Topic A2
Service Oriented Architecture

DAAD Project
"Joint Course on Software Engineering"

**Humboldt University Berlin, University of Novi Sad, University of Plovdiv,
University of Skopje, University of Belgrade, University of Niš, University of Kragujevac**

Version: Sept. 4, 2004

A2. Service Oriented Architecture (SOA)

- a) Definition
- b) History
- c) SOA vs. Distributed Objects
- d) Web Services and Protocols
- e) New Developments
- f) Resources

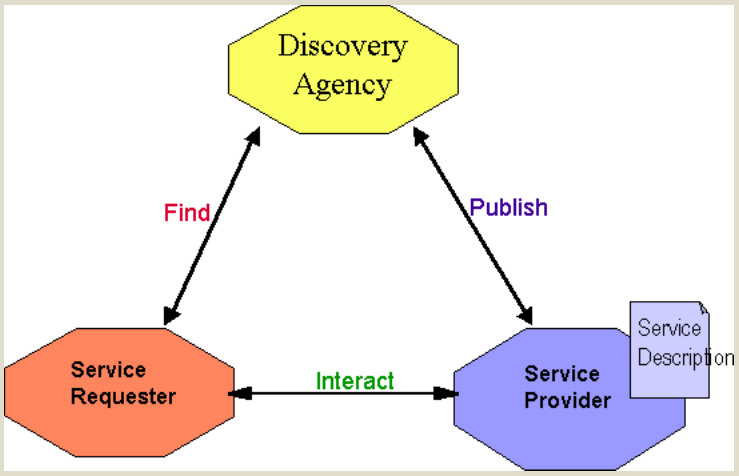
What is SOA? (1)

- « A design style for developing cross-patform, standards-oriented, loosely coupled distributed-application environments »
- « An architectural style that promotes business process orchestration of enterprise-level business services »
- « SOA is an architectural style whose goal is to achieve loose coupling among interacting software agents »
- « An SOA is a distributed software model »
- « SOA presents the big picture of what you can do with Web services »
- « … an important new paradigm that supports modularized implementation of solutions within a middle tier »
- « SOA is about unleashing the shared business services that are currently bound up in monolithic and often isolated applications »

What is SOA (2)

- “A means of developing distributed systems where the components are stand-alone services”
- “An open, agile, extensible, federated, composable architecture comprised of autonomous, QoS capable, vendor diverse, interoperable, discoverable, and potentially reusable services, implemented as Web services”
- “SOA can establish an abstraction of business logic and technology, resulting in a loose coupling between these domains.”
- “SOA is an evolution of past platforms, preserving successful characteristics of traditional architectures…”

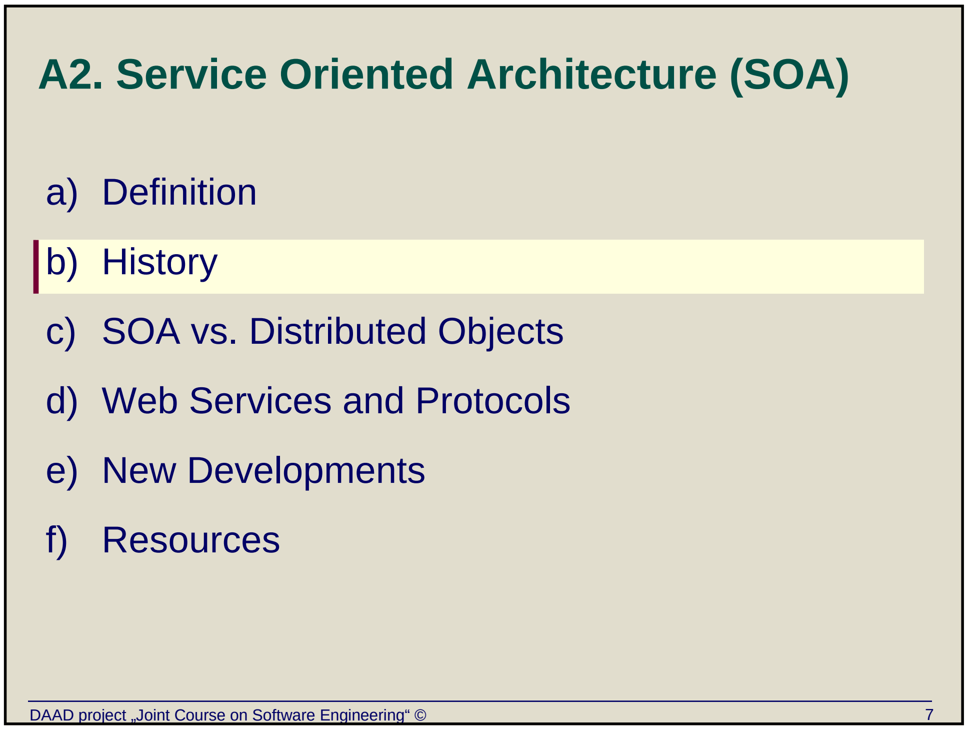
SOA – The big picture



SOA – benefits and disadvantages

- + Provides location independence: Services need not be associated with a particular system on a particular network
- + Protocol-independent communication framework renders code reusability
- + Offers better adaptability and faster response rate to changing business requirements
- + Allows easier application development, run-time deployment and better service management
- + Loosely coupled system architecture allows easy integration by composition of applications, processes, or more complex services from other less complex services
- + Provides authentication and authorization of Service consumers, and all security functionality via Services interfaces rather than tightly-coupled mechanisms
- + Allows service consumers (ex. Web Services) to find and connect to available Services dynamically

- For a stable or homogeneous enterprise IT environment, SOA may not be important or cost effective to implement
- If an organization is not offering software functionality as services to external parties or not using external services, which require flexibility and standard-based accessibility, SOA may not be useful.
- SOA is not desirable in case of real time requirements because SOA relies on loosely coupled asynchronous communication.



A2. Service Oriented Architecture (SOA)

a) Definition

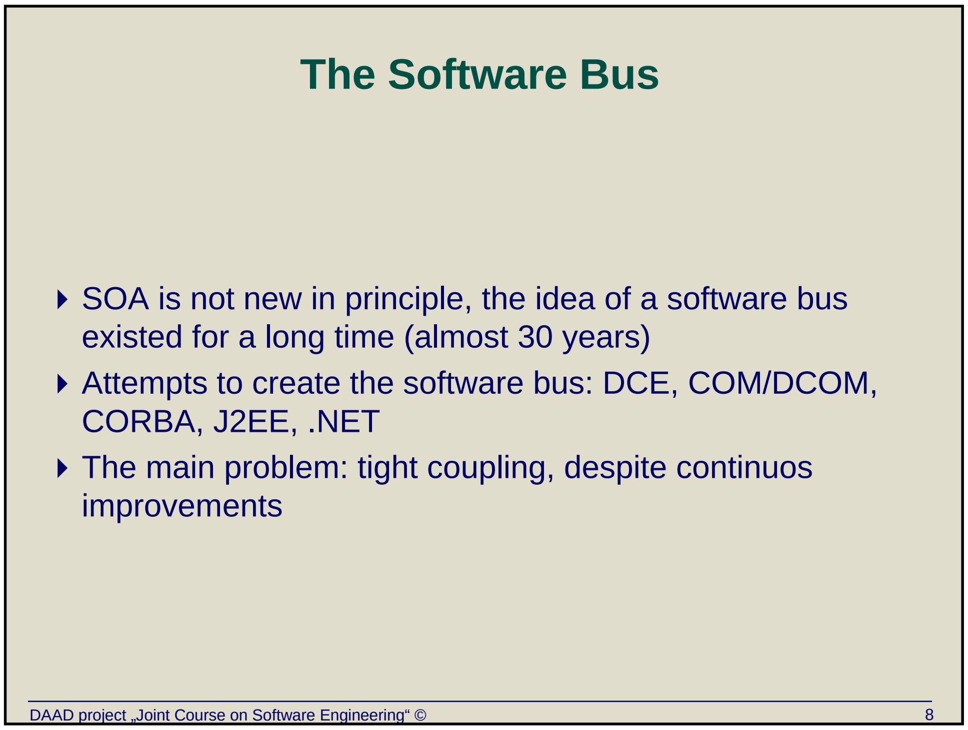
b) History

c) SOA vs. Distributed Objects

d) Web Services and Protocols

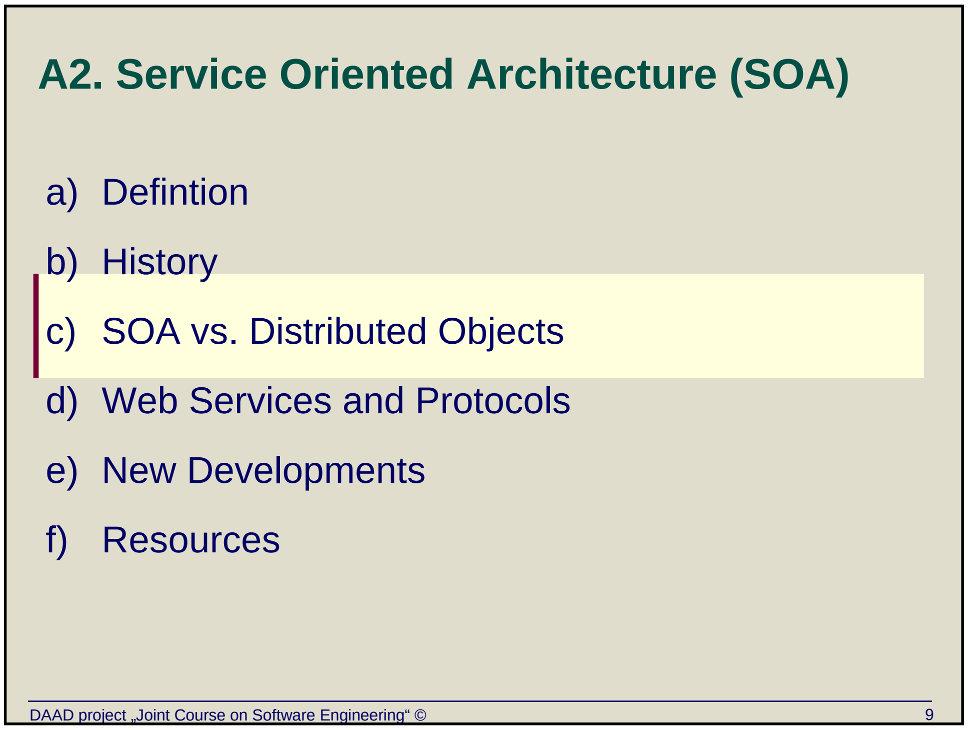
e) New Developments

f) Resources



The Software Bus

- SOA is not new in principle, the idea of a software bus existed for a long time (almost 30 years)
- Attempts to create the software bus: DCE, COM/DCOM, CORBA, J2EE, .NET
- The main problem: tight coupling, despite continuos improvements



A2. Service Oriented Architecture (SOA)

a) Defintion

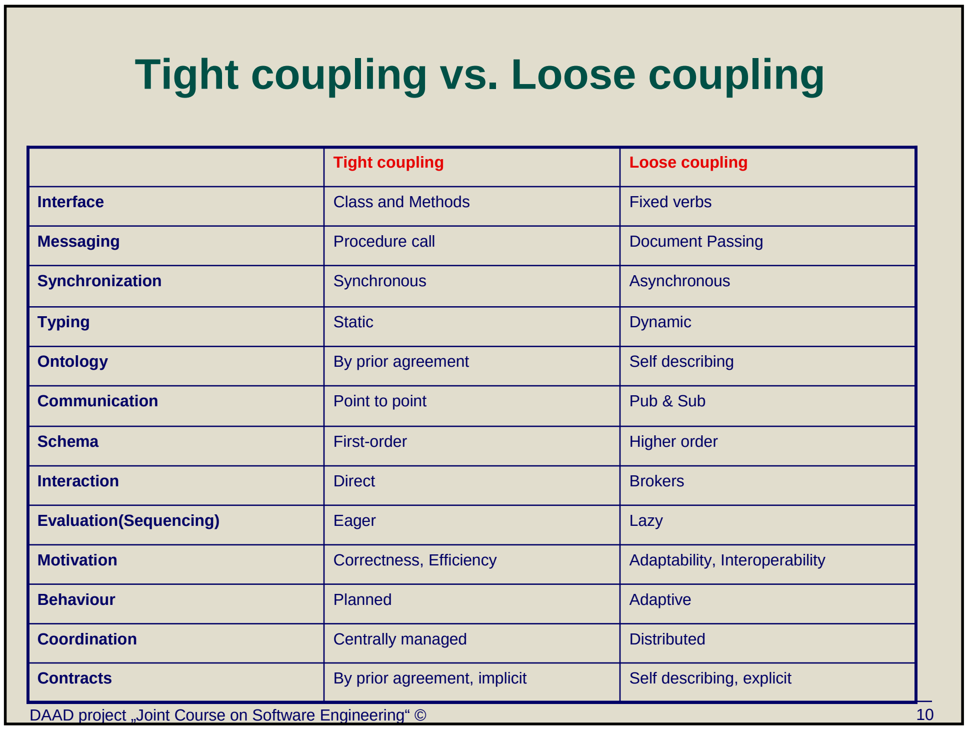
b) History

c) SOA vs. Distributed Objects

d) Web Services and Protocols

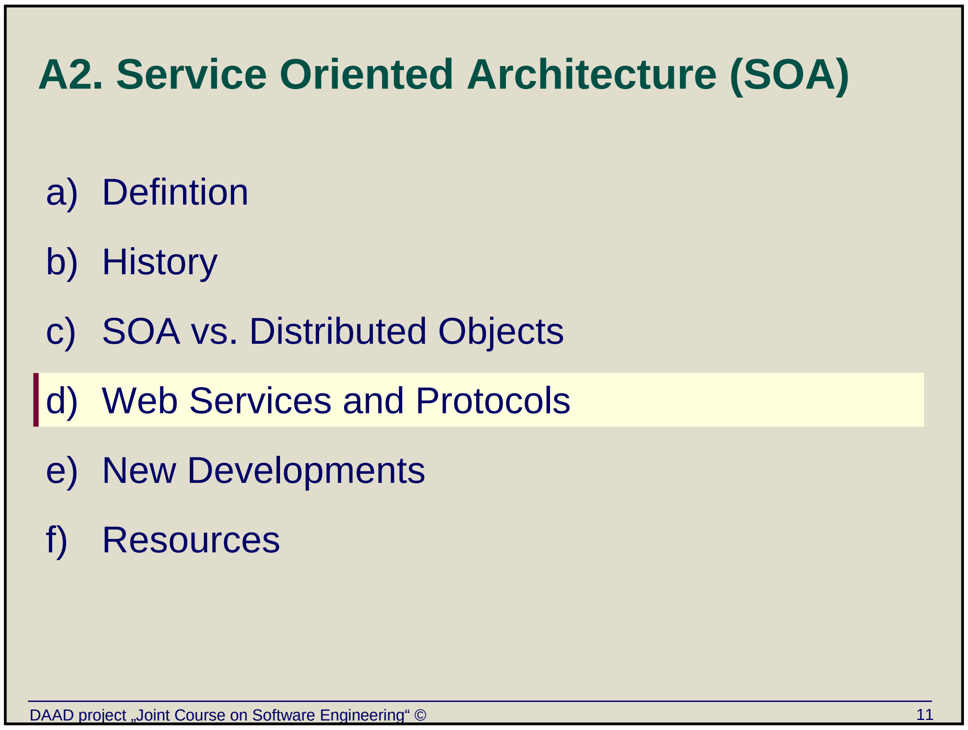
e) New Developments

f) Resources



Tight coupling vs. Loose coupling

	Tight coupling	Loose coupling
Interface	Class and Methods	Fixed verbs
Messaging	Procedure call	Document Passing
Synchronization	Synchronous	Asynchronous
Typing	Static	Dynamic
Ontology	By prior agreement	Self describing
Communication	Point to point	Pub & Sub
Schema	First-order	Higher order
Interaction	Direct	Brokers
Evaluation(Sequencing)	Eager	Lazy
Motivation	Correctness, Efficiency	Adaptability, Interoperability
Behaviour	Planned	Adaptive
Coordination	Centrally managed	Distributed
Contracts	By prior agreement, implicit	Self describing, explicit



A2. Service Oriented Architecture (SOA)

a) Defintion

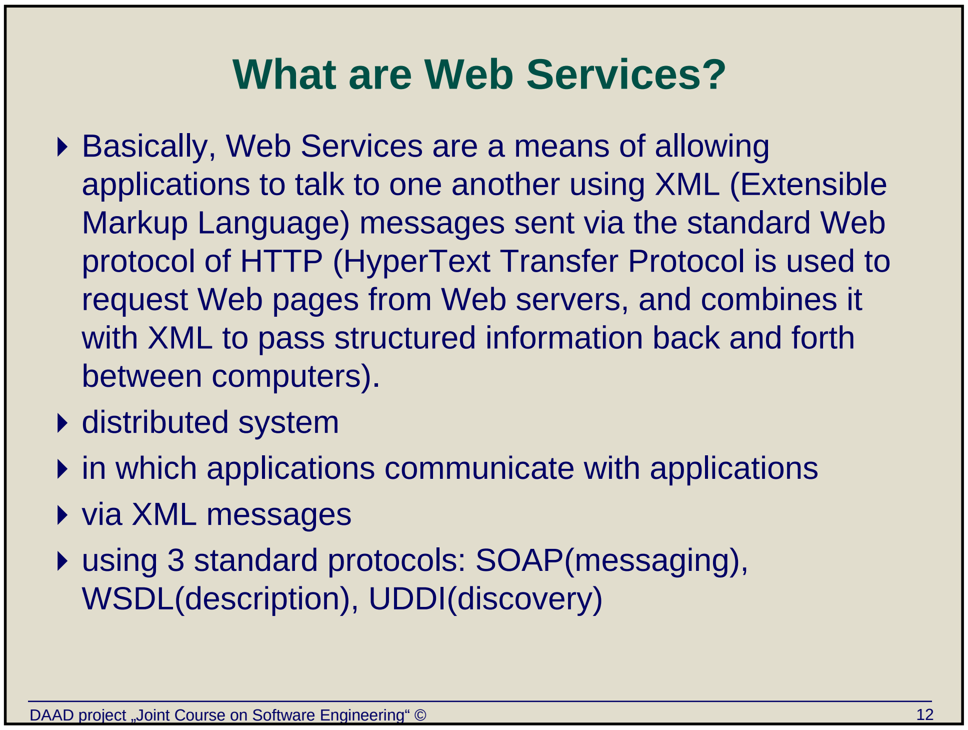
b) History

c) SOA vs. Distributed Objects

d) Web Services and Protocols

e) New Developments

f) Resources



What are Web Services?

- Basically, Web Services are a means of allowing applications to talk to one another using XML (Extensible Markup Language) messages sent via the standard Web protocol of HTTP (HyperText Transfer Protocol is used to request Web pages from Web servers, and combines it with XML to pass structured information back and forth between computers).
- distributed system
- in which applications communicate with applications
- via XML messages
- using 3 standard protocols: SOAP(messaging), WSDL(description), UDDI(discovery)

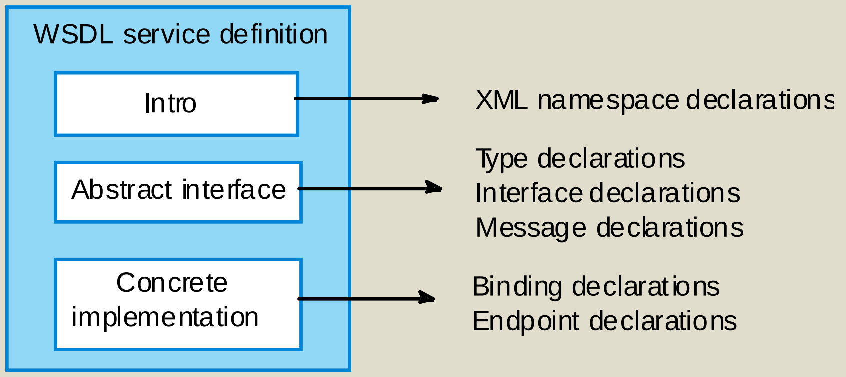
SOAP

- Defines:
  - Message construction (envelope, header, body)
  - Message exchange patterns and how to define more
  - Processing model for messages: originator, intermediaries, destination
  - Extensibility mechanism and *mustUnderstand* attribute
  - Fault system
  - Binding to transport protocols (HTTP, SMTP, etc.)

WSDL

- A WSDL document describes a service:
- message(s) accepted and emitted: abstract description (XML Schema)
- network protocol(s) and message format(s)
- operation: exchange of messages
- port type: collection of operations
- port: implementation of a port type
- service: collection of ports
- N.B. operations are atomic response+request pairs, not transition networks or state machines. Rules of operation are not captured completely.
- Controversy over general vs specific interfaces.

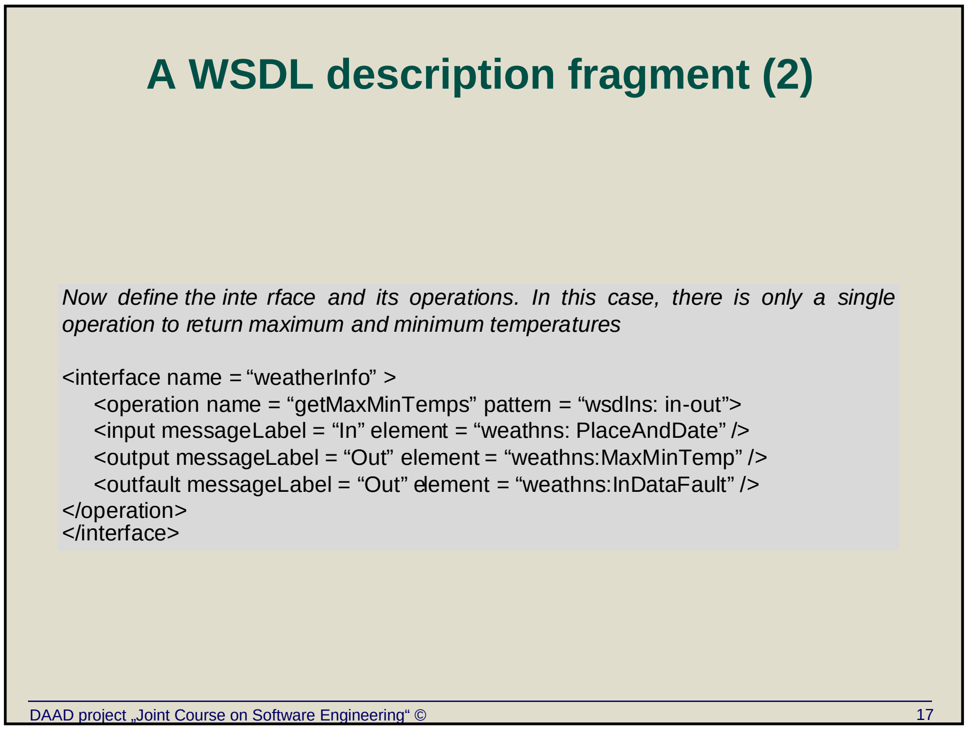
Structure of a WSDL specification



A WSDL description fragment (1)

Define some of the types used. Assume that the namespace prefixes 'ws' refers to the namespace URI for XML schemas and the namespace prefix associated with this definition is weathns.

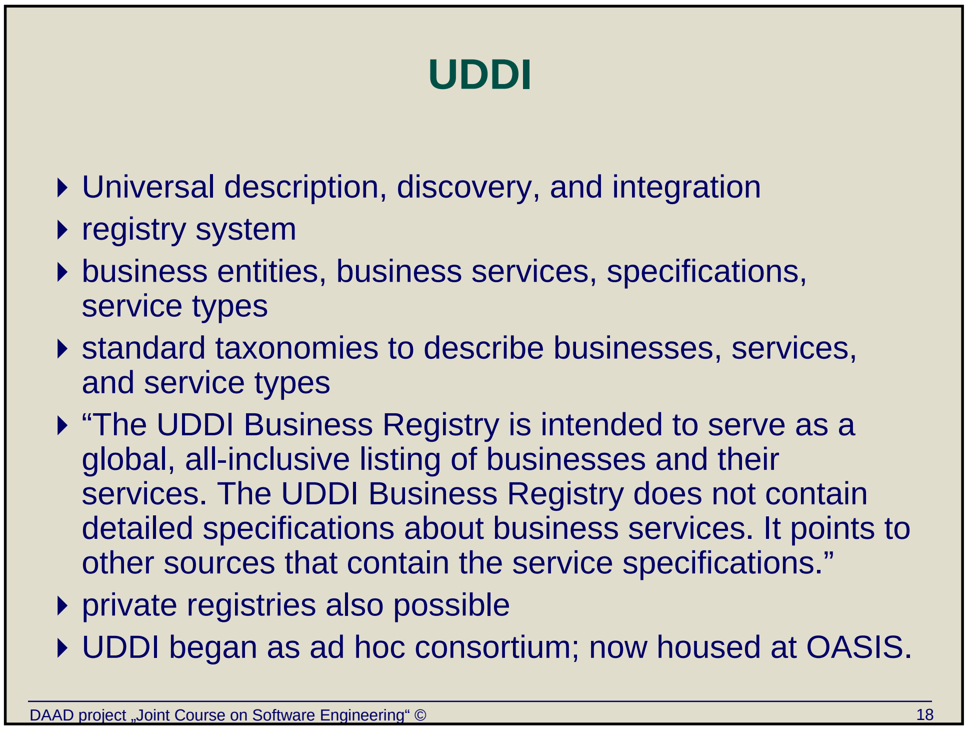
```
<types>
    <xs: schema targetNameSpace = "http ://.../weathns"
        xmlns: weathns = "http://…/weathns" >
        <xs:element name = "PlaceAndDate" type =  "pdrec" />
        <xs:element name = "MaxMinTem p" type = "mmtrec" />
        <xs: element name = "InDataFault" type = "errmess" />

        <xs: complexType name = "pdrec"
         <xs: sequen ce>
                <xs:element name = "tow n" type = "xs:string"/>
                <xs:element name = "country " type = "xs:string"/>
                <xs:element name = "day"  type = "xs:date" />
        </xs:complexTyp e>

        Definitions of MaxMinType and InDataFault here
    </schema>
</types>
```



A WSDL description fragment (2)

Now define the inte rface and its operations. In this case, there is only a single operation to return maximum and minimum temperatures

```
<interface name = “weatherInfo” >
   <operation name = “getMaxMinTemps” pattern = “wsdlns: in-out”>
   <input messageLabel = “In” element = “weathns: PlaceAndDate” />
   <output messageLabel = “Out” element = “weathns:MaxMinTemp” />
   <outfault messageLabel = “Out” element = “weathns:InDataFault” />
</operation>
</interface>
```



UDDI

- Universal description, discovery, and integration
- registry system
- business entities, business services, specifications, service types
- standard taxonomies to describe businesses, services, and service types
- “The UDDI Business Registry is intended to serve as a global, all-inclusive listing of businesses and their services. The UDDI Business Registry does not contain detailed specifications about business services. It points to other sources that contain the service specifications.”
- private registries also possible
- UDDI began as ad hoc consortium; now housed at OASIS.

A2. Service Oriented Architecture (SOA)

a) Defintion

b) History

c) SOA vs. Distributed Objects

d) Web Services and Protocols

e) New Developments

f) Resources

Service oriented software engineering

- Existing approaches to software engineering have to evolve to reflect the service-oriented approach to software development
  - Service engineering. The development of dependable, reusable services
    - Software development for reuse
  - Software development with services. The development of dependable software where services are the fundamental components
    - Software development with reuse

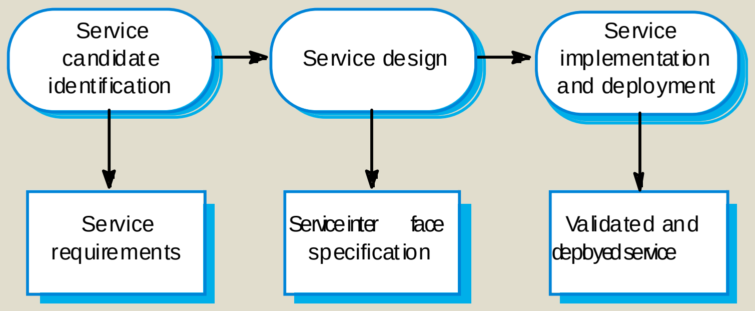
Services as reusable components

- A service can be defined as:
  - *A loosely-coupled, reusable software component that encapsulates discrete functionality which may be distributed and programmatically accessed. A web service is a service that is accessed using standard Internet and XML-based protocols*
- A critical distinction between a service and a component as defined in CBSE is that services are independent
  - Services do not have a ‘requires’ interface
  - Services rely on message-based communication with messages expressed in XML

Service engineering

- The process of developing services for reuse in service-oriented applications
- The service has to be designed as a reusable abstraction that can be used in different systems
- Involves
  - Service candidate identification
  - Service design
  - Service implementation

The service engineering process


Service candidate identification
Service design
Service implementation and deployment
Service requirements
Service interface specification
Validated and deployed service

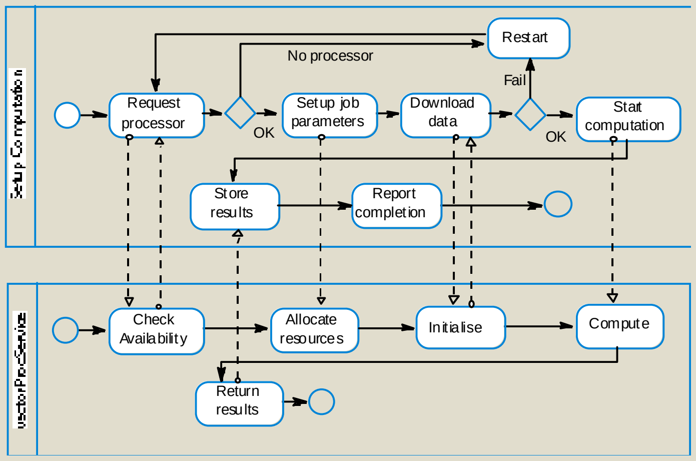
Software development with services

- Existing services are composed and configured to create new composite services and applications
- The basis for service composition is often a workflow
  - Workflows are logical sequences of activities that, together, model a coherent business process
  - For example, provide a travel reservation services which allows flights, car hire and hotel bookings to be coordinated

Workflow design and implementation

- WS-BPEL is an XML-standard for workflow specification. However, WS-BPEL descriptions are long and unreadable
- Graphical workflow notations, such as BPMN, are more readable and WS-BPEL can be generated from them
- In inter-organisational systems, separate workflows are created for each organisation and linked through message exchange

Interacting workflows



Service testing

- Testing is intended to find defects and demonstrate that a system meets its functional and non-functional requirements
- Service testing is difficult as (external) services are ‘black-boxes’. Testing techniques that rely on the program source code cannot be used

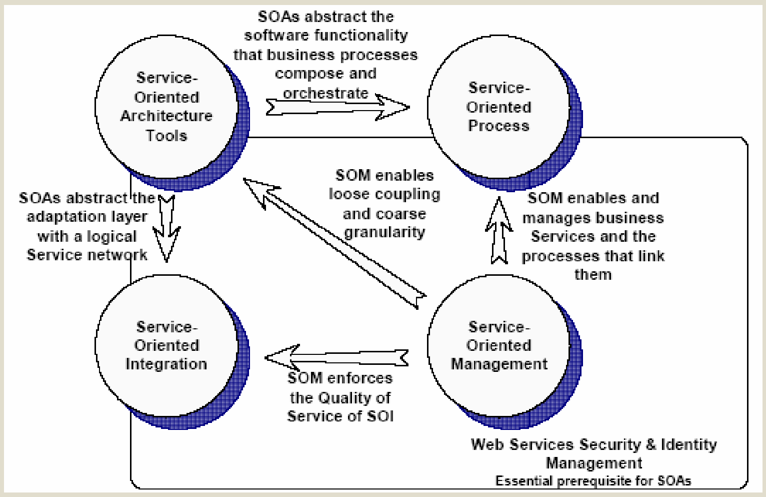
Service testing problems

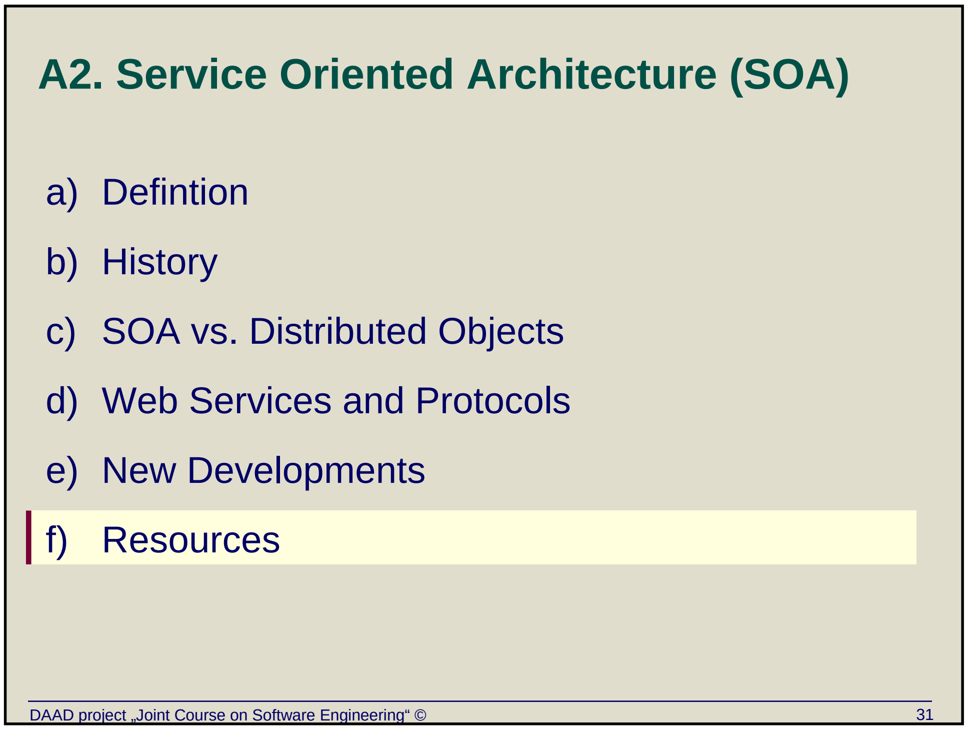
- External services may be modified by the service provider thus invalidating tests which have been completed
- Dynamic binding means that the service used in an application may vary - the application tests are not, therefore, reliable
- The non-functional behaviour of the service is unpredictable because it depends on load
- If services have to be paid for as used, testing a service may be expensive
- It may be difficult to invoke compensating actions in external services as these may rely on the failure of other services which cannot be simulated

Service-Oriented Architecture Implementation Framework

- Service-Oriented Architecture Implementation Framework (SOAIF) provides the run-time deployment infrastructure for SOA across the network by incorporating all the software required to develop, deploy, secure, manage, and extend service-oriented processes and solutions.

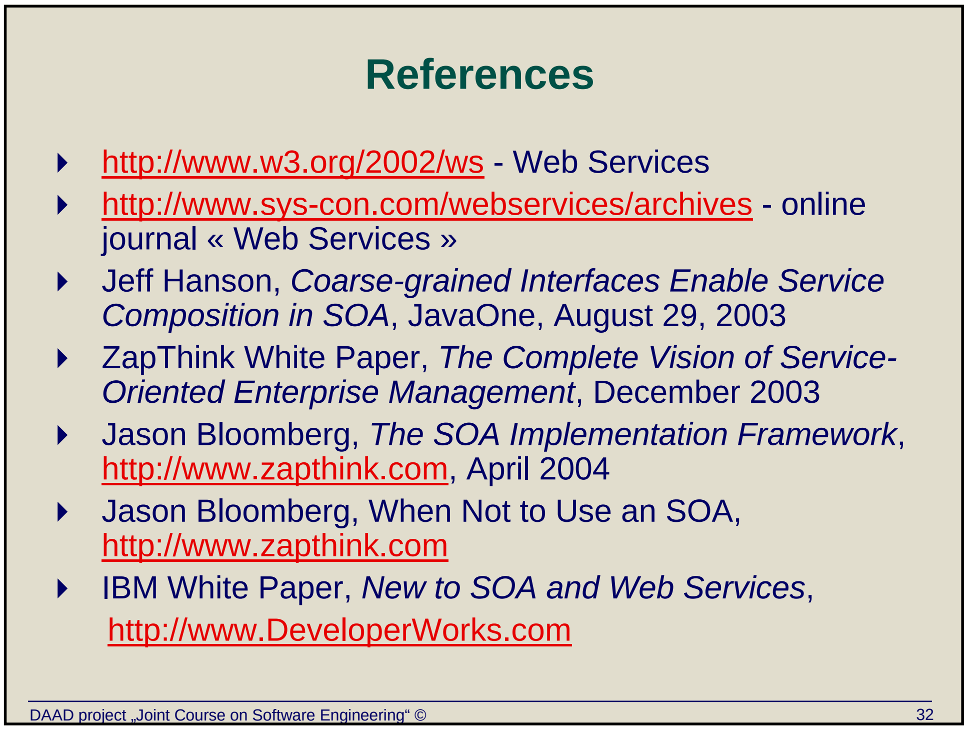
SOAIF


SOAs abstract the software functionality that business processes compose and orchestrate
Service-Oriented Architecture Tools
Service-Oriented Process
SOAs abstract the adaptation layer with a logical Service network
SOM enables loose coupling and coarse granularity
SOM enables and manages business Services and the processes that link them
Service-Oriented Integration
Service-Oriented Management
SOM enforces the Quality of Service of SOI
Web Services Security & Identity Management
Essential prerequisite for SOAs



A2. Service Oriented Architecture (SOA)

a) Defintion
b) History
c) SOA vs. Distributed Objects
d) Web Services and Protocols
e) New Developments
f) Resources



References

- http://www.w3.org/2002/ws - Web Services
- http://www.sys-con.com/webservices/archives - online journal « Web Services »
- Jeff Hanson, *Coarse-grained Interfaces Enable Service Composition in SOA*, JavaOne, August 29, 2003
- ZapThink White Paper, *The Complete Vision of Service-Oriented Enterprise Management*, December 2003
- Jason Bloomberg, *The SOA Implementation Framework*, http://www.zapthink.com, April 2004
- Jason Bloomberg, When Not to Use an SOA, http://www.zapthink.com
- IBM White Paper, *New to SOA and Web Services*, http://www.DeveloperWorks.com

Resources (2)

- Thomas Earl: “Service-Oriented Architecture. Concepts, technology, and Design”, Prentice Hall, 2005, ISBN 0-13-185858-0
- Thomas Earl: “Service-Oriented Architecture. A Field Guide to Integrating XML and Web Services”, Prentice Hall, 2004, ISBN 0-13-142898-5
- Ian Sommerville: “Software Engineering”, 8th edition, Addison-Wesley, 2006, ISBN 0321313798