



# Reference Model for Service Oriented Architecture

## Committee Draft 1.0, 31 January 2006

### Document identifier:

wd-soa-rm-cd1

### Location:

[http://www.oasis-open.org/committees/tc\\_home.php?wg\\_abbrev=soa-rm](http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=soa-rm)

### Editors:

C. Matthew MacKenzie, Adobe Systems Incorporated, [mattm@adobe.com](mailto:mattm@adobe.com)  
Ken Laskey, MITRE Corporation, [klaskey@mitre.org](mailto:klaskey@mitre.org)  
Francis McCabe, Fujitsu Limited, [frank.mccabe@us.fujitsu.com](mailto:frank.mccabe@us.fujitsu.com)  
Peter Brown, Individual, [peter@justbrown.net](mailto:peter@justbrown.net)  
Rebekah Metz, Booz Allen Hamilton, [metz\\_rebekah@bah.com](mailto:metz_rebekah@bah.com)

### Abstract:

This Reference Model for Service Oriented Architecture is an abstract framework for understanding significant entities and relationships between them within a service-oriented environment, and for the development of consistent standards or specifications supporting that environment. It is based on unifying concepts of SOA and may be used by architects developing specific service oriented architectures or in training and explaining SOA. A reference model is not directly tied to any standards, technologies or other concrete implementation details. It does seek to provide a common semantics that can be used unambiguously across and between different implementations.

While service-orientation may be a popular concept found in a broad variety of applications, this reference model focuses on the field of software architecture. The concepts and relationships described may apply to other "service" environments; however, this specification makes no attempt to completely account for use outside of the software domain.

### Status:

This document is updated periodically on no particular schedule. Send comments to the editor(s).

Committee members should send comments on this specification to the [soa-rm@lists.oasis-open.org](mailto:soa-rm@lists.oasis-open.org) list. Others should visit the SOA-RM TC home page at [http://www.oasis-open.org/committees/tc\\_home.php?wg\\_abbrev=soa-rm](http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=soa-rm), and record comments using the web form available there.

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 SOA-RM TC web page at:

[http://www.oasis-open.org/committees/tc\\_home.php?wg\\_abbrev=soa-rm](http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=soa-rm)

The errata page for this specification is at:

[http://www.oasis-open.org/committees/tc\\_home.php?wg\\_abbrev=soa-rm](http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=soa-rm).

---

## 42 Table of Contents

43	1	Introduction.....	3
44	1.1	What is a reference model.....	3
45	1.2	A Reference Model for Service Oriented Architectures.....	3
46	1.3	Audience .....	4
47	1.4	Guide to using the reference model.....	5
48	1.5	Notational Conventions .....	5
49	1.6	Relationships to Other Standards .....	5
50	2	Service Oriented Architecture .....	6
51	2.1	What is Service Oriented Architecture?.....	6
52	2.1.1	A worked Service Oriented Architecture example.....	7
53	2.2	How is Service Oriented Architecture different? .....	8
54	2.3	The Benefits of Service Oriented Architecture .....	8
55	3	The Reference Model.....	9
56	3.1	Service.....	9
57	3.2	Dynamics of Services.....	10
58	3.2.1	Visibility .....	10
59	3.2.2	Interacting with services .....	12
60	3.2.3	Real World Effect .....	15
61	3.3	About services .....	16
62	3.3.1	Service description .....	17
63	3.3.2	Policies and Contracts.....	19
64	3.3.3	Execution context.....	21
65	4	Conformance Guidelines.....	22
66	5	References .....	23
67	5.1	Normative .....	23
68	5.2	Non-Normative.....	23
69		Appendix A. Glossary.....	24
70		Appendix B. Acknowledgments.....	27
71		Appendix C. Notices.....	28
72			

---

## 73 1 Introduction

74 The notion of Service Oriented Architecture (SOA) has received significant attention within  
75 the software design and development community. The result of this attention is the  
76 proliferation of many conflicting definitions of SOA. Whereas SOA architectural patterns (or  
77 *reference architectures*) may be developed to explain and underpin a generic design  
78 template supporting a specific SOA, a **reference model** is intended to provide an even  
79 higher level of commonality, with definitions that should apply to *all* SOA.

### 80 1.1 What is a reference model

81 A reference model is an abstract **framework** for understanding significant relationships  
82 among the entities of some environment. It enables the development of specific  
83 architectures using consistent standards or specifications supporting that environment. A  
84 reference model consists of a minimal set of unifying concepts, axioms and relationships  
85 within a particular problem domain, and is independent of specific standards, technologies,  
86 implementations, or other concrete details.

87 As an illustration of the relationship between a reference model and the architectures that can  
88 derive from such a model, consider what might be involved in modeling what is important about  
89 residential housing. In the context of a reference model, we know that concepts such as eating  
90 areas, hygiene areas and sleeping areas are all important in understanding what goes into a  
91 house. There are relationships between these concepts, and constraints on how they are  
92 implemented. For example, there may be physical separation between eating areas and hygiene  
93 areas.

94 The role of a **reference architecture** for housing would be to identify abstract solutions to the  
95 problems of providing housing. A general pattern for housing, one that addresses the needs of its  
96 occupants in the sense of, say, noting that there are bedrooms, kitchens, hallways, and so on is a  
97 good basis for an abstract reference architecture. The concept of eating area is a reference  
98 model concept, a kitchen is a realization of eating area in the context of the reference  
99 architecture.

100 There may be more than one reference architecture that addresses how to design housing; for  
101 example, there may be a reference architecture to address the requirements for developing  
102 housing solutions in large apartment complexes, another to address suburban single family  
103 houses, and another for space stations. In the context of high density housing, there may not be  
104 a separate kitchen but rather a shared cooking space or even a communal kitchen used by many  
105 families.

106 An actual – or concrete – architecture would introduce additional elements. It would incorporate  
107 particular architectural styles, particular arrangements of windows, construction materials to be  
108 used and so on. A blueprint of a particular house represents an instantiation of an architecture as  
109 it applies to a proposed or actually constructed dwelling.

110 The reference model for housing is, therefore, at least three levels of abstraction away from a  
111 physical entity that can be lived in. The purpose of a reference model is to provide a common  
112 conceptual framework that can be used consistently across and between different  
113 implementations and is of particular use in modeling specific solutions.

### 114 1.2 A Reference Model for Service Oriented Architectures

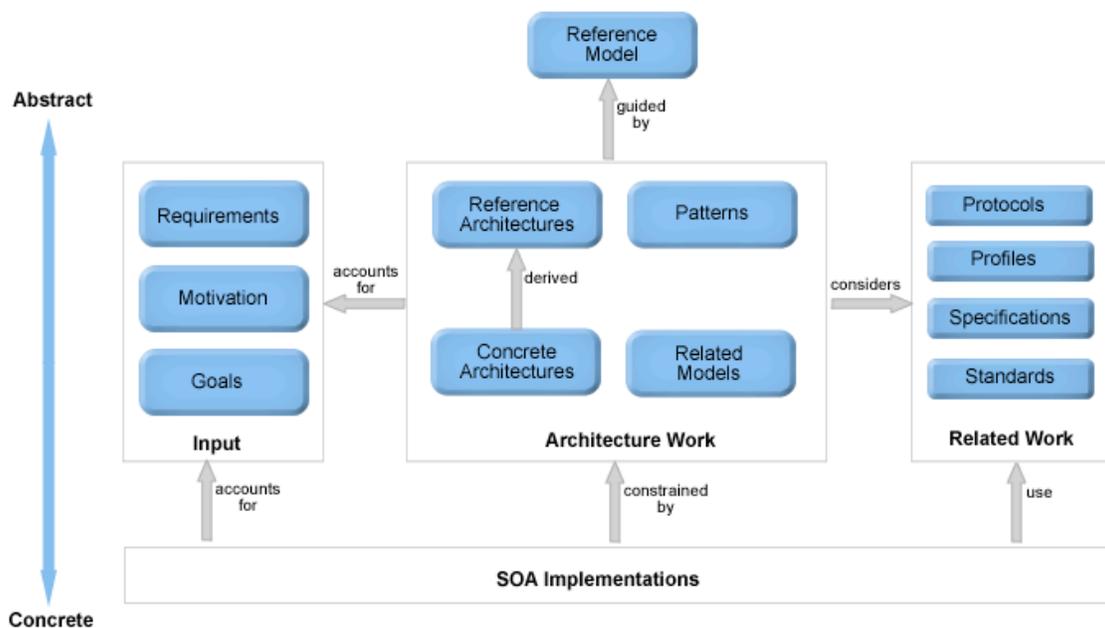
115 The goal of this reference model is to define the essence of service oriented architecture, and  
116 emerge with a vocabulary and a common understanding of SOA. It provides a normative  
117 reference that remains relevant for SOA as an abstract and powerful model, irrespective of the  
118 various and inevitable technology evolutions that will influence SOA deployment.

119 Figure 1 shows how a reference model for SOA relates to other distributed systems  
 120 architectural inputs. The concepts and relationships defined by the reference model are  
 121 intended to be the basis for describing references architectures and patterns that will define  
 122 more specific categories of SOA designs. Concrete architectures arise from a combination  
 123 of reference architectures, architectural patterns and additional requirements, including  
 124 those imposed by technology environments.

125 Architecture is not done in isolation but must account for the goals, motivation, and  
 126 requirements that define the actual problems being addressed. While reference  
 127 architectures can form the basis of classes of solutions, concrete architectures will define  
 128 specific solution approaches.

129 Architecture is often developed in the context of a pre-defined environment, such as the  
 130 protocols, profiles, specifications, and standards that are pertinent.

131 SOA implementations combine all of these elements, from the more generic architectural  
 132 principles and infrastructure to the specifics that define the current needs, and represent  
 133 specific implementations that will be built and used in an operational environment.



134  
 135 *Figure 1 How the Reference Model relates to other work*

### 136 1.3 Audience

137 The intended audiences of this document include non-exhaustively:

- 138 • Architects and developers designing, identifying or developing a system based on the
- 139 service-oriented paradigm.
- 140 • Standards architects and analysts developing specifications that rely on service oriented
- 141 architecture concepts.
- 142 • Decision makers seeking a "consistent and common" understanding of service oriented
- 143 architectures.
- 144 • Users who need a better understanding of the concepts and benefits of service oriented
- 145 architecture.

## 146 **1.4 Guide to using the reference model**

147 New readers are encouraged to read this reference model in its entirety. Concepts are presented  
148 in an order that the authors hope promote rapid understanding.

149 This section introduces the conventions, defines the audience and sets the stage for the rest of  
150 the document. Non-technical readers are encouraged to read this information as it provides  
151 background material necessary to understand the nature and usage of reference models.

152 Section 2 introduces the concept of SOA and identifies some of the ways that it differs from  
153 previous paradigms for distributed systems. Section 2 offers guidance on the basic principles of  
154 service oriented architecture. This can be used by non-technical readers to gain an explicit  
155 understanding of the core principles of SOA and by architects as guidance for developing specific  
156 service oriented architectures.

157 Section 3 introduces the Reference Model for SOA. In any framework as rich as SOA, it is difficult  
158 to avoid a significant amount of cross referencing between concepts. This makes presentation of  
159 the material subject to a certain amount of arbitrariness. We resolve this by introducing the  
160 concept of **service** itself, then we introduce concepts that relate to the dynamic aspects of service  
161 and finally we introduce those concepts that refer to the meta-level aspects of services such as  
162 service description and policies as they apply to services.

163 Section 4 addresses compliance with this reference model.

164 The glossary provides definitions of terms that are relied upon within the reference model  
165 specification but do not necessarily form part of the specification itself. Terms that are defined in  
166 the glossary are marked in **bold** at their first occurrence in the document.

167 Note that while the concepts and relationships described in this reference model may apply to  
168 other "service" environments, the definitions and descriptions contained herein focus on the field  
169 of software architecture and make no attempt to completely account for use outside of the  
170 software domain. Examples included in this document that are taken from other domains are  
171 used strictly for illustrative purposes.

## 172 **1.5 Notational Conventions**

173 The key words MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD, SHOULD NOT,  
174 RECOMMENDED, MAY, and OPTIONAL in this document are to be interpreted as described in  
175 **[RFC2119]**.

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

## 177 **1.6 Relationships to Other Standards**

178 Due to its nature, this reference model may have an implied relationship with any group that:

- 179 • Considers its work "service oriented";
- 180 • Makes (publicly) an adoption statement to use the Reference Model for SOA as a base or  
181 inspiration for their work; and
- 182 • Standards or technologies that claim to be service oriented.

183 The reference model does not endorse any particular service-oriented architecture, or attest to  
184 the validity of third party reference model conformance claims.

185

## 2 Service Oriented Architecture

186

### 2.1 What is Service Oriented Architecture?

187

**Service Oriented Architecture** (SOA) is a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains.

188

189

In general, entities (people and organizations) create **capabilities** to solve or support a solution for the problems they face in the course of their business. It is natural to think of one person's needs being met by capabilities offered by someone else; or, in the world of distributed computing, one computer agent's requirements being met by a computer agent belonging to a different owner.

190

191

192

193

194

There is not necessarily a one-to-one correlation between needs and capabilities; the granularity of needs and capabilities vary from fundamental to complex, and any given need may require the combining of numerous capabilities while any single capability may address more than one need. The perceived value of SOA is that it provides a powerful framework for matching needs and capabilities and for combining capabilities to address those needs.

195

196

197

198

199

Visibility, interaction, and effect are key concepts for describing the SOA paradigm. **Visibility** refers to the capacity for those with needs and those with capabilities to be able to see each other. This is typically done by providing descriptions for such aspects as functions and technical requirements, related constraints and policies, and mechanisms for access or response. The descriptions need to be in a form (or can be transformed to a form) in which their syntax and **semantics** are widely accessible and understandable.

200

201

202

203

204

205

Whereas visibility introduces the possibilities for matching needs to capabilities (and vice versa), **interaction** is the activity of using a capability. Typically mediated by the exchange of messages, an interaction proceeds through a series of information exchanges and invoked actions. There are many facets of interaction; but they are all grounded in a particular **execution context** – the set of technical and business elements that form a path between those with needs and those with capabilities. This permits service providers and consumers to interact and provides a decision point for any policies and contracts that may be in force.

206

207

208

209

210

211

212

The purpose of using a capability is to realize one or more **real world effects**. At its core, an interaction is “an act” as opposed to “an object” and the result of an interaction is an effect (or a set/series of effects). We are careful to distinguish between *public* actions and *private* actions; private actions are inherently unknowable by other parties. On the other hand, public actions result in changes to the *state* that is shared at least between those involved in the current execution context and possibly shared by others. Real world effects are, then, couched in terms of changes to this shared state.

213

214

215

216

217

218

219

The expected real world effects form an important part of the decision on whether a given capability matches similarly described needs. At the interaction stage, the description of real world effects establishes the expectations of those using the capability. A cornerstone of SOA is that we can use capabilities without needing to know all the details.

220

221

222

223

To this point, this description of SOA has yet to mention what is usually considered the central concept: the **service**. The noun “service” is defined in dictionaries as “The performance of work (a function) by one for another.” However, service, as the term is generally understood, also combines the following related ideas:

224

225

226

227

- The capability to perform work for another

228

- The specification of the work offered for another

229

- The offer to perform work for another

230 These concepts emphasize a distinction between a capability and the ability to bring that  
231 capability to bear. While both needs and capabilities exist independently of SOA, **in SOA,**  
232 **services are the mechanism by which needs and capabilities are brought together.**

233 SOA is a means of organizing solutions that promotes reuse, growth and interoperability. It is not  
234 itself a solution to domain problems but rather an organizing and delivery paradigm that enables  
235 one to get more value from use both of capabilities which are locally “owned” and those under the  
236 control of others. It also enables one to express solutions in a way that makes it easier to modify  
237 or evolve the identified solution or to try alternate solutions. SOA does not provide any domain  
238 elements of a solution that do not exist without SOA.

239 The concepts of visibility, interaction, and effect apply directly to services in the same manner as  
240 these were described for the general SOA paradigm. Visibility is promoted through the **service**  
241 **description** which contains the information necessary to interact with the service and describes  
242 this in such terms as the service inputs, outputs, and associated semantics. The service  
243 description also conveys what is accomplished when the service is invoked and the conditions for  
244 using the service.

245 In general, entities (people and organizations) offer capabilities and act as **service providers**.  
246 Those with needs who make use of services are referred to as **service consumers**. The service  
247 description allows prospective consumers to decide if the service is suitable for their current  
248 needs and establishes whether a consumer satisfies any requirements of the service provider.

249 (Note, service providers and service consumers are sometimes referred to jointly as **service**  
250 **participants**.)

251 In most discussions of SOA, the terms “loose coupling” and “coarse-grained” are commonly  
252 applied as SOA concepts, but these terms have intentionally not been used in the current  
253 discussion because they are subjective trade-offs and without useful metrics. In terms of needs  
254 and capabilities, granularity and coarseness are usually relative to detail for the level of the  
255 problem being addressed, e.g. one that is more strategic vs. one down to the algorithm level, and  
256 defining the optimum level is not amenable to counting the number of interfaces or the number or  
257 types of information exchanges connected to an interface.

258 Note that although SOA is commonly implemented using Web services, services can be made  
259 visible, support interaction, and generate effects through other implementation strategies. Web  
260 service-based architectures and technologies are specific and concrete. While the concepts in the  
261 Reference Model apply to such systems, they are too solution specific to be part of a general  
262 reference model.

### 263 **2.1.1 A worked Service Oriented Architecture example**

264 An electric utility has the capacity to generate and distribute electricity (the underlying capability).  
265 The wiring from the electric company’s distribution grid (the service) provides the means to supply  
266 electricity to support typical usage for a residential consumer’s house (service functionality), and  
267 a consumer accesses electricity generated (the output of invoking the service) via a wall outlet  
268 (service interface). In order to use the electricity, a consumer needs to understand what type of  
269 plug to use, what is the voltage of the supply, and possible limits to the load; the utility presumes  
270 that the customer will only connect devices that are compatible with the voltage provided and load  
271 supported; and the consumer in turn assumes that compatible consumer devices can be  
272 connected without damage or harm (service technical assumptions).

273 A residential or business user will need to open an account with the utility in order to use the  
274 supply (service constraint) and the utility will meter usage and expects the consumer to pay for  
275 use at the rate prescribed (service policy). When the consumer and utility agree on constraints  
276 and polices (service contract), the consumer can receive electricity using the service as long as  
277 the electricity distribution grid and house connection remain intact (e.g. a storm knocking down  
278 power lines would disrupt distribution) and the consumer can have payment sent (e.g. a check by  
279 mail or electronic funds transfer) to the utility (reachability).

280 Another person (for example, a visitor to someone else's house) may use a contracted supply  
281 without any relationship with the utility or any requirement to also satisfy the initial service  
282 constraint (i.e. reachability only requires intact electricity distribution) but would nonetheless be  
283 expected to be compatible with the service interface.

284 In certain situations (for example, excessive demand), a utility may limit supply or institute rolling  
285 blackouts (service policy). A consumer might lodge a formal complaint if this occurred frequently  
286 (consumer's implied policy).

287 If the utility required every device to be hardwired to its equipment, the underlying capability  
288 would still be there but this would be a very different service and have a very different service  
289 interface.

## 290 **2.2 How is Service Oriented Architecture different?**

291 Unlike Object Oriented Programming paradigms, where the focus is on packaging data with  
292 operations, the central focus of Service Oriented Architecture is the task or business function –  
293 getting something done. This is a more viable basis for large scale systems because it is a better  
294 fit to the way human activity itself is managed – by delegation.

295 How does this paradigm of SOA differ from other approaches to organizing and understanding  
296 Information Technology assets? Essentially, there are two areas in which it differs both of which  
297 shape the framework of concepts that underlie distributed systems.

298 First, SOA reflects the reality that ownership boundaries are a motivating consideration in the  
299 architecture and design of systems. This recognition is evident in the core concepts of visibility,  
300 interaction and effect. However, SOA does not itself address all the concepts associated with  
301 ownership, ownership domains and actions communicated between legal peers. To fully account  
302 for concepts such as trust, business transactions, authority, delegation and so on – additional  
303 conceptual frameworks and architectural elements are required. Within the context of SOA,  
304 these are likely to be represented and referenced within **service descriptions** and **service**  
305 **interfaces**.

306 Second, SOA applies the lessons learned from commerce to the organization of IT assets to  
307 facilitate the matching of capabilities and needs. That two or more entities come together within  
308 the context of a single interaction implies the exchange of some type of value. This is the same  
309 fundamental basis as trade itself, and suggests that as SOAs evolve away from interactions  
310 defined in a point-to-point manner to a marketplace of services; the technology and concepts can  
311 scale as successfully as the commercial marketplace.

## 312 **2.3 The Benefits of Service Oriented Architecture**

313 The main drivers for SOA-based architectures are to facilitate the manageable growth of large-  
314 scale enterprise systems, to facilitate Internet-scale provisioning and use of services and to  
315 reduce costs in organization to organization cooperation.

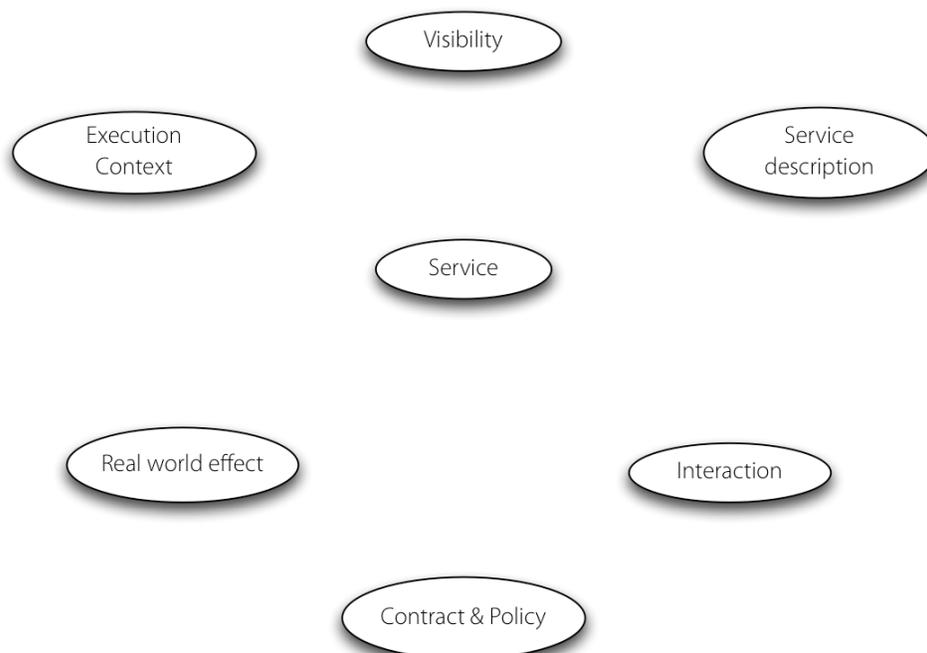
316 The value of SOA is that it provides a simple scalable paradigm for organizing large networks of  
317 systems that require interoperability to realize the value inherent in the individual components.  
318 Indeed, SOA is scalable because it makes the fewest possible assumptions about the network  
319 and also minimizes any trust assumptions that are often implicitly made in smaller scale systems.

320 An architect using SOA principles is better equipped, therefore, to develop systems that are  
321 scalable, evolvable and manageable. It should be easier to decide how to integrate functionality  
322 across ownership boundaries. For example, a large company that acquires a smaller company  
323 must determine how to integrate the acquired IT infrastructure into its overall IT portfolio.

324 Through this inherent ability to scale and evolve, SOA enables an IT portfolio which is also  
325 adaptable to the varied needs of a specific problem domain or process architecture. The  
326 infrastructure SOA encourages is also more agile and responsive than one built on an  
327 exponential number of pair-wise interfaces. Therefore, SOA can also provide a solid foundation  
328 for business agility and adaptability.

### 329 **3 The Reference Model**

330 Figure 2 illustrates the principal concepts this reference model defines. The relationships between  
331 them are developed as each concept is defined in turn.



332  
333 *Figure 2 Principal concepts in the Reference Model*

#### 334 **3.1 Service**

335 A **service** is a mechanism to enable access to a set of one or more capabilities, where the  
336 access is provided using a prescribed interface and is exercised consistent with constraints and  
337 policies as specified by the service description. A service is provided by one entity – the **service**  
338 **provider** – for use by others, but the eventual consumers of the service may not be known to the  
339 service provider and may demonstrate uses of the service beyond the scope originally conceived  
340 by the provider.

341 A service is accessed by means of a service interface (see Section 3.3.1.4), where the interface  
342 comprises the specifics of how to access the underlying capabilities. There are no constraints on  
343 what constitutes the underlying capability or how access is implemented by the service provider.  
344 Thus, the service could carry out its described functionality through one or more automated  
345 and/or manual processes that themselves could invoke other available services.

346 A service is opaque in that its implementation is typically hidden from the **service consumer**  
347 except for (1) the information and behavior models exposed through the service interface and (2)  
348 the information required by service consumers to determine whether a given service is  
349 appropriate for their needs.

350 The consequence of invoking a service is a realization of one or more real world effects (see  
351 Section 3.2.3). These effects may include:

- 352  
353 1. information returned in response to a request for that information,

- 354 2. a change to the shared state of defined entities, or  
355 3. some combination of (1) and (2).

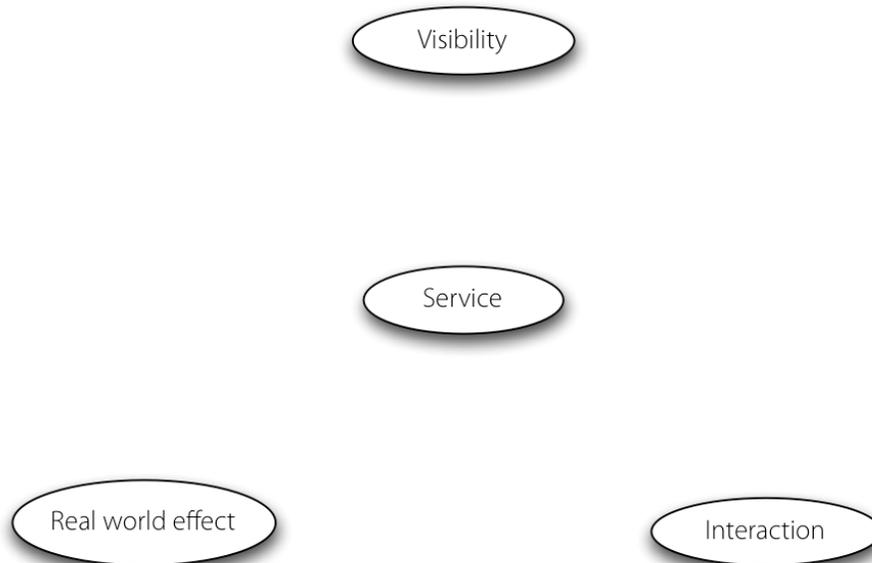
356

357 Note, the service consumer in (1) does not typically know how the information is generated, e.g.  
358 whether it is extracted from a database or generated dynamically; in (2), it does not typically know  
359 how the state change is effected.

360 The service concept above emphasizes a distinction between a capability that represents some  
361 functionality created to address a need and the point of access to bring that capability to bear in  
362 the context of SOA. It is assumed that capabilities exist outside of SOA. In actual use,  
363 maintaining this distinction may not be critical (i.e. the service may be talked about in terms of  
364 being the capability) but the separation is pertinent in terms of a clear expression of the nature of  
365 SOA and the value it provides.

## 366 3.2 Dynamics of Services

367 From a dynamic perspective, there are three fundamental concepts that are important in  
368 understanding what is involved in interacting with services: the visibility between service providers  
369 and consumers, the interaction between them, and the real world effect of interacting with a  
370 service.

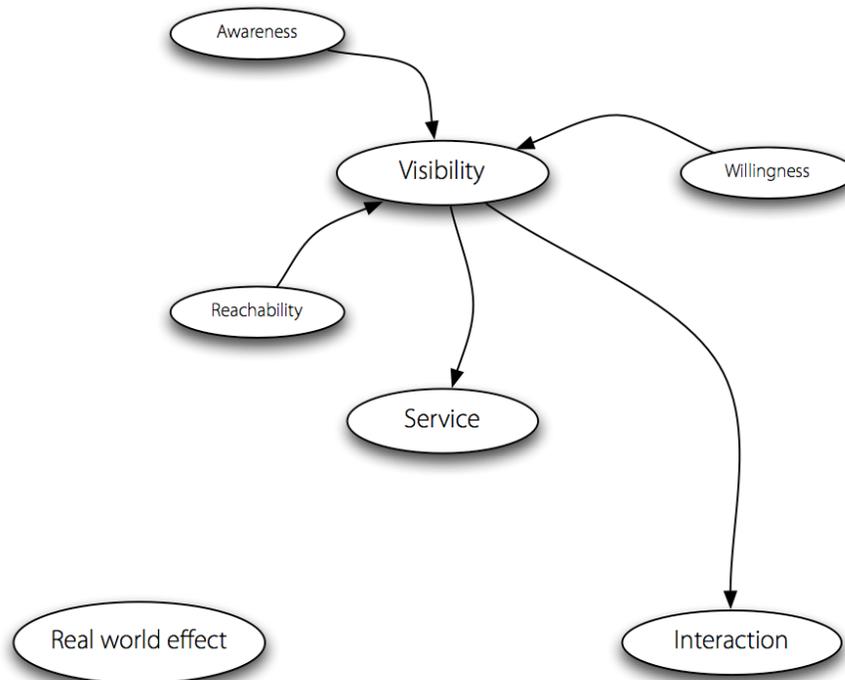


371

372 *Figure 3 Concepts around the dynamics of service*

### 373 3.2.1 Visibility

374 For a service provider and consumer to interact with each other they have to be able to 'see' each  
375 other. This is, in fact, true for any consumer/provider relationship – including in an application  
376 program where one program calls another: without the proper libraries being present the function  
377 call cannot complete. In the case of SOA, visibility needs to be emphasized because it is not  
378 necessarily obvious how service participants *can* see each other.



379

380 *Figure 4 Concepts around Visibility*

381 Visibility is the relationship between service consumers and providers that is satisfied when they  
 382 are able to interact with each other. Preconditions to visibility are **awareness**, **willingness** and  
 383 **reachability**. The initiator in a service interaction **MUST** be aware of the other parties, the  
 384 participants **MUST** be predisposed to interaction, and the participants **MUST** be able to interact.

### 385 3.2.1.1 Awareness

386 Both the service provider and the service consumer **MUST** have information that would lead them  
 387 to know of the other's existence. Technically, the prime requirement is that the *initiator* of a  
 388 service interaction has knowledge of the responder. The fact of a successful initiation is often  
 389 sufficient to inform the responder of the other's existence.

390 Awareness is a state whereby one party has knowledge of the existence of the other party.  
 391 Awareness does not imply willingness or reachability. Awareness of service offerings is often  
 392 effected by various *discovery* mechanisms. For a service consumer to discover a service, the  
 393 service provider must be capable of making details of the service (notably service description and  
 394 policies) available to potential consumers; and consumers must be capable of becoming aware of  
 395 that information. Conversely, the service provider may want to discover likely consumers and  
 396 would need to become aware of the consumer's description. In the following, we will discuss  
 397 awareness in terms of service visibility but the concepts are equally valid for consumer visibility.

398 Service awareness requires that the **service description** and **policy** – or at least a suitable  
 399 subset thereof – be available in such a manner and form that, directly or indirectly, a potential  
 400 consumer is aware of the existence and capabilities of the service. The extent to which the  
 401 description is “pushed” by the service provider, “pulled” by a potential consumer, subject to a  
 402 probe or another method, will depend on many factors.

403 For example, a service provider may advertise and promote their service by either including it in a  
 404 service directory or broadcasting it to all consumers; potential consumers may broadcast their  
 405 particular service needs in the hope that a suitable service responds with a proposal or **offer**, or a  
 406 service consumer might also probe an entire network to determine if suitable services exist.  
 407 When the demand for a service is higher than the supply, then, by advertising their needs,

408 potential consumers are likely to be more effective than service providers advertising offered  
409 services.

410 One way or another, the potential consumer must acquire sufficient descriptions to evaluate  
411 whether a given service matches its needs and, if so, the method for the consumer to interact  
412 with the service.

### 413 **3.2.1.2 Willingness**

414 Associated with all service interactions is intent – it is an intentional act to initiate and to  
415 participate in a service interaction. For example, if a service consumer discovers a service via its  
416 description in a registry, and the consumer initiates an interaction, if the service provider does not  
417 cooperate then there can be no interaction. In some circumstances it is precisely the correct  
418 behavior for a service to fail to respond – for example, it is the classic defense against certain  
419 denial-of-service attacks.

420 The extent of a service participant's willingness to engage in service interactions may be the  
421 subject of policies. Those policies may be documented in the service description.

422 Of course, willingness on the part of service providers and consumers to interact is not the same  
423 as a willingness to perform requested actions. A service provider that rejects all attempts to cause  
424 it to perform some action may still be fully willing and engaged in interacting with the consumer.

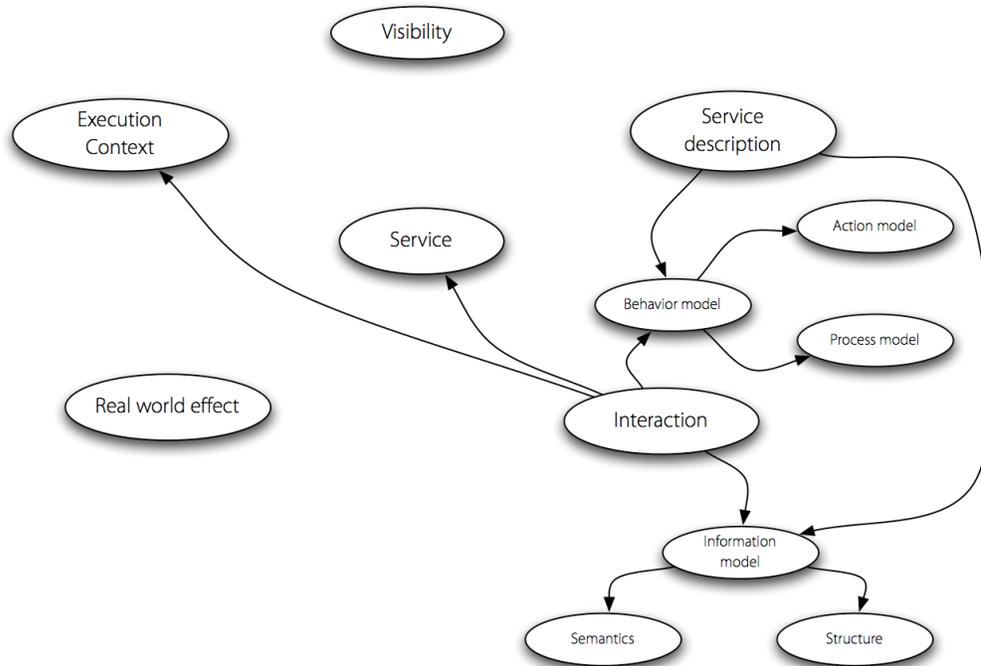
### 425 **3.2.1.3 Reachability**

426 Reachability is the relationship between service participants where they are able to interact;  
427 possibly by exchanging information. Reachability is an essential pre-requisite for service  
428 interaction – participants **MUST** be able to communicate with each other.

429 A service consumer may have the intention of interacting with a service, and may even have all  
430 the information needed to communicate with it. However, if the service is not reachable, for  
431 example if there is not a communication path between the consumer and provider, then,  
432 effectively, the service is not visible to the consumer.

### 433 **3.2.2 Interacting with services**

434 Interacting with a service involves performing actions against the service. In many cases, this is  
435 accomplished by sending and receiving messages, but there are other modes possible that do  
436 not involve explicit message transmission. For example, a service interaction may be effected by  
437 modifying the state of a shared resource. However, for simplicity, we often refer to message  
438 exchange as the primary mode of interaction with a service.



439

440 *Figure 5 Service Interaction concepts*

441 Figure 5 illustrates the key concepts that are important in understanding what it is involved in  
 442 interacting with services; these revolve around the service description – which references a  
 443 **information model** and a **behavior model**.

### 444 3.2.2.1 Information model

445 The information model of a service is a characterization of the information that may be exchanged  
 446 with the service. Only information and data that are potentially exchanged with a service are  
 447 generally included within that service's information model.

448 The scope of the information model includes the format of information that is exchanged, the  
 449 structural relationships within the exchanged information and also the definition of terms used.

450 Particularly for information that is exchanged across an ownership boundary, an important aspect  
 451 of the service information model is the consistent interpretation of strings and other tokens in the  
 452 information.

453 The extent to which one system can effectively interpret information from another system is  
 454 governed by the **semantic engagement** of the various systems. The semantic engagement of a  
 455 system is a relationship between the system and information it may encounter. This is highly  
 456 variable and application dependent; for example an encryption service interprets all information  
 457 as a stream of bytes for it to encrypt or decrypt, whereas a database service would attempt to  
 458 interpret the same information stream in terms of requests to query and/or modify the database.

459 Loosely, one might partition the interpretation of an informational block into structure (syntax) and  
 460 meaning (semantics); although both are part of the information model.

#### 461 3.2.2.1.1 Structure

462 Knowing the representation, structure, and form of information required is a key initial step in  
 463 ensuring effective interactions with a service. There are several levels of such structural  
 464 information; including the encoding of character data, the format of the data and the structural  
 465 data types associated with elements of the information.

466 A described information model typically has a great deal to say about the form of messages.  
467 However, knowing the type of information is not sufficient to completely describe the appropriate  
468 interpretation of data. For example, within a street address structure, the city name and the street  
469 name are typically given the same data type – some variant of the string type. However, city  
470 names and street names are not really the same type of thing at all. Distinguishing the correct  
471 interpretation of a city name string and a street name string is not possible using type-based  
472 techniques – it requires additional information that cannot be expressed purely in terms of the  
473 structure of data.

#### 474 **3.2.2.1.2 Semantics**

475 The primary task of any communication infrastructure is to facilitate the exchange of information  
476 and the exchange of intent. For example, a purchase order combines two somewhat orthogonal  
477 aspects: the description of the items being purchased and the fact that one party intends to  
478 purchase those items from another party. Even for exchanges that do not cross any ownership  
479 boundaries, exchanges with services have similar aspects.

480 Especially in the case where the exchanges are across ownership boundaries, a critical issue is  
481 the interpretation of the data. This interpretation **MUST** be consistent between the participants in  
482 the service interaction. Consistent interpretation is a stronger requirement than merely type (or  
483 structural) consistency – the tokens in the data itself must also have a shared basis.

484 There is often a huge potential for variability in representing street addresses. For example, an  
485 address in San Francisco, California may have variations in the way the city is represented: SF,  
486 San Francisco, San Fran, the City by the Bay are all alternate denotations of the same city. For  
487 successful exchange of address information, all the participants must have a consistent view of  
488 the meaning of the address tokens if address information is to be reliably shared.

489 The formal descriptions of terms and the relationships between them (e.g., an ontology) provides  
490 a firm basis for selecting correct interpretations for elements of information exchanged. For  
491 example, an ontology can be used to capture the alternate ways of expressing the name of a city  
492 as well as distinguishing a city name from a street name.

493 Note that, for the most part, it is not expected that service consumers and providers would  
494 actually exchange descriptions of terms in their interaction but, rather, would reference existing  
495 descriptions – the role of the semantics being a background one – and these references would be  
496 included in the service descriptions.

497 Specific domain semantics are beyond the scope of this reference model; but there is a  
498 requirement that the service interface enable providers and consumers to identify unambiguously  
499 those definitions that are relevant to their respective domains.

#### 500 **3.2.2.2 Behavior model**

501 The second key requirement for successful interactions with services is knowledge of the actions  
502 invoked against the service and the process or temporal aspects of interacting with the service.  
503 This is characterized as knowledge of the actions on, responses to, and temporal dependencies  
504 between actions on the service.

505 For example, in a security-controlled access to a database, the actions available to a service  
506 consumer include presenting credentials, requesting database updates and reading results of  
507 queries. The security may be based on a challenge-response protocol. For example, the initiator  
508 presents an initial token of identity, the responder presents a challenge and the initiator responds  
509 to the challenge in a way that satisfies the database. Only after the user's credentials have been  
510 verified will the actions that relate to database update and query be accepted.

511 The sequences of actions involved are a critical aspect of the knowledge required for successful  
512 use of the secured database.

### 513 **3.2.2.2.1 Action model**

514 The **action model** of a service is the characterization of the actions that may be invoked against  
515 the service. Of course, a great portion of the behavior resulting from an action may be private;  
516 however, the expected public view of a service surely includes the implied effects of actions.

517 For example, in a service managing a bank account, it is not sufficient to know that you need to  
518 exchange a given message (with appropriate authentication tokens), in order to use the service. It  
519 is also necessary to understand that using the service may actually affect the state of the account  
520 (for example, withdrawing cash); that dependencies are involved (for example, a withdrawal  
521 request must be less than the account balance); or that the data changes made have different  
522 value in different contexts (for example, changing the data in a bank statement is not the same as  
523 changing the actual data representing the amount in an account).

### 524 **3.2.2.2.2 Process Model**

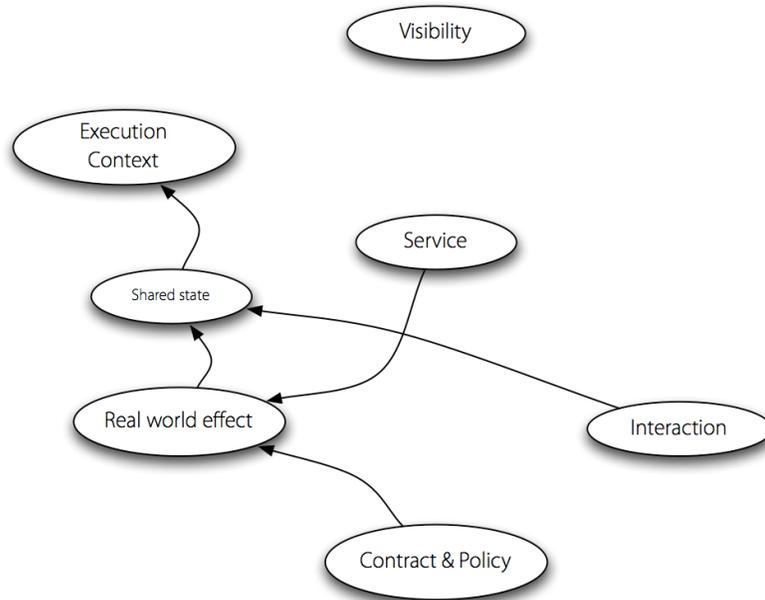
525 The **process model** characterizes the temporal relationships between and temporal properties of  
526 actions and events associated with interacting with the service.

527 Note that although the process model is an essential part of this Reference Model, its extent is  
528 not completely defined. In some architectures the process model will include aspects that are not  
529 strictly part of SOA – for example, in this Reference Model we do not address the orchestration of  
530 multiple services, although orchestration and choreography may be part of the process model of  
531 a given architecture. At a minimum, the process model **MUST** cover the interactions with the  
532 service itself.

533 Beyond the straightforward mechanics of interacting with a service there are other, higher-order,  
534 attributes of services' process models that are also often important. These can include whether  
535 the service is **idempotent**, whether the service is **long-running** in nature and whether it is  
536 important to account for any **transactional** aspects of the service.

### 537 **3.2.3 Real World Effect**

538 There is always a particular purpose associated with interacting with a service. Conversely, a  
539 service provider (and consumer) often has a priori conditions that apply to its interactions. The  
540 service consumer is trying to achieve some result by using the service, as is the service provider.  
541 At first sight, such a goal can often be expressed as “trying to get the service to do something”.  
542 This is sometimes known as the real world effect of using a service. For example, an airline  
543 reservation service can be used in order to book travel – the desired real world effect being a seat  
544 on the right airplane.



545

546 *Figure 6 Real World Effect and shared state*

547 The internal actions that service providers and consumers perform as a result of participation in  
 548 service interactions are, by definition, private and fundamentally unknowable. By unknowable we  
 549 mean both that external parties cannot see others' private actions and, furthermore, SHOULD  
 550 NOT have explicit knowledge of them. Instead we focus on the set of facts shared by the parties  
 551 – the *shared state*. Actions by service providers and consumers lead to modifications of this  
 552 shared state; and the real world effect of a service interaction is the accumulation of the changes  
 553 in the shared state.

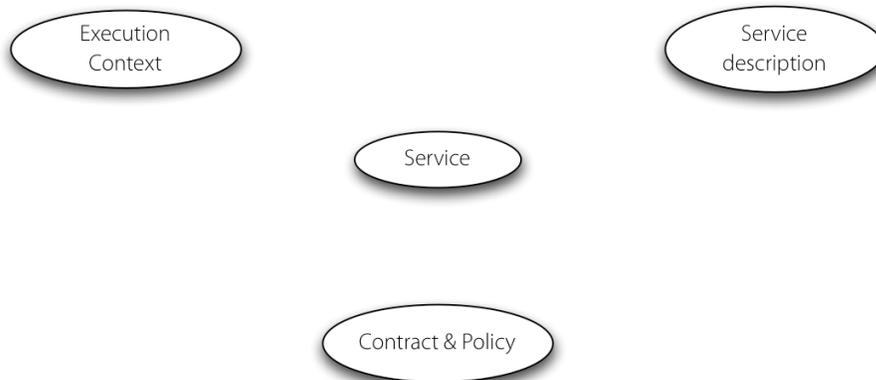
554 There is a strong relationship between the shared state and the interactions that lead up to that  
 555 state. The elements of the shared state SHOULD be inferable from that prior interaction together  
 556 with other context as necessary. In particular, it is not required that the state be recorded;  
 557 although without such recording it may become difficult to audit the interaction at a subsequent  
 558 time.

559 For example, when an airline has confirmed a seat for a passenger on a flight this represents a  
 560 fact that both the airline and the passenger share – it is part of their shared state. Thus the real  
 561 world effect of booking the flight is the modification of this shared state – the creation of the fact of  
 562 the booking. Flowing from the shared facts, the passenger, the airline, and interested third  
 563 parties may make inferences – for example, when the passenger arrives at the airport the airline  
 564 confirms the booking and permits the passenger onto the airplane (subject of course to the  
 565 passenger meeting the other requirements for traveling).

566 For the airline to know that the seat is confirmed it will likely require some private action to record  
 567 the reservation. However, a passenger should not have to know the details of the airline internal  
 568 procedures. The passenger's understanding of the reservation is independent of how the airline  
 569 maintains its records.

### 570 **3.3 About services**

571 In support of the dynamics of interacting with services are a set of concepts that are about  
 572 services themselves. These are the service description, the execution context of the service and  
 573 the contracts and policies that relate to services and service participants.



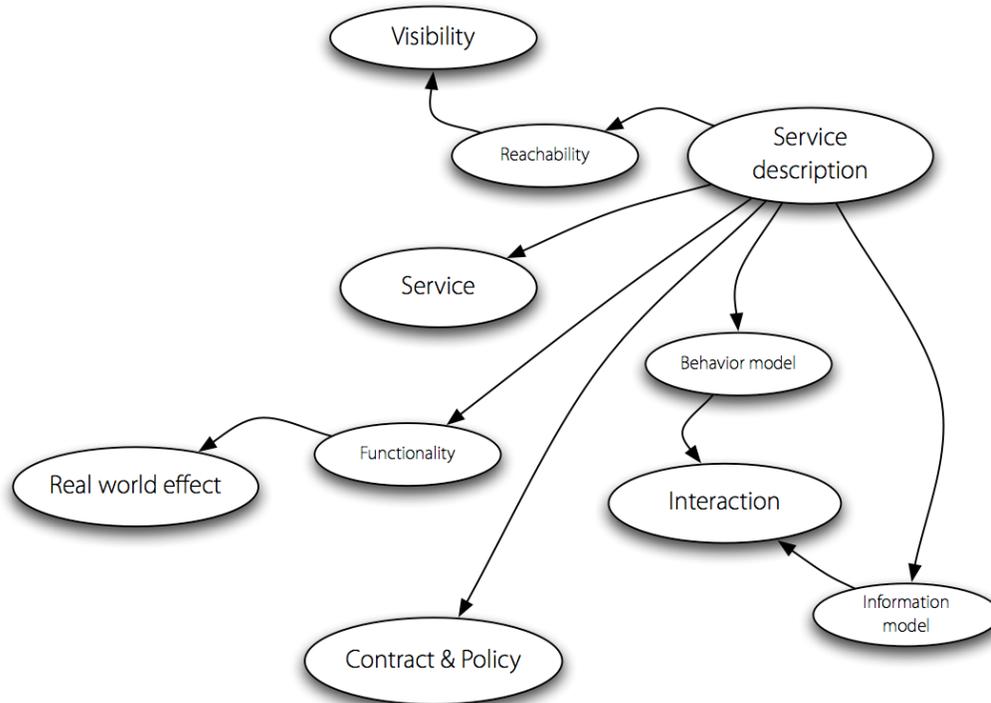
574

575 *Figure 7 About services*

### 576 **3.3.1 Service description**

577 One of the hallmarks of a Service Oriented Architecture is the large amount of associated  
578 documentation and description.

579 The service description represents the information needed in order to use a service. In most  
580 cases, there is no one “right” description but rather the elements of description required depend  
581 on the context and the needs of the parties using the associated entity. While there are certain  
582 elements that are likely to be part of any service description, most notably the information model,  
583 many elements such as function and policy may vary.



584

585 *Figure 8 Service description*

586 The purpose of description is to facilitate interaction and visibility, particularly when the  
587 participants are in different ownership domains, between participants in service interactions. By  
588 providing descriptions, it makes it possible for potential participants to construct systems that use  
589 services and even offer compatible services.

590 For example, descriptions allow participants to discriminate amongst possible choices for service  
591 interaction; such as whether the service provides required capabilities, how to access the service,  
592 and negotiate over specific service functionality. In addition, descriptions can be used to support  
593 the management of services, both from the service provider's perspective and the service  
594 consumer's perspective.

595 Best practice suggests that the service description SHOULD be represented using a standard,  
596 referenceable format. Such a format facilitates the use of common processing tools (such as  
597 discovery engines) that can capitalize on the service description.

598 While the concept of a SOA supports use of a service without the service consumer needing to  
599 know the details of the service implementation, the service description makes available critical  
600 information that a consumer needs in order to decide whether or not to use a service. In  
601 particular, a service consumer needs to possess the following items of information:

- 602 1. That the service exists and is **reachable**;
- 603 2. That the service performs a certain function or set of functions;
- 604 3. That the service operates under a specified set of constraints and policies;
- 605 4. That the service will (to some implicit or explicit extent) comply with policies as prescribed  
606 by the service consumer;
- 607 5. How to interact with the service in order to achieve the required objectives, including the  
608 format and content of information exchanged between the service and the consumer and  
609 the sequences of information exchange that may be expected.

610 While each of these items SHOULD be represented in any service description, the details can be  
611 included through references (links) to external sources and are NOT REQUIRED to be  
612 incorporated explicitly. This enables reuse of standard definitions, such as for functionality or  
613 policies.

614 Other sections of this document deal with these aspects of a service, but the following  
615 subsections discuss important elements as these relate to the service description itself.

### 616 **3.3.1.1 Service Reachability**

617 Reachability is an inherently pairwise relationship between service providers and service  
618 consumers. However, a service description SHOULD include sufficient data to enable a service  
619 consumer and service provider to interact with each other. This MAY include metadata such as  
620 the location of the service and what information protocols it supports and requires. It MAY also  
621 include dynamic information about the service, such as whether it is currently available.

### 622 **3.3.1.2 Service Functionality**

623 A service description SHOULD unambiguously express the function(s) of the service and the real  
624 world effects (see Section 3.2.3) that result from it being invoked. This portion of the description  
625 SHOULD be expressed in a way that is generally understandable by service consumers but able  
626 to accommodate a vocabulary that is sufficiently expressive for the domain for which the service  
627 provides its functionality. The description of functionality may include, among other possibilities,  
628 a textual description intended for human consumption or identifiers or keywords referenced to  
629 specific machine-processable definitions. For a full description, it MAY indicate multiple  
630 identifiers or keywords from a number of different collections of definitions.

631 Part of the description of functionality may include underlying technical assumptions that  
632 determine the limits of functionality exposed by the service or of the underlying capability. For  
633 example, the amounts dispensed by an automated teller machine (ATM) are consistent with the  
634 assumption that the user is an individual rather than a business. To use the ATM, the user must  
635 not only adhere to the policies and satisfy the constraints of the associated financial institution  
636 (see Section 3.3.1.3 for how this relates to service description and Section 3.3.2 for a detailed  
637 discussion) but the user is limited to withdrawing certain fixed amounts of cash and a certain

638 number of transactions in a specified period of time. The financial institution, as the underlying  
639 capability, does not have these limits but the service interface as exposed to its customers does,  
640 consistent with its assumption of the needs of the intended user. If the assumption is not valid,  
641 the user may need to use another service to access the capability.

### 642 **3.3.1.3 Policies Related to a Service**

643 A service description MAY include support for associating policies with a service and providing  
644 necessary information for prospective consumers to evaluate if a service will act in a manner  
645 consistent with the consumer's constraints.

### 646 **3.3.1.4 Service Interface**

647 The service interface is the means for interacting with a service. It includes the specific protocols,  
648 commands, and information exchange by which actions are initiated that result in the real world  
649 effects as specified through the service functionality portion of the service description.

650 The specifics of the interface SHOULD be syntactically represented in a standard referenceable  
651 format. These prescribe what information needs to be provided to the service in order to access  
652 its capabilities and interpret responses. This is often referred to as the service's information  
653 model, see Section 3.2.2.1. It should be noted that the particulars of the interface format are  
654 beyond the scope of the reference model. However, the existence of interfaces and accessible  
655 descriptions of those interfaces are fundamental to the SOA concept.

656 While this discussion refers to a standard referenceable syntax for service descriptions, it is not  
657 specified how the consumer accesses the interface definition nor how the service itself is  
658 accessed. However, it is assumed that for a service to be usable, its interface MUST be  
659 represented in a format that allows interpretation of the interface information by its consumers.

### 660 **3.3.1.5 The Limits of Description**

661 There are well-known theoretic limits on the effectiveness of descriptions – it is simply not  
662 possible to specify, completely and unambiguously, the precise semantics of and all related  
663 information about a service.

664 There will always be unstated assumptions made by the describer of a service that must be  
665 implicitly shared by readers of the description. This applies to machine processable descriptions  
666 as well as to human readable descriptions.

667 Fortunately, complete precision is not necessary – what is required is sufficient scope and  
668 precision to support intended use.

669 Another kind of limit of service descriptions is more straightforward: whenever a repository is  
670 searched using any kind of query there is always the potential for *zero or more* responses – no  
671 matter how complete the search queries or the available descriptions appear to be. This is  
672 inherent in the principles involved in search.

673 In the case that there is more than one response, this set of responses has to be converted into a  
674 single choice. This is a private choice that must be made by the consumer of the search  
675 information.

### 676 **3.3.2 Policies and Contracts**

677 A **policy** represents some constraint or condition on the use, deployment or description of an  
678 owned entity as defined by any participant. A **contract**, on the other hand, represents an  
679 agreement by two or more parties. Like policies, agreements are also about the conditions of use  
680 of a service; they may also constrain the expected real world effects of using a service. The  
681 reference model is focused primarily on the concept of policies and contracts as they apply to  
682 services. We are not concerned with the form or expressiveness of any language used to  
683 express policies and contracts.

### 684 **3.3.2.1 Service Policy**

685 Conceptually, there are three aspects of policies: the policy assertion, the policy owner  
686 (sometimes referred to as the policy subject) and policy enforcement.

687 For example, the assertion: “All messages are encrypted” is an assertion regarding the forms of  
688 messages. As an assertion, it is measurable: it may be true or false depending on whether the  
689 traffic is encrypted or not. Policy assertions are often about the way the service is realized; i.e.,  
690 they are about the relationship between the service and its execution context, see 3.3.3.

691 A policy always represents a participant’s point of view. An assertion becomes the policy of a  
692 participant when they adopt the assertion as their policy. This linking is normally not part of the  
693 assertion itself. For example, if the service consumer declares that “All messages are encrypted”,  
694 then that reflects the policy of the service consumer. This policy is one that may be asserted by  
695 the service consumer independently of any agreement from the service provider.

696 Finally, a policy may be enforced. Techniques for the enforcement of policies depend on the  
697 nature of the policy. Conceptually, service policy enforcement amounts to ensuring that the policy  
698 assertion is consistent with the real world. This might mean preventing unauthorized actions to be  
699 performed or states to be entered into; it can also mean initiating compensatory actions when a  
700 policy violation has been detected. An unenforceable constraint is not a policy; it would be better  
701 described as a wish.

702 Policies potentially apply to many aspects of SOA: security, privacy, manageability, Quality of  
703 Service and so on. Beyond such infrastructure-oriented policies, participants MAY also express  
704 business-oriented policies – such as hours of business, return policies and so on.

705 Policy assertions SHOULD be written in a form that is understandable to, and processable by, the  
706 parties to whom the policy is directed. Policies MAY be automatically interpreted, depending on  
707 the purpose and applicability of the policy and how it might affect whether a particular service is  
708 used or not.

709 A natural point of contact between service participants and policies associated with the service is  
710 in the service description – see Section 3.3.1. It would be natural for the service description to  
711 contain references to the policies associated with the service.

### 712 **3.3.2.2 Service Contract**

713 Whereas a policy is associated with the point of view of individual participants, a contract  
714 represents an agreement between two or more participants. Like policies, contracts can cover a  
715 wide range of aspects of services: quality of service agreements, interface and choreography  
716 agreements and commercial agreements. Note that we are not necessarily referring to legal  
717 contracts here.

718 Thus, following the discussion above, a service contract is a measurable assertion that governs  
719 the requirements and expectations of two or more parties. Unlike policy enforcement, which is  
720 usually the responsibility of the policy owner, contract enforcement may involve resolving  
721 disputes between the parties to the contract. The resolution of such disputes may involve appeals  
722 to higher authorities.

723 Like policies, contracts may be expressed in a form that permits automated interpretation. Where  
724 a contract is used to codify the results of a service interaction, it is good practice to represent it in  
725 a machine processable form. Among other purposes, this facilitates automatic service  
726 composition. Where a contract is used to describe over-arching agreements between service  
727 providers and consumers, then the priority is likely to make such contracts readable by people.

728 Since a contract is inherently the result of agreement by the parties involved, there is a *process*  
729 associated with the agreement action. Even in the case of an implicitly agreed upon contract,  
730 there is logically an agreement action associated with the contract, even if there is no overt action  
731 of agreement. A contract may be arrived at by a mechanism that is not directly part of an SOA –  
732 an out of band process. Alternatively, a contract may be arrived at during the course of a service  
733 interaction – an in-band process.

### 734 3.3.3 Execution context

735 The **execution context** of a service interaction is the set of infrastructure elements, process  
736 entities, policy assertions and agreements that are identified as part of an instantiated service  
737 interaction, and thus forms a path between those with needs and those with capabilities. The  
738 consumer and provider can be envisioned as separate places on a map and, for a service to  
739 actually be invoked, a path must be established between those two places. This path is the  
740 execution context. As with a path between places, it can be a temporary connection (e.g. a  
741 tenuous footbridge of an ad hoc exchange) or a well-defined coordination (e.g. a super highway)  
742 that can be easily reused in the future.

743 The execution context is not limited to one side of the interaction; rather it concerns the totality of  
744 the interaction – including the service provider, the service consumer and the common  
745 infrastructure needed to mediate the interaction. While there may be third parties, for example,  
746 government regulators, who set some of the conditions for the execution context, this merely  
747 increases the conditions and constraints needing to be coordinated and may require additional  
748 information exchange to complete the execution context.

749 The execution context is central to many aspects of a service interaction. It defines, for example,  
750 a decision point for policy enforcement relating to the service interaction. Note that a policy  
751 decision point is not necessarily the same as an enforcement point: an execution context is not by  
752 itself something that lends itself to enforcement. On the other hand, any enforcement mechanism  
753 of a policy is likely to take into account the particulars of the actual execution context.

754 The execution context also allows us to distinguish services from one another. Different instances  
755 of the same service – denoting interactions between a given service provider and different service  
756 consumers for example – are distinguished by virtue of the fact that their execution contexts are  
757 different.

758 Finally, the execution context is also the context in which the interpretation of data that is  
759 exchanged takes place. A particular string has a particular meaning in a service interaction in a  
760 particular context – the execution context.

761 An execution context often evolves during a service interaction. The set of infrastructure  
762 elements, the policies and agreements that apply to the interaction, may well change during a  
763 given service interaction. For example, at an initial point in an interaction, it may be decided by  
764 the parties that future communication should be encrypted. As a result the execution context also  
765 changes – to incorporate the necessary infrastructure to support the encryption and continue the  
766 interaction.

767

---

## 4 Conformance Guidelines

768 The authors of this reference model envision that architects may wish to declare their architecture  
769 is conformant with this reference model. Conforming to a Reference Model is not generally an  
770 easily automatable task – given that the Reference Model’s role is primarily to define concepts  
771 that are important to SOA rather than to give guidelines for implementing systems.

772 However, we do expect that any given Service Oriented Architecture will reference the concepts  
773 outlined in this specification. As such, we expect that any design for a system that adopts the  
774 SOA approach will

- 775 • Have entities that can be identified as services as defined by this Reference Model;
- 776 • Be able to identify how visibility is established between service providers and consumers;
- 777 • Be able to identify how interaction is mediated;
- 778 • Be able to identify how the effect of using services is understood;
- 779 • Have descriptions associated with services;
- 780 • Be able to identify the execution context required to support interaction; and
- 781 • It will be possible to identify how policies are handled and how contracts may be modeled  
782 and enforced.

783 It is not appropriate for this specification to identify *best practices* with respect to building SOA-  
784 based systems. However, the ease with which the above elements can be identified within a  
785 given SOA-based system could have significant impact on the scalability, maintainability and  
786 ease of use of the system.

---

787 **5 References**

788 **5.1 Normative**

789     **[RFC2119]**     S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*,  
790                     <http://www.ietf.org/rfc/rfc2119.txt>, IETF RFC 2119, March 1997.  
791

792 **5.2 Non-Normative**

793     **[W3C WSA]**     W3C Working Group Note "Web Services Architecture",  
794                     <http://www.w3.org/TR/ws-arch/> , 11 February 2004

---

## 795 Appendix A. Glossary

796 The glossary contains a concise definition of terms used within this specification, but the full  
797 description in the text is the normative description.

798

### 799 Action Model

800 The characterization of the permissible actions that may be invoked against a service.  
801 See Section 3.2.2.2.1.

### 802 Architecture

803 A set of artifacts (that is: principles, guidelines, policies, models, standards and  
804 processes) and the relationships between these artifacts, that guide the selection,  
805 creation, and implementation of solutions aligned with business goals.

806 Software architecture is the structure or structures of an information system consisting of  
807 entities and their externally visible properties, and the relationships among them.

### 808 Awareness

809 A state whereby one party has knowledge of the existence of the other party. Awareness  
810 does not imply willingness or reachability. See Section 3.2.1.1.

### 811 Behavior Model

812 The characterization of (and responses to, and temporal dependencies between) the  
813 actions on a service. See Section 3.2.2.2.

### 814 Capability

815 A real-world effect that a service provider is able to provide to a service consumer. See  
816 Section 2.1.

### 817 Execution context

818 The set of technical and business elements that form a path between those with needs  
819 and those with capabilities and that permit service providers and consumers to interact.  
820 See Section 3.3.3.

### 821 Framework

822 A set of assumptions, concepts, values, and practices that constitutes a way of viewing  
823 the current environment.

### 824 Idempotency/Idempotent

825 A characteristic of a service whereby multiple attempts to change a state will always and  
826 only generate a single change of state if the operation has already been successfully  
827 completed once. See Section 3.2.2.2.2.

### 828 Information model

829 The characterization of the information that is associated with the use of a service. See  
830 Section 3.2.2.1.

### 831 Interaction

832 The activity involved in making using of a capability offered, usually across an ownership  
833 boundary, in order to achieve a particular desired real-world effect. See Section 3.2.3.

- 834 Offer  
835 An invitation to use the capabilities made available by a service provider in accordance  
836 with some set of policies.
- 837 Policy  
838 A statement of obligations, constraints or other conditions of use of an owned entity as  
839 defined by a participant. See Section 3.3.2.
- 840 Process Model  
841 The characterization of the temporal relationships between and temporal properties of  
842 actions and events associated with interacting with the service. See Section 3.2.2.2.2.
- 843 Reachability  
844 The ability of a service consumer and service provider to interact. Reachability is an  
845 aspect of visibility. See Section 3.2.1.3.
- 846 Real world effect  
847 The actual result of using a service, rather than merely the capability offered by a service  
848 provider. See Section 3.2.3.
- 849 Reference Architecture  
850 A reference architecture is an architectural design pattern that indicates how an abstract  
851 set of mechanisms and relationships realizes a predetermined set of requirements. See  
852 Section 1.1.
- 853 Reference Model  
854 A reference model is an abstract framework for understanding significant relationships  
855 among the entities of some environment that enables the development of specific  
856 architectures using consistent standards or specifications supporting that environment.  
857 A reference model consists of a minimal set of unifying concepts, axioms and  
858 relationships within a particular problem domain, and is independent of specific  
859 standards, technologies, implementations, or other concrete details. See Section 1.1.
- 860 Semantics  
861 A conceptualization of the implied meaning of information, that requires words and/or  
862 symbols within a usage context. See Section 3.2.2.1.2.
- 863 Semantic Engagement  
864 The relationship between an agent and a set of information that depends on a particular  
865 interpretation of the information. See Section 3.2.2.1.
- 866 Service  
867 The means by which the needs of a consumer are brought together with the capabilities  
868 of a provider. See Section 3.1.
- 869 Service Consumer  
870 An entity which seeks to satisfy a particular need through the use capabilities offered by  
871 means of a service.
- 872 Service description  
873 The information needed in order to use, or consider using, a service. See Section 3.3.1.
- 874 Service Interface  
875 The means by which the underlying capabilities of a service are accessed. See Section  
876 3.3.1.4.

- 877 Service Oriented Architecture (SOA)
- 878 Service Oriented Architecture is a paradigm for organizing and utilizing distributed  
879 capabilities that may be under the control of different ownership domains. It provides a  
880 uniform means to offer, discover, interact with and use capabilities to produce desired  
881 effects consistent with measurable preconditions and expectations. See Section 2.1.
- 882 Service Provider
- 883 An entity (person or organization) that offers the use of capabilities by means of a  
884 service.
- 885 Visibility
- 886 The capacity for those with needs and those with capabilities to be able to interact with  
887 each other. See Section 3.2.1.
- 888 Willingness
- 889 A predisposition of service providers and consumers to interact. See Section 3.2.1.2.

---

890 **Appendix B. Acknowledgments**

891 The following individuals were members of the committee during the development of this  
892 specification:

893 [ TODO: insert cte. Members ]

894

---

## Appendix C. Notices

896 OASIS takes no position regarding the validity or scope of any intellectual property or other rights  
897 that might be claimed to pertain to the implementation or use of the technology described in this  
898 document or the extent to which any license under such rights might or might not be available;  
899 neither does it represent that it has made any effort to identify any such rights. Information on  
900 OASIS's procedures with respect to rights in OASIS specifications can be found at the OASIS  
901 website. Copies of claims of rights made available for publication and any assurances of licenses  
902 to be made available, or the result of an attempt made to obtain a general license or permission  
903 for the use of such proprietary rights by implementers or users of this specification, can be  
904 obtained from the OASIS Executive Director.

905 OASIS invites any interested party to bring to its attention any copyrights, patents or patent  
906 applications, or other proprietary rights, which may cover technology that may be required to  
907 implement this specification. Please address the information to the OASIS Executive Director.

908 Copyright © OASIS Open 2005. *All Rights Reserved.*

909 This document and translations of it may be copied and furnished to others, and derivative works  
910 that comment on or otherwise explain it or assist in its implementation may be prepared, copied,  
911 published and distributed, in whole or in part, without restriction of any kind, provided that the  
912 above copyright notice and this paragraph are included on all such copies and derivative works.  
913 However, this document itself does not be modified in any way, such as by removing the  
914 copyright notice or references to OASIS, except as needed for the purpose of developing OASIS  
915 specifications, in which case the procedures for copyrights defined in the OASIS Intellectual  
916 Property Rights document must be followed, or as required to translate it into languages other  
917 than English.

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

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