Semantic Web Services

Service Science

MOTIVATION

Outline

- Motivation
- Technical Solution
  - What is a Service?
  - What is Service Science?
  - Service Oriented Architecture
  - Semantic Enables Service Oriented Architecture
- Examples
  - Service science in action
- Summary
- References
Motivation

- Service sector vs Industrial sector vs agriculture sector
  - Service sector is becoming more important than industrial sector
  - Products today have a higher service component than in previous decades
- The current list of Fortune 500 companies contains more service companies and fewer manufacturers than in previous decades.


The Rise of the Service Economy

- Services have a growing share in any of the big countries
- US or Germany have a long tradition in service economy
- China or India have switched to services more into the last century
- In most countries, services represent more than 50% of the overall economy
- Service economy is growing due to various reasons:
  - substituting technology for human labor in many agricultural and manufacturing process
  - labor is moved to low cost locations (countries) which makes allows countries that outsource production to focus on services.

IBM WW Revenue

- 1995: Total $71,940
- 2000: Total $88,366
- 2007: Total $98,760

Source: IBM Annual Reports
Why Service Science?

- Services are becoming more important than any other sector of the economy
- Economies of the world shifting from agriculture and manufacturing to services: The Rise of the Service Economy
- A new discipline dedicated to the study, design, and implementation of service systems is needed - Service Science

WHAT IS A SERVICE?

Services

- The word service is used in several contexts:
  - Communication Service
  - Ticket Reservation Service
  - Transport Service
  - Information Service
  - Finance Service
  - E-government Service
  - ...

But what is a Service?
What is a service?

Main Entry: service

Function: noun

Etymology: Middle English, from Anglo-French, from Latin servitium, from servus slave

1 a: the occupation or function of serving in active service b: employment as a servant c: entered his service

2 a: the work performed by one that serves <good service> b: benefit c: glad to be of service d: contribution to the welfare of others e: disposed to do or yield the service at your service

3 a: a form followed in worship or in a religious ceremony b: a meeting for worship —often used in plural c: the act of serving d: a helpful act d: useful labor that does not produce a tangible commodity —usually used in plural e: charge for professional services f: a set of articles for a particular use (as silver tea service) g: an administrative division (as of a government or business) h: the consular service i: one of a nation's military forces (as the army or navy) j: a facility supplying some public demand k: a facility providing maintenance and repair l: the materials (as spun yarn, small lines, or canvas) used for serving a rope m: the act of bringing a legal writ, process, or summons to notice as prescribed by law n: the act of a male animal copulating with a female animal o: a branch of a hospital medical staff devoted to a particular specialty p: obstetrical service

Merriam-Webster Online, http://www.m-w.com

For different people the term Service has different meaning

In Business and Economics

– a service is seen as a business activity that often results in intangible outcomes or benefits
– a service is the non-material equivalent of a good.

Service provision has been defined as an economic activity that does not result in ownership, and this is what differentiates it from providing physical goods.

– a process that creates benefits by facilitating either a change in customers, a change in their physical possessions, or a change in their intangible assets.

In Computer Science

– the terms service and Web service are often regarded as interchangeable to name a software entity accessible over the Internet.
– a (Web) service is seen software system designed to support interoperable machine-to-machine interaction over a network.

Service vs. Web Service

– Service
  – A provision of value in some domain (not necessarily monetary, independent of how service provider and requestor interact)

– Web Service
  – Computational entity accessible over the Internet (using Web Service Standards & Protocols), provides access to {concrete} services for the clients.
Service vs. Web Service

- **Service** - A provision of value in some domain
- **Example of a Service:**
  - Let’s consider a user who wants to book a train ticket from Innsbruck to Munich on a given date. The service he is looking for, is the provision of a train ticket with the specified constraints. Such provision is independent on how the supplier and the provider interact, i.e., it does not matter whether the requester goes to a train tickets office or uses the train Web site to book his trip.

Web Service vs. Web Service

- **Web Service** - Computational entity accessible over the, provides access to (concrete) services for the clients.
- **Example of a Web Service:**
  - a railway company might provide a software component accessible via Web service standards, i.e., a Web service to request the booking of a trip. Thus, the Web service is an electronic means by which a client is able to request a specific service from a provider, but not the service itself.

Web Service properties

- **Functional**
  - contains the formal specification of what exactly the service can do.

- **Behavioral**
  - how the functionality of the service can be achieved in terms of interaction with the service and as well in terms of functionality required from the other Web services.

- **Non-functional properties**
  - captures constraints over the previous mentioned properties

Web Service related tasks

- **Discovery**: “Find services that matches to the service requester specification”.
- **Selection and Ranking**: “Choose the most appropriate services among the available ones”.
- **Composition**: “Assembly of services based in order to achieve a given goal and provide a higher order of functionality”.
- **Mediation**: “Solve mismatches among domain knowledge used to describe the services, protocols used in the communication, data exchanged in the interaction (types used, and meaning of the information) and business models of the different parties”.
- **Execution**: “Invocation of a concrete set of services, arranged in a particular way following programmatic conventions that realizes a given task”.
- **Monitoring**: “Supervision of the correct execution of services and dealing with exceptions thrown by composed services or the composition workflow itself”.
- **Handover**: “Replacement of services by equivalent ones, which solely or in combination can realize the same functionality as the replaced one, in case of failure while execution”.
WHAT IS SERVICE SCIENCE?

Definition

• “Service Science, Management and Engineering (SSME) is a new multi-disciplinary research and academic effort that integrates aspects of established fields such as computer science, operations research, engineering, management sciences, business strategy, social and cognitive sciences, and legal sciences.”
  IBM's definition

• “Service Science, Management, and Engineering (SSME) is an interdisciplinary approach to the study, design, and implementation of service systems – complex systems in which specific arrangements of people and technologies take actions that provide value for others.”
  Wikipedia's definition

Why Service Science?

• Driven by a business environment that includes advanced telecommunications, accelerated business globalization, and rapid technology innovations, emphasis in service has evolved from a traditional labor-based business to sources of innovation, collaboration, and value co-creation.

• However, the focus shift to service has created a research and education gap due to the complexity of inter-disciplinary issues across service business strategy and modeling, operations research, information technologies, industrial engineering, management science, social and cognitive science, work force management, and legal science, etc.

The Goal of Service Science

• To provide concepts, methods and techniques to understand and engineer service based systems

• To ensure the social benefit of service based systems

• Ultimately to provide a sound and complete theory for bringing Service Economy at global scale through IT

[Paul P. Maglio et al. Steps Toward a Science of Service Systems]
Service Science: multidisciplinary

- A multidisciplinary science influenced by
  - Computer science
  - Cognitive science
  - Economics
  - Organizational behavior
  - Marketing
  - Operations research
  - Policy and Law
  - …

Towards a theory of service systems

- A general theory of service systems should consist of:
  - **Science** – what service systems are and how to understand their evolution
  - **Management** – how to invest to improve service systems
  - **Engineering** – how to invent new technologies that improve the scaling of service systems


Service Science References

- Service Science is recognized as a very important emerging science:
  - **IBM**: Service Science, Management and Engineering
    http://www.ibm.com/university/issme
  - **HP**: Center for Systems and Service Sciences
    http://www.services-sciences.org/
  - **Oracle**: Service Research and Innovation Initiative
    http://www.thesrii.org/

SERVICE ORIENTED ARCHITECTURE
In this part we discuss:

- What is Service Oriented Architecture (SOA)?
- SOA Principles
- SOA Properties – Self-*
- SOA Benefits

What is Service Oriented Architecture (SOA)?


- "A service-oriented architecture is essentially a collection of services. These services communicate with each other. The communication can involve either simple data passing or it could involve two or more services coordinating some activity. Some means of connecting services to each other is needed."[1]


- "Service-oriented architecture (SOA) provides methods for systems development and integration where systems group functionality around business processes and package these as interoperable services. An SOA infrastructure allows different applications to exchange data with one another as they participate in business processes. SOA separates functions into distinct units, or services, which developers make accessible over a network in order that users can combine and reuse them in the production of business applications ."

Wikipedia

http://www.service-architecture.com/web-services/articles/ service-oriented_architecture_soa_definition.html

SOA Actors

- Service Provider
  - From a business perspective, this is the owner of the service. From an architectural perspective, this is the platform that provides access to the service.

- Service Registry
  - This is an information space of service descriptions where service providers publish their services and service requesters find services and obtain binding information for services.
  - Allows service consumers to locate service providers that meet required criteria

- Service Consumer
  - From a business perspective, this is the business that requires certain function to be fulfilled. From an architectural perspective, this is the client application that is looking for and eventually invoking a service.
SOA Principles

• Formal contract
• Loose coupling
• Abstraction
• Reusability
• Autonomy
• Statelessness
• Discoverability
• Composability


SOA Principles – Formal contract

• According to SOA Formal contract principle every service needs to have an official, standardized, formal contract.
• A great deal of emphasis is placed on specific aspects of contract design, including:
  – the manner in which services express functionality (functional description contract)
  – how data types and data models are defined (information model)
  – how policies are asserted and attached. (non-functional description contract)
  – how interaction with the service is to be performed (behavioral contract)

SOA Principles – Loose coupling

• SOA is a loosely coupled arrangement of services and service consumers. At design time, loose coupling means that services are designed with no affinity to any particular service consumer. Inside the service, no information is assumed as to the purpose, technical nature or business nature of the service consumer. Thus, a service is fully decoupled from a service consumer.
• However, the service consumer is dependent on the service (that is, it embeds literal references to service interfaces). Thus, SOA is a semi-coupled (or loosely coupled) architecture. It differs from an event-driven architecture, in which all participating software components are decoupled from others, and also from a monolithic architecture, in which all software components are designed to operate only in the initially intended context (that is, logically tightly coupled).
• Design-time loose coupling is essential to SOA because it enables the non-intrusive reuse of service interfaces. However, tools can’t guarantee design-time loose coupling. Poorly designed services, which are logically locked into their service consumers, may render the entire application monolithic — despite the use of SOA-style technologies.

"Introduction to Service-Oriented Architecture", Yefim V. Natis, Roy W. Schulte, 14 April 2003

SOA Principles – Abstraction

• This principle emphasizes the need to hide as much of the underlying details of a service as possible.
• By using abstraction previously described loosely coupled relationship is directly enabled and preserved
• There are 4 levels of abstraction in SOA as suggested in [4]:
  – technology abstraction
  – functional abstraction
  – programming logic abstraction
  – quality of service abstraction
SOA Principles – Reusability

• The reusability principle suggests to contain and express agnostic logic as services that can be positioned as reusable enterprise resources.

• Reusability will:
  - Allow for service logic to be repeatedly leveraged over time so as to achieve a high ROI.
  - Increase business agility on an organizational level.
  - Enable the creation of service inventories that can be easily integrated and used in various use-cases.


SOA Principles – Autonomy

• SOA Autonomy principle implies that services have control over the solution logic they implement.

• SOA Autonomy/ Service Autonomy can be observed at various levels:
  - Runtime autonomy – represents the amount of control a service has over its execution environment at runtime.
  - Design-time autonomy – represents the amount of governance control a service owner has over the service design.

SOA Principles – Statelessness

• This means a service must do its best to hold onto state information pertaining to an interaction for as small a duration as possible, e.g., do not retain awareness of a message once it is processed.

• Statelessness in a service means that if the service is enlisted in a flow, it doesn’t retain any state referring to the enclosing flow. Form a message perspective, it means that once a service has received and processed a message, it doesn’t retain memory of the passage of that message.

• This helps with concurrent access scaling.

Statelessness in SOA and REST

• SOA and REST share the Statelessness principle.

• REST provides explicit state transitions.

• REST Servers are stateless and messages can be interpreted without examining history.

• Persistent data can be given explicit URIs on the server.

• Messages can refer to persistent data through links to URIs.
Statelessness in SOA and REST

• In SOA
  – Stateless communication although communication can be
    stateful as well
    • Received or sent messages can trigger state change
    • Operations requiring sequence of messages
    – Capable to support transactions
    • set of operations with pass or fail results
    – Tighter coupling between components

• In REST
  – Stateless communication
  • Document transfer only
  • A party is not aware of its partner current state
  • Party receiving information can decide how to process it
  • HTTP caching possible
  – Looser coupling between components

SOA Principles – Discoverability

• SOA Discoverability is meant to help one avoid the accidental creation of
  services that are either redundant or implement logic that is redundant. The
  discoverability principle can be referred to the design of an individual
  service so that it becomes as discoverable as possible – no matter
  whether the discoverability extension or product actually exists in the
  surrounding implementation environment.

• Discovery is a central task in SOA. SOA Discoverability is centered on
  Service Discoverability. Service Discoverability is meant to refer to the
  technology architecture’s ability to provide a mechanism of discovery, for
  example a service directory, service registry or a service search
  engine.

• Services be designed as resources that are highly discoverable in some
  fashion. Each service should be equipped with the metadata that is required
  to properly communicate its capabilities and meaning.

SOA Principles – Composability

• Allow us to chain services together to provide new services
• Composition has the advantage that one can put
  together composite applications at a speed greater than
  writing one from scratch
• Building new services and application becomes quicker
  and cheaper

SOA Properties – Self-* Properties

• Most service architectures aim for „self-*“
  properties to reduce management load by design:
  – Self-Configuration
  – Self-Organization
  – Self-Healing
  – Self-Optimization
  – Self-Protection
Service architectures comprise of a huge amount of different components (services and hardware). Configuration is a challenging task in such environments. The idea of self-configuration is the adoption of the self-organization and fully distributed cooperation capabilities known from groups with cooperative social behavior which collaborate to solve a problem. Every member of the group can decide which part of the problem it can solve and which “QoS” it can provide.

A system is self-organizing if it automatically, dynamically and autonomously adapts itself to achieve global goals more efficiently under changing conditions.

The task of self-healing is to assure that a system meets some defined conditions as far as possible, i.e. to guarantee that all services running in the framework stay available, even in the case of partial outages in the system.

The self-configuration is responsible to find a good distribution of the services in terms of the given resources of the service description. The target of the self-optimization is to distribute the services of the application in a way that the considered resources are utilized evenly. A typical approach is to find an adequate configuration at the beginning and to optimize the application during runtime.
Self-Protection

- **Self-protection** techniques cope with intentionally or unintentionally malicious peers or services in a framework. The behave as the “immune system” of a service framework as they are permissive to good-natured services and messages but can detect appearing malicious events.

SOA Benefits

**Business Benefits**
- Focus on Business Domain solutions
- Leverage Existing Infrastructure
- Agility

**Technical Benefits**
- Loose Coupling
- Autonomous Service
- Location Transparency
- Late Binding

SESA (1)

- Currently, computer science is in the next period of abstraction.
- A generation ago we learnt to abstract from hardware and currently we learn to abstract from software in terms of SERVICE oriented architectures (SOA).
- It is the service that counts for a customer and not the specific software or hardware that is used to implement the service.
- In a later stage, we may even talk in terms of problem-oriented architectures (or more positively expressed in terms of problem solving oriented architectures) because SOAs are biased towards the service provider and not towards the customer that has a problem that needs to be solved.
• Service-oriented architectures will become quickly the leading software paradigm.
• However, SOAs will not scale without significant mechanization of – service discovery, service adaptation, negotiation, service composition, service invocation, and service monitoring; and data and process mediation.
• Therefore, machine processable semantics needs to be added to bring SOAs to their full potential.
• Development of open standards (languages) and open source architectures and tools that add semantics to service descriptions

Three Layers of SESA

• Problem-Solving Layer
  – Turns a service-oriented architecture into a domain specific problem-solving environment
• Common Services Layer
  – The execution environment and the supporting infrastructure that maps the problem descriptions generated at the Problem Solving Layer to the services that can solve the problems
• Resource Layer
  – Covers the deployment and provisioning of physical resource being responsible for the actual execution of the applications

Problem solving layer

• This layer turns a service-oriented architecture into a domain specific problem-solving environment.
• It represents the transparent interface to the user(s), where all computing resources are turned into or expressed as services
• Supports the full set of operations from an e-commerce framework: information negotiation, etc.
• Provides clear separation between business/process logic on one hand and the common service layer

Common Services Layer

• Provides an adaptive execution environment and the supporting infrastructure that maps the problem descriptions generated at the Problem Solving Layer to the services that can solve the problems.
• Existing architectures and standards from Web service and Grid areas (e.g. OGSA, WSRF, WSDL) which operate only at a syntactic level are semantically enriched and integrated into this layer.
• Semantically enrichment of SOAs that implement the Common Service Layer capabilities will help to automate: service discovery, service adaptation, negotiation, service composition, etc.
• This layer could be implemented using the W<Triple> technology
Resource Layer

• Responsible for actual execution of applications.
• All tasks that involve resources such as resource discovery, selection and negotiation for advanced or on-the-fly reservation of resources are supported and implemented in this layer.
• Covers the deployment and provisioning of physical resource (e.g. computers, data servers, and networks, usually connected into a Grid) and logical resources (e.g. application components or common services).
• This layer may rely on two prominent and widely discussed areas that deal with distributed resources in the context of service oriented computing are Ubiquitous Computing and Grid Computing.

W<Triple>

W<Triple> which stands for:
• WSMO: A conceptual model for describing service oriented architectures
• WSML: A formal language for describing service oriented architectures
• WSMX: A service oriented architecture
• Triple space: A shared space for heterogeneous services that communicate via persistent publication

W<Triple>: WSMO

Objectives that a client wants to achieve by using Web Services

Connectors between components with mediation facilities for handling heterogeneities

More in Lecture 7

W<Triple>: WSML

A set of concrete languages for the various tasks:
• Ontology / Rule Languages (static view)
  – WSML Core
  – efficiency and comparability
  – WSML DL
  – decidability, open world semantics
  – WSML Rule
  – efficient existing rule engines
  – WSML Full
  – unifying language, theorem proving
• Languages for dynamics
  – Transaction Logic over ASMs
• Mapping languages
  – for dynamics (process mediation)
  – or data (data mediation)

More in Lecture 8
WSMX: The Web Service EXecution Environment
- A service oriented architecture.
- Reference implementation of SESA and WSMO

More on Lecture 9

Message

Email

Human net

Email:
- Human to Human communication (Human net)
- Dissemination of information was based on message exchange (Message)

Web:
- Human to Human communication (Human net)
- Dissemination of information based on publishing (Publishing)

So-called Web Service

Publishing

Human net

Machine net

Web Services:
- Machine to Machine communication (Machine net)
- Dissemination of information based on messages (Message)
**Triple Space Computing**

**Message Publishing**

<table>
<thead>
<tr>
<th>So-called Web Service</th>
<th>Triple Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>Email</td>
<td>Web</td>
</tr>
</tbody>
</table>

**Machine net**

**Human net**

**Triple Space:**
- Machine to Machine communication (**Machine net**)
- Dissemination of information based on Web principles (**Publishing**)

**Communication platform for Semantic Web services based on Web principles:**

"Persistently publish and read semantic data that is denoted by unique identifiers"

**Fundamentals:**
- Space-based computing – sharing information, knowledge
- RDF triples of the form: <subject, predicate, object>
- URI – Uniform Resource Identifier

**Triple Spaces**

- Time autonomy
- Location autonomy
- Reference autonomy
- Vocabulary autonomy

**Triple Spaces provide a communication paradigm for anonymous, asynchronous information exchange that ensure the persistency and unique identification of the communicated semantic data.**

**Service Science: why this matter?**

- SOA will be a reality
  - It can knit together a diverse world
- SOA makes BPM possible
  - SOA’s benefits are fundamentally about improving business processes
- Service-oriented architectures are an arising software paradigm with big potential for the IT market.
- Bringing service orientation to its full potential requires its combination with semantics to mechanize important aspects and make it scalable.
**Examples**

**Google – Unified Cloud Computing**

- An attempt to create an open and standardized cloud interface for the unification of various cloud APIs.
- Key drivers of the unified cloud interface is to create an API about other APIs.
- Use of the resource description framework (RDF) to describe a semantic cloud data model (taxonomy & ontology).

---

**Amazon - Mechanical Turk**

“People as a service”

- **Amazon Mechanical Turk**
  - An API to Human Processing Power
  - The Computer Calls People
  - An Internet Scale Workforce
  - Game-Changing Economics
Amazon – S3 & EC2

“Infrastructure as a service”

- Amazon Simple Storage Service (S3)
  - Write and read objects up to 5GB
  - 15 cents GB / month to store
  - 20 cents GB / month to transfer

- Amazon Elastic Compute Cloud (EC2)
  - Allows customers to rent computers
    on which to run their own computer applications
  - Virtual server technology
  - 10 cents / hour

Summary

- Services rather than goods become the focus of economic and social exchange
- The service industry is increasing very rapidly. A new discipline
dedicated to the study, design, and implementation of service systems,
called Service Science, is emerging.
- Service Oriented Architecture (SOA) is in the center of this discipline
- However SOA will not scale without proper solutions for service related
tasks such as discovery and mediation
- Semantics offers a scalable solutions for the above mentioned problems
- Integrating semantics into SOA will result into the next generation of
intelligent service-based systems - SESA

REFERENCES
References

• Mandatory reading:

• Further reading:
  – Yefim V. Natis, Roy W. Schulze. Introduction to Service-Oriented Architecture, 14 April 2003

References

• Wikipedia links:

Next Lecture

# Title
1 Introduction
2 Web Science
3 Service Science
4 Web services
5 Web2.0 services
6 Semantic Web
7 Web Service Modeling Ontology (WSMO)
8 Web Service Modeling Language (WSML)
9 Web Service Execution Environment (WSMX)
10 OWL-S and other
11 Light-weight Annotations
12 Applications
13 Mobile Services

Questions?