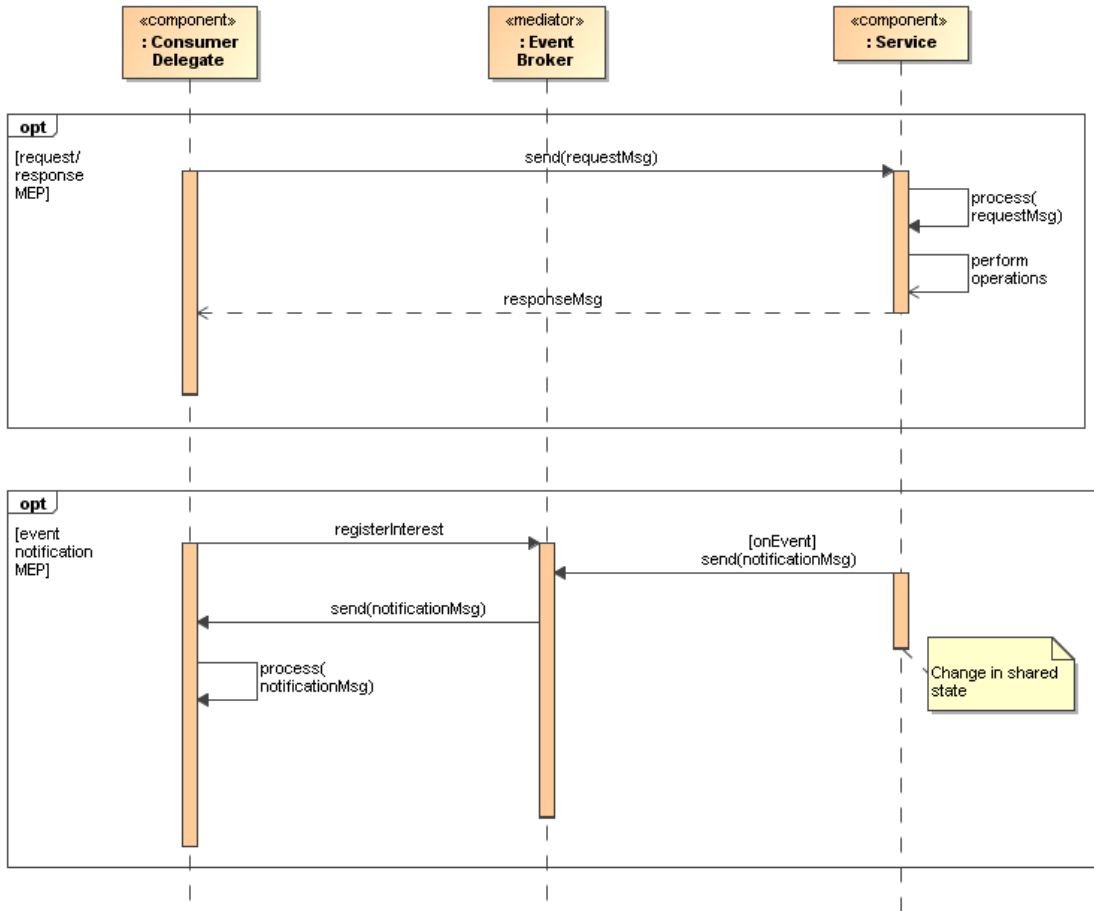
2097 4.3.3.1 Message Exchange Patterns (MEPs)

2098 The basic temporal aspect of service interaction can be characterized by two fundamental message
2099 exchange patterns (MEPs):

- 2100 • Request/response to represent how actions cause a real world effect
- 2101 • Event notification to represent how events report a real world effect

⁸ The notion of “joint” in joint action implies that you have to have a speaker *and* a listener in order to interact.

2102 This is by no means a complete list of all possible MEPs used for inter- or intra-enterprise messaging but
 2103 it does represent those that are most commonly used in exchange of information and reporting changes
 2104 in state both within organizations and across organizational boundaries.



2105
 2106 *Figure 29 - Fundamental SOA message exchange patterns (MEPs)*

2107 Recall from the Reference Model that the Process Model characterizes “the temporal relationships
 2108 between and temporal properties of actions and events associated with interacting with the service.”
 2109 Thus, MEPs are a key element of the Process Model. The meta-level aspects of the Process Model (just
 2110 as with the Action Model) are provided as part of the Service Description Model (see Section 4.1).

2111 In the UML sequence diagram shown in Figure 29 it is assumed that the service participants (consumer
 2112 and provider) have delegated message handling to hardware or software delegates acting on their behalf.
 2113 In the case of the service consumer, this is represented by the *Consumer Delegate* component. In the
 2114 case of the service provider, the delegate is represented by the *Service* component. The message
 2115 interchange model illustrated represents a logical view of the MEPs and not a physical view. In other
 2116 words, specific hosts, network protocols, and underlying messaging system are not shown, as these tend
 2117 to be implementation specific. Although such implementation-specific elements are considered outside
 2118 the scope of this document, they are important considerations in modeling the SOA execution context.
 2119 Recall from the Reference Model that the *execution context* of a service interaction is “the set of
 2120 infrastructure elements, process entities, policy assertions and agreements that are identified as part of
 2121 an instantiated service interaction, and thus forms a path between those with needs and those with
 2122 capabilities.”

2123 4.3.3.2 Request/Response MEP

2124 In a request/response MEP, the Consumer Delegate component sends a request message to the Service
 2125 component. The Service component then processes the request message. Based on the content of the
 2126 message, the Service component performs the service operation and the associated private actions.

2127 Following the completion of these operations, a response message is returned to the Consumer Delegate
2128 component. The response could be that a step in a process is complete, the initiation of a follow-on
2129 operation, or the return of requested information.⁹

2130 Although the sequence diagram shows a *synchronous* interaction (because the sender of the request
2131 message, i.e., Consumer Delegate, is blocked from continued processing until a response is returned
2132 from the Service) other variations of request/response are valid, including *asynchronous* (non-blocking)
2133 interaction through use of queues, channels, or other messaging techniques.

2134 What is important to convey here is that the request/response MEP represents action, which causes a
2135 real world effect, irrespective of the underlying messaging techniques and messaging infrastructure used
2136 to implement the request/response MEP.

2137 **4.3.3.3 Event Notification MEP**

2138 An event is made visible to interested consumers by means of an event notification message exchange
2139 that reports a real world effect; specifically, a change in shared state between service participants. The
2140 basic event notification MEP takes the form of a one-way message sent by a notifier component (in this
2141 case, the Service component) and received by components with an interest in the event (here, the
2142 Consumer Delegate component).

2143 Often the sending component may not be fully aware of all the components that wish to receive the
2144 notification; particularly in so-called publish/subscribe ('pub/sub') situations. In event notification message
2145 exchanges, it is rare to have a tightly-coupled link between the sending and the receiving component(s)
2146 for a number of practical reasons. One of the most common constraints for pub/sub messaging is the
2147 potential for network outages or communication interrupts that can result in loss of notification of events.
2148 Therefore, a third-party mediator component is often used to decouple the sending and receiving
2149 components.

2150 Although this is typically an implementation issue, because this type of decoupling is so common in
2151 event-driven systems, it is warranted for use in modeling this type of message exchange in the SOA-RAF.
2152 This third-party intermediary is shown in Figure 29 as an Event Broker mediator. As with the
2153 request/response MEP, no distinction is made between synchronous versus asynchronous
2154 communication, although synchronous message exchange is illustrated in the UML sequence diagram
2155 depicted in Figure 29.

2156 **4.3.4 Composition of Services**

2157 Composition of services is the act of aggregating or 'composing' a single service from one or more other
2158 services. A simple model of service composition is illustrated in Figure 30.

⁹ There are cases when a response is not always desired and this would be an example of a "one-way" MEP. Similarly, while not shown here, there are cases when some type of "callback" MEP is required in which the consumer agent is actually exposed as a service itself and is able to process incoming messages from another service.

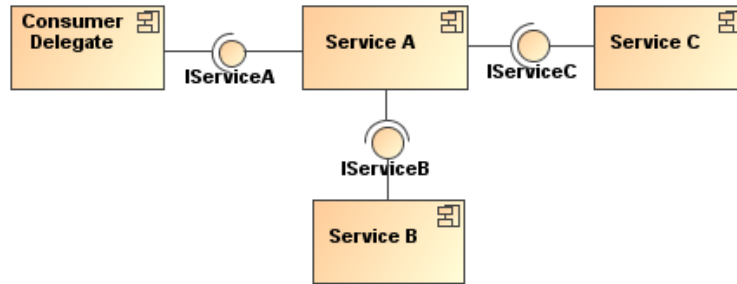


Figure 30 - Simple model of service composition

2159
2160

2161 Here, Service A is a service that has an exposed interface IServiceA, which is available to the Consumer Delegate and relies on two other services in its implementation. The Consumer Delegate does not know
2162 that Services B and C are used by Service A, or whether they are used in serial or parallel, or if their
2163 operations succeed or fail. The Consumer Delegate only cares about the success or failure of Service A.
2164 The exposed interfaces of Services B and C (IService B and IServiceC) are not necessarily hidden from
2165 the Consumer Delegate; only the fact that these services are used as part of the composition of Service
2166 A. In this example, there is no practical reason the Consumer Delegate could not interact with Service B
2167 or Service C in some other interaction scenario.
2168

2169 While the service composition is opaque from the Consumer Delegate's perspective, it is transparent to
2170 the service owner. This transparency is necessary for service management to properly manage the
2171 dependencies between the services used in constructing the composite service—including managing the
2172 service's lifecycle. The subject of services as management entities is described and modeled in the
2173 *Ownership in a SOA Ecosystem View* of the SOA-RAF and is not further elaborated in this section. The
2174 point to be made here is that there can be different levels of opacity or transparency when it comes
2175 to visibility of service composition.

2176 Services can be composed in a variety of ways, including direct consumer-to-service interaction, by using
2177 programming techniques or using an intermediary, such as an orchestration engine leveraging higher
2178 level orchestration languages. Such approaches are further elaborated in the following sub-sections.

2179 4.3.5 Implementing Service Composition

2180 Services are implemented through a combination of processes and collaboration. The concepts involved
2181 and that would be used in the context of exchanges both within and across organizational boundaries are
2182 described and modeled as part of the *Participation in a SOA Ecosystem* view of this reference
2183 architecture (see Section 3).

2184 The principles involved in the composition of services (including but not limited to loose coupling,
2185 selective transparency and opacity, dynamic interactions) are equally applicable to services which
2186 implement business processes and collaborations. Business processes and collaborations represent
2187 complex, multi-step business functions that may involve multiple participants, both within the enterprise
2188 and beyond, including external customers and trading partners. Therefore, such complexities cannot
2189 simply be ignored when transforming traditional business processes and collaborations to their service-
2190 oriented variants.

2191 While business processes are primarily concerned with describing how services are invoked and
2192 executed, business collaborations are more concerned with how actors (usually from different
2193 organizations) interact to realize a desired effect.

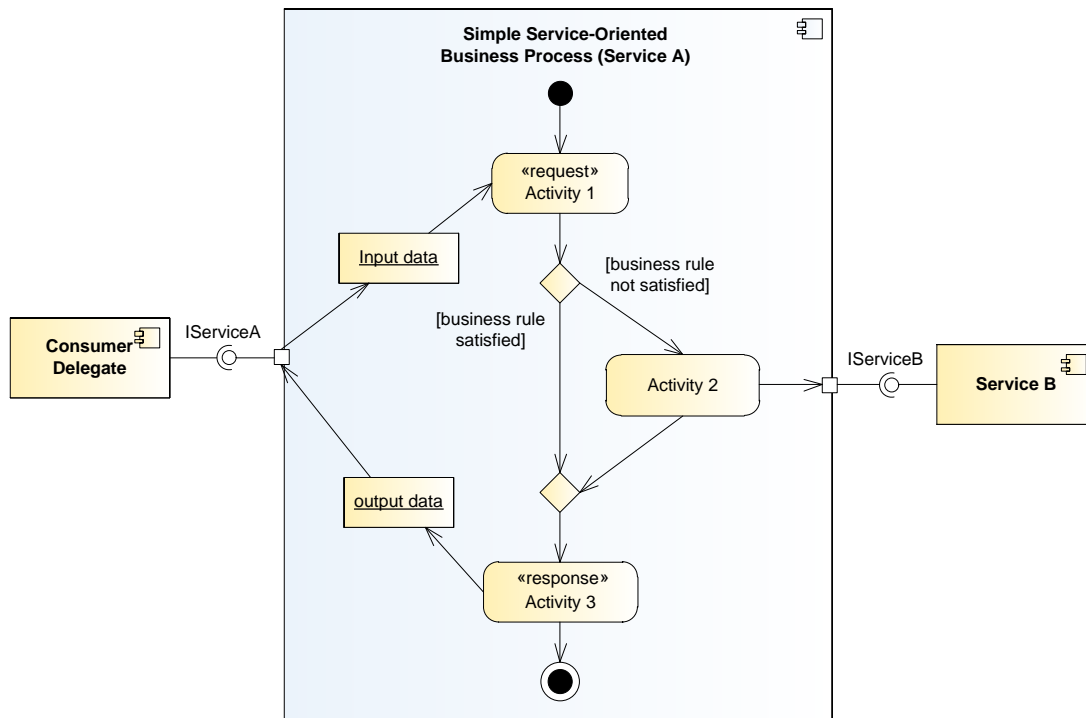
2194 Collaborations can include processes (for example, when one actor executes a particular activity
2195 according to the predefined steps of a process) as much as processes can include collaborations (a
2196 predefined step of a particular process may include agreed-upon activities provided by other participants).

2197 The techniques discussed below can be applied to any combination of services that instantiate service-
2198 oriented business processes or service-oriented business collaborations.

2199 4.3.5.1 Service-Oriented Business Processes

2200 Service orientation as applied to a business process includes

2201 • abstracting the set of activities and rules governing a business process; and
 2202 • composing and exposing the resultant logic as a reusable service.
 2203 When business processes are implemented as SOA services, all of the concepts used to describe and
 2204 model composition of services that were articulated in Section 4.3.4 apply.
 2205 Business processes have temporal properties and can be short-lived or long-lived. Further, these
 2206 processes may involve many participants and may be important considerations for the consumer of a
 2207 service-oriented business process. For example, a consumer may need to know certain details of the
 2208 business process in order to have confidence in the resulting real world effects. For business processes
 2209 implemented as SOA-based services, ensuring that the meta-level aspects of the service-oriented
 2210 business process are included in its Service Description can provide needed insight for the consumer.



2211 *Figure 31 - Abstract example of a simple business process exposed as a service*
 2212

2213 In Figure 31, we use a UML activity diagram to model the simple service-oriented business process. This
 2214 allows us to capture the major elements, such as the set of related activities to be performed (an activity
 2215 being made up of one or more related actions, as explained in Section 3.3.2); the links between these
 2216 activities in a logical flow; data that is passed between activities, and any relevant business rules that
 2217 govern the transitions between the activities. While specific actions and activities can be readily modeled
 2218 in more detail, they are not illustrated in the model in Figure 31.

2219 This example is based on a request/response MEP and captures how one process can leverage
 2220 fulfillment of a particular activity (Activity 2) by calling upon an externally-provided service, Service B. The
 2221 entire service-oriented business process is exposed as Service A that is accessible via its externally
 2222 visible interface, IServiceA. It is the availability of this external interface, and the description of what the
 2223 service intends, that distinguishes this from a simple business process.

2224 Although not explicitly shown in the model above, it is assumed that there exists a software or hardware
 2225 component that executes the process flow (Functionality of Service A). However, human actors may also
 2226 take part. This may be particularly important in cases where the automation fails and human intervention
 2227 becomes necessary.

2228 4.3.5.2 Service-Oriented Business Collaborations

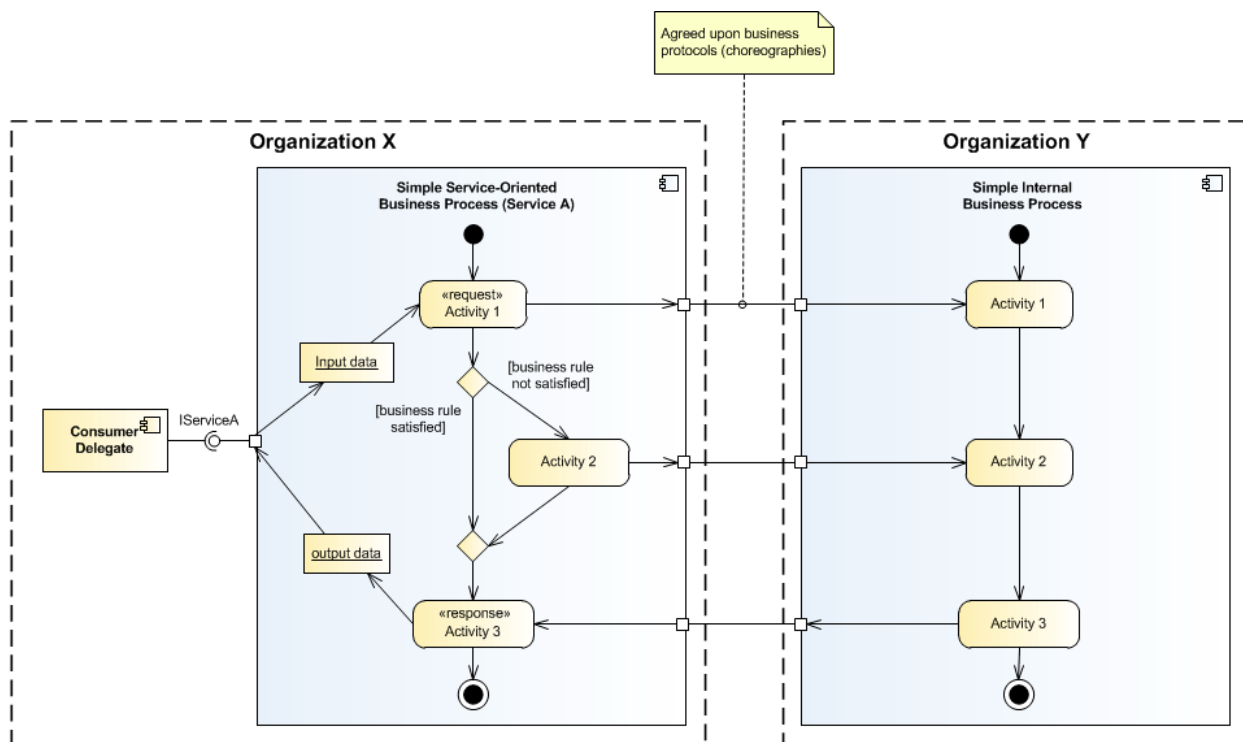
2229 Whereas a service can execute according to a predefined business process determined by one
 2230 organization, service composition can also be accomplished as a cooperation, or business collaboration,
 2231 between actors in different organizations and systems.

2232 In a service-oriented business collaboration, multiple participants interact in a peer-style communication
 2233 as part of some larger business transaction by exchanging messages with trading partners and external
 2234 organizations (e.g., suppliers) [NEWCOMER/LOMOW]. Participants do not necessarily expose the
 2235 entirety of their respective capabilities but rather use service-based interactions to access those
 2236 capabilities needed to fulfill the collaboration.

2237 Service-orientation as applied to a business collaboration includes:

- 2238 - ability of participants to individually provide and commit to what is required during an interaction for a
 2239 collaboration to be successfully realized, including acceptance of preconditions and expected
 2240 outcome;
- 2241 - availability of service functionality sufficient to realize the effects expected from the business
 2242 collaboration;
- 2243 - willingness of participants to engage in interactions that are required as part of the collaboration;
- 2244 - availability of shared state and notifications to all participants who require them, such that they can
 2245 fulfill their respective parts of the collaboration.

2246 Any service contributing to such a service-oriented business collaboration participates “as is”, without
 2247 modification, and consistent with its own service description. Each contributing service is only an
 2248 instrument in the collaboration and is not typically “aware” of its own contribution except as would be
 2249 conveyed through inputs, access to shared states, or event notifications that are generally available to the
 2250 service.



2251
 2252 *Figure 32 - Abstract example of a more complex composition that relies on collaboration*

2253 Figure 32, which is a variant of the example illustrated earlier (in Figure 31), includes trust boundaries
 2254 between two organizations; namely, Organization X and Organization Y. It is assumed that these two
 2255 organizations are peer entities that have an interest in a business collaboration, for example,
 2256 Organization X and Organization Y could be trading partners. Organization X retains the service-oriented
 2257 business process Service A, which is exposed to internal consumers via its provided service interface,
 2258 IServiceA. Organization Y also has a business process that is involved in the business collaboration; in
 2259 this example, it is an internal business process but it could also be exposed to potential consumers either
 2260 within or outside its organizational boundary if it is designed as a reusable service in accordance with
 2261 SOA design principles.

2262 In Figure 32, the communications between Organization X and Organization Y are shown through ports
 2263 where there are “agreed-upon business protocols” that also cover the order in which activities are carried

2264 out. These ports do not explicitly show service interfaces in order to emphasize that in the example these
2265 are not intended to be generally available to any actor in the SOA ecosystem; however, the interfaces
2266 should adhere to the principles involved in the composition of services.

2267 The message exchanges that are used need to specify how and when to initiate activity by the other
2268 trading partner, i.e., communication between Organization X and Organization Y. Defining the business
2269 protocols used in the business collaboration involves precisely specifying the visible message exchange
2270 behavior and order of each of the parties involved in the protocol, without revealing internal
2271 implementation details **[NEWCOMER/LOMOW]**. This is consistent with the Information and Behavior
2272 Models discussed in the Reference Model and as part of service description in section 4.1.

2273 Business processes and collaboration are thus both facets of SOA service composition. The degree to
2274 which one predominates over the other (and the mix of the two that emerges) will be a reflection of many
2275 factors including the relative autonomy of participants and actors, the desired flexibility of a system, the
2276 extent of trust involved and the assessment of risk, among others.

2277 4.3.6 Architectural Implications of Interacting with Services

2278 Interacting with Services has the following architectural implications on mechanisms that facilitate service
2279 interaction:

- 2280 • A well-defined service Information Model **MUST** be provided that:
 - 2281 ○ describes the syntax and semantics of the messages used to denote actions and events;
 - 2282 ○ describes the syntax and semantics of the data payload(s) contained within messages;
 - 2283 ○ documents exception conditions in the event of faults due to network outages, improper
2284 message/data formats, etc.;
 - 2285 ○ is both human readable and machine processable;
 - 2286 ○ is referenceable from the Service Description artifact.
- 2287 • A well-defined service Behavior Model (as defined in the SOA-RM) **MUST** be provided that:
 - 2288 ○ characterizes the knowledge of the actions invoked against the service and events that report real
2289 world effects as a result of those actions;
 - 2290 ○ characterizes the temporal relationships and temporal properties of actions and events associated
2291 in a service interaction;
 - 2292 ○ describe activities involved in a workflow activity that represents a unit of work;
 - 2293 ○ describes the role (s) performed in a service-oriented business process or service-oriented
2294 business collaboration;
 - 2295 ○ is both human readable and machine processable;
 - 2296 ○ is referenceable from the Service Description artifact.
- 2297 • Mechanisms **MUST** be included to support composition of service-oriented business processes and
2298 service-oriented business collaborations such as:
 - 2299 ○ Declarative and programmatic compositional languages;
 - 2300 ○ Orchestration and/or choreography engines that support multi-step processes as part of a short-
2301 lived or long-lived business transaction;
 - 2302 ○ Orchestration and/or choreography engines that support compensating transactions in the
2303 presences of exception and fault conditions.
- 2304 • Infrastructure **MUST** be specified that provides mechanisms to support service interaction, including
2305 but not limited to:
 - 2306 ○ mediation within service interactions based on shared semantics;
 - 2307 ○ translation and transformation of multiple application-level protocols to standard network transport
2308 protocols;
 - 2309 ○ auditing and logging that provide a data store and mechanism to record information related to
2310 service interaction activity such as message traffic patterns, security violations, and service
2311 contract and policy violations
 - 2312 ○ security that provides authorization and authentication support, etc., which provide protection
2313 against common security threats in a SOA ecosystem;
 - 2314 ○ monitoring such as hardware and software mechanisms that both monitor the performance of
2315 systems that host services and network traffic during service interaction, and are capable of
2316 generating regular monitoring reports.

- 2317 • In a service-oriented business collaboration, any language used **MUST** be capable of describing the
2318 coordination required of those service-oriented business processes that cross organizational
2319 boundaries. This **SHOULD** provide for contingencies, in case of an upset or when automation fails,
2320 including any necessary human intervention.

2321 4.4 Policies and Contracts Model

2322 A common phenomenon of many machines and systems is that the scope of potential behavior is much
2323 broader than is actually needed for a particular circumstance. This is especially true of a system as
2324 powerful as a SOA ecosystem. As a result, the behavior and performance of the system tend to be under-
2325 constrained by the implementation; instead, the actual behavior is expressed by means of policies of
2326 some form. Policies define the choices that stakeholders make; these choices are used to guide the
2327 actual behavior of the system to the desired behavior and performance.

2328 As noted in Section 3.2.5.2, a policy is an expression of constraints that is promulgated by a stakeholder
2329 who has the responsibility of ensuring that the constraint is enforceable. In contrast, contracts are
2330 **agreements** between participants.

2331 While responsibility for enforcement may differ, both contracts and policies share a common characteristic
2332 – there is a constraint that must be enforced. In both cases, the mechanisms needed to enforce
2333 constraints are likely to be identical; in this model, we focus on the issues involved in representing
2334 policies and contracts and on some of the principles behind their enforcement.

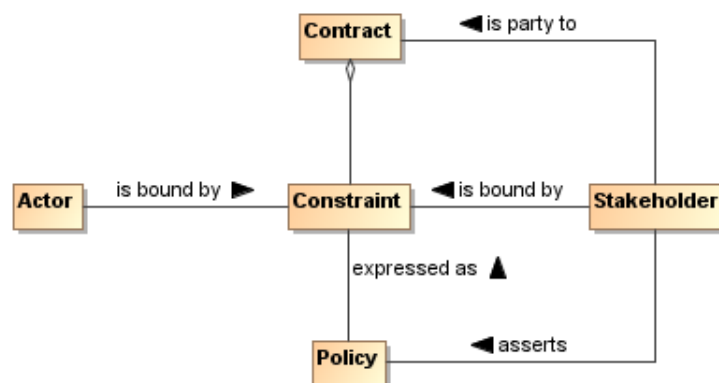
2335 4.4.1 Policy and Contract Representation

2336 A policy constraint is a specific kind of constraint: the ontology of policies and contracts includes the core
2337 concepts of permission, obligation, owner, and subject. In addition, it may be necessary to be able to
2338 combine policy constraints and to be able to resolve policy conflicts.

2339 Policy Framework

2340 A policy framework is a language in which **policy constraints** may be expressed.

2341 A policy framework combines syntax for expressing policy constraints together with a decision procedure
2342 for determining if a policy constraint is satisfied.



2343
2344

Figure 33 - Policies and Contracts

2345 We can characterize a policy framework in terms of a **logical framework** and an ontology of policies. The
2346 **policy ontology** details specific kinds of policy constraints that can be expressed; and the logical
2347 framework is a 'glue' that allows us to express combinations of policies.

2348 Logical Framework

2349 A linguistic framework consisting of a syntax – a way of writing expressions – and a semantics –
2350 a way of interpreting the expressions.

2351 Policy Ontology

2352 A formalization of a set of concepts that are relevant to forming policy expressions.

2353 For example, a policy ontology that allows identification of simple constraints – such as the existence of a
2354 property, or that a value of a property should be compared to a fixed value – is often enough to express
2355 many basic constraints.

2356 Included in many policy ontologies are the basic signals of permissions and obligations. Some policy
2357 frameworks are sufficiently constrained that there is no possibility of representing an obligation; in which
2358 case there is often no need to ‘call out’ the distinction between permissions and obligations.

2359 The logical framework is also a strong determiner of the expressivity of the policy framework: the richer
2360 the logical framework, the richer the set of policy constraints that can be expressed. However, there is a
2361 strong inverse correlation such that increasing expressivity yields less ease and greater inefficiency of
2362 implementation.

2363 In the discussion that follows we assume the following basic policy ontology:

2364 **Policy Owner**

2365 A **stakeholder** that asserts and enforces the **policy**.

2366 **Policy Subject**

2367 An **actor** whose action, or a **resource** whose maintenance or use, is constrained by a **policy**.

2368 **Policy Constraint**

2369 A measurable and enforceable assertion found within a **policy**.

2370 **Policy Object**

2371 An identifiable **state**, **action** or **resource** that is potentially constrained by the **policy**.

2372 4.4.2 Policy and Contract Enforcement

2373 The enforcement of policy constraints has to address two core problems: how to enforce the atomic policy
2374 constraints, and how to enforce combinations of policy constraints. In addition, it is necessary to address
2375 the resolution of policy conflicts. Contracts are the documented agreement between two or more parties
2376 but otherwise have the same enforcement requirements as policies.

2377 The concepts relating to policy decisions and enforcement are discussed here in general terms; how
2378 these concepts are applied specifically can be seen, for example, in the Governance Model (5.1) and the
2379 Security Model (5.2).

2380 4.4.2.1 Enforcing Simple Policy Constraints

2381 The two primary kinds of policy constraint – permission and obligation – naturally lead to different styles
2382 of enforcement. A permission constraint must typically be enforced prior to the policy subject invoking the
2383 policy object. On the other hand, an obligation constraint must typically be enforced after the fact through
2384 some form of auditing process and remedial action.

2385 For example, if a communications policy required that all communication be encrypted, this is enforceable
2386 at the point of communication: any attempt to communicate a message that is not encrypted can be
2387 blocked.

2388 Similarly, an obligation to pay for services rendered is enforced by ensuring that payment arrives within a
2389 reasonable period of time. Invoices are monitored for prompt (or lack of) payment.

2390 The key concepts in enforcing both forms of policy constraint are the policy decision and the policy
2391 enforcement.

2392 **Policy Decision**

2393 A determination as to whether a given **policy constraint** is satisfied.

2394 A policy decision is effectively a measurement of some state – typically a portion of the SOA ecosystem’s
2395 **shared state**. This implies a certain *timeliness* in the measuring: a measurement that is too early or is too
2396 late does not actually help in determining if the policy constraint is satisfied appropriately.

2397 **Policy Enforcement**

2398 A mechanism that limits the behavior and/or **state of policy subjects** to comply with a **policy**
2399 **decision**.

2400 A policy enforcement implies the use of some mechanism to ensure compliance with a policy decision.
2401 The range of mechanisms is completely dependent on the kinds of atomic policy constraints that the
2402 policy framework may support.

2403 4.4.2.2 Conflict Resolution

2404 Whenever it is possible that more than one policy constraint applies in a given situation, there is the
2405 potential that the policy constraints themselves are not mutually consistent. For example, a policy
2406 constraint that requires communication to be encrypted and a policy constraint that requires an
2407 administrator to read every communication conflict with each other – the two policy constraints cannot
2408 both be satisfied concurrently.

2409 In general, with sufficiently rich policy frameworks, it is not possible to always resolve policy conflicts
2410 automatically. However, a reasonable approach is to augment the policy decision process with simple
2411 policy conflict resolution rules; with the potential for *escalating* a policy conflict to human adjudication.

2412 Policy Conflict

2413 A state in a **policy decision** process in which the satisfaction of one or more **policy constraints**
2414 leads directly to the violation of one or more other policy constraints.

2415 Policy Conflict Resolution

2416 A **rule** determining which **policy constraint(s)** should prevail if a **policy conflict** occurs.

2417 The inevitable consequence of policy conflicts is that it is not possible to guarantee that all policy
2418 constraints are satisfied at all times. This, in turn, implies certain *flexibility* in the application of policy
2419 constraints: each individual constraint may not always be honored.

2420 4.4.3 Architectural Implications

2421 The key choices that must be made in a system of policies center on the policy framework, policy
2422 enforcement, and conflict resolution:

- 2423 • There **SHOULD** be a standard policy framework that is adopted across ownership domains within the
2424 SOA ecosystem:
 - 2425 ○ This framework **MUST** permit the expression of simple policy constraints
 - 2426 ○ The framework **MAY** allow (to a varying extent) the combination of policy constraints, including:
 - 2427 • Both positive and negative constraints
 - 2428 • Conjunctions and disjunctions of constraints
 - 2429 • The quantification of constraints
 - 2430 ○ The framework **MUST** at least allow the policy subject and the policy object to be identified as well
2431 as the policy constraint.
 - 2432 ○ The framework **MAY** allow further structuring of policies into modules, inheritance between policies
2433 and so on.
- 2434 • There **SHOULD** be mechanisms that facilitate the application of policies:
 - 2435 ○ There **SHOULD** be mechanisms that allow policy decisions to be made, consistent with the policy
2436 frameworks.
 - 2437 ○ There **SHOULD** be mechanisms to enforce policy decisions:
 - 2438 • There **SHOULD** be mechanisms to support the measurement of whether certain policy
2439 constraints are satisfied, or to what degree they are satisfied.
 - 2440 • Such enforcement mechanisms **MAY** include support for both permission-style constraints and
2441 obligation-style constraints.
 - 2442 • Enforcement mechanisms **MAY** support the simultaneous enforcement of multiple policy
2443 constraints across multiple points in the SOA ecosystem.
 - 2444 ○ There **SHOULD** be mechanisms to resolve policy conflicts:
 - 2445 • This **MAY** involve escalating policy conflicts to human adjudication.
 - 2446 ○ There **SHOULD** be mechanisms that support the management and promulgation of policies.

5 Ownership in a SOA Ecosystem View

*Governments are instituted among Men,
deriving their just power from the consent of the governed*
American Declaration of Independence

The *Ownership in a SOA Ecosystem View* focuses on the issues, requirements and responsibilities involved in owning a SOA-based system.

Ownership of a SOA-based system in a SOA ecosystem raises significantly different challenges to owning other complex systems – such as Enterprise suites – because there are strong limits on the control and authority of any one party when a system spans multiple ownership domains.

Even when a SOA-based system is deployed internally within an organization, there are multiple internal stakeholders involved and there may not be a simple hierarchy of control and management. Thus, an early consideration of how multiple boundaries affect SOA-based systems provides a firm foundation for dealing with them in whatever form they are found rather than debating whether the boundaries should exist.

This view focuses on the governance and management of SOA-based systems, on the security challenges involved in running a SOA-based system, and testing challenges.

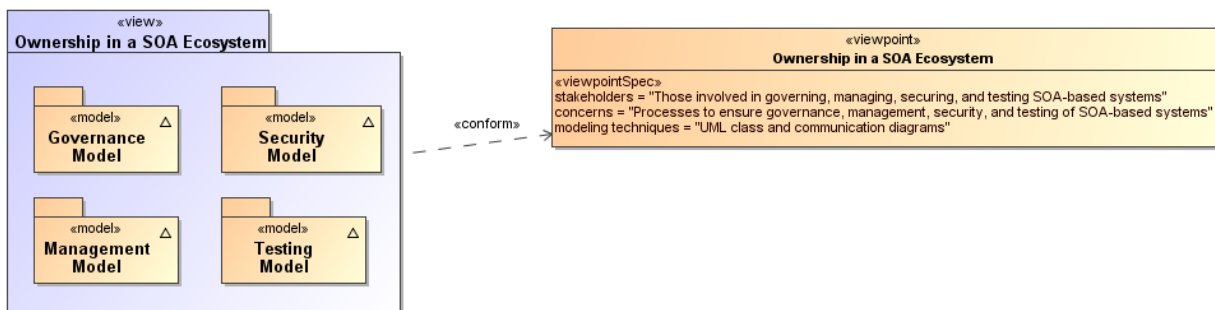


Figure 34 - Model Elements Described in the Ownership in a SOA Ecosystem View

5.1 Governance Model

The Reference Model defines Service Oriented Architecture as an architectural paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains [SOA-RM]. Consequently, it is important that organizations that plan to engage in service interactions adopt governance policies and procedures sufficient to ensure that there is standardization across both internal and external organizational boundaries to promote the effective creation and use of SOA-based services.

5.1.1 Understanding Governance

5.1.1.1 Terminology

Governance is about making decisions that are aligned with the overall organizational strategy and culture of the enterprise. [HOTLE] It specifies the decision rights and accountability framework to encourage desirable behaviors towards realizing the strategy [WEILL] and defines incentives (positive or negative) towards that end. It is less about overt control and strict adherence to rules, and more about guidance and effective and equitable usage of resources to ensure sustainability of an organization's strategic objectives. [TOGAF v9]

To accomplish this, governance requires organizational structure and processes and must identify who has authority to define and carry out its mandates. It must address the following questions:

- 2483 1. what decisions must be made to ensure effective management and use?
2484 2. who should make these decisions?
2485 3. how will these decisions be made and monitored? and
2486 4. how will these decisions be communicated?

2487 The intent is to achieve goals, add value, and reduce risk.

2488 Within a single ownership domain such as an enterprise, generally there is a hierarchy of governance
2489 structures. Some of the more common enterprise governance structures include corporate governance,
2490 technology governance, IT governance, and architecture governance [TOGAF v9]. These governance
2491 structures can exist at multiple levels (global, regional, and local) within the overall enterprise.

2492 It is often asserted that SOA governance is a specialization of IT governance as there is a natural
2493 hierarchy of these types of governance structures; however, the focus of SOA governance is less on
2494 decisions to ensure effective management and use of IT as it is to ensure effective management and use
2495 of SOA-based systems. Certainly, SOA governance must still answer the basic questions also associated
2496 with IT governance, i.e., who should make the decisions, and how these decisions will be made and
2497 monitored.

2498 5.1.1.2 Relationship to Management

2499 There is often confusion centered on the relationship between governance and management. As
2500 described earlier, governance is concerned with decision making. Management, on the other hand, is
2501 concerned with execution. Put another way, governance describes the world as **leadership** wants it to
2502 be; management executes activities that intend to make the leadership's desired world a reality. Where
2503 governance determines who has the authority and responsibility for making decisions and the
2504 establishment of guidelines for how those decisions should be made, management is the actual process
2505 of making, implementing, and measuring the impact of those decisions. Consequently, governance and
2506 management work in concert to ensure a well-balanced and functioning organization as well as an
2507 ecosystem of inter-related organizations. In the sections that follow, we elaborate further on the
2508 relationship between governance and management in terms of setting and enforcing service policies,
2509 contracts, and standards as well as addressing issues surrounding regulatory compliance.

2510 5.1.1.3 Why is SOA Governance Important?

2511 One of the hallmarks of SOA that distinguishes it from other architectural paradigms for distributed
2512 computing is the ability to provide a uniform means to offer, discover, interact with and use capabilities
2513 (as well the ability to compose new capabilities from existing ones) all in an environment that transcends
2514 domains of ownership. Consequently, ownership and issues surrounding it, such as obtaining acceptable
2515 terms and conditions (T&Cs) in a contract, are primary topics for SOA governance. Generally, IT
2516 governance does not include T&Cs, for example, as a condition of use as its primary concern.

2517 Just as other architectural paradigms, technologies, and approaches to IT are subject to change and
2518 evolution, so too is SOA. Setting policies that allow change management and evolution, establishing
2519 strategies for change, resolving disputes that arise, and ensuring that SOA-based systems continue to
2520 fulfill the goals of the business are all reasons why governance is important to SOA.

2521 5.1.1.4 Governance Stakeholders and Concerns

2522 As noted in Section 3.2.1 the participants in a service interaction include the service provider, the service
2523 consumer, and other interested or unintentional third parties. Depending on the circumstances, it may
2524 also include the owners of the underlying capabilities that the SOA services access. Governance must
2525 establish the policies and rules under which duties and responsibilities are defined and the expectations
2526 of participants are grounded. The expectations include transparency in aspects where transparency is
2527 mandated; trust in the impartial and consistent application of governance; and assurance of reliable and
2528 robust behavior throughout the SOA ecosystem.

2529 **5.1.2 A Generic Model for Governance**

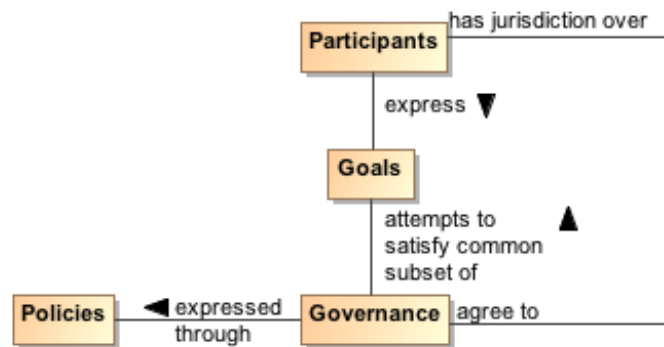
2530 **Governance**

2531 The prescription of conditions and constraints consistent with satisfying common goals and the
2532 structures and processes needed to define and respond to actions taken towards realizing those
2533 goals.

2534 The following is a generic model of governance represented by segmented models that begin with
2535 motivation and proceed through measuring compliance. It is not all-encompassing but a focused subset
2536 that captures the aspects necessary to describe governance for SOA. It does not imply that practical
2537 application of governance is a single, isolated instance of these models; in reality, there may be
2538 hierarchical and parallel chains of governance that deal with different aspects or focus on different goals.
2539 This is discussed further in section 5.1.2.5. The defined models are simultaneously applicable to each of
2540 the overlapping instances.

2541 A given enterprise may already have portions of these models in place. To a large extent, the models
2542 shown here are not specific to SOA; discussions on direct applicability begin in section 5.1.3.

2543 **5.1.2.1 Motivating Governance**



2544 *Figure 35 - Motivating Governance*

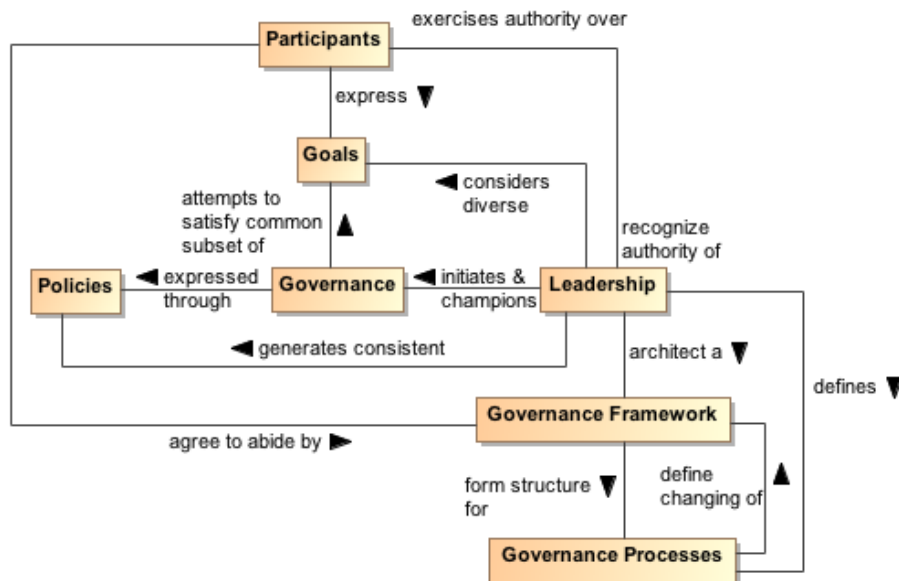
2547 An organizational domain such as an enterprise is made up of participants who may be individuals or
2548 groups of individuals forming smaller organizational units within the enterprise. The overall business
2549 strategy should be consistent with the goals of the participants; otherwise, the business strategy would
2550 not provide value to the participants and governance towards those ends becomes difficult if not
2551 impossible. This is not to say that an instance of governance simultaneously satisfies all the goals of all
2552 the participants; rather, the goals of any governance instance must sufficiently satisfy a useful subset of
2553 each participant's goals so as to provide value and ensure the cooperation of all the participants.

2553 A policy is the formal characterization of the conditions and constraints that governance deems as
2554 necessary to realize the goals which it is attempting to satisfy. Policy may identify required conditions or
2555 actions or may prescribe limitations or other constraints on permitted conditions or actions. For example,
2556 a policy may prescribe that safeguards must be in place to prevent unauthorized access to sensitive
2557 material. It may also prohibit use of computers for activities unrelated to the specified work assignment.
2558 Policy is made operational through the promulgation and implementation of Rules and Regulations (as
2559 defined in section 5.1.2.3).

2560 As noted in section 4.4.2, policy may be asserted by any participant or on behalf of the participant by its
2561 organization. Part of the purpose of governance is to arbitrate among diverse goals of participants and
2562 the diverse policies articulated to realize those goals. The intent is to form a consistent whole that allows
2563 governance to minimize ambiguity about its purpose. While resolving all ambiguity would be an ideal, it is
2564 unlikely that all inconsistencies will be identified and resolved before governance becomes operational.

2565 For governance to have effective jurisdiction over participants, there must be some degree of agreement
2566 by all participants that they will abide by the governance mandates. A minimal degree of agreement often
2567 presages participants who 'slow-roll' if not actively rejecting compliance with policies that express the
2568 specifics of governance.

2569 **5.1.2.2 Setting Up Governance**



2570
2571 *Figure 36 - Setting Up Governance*

2572 **Leadership**

2573 The entity having the **responsibility** and **authority** to generate consistent **policies** through which
2574 the goals of **governance** can be expressed and to define and champion the structures and
2575 processes through which governance is realized.

2576 **Governance Framework**

2577 The set of organizational structures that enable **governance** to be consistently defined, clarified,
2578 and as needed, modified to respond to changes in its domain of concern.

2579 **Governance Process**

2580 The defined set of activities performed within the **Governance Framework** to enable the
2581 consistent definition, application, and as needed, modification of **rules** that organize and regulate
2582 the activities of **participants** for the fulfillment of expressed **policies**.

2583 See section 5.1.2.3 for elaboration on the relationship of Governance Processes and Rules.

2584 As noted earlier, governance requires an appropriate organizational structure and identification of who
2585 has authority to make governance decisions. In Figure 36, the entity with governance authority is
2586 designated the Leadership. This is someone, possibly one or more of the participants, which participants
2587 recognize as having authority for a given purpose or over a given set of issues or concerns.

2588 The leadership is responsible for prescribing or delegating a working group to prescribe the governance
2589 framework that forms the structure for governance processes that define how governance is to be carried
2590 out. This does not itself define the specifics of how governance is to be applied, but it does provide an
2591 unambiguous set of procedures that should ensure consistent actions which participants agree are fair
2592 and account for sufficient input on the subjects to which governance is applied.

2593 The participants may be part of the working group that codifies the governance framework and
2594 processes. When complete, the participants must acknowledge and agree to abide by the products
2595 generated through application of this structure.

2596 The governance framework and processes are often documented in the constitution or charter of a body
2597 created or designated to oversee governance. This is discussed further in the next section. Note that the
2598 governance processes should also include those necessary to modify the governance framework itself.

2599 An important function of leadership is not only to initiate but also be the consistent champion of
2600 governance. Those responsible for carrying out governance mandates must have leadership who make it

2601 clear to participants that expressed policies are seen as a means to realizing established goals and that
2602 compliance with governance is required.

2603 5.1.2.3 Carrying Out Governance

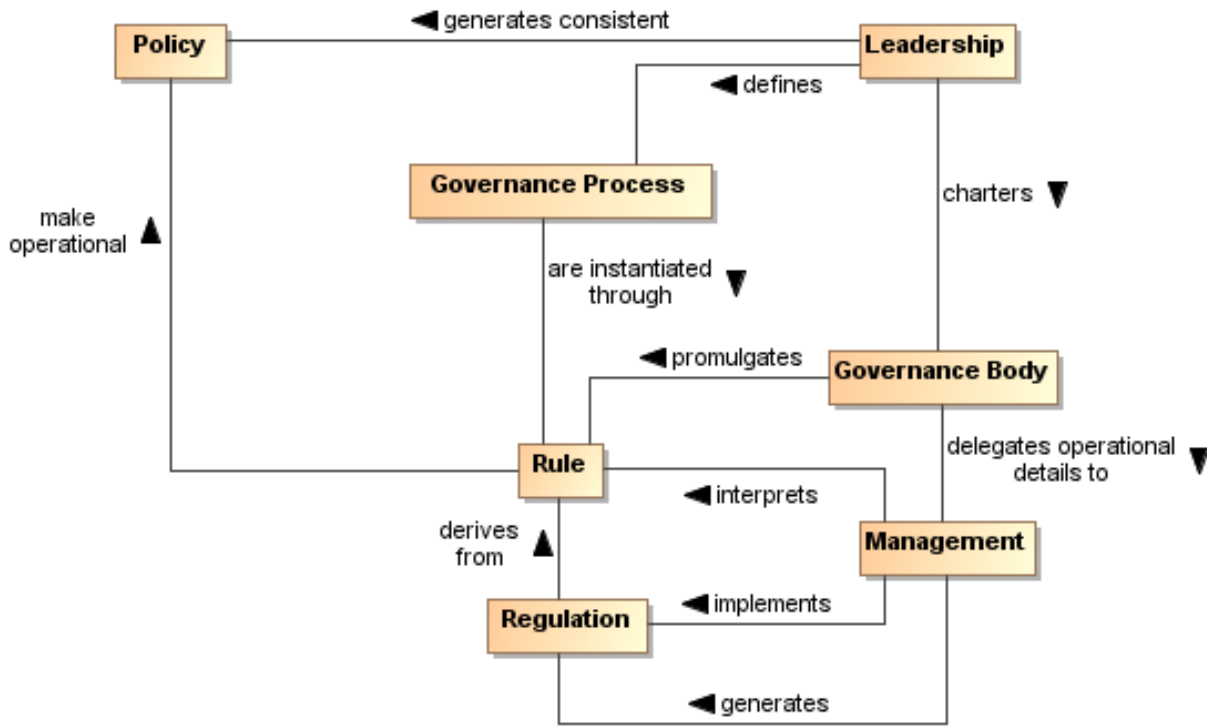


Figure 37 - Carrying Out Governance

2604
2605
2606 **Rule**

A prescribed guide for carrying out activities and processes leading to desired results, e.g. the operational realization of **policies**.

2609 **Regulation**

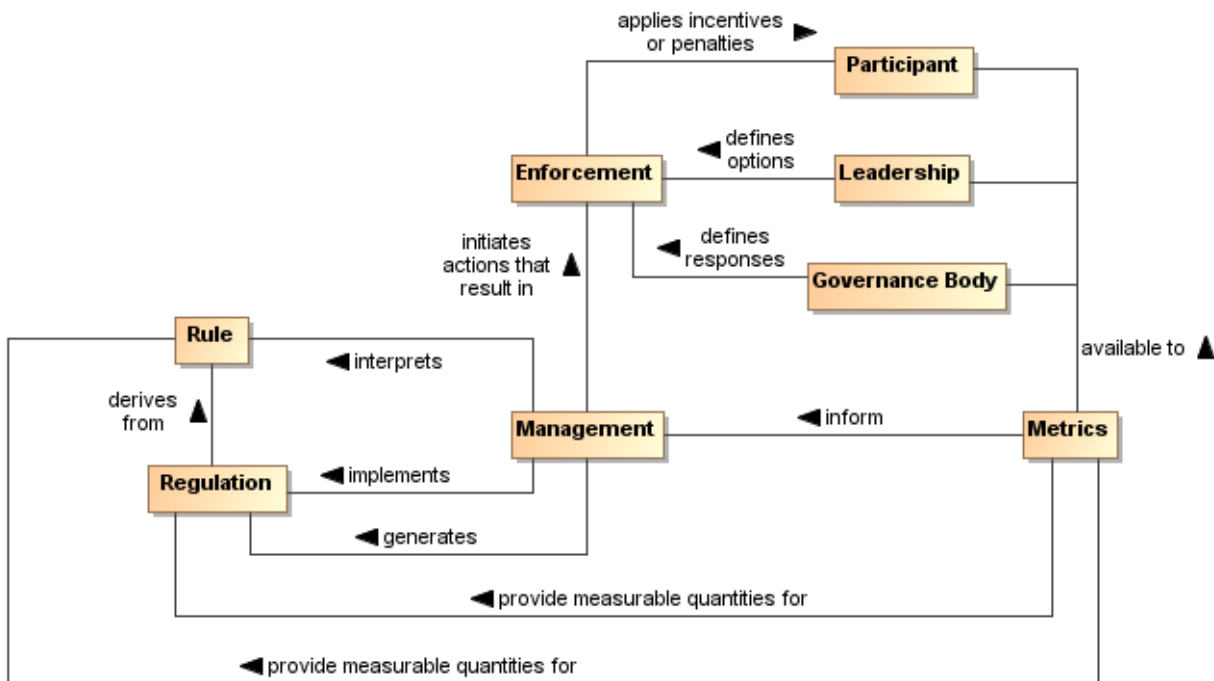
A mandated process or the specific details that derive from the interpretation of **rules** and lead to measureable quantities against which compliance can be measured.

2612 To carry out governance, leadership charts a governance body to promulgate the rules needed to make
2613 the policies operational. The governance body acts in line with governance processes for its rule-making
2614 process and other functions. Whereas governance is the setting of policies and defining the rules that
2615 provide an operational context for policies, governance body may delegate the operational details of
2616 governance to management. Management generates regulations that specify details for rules and other
2617 procedures to implement both rules and regulations. For example, leadership could set a policy that all
2618 authorized parties should have access to data, the governance body would promulgate a rule that PKI
2619 certificates are required to establish identity of authorized parties, and management can specify a
2620 regulation of who it deems to be a recognized PKI issuing body. In summary, policy is a predicate to be
2621 satisfied and rules prescribe the activities by which that satisfying occurs. A number of rules may be
2622 required to satisfy a given policy; the carrying out of a rule may contribute to several policies being
2623 realized.

2624 Whereas the governance framework and processes are fundamental for having participants acknowledge
2625 and commit to compliance with governance, the rules and regulations provide operational constraints that
2626 may require resource commitments or other levies on the participants. It is important for participants to
2627 consider the framework and processes to be fair, unambiguous, and capable of being carried out in a
2628 consistent manner and to have an opportunity to formally accept or ratify this situation. Rules and
2629 regulations, however, do not require individual acceptance by any given participant although some level

2630 of community comment may be part of the governance processes. Having agreed to governance, the
 2631 participants are bound to comply or be subject to prescribed mechanisms for enforcement.

2632 **5.1.2.4 Ensuring Governance Compliance**



2633
 2634 *Figure 38 - Ensuring Governance Compliance*

2635 Setting rules and regulations does not ensure effective governance unless compliance can be measured
 2636 and rules and regulations can be enforced. Metrics are those conditions and quantities that can be
 2637 measured to characterize actions and results. Rules and regulations must be based on collected metrics
 2638 or there is no means for management to assess compliance. The metrics are available to the participants,
 2639 the leadership, and the governance body so what is measured and the results of measurement are clear
 2640 to everyone.

2641 The leadership in its relationship with participants has certain options that can be used for enforcement. A
 2642 common option may be to affect future funding. The governance body defines specific enforcement
 2643 responses, such as what degree of compliance is necessary for full funding to be restored. It is up to
 2644 management to identify compliance shortfalls and to initiate the enforcement process.

2645 Note, enforcement does not strictly need to be negative consequences. Management can use metrics to
 2646 identify exemplars of compliance and leadership can provide options for rewarding the participants. The
 2647 governance body defines awards or other incentives.

2648 **5.1.2.5 Considerations for Multiple Governance Chains**

2649 As noted in section 5.1.2, instances of the governance model often occur as a tiered arrangement, with
 2650 governance at some level delegating specific authority and responsibility to accomplish a focused portion
 2651 of the original level's mandate. For example, a corporation may encompass several lines of business and
 2652 each line of business governs its own affairs in a manner that is consistent with and contributes to the
 2653 goals of the parent organization. Within the line of business, an IT group may be given the mandate to
 2654 provide and maintain IT resources, giving rise to IT governance.

2655 In addition to tiered governance, there may be multiple governance chains working in parallel. For
 2656 example, a company making widgets has policies intended to ensure they make high quality widgets and
 2657 make an impressive profit for their shareholders. On the other hand, Sarbanes-Oxley is a parallel
 2658 governance chain in the United States that specifies how the management must handle its accounting

2659 and information that must be given to its shareholders. The parallel chains may just be additive or may be
2660 in conflict and require some harmonization.

2661 Being distributed and representing different ownership domains, a SOA participant falls under the
2662 jurisdiction of multiple governance domains simultaneously and may individually need to resolve
2663 consequent conflicts. The governance domains may specify precedence for governance conformance or
2664 it may fall to the discretion of the participant to decide on the course of actions they believe appropriate.

2665 **5.1.3 Governance Applied to SOA**

2666 **5.1.3.1 Where SOA Governance is Different**

2667 SOA governance is often discussed in terms of IT governance, but rather than a parent-child relationship,
2668 Figure 39 shows the two as siblings within the general governance described in section 5.1.2. There are
2669 obvious dependencies and a need for coordination between the two, but the idea of aligning IT with
2670 business already demonstrates that resource providers and resource consumers must be working
2671 towards common goals if they are to be productive and efficient. While SOA governance is shown to be
2672 active in the area of infrastructure, it is a specialized concern for having a dependable platform to support
2673 service interaction; a range of traditional IT issues is therefore out of scope of this document. A SOA
2674 governance plan for an enterprise will not of itself resolve shortcomings with the enterprise's IT
2675 governance.

2676 Governance in the context of SOA is that organization of services: that promotes their visibility; that
2677 facilitates interaction among service participants; and that directs that the results of service interactions
2678 are those real world effects as described within the service description and constrained by policies and
2679 contracts as assembled in the execution context.

2680 SOA governance must specifically account for control across different ownership domains, i.e. all the
2681 participants may not be under the jurisdiction of a single governance authority. However, for governance
2682 to be effective, the participants must agree to recognize the authority of the governance body and must
2683 operate within the Governance Framework and through the Governance Processes so defined.

2684 SOA governance must account for interactions across ownership boundaries, which may also imply
2685 across enterprise governance boundaries. For such situations, governance emphasizes the need for
2686 agreement that some governance framework and governance processes have jurisdiction, and the
2687 governance defined must satisfy the goals of the participants for cooperation to continue. A standards
2688 development organization such as OASIS is an example of voluntary agreement to governance over a
2689 limited domain to satisfy common goals.

2690 The specifics discussed in the figures in the previous sections are equally applicable to governance
2691 across ownership boundaries as it is within a single boundary. There is a charter agreed to when
2692 participants become members of the organization, and this charter sets up the structures and processes
2693 to be followed. Leadership may be shared by the leadership of the overall organization and the leadership
2694 of individual groups themselves chartered per the governance processes. There are rules and regulations
2695 specific to individual efforts for which participants agree to local goals, and enforcement can be loss of
2696 voting rights or under extreme circumstances, expulsion from the group.

2697 Thus, the major difference for SOA governance is an appreciation for the cooperative nature of the
2698 enterprise and its reliance on furthering common goals if productive participation is to continue.

2699 **5.1.3.2 What Must be Governed**

2700 An expected benefit of employing SOA principles is the ability to quickly bring resources to bear to deal
2701 with unexpected and evolving situations. This requires a great deal of confidence in the underlying
2702 capabilities that can be accessed and in the services that enable the access. It also requires considerable
2703 flexibility in the ways these resources can be employed. Thus, SOA governance requires establishing
2704 confidence and trust (see Section 3.2.5.1) while instituting a solid framework that enables flexibility,
2705 indicating a combination of strict control over a limited set of foundational aspects but minimum
2706 constraints beyond those bounds.

2707

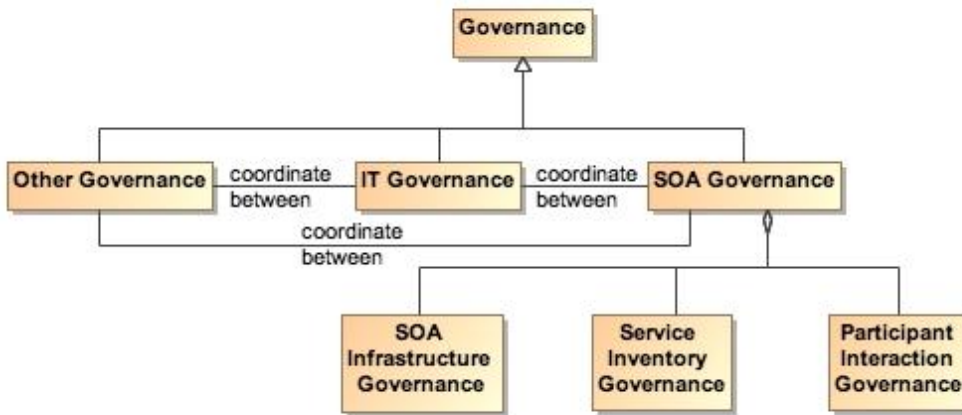


Figure 39 - Relationship Among Types of Governance

2708
2709
2710
2711
2712
2713
2714
2715
2716

SOA governance applies to three aspects of service definition and use:

- SOA infrastructure – the ‘plumbing’ that provides utility functions that enable and support the use of the service
- Service inventory – the requirements on a service to permit it to be accessed within the infrastructure
- Participant interaction – the consistent expectations with which all participants are expected to comply

2717 5.1.3.2.1 Governance of SOA Infrastructure

2718 The SOA infrastructure is likely composed of several families of SOA services that provide access to
2719 fundamental computing business services. These include, among many others, services such as
2720 messaging, security, storage, discovery, and mediation. The provisioning of an infrastructure on which
2721 these services may be accessed and the general realm of those contributing as utility functions of the
2722 infrastructure are a traditional IT governance concern. In contrast, the focus of SOA governance is how
2723 the existence and use of the services enables the SOA ecosystem.

2724 By characterizing the environment as containing families of SOA services, the assumption is that there
2725 may be multiple approaches to providing the business services or variations in the actual business
2726 services provided. For example, discovery could be based on text search, on metadata search, on
2727 approximate matches when exact matches are not available, and numerous other variations. The
2728 underlying implementation of search algorithms are not the purview of SOA governance, but the access
2729 to the resulting service infrastructure enabling discovery must be stable, reliable, and extremely robust to
2730 all operating conditions. Such access enables other specialized SOA services to use the infrastructure in
2731 dependable and predictable ways, and is where governance is important.

2732 5.1.3.2.2 Governance of the Service Inventory

2733 Given an infrastructure in which other SOA services can operate, a key governance issue is which SOA
2734 services to allow in the ecosystem. The major concern should be a definition of well-behaved services,
2735 where the required behavior will inherit their characteristics from experiences with distributed computing
2736 but also evolve with SOA experience. A major need for ensuring well-behaved services is collecting
2737 sufficient metrics to know how the service affects the SOA infrastructure and whether it complies with
2738 established infrastructure policies.

2739 Another common concern of service approval is whether there is a possibility of duplication of function by
2740 multiple services. Some governance models talk to a tightly controlled environment where a primary
2741 concern is to avoid any service duplication. Other governance models talk to a market of services where
2742 the consumers have wide choices. For the latter, it is anticipated that the better services will emerge from
2743 market consensus and the availability of alternatives will drive innovation.

2744 Some combination of control and openness will emerge, possibly with a different appropriate balance for
2745 different categories of use. For SOA governance, the issue is less which services are approved but rather
2746 ensuring that sufficient description is available to support informed decisions for appropriate use. Thus,
2747 SOA governance should concentrate on identifying the required attributes to adequately describe a
2748 service, the required target values of the attributes, and the standards for defining the meaning of the
2749 attributes and their target values. Governance may also specify the processes by which the attribute
2750 values are measured and the corresponding certification that some realized attribute set may imply.

2751 For example, unlimited access for using a service may require a degree of life cycle maturity that has
2752 demonstrated sufficient testing over a certain size community. Alternately, the policy may specify that a
2753 service in an earlier phase of its life cycle may be made available to a smaller, more technically
2754 sophisticated group in order to collect the metrics that would eventually allow the service to advance its
2755 life cycle status.

2756 This aspect of governance is tightly connected to description because, given a well-behaved set of
2757 services, it is the responsibility of the consumer (or policies promulgated by the consumer's organization)
2758 to decide whether a service is sufficient for that consumer's intended use. The goal is to avoid global
2759 governance specifying criteria that are too restrictive or too lax for local needs of which global governance
2760 has little insight.

2761 Such an approach to specifying governance allows independent domains to describe services in local
2762 terms while still having the services available for informed use across domains. In addition, changes to
2763 the attribute sets within a domain can be similarly described, thus supporting the use of newly described
2764 resources with the existing ones without having to update the description of the entire legacy content.

2765 **5.1.3.2.3 Governance of Participant Interaction**

2766 Finally, given a reliable services infrastructure and a predictable set of services, the third aspect of
2767 governance is prescribing what is required during a service interaction.

2768 Governance would specify adherence to service interface and service reachability parameters and would
2769 require that the result of an interaction correspond to the real world effects as contained in the service
2770 description. Governance would ensure preconditions for service use are satisfied, in particular those
2771 related to security aspects such as user authentication, authorization, and non-repudiation. If conflicts
2772 arise, governance would specify resolution processes to ensure appropriate agreements, policies, and
2773 conditions are met.

2774 It would also rely on sufficient monitoring by the SOA infrastructure to ensure services remain well-
2775 behaved during interactions, e.g. do not use excessive resources or exhibit other prohibited behavior.
2776 Governance would also require that policy agreements as documented in the execution context for the
2777 interaction are observed and that the results and any after effects are consistent with the agreed policies.
2778 Here, governance focuses more on contractual and legal aspects rather than the precursor descriptive
2779 aspects. SOA governance may prescribe the processes by which SOA-specific policies are allowed to
2780 change, but there are probably more business-specific policies that will be governed by processes
2781 outside SOA governance.

2782 **5.1.3.3 Overarching Governance Concerns**

2783 There are numerous governance related concerns whose effects span the three areas just discussed.
2784 One is the area of standards, how these are mandated, and how the mandates may change. The Web
2785 Services standards stack is an example of relevant standards.

2786 Standards are critical to creating a SOA ecosystem where SOA services can be introduced, used
2787 singularly, and combined with other services to deliver complex business functionality. As with other
2788 aspects of SOA governance, the governance body should identify the minimum set felt to be needed and
2789 rigorously enforce that that set be used where appropriate. The governance body takes care to expand
2790 and evolve the mandated standards in a predictable manner and with sufficient technical guidance that
2791 new services are able to coexist as much as possible with the old, and changes to standards do not
2792 cause major disruptions.

2793 Another area that may see increasing activity as SOA expands is additional regulation by governments
2794 and associated legal institutions. New laws may deal with transactions that are service based, possibly

2795 including taxes on the transactions. Disclosure laws may mandate certain elements of description so both
2796 the consumer and provider act in a predictable environment and are protected from ambiguity in intent or
2797 action. Such laws spawn rules and regulations that will influence the metrics collected for evaluation of
2798 compliance.

2799 **5.1.3.4 Considerations for SOA Governance**

2800 The Reference Architecture definition of a loosely coupled system is one in which the constraints on the
2801 interactions between components are minimal yet sufficient to permit interoperability without additional
2802 constraints that may be an artifact of implementation technology. While governance experience for
2803 standalone systems provides useful guides, we must be careful not to apply constraints that would
2804 preclude the flexibility, agility, and adaptability we expect to realize from a SOA ecosystem.

2805 One of the strengths of the SOA paradigm is it can make effective use of diversity rather than requiring
2806 monolithic solutions. Heterogeneous organizations can interact without requiring each conforms to
2807 uniform tools, representation, and processes. However, with this diversity comes the need to adequately
2808 define those elements necessary for consistent interaction among systems and participants, such as
2809 which communication protocol, what level of security, which vocabulary for payload content of messages.
2810 The solution is not always to lock down these choices but to standardize alternatives and standardize the
2811 representations through which an unambiguous identification of the alternative chosen can be conveyed.
2812 For example, the URI standard specifies the URI string, including what protocol is being used, what is the
2813 target of the message, and how parameters may be attached. It does not limit the available protocols, the
2814 semantics of the target address, or the parameters that can be transferred. Thus, as with our definition of
2815 loose coupling, it provides absolute constraints but minimizes which constraints it imposes.

2816 There is not a one-size-fits-all governance but a need to understand the types of things governance is
2817 called upon to do in the context of the goals of the SOA paradigm. Some communities may initially desire
2818 and require very stringent governance policies and procedures while others see need for very little. Over
2819 time, best practices will evolve, resulting in some consensus on a sensible minimum and, except in
2820 extreme cases where it is demonstrated to be necessary, a loosening of strict governance toward the
2821 best practice mean.

2822 A question of how much governance may center on how much time governance activities require versus
2823 how quickly is the system being governed expected to respond to changing conditions. For large single
2824 systems that take years to develop, the governance process could move slowly without having a serious
2825 negative impact. For example, if something takes two years to develop and the steps involved in
2826 governance take two months to navigate, then the governance can go along in parallel and may not have
2827 a significant impact on system response to changes. Situations where it takes as long to navigate
2828 governance requirements as it does to develop a response are examples where governance may need to
2829 be reevaluated as to whether it facilitates or inhibits the desired results. Thus, the speed at which services
2830 are expected to appear and evolve must be considered when deciding the processes for control. The
2831 added weight of governance should be appropriate for overall goals of the application domain and the
2832 service environment.

2833 Governance, as with other aspects of any SOA implementation, should start small and be conceptualized
2834 in a way that keeps it flexible, scalable, and realistic. A set of useful guidelines would include:

- 2835 • Do not hardwire things that will inevitably change. For example, develop a system that uses the
2836 representation of policies rather than code the policies into the implementations.
- 2837 • Avoid setting up processes that demo well for three services without considering how they may
2838 work for 300. Similarly, consider whether the display of status and activity for a small number of
2839 services will also be effective for an operator in a crisis situation looking at dozens of services,
2840 each with numerous, sometimes overlapping and sometimes differing activities.
- 2841 • Maintain consistency and realism. A service solution responding to a natural disaster cannot be
2842 expected to complete a 6-week review cycle but be effective in a matter of hours.

2843 **5.1.4 Architectural Implications of SOA Governance**

2844 The description of SOA governance indicates numerous architectural requirements on the SOA
2845 ecosystem:

- 2846 • Governance is expressed through policies and assumes multiple use of focused policy modules
2847 that can be employed across many common circumstances. The following are thus **REQUIRED**:
2848
 - 2849 ○ descriptions to enable the policy modules to be visible, where the description **SHOULD**
2850 include a unique identifier for the policy as well as a sufficient, and preferably machine
2851 process-able, representation of the meaning of terms used to describe the policy, its
2852 functions, and its effects;
 - 2853 ○ one or more discovery mechanisms that enable searching for policies that best meet the
2854 search criteria specified by a participant; where the discovery mechanism will have
2855 access to the individual policy descriptions, possibly through some repository
2856 mechanism;
 - 2857 ○ accessible storage of policies and policy descriptions, so participants can access,
2858 examine, and use the policies as defined.
- 2859 • Governance requires that the participants understand the intent of governance, the structures
2860 created to define and implement governance, and the processes to be followed to make
2861 governance operational. This **REQUIRES**:
2862
 - 2863 ○ an information collection site, such as a Web page or portal, where governance
2864 information is stored and from which the information is always available for access;
 - 2865 ○ a mechanism to inform participants of significant governance events, such as changes in
2866 policies, rules, or regulations;
 - 2867 ○ accessible storage of the specifics of Governance Processes;
 - 2868 ○ SOA services to access automated implementations of the Governance Processes
- 2869 • Governance policies are made operational through rules and regulations. This **REQUIRES**:
2870
 - 2871 ○ descriptions to enable the rules and regulations to be visible, where the description
2872 **SHOULD** include a unique identifier and a sufficient, and preferably a machine process-
2873 able, representation of the meaning of terms used to describe the rules and regulations;
 - 2874 ○ one or more discovery mechanisms that enable searching for rules and regulations that
2875 may apply to situations corresponding to the search criteria specified by a participant;
2876 where the discovery mechanism will have access to the individual descriptions of rules
2877 and regulations, possibly through some repository mechanism;
 - 2878 ○ accessible storage of rules and regulations and their respective descriptions, so
2879 participants can understand and prepare for compliance, as defined.
 - 2880 ○ SOA services to access automated implementations of the Governance Processes.
- 2881 • Governance implies management to define and enforce rules and regulations. Management is
2882 discussed more specifically in section 5.3, but in a parallel to governance, management
2883 **REQUIRES**:
2884
 - 2885 ○ an information collection site, such as a Web page or portal, where management
2886 information is stored and from which the information is always available for access;
 - 2887 ○ a mechanism to inform participants of significant management events, such as changes
2888 in rules or regulations;
 - 2889 ○ accessible storage of the specifics of processes followed by management.
- 2890 • Governance relies on metrics to define and measure compliance. This **REQUIRES**:
2891
 - 2892 ○ the infrastructure monitoring and reporting information on SOA resources;
 - 2893 ○ possible interface requirements to make accessible metrics information generated or
2894 most easily accessed by the service itself.

2890 5.2 Security Model

2891 Security is one aspect of confidence – the confidence in the integrity, reliability, and confidentiality of the
2892 system. In particular, security in a SOA ecosystem focuses on those aspects of assurance that involve
2893 the accidental or malicious intent of other people to damage, compromise trust, or hinder the availability
2894 of SOA-based systems to perform desired capability.

2895 Security

2896 The set of mechanisms for ensuring and enhancing **trust** and confidence in the **SOA ecosystem**.

2897 Although many of the same principles apply equally to SOA as they do to other systems, implementing
2898 security for a SOA ecosystem is somewhat different than for other contexts. The distributed nature of
2899 SOA brings challenges related to the protection of resources against inappropriate access, and because

2900 SOA embraces the crossing of ownership boundaries, the security issues associated with the movement
2901 of data and access to functionality become more apparent in a SOA ecosystem.

2902 Any comprehensive security solution for a SOA-based system must take into account that people are
2903 effectively managing, maintaining, and utilizing the system appropriately. The roles and responsibilities of
2904 the actors, and the relationships between them must also be explicitly understood and incorporated into a
2905 solution: any security assertions that may be associated with particular interactions originate in the people
2906 that are behind the interaction.

2907 We analyze security in terms of the social structures that define the legitimate permissions, obligations
2908 and roles of people in relation to the system, and mechanisms that must be put into place to realize a
2909 secure system. The former are typically captured in a series of security policy statements; the latter in
2910 terms of security guards that ensure that policies are enforced.

2911 How and when to apply these derived security policy mechanisms is directly associated with the
2912 assessment of the *threat model* and a *security response model*. The threat model identifies the kinds of
2913 threats that directly impact the messages, services, and/or the application of constraints. The response
2914 model is the proposed mitigation to those threats. Properly implemented, the result can be an acceptable
2915 level of risk to the safety and integrity within the SOA ecosystem.

2916 **5.2.1 Secure Interaction Concepts**

2917 We can characterize secure interactions in terms of key security concepts: confidentiality, integrity,
2918 authentication, authorization, non-repudiation, and availability [ISO/IEC 27002]. The concepts for secure
2919 interactions are well -defined in several other standards and publications. The security concepts are
2920 therefore not explicitly defined here, but are discussed related to the SOA ecosystem perspective of the
2921 SOA-RAF.

2922 Related to the security goals in this section, there may be significant security policy differences between
2923 participants in different ownership domains. It is therefore important that these security policies and
2924 security parameters are negotiated at the start of the relationship between systems of differing ownership
2925 domains, and also when policies change between these domains. As with other policy conflicts, this is not
2926 to say that every policy negotiation is a custom, point-to-point interaction. Rather, common mechanisms
2927 and policies should be well known and appropriately accessible so the negotiation can be efficient and
2928 lead to predictable conclusions. Unnecessary complexity does not lead to effective security.

2929 **5.2.1.1 Confidentiality**

2930 Confidentiality is concerned with the protection of privacy of participants in their interactions.
2931 Confidentiality refers to the assurance that unauthorized entities are not able to read messages or parts
2932 of messages that are transmitted, and is typically implemented by using encryption. Confidentiality has
2933 degrees: in a completely confidential exchange, third parties would not even be aware that a confidential
2934 exchange has occurred. In some cases, the identities of the participants may be known but the content of
2935 the exchange obscured. In other cases, only portions of sensitive data in the exchange are encrypted.

2936 Different ownership domains may have policies related to encryption mechanisms between consumers
2937 and providers, and such policies need to be negotiated and understood prior to any interaction.

2938 **5.2.1.2 Integrity**

2939 Integrity refers to the assurance that information has not been altered in transit, and is concerned with the
2940 protection of information that is exchanged – either from inadvertent or intentional corruption. Section
2941 5.2.4 describes common computing techniques for providing both confidentiality and integrity during
2942 message exchanges.

2943 **5.2.1.3 Authentication**

2944 Authentication is concerned with adequately identifying actors in a potential interaction or joint action.
2945 Various mechanisms and protocols can be used to achieve this goal. A combination of **identifiers** (as
2946 discussed in section 3.2.4.1) and other attributes of an actor is typically used to achieve this. The set of
2947 attribute values that claim to identify a specific actor are matched against the set of reference values

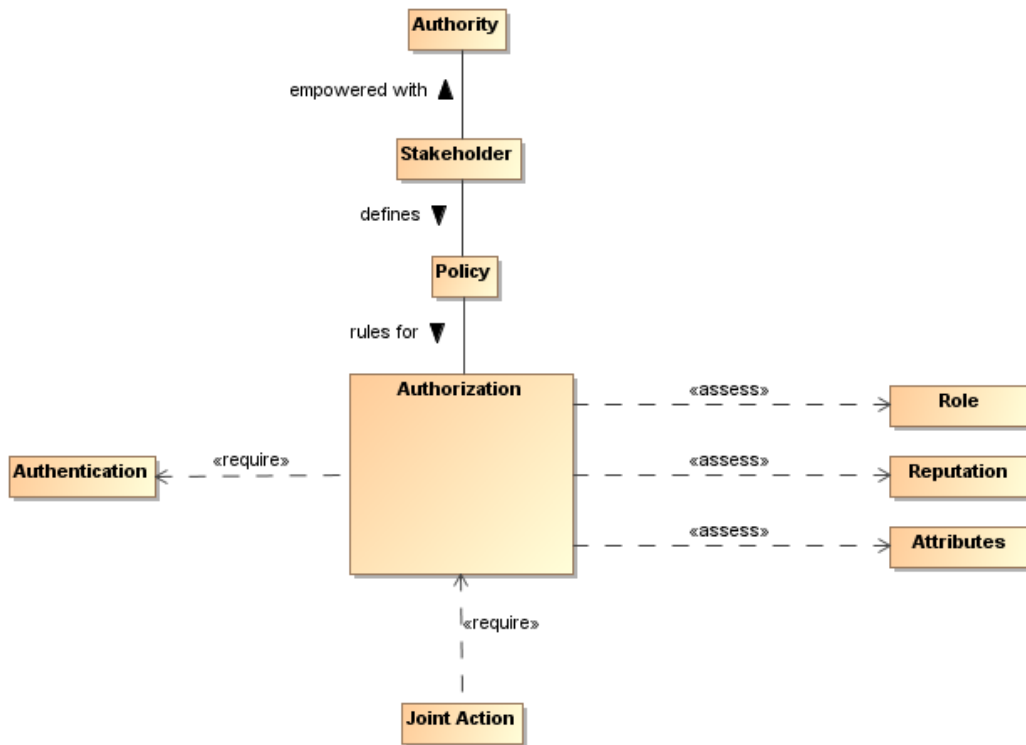
2948 expected for that actor and that are maintained by some trusted authority. If the comparison results in a
2949 sufficient match, authentication has been achieved. Which specific set of attributes is considered an
2950 adequate basis for comparison will be context-dependent and specifying such sets is not within the scope
2951 of the SOA-RAF.

2952 In addition to the concern of adequately identifying each actor involved in the interaction, there may also
2953 be a need to provide authentication information related to the subject that initiated an interaction involving
2954 the combination of intermediary actors in a service orchestration scenario. In such a case, consumers
2955 and services work *on behalf of* the initiator of the interaction, and there may need to be mechanisms in
2956 place to identify the interaction initiator. This concern is covered later in section 5.2.5.

2957 Authentication merely provides an assertion that an actor is the person or agent that it claims to be. Of
2958 itself, it does not provide a 'green light' to proceed with the interaction – this is rather the concern of
2959 **authorization**, covered below.

2960 5.2.1.4 Authorization

2961 Authorization concerns the legitimacy of the interaction, providing assurance that the actors have
2962 permission to participate in the interaction. Authorization refers to the means by which a stakeholder may
2963 be assured that the information and actions that are exchanged are either explicitly or implicitly approved.



2964
2965 *Figure 40 - Authorization*

2966 The role of access control policy for security is to permit stakeholders to express their choices. In Figure
2967 40, such a policy is a written constraint and the role, reputation, and attribute assertions of actors are
2968 evaluated according to the constraints in the authorization policy. A combination of security mechanisms
2969 and their control via explicit policies can form the basis of an authorization solution.

2970 The roles and attributes which provide a participant's credentials are expanded to include reputation.
2971 Reputation often helps determine willingness to interact; for example, reviews of a service provider will
2972 influence the decision to interact with the service provider. The roles, reputation, and attributes are
2973 represented as assertions measured by authorization decision points.

2974 **5.2.1.5 Non-repudiation**

2975 Non-repudiation concerns the accountability of participants. To foster trust in the performance of a system
2976 used to conduct shared activities, it is important that the participants are not able to later deny their
2977 actions: to repudiate them. Non-repudiation refers to the means by which a participant may not, at a later
2978 time, successfully deny having participated in the interaction or having performed the actions as reported
2979 by other participants.

2980 **5.2.1.6 Availability**

2981 Availability concerns the ability of systems to use and offer the services for which they were designed. An
2982 example of threats against availability is a Denial Of Service (DoS) attack in which attackers attempt to
2983 prevent legitimate access to service or set of services by flooding them with bogus requests. As
2984 functionality is distributed into services in a SOA ecosystem, availability protection is paramount.

2985 **5.2.2 Where SOA Security is Different**

2986 The distributed nature of the SOA ecosystem brings challenges related to the protection of resources
2987 against inappropriate access, and because the SOA paradigm embraces the crossing of ownership
2988 boundaries, providing security in such an environment provides unique challenges. The evolution of
2989 sharing information within a SOA ecosystem requires the flexibility to dynamically secure computing
2990 interactions where the owning social groups, roles, and authority are constantly changing as described in
2991 section 5.1.3.1.

2992 Standards for security, as is the case with all aspects of SOA implementation and use, play a large role in
2993 flexible security on a global scale. SOA security may also involve greater auditing and reporting to adhere
2994 to regulatory compliance established by governance structures.

2995 **5.2.3 Security Threats**

2996 There are a number of ways in which an attacker may attempt to compromise the security within a SOA
2997 ecosystem, primarily as attacks on the security concerns listed in section 5.2.1. The two primary sources
2998 of attack are (1) third parties attempting to subvert interactions between legitimate participants; and (2)
2999 entities that are participating but attempting to subvert other participants.

3000 In a SOA ecosystem where there may be multiple ownership boundaries and trust boundaries, it is
3001 important to understand these threats and protections that must be effective. Each technology choice in
3002 the realization of a SOA-based system can potentially have many threats to consider. Although these
3003 threats are not unique to SOA and can be mitigated by applying cryptographic techniques (digital
3004 signatures, encryption, and various cryptographic protocols) and security technologies, it is important that
3005 such threats are understood in order to provide solutions for thwarting such attacks and minimizing risk.

3006 **5.2.3.1 Message alteration**

3007 If an attacker is able to modify the content (or even the order) of messages that are exchanged without
3008 the legitimate participants being aware of it then the attacker has successfully compromised the security
3009 of the system. In effect, the participants may unwittingly serve the needs of the attacker rather than their
3010 own. Cryptographic mechanisms (hash codes, digital signatures, and cryptographic protocols) can be
3011 used as a protection mechanism against alteration.

3012 **5.2.3.2 Message interception**

3013 If an attacker is able to intercept and understand messages exchanged between participants, then the
3014 attacker may be able to gain advantage. Cryptographic protocols can be used as a protection against
3015 interception.

3016 **5.2.3.3 Man in the middle**

3017 In a man-in-the-middle attack, the legitimate participants believe that they are interacting with each other;
3018 but are in fact interacting with an attacker. The attacker attempts to convince each participant that he is
3019 their correspondent; whereas in fact he is not.

3020 In a successful man-in-the-middle attack, legitimate participants do not have an accurate understanding
3021 of the state of the other participants. The attacker can use this to subvert the intentions of the participants.

3022 **5.2.3.4 Spoofing**

3023 In a spoofing attack, the attacker convinces a participant that he is another party.

3024 **5.2.3.5 Denial of service attack**

3025 A Denial of Service (DoS) attack is an attack on the availability and performance of a service or set of
3026 services. In a DoS attack, the attacker attempts to prevent legitimate use of the service. A DoS attack is
3027 easy to mount and can cause considerable harm by preventing legitimate interactions in a SOA
3028 ecosystem, or by slowing them down enough, the attacker may be able to simultaneously prevent
3029 legitimate access to a service and to attack the service by another means. One of the features of a DoS
3030 attack is that it does not require valid interactions to be effective: responding to invalid messages also
3031 takes resources and that may be sufficient to cripple the target. A variation of the DoS attack is the
3032 Distributed Denial of Service (DDoS) attack, where an attacker uses multiple agents to attack the target.

3033 **5.2.3.6 Replay attack**

3034 In a replay attack, the attacker captures the message traffic during a legitimate interaction and then
3035 replays part of it to the target. The target is persuaded that an interaction similar to the previous one is
3036 being repeated and it responds as though it were a legitimate interaction.

3037 **5.2.3.7 False repudiation**

3038 In false repudiation, a normal interaction is completed but an actor later attempts to deny that the
3039 interaction occurred.

3040 **5.2.4 Security Responses**

3041 Security goals are never absolute: it is not possible to guarantee 100% confidentiality, non-repudiation,
3042 etc. However, a well-designed and implemented security response model can reduce security risk to
3043 acceptable levels. For example, using a well-designed cipher to encrypt messages may make the cost of
3044 breaking communications so great and so lengthy that the information obtained is valueless.

3045 Performing threat assessments, devising mitigation strategies, and determining acceptable levels of risk
3046 are the foundation for an effective process to mitigating threats in a cost-effective way.¹⁰ Architectural
3047 choices, as well as choices in hardware and software to realize a SOA implementation will be used as the
3048 basis for threat assessments and mitigation strategies.

¹⁰ In practice, there are perceptions of security from all participants regardless of ownership boundaries. Satisfying security policy often requires asserting sensitive information about the message initiator. The perceptions of this participant about information privacy may be more important than actual security enforcement within the SOA ecosystem for this stakeholder.

3049 **5.2.4.1 Privacy Enforcement**

3050 The most efficient mechanism to assure confidentiality is the encryption of information. Encryption is
3051 particularly important when messages must cross trust boundaries; especially over the Internet. Note that
3052 encryption need not be limited to the content of messages: it is possible to obscure even the existence of
3053 messages themselves through encryption and 'white noise' generation in the communications channel.

3054 The specifics of encryption are beyond the scope of this Reference Architecture Framework. However, we
3055 are concerned about how the connection between privacy-related policies and their enforcement is made.

3056 Service contracts may express confidentiality security policies and the cryptographic mechanisms
3057 required (e.g. ciphers, cryptographic protocols). Between ownership boundaries, there may also be
3058 similar security policies that define requirements for privacy between them. Between such boundaries,
3059 there may be a Policy Enforcement Point (PEP) for enforcing such requirements which may, for example,
3060 automatically encrypt messages as they leave a trust boundary; or perhaps simply ensuring that such
3061 messages are suitably encrypted in such a way as to comply with the policy.

3062 **5.2.4.2 Integrity Protection**

3063 To protect against message tampering or inadvertent message alteration, messages may be
3064 accompanied by the digital signature of the hash code of a message. Any alteration of the message or
3065 signature would result in a failed signature validation, indicating an integrity compromise. Digital
3066 signatures therefore provide a mechanism for integrity protection.

3067 A digital signature also provides non-repudiation, which is an assurance of proof that a subject signed a
3068 message. Utilizing a digital signature algorithm based on public key cryptography, a digital signature
3069 cryptographically binds the signer of the message to its contents, ensuring that the signer cannot
3070 successfully deny sending the message.

3071 The use of a Public Key Infrastructure (PKI) provides the support and infrastructure for digital signature
3072 capabilities, and there may also be security policies related to digital signatures between organizational
3073 boundaries, as well as trust relationships between multiple Certificate Authorities (CAs) across the
3074 boundaries.

3075 **5.2.4.3 Message Replay Protection**

3076 To protect against replay attacks, messages may also contain information that can be used to detect
3077 replayed messages. A common approach involves the use of a message ID, a timestamp, and the
3078 message's intended destination, signed along with the message itself. A message recipient may be able
3079 to thwart a message replay attack by

- 3080 • checking to ensure that it has previously not processed the message ID
- 3081 • validating that the timestamp is within a certain time threshold to ensure message freshness
- 3082 • ensuring that the recipient is indeed the intended destination
- 3083 • validating the digital signature, which provides non-repudiation of the message sender and
3084 checks the integrity of the message ID, timestamp, the destination, and the message itself,
3085 proving that none of the information was altered

3086 Cryptographic protocols between participants can also be used to thwart replay attacks.

3087 **5.2.4.4 Auditing and Logging**

3088 False repudiation involves a participant denying that it authorized a previous interaction. In addition to the
3089 use of digital signatures, an effective strategy for responding to such a denial involves logging of
3090 interactions and the ability to audit the resulting logs. The more detailed and comprehensive an audit trail
3091 is, the less likely it is that a false repudiation would be successful.

3092 Given the distributed nature of the SOA ecosystem, one challenge revolves around the location of the
3093 audit logs of services. It would be very difficult, for example, to do cross-log analysis of services that write
3094 logs to their own file system. For this reason, a common approach revolves around the use of auditing

3095 services, where services may stream auditing information to a common auditing component which can
3096 then be used to provide interaction analysis and a common view.

3097 **5.2.4.5 Graduated engagement**

3098 Although many DoS attacks can typically be thwarted by intrusion detection systems, they are sometimes
3099 difficult to detect because requests to services seem to be legitimate. It is therefore prudent to exercise
3100 care in the use of resources when responding to requests. If a known consumer tries to interact via a
3101 public interface that is not specified in the service contract, a service is not obliged to recognize or
3102 acknowledge such an interaction request. In order to avoid vulnerability to DoS attacks, a service provider
3103 should be careful not to commit resources beyond those implied by the current state of interactions; this
3104 permits a graduation in commitment by the service provider that mirrors any commitment on the part of
3105 service consumers and attackers alike. A successful approach, however, cannot be implemented at the
3106 service-level alone – it involves a defense-in-depth strategy, coupling the use of intrusion detection
3107 systems, routers, firewalls, etc. with the protections discussed in this section.

3108 **5.2.5 Access Control**

3109 **5.2.5.1 Conveying Authentication and Authorization Information**

3110 When an actor initiates an interaction with a service, that service may call other services or be part of a
3111 chain of service interactions as it carries out its functionality. Any service provider is aware of the
3112 immediate service consumer but, in some cases, for example, to provide proper access control to its data,
3113 a service provider may want information on who besides the immediate consumer is expected to see the
3114 data that is being requested. A significant question is whether trust of the immediate consumer should
3115 include trust that the immediate consumer will ensure proper data handling by its immediate consumer
3116 and back through any chain of service interactions. If this is not sufficient, conveying authentication and
3117 authorization information becomes a necessity, and the challenge becomes one of creating a conveyance
3118 process that gives more assurance than merely trust of the immediate consumer. This is a challenge both
3119 within and between ownership domains.

3120 The security concerns related to conveying authentication and authorization information throughout
3121 intermediaries introduce significant complexity. Although an actor may directly authenticate to a service
3122 provider, that service provider may interact with other service providers in order to carry out its
3123 functionality, possibly without the knowledge of the initiator. There may therefore be privacy and
3124 confidentiality concerns related to conveying security information about the initiating actor. There may
3125 also be issues related to authorization, in that the initiating actor may need to explicitly delegate consent
3126 for intermediate services to act on the initiator's behalf.

3127 The following sections cover two approaches for conveying authentication and authorization information
3128 in a SOA ecosystem. These approaches involves conveying sufficient attributes, as discussed in section
3129 5.2.1.3, which may be a single unique identifier or a set of identifiers that can be used in access control
3130 decisions.

3131 In the first approach, the service consumer creates and passes an assertion about the initiating actor. In
3132 the second approach, a service is trusted to issue assertions about subjects. Each has specific
3133 implications for a SOA ecosystem.

3134 5.2.5.1.1 Sender-Vouches Approaches

3135 In a “sender vouches” approach, a service consumer creates an assertion, *vouching* for certain security
3136 information about the initiator of the interaction, and possibly about other actors in a series (chain) of
3137 service interactions. This assertion contains sufficient attributes that can be used in access control
3138 decisions, and is sent, or propagated, to the service provider. Trust of such an assertion is therefore
3139 based on the provider’s trust of the consumer, and also there needs to be an understanding of such
3140 assertions between ownership boundaries. In a SOA ecosystem, such trust must be established at the
3141 beginning of each relationship.

3142 When such assertions are reused in service orchestration scenarios beyond the initial consumer-provider
3143 interaction, there can be significant security risks¹¹.

- 3144 • *Trust of Message Senders*. Because the trust of the assertion is based on the trust of the
3145 message senders, the more intermediaries there are, trust can degrade as the distance between
3146 the initiator and the service being called becomes greater. Trust may, therefore, be dependent on
3147 the trust of every sender in the chain to properly pass the claim.
- 3148 • *Risk of Vulnerabilities in Intermediaries*. Because the trust of the assertion relies on the trust of
3149 each participant in the interaction, a risk is that intermediary services may become compromised
3150 and may inaccurately send false claims. Depending on the exact messaging syntax, an
3151 intermediary service could potentially manipulate the assertion or substitute another assertion.
3152 There could also be impersonation of the intermediary services, affecting the reliability of the
3153 interaction.

3154 Approaches for mitigating risks in sender-vouches approaches involve a careful combination of SOA
3155 security governance, limiting the re-use of assertions beyond a certain number of points, establishing
3156 conditions of use for propagated assertions, keeping track of the history of the assertion in the interaction,
3157 and the use of digital signatures by an asserting party.

3158 Between ownership domains, such an approach is even more challenging, as different ownership
3159 domains may recognize different authentication authorities and may not recognize identities from other
3160 organizations. Security policies that relate to the conveying of security information across boundaries
3161 must occur at the start of the relationship, with many solutions involving reciprocity of trust between
3162 authentication and authorization authorities from each domain.

3163 5.2.5.1.2 Token Service-based Approaches

3164 This approach revolves around use of a *token service* or a set of token services trusted to vouch for
3165 security information about authenticated actors in the interaction. In this approach, a token service issues
3166 a token which is an assertion that contains sufficient attributes that can be used in access control
3167 decisions. The service consumer passes this token, along with a request, to a service provider.

3168 After the original consumer passes the issued token to the service, the recipient service later acting as a
3169 consumer may then choose to propagate the token to other service providers. Much like the risks
3170 associated with the reuse of assertions in sender-vouches approaches, there are risks associated with
3171 the reuse of tokens issued by the token service beyond the initial consumer-provider interaction. Most
3172 token service protocols and specifications, therefore, provide the capability for “refreshing” tokens for
3173 reuse in such situations. In this case, each actor retrieving a token may request that the token service
3174 issue a “refresh token” that can be propagated for a subsequent service interaction. Utilizing refresh
3175 tokens removes the risks associated with reuse.

¹¹ Such risks and others are documented in [SMITH]

3176 This approach differs from the sender-vouches model in that trust of the token is not based on the
3177 message sender, but is based on the trust of the token service that issued it. In interactions between
3178 ownership domains, the establishment of the trust of the token services must be agreed to at the start of
3179 the relationship, and there must be an understanding of the policies associated with processing the
3180 tokens. To facilitate this, token services in one domain can often be used to “translate” tokens from other
3181 domains, issuing new tokens that are understood by services and consumers in its domain.

3182 Unlike sender-vouches approaches, the token service approach revolves around a trusted token service
3183 or a set of trusted token services, and there may be architectural implications related to performance and
3184 availability. It is therefore advised that solutions that provide elastic scalability be used to ensure that
3185 token services are readily available to respond to requests.

3186 **5.2.5.2 Access Control Approaches**

3187 Access control revolves around security policy. If access control policy can be discovered and processed,
3188 and if authorization credentials of actors can be retrieved, access control can be successfully enforced.
3189 Architectural flexibility for authorization is achieved by logically separating duties into Policy Decision
3190 Points (PDPs) and Policy Enforcement Points (PEPs). A PDP is the point at which access control
3191 decisions are made, based on an expressed access control policy and an actor’s authorization
3192 credentials. The enforcement of the decision is delegated to a PEP. Some standards, such as XACML
3193 (the eXtensible Access Control Markup Language), decompose the policy model further into Policy
3194 Administration Points (PAPs) that create policy and the Policy Information Points (PIPs) that query
3195 attributes for actors requesting access to resources. There are many strategies for how PDPs and PEPs
3196 can work together, each with architectural implications that have an impact on security, performance, and
3197 scalability.

3198 As access control policy may vary between ownership domains, the negotiation of access control policies
3199 between such domains must occur at the start of the relationship, regardless of the underlying
3200 architectural approaches.

3201 Different security services implementations may dictate different architectural approaches and have
3202 different implications. This section provides a brief overview of such approaches.

3203 **5.2.5.2.1 Centralized Access Control Approaches**

3204 A centralized approach uses a policy server (or a set of policy servers) to act as a PDP, and utilizes the
3205 current access control policy to make an access control decision for an actor requesting access to a
3206 resource. A positive aspect of this approach can be information hiding because services may not need to
3207 know the authorization credentials of the actor or the specific policy being enforced. The centralized
3208 model protects that information in cases where this information may be sensitive or confidential. Another
3209 positive aspect of this approach is that the policy services can provide access control decisions
3210 consistently, and any change to access control policy can be changed in one place.

3211 However, negative aspects of this model are those common with any type of centralized architecture,
3212 including performance and availability. Given performance, availability, and scalability concerns, any
3213 centralized solution should be coupled with alternative approaches for greater flexibility.

3214 **5.2.5.2.2 Decentralized Access Control Approaches**

3215 In a decentralized approach, the service consumer propagates a token related to its identity (and possibly
3216 other identities in a service chain), and this is assessed by a “local” PDP and PEP. The service PDP
3217 refers to locally expressed policy, and therefore, its PDP can inspect the policy and the security
3218 credentials propagated in order to make an access control decision. If only identity information about the
3219 initiator is propagated into the service, the service may retrieve additional authorization credentials from
3220 an Attribute Service lookup based on the identity.

3221 The decentralized model alleviates the performance concerns of the purely central model, as it does not
3222 require access to a set of centralized servers used to make access control decisions. Because the policy
3223 is locally expressed, the service may enforce its own policy, expressed in its service contract with service
3224 consumers.

3225 There are two potential concerns with this model. One concern is that there is no information hiding. If an
3226 assertion about the initiator is propagated into the service, the service may need security credentials of
3227 the consumer in order to execute access control policy, and these credentials may be sensitive or
3228 confidential. A second concern revolves around access control policy management. As this decentralized
3229 model is based on making “local” (not centralized) access control decisions at the service level, there is a
3230 possibility that

- 3231 • Access control policies may not be consistently enforced throughout the SOA ecosystem
- 3232 • Changing organizational access control policies require policy changes throughout the SOA
3233 ecosystem (vs. in a central location) and may be therefore difficult to immediately enforce.
- 3234 Therefore, there is a danger that access control policies may be out-of-date and inconsistent.
- 3235 It is therefore prudent that in using such an approach, that these concerns be addressed.

3236 5.2.5.2.3 Hybrid Access Control Approaches

3237 A purely centralized approach has significant weaknesses related to performance, availability, and
3238 scalability; a purely decentralized approach does not support a requirement to have centralized control of
3239 access control policy. In response, hybrid approaches have emerged to provide a “happy medium”
3240 between local control of policy (where services express all policy) and central control of policy (where a
3241 central policy server expresses all policy). In hybrid models, each service can both express local policy
3242 and leverage global organizational policy (which can be periodically downloaded or syndicated to the
3243 local services) in order to make decisions. The balance between the models will depend on the context in
3244 which the hybrid is applied.

3245 5.2.6 Architectural Implications of SOA Security

3246 Providing SOA security in an ecosystem of governed services has the following implications on the policy
3247 support and the distributed nature of mechanisms used to assure SOA security:

- 3248 • Security expressed through security messaging policies **SHOULD** follow the same architectural
3249 implications as described in Section 4.4.3 for policies and contracts architectural implications.
- 3250 • Security policies **MUST** have mechanisms to support security description administration, storage,
3251 and distribution.
- 3252 • Service descriptions **SHOULD** include a sufficiently rich meta-structure to unambiguously indicate
3253 which security policies are required and where policy options are possible.
- 3254 • The mechanisms that make-up the execution context in secure SOA-based systems **SHOULD**:
 - 3255 ○ provide protection of the confidentiality and integrity of message exchanges;
 - 3256 ○ be distributed so as to provide available policy-based identification, authentication, and
3257 authorization;
 - 3258 ○ ensure service availability to consumers;
 - 3259 ○ be able to scale to support security for a growing ecosystem of services;
 - 3260 ○ be able to support security between different communication means or channels;
 - 3261 ○ have a framework for resolving conflicts between security policies.
- 3262 • Common security services **SHOULD** include the ability for:
 - 3263 ○ authentication and establishing/validating credentials
 - 3264 ○ retrieval of authorization credentials (attribute services);
 - 3265 ○ enforcing access control policies;
 - 3266 ○ intrusion detection and prevention;
 - 3267 ○ auditing and logging interactions and security violations.

3268 5.3 Management Model

3269 5.3.1 Management

3270 Management is a process of controlling resources in accordance with the policies and principles defined
3271 by Governance.

3272 There are three separate but linked domains of interest within the management of a SOA ecosystem:

- 3273 1. the management and support of the resources that are involved in any complex structures – of
3274 which SOA ecosystems are excellent examples;
3275 2. the promulgation and enforcement of the policies and service contracts agreed to by the
3276 stakeholders in the SOA ecosystem;
3277 3. the management of the relationships of the participants – both to each other and to the services
3278 that they use and offer.

3279 There are many artifacts related to management. Historically, systems management capabilities have
3280 been organized by the FCAPS functions (based on ITU-T Rec. M.3400 (02/2000), *TMN Management*
3281 *Functions*):

- 3282 • fault management,
- 3283 • configuration management,
- 3284 • account management,
- 3285 • performance and security management.

3286 The primary task of the functional groups is to concentrate on maintaining systems in a trusted, active,
3287 and accessible state.

3288 In the context of the SOA ecosystem, we see many possible resources that may require management
3289 such as services, service descriptions, service contracts, policies, roles, relationships, security, people
3290 and systems that implement services and infrastructure elements. In addition, given the ecosystem
3291 nature, it is also potentially necessary to manage the business relationships between participants.

3292 Successful operation of a SOA ecosystem requires trust among the stakeholders and between them and
3293 the SOA-based system elements. In contrast, regular systems in technology are not necessarily operated
3294 or used in an environment requiring trust before the stakeholders make use of the system. Indeed, many
3295 of these systems exist in hierarchical management structures, within which use may be mandated by
3296 legal requirement, executive decision, or good business practice in furthering the business' strategy. The
3297 pre-condition of trust in the SOA ecosystem is rooted both in the principles of service orientation and in
3298 the distributed, authoritative ownership of independent services. Even for hierarchical management
3299 structures applied to a SOA ecosystem, the service in use should have a contractual basis rather than
3300 solely being mandated.

3301 Trust may be established through agreements/contracts, policies, or implicitly through observation of
3302 repeated interactions with others. Explicit trust is usually accompanied by formalized documents suitable
3303 for management. Implicit trust adds fragility to the management of a SOA ecosystem because failure to
3304 maintain consistent and predictable interactions will undermine the trust between participants and within
3305 the ecosystem as a whole.

3306 Management in a SOA ecosystem is thus concerned with management taking actions that will establish
3307 the condition of trust that must be present before engaging in service interactions. These concerns should
3308 largely be handled within the governance of the ecosystem. The policies, agreements, and practices
3309 defined through governance provide the boundaries within which management operates and for which
3310 management must provide enforcement and feedback. However, governance alone cannot foresee all
3311 circumstances but must offer sufficient guidance where agreement between all stakeholders cannot be
3312 reached. Management in these cases must be flexible and adaptable to handle unanticipated conditions
3313 without unnecessarily breaking trust relationships.

3314 Service management is the process – manual, automated, or a combination – of proactively monitoring
3315 and controlling the behavior of a service or a set of services. Service management operates under
3316 constraints attributed to the business and social context. Specific policies may be used to govern cross-
3317 boundary relationships. Managing solutions based on such policies (and that may be used across
3318 ownership boundaries) raises issues that are not typically present when managing a service within a
3319 single ownership domain. Care is therefore required in managing a service when the owner of the
3320 service, the provider of the service, the host of the service and mediators to the service may all belong to
3321 different stakeholders.

3322 Cross-boundary service management takes place in, at least, the following situations:

- 3323 • using combinations of services that belong to different ownership domains
- 3324 • using of services that mediate between ownership domains
- 3325 • sharing monitoring and reporting means and results.

3326 These situations are particularly important in ecosystems that are highly decentralized, in which the
 3327 participants interact as peers as well as in the 'master-servant' mode.

3328 The management model shown in Figure 41 conveys how the SOA paradigm applies to managing
 3329 services. Services management operates via service metadata, such as properties associated with
 3330 service lifecycles and with service use, which are typically collected in or accessed through the service
 3331 description.

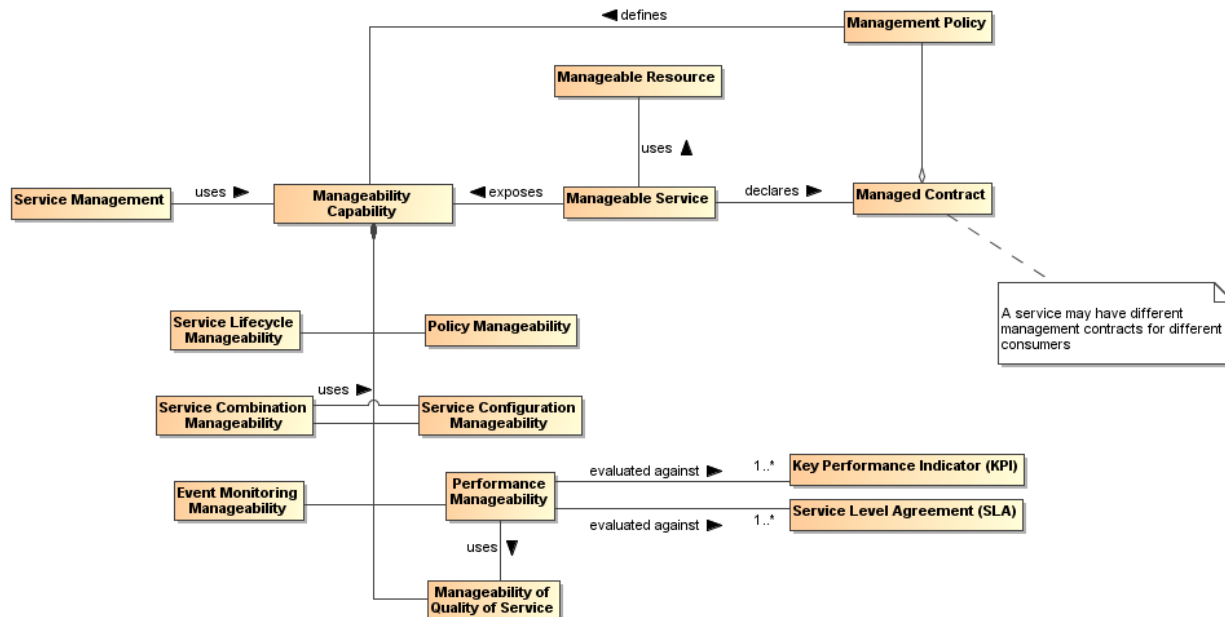


Figure 41 - Management model in SOA ecosystem

3332
 3333 The service metadata of interest is that set of service properties that is manageable. These manageability
 3334 properties are generally identifiable for any service consumed or supplied within the ecosystem. The
 3335 necessary existence of these properties within the SOA ecosystem motivates the following definitions:
 3336

3337 **Manageability**

3338 A capability that allows a **resource** to be controlled, monitored, and reported on with respect to
 3339 some properties.

3340 **Manageability property**

3341 A property used in the **manageability** of a **resource**. The fundamental unit of management in
 3342 systems management.

3343 Note that manageability is not necessarily a part of the managed entities themselves and are generally
 3344 considered to be external to the managed entities.

3345 Each resource may be managed through a number of aspects of management, and the resources may
 3346 be grouped based on similar aspects. For example, resources may be grouped according to the aspect
 3347 referred to as 'Configuration Manageability' for the collection of services. Some resources may not be
 3348 managed under a particular capability if there are no manageability aspects with a clear meaning or use.
 3349 As an example, all resources within a SOA ecosystem have a lifecycle that is meaningful within the
 3350 ecosystem. Thus, all resources are manageable under Lifecycle Manageability. In contrast, not all
 3351 resources report or handle events. Thus, Event Manageability is only concerned with those resources for
 3352 which events are meaningful.

3353 **Life-cycle Manageability** of a service typically refers to how the service is created, how it is retired and
 3354 how service versions must be managed. This manageability is a feature of the SOA ecosystem because
 3355 the service cannot manage its own life cycle. Related properties may include the necessary state of the
 3356 ecosystem for the creation and retirement of the service and the state of the ecosystem following the
 3357 retirement of the service. The SOA ecosystem distinguishes between service composition and service
 3358 aggregation: retiring of service composition leads to retiring of all services comprising the composition

3359 while retiring of service aggregation assumes that comprising services have their own life-cycle and can
3360 be used in another aggregation.

3361 Another important consideration is that services may have resource requirements, such as concurrent
3362 connectivity to a data source, which must be established at various points in the services' life cycles.
3363 However, actual providers of these resources may not be known at the time of the service creation and,
3364 thus, have to be managed at service run-time.

3365 **Combination Manageability** of a service addresses management of service characteristics that allow for
3366 creating and changing combinations in which the service participates or that the service combines itself.
3367 Known models of such combinations are aggregations and compositions. Examples of patterns of
3368 combinations are choreography and orchestration. In cases of business collaboration, combination of
3369 services appears as cooperation of services. Combination Manageability drives implementation of the
3370 Service Composability Principle of service orientation.

3371 Service combination manageability resonates with the methodology of process management.
3372 Combination Manageability may be applied at different phases of service creation and execution and, in
3373 some cases, can utilize Configuration Manageability.

3374 Service combinations typically contribute the most in delivering business values to the stakeholders.
3375 Managing service combinations is one of the most important tasks and features of the SOA ecosystem.

3376 **Configuration Manageability** of a service allows managing the identity of and the interactions among
3377 internal elements of the service, for example, a use of data encryption for internal inter-component
3378 communication in particular deployment conditions. Also, Configuration Manageability correlates with the
3379 management of service versions and configuration of the deployment of new services into the ecosystem.
3380 Configuration Management differs from the Combination Manageability in the scope and scale of
3381 manageability, and addresses lower level concerns than the architectural combination of services.

3382 **Event Monitoring Manageability** allows managing the categories of events of interest related to services
3383 and reporting recognized events to the interested stakeholders. Such events may be the ones that trigger
3384 service invocations as well as execution of particular functionality provided by the service. For example,
3385 an execution of a set of financial market risk services, which implements a choreography pattern, may be
3386 started if a certain financial event occurs in a stock exchange.

3387 Event Monitoring Manageability is a key lower-level manageability aspect, in which the service provider
3388 and associated stakeholders are interested. Monitored events may be internal or external to the SOA
3389 ecosystem. For example, a disaster in the oil industry, which is outside the SOA ecosystem of the Insurer,
3390 can trigger the service's functionality that is responsible for immediate or constant monitoring of oil prices
3391 in the oil trading exchanges and, respectively, modify the premium paid by the insured oil companies.

3392 **Performance Manageability** of a service allows controlling the service results, shared and sharable real
3393 world effects against the business goals and objectives of the service. This manageability assumes
3394 monitoring of the business performance as well as the management of this monitoring itself. Performance
3395 Manageability includes business and technical performance manageability through a performance criteria
3396 set, such as business key performance indicators (KPI) and service-level agreements (SLA).

3397 The performance business- and technical-level characteristics of the service should be known from the
3398 service contract. The service provider and consumer must be able to monitor and measure these
3399 characteristics or be informed about the results measured by a third party. An example of such monitoring
3400 would be when the comparison of service performance results against an SLA is not satisfactory to the
3401 consumer, and as a consequence, the consumer may replace the service by a service from a competitor.

3402 Performance Manageability is the instrument for providing compliance of the service with its service
3403 contracts. Performance Manageability utilizes Manageability of Quality of Service.

3404 **Manageability of Quality of Service** deals with management of service non-functional characteristics
3405 that may be of significant value to the service consumers and other stakeholders in the SOA ecosystem.
3406 A classic example of this is managing bandwidth offerings associated with a service.

3407 Manageability of quality of service assumes that the properties associated with service qualities are
3408 monitored during the service execution. Results of monitoring may be compared against an SLA or a KPI,
3409 which results in the continuous validation of how the service contract is preserved by the service provider.

3410 **Policy Manageability** allows additions, changes and replacements of the policies associated with a
3411 resource in the SOA ecosystem. The ability to manage those policies (such as promulgating policies,

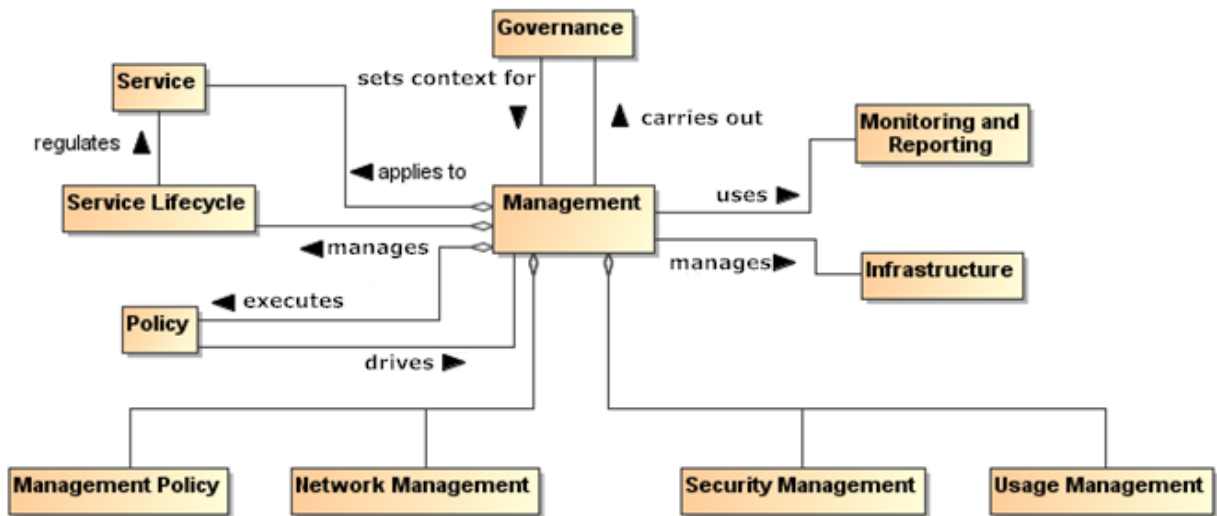
3412 retiring policies and ensuring that policy decision points and enforcement points are current) enables the
3413 ecosystem to apply policies and *evaluate* the results.

3414 The ability to manage, i.e. use a particular manageability, requires policies from governance to be
3415 translated into detailed rules and regulations which are measured and monitored providing corresponding
3416 feedback for enforcement. At the same time, the execution of a management capability must adhere to
3417 certain policies governing the management itself. For example, a management has to enforce and control
3418 policies of compliance with particular industry regulation while the management is obliged by another
3419 policy to report on the compliance status periodically.

3420 Management of SOA ecosystem recognizes the manageability challenge and requires manageability
3421 properties to be considered for all aforementioned manageability cases. In the following subsections, we
3422 describe how these properties are used in the management as well as some relationships between
3423 management and other components of SOA ecosystem.

3424 5.3.2 Management Means and Relationships

3425 A minimal set of management issues for the SOA ecosystem is shown in Figure 42 and elaborated in the
3426 following sections.



3427
3428 *Figure 42 - Management Means and Relationships in a SOA ecosystem*

3429 5.3.2.1 Management Policy

3430 The management of resources within the SOA ecosystem may be governed by management policies. In
3431 a deployed SOA-based solution, it may well be that different aspects of the management of a given
3432 service are managed by different management services. For example, the life-cycle management of
3433 services often involves managing service versions. Managing quality of service is often very specific to
3434 the service itself. For example, the quality of service attributes for a video streaming service are quite
3435 different to those for a banking system.

3436 5.3.2.2 Network Management

3437 Network management deals with the maintenance and administration of large scale physical networks
3438 such as computer networks and telecommunication networks. Specifics of the networks may affect
3439 service interactions from performance and operational perspectives.

3440 Network and related system management execute a set of functions required for controlling, planning,
3441 deploying, coordinating, and monitoring the distributed services in the SOA ecosystem. However, while
3442 recognizing their importance, the specifics of systems management or network management are out of
3443 scope for this Reference Architecture Foundation.

3444 **5.3.2.3 Security Management**

3445 Security Management includes identification of roles, permissions, access rights, and policy attributes
3446 defining security boundaries and events that may trigger a security response.

3447 Security management within a SOA ecosystem is essential to maintaining the trust relationships between
3448 participants residing in different ownership domains. Security management must consider not just the
3449 internal properties related to interactions between participants but ecosystem properties that preserve the
3450 integrity of the ecosystem from external threats.

3451 **5.3.2.4 Usage Management**

3452 Usage Management is concerned with how resources are used, including:

- 3453 • how the resource is accessed, who is using the resource, and the state of the resource (access
3454 properties);
 - 3455 • controlling or shaping demand for resources to optimize the overall operation of the ecosystem
3456 (demand properties);
 - 3457 • assigning costs to the use of resources and distributing those cost assignments to the
3458 participants in an appropriate manner (financial properties).
- 3459

3460 **5.3.3 Management and Governance**

3461 The primary role of governance in the context of a SOA ecosystem is to foster an atmosphere of
3462 predictability, trust, and efficiency, and it accomplishes this by allowing the stakeholders to negotiate and
3463 set the key policies that govern the running of the SOA-based solution. Recall that in an ecosystem
3464 perspective, the goal of governance is less to have complete fine-grained control but more to enable the
3465 individual participants to work together.

3466 Policies for a SOA ecosystem will tend to focus on the rules of engagement between participants; for
3467 example, what kinds of interactions are permissible, how disputes are resolved, etc. While governance
3468 may primarily focus on setting policies, management will focus on the realization and enforcement of
3469 policies. Effective management in the SOA ecosystem requires an ability for governance to understand
3470 the consequences of its policies, guidelines, and principles, and to adjust those as needed when
3471 inconsistencies or ambiguity become evident from the operation of the management functions. This
3472 understanding and adjustment must be facilitated by the results of management and so the mechanisms
3473 for providing feedback from management into governance must exist.

3474 Governance operates via specialized activities and, thus, should be managed itself. Governance policies
3475 are included in the Governance Framework and Processes, and driven by the enterprise business model,
3476 business objectives and strategies. Where corporate management policies exist, these are usually guided
3477 and directed by the corporate executives. In peer relationships, governance policies are set by either an
3478 external entity and accepted by the peers or by the peers themselves. This creates the appropriate
3479 authoritative level for the policies used for the management of the Governance Framework and
3480 Processes. Management to operationalize governance controls the life-cycle of the governing policies,
3481 including procedures and processes, for modifying the Governance Framework and Processes.

3482 **5.3.4 Management and Contracts**

3483 **5.3.4.1 Management for Contracts and Policies**

3484 As we noted above, management can often be viewed as the application of contracts and individual
3485 policies to ensure the smooth running of the SOA ecosystem. Policies and service contracts specify the
3486 service characteristics that have to be monitored, analyzed and managed. These also play an important
3487 role as the guiding constraints for management, as well as being artifacts (e.g., policy and contractual
3488 documents) that also need to be managed.

3489 **5.3.4.2 Contracts**

3490 As described in sections *Participation in a SOA Ecosystem* view and *Realization of a SOA*
3491 *Ecosystem* view, there are several types of contractual information in the SOA ecosystem. From the
3492 management perspective, three basic types of the contractual information relate to:

- 3493 • relationship between service provider and consumer;
- 3494 • communication with the service;
- 3495 • control of the quality of the service execution.

3496 When a consumer prepares to interact with a service, the consumer and the service provider must come
3497 to an agreement on the service features and characteristics that will be provided by the service and made
3498 available to the consumer. This agreement is known as a service contract.

3499 **Service Contract**

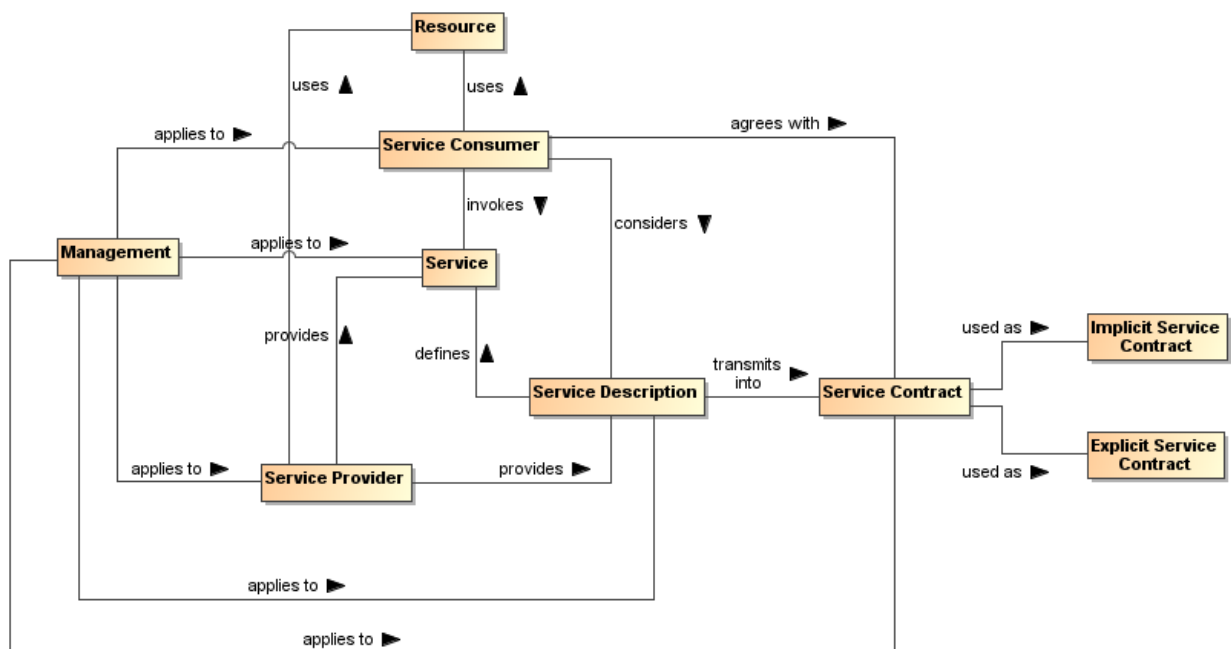
3500 An implicit or explicit documented agreement between the service **consumer** and service
3501 **provider** about the use of the service based on

- 3502 • the commitment by a service provider to provide service functionality and results consistent
3503 with identified **real world effects** and
- 3504 • the commitment by a service **consumer** to interact with the service per specific means and
3505 per specified **policies**,
- 3506 where both consumer and provider actions are in the manner described in the service description.

3507 The service description provides the basis for the service contract and, in some situations, may be used
3508 as an implicit default service contract. In addition, the service description may set mandatory aspects of a
3509 service contract, e.g. for security services, or may specify acceptable alternatives. As an example of
3510 alternatives, the service description may identify which versions of a vocabulary will be recognized, and
3511 the specifics of the contract are satisfied when the consumer uses one of the alternatives. Another
3512 alternative could have a consumer identify a policy they require be satisfied, e.g. a standard privacy policy
3513 on handling personal information, and a provider that is prepared to accept a policy request would
3514 indicate acceptance as part of the service contract by continuing with the interaction. In each of these
3515 cases, the actions of the participants are consistent with an implicit service contract without the existence
3516 of a formal agreement between the participants.

3517 In the case of business services, it is anticipated that the service contract may take an explicit form and
3518 the agreement between business consumer and business service provider is formalized. Formalization
3519 requires up-front interactions between service consumer and service provider. In many business
3520 interactions, especially between business organizations within or across corporate boundaries, a
3521 consumer must have a contractual assurance from the provider or wants to explicitly indicate choices
3522 among alternatives, e.g., only use a subset of the business functionality offered by the service and pay a
3523 prorated

3524 cost.



3525
3526

Figure 43 - Management of the service interaction

3527 Consequently, an implicit service contract is an agreement (1) on the consumer side with the terms,
3528 conditions, features and interaction means specified in the service description "as is" or (2) a selection
3529 from alternatives that are made available through mechanisms included in the service description, and
3530 neither of these require any a priori interactions between the service consumer and the service provider.
3531 For example, a browser interface may display a checked box indicating the consumer agrees to accept
3532 future advertisement; the consumer can uncheck the box to indicate advertisements should not be sent.

3533 An explicit service contract always requires a form of interaction between the service consumer and the
3534 service provider prior to the service invocation. This interaction may regard the choice or selection of the
3535 subset of the elements of the service description or other alternatives introduced through the formal
3536 agreement process that would be applicable to the interaction with the service and affect related joint
3537 action.

3538 Any form of explicit contract couples the service consumer and provider. While explicit contracts may be
3539 necessary or desirable in some cases, such as in supply chain management, commerce often uses a mix
3540 of implicit and explicit contracts, and a service provider may offer (via service description) a conditional
3541 shift from implicit to explicit contract. For example, Twitter offers an implicit contract on the use of its APIs
3542 to any application with the limit on the amount of service invocations; if the application has to use more
3543 invocations, one has to enter into the explicit fee-based contract with the provider. A case where an
3544 implicit contract transforms into an explicit contract may be illustrated when one buys a new computer and
3545 it does not work. The buyer returns the computer for repair under the manufacturer's warranty as stated
3546 by an implicit purchase contract. However, if the repair does not fix the problem and the seller offers an
3547 upgraded model in replacement, the buyer may agree to an explicit contract that limits the rights of the
3548 buyer to make the explicit agreement public.

3549 Control of the quality of the service execution, often represented as a service level agreement (SLA), is
3550 performed by service monitoring systems and includes both technical and operational business controls.
3551 SLA is a part of the service contract and, because of the individual nature of such contracts, may vary
3552 from one service contract to another, even for the same consumer. Typically, a particular SLA in the
3553 service contract is a concrete instance of the SLA declared in the service description.

3554 Management of the service contracts is based on management policies that may be mentioned in the
3555 service description and in the service contracts. Management of the service contracts is mandatory for
3556 consumer relationship management. In the case of explicit service contracts, the contracts have to be
3557 created, stored, maintained, reviewed/controlled and archived/destroyed as needed. All the activities are

3558 management concerns. Explicit service contracts may be stored in specialized repositories that provide
3559 appropriate levels of security.

3560 Management of the service interfaces is based on several management policies that regulate

- 3561 • availability of interfaces specified in the service contracts,
- 3562 • accessibility of interfaces,
- 3563 • procedures for interface changes,
- 3564 • interface versions as well as the versions of all parts of the interfaces,
- 3565 • traceability of the interfaces and their versions back to the service description document.

3566 Management of the SLA is integral to the management of service monitoring and operational service
3567 behavior at run-time. An SLA usually enumerates service characteristics and expected performances of
3568 the service. Since an SLA carries the connotation of a 'promise', monitoring is needed to know if the
3569 promise is being kept. Existence of an SLA itself does not guarantee that the consumer will be provided
3570 with the service level specified in the service contract.

3571 The use of an SLA in a SOA ecosystem can be wider than just an agreement on technical performances.
3572 An SLA may contain remedies for situations where the promised service cannot be maintained, or the
3573 real world effect cannot be achieved due to developments subsequent to the agreement. A service
3574 consumer that acts accordingly to realize the real world effect may be compensated for the breach of the
3575 SLA if the effect is not realized.

3576 Management of the SLA includes, among others, policies to change, update, and replace the SLA. This
3577 aspect concerns service Execution Context because the business logic associated with a defined
3578 interface may differ in different Execution Contexts and affect the overall performance of the service.

3579 **5.3.4.3 Policies**

3580 "Although provision of management capabilities enables a service to become manageable, the extent and
3581 degree of permissible management are defined in management policies that are associated with the
3582 services. Management policies are used to define the obligations for, and permissions to, managing the
3583 service" [WSA]. Management policies, in essence, are the realization of governing rules and regulations.
3584 As such, some management policies may target services while other policies may target the management
3585 of the services.

3586 In practice, a policy without any means of enforcing it is vacuous. In the case of management policy, we
3587 rely on a management infrastructure to realize and enforce management policy.

3588 **5.3.4.4 Service Description and Management**

3589 The service description identifies several management objects such as a set of service interfaces and
3590 related set of SLAs. Service behavioral characteristics and performances specified in the SLA depend on
3591 the interface type and its Execution Context. In the service description, a service consumer can find
3592 references to management policies, SLA metrics, and the means of accessing measured values that
3593 together increase assurance in the service quality. At the same time, service description is an artifact that
3594 must be managed.

3595 In the SOA ecosystem, the service description is the assembled information that describes the service but
3596 it may be reported or displayed in different presentations. While each separate version of the service has
3597 one and only one service description, different categories of service consumers may focus their interests
3598 on different aspects of the service description. Thus, the same service description may be displayed not
3599 only in different languages but also with different cultural and professional accents in the content.

3600 New service description may be issued to reflect changes and update in the service. If the change in the
3601 service does not affect its service description, the new service version may have the same service
3602 description as the previous version except for the updated version identifier. For example, a service
3603 description may stay the same if bugs were fixed in the service. However, if a change in the service
3604 influences any aspects of the service quality that can affect the real world effect resulting from
3605 interactions with the service, the service description must reflect this change even if there are no changes
3606 to the service interface.

3607 Management of the service description as well as of the explicit service contracts is essential for delivery
3608 of the service to the consumer satisfaction. This management can also prevent business problems rooted
3609 in poor communication between the service consumers and the service providers.

3610 Thus, management of service description contains, among others, management of the service description
3611 presentations, the life-cycles of the service descriptions, service description distribution practices and
3612 storage of the service descriptions and related service contracts. Collections of service descriptions in the
3613 enterprise may manifest a need for specialized registries and/or repositories. Depending on the enterprise
3614 policies, an allocation of purposes and duties of registries and repositories may vary but this topic is
3615 beyond the current scope.

3616 **5.3.5 Management for Monitoring and Reporting**

3617 The successful application of management relies on the monitoring and reporting aspects of management
3618 to enable the control aspect. Monitoring in the context of management consists of measuring values of
3619 managed aspects and evaluating that measurement in relationship to some expectation. Monitoring in a
3620 SOA ecosystem is enabled through the use of mechanisms by resources for exposing managed aspects.
3621 In the SOA framework, this mechanism may be a service for obtaining the measurement. Alternatively,
3622 the measurement may be monitored by means of event generation containing updated values of the
3623 managed aspect.

3624 Approaches to monitoring may use a polling strategy in which the measurements are requested from
3625 resources in periodic intervals, in a pull strategy in which the measurements are requested from
3626 resources at random times, or in a push strategy in which the measurements are supplied by the resource
3627 without request. The push strategy can be used in a periodic update approach or in an 'update on
3628 change' approach. Management services must be capable of handling these different approaches to
3629 monitoring.

3630 Reporting is the complement to monitoring. Where monitoring is responsible for obtaining measurements,
3631 reporting is responsible for distributing those measurements to interested stakeholders. The separation
3632 between monitoring and reporting is made to include the possibility that data obtained through monitoring
3633 might not be used until an event impacting the ecosystem occurs or the measurement requires further
3634 processing to be useful. In the SOA framework, reporting is provided using services for requesting
3635 measurement reports. These reports may consist of raw measurement data, formatted collections of data,
3636 or the results of analysis performed on measurement data from collections of different managed aspects.
3637 Reporting is also used to support logging and auditing capabilities, where the reporting mechanisms
3638 create log or audit entries.

3639 **5.3.6 Management for Infrastructure**

3640 All of the properties, policies, interactions, resources, and management are only possible if a SOA
3641 ecosystem infrastructure provides support for managed capabilities. Each managed capability imposes
3642 different requirements on the capabilities supplied by the infrastructure in a SOA ecosystem and requires
3643 that those capabilities be usable as services or at the very least be interoperable.

3644 While not providing a full list of infrastructural elements of a SOA ecosystem, we list some examples here:

- 3645 • Registries and repositories for services, policies, and related descriptions and contracts
- 3646 • Synchronous and asynchronous communication channels for service interactions (e.g., network,
3647 e-mail, message routing with ability of mediating transport protocols, etc.)
- 3648 • Recovery capabilities
- 3649 • Security controls

3650 A SOA ecosystem infrastructure, enabling service management, should also support:

- 3651 • Management enforcement and control means
- 3652 • Monitoring and SLA validation controls
- 3653 • Testing and Reporting capabilities

3654 The combination of manageability properties, related capabilities and infrastructure elements constitutes
3655 a certain level of SOA management maturity. While several maturity models exist, this topic is out of the
3656 scope of the current document.

3657 5.3.7 Architectural Implication of SOA Management

3658 SOA Management is one of the fundamental elements of the SOA ecosystem; it impacts all aspects of a
3659 service life-cycle, service activities and actions, and a service usage. The key choices that must be made
3660 center on management means, methods and manageability properties:

- 3661 • Every resource of the SOA ecosystem and, particularly, services **MUST** provide manageability
3662 properties:
 - 3663 ○ The set of manageability properties **SHOULD** include as minimum such properties as life-
3664 cycle, combination, configuration, event monitoring, performance, quality of services, and
3665 policy manageability;
 - 3666 ○ Combinations of manageability properties **MAY** be used in different management
3667 methods and tools;
- 3668 • Manageability properties and applicable policies **SHOULD** be appropriately described in the
3669 service's description and contracts;
- 3670 • Management processes **SHOULD** operate (control, enforce and provide a feedback to the
3671 governance) via policies, agreements/contracts, and practices defined through governance;
- 3672 • Management functions and information **MAY** be realized as services and, thus, **MUST** be
3673 managed itself;
- 3674 • Management in the cases where sufficient guidance is unavailable or for which agreement
3675 between all stakeholders cannot be reached **MUST** be flexible and adaptable to handle
3676 unanticipated conditions without unnecessarily breaking trust relationships;
- 3677 • Management **SHOULD** engage a monitoring mechanism to enable manageability. Monitoring
3678 **MUST** include:
 - 3679 ○ Access mechanisms to collected SLA metrics;
 - 3680 ○ Assessment mechanisms to compare metrics against policies and contracts;
- 3681 • Results of monitoring and reporting **MUST** be made accessible to participants in different
3682 ownership domains.

3683 5.4 SOA Testing Model

3684 Testing for SOA combines the typical challenges of software testing and certification with the addition of
3685 accommodating the distributed nature and independence of the [resources](#), the greater access of a more
3686 unbounded consumer population, and the desired flexibility to create new solutions from existing
3687 components over which the solution developer has little if any control. The [purpose](#) of testing is to
3688 demonstrate a required level of reliability, correctness, and effectiveness that enable prospective
3689 consumers to have adequate confidence in using a service. Adequacy is defined by the consumer based
3690 on the consumer's needs and context of use. Absolute correctness and completeness cannot be proven
3691 by testing. For SOA, however, it is critical for the prospective consumer to know what testing has been
3692 performed; how it has been performed; and what were the results.

3693 5.4.1 Traditional Software Testing as Basis for SOA Testing

3694 SOA services are largely software artifacts and can leverage the body of experience that has evolved
3695 around software testing. **[IEEE 829]** specifies the basic set of software test documents while allowing
3696 flexibility for tailored use. Many testing frameworks are available but the SOA-RAF does not prescribe the
3697 use of any one in particular and choice will be driven by a framework that offers the right amount and
3698 level of testing. As such, IEEE-829 can provide guidance to SOA testing and a point of reference for
3699 additional test concerns introduced by a SOA approach.

3700 IEEE-829 covers test specification and test reporting through use of several document types, including
3701 test plans; test design, test case, and test procedure specifications; and documents to identify, log, and
3702 report on test occurrences and artifacts. In summary, IEEE-829 captures (1) what was tested, (2) how it
3703 was tested, e.g. the test procedure used, and (3) the results of the test. While the SOA-RAF does not
3704 require IEEE-829 artifacts, those with responsibilities for testing should consider how aspects of IEEE-
3705 829 apply.

3706 **5.4.1.1 Types of Testing**

3707 There are numerous aspects of testing that, in total, work to establish that an entity is (1) built as required
3708 per policies and related specifications prescribed by the entity's owner, and (2) delivers the functionality
3709 required by its intended consumers. This is often referred to as verification and validation.

3710 In Section 4.4, Policies are described that can be related to testing. These policies may prescribe but are
3711 not limited to the business processes to be followed. Policies may also prescribe the standards with which
3712 an implementation must comply, as well as the [qualifications](#) of and restrictions on the actors. In addition
3713 to the functional requirements prescribing what an entity does, there may also be non-functional
3714 performance and/or quality metrics that state how well the entity performs. The relation of these policies
3715 to SOA testing is discussed further below.

3716 The identification of policies is the purview of governance (section 5.1) and the assuring of compliance
3717 (including response to noncompliance) with policies is a matter for management (section 5.3).

3718 **5.4.1.2 Range of Test Conditions**

3719 Test conditions and expected responses are detailed in the test case specification. The test conditions
3720 should be designed to cover the areas for which the entity's response must be documented and may
3721 include:

- 3722 • nominal conditions;
- 3723 • boundaries and extremes of expected conditions;
- 3724 • breaking point where the entity has degraded below a certain level or has otherwise ceased
3725 effective functioning;
- 3726 • random conditions to investigate unidentified dependencies among combinations of conditions
- 3727 • error conditions to test error handling.

3728 The specification of how each of these conditions should be tested for SOA resources, including the
3729 infrastructure elements of the SOA ecosystem, is beyond the scope of this document but is an area that
3730 evolves along with operational SOA experience.

3731 **5.4.2 Testing and the SOA Ecosystem**

3732 Testing of SOA artifacts for use in the SOA ecosystem differs from traditional software testing for several
3733 reasons. These include a difference in what constitutes the consumer community and what constitutes
3734 the evolving environment that comprises the SOA ecosystem. In response, testing must include
3735 considerations for making a service testable throughout its lifetime.

3736 **5.4.2.1 Testing and the Consumer Communities**

3737 A highly touted benefit of SOA is to enable unanticipated consumers to make use of services for
3738 unanticipated purposes. Examples of this could include the consumer using a service for a result that
3739 was not considered the primary one by the provider or the service may be used in combination with other
3740 services in a scenario that is different from the one considered when designing for the initial target
3741 consumer community. It is unlikely that a new consumer will push the services back to anything
3742 resembling the initial test phase to test the new use, and thus additional paradigms for testing are
3743 necessary. The potential [responsibilities](#) related to such "consumer testing" are discussed further below.

3744 In addition to consumers who interact with a service to realize the described [real world effects](#), the
3745 developer community is also intended to be a consumer. In the SOA vision of reuse, the developer
3746 composes new solutions using existing services, where the existing services provide desired [real world](#)
3747 [effects](#) that are needed by the new solution. The composed solution must be tested for its intended
3748 functionality, and the component service may need particular attention if its use is different from its typical
3749 use as a separate offering. Note, the composition developer is not expected to own a private copy of a
3750 component service, and testing may be dependent on test interfaces provided by the component service.

3751 **5.4.2.2 Testing and the Evolving SOA Ecosystem**

3752 The distributed, unbounded nature of the SOA ecosystem makes it unlikely to have an isolated test
3753 environment that duplicates the operational environment. A traditional testing approach often makes use

3754 of a test system that is identical to the eventual operational system but isolated for testing. After testing is
3755 successfully completed, the tested entity would be migrated to the operational environment, or the test
3756 environment may be delivered as part of the system to become operational. This is not feasible for the
3757 SOA ecosystem as a whole.

3758 SOA services must be testable in the environment and under the conditions that can be encountered in
3759 the operational SOA ecosystem. As the ecosystem is in constant change, so some level of testing is
3760 continuous through the lifetime of the service, leveraging utility services used by the ecosystem
3761 infrastructure to monitor its own health and respond to situations that could lead to degraded
3762 performance. This implies the test resources must incorporate aspects of the SOA paradigm, and a
3763 category of services may be created to specifically support and enable effective monitoring and
3764 continuous testing for [resources](#) participating in the SOA ecosystem.

3765 While SOA within an enterprise may represent a more constrained and predictable operational
3766 environment, the composability and unanticipated use aspects are highly touted within the enterprise.
3767 The expanded perspective on testing may not be as demanding within an enterprise but fuller
3768 consideration of the ecosystem enables the enterprise to be more responsive should conditions change.

3769 **5.4.3 Elements of SOA Testing**

3770 IEEE-829 emphasizes identifying what is to be tested, how it is to be tested, and by whom the testing is to
3771 be done. This is equally applicable to SOA testing.

3772 **5.4.3.1 What is to be Tested**

3773 The focus of this discussion is the SOA service. It is recognized that the infrastructure components of
3774 any SOA environment are likely to also be SOA services and, as such, falls under the same testing
3775 guidance. Other resources that contribute to a SOA environment may not be SOA services, but are
3776 expected to satisfy the intent if not the letter of guidance presented here.

3777 The following discussion often focuses on a singular SOA service but it is implicit that any service may be
3778 a composite of other services. As such, testing the functionality of a composite service may effectively be
3779 testing an end-to-end business process that is being provided by the composite service. If new versions
3780 are available for the component services, appropriate end-to-end testing of the composite may be
3781 required in order to verify that the composite functionality is still adequately provided. The level of
3782 required testing of an updated composite service depends on policies of those providing the service,
3783 policies of those using the service, and mission criticality of those depending on the service results.

3784 The Service Description model (Figure 16) elaborates on described aspects of a service:

- 3785 • The service functionality and technical assumptions that underlie the functionality;
- 3786 • The policies that describe conditions of use;
- 3787 • The service interface that defines information exchange with the service;
- 3788 • Service reachability that identifies how and where message exchange is to occur; and
- 3789 • Metrics access for any [participant](#) to have information on how a service is performing.

3790 The aspects represent joint concerns of all the stakeholders, and service testing must provide adequate
3791 assurance that each of these aspects is operational as defined. In particular:

- 3792 • Service functionality is an early and ongoing focus of testing to ensure the service accurately
3793 reflects the described functionality and the described functionality accurately addresses the
3794 consumer needs.
- 3795 • Policies constraining service development, such as coding standards and best practices, require
3796 appropriate testing and auditing during development to ensure compliance. Policies that define
3797 conditions of use are initially tested during service development and are continuously monitored
3798 during the operational lifetime of the service.
- 3799 • At any point where the interface is modified or exposes a new [resource](#), the message exchange
3800 should be monitored both to ensure the message reaches its intended destination and it is parsed
3801 correctly once received.
- 3802 • The service interface is also tested when the service endpoint changes. Functioning of a service
3803 endpoint at one time does not guarantee it is functioning at another time, e.g. the server with the

3804 endpoint address may be down, making testing of service reachability a continual monitoring
3805 function through the life of the service's use of the endpoint.
3806 • Metrics are a key indicator for consumers to decide if a service is adequate for their needs. For
3807 instance, the average response time or the recent availability can be determining factors even if
3808 there are no rules or regulations promulgated through the governance process against which
3809 these metrics are assessed. Testing will ensure that the metrics access indicated in the service
3810 description is accurate.

3811 The individual test requirements highlight aspects of the service that testing must consider but testing
3812 must establish more than isolated behavior. The emphasis is the holistic results of interacting with the
3813 service in the SOA environment. Recall that the execution context is the set of agreements between a
3814 consumer and a provider that define the conditions under which service interaction occurs. Variations in
3815 the execution context require monitoring to ensure that different combinations of conditions perform
3816 together as desired. For example, if a new privacy policy takes additional [resources](#) to apply, this may
3817 affect quality of service and propagate other effects. These could not be tested during the original testing
3818 if the alternate policy did not exist at that time.

3819 **5.4.3.2 How Testing is to be Done**

3820 Testing should follow well-defined methodologies and, if possible, should reuse test artifacts that have
3821 proven generally useful for past testing. For example, IEEE-829 notes that test cases are separated from
3822 test designs to allow for use in more than one design and to allow for reuse in other situations. As with
3823 description of a service in the SOA ecosystem, description of testing artifacts enables awareness of the
3824 artifact and describes how the artifact may be accessed or used.

3825 As with traditional testing, the specific test procedures and test case inputs are important so the tests are
3826 unambiguously defined and entities can be retested in the future. Automated testing and regression
3827 testing may be more important in the SOA ecosystem in order to re-verify a service is still acceptable
3828 when incorporated in a new use. For example, if a new use requires the services to deal with input
3829 parameters outside the range of initial testing, the tests could be rerun with the new parameters. If the
3830 testing resources (e.g. services that support re-executing test cases) are available to consumers within
3831 the SOA ecosystem, the testing as designed by test professionals could be consumed through a service
3832 accessed by consumers, and their results could augment those already in place. This is discussed
3833 further in the next section.

3834 **5.4.3.3 Who Performs the Testing**

3835 As with any software, the first line of testing is unit testing done by software developers. It is likely that
3836 initial testing will be done by those developing the software but may also be done independently by other
3837 developers. For SOA development, unit testing is likely confined to a development sandbox isolated from
3838 the SOA ecosystem.

3839 SOA testing will differ from traditional software testing in that testing beyond the development sandbox
3840 must incorporate aspects of the SOA ecosystem, and those doing the testing must be familiar with both
3841 the characteristics and responses of the ecosystem and the tools, especially those available as services,
3842 to facilitate and standardize testing. Test professionals will know what level of assurance must be
3843 established as the exposure of the service to the ecosystem and ecosystem to the service increases
3844 towards operational status. These test professionals may be internal resources to an organization or may
3845 evolve as a separate discipline provided through external contracting.

3846 As noted above, it is unlikely that a complete duplicate of the SOA ecosystem will be available for isolated
3847 testing, and thus use of ecosystem [resources](#) will manifest as a transition process rather than a step
3848 change from a test environment to an operational one. This is especially true for new composite services
3849 that incorporate existing operational services to achieve the new functionality. The test professionals will
3850 need to understand the available resources and the ramifications of this transition.

3851 As with current software development, and following the work by test professionals, a select group of
3852 typical customers (commonly referred to as beta testers) will report on service response during typical
3853 intended use. This provides final validation, by the intended customers, of previously verified processes,
3854 requirements, and final implementation.

3855 In traditional software development, beta testing is the end of testing for a given version of the software.
3856 However, although the initial test phase can establish an appropriate level of confidence consistent with
3857 the designed use for the initial target consumer community, the operational service will exist in an
3858 evolving ecosystem, and later conditions of use may differ from those thought to be sufficient during the
3859 initial testing. Thus, operational monitoring becomes an extension of testing through the service lifetime.
3860 This continuous testing will attempt to ensure that a service does not consume an inordinate amount of
3861 ecosystem resources or display other behavior that degrades the ecosystem, but it will not undercover
3862 functional errors that may surface over time.

3863 As with any software, it is the responsibility of the consumers to consider the reasonableness of solutions
3864 in order to spot errors in either the software or the way the software is being used. This is especially
3865 important for consumers with unanticipated uses that may go beyond the original test conditions. It is
3866 unlikely the consumers will initiate a new round of formal testing unless the new use requires a
3867 significantly higher level of confidence in the service. Rather the consumer becomes a new extension to
3868 the testing regiment. Obvious testing would include a sanity check of results during the new use.
3869 However, if the details of legacy testing are associated with the service through the service description
3870 and if testing resources are available through automated testing services, then the new consumers can
3871 rerun and extend previous testing to include the extended test conditions. If the test results are
3872 acceptable, these can be added to the documentation of previous results and become the extended basis
3873 for future decisions by prospective consumers on the appropriateness of the service. If the results are not
3874 acceptable or in some way questionable, the responsible party for the service or testing professionals can
3875 be brought in to decide if remedial action is necessary.

3876 **5.4.3.4 How Testing Results are Reported**

3877 For any SOA service, an accurate reporting of the testing a service has undergone and the results of the
3878 testing is vital to consumers deciding whether a service is appropriate for intended use. Appropriateness
3879 may be defined by a consumer organization and require specific test regiments culminating in a
3880 certification; appropriateness could be established by accepting testing and certifications that have been
3881 conferred by others.

3882 The testing and certification information should be identified in the service description. Referring to the
3883 general description model of Figure 14, tests conducted by or under a request from the service owner
3884 (see [ownership](#) in section 3.2.4) would be captured under Annotations from Owners. Testing done by
3885 others (such as consumers with unanticipated uses) could be associated through Annotations from 3rd
3886 Parties.

3887 Consumer testing and the reporting of results raise additional issues. While stating who did the testing is
3888 mandatory, there may be formal requirements for authentication of the tester to ensure traceability of the
3889 testing claims. In some circumstances, persons or organizations would not be allowed to state testing
3890 claims unless the tester was an approved entity. In other cases, ensuring the tester had a valid email may
3891 be sufficient. In either case, it would be at the discretion of the potential consumer to decide what level of
3892 authentication was acceptable and which testers are considered authoritative in the context of their
3893 anticipated use.

3894 Finally, in a world of openly shared information, we would see an ever-expanding set of testing
3895 information as new uses and new consumers interact with a service. In reality, these new uses may
3896 represent proprietary processes or classified use that should only be available to authorized parties.
3897 Testing information, as with other elements of description, may require special access controls to ensure
3898 appropriate access and use.

3899 **5.4.4 Testing SOA Services**

3900 Testing of SOA services should be consistent with the SOA paradigm. In particular, testing resources and
3901 artifacts should be visible in support of service interaction between providers and consumers, where here
3902 the interaction is between the testing resource and the tester. In addition, the idea of opacity of the
3903 implementation should limit the details that need to be available for effective use of the test resources.

3904 Software testing is a gradual exercise going from micro inspection to testing macro effects. A typical
3905 testing process is likely to begin with the traditional code reviews. SOA considerations would account for

3906 the distributed nature of SOA, including issues of distributed security and best practices to ensure secure
3907 resources.

3908 Code review is likely followed by unit testing in a development sandbox isolated from the operational
3909 environment. The unit testing is done with full knowledge of the service internal structure and knowledge
3910 of resources representing underlying capabilities. Some aspects of testing may require that external
3911 dependencies be satisfied, and this is often done using substitutes that mimic some aspects of the
3912 performance of an operational service without committing to the [real world effects](#) that the operational
3913 service would produce. Unit testing includes tests of the service interface to ensure exchanged messages
3914 are as specified in the service description and the messages can be parsed and interpreted as intended.
3915 Unit testing also verifies intended functionality and that the software has dealt correctly with internal
3916 dependencies, such as access to other dedicated resources.

3917 After unit testing has demonstrated an adequate level of confidence in the service, the testing must
3918 transition from the tightly controlled environment of the development sandbox to an environment that
3919 more closely resembles the operational SOA ecosystem or, at a minimum, the intended enterprise. While
3920 sandbox testing will substitute for some interactions with the SOA environment, such as an interface to a
3921 security service without the security service functionality, the dynamic nature of SOA makes a full
3922 simulation infeasible to create or maintain. This is especially true when a new composite service makes
3923 use of operational services provided by others. Thus, at some point before testing is complete, the
3924 service will need to demonstrate its functionality by using resources and dealing with conditions that only
3925 exist in the full ecosystem or the intended enterprise. Some of these resources may still provide test
3926 interfaces but the interfaces will be accessible using the SOA environment and not just implemented for
3927 the sandbox.

3928 At this stage, the opacity of the service becomes important as the details of interacting with the service
3929 now rely on correct use of the service interface and not knowledge of the service internals. The workings
3930 of the service will only be observable through the [real world effects](#) realized through service interactions
3931 and external indications that conditions of use, such as user authentication, are satisfied. Monitoring the
3932 behavior of the service will depend on service interfaces that expose internal monitoring or provide
3933 required information to the SOA infrastructure monitoring function. The monitoring required to test a new
3934 service is likely to have significant overlap with the monitoring the SOA infrastructure includes to monitor
3935 its own health and to identify and isolate behavior outside of acceptable bounds. This is exactly what is
3936 needed as part of service testing, and it is reasonable to assume that the ecosystem transition includes
3937 use of operational monitoring rather than solely dedicated monitoring for each service being tested. Use
3938 of SOA monitoring resources during the explicit testing phase sets the stage for monitoring and a level of
3939 continual testing throughout the service lifetime.

3940 In summary, consider the example of a new composite service that combines the [real world effects](#) and
3941 complies with the conditions of use of five existing operational services. The developer of the composite
3942 service does not own any of the component services and has limited, if any, ability to get the distributed
3943 owners to do any customization. The developer also is limited by the principle of opacity to information
3944 comprising the service description, and does not know internal details of the component services. The
3945 developer of the composite service must use the component services as they exist as part of the SOA
3946 environment, including what is provided to support testing by new customers.

3947 **5.4.5 Architectural Implications for SOA Testing**

3948 The discussion of SOA Testing indicates numerous architectural implications that are to be considered for
3949 testing of resources and interactions within the SOA ecosystem:

- 3950 • SOA services **MUST** be testable in the environment and under the conditions that can be
3951 encountered in the operational SOA ecosystem.
- 3952 • The distributed, boundary-less nature of the SOA ecosystem makes it infeasible to create and
3953 maintain a single testing substitute of the entire ecosystem to support testing activities. Test
3954 protocols **MUST** recognize and accommodate changes to and activities within the ecosystem.
- 3955 • A standard suite of monitoring services **SHOULD** be defined, developed, and maintained. This
3956 **SHOULD** be done in a manner consistent with the evolving nature of the ecosystem.
- 3957 • Services **SHOULD** provide interfaces that support access in a test mode.

- 3958
- 3959
- 3960
- 3961
- 3962
- 3963
- 3964
- Testing resources **MUST** be described and their descriptions **MUST** be catalogued in a manner that enables their discovery and access.
 - Guidelines for testing and ecosystem access **MUST** be established and the ecosystem **MUST** be able to enforce those guidelines asserted as policies.
 - Services **SHOULD** be available to support automated testing and regression testing.
 - Services **SHOULD** be available to facilitate updating service description by authorized participants who has performed testing of a service.

3965 6 Conformance

3966 6.1 Conformance Targets

3967 This Reference Architecture Foundation is an abstract architectural description of Service Oriented
3968 Architecture. As such, tests of conformance to the RAF should be concerned primarily with adherence to
3969 principles rather than technical details such as prescribed syntax or coding conventions. Relevant
3970 principles are set out in the RAF through

- 3971 • the modeling of concepts and relationships (defining what it means to realize, own, and use SOA-
3972 based systems and have such systems participate in a SOA ecosystem); and
- 3973 • a series of Architectural Implications.

3974 The discussion of concepts and relationships that elaborate the SOA principles in each of the main
3975 sections above culminates in an 'Architectural Implications' section (sections 3.4, 4.1.4, 4.2.3, 4.3.6,
3976 4.4.3, 5.1.4, 5.2.5, 5.3.7, and 5.4.5), where these sections contain formal conformance requirements
3977 ("MAY", "MUST", "SHOULD") in accordance with [RFC 2119].

3978 In discussing conformance, we use the term **SOA-RAF Target Architecture** to identify the (typically
3979 concrete) architecture that may be considered as conforming to the abstract principles outlined in this
3980 document.

3981 **SOA-RAF Target Architecture**

3982 An architectural description of a system that is intended to be viewed as conforming to the
3983 SOA-RAF

3984 While we cannot guarantee interoperability between target architectures (or more specifically between
3985 applications and systems residing within the ecosystems of those target architectures), the likelihood of
3986 interoperability between target architectures is increased by conformance to this Reference Architecture
3987 Framework as it facilitates semantic engagement between the different ecosystems.

3988 6.2 Conformance and Architectural Implications

3989 The SOA-RAF focuses on concepts, and the relationships between them, that are needed to enable
3990 SOA-based systems to be realized, owned, and used. The Architectural Implications reflect specific
3991 elements that will be reflected in a more concrete architecture based on the SOA-RAF.

3992 Conformance can therefore be measured both in terms of how a SOA-RAF Target Architecture uses the
3993 concepts and models outlined in the SOA-RAF; and how the various Architectural Implications have been
3994 addressed.

3995 6.3 Conformance Summary

3996 Concepts described in the RAF **SHOULD** be expressed and used in the target architecture. If used, such
3997 expression **MUST** reflect the relationships identified within this document.

3998 Terminology within the target architecture **SHOULD** be identical to that in the RAF and the terms used
3999 refer to the same concepts; and any graph of concepts and relationships between them that *are* used
4000 **MUST** be consistent with the RAF.

4001 The SOA-RAF Target Architecture **MUST** take account of the Architectural Implications in the sections
4002 listed above.

4003 **Appendix A. Acknowledgements**

4004 The following individuals have participated in the work of the technical committee responsible for creation
4005 of this specification and are gratefully acknowledged:

4006 **Participants:**

4007 Chris Bashioum, MITRE Corporation
4008 Rex Brooks, Net Centric Operations Industry Consortium
4009 Peter F Brown, Individual Member
4010 Scott Came, Search Group Inc.
4011 Joseph Chiusano, Booz Allen Hamilton
4012 Robert Ellinger, Northrop Grumman Corporation
4013 David Ellis, Sandia National Laboratories
4014 Jeff A. Estefan, Jet Propulsion Laboratory
4015 Don Flinn, Individual Member
4016 Anil John, Johns Hopkins University
4017 Ken Laskey, MITRE Corporation
4018 Boris Lublinsky, Nokia Corporation
4019 Francis G. McCabe, Individual Member
4020 Christopher McDaniels, USSTRATCOM
4021 Tom Merkle, Lockheed Martin Corporation
4022 Jyoti Namjoshi, Patni Computer Systems Ltd.
4023 Duane Nickull, Adobe Inc.
4024 James Odell, Associate
4025 Michael Poulin, Individual Member
4026 Kevin T Smith, Novetta Solutions
4027 Michael Stiefel, Associate
4028 Danny Thornton, Northrop Grumman
4029 Timothy Vibbert, Lockheed Martin Corporation
4030 Robert Vitello, New York Dept. of Labor

4031 The committee would particularly like to underline the significant writing and conceptualization
4032 contributions made by Chris Bashioum, Rex Brooks, Peter Brown, Dave Ellis, Jeff Estefan, Ken Laskey,
4033 Boris Lublinsky, Frank McCabe, Michael Poulin, Kevin Smith and Danny Thornton

Appendix B. Index of Defined Terms

Action	39	Policy Conflict	75
Action Level Real World Effect.....	50	Policy Conflict Resolution	75
Actor	25	Policy Constraint	74
Authority	27	Policy Decision.....	74
Business functionality.....	29	Policy Enforcement	74
Business solution	37	Policy Framework	73
Capability.....	30	Policy Object.....	74
Communication	35	Policy Ontology	73
Composability.....	37	Policy Owner	74
Constitution	24	Policy Subject	74
Consumer.....	28	Presence	63
Contract.....	35	Private State	40
Delegate	25	Protocol.....	63
Endpoint	63	Provider.....	28
Governance.....	78	Real World Effect.....	30
Governance Framework.....	79	Regulation.....	80
Governance Processes.....	79	Requirement	30
Identifier.....	31	Resource.....	31
Joint Action.....	39	Responsibility.....	27
Leadership.....	79	Right.....	27
Logical Framework.....	73	Risk	33
Manageability	97	Rule.....	80
Manageability property.....	97	Security	86
Mediator	28	Semantic Engagement	36
Message Exchange.....	66	Service Contract	101
Need.....	30	Service Level Real World Effect	50
Non-Participant	25	Shared State	40
Obligation	28	SOA Ecosystem.....	21
Operations.....	66	SOA-based System	21
Owner.....	28	Social Structure.....	23
Ownership	32	Stakeholder.....	25
Ownership Boundary.....	32	State.....	40
Participant	25	Trust.....	33
Permission.....	27	Willingness.....	33
Policy.....	34		

4035
4036
4037
4038
4039
4040
4041
4042
4043
4044
4045
4046
4047
4048
4049
4050
4051
4052
4053
4054
4055
4056
4057
4058
4059
4060
4061
4062
4063
4064
4065
4066
4067
4068
4069
4070
4071
4072
4073
4074
4075
4076
4077
4078
4079
4080
4081
4082
4083

Appendix C. Relationship to other SOA Open Standards

Numerous efforts have been working in the space of defining standards for SOA and its applications. The OASIS SOA-RM Technical Committee and its SOA-RA Sub-Committee has established communications with several of these efforts in an attempt to coordinate and facilitate among the efforts. This appendix notes some of these efforts.

C.1 Navigating the SOA Open Standards Landscape Around Architecture

The white paper *Navigating the SOA Open Standards Landscape Around Architecture* issued jointly by OASIS, OMG, and The Open Group **[SOA NAV]** was written to help the SOA community at large navigate the myriad of overlapping technical products produced by these organizations with specific emphasis on the 'A' in SOA, i.e., Architecture.

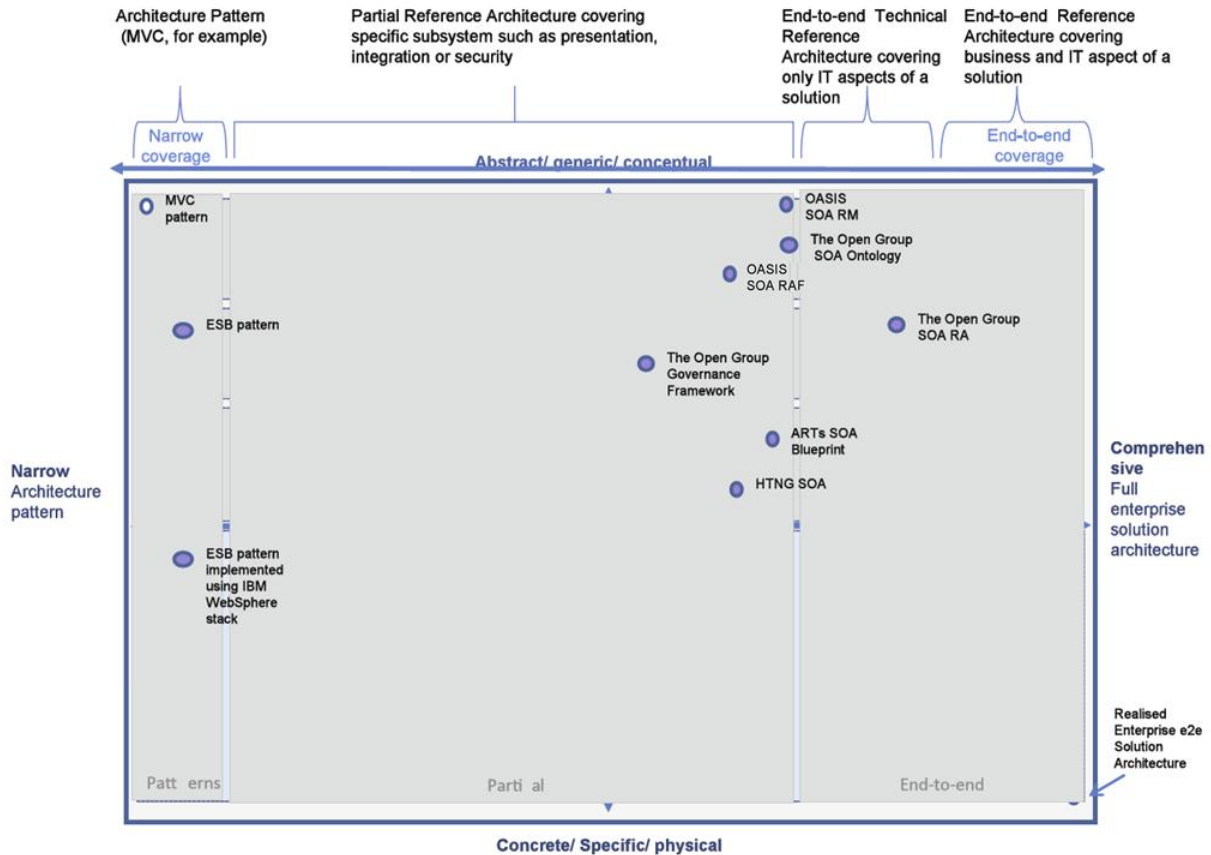
The white paper explains and positions standards for SOA reference models, ontologies, reference architectures, maturity models, modeling languages, and standards work on SOA governance. It outlines where the works are similar, highlights the strengths of each body of work, and touches on how the work can be used together in complementary ways. It is also meant as a guide to implementers for selecting those specifications most appropriate for their needs.

While the understanding of SOA and SOA Governance concepts provided by these works is similar, the evolving standards are written from different perspectives. Each specification supports a similar range of opportunity, but has provided different depths of detail for the perspectives on which they focus. Although the definitions and expressions may differ, there is agreement on the fundamental concepts of SOA and SOA Governance.

The following is a summary taken from **[SOA NAV]** of the positioning and guidance on the specifications:

- The OASIS Reference Model for SOA (SOA RM) is, by design, the most abstract of the specifications positioned. It is used for understanding core SOA concepts
- The Open Group SOA Ontology extends, refines, and formalizes some of the core concepts of the SOA RM. It is used for understanding core SOA concepts and facilitates a model-driven approach to SOA development.
- The OASIS Reference Architecture Foundation for SOA (this document) is an abstract, foundational reference architecture addressing a broader ecosystem viewpoint for building and interacting within the SOA paradigm. It is used for understanding different elements of SOA, the completeness of SOA architectures and implementations, and considerations for reaching across ownership boundaries where there is no single authoritative entity for SOA and SOA governance.
- The Open Group SOA Reference Architecture is a layered architecture from consumer and provider perspective with cross cutting concerns describing these architectural building blocks and principles that support the realizations of SOA. It is used for understanding the different elements of SOA, deployment of SOA in enterprise, basis for an industry or organizational reference architecture, implication of architectural decisions, and positioning of vendor products in a SOA context.
- The Open Group SOA Governance Framework is a governance domain reference model and method. It is for understanding SOA governance in organizations. The OASIS Reference Architecture for SOA Foundation contains an abstract discussion of governance principles as applied to SOA across boundaries
- The Open Group SOA Integration Maturity Model (OSIMM) is a means to assess an organization's maturity within a broad SOA spectrum and define a roadmap for incremental adoption. It is used for understanding the level of SOA maturity in an organization
- The Object Management Group SoaML Specification supports services modeling UML extensions. It can be seen as an instantiation of a subset of the Open Group RA used for representing SOA artifacts in UML.

4084 Fortunately, there is a great deal of agreement on the foundational core concepts across the many
 4085 independent specifications and standards for SOA. This can be best explained by broad and common
 4086 experience of implementers of SOA and its maturity in the marketplace. It also provides assurance that
 4087 investing in SOA-based business and IT transformation initiatives that incorporate and use these
 4088 specifications and standards helps to mitigate risks that might compromise a successful SOA solution.



4089
 4090 Figure 44 - SOA Reference Architecture Positioning (from 'Navigating the SOA Open
 4091 Standards Landscape Around Architecture', © OASIS, OMG, The Open Group)

4092 C.2 The Service-Aware Interoperability Framework: Canonical

4093 Readers of the RAF are strongly encouraged to review a document recently published by the Health
 4094 Level Seven (HL7) Architecture Board (ArB) entitled *The Service-Aware Interoperability Framework:
 4095 Canonical*. The document was developed over the past four years, and represents a substantive,
 4096 industry-specific effort (i.e. the large but vertical healthcare industry) to surface, define, and discuss in
 4097 detail various aspects of a number of critical success factors involved in implementing large-scale (i.e.
 4098 enterprises-level) architectures with a focus on achieving both intra- and inter-enterprise technical
 4099 interoperability irrespective of the particular exchange mechanism involved, e.g. service interface,
 4100 messages, or structure documents.

4101 In addition to providing an independent validation for the both the general focus as well as some of the
 4102 specifics of the RAF (especially those involving the importance of governance in achieving large-scale
 4103 interoperability), the HL7 document underscores several important aspects of the RAF including:

- 4104 1. A validation of one of the RAF's primary claims, i.e. the need to specifically focus on intra- and inter-
 4105 enterprise interoperability as a first-class citizen in any enterprise (or cross-enterprise) architecture
 4106 discussion irrespective of the particular choice of enterprise architecture approach, framework, or
 4107 implementation technology, e.g. TOGAF, Zachman, ODP, SOA, etc. In addition, the HL7 document
 4108 clearly articulates – as the RAF does as well – the difficulties involved in achieving that focus in such

- 4109 a manner that it can be manifest in operationally effective and manageable processes and
4110 deliverables.
- 4111 2. An agreement as to the critical importance of governance as the root of any successful effort to
4112 implement large-scale, cross-boundary interoperability aimed at achieving a collective shared mission
4113 or goal. In particular, both documents share the notion that 'technical-level' governance – e.g. service
4114 – or message-level technical interchange specifications – must itself be a manifestation of a higher-
4115 level, cross-jurisdictional agreement on desired goals, responsibilities, accountabilities, and
4116 deliverables.
- 4117 3. A validation of the importance of core SOA constructs as constructs useful in expressing many of the
4118 central aspects of interoperability irrespective of whether a particular interoperability scenario is
4119 actually 'realized' using SOA-compatible technologies. (NOTE: Although it might at first appear that
4120 the OASIS document is more 'service-focused' than the 'service-aware' document from HL7, there
4121 are considerably more similarities than differences in these slightly different foci secondary to the fact
4122 that both documents are intent on describing principles and framework concepts rather than delving
4123 into technical details. There are, however, certain instances where content of the OASIS document
4124 would be likely to find its analogue in SAIF Implementation Guides rather than in the SAIF Canonical
4125 Definition document.)
- 4126 4. The need for specific, explicit statements of those aspects of a given component that affects its ability
4127 to participate in a reliable, predictable manner in a variety of interoperability scenarios. In particular,
4128 component characteristics must be explicitly expressed in both design-time and run-time contexts as
4129 implicit assumptions are the root of most failures to achieve successfully cross-boundary
4130 interoperability irrespective of the chosen technical details of a particular interoperability instance.

4131 In summary, although the two documents are clearly not identical in their specifics, e.g. there are
4132 differences in the language used to name various concepts, constructs, and relationships; there are some
4133 differences in levels of abstraction regarding certain topics, etc.; and although the OASIS RAF is more
4134 directly focused on services as a final implementation architecture than the HL7 SAIF CD, the
4135 commonalities of purpose, content, and approach present in the two documents – documents which were
4136 developed by each organization without any knowledge of the others' work in what clearly are areas of
4137 common interest and concern – far outweighs their differences. As such, the HL7 ArB and the OASIS
4138 RAF Task Force have agreed to work together going forward to obtain the highest degree of alignment
4139 and harmonization possible between the two documents including the possible development of a joint
4140 document under the auspices of one of the ISO software engineering threads.

4141 The current version of the HL7 document – as well as all future versions – is available at:
4142 <http://www.hl7.org/permalink/?SAIFCDR1PUBLIC>

4143 **C.3 IEEE Reference Architecture**

4144 As the RAF has been finalized, a new initiative has appeared from the Institute of Electrical and
4145 Electronics Engineers (IEEE) to develop a SOA Reference Architecture. Encouragingly, the working
4146 group established decided not to start from scratch but instead take account of existing work. Its initial
4147 phase of work is currently ongoing (Summer 2012) and is concentrating on assessing both the current
4148 RAF and The Open Group's SOA Reference Architecture. The desire at this stage is to endorse these
4149 two works rather than to create a new one.

4150 **C.4 RM-ODP**

4151 The Reference Model for Open Distributed Processing (the RM-ODP) is an international standard
4152 developed by the ISO and ITU-T standardization organizations [**ISO/IEC 10746**]. It provides a set of
4153 concepts and structuring rules for describing and building open distributed systems, structured in terms of
4154 five viewpoints, representing concerns of different stakeholders.

4155 From an architectural point of view, there is no significant difference between service-oriented
4156 architectures (SOA) and the architectural framework defined in ODP. Some argue that current service-
4157 oriented approaches can be understood as a subset of the more general ODP approach [LININGTON].
4158 Many of the concepts and principles in the RAF and the RM-ODP are indeed closely aligned.

4159 In common with the RAF, RM-ODP uses the Viewpoint construct of **[ISO/IEC 42010]** in order to articulate
4160 the work, context and concepts.

4161 There is a danger of over-simplifying the comparison and losing some of the important mapping between
4162 the concepts in the two works but a high-level comparison follows.

4163 The **enterprise viewpoint** and the **information viewpoint** share many aspects in common with the
4164 RAF's SOA Ecosystem view and its associated models and are mainly concerned with: understanding,
4165 defining and modeling organizational context in which a distributed system is to be built and operated;
4166 defines how sets of participants should behave in order to achieve specific objectives; roles played;
4167 processes and interactions involved; enterprise policies (obligations, permissions, prohibitions,
4168 authorizations) that constrain behavior in different roles; and descriptions of behavior expressing
4169 functionality or capability provided by one party to others who can use the service to satisfy their own
4170 business needs, resulting in an added value to them.

4171 The **computational viewpoint** maps closely to the RAF Service Model and is concerned with describing
4172 basic functionality of the processes and applications supporting enterprise activities. They are both
4173 concerned with interactions at interfaces between and across organizational or ownership boundaries.

4174 The RM-ODP standard also provides a well-defined **conformance framework**, providing links between
4175 specifications and implementations and thus supporting testing and which corresponds to the RAF's
4176 Architectural Implications sections.

4177 The ODP viewpoint languages are defined in an abstract way and can be supported by several notations.
4178 The use of UML notation in expressing ODP viewpoint languages is defined in a separate ISO standard,
4179 *Use of UML for ODP system specification* ('UML4ODP' for short) **[ISO/IEC IS 19793]**.

4180