

# Lecture 2: OASIS SOA Reference Model

Lian Yu

The School of Software and Microelectronics

Peking University

No.24 Jin yuan RD, Beijing 102600

# Agenda

---

## Overview on Reference Model for SOA

- ▶ Introduction
- ▶ Service Oriented Architecture
- ▶ The Reference Model
  - ▶ Service
  - ▶ Dynamics of Services
  - ▶ About services
- ▶ SOA Concepts Example
- ▶ SOA principles
- ▶ Summary
- ▶ References

# OASIS SOA Reference Model

---

- ▶ The **OASIS SOA Reference Model** is a **reference model** for **Service-oriented architecture** (SOA) produced by **OASIS**, an IT industry standards body.
- ▶ SOA is an architectural paradigm for matching needs and capabilities that may be under disparate domains of ownership.
- ▶ While the term is often used as a noun, it is a very distinct style of architecture, applicable to domains beyond software systems.

# Reference model

---

- ▶ A **reference model** in systems, enterprise, and software engineering is an abstract framework for understanding significant relationships among the entities of some environment, and for the development of consistent standards or specifications supporting that environment.
- ▶ A reference model is based on a small number of unifying concepts and may be used as a basis for education and explaining standards to a non-specialist.
- ▶ A reference model is not directly tied to any standards, technologies or other concrete implementation details, but it does seek to provide a common semantics that can be used unambiguously across and between different implementations.
- ▶ The Reference Model for SOA is a lexicon that captures the style of architecture known as SOA.

# History

---

- ▶ The OASIS SOA Reference Model is a product of the OASIS SOA Reference Model (SOA-RM) Technical Committee (TC).
- ▶ Prior to this initiative, no standard definition of SOA had existed.
  - ▶ The SOA-RM TC was chartered in February 2005 to develop a core **Reference Model** to guide and foster the creation of specific service-oriented architectures, and to publish a reference model for SOA, as well as one or more reference architectures based on the Reference Model.
- ▶ The reference model was approved as an OASIS Standard by OASIS members in October 2006.

# Current status

---

- ▶ While the OASIS SOA Reference model has been welcomed in some quarters, there is some indication of rival initiatives by other standards bodies, including the [Open Group](#).
- ▶ **Future Work**
- ▶ At this time, the future work of the SOA-RM TC beyond the current specification has not yet been defined. The SOA-RA subcommittee is continuing to developing an SOA reference architecture based on the SOA-RM specification.

# Reference Model for SOA (1 / 3)

---

- ▶ 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.

# Reference Model for SOA (2/3)

---

- ▶ 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.
- ▶ The relationship between the Reference Model and particular architectures, technologies and other aspects of SOA is illustrated in Figure 1.



# Reference Model for SOA (3/3)

---

- ▶ 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.

# Agenda

---

- ▶ Overview on Reference Model for SOA

## Introduction

- ▶ Service Oriented Architecture
- ▶ The Reference Model
  - ▶ Service
  - ▶ Dynamics of Services
  - ▶ About services
- ▶ SOA Concepts Example
- ▶ SOA principles
- ▶ Summary
- ▶ References

# Introduction

---

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



# What is a reference model?

---

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

# The purpose of a reference model

---

- ▶ The purpose of a reference model is to provide a **common conceptual framework** that can be used consistently across and between different implementations and is of particular use in modeling specific solutions.

## Case Examples

Example: residential housing

A reference model: eating areas, hygiene areas and sleeping areas

A **reference architecture**: bedrooms, kitchens, hallways

An actual – or concrete – architecture: particular arrangements of windows, construction materials to be used and so on

# A Reference Model for Service Oriented Architectures (1 / 3)

---

- ▶ The goal of this reference model is to define the essence of service oriented architecture, and emerge with a **vocabulary** and **a common understanding** of SOA.
- ▶ It provides a normative reference that remains relevant for SOA as an abstract and powerful model, **irrespective of** the various and inevitable technology evolutions that will influence SOA deployment.
- ▶ Figure 1 shows how a reference model for SOA relates to other distributed systems architectural inputs.

# OASIS SOA Reference Model

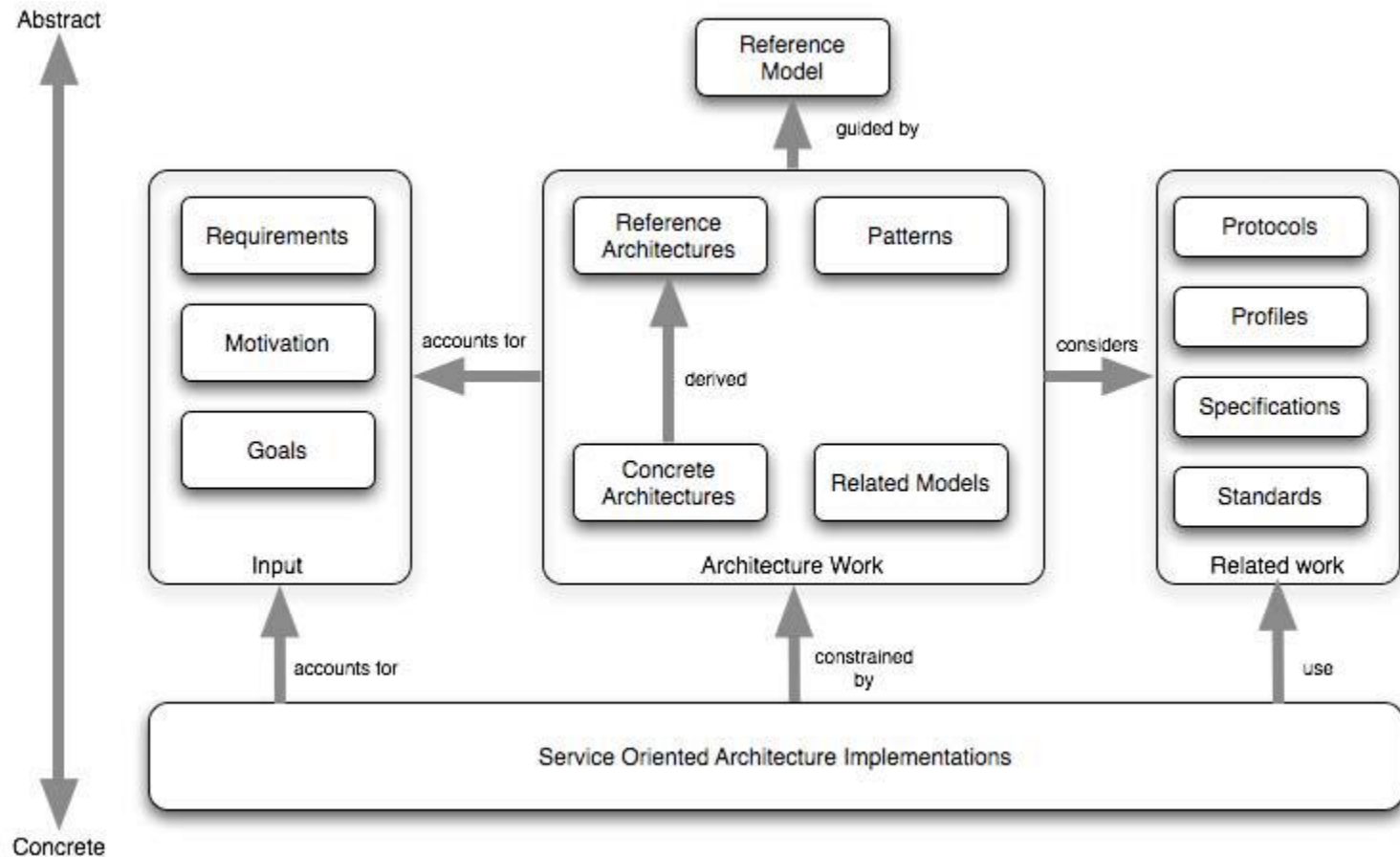


Figure 1 How the Reference Model relates to other work

# A Reference Model for Service Oriented Architectures (2/3)

---

- ▶ The concepts and relationships defined by the reference model are intended to be **the basis** for describing references architectures and patterns that will define more specific categories of SOA designs.
- ▶ Concrete architectures arise from a combination of reference architectures, architectural patterns and additional requirements, including those imposed by technology environments.
- ▶ Architecture must account for **the goals, motivation, and requirements** that define the actual problems being addressed.




# A Reference Model for Service Oriented Architectures (3/3)

---

- ▶ While reference architectures can form the basis of classes of solutions, concrete architectures will define specific solution approaches.
- ▶ Architecture is often developed in the context of a pre-defined **environment**, such as the protocols, profiles, specifications, and standards that are pertinent.
- ▶ SOA implementations combine all of these elements, from the more generic architectural principles and infrastructure to the specifics that define the current needs, and represent specific implementations that will be built and used in an operational environment.

# Agenda

---

- ▶ Overview on Reference Model for SOA
- ▶ Introduction
-  Service Oriented Architecture
- ▶ The Reference Model
  - ▶ Service
  - ▶ Dynamics of Services
  - ▶ About services
- ▶ SOA Concepts Example
- ▶ SOA principles
- ▶ Summary
- ▶ References

# What is SOA? (1 / 4)

---

- ▶ According to the OASIS SOA-RM (Reference Model) specification:
  - ▶ SOA is a paradigm for organizing and utilizing **distributed capabilities** that may be under the control of different **ownership domains**.
  - ▶ It provides **a uniform means** to offer, discover, interact with and use capabilities to produce desired effects consistent with **measurable preconditions** and **expectations**.
- ▶ The SOA-RM specification bases its definition of SOA around the concept of “**needs** and **capabilities**”,
  - ▶ where SOA provides a mechanism for matching needs of service consumers with capabilities provided by service providers.

<http://www.oasis-open.org/committees/download.php/19679/soa-rm-cs.pdf>

# What is SOA? (2/4)

---

- ▶ SOA is a design for **linking business** and **computational resources** (principally organizations, applications and data) on demand
  - ▶ to achieve the desired results for service consumers (which can be end users or other services).

[http://en.wikipedia.org/wiki/Service-oriented\\_architecture#SOA\\_definitions](http://en.wikipedia.org/wiki/Service-oriented_architecture#SOA_definitions)

# What is SOA? (3/4)

---

- ▶ In Service-Oriented Architecture, **autonomous**, **loosely-coupled** and **coarse-grained** services with well-defined interfaces provide **business functionality** and can be **discovered** and accessed through a supportive infrastructure.
- ▶ This allows internal and external system **integration** as well as the flexible **reuse** of application logic through the **composition of services**.

<http://blogs.zdnet.com/service-oriented/?p=490>

# What is SOA? (4/4)

---

- ▶ The modularization of business functions for greater **flexibility and reusability**.
  - ▶ Instead of building monolithic applications for each department, an SOA organizes business software in a granular fashion so that
    - ▶ common functions can be used interchangeably by different departments internally and by external business partners as well.
  - ▶ The more granular the components (the more pieces), the more they can be reused.
- ▶ SOA is a way of thinking about **IT assets as service components**.
  - ▶ When functions in a large application are made into stand-alone services that can be accessed separately, they are beneficial to several parties.



<http://www.techweb.com/encyclopedia/defineterm.jhtml?term=SOA>

## Summary – What is SOA?

---

- ▶ 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.

## SOA differ from other approaches? (2/2)

---

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



# Why SOA? (1 / 4)

---

- ▶ SOA can help businesses respond more quickly and cost-effectively to changing market conditions.
- ▶ This style of **architecture** promotes reuse at the macro (service) level rather than micro (classes) level.
- ▶ It can also simplify interconnection to - and usage of - existing IT (legacy) assets.

## Why SOA? (2/4)

---

- ▶ SOA promotes the goal of separating users (consumers) from the service implementations.
- ▶ Services can therefore be run on various distributed platforms and be accessed across networks.
- ▶ This can also maximize reuse of services.

# Why SOA? (3/4)

---

- ▶ SOA is an architectural and design discipline conceived to achieve the goals of
  - ▶ increased **interoperability** (information exchange, reusability, and composability),
  - ▶ increased **federation** (uniting resources & apps while maintaining their individual autonomy & self-governance), and
  - ▶ increased business & technology domain **alignment**.

# Why SOA? (4 / 4)

---

- ▶ A service is a stand-alone unit of functionality which is available only via a formally defined interface.
  - ▶ Services can be some kind of "nano-enterprises" which are easy to produce and improve.
  - ▶ Also services can be "mega-corporations" which are constructed as coordinated work of sub-ordinate services .
- ▶ Services generally adhere to the following principles of service-orientation:
  - ▶ *formal contract, loose coupling, abstraction, reusability, autonomy, statelessness, discoverability, and composability.*

# How is Service Oriented Architecture different?

---

- ▶ Unlike Object Oriented Programming paradigms, where the focus is on **packaging data with operations**, the central focus of Service Oriented Architecture is the task or business function – **getting something done**.
- ▶ This distinction manifests itself in several ways:
  - ▶ OO has intentional melding of methods to a given data object. The methods can be thought of as a property of the object. For SOA, one can think of the services **as being the access** to methods but the actual existence of methods and any connection to objects is incidental.
  - ▶ To use an object, it must first be instantiated while one interacts with a service where it exists.
  - ▶ An object exposes structure but there is no way to express semantics other than what can be captured as comments in the class definition. SOA emphasizes the need for clear semantics.

# SOA differ from other approaches? (1 / 2)

---

- ▶ How does this paradigm of SOA differ from other approaches to organizing and understanding Information Technology assets?
  - ▶ Essentially, there are two areas in which it differs both of which shape the framework of concepts that underlie distributed systems.
- ▶ First, SOA reflects **the reality** that ownership boundaries are a motivating consideration in the architecture and design of systems. This recognition is evident in the core concepts of visibility, interaction and effect.

# About Service

---

- ▶ 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:
  - ▶ The **CAPABILITY** to perform work for another
  - ▶ The **SPECIFICATION** of the work offered for another
  - ▶ The **OFFER** to perform work for another

# The Benefits of Service Oriented Architecture (1 / 4)

---

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



# The Benefits of Service Oriented Architecture (2/4)

---

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

# The Benefits of Service Oriented Architecture (3/4)

---

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


# The Benefits of Service Oriented Architecture (4/4)

---

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

# Agenda

---

- ▶ Overview on Reference Model for SOA
- ▶ Introduction
- ▶ Service Oriented Architecture
- ▶  The Reference Model
  - ▶ Service
  - ▶ Dynamics of Services
  - ▶ About services
- ▶ SOA Concepts Example
- ▶ SOA principles
- ▶ Summary
- ▶ References

# The Reference Model

---

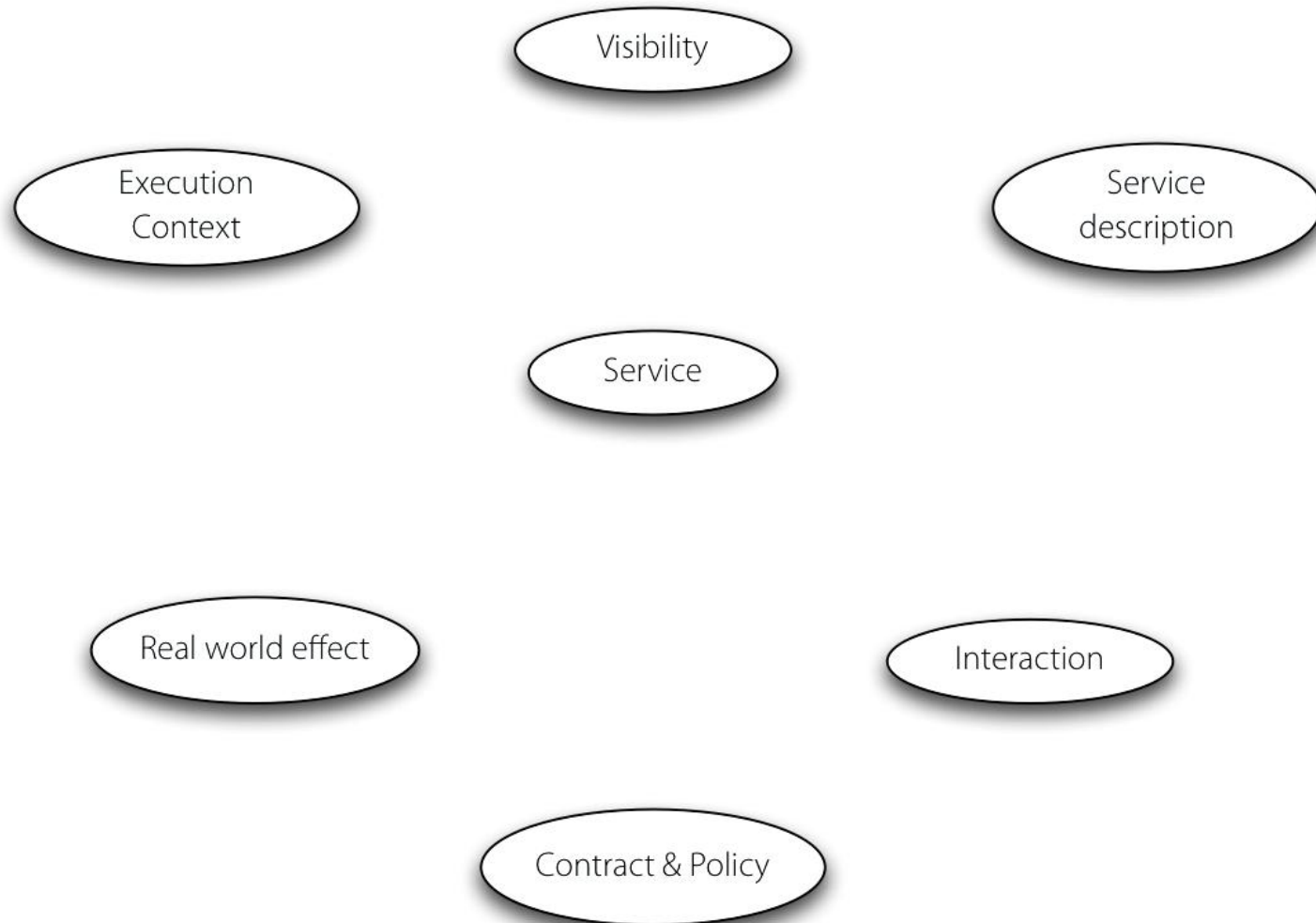
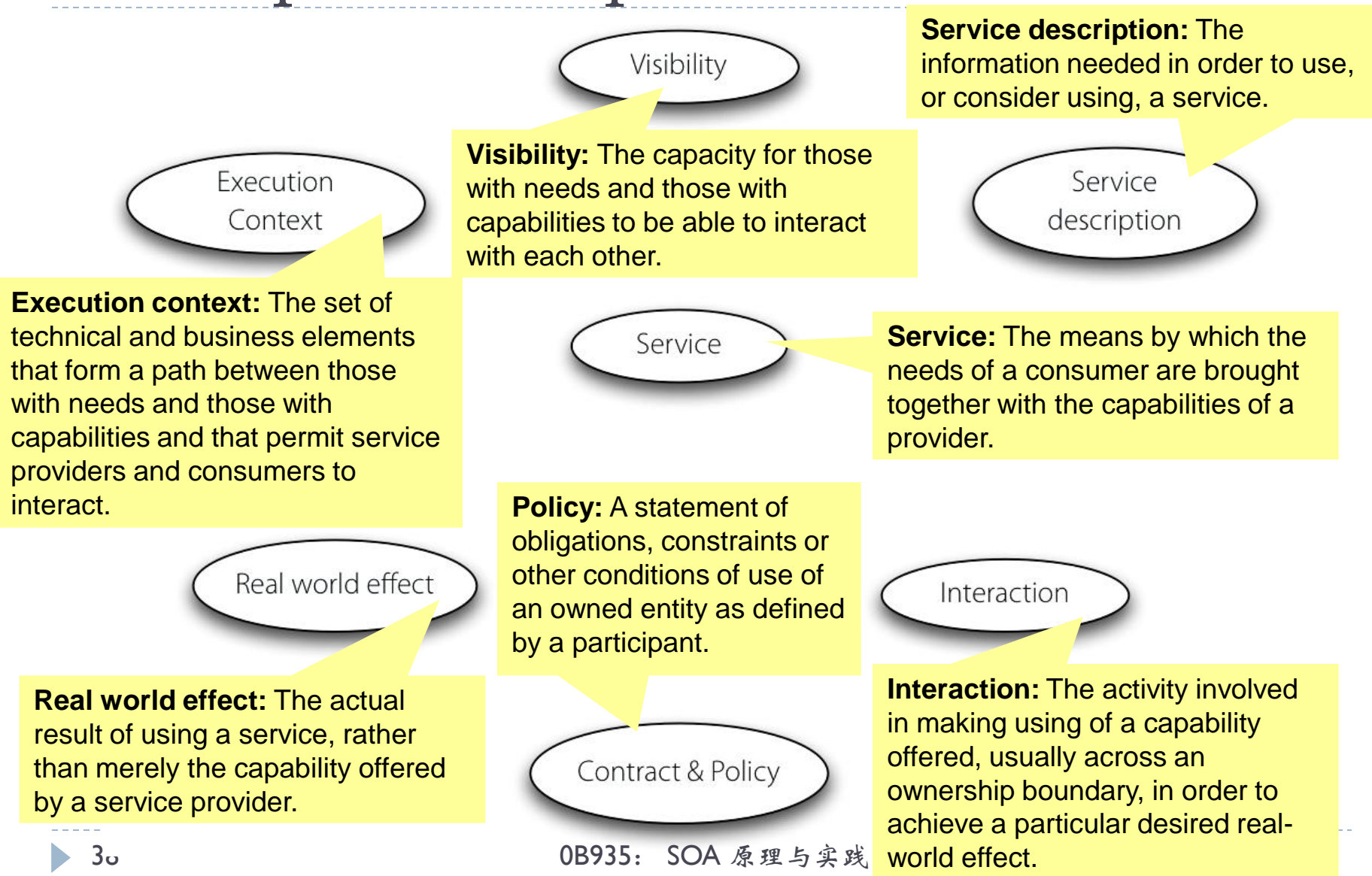


Figure 3 Principal concepts in the Reference Model

# Principal concepts



# Service (1 / 4)

---

- ▶ What is Service?
- ▶ OASIS Reference Model defines as follows:
  - ▶ A mechanism to enable access to one or more *capabilities*, where the access is provided using a *prescribed interface* and is exercised *consistent with constraints and policies* as specified by the *service description*.
  - ▶ A service is provided by an entity – the **SERVICE PROVIDER** – for use *by others*, but the eventual consumers of the service may not be known to the service provider and may demonstrate uses of the service beyond the scope originally conceived by the provider.

## Service (2/4)

---

- ▶ A service is accessed by means of a **service interface**, where the interface comprises the specifics of how to access the underlying capabilities. There are **no constraints** on **what constitutes** the underlying capability or **how access** is implemented by the service provider.
- ▶ A service is opaque in that its implementation is typically hidden from the **SERVICE CONSUMER** except for
  - ▶ (1) the information and behavior models exposed through the service interface and
  - ▶ (2) the information required by service consumers to determine whether a given service is appropriate for their needs.



## Service (3/4)

---

- ▶ The consequence of invoking a service is a realization of one or more **REAL WORLD EFFECTS**. These effects may include:
  - ▶ (1) information returned in response to a request for that information,
  - ▶ (2) a change to the shared state of defined entities, or
  - ▶ some combination of (1) and (2).
- ▶ Note, the service consumer in (1) does not typically know how the information is generated, e.g. whether it is extracted from a database or generated dynamically; in (2), it does not typically know how the state change is effected.

## Service (4/4)

---

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

# Dynamics of services

---

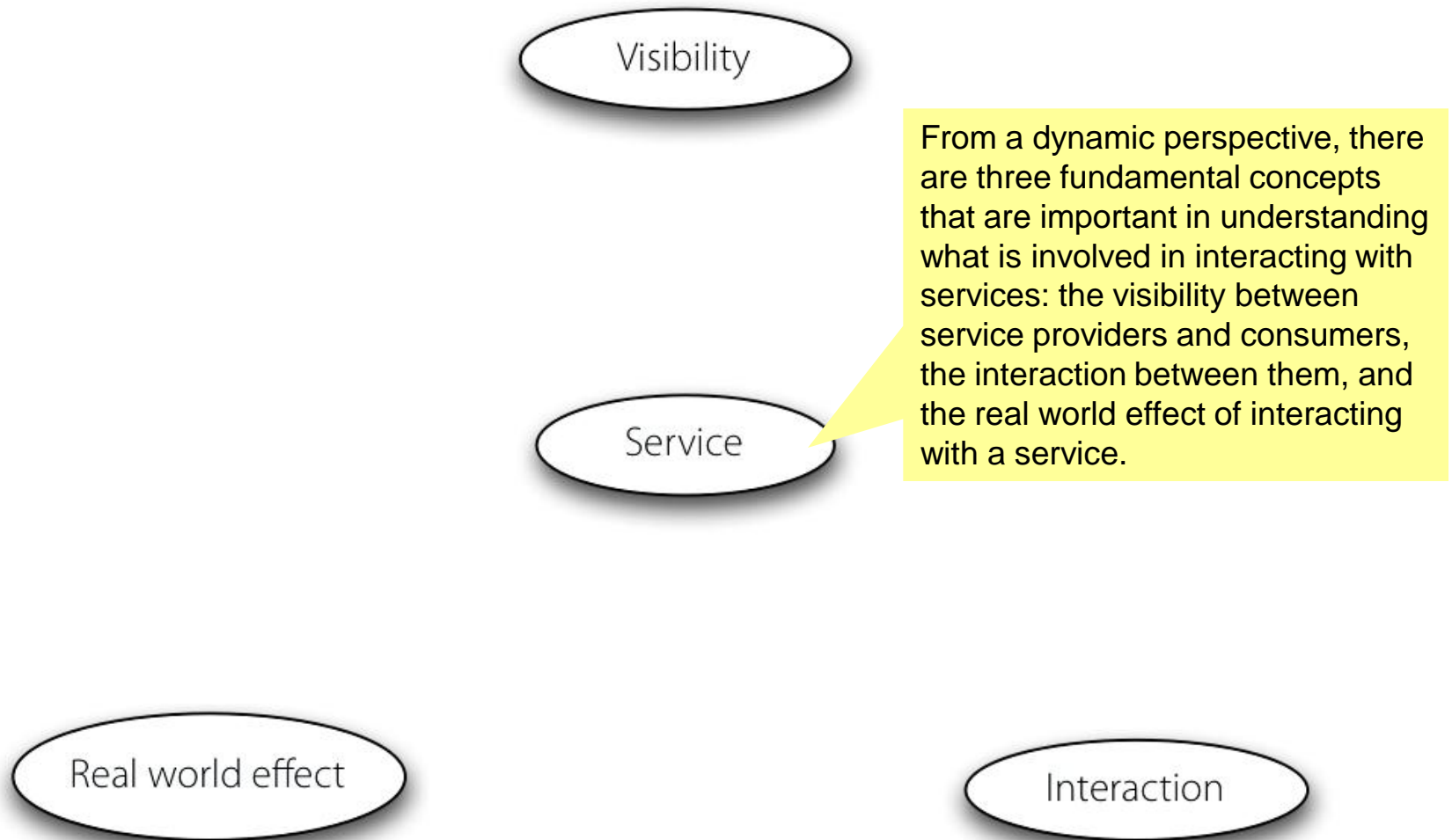


Figure 4 Concepts around the dynamics of service

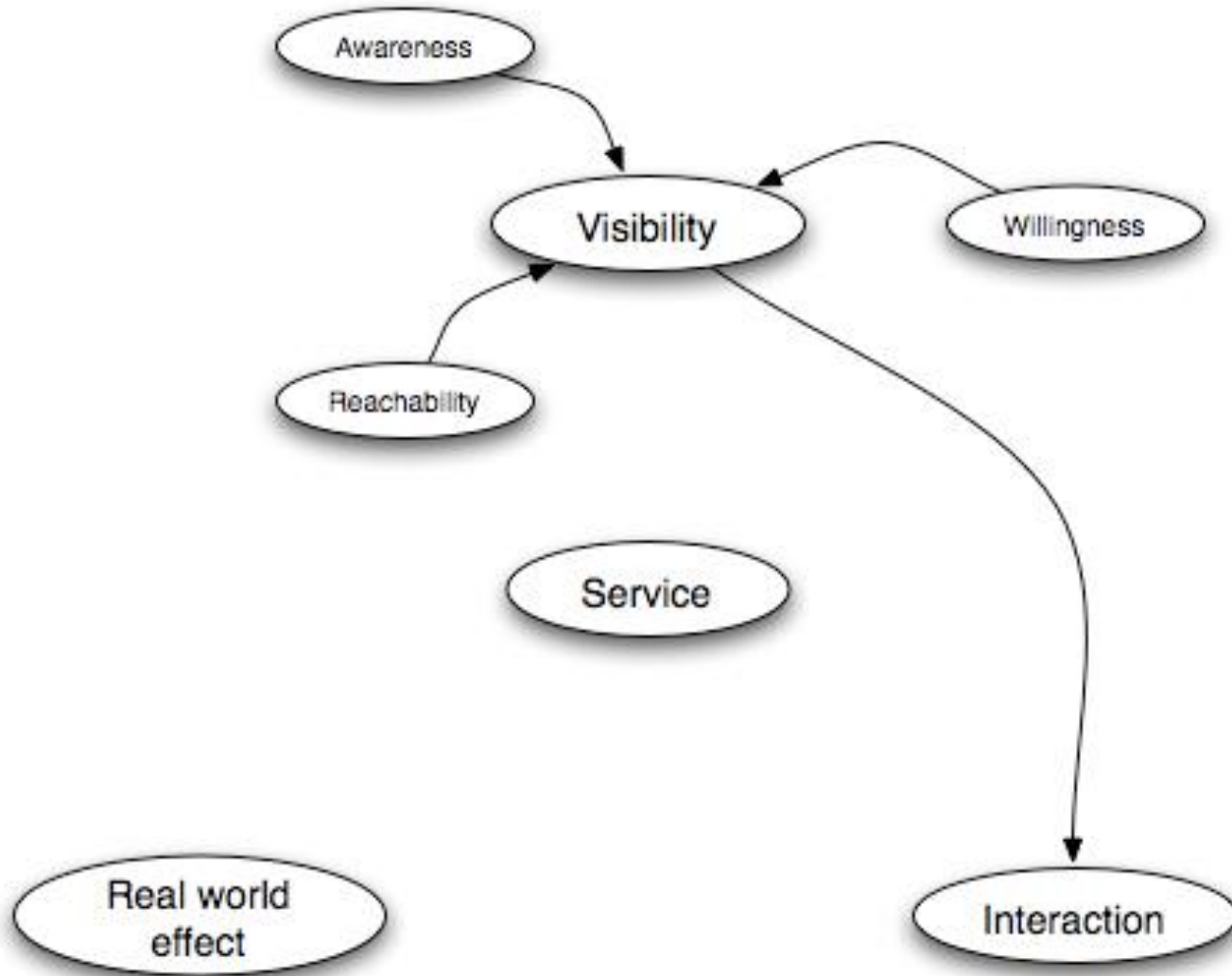
# Visibility

---

- ▶ The capacity for those with needs and those with capabilities to be able to interact with 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**.

# Visibility

---

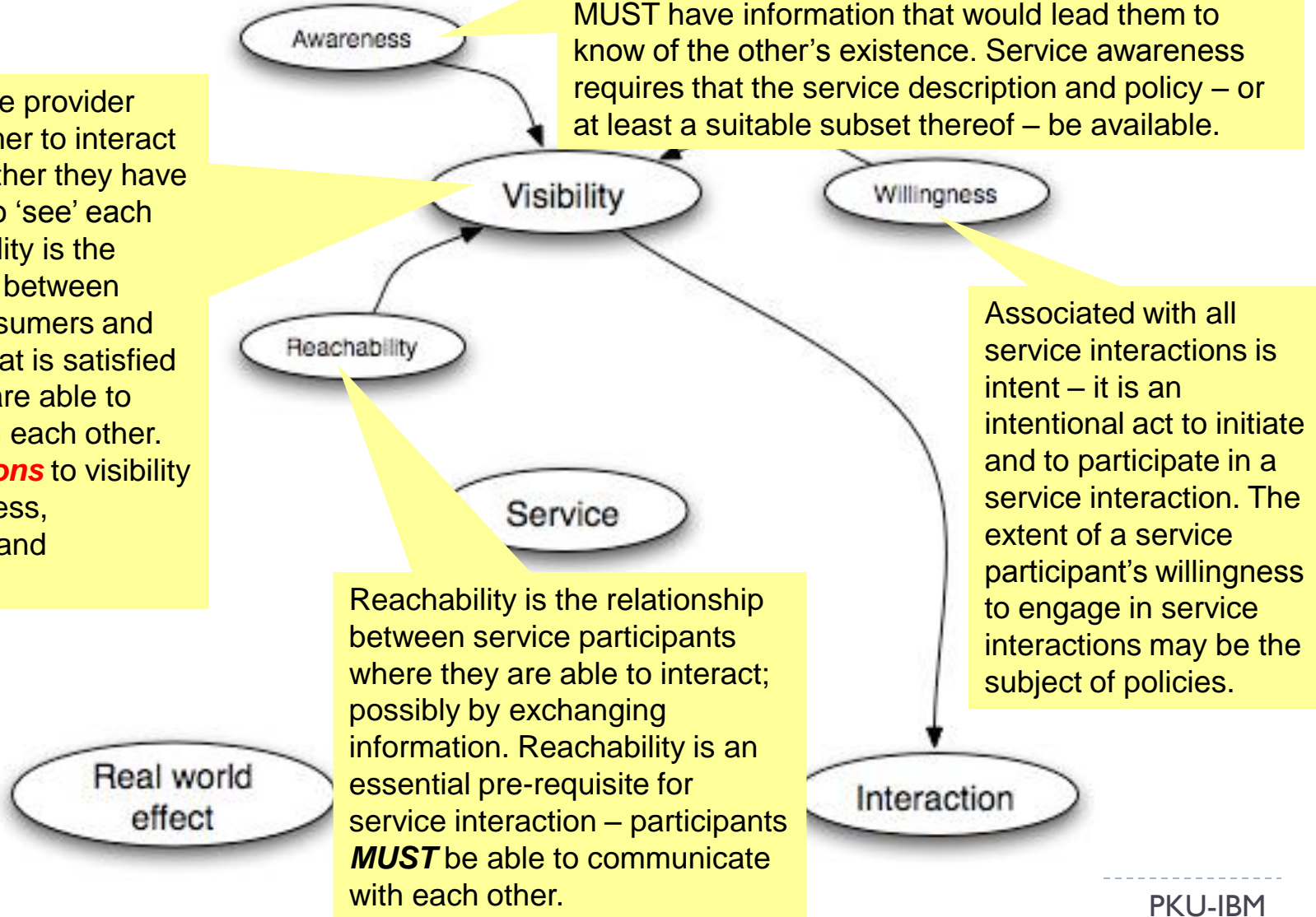


# Visibility

For a service provider and consumer to interact with each other they have to be able to 'see' each other. Visibility is the relationship between service consumers and providers that is satisfied when they are able to interact with each other.

**Preconditions** to visibility are awareness, willingness and reachability.

Both the service provider and the service consumer **MUST** have information that would lead them to know of the other's existence. Service awareness requires that the service description and policy – or at least a suitable subset thereof – be available.

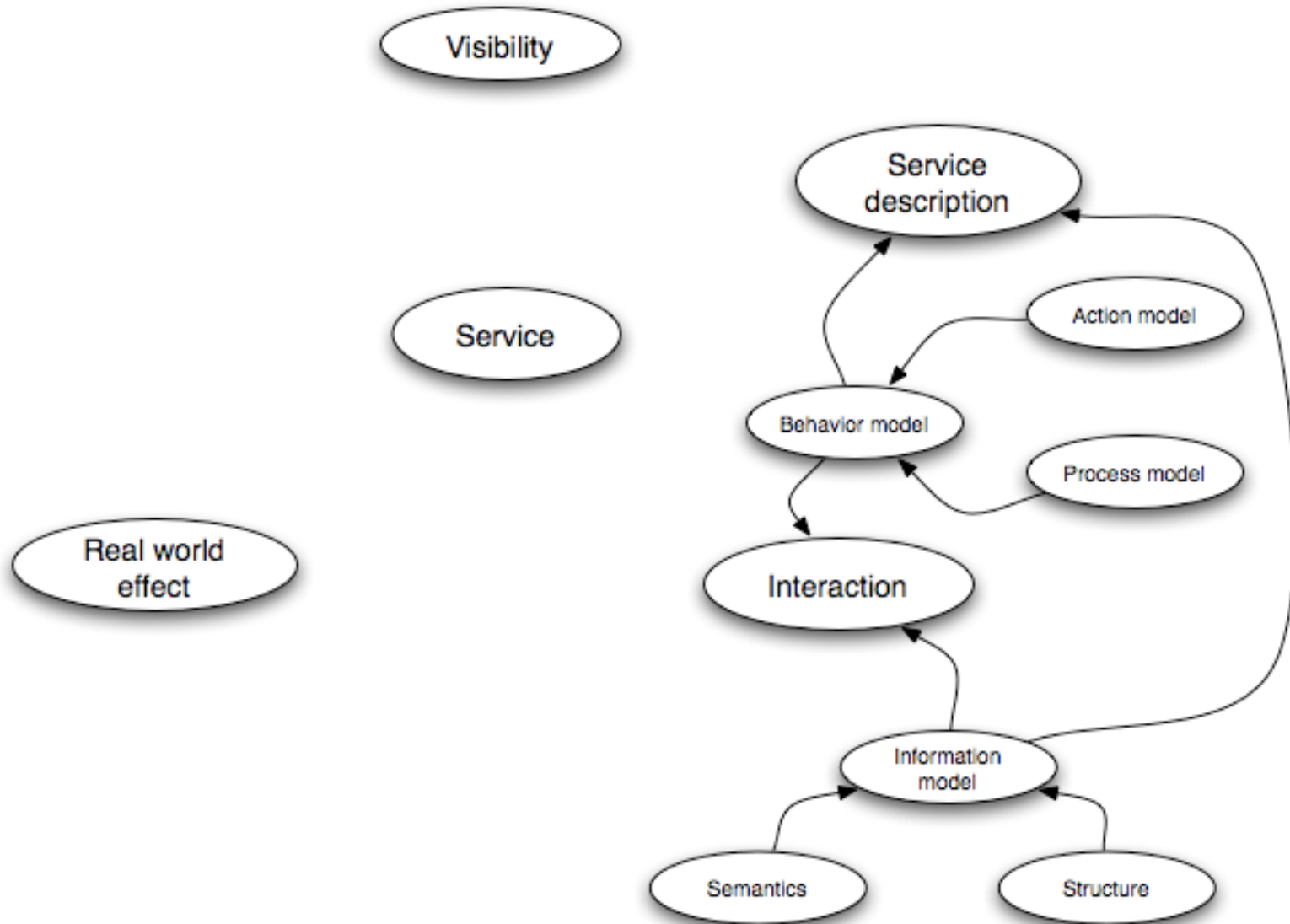


# *Interaction*

---

- ▶ Refers to the interaction between service providers and consumers.
- ▶ Typically mediated **by the exchange of messages**, an interaction proceeds through a series of information exchanges and invoked actions.
- ▶ The result of an interaction is a real world effect.

# Interacting with services





# Interacting with services

Visibility

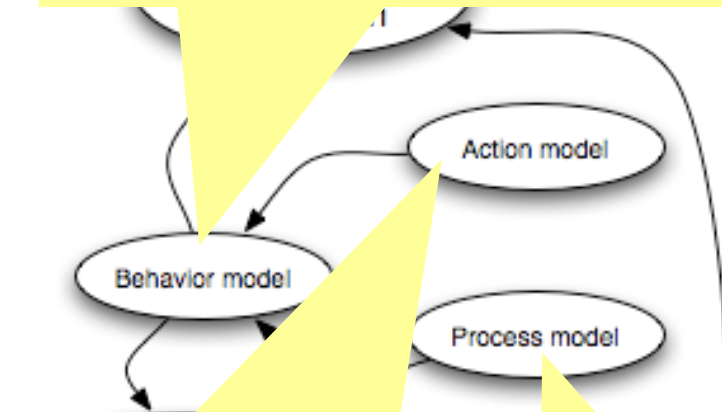
The formal descriptions of terms and the relationships between them (e.g., an ontology) provides a firm basis for selecting correct interpretations for elements of information exchanged.

Knowing the representation, structure, and form of information required is a key in ensuring effective interactions with a service.

The information model of a service is a characterization of the information that may be exchanged with the service.

Interacting with a service involves performing actions against the service. In many cases, this is accomplished by sending and receiving messages. Key concepts that are important in understanding what it is involved in interacting with services revolve around the service description – which references an information model and a behavior model.

The second key requirement for successful interactions with services is knowledge of the actions invoked against the service and the process or temporal aspects of interacting with the service.



The action model of a service is a characterization of the actions that may be invoked against the service.

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

Semantics

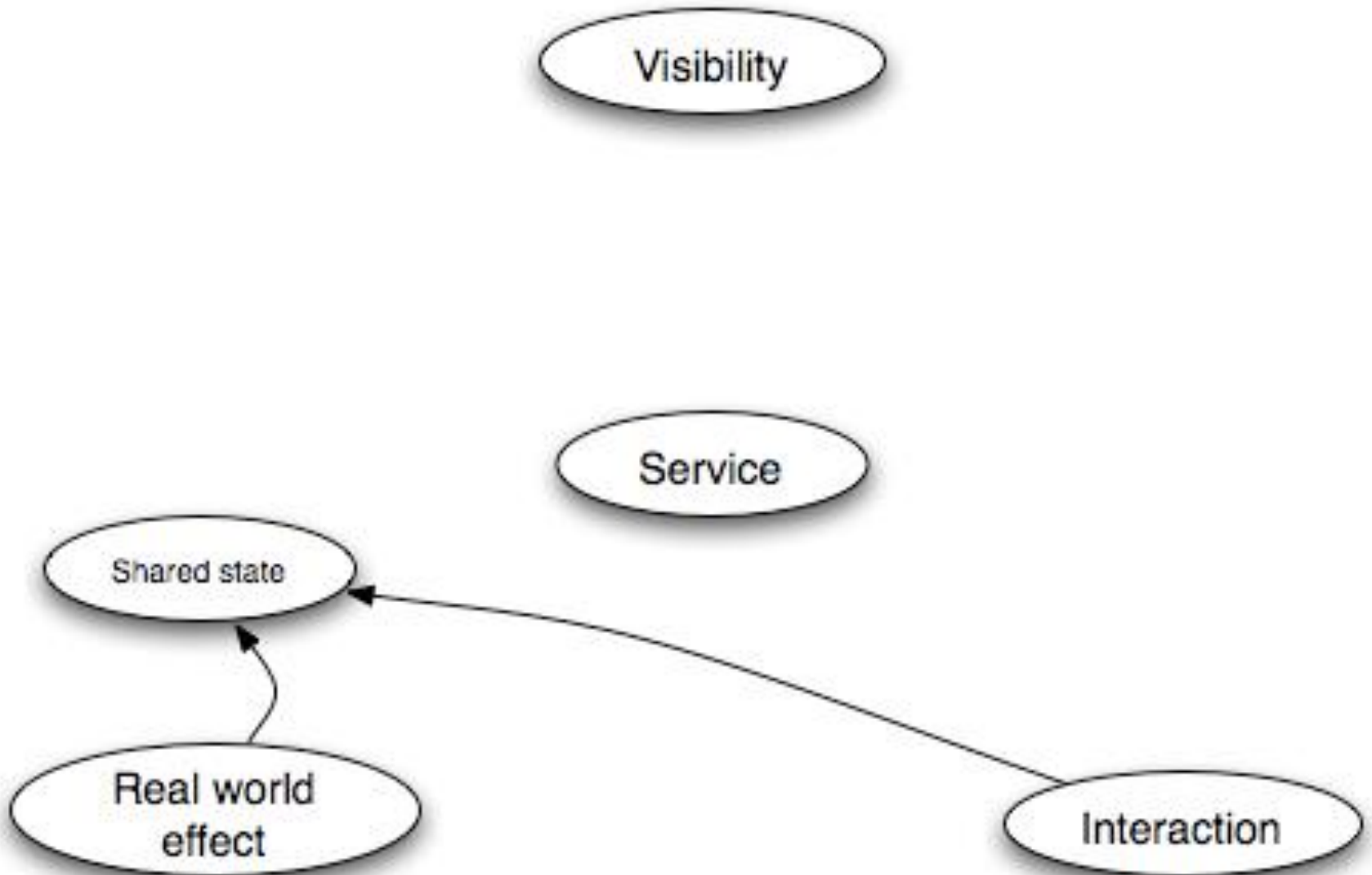
Structure

## *Real World Effect*

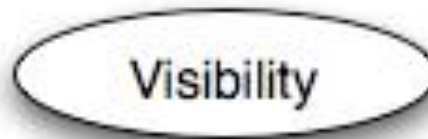
---

- ▶ The actual result of using a service.
- ▶ This may be **the return of information** or **the change in the state of entities** (known or unknown) that are involved in the interaction.

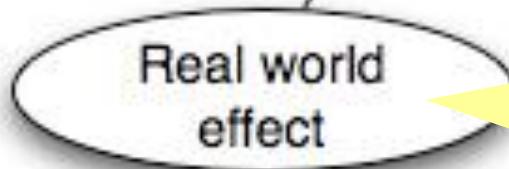
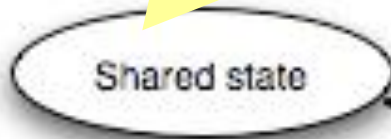
# Real world effect



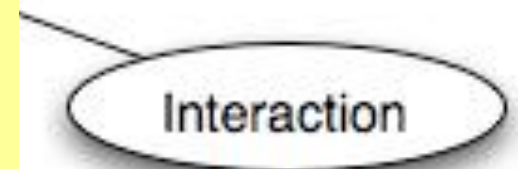
# Real world effect



In this context, the shared state does not necessarily refer to specific state variables being saved in physical storage but rather represents shared information about the affected entities. In addition, the internal actions that service providers and consumers perform as a result of participation in service interactions are, by definition, private and fundamentally unknowable. By unknowable we mean both that external parties cannot see others' private actions and, furthermore, SHOULD NOT have explicit knowledge of them.

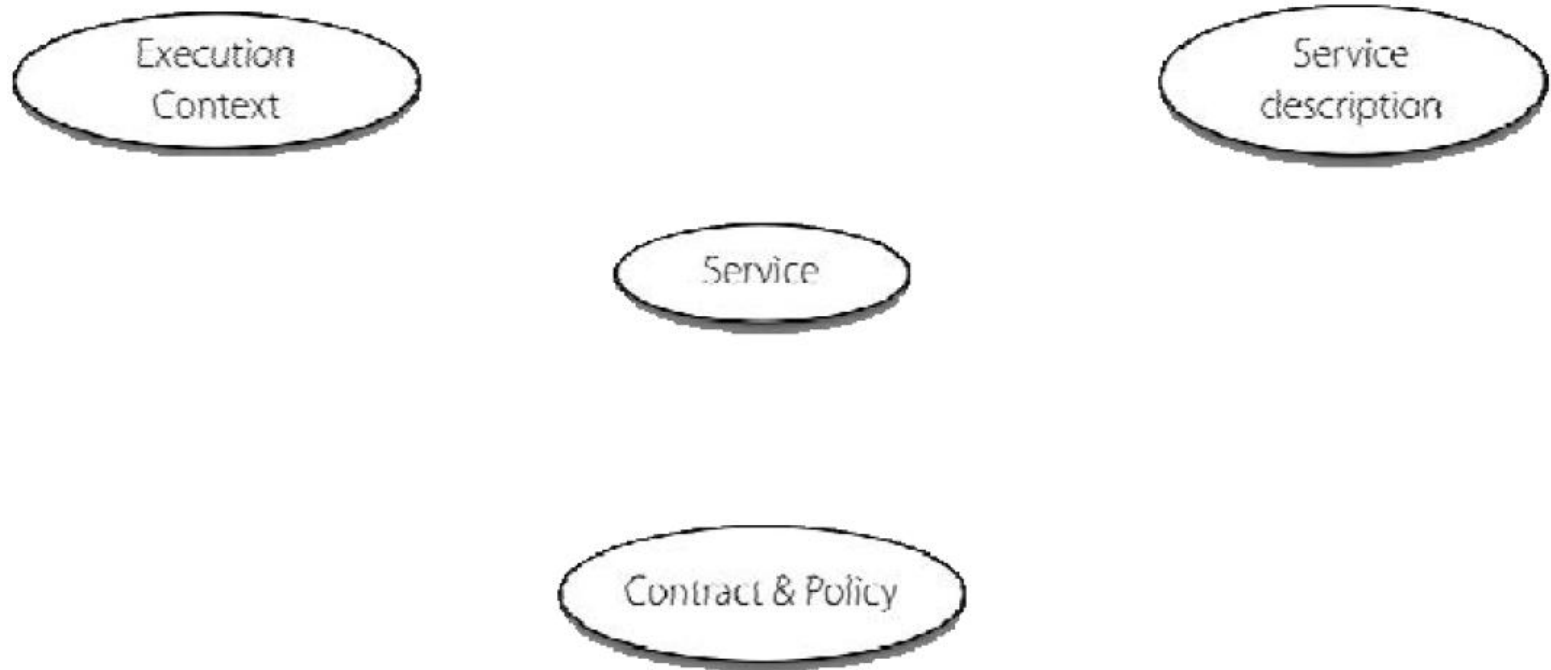


A real world effect can be the response to a request for information or the change in the state of some defined entities shared by the service participants.



# About services

---

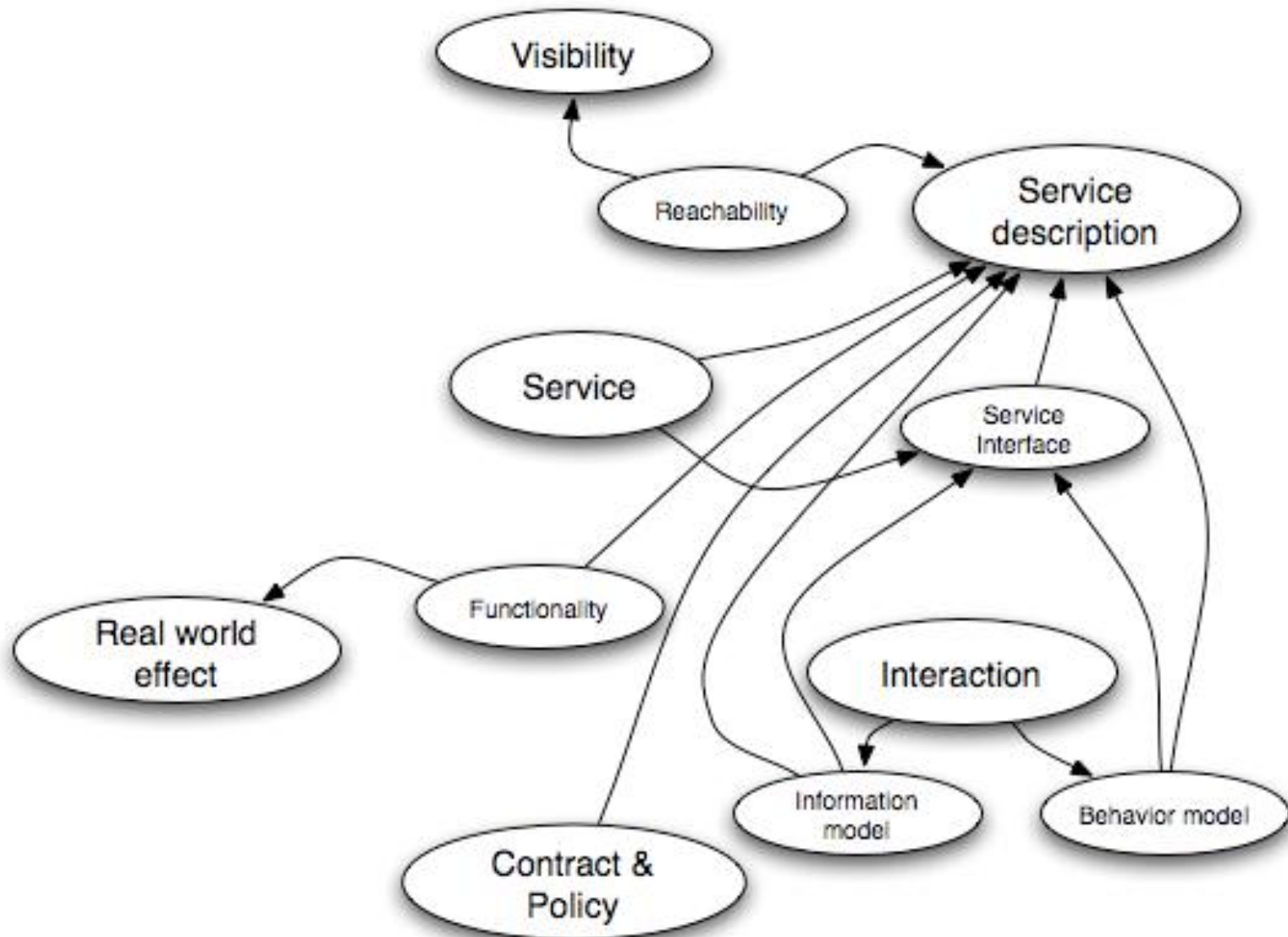


# *Service Description*

---

- ▶ The information needed in order to use, or consider using a service.
- ▶ The purpose of description is to facilitate interaction and visibility,
  - ▶ particularly when the participants are in different ownership domains, between participants in service interactions.

# Service description



# Service description

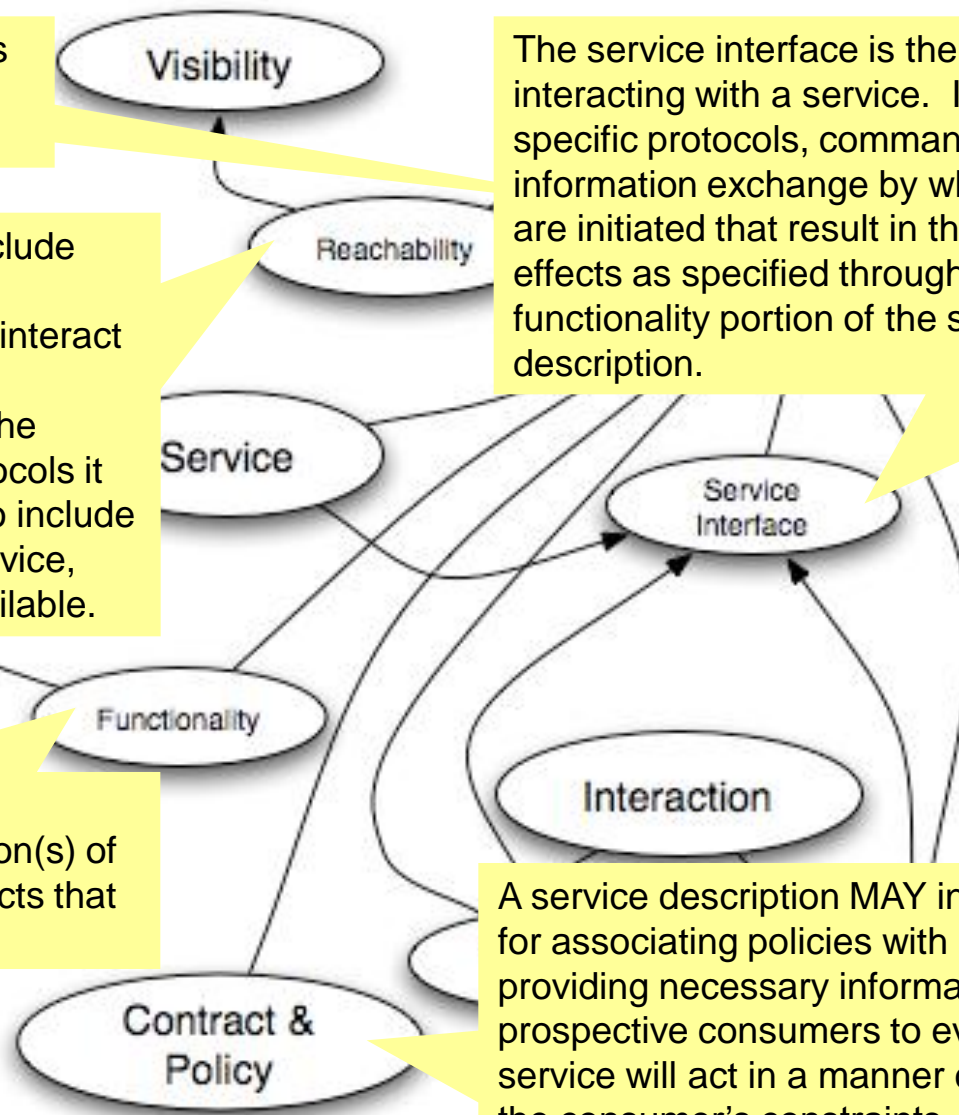
The service description represents the information needed in order to use a service.

A service description SHOULD include sufficient data to enable a service consumer and service provider to interact with each other. This MAY include metadata such as the location of the service and what information protocols it supports and requires. It MAY also include dynamic information about the service, such as whether it is currently available.

A service description SHOULD unambiguously express the function(s) of the service and the real world effects that result from it being invoked.

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

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



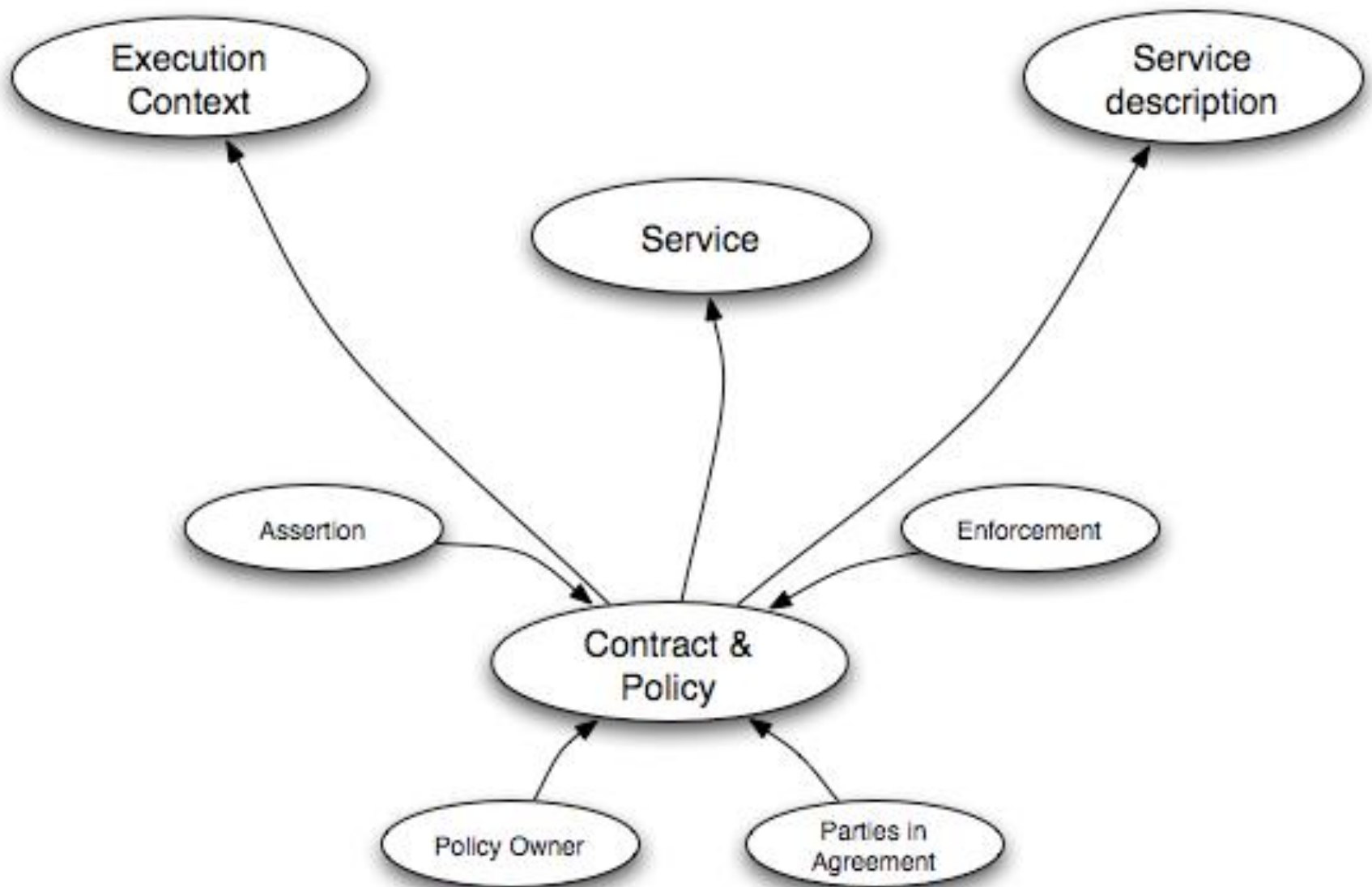


# *Contract & Policy*

---

- ▶ A policy represents some **constraint** or **condition** on the use, deployment or description of an owned entity as defined by any participant
- ▶ A contract represents an **agreement** by two or more parties.

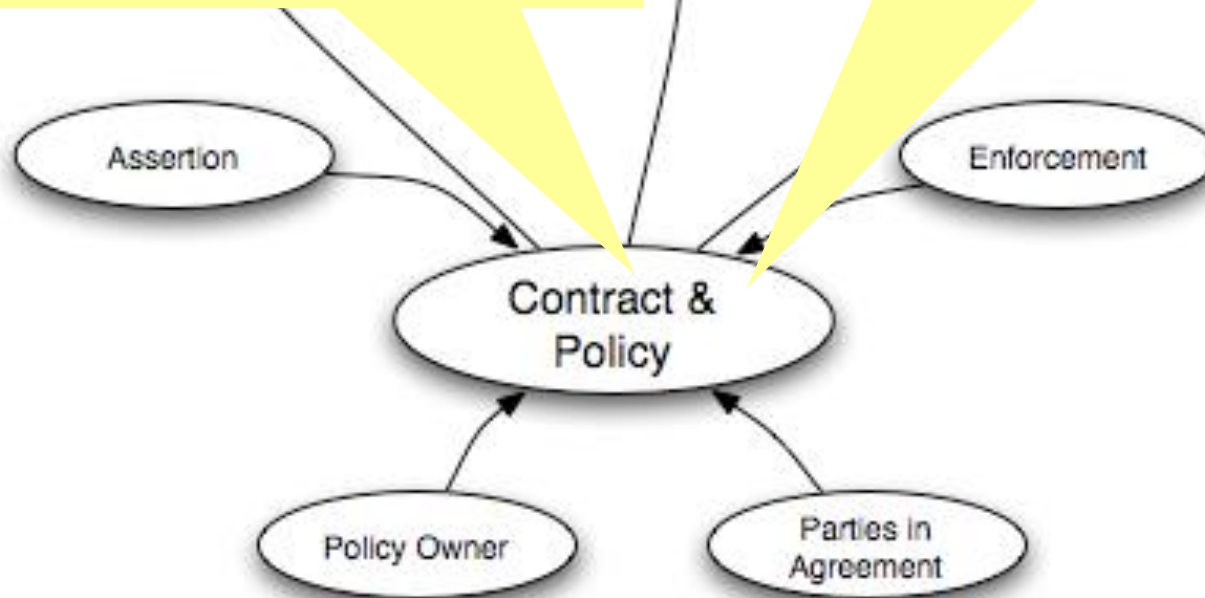
# Policies and contracts



# Policies and contracts

A policy represents some constraint or condition on the use, deployment or description of an owned entity as defined by any participant. A contract, on the other hand, represents an agreement by two or more parties. Conceptually, there are three aspects of policies: the policy assertion, the policy owner (sometimes referred to as the policy subject) and policy enforcement.

Whereas a policy is associated with the point of view of individual participants, a contract represents an agreement between two or more participants. Like policies, contracts can cover a wide range of aspects of services: quality of service agreements, interface and choreography agreements and commercial agreements.

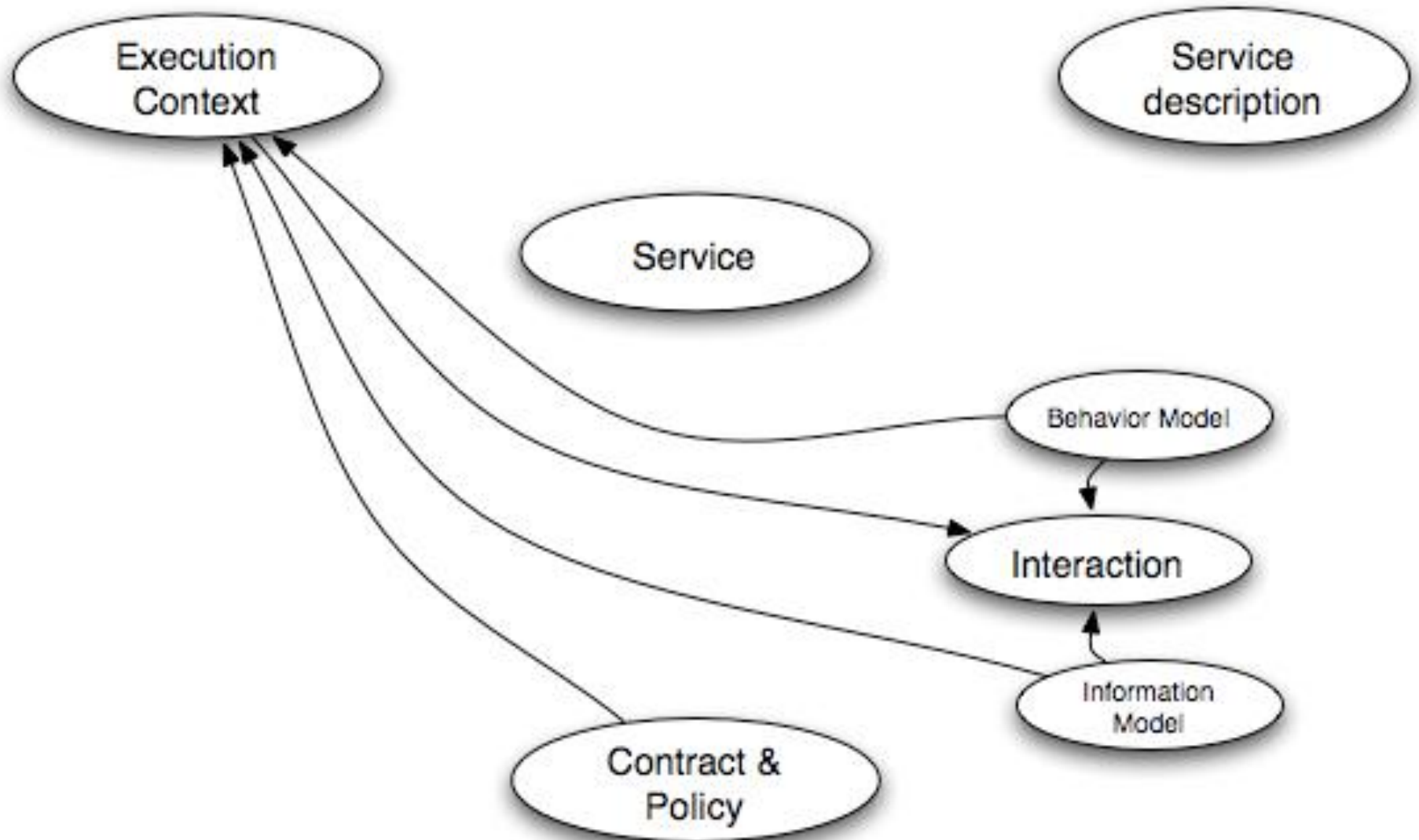


# Execution Context

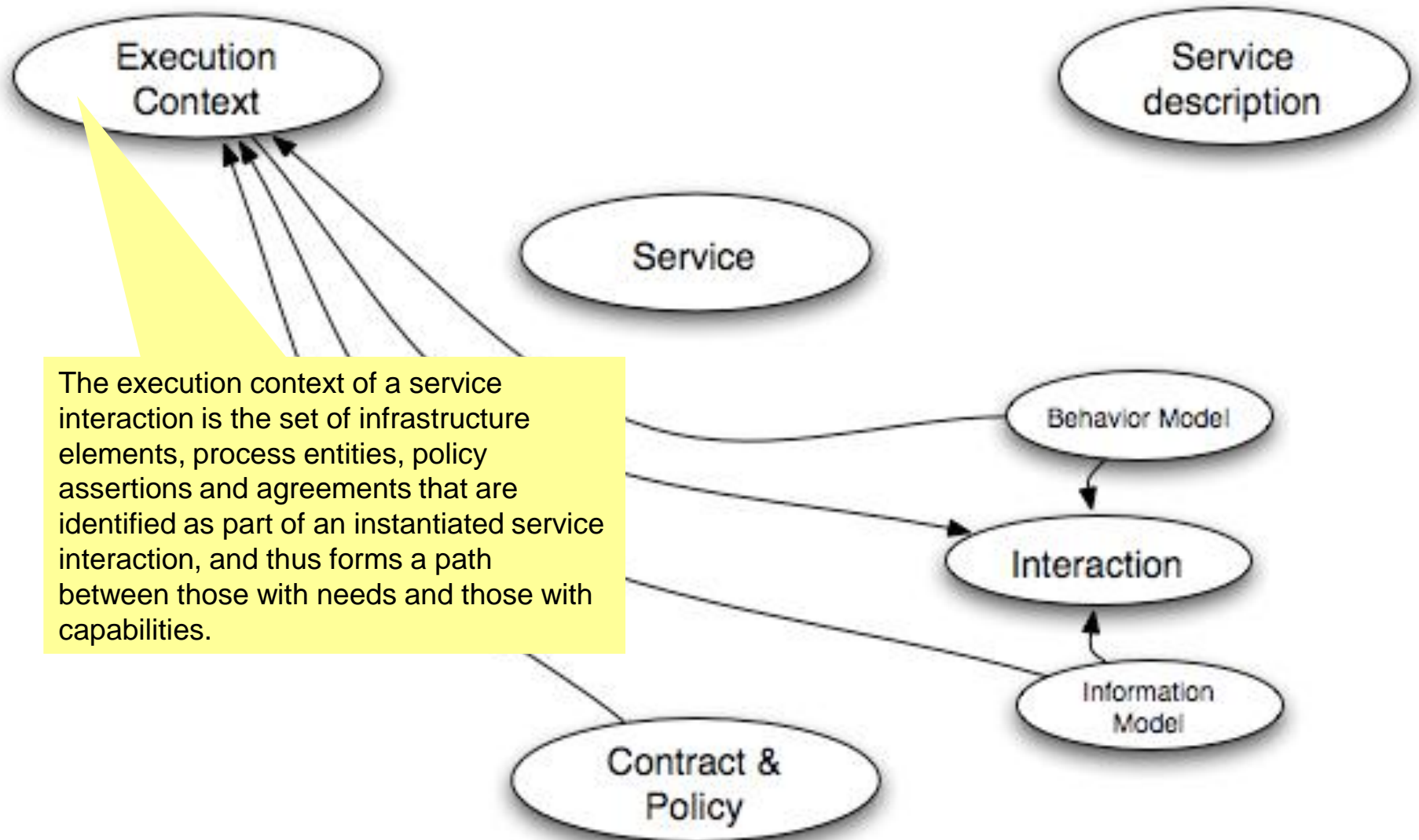
---

- ▶ The set of *technical* and *business elements*
  - ▶ that form a path between those with needs and those with capabilities, and
  - ▶ that permit service providers and consumers to interact.
- ▶ All interactions are grounded in a particular execution context,
  - ▶ which permits service providers and consumers to interact and provides *a decision point* for any policies and contracts that may be in force.

# Execution context



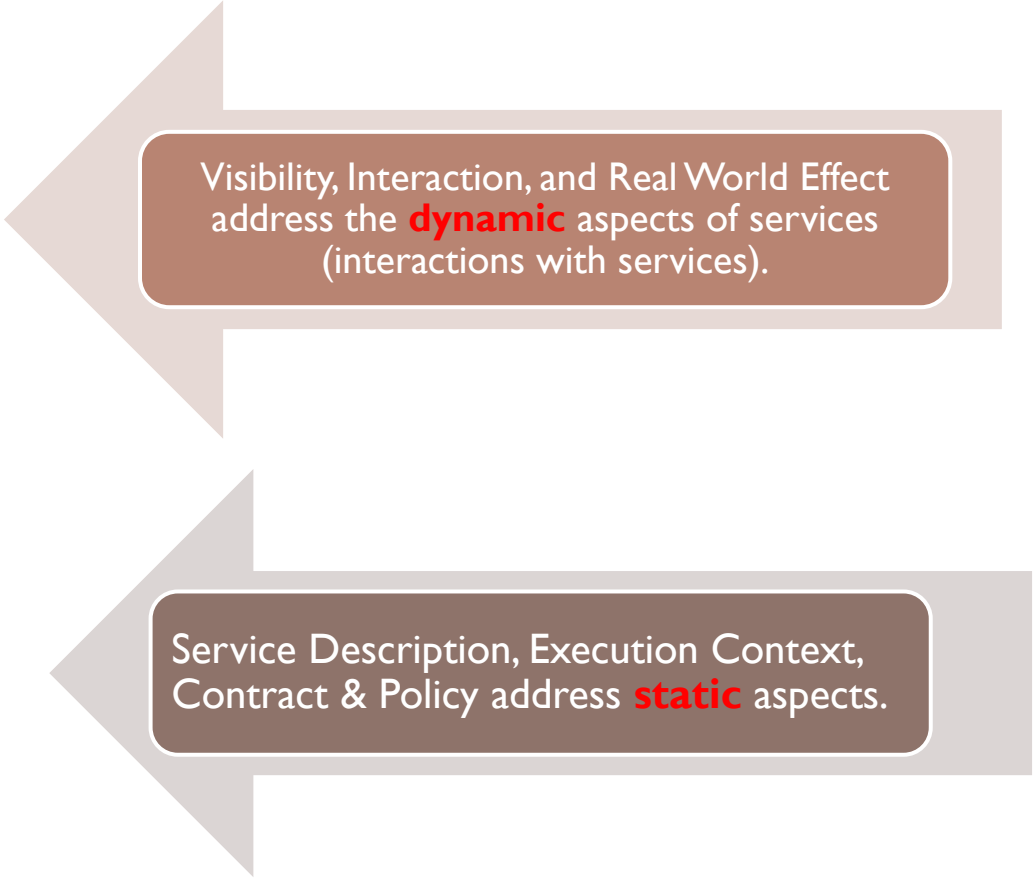
# Execution context



# Principal Concepts of the Reference Model

---

- ▶ Visibility
- ▶ Interaction
- ▶ Real World Effect



Visibility, Interaction, and Real World Effect address the **dynamic** aspects of services (interactions with services).

- ▶ Service Description
- ▶ Execution Context
- ▶ Contract & Policy

Service Description, Execution Context, Contract & Policy address **static** aspects.

# Agenda

---

- ▶ Overview on Reference Model for SOA
- ▶ Introduction
- ▶ Service Oriented Architecture
- ▶ The Reference Model
  - ▶ Service
  - ▶ Dynamics of Services
  - ▶ About services
- ▶ SOA Concepts Example
- ▶ SOA principles
- ▶ Summary
- ▶ References



# SOA Concepts Example (1 / 5)

---

- ▶ An electric utility has the capacity to generate and distribute electricity (*the underlying **capability***).
- ▶ The wiring from the electric company's distribution grid (*the **service***) provides the means to supply electricity to support typical usage for a residential consumer's house (*service **functionality***), and
- ▶ a consumer accesses electricity generated (*the **output** of invoking the service*) via a wall outlet (*service **interface***).

# SOA Concepts Example (2/5)

---

- ▶ In order to use the electricity, a consumer needs to understand
  - ▶ what type of plug to use, what is the voltage of the supply, and possible limits to the load;
  - ▶ the utility presumes that the customer will only connect devices that are compatible with the voltage provided and load supported; and
  - ▶ the consumer in turn assumes that compatible consumer devices can be connected without damage or harm.

*service technical assumptions*

## SOA Concepts Example (3/5)

---

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

## SOA Concepts Example (4 / 5)

---

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

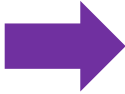
## SOA Concepts Example (5/5)

---

- ▶ In certain situations (for example, excessive demand), a utility may limit supply or institute rolling blackouts (*service **policy***).
- ▶ A consumer might lodge a formal complaint if this occurred frequently (*consumer's implied **policy***).
- ▶ If the utility required every device to be hardwired to its equipment, the underlying capability would still be there but this would be a very different service and have a very different service interface.

# Agenda

---

- ▶ Overview on Reference Model for SOA
- ▶ Introduction
- ▶ Service Oriented Architecture
- ▶ The Reference Model
  - ▶ Service
  - ▶ Dynamics of Services
  - ▶ About services
- ▶ SOA Concepts Example
-  SOA principles
- ▶ Summary
- ▶ References

# SOA principles

---

- ▶ The following **guiding principles** define the ground rules for development, maintenance, and usage of the SOA:
  - ▶ Reuse, granularity, modularity, composability, componentization, and interoperability
  - ▶ Compliance to standards (both common and industry-specific)
  - ▶ Services identification and categorization, provisioning and delivery, and monitoring and tracking

# Architectural principles (1 / 5)

---

- ▶ The following **specific architectural principles** for design and service definition focus on specific themes that influence the intrinsic behaviour of a system and the style of its design.
  - ▶ Service encapsulation
  - ▶ Service loose coupling
  - ▶ Service contract
  - ▶ Service abstraction
  - ▶ Service reusability
  - ▶ Service composability
  - ▶ Service autonomy
  - ▶ Service optimization
  - ▶ Service discoverability



# Architectural principles (2/5)

---

- ▶ **Service encapsulation** - Many web-services are *consolidated* to be used under the SOA Architecture.
  - ▶ Often such services have not been planned to be under SOA.
- ▶ **Service loose coupling** - Services maintain a relationship that minimizes *dependencies* and only requires that they maintain an *awareness* of each other

# Architectural principles (3/5)

---

- ▶ **Service contract** - Services adhere to a **communications agreement**, as defined collectively by one or more service description documents
- ▶ **Service abstraction** - Beyond what is described in the service contract, services **hide logic** from the outside world

# Architectural principles (4/5)

---

- ▶ **Service reusability** - Logic is divided into services with the intention of promoting reuse
- ▶ **Service composability** - Collections of services can be *coordinated and assembled* to form composite services
- ▶ **Service autonomy** – Services have control over the logic they encapsulate

# Architectural principles (5/5)

---

- ▶ **Service optimization** – All else equal, high-quality services are generally considered preferable to low-quality ones
- ▶ **Service discoverability** – Services are designed to be outwardly descriptive so that they can be found and assessed via available discovery mechanisms

# SOA Reference Model - Why?

---

- ▶ SOA itself is used in multiple contexts within the software industry with confusing, differing and even conflicting definitions.
- ▶ If SOA is architecture, as the name implies, how can it be defined and what makes it different from other architectures?
- ▶ How can SOA be described in an architectural manner that is abstract of all implementations?

# Web services approach to a service-oriented architecture (1 / 2)

---

- ▶ Web services can be used to implement a service-oriented architecture.
- ▶ A major focus of Web services is to make functional building blocks accessible over **standard Internet** protocols that are independent from platforms and programming languages.
- ▶ These services can be new applications or just wrapped around existing legacy systems to make them network-enabled.


# Web services approach to a service-oriented architecture (2/2)

---

- ▶ Each SOA building block can play one or more of three roles:
  - ▶ Service **provider** creates a Web service and possibly publishes its interface and access information to the service registry.
  - ▶ Service **broker**, also known as service registry, is responsible for making the Web service interface and implementation access information available to any potential service requestor.
  - ▶ Service **requestor** or Web service client locates entries in the broker registry using various find operations and then binds to the service provider in order to invoke one of its Web services.

# Agenda

---

- ▶ Overview on Reference Model for SOA
- ▶ Introduction
- ▶ Service Oriented Architecture
- ▶ The Reference Model
  - ▶ Service
  - ▶ Dynamics of Services
  - ▶ About services
- ▶ SOA Concepts Example
- ▶ SOA principles
-  Summary
- ▶ References



# Summary

---

- ▶ Any design for a system that adopts the SOA approach will
  - ▶ Have entities that can be identified as services as defined by this Reference Model;
  - ▶ Be able to identify how visibility is established between service providers and consumers;
  - ▶ Be able to identify how interaction is mediated;
  - ▶ Be able to identify how the effect of using services is understood;
  - ▶ Have descriptions associated with services;
  - ▶ Be able to identify the execution context required to support interaction; and
  - ▶ It will be possible to identify how policies are handled and how contracts may be modeled and enforced.

# References

---

- ▶ Dion Hinchcliffe Is Web 2.0 The Global SOA?, SOA Web Services Journal, 28 October 2005
- ▶ Schroth, Christoph ; Janner, Till; (2007). "Web 2.0 and SOA: Converging Concepts Enabling the Internet of Services". . IT Professional 9 (2007), Nr. 3, p. 36-41, IEEE Computer Society Retrieved on 2008-02-23.
- ▶ Jason Bloomberg Mashups and SOBAs: Which is the Tail and Which is the Dog?, Zapthink
- ▶ What Is Web 2.0. Tim O'Reilly (2005-09-30). Retrieved on 2008-06-10.
- ▶ Schroth, Christoph ; Janner, Till; (2007). "Web 2.0 and SOA: Converging Concepts Enabling the Internet of Services". . IT Professional 9 (2007), Nr. 3, p. 36-41, IEEE Computer Society Retrieved on 2008-02-23.
- ▶ Schroth, Christoph ; Janner, Till; (2007). "Web 2.0 and SOA: Converging Concepts Enabling the Internet of Services". . IT Professional 9 (2007), Nr. 3, p. 36-41, IEEE Computer Society Retrieved on 2008-02-23.
- ▶ Ruggaber, Rainer; (2007). "Internet of Services—A SAP Research Vision". . IEEE Computer Society Retrieved on 2008-02-23.
- ▶ Paul Krill Make way for SOA 2.0, InfoWorld , May 17, 2006
- ▶ Yefim Natis & Roy Schulte Advanced SOA for Advanced Enterprise Projects, Gartner, July 13, 2006
- ▶ Joe McKendrick Anti-SOA 2.0 petition nears 400, ZDNet.com, June 29, 2006