



Reference Model for Service Oriented Architecture

Committee Draft 1.0, 2 February 2006

Document identifier:

wd-soa-rm-cd1

Location:

http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=soa-rm

Editors:

C. Matthew MacKenzie, Adobe Systems Incorporated, mattm@adobe.com
Ken Laskey, MITRE Corporation, klaskey@mitre.org
Francis McCabe, Fujitsu Limited, frank.mccabe@us.fujitsu.com
Peter Brown, Individual, peter@justbrown.net
Rebekah Metz, Booz Allen Hamilton, 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 list. Others should visit the SOA-RM TC home page at 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

The errata page for this specification is at:

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

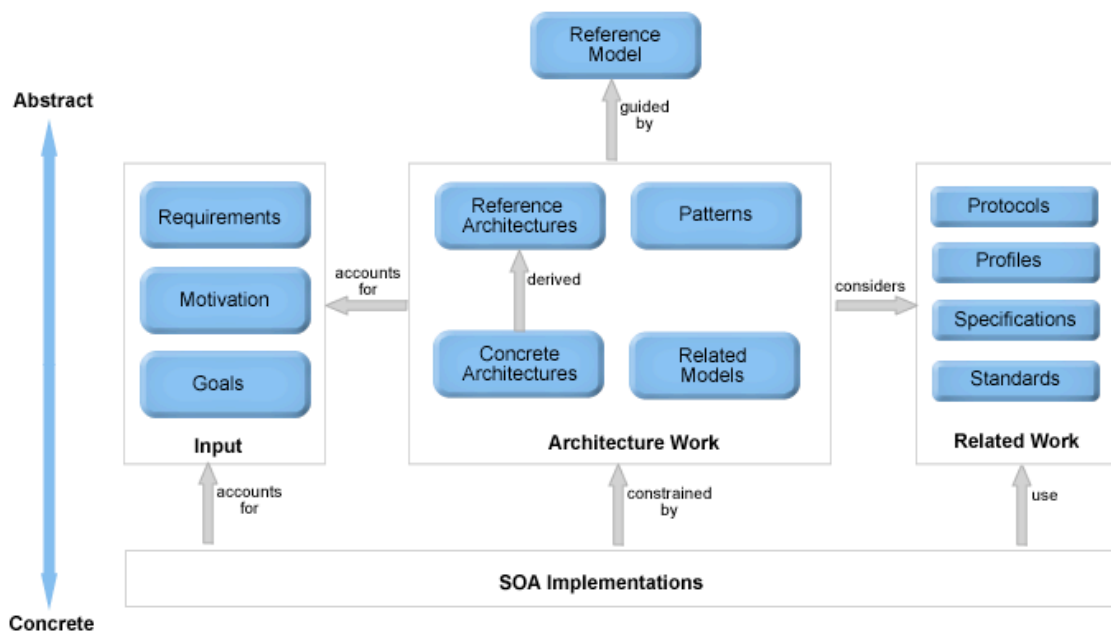
115 The goal of this reference model is to define the essence of service oriented architecture, and
116 emerge with a vocabulary and a common understanding of SOA. It provides a normative
117 reference that remains relevant for SOA as an abstract and powerful model, irrespective of the
118 various and 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. Note, it is not possible to describe every effect from using a capability, a cornerstone of SOA is that we can use capabilities without needing to know all the details.

220

221

222

223

224

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:

225

226

227

228

- The capability to perform work for another

229

- The specification of the work offered for another

230

- The offer to perform work for another

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

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

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

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

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

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

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

264 **2.1.1 A worked Service Oriented Architecture example**

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

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

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

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

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

291 **2.2 How is Service Oriented Architecture different?**

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

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

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

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

313 **2.3 The Benefits of Service Oriented Architecture**

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

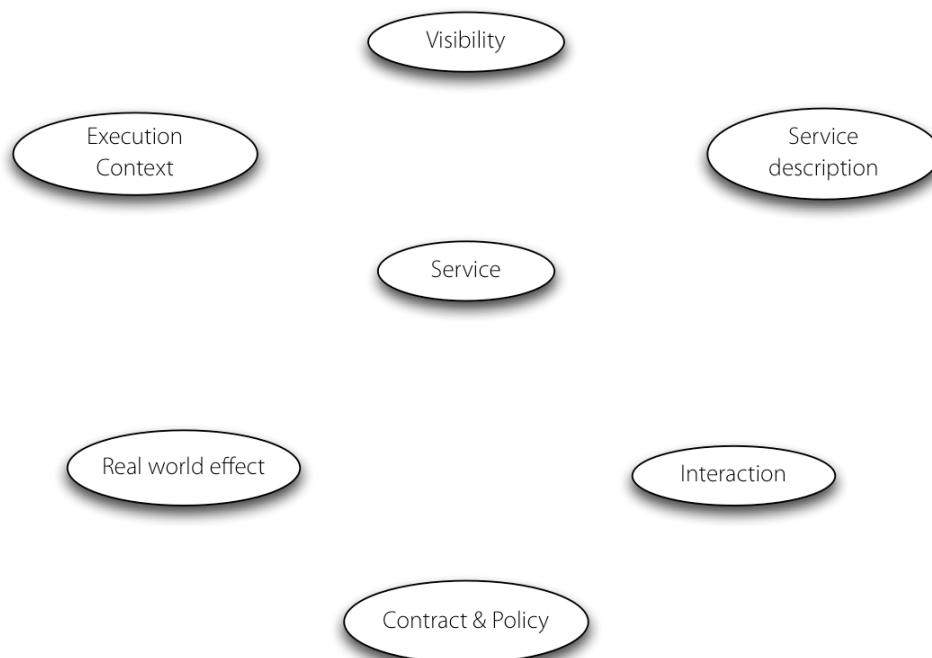
317 The value of SOA is that it provides a simple scalable paradigm for organizing large networks of
318 systems that require interoperability to realize the value inherent in the individual components.
319 Indeed, SOA is scalable because it makes the fewest possible assumptions about the network
320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

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

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

330 **3 The Reference Model**

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



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

335 **3.1 Service**

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

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

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

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

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

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

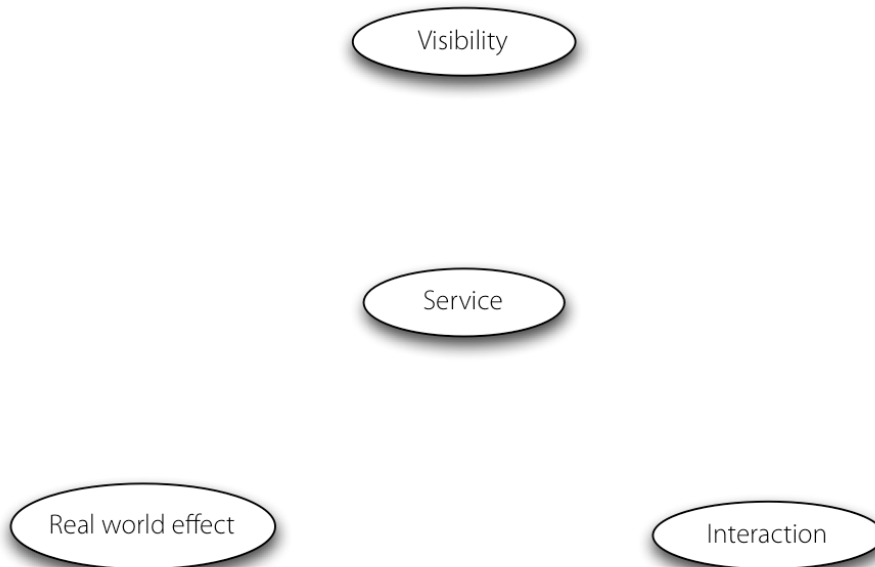
357

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

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

367 3.2 Dynamics of Services

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

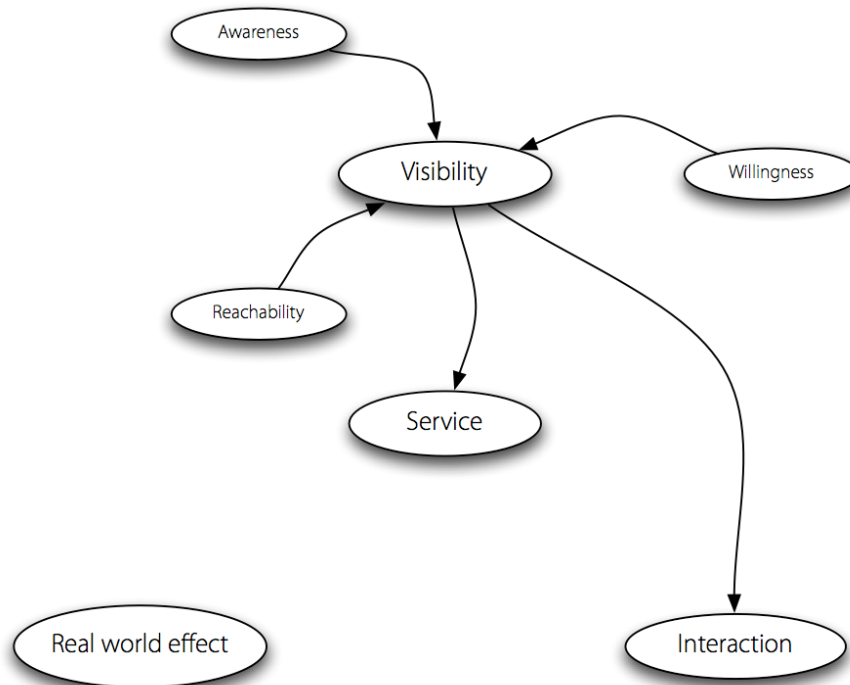


372

373 *Figure 3 Concepts around the dynamics of service*

374 3.2.1 Visibility

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



380

381 *Figure 4 Concepts around Visibility*

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

386 3.2.1.1 Awareness

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

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

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

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

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

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

414 **3.2.1.2 Willingness**

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

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

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

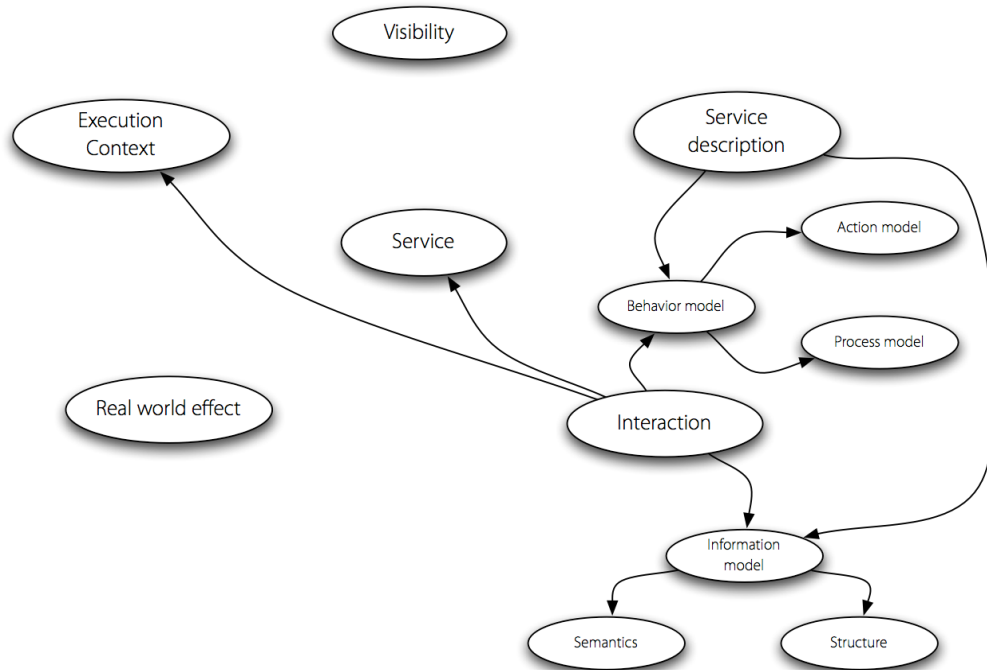
426 **3.2.1.3 Reachability**

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

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

434 **3.2.2 Interacting with services**

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



440

441 *Figure 5 Service Interaction concepts*

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

445 3.2.2.1 Information model

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

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

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

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

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

462 3.2.2.1.1 Structure

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

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

475 **3.2.2.1.2 Semantics**

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

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

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

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

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

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

501 **3.2.2.2 Behavior model**

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

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

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

514 **3.2.2.2.1 Action model**

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

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

525 **3.2.2.2.2 Process Model**

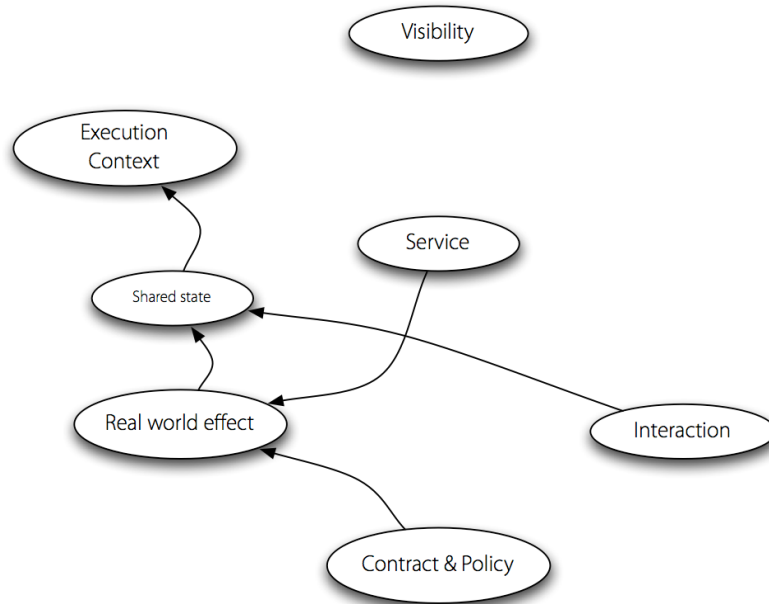
526 The **process model** characterizes the temporal relationships between and temporal properties of
527 actions and events associated with interacting with the service.

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

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

538 **3.2.3 Real World Effect**

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



546

547 *Figure 6 Real World Effect and shared state*

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

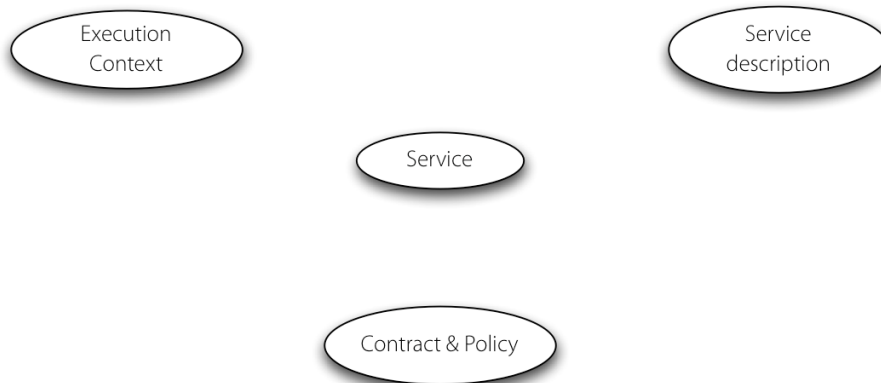
555 There is a strong relationship between the shared state and the interactions that lead up to that
 556 state. The elements of the shared state SHOULD be inferable from that prior interaction together
 557 with other context as necessary. In particular, it is not required that the state be recorded;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

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

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

571 **3.3 About services**

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



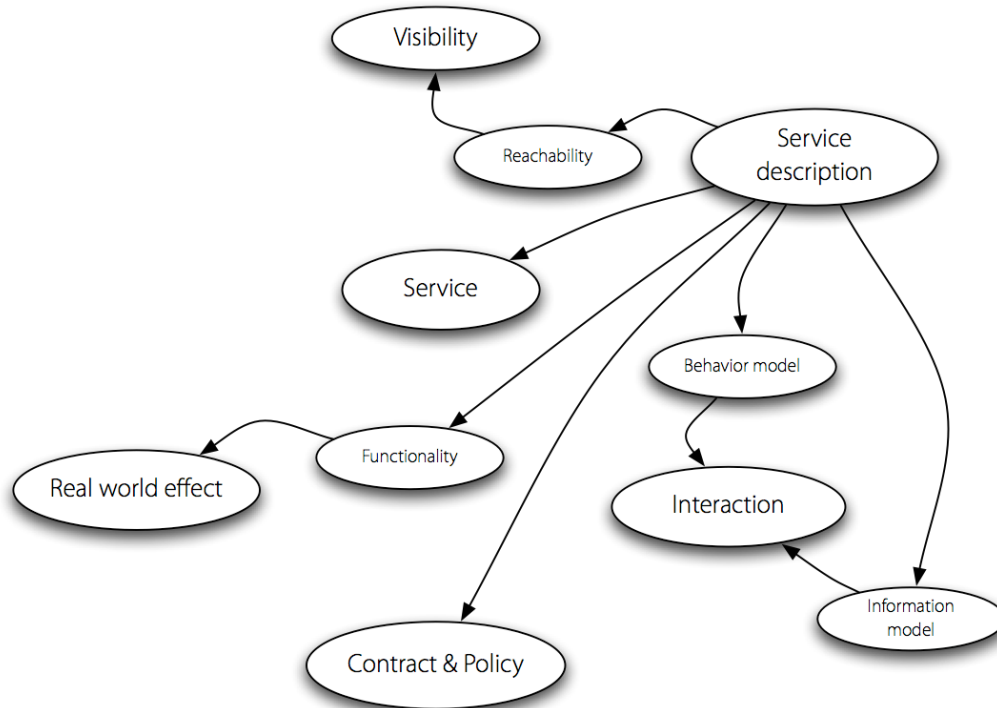
575

576 *Figure 7 About services*

577 **3.3.1 Service description**

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

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



585

586 *Figure 8 Service description*

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

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

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

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

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

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

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

617 **3.3.1.1 Service Reachability**

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

623 **3.3.1.2 Service Functionality**

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

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

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

643 **3.3.1.3 Policies Related to a Service**

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

647 **3.3.1.4 Service Interface**

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

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

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

661 **3.3.1.5 The Limits of Description**

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

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

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

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

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

677 **3.3.2 Policies and Contracts**

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

685 **3.3.2.1 Service Policy**

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

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

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

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

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

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

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

713 **3.3.2.2 Service Contract**

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

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

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

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

735 3.3.3 Execution context

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

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

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

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

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

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

768

4 Conformance Guidelines

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

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

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

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

788 **5 References**

789 **5.1 Normative**

790 **[RFC2119]** S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*,
791 <http://www.ietf.org/rfc/rfc2119.txt>, IETF RFC 2119, March 1997.
792

793 **5.2 Non-Normative**

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

796 Appendix A. Glossary

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

799

800 Action Model

801 The characterization of the permissible actions that may be invoked against a service.

802 See Section 3.2.2.2.1.

803 Architecture

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

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

809 Awareness

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

812 Behavior Model

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

815 Capability

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

818 Execution context

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

822 Framework

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

825 Idempotency/Idempotent

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

829 Information model

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

832 Interaction

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

835 Offer
836 An invitation to use the capabilities made available by a service provider in accordance
837 with some set of policies.

838 Policy
839 A statement of obligations, constraints or other conditions of use of an owned entity as
840 defined by a participant. See Section 3.3.2.

841 Process Model
842 The characterization of the temporal relationships between and temporal properties of
843 actions and events associated with interacting with the service. See Section 3.2.2.2.2.

844 Reachability
845 The ability of a service consumer and service provider to interact. Reachability is an
846 aspect of visibility. See Section 3.2.1.3.

847 Real world effect
848 The actual result of using a service, rather than merely the capability offered by a service
849 provider. See Section 3.2.3.

850 Reference Architecture
851 A reference architecture is an architectural design pattern that indicates how an abstract
852 set of mechanisms and relationships realizes a predetermined set of requirements. See
853 Section 1.1.

854 Reference Model
855 A reference model is an abstract framework for understanding significant relationships
856 among the entities of some environment that enables the development of specific
857 architectures using consistent standards or specifications supporting that environment.
858 A reference model consists of a minimal set of unifying concepts, axioms and
859 relationships within a particular problem domain, and is independent of specific
860 standards, technologies, implementations, or other concrete details. See Section 1.1.

861 Semantics
862 A conceptualization of the implied meaning of information, that requires words and/or
863 symbols within a usage context. See Section 3.2.2.1.2.

864 Semantic Engagement
865 The relationship between an agent and a set of information that depends on a particular
866 interpretation of the information. See Section 3.2.2.1.

867 Service
868 The means by which the needs of a consumer are brought together with the capabilities
869 of a provider. See Section 3.1.

870 Service Consumer
871 An entity which seeks to satisfy a particular need through the use capabilities offered by
872 means of a service.

873 Service description
874 The information needed in order to use, or consider using, a service. See Section 3.3.1.

875 Service Interface
876 The means by which the underlying capabilities of a service are accessed. See Section
877 3.3.1.4.

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

891 **Appendix B. Acknowledgments**

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

894 [TODO: insert cte. Members]

895

Appendix C. Notices

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

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

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

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

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

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