Software Service Engineering

Prof. Dr.-Ing. Martin Gaedke
Technische Universität Chemnitz
Fakultät für Informatik
Professur Verteilte und selbstorganisierende Rechnersysteme

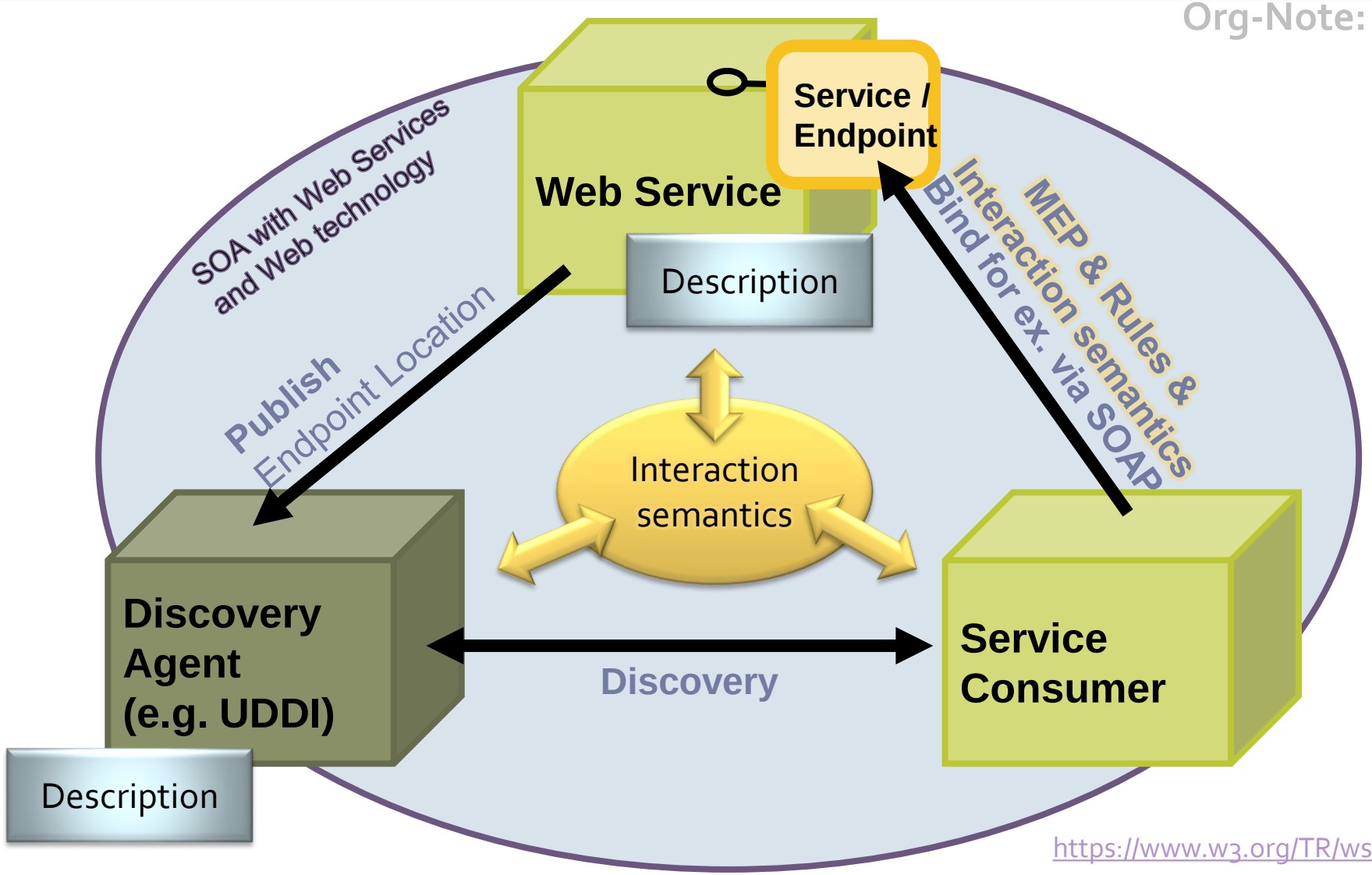
http://vsr.informatik.tu-chemnitz.de

Web Services Architecture

- **Web Services Architecture (WSA)**
  - W3C Working Group Note 11 February 2004
  - https://www.w3.org/TR/ws-arch/
  - Overview of SOA implementation using Web Services
  - Describes various aspects of such an infrastructure
    - Web Service
    - Architectural approaches for Message Oriented Model (MOM), Service Oriented Model, Resource Oriented Model and Policy Model
    - SOA aspects with respect to distributed computing, architectural characteristics and connection to REST
    - Web Service technologies, XML, SOAP, WSDL
    - Use of Web Services
    - Various further aspects, such as Web Service Discovery, Semantics, Security

WSA in a (simplified) Overview

Org-Note: W3



https://www.w3.org/TR/ws-arch/

w3.org/TR/ws-arch/

W3C Working Group Note

W3C

Web Services Architecture

W3C Working Group Note 11 February 2004

This version:
http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/
Latest version:
http://www.w3.org/TR/ws-arch/
Previous version:
http://www.w3.org/TR/2003/WD-ws-arch-20030808/
Editors:
David Booth, W3C Fellow / Hewlett-Packard
Hugo Haas, W3C
Francis McCabe, Fujitsu Labs of America
Eric Newcomer (until October 2003), Iona
Michael Champion (until March 2003), Software AG
Chris Ferris (until March 2003), IBM
David Orchard (until March 2003), BEA Systems

This document is also available in these non-normative formats: PostScript version and PDF version.

Abstract

This document defines the Web Services Architecture. It identifies the functional components and defines the relationships among those components to effect the desired properties of the overall architecture.

Status of this Document

This section describes the status of this document at the time of its publication. Other documents may supersede this document. A list of current W3C publications and the latest revision of this technical report can be found in the W3C technical reports index at http://www.w3.org/TR/.

This is a public Working Group Note produced by the W3C Web Services Architecture Working Group, which is part of the W3C Web Services Activity. This publication as a Working Group Note coincides with the end of the Working Group's charter period, and represents the culmination of the group's work.

Discussion of this document is invited on the public mailing list www-ws-arch@w3.org (public archives). A list of remaining open issues is included in **4 Conclusions**.

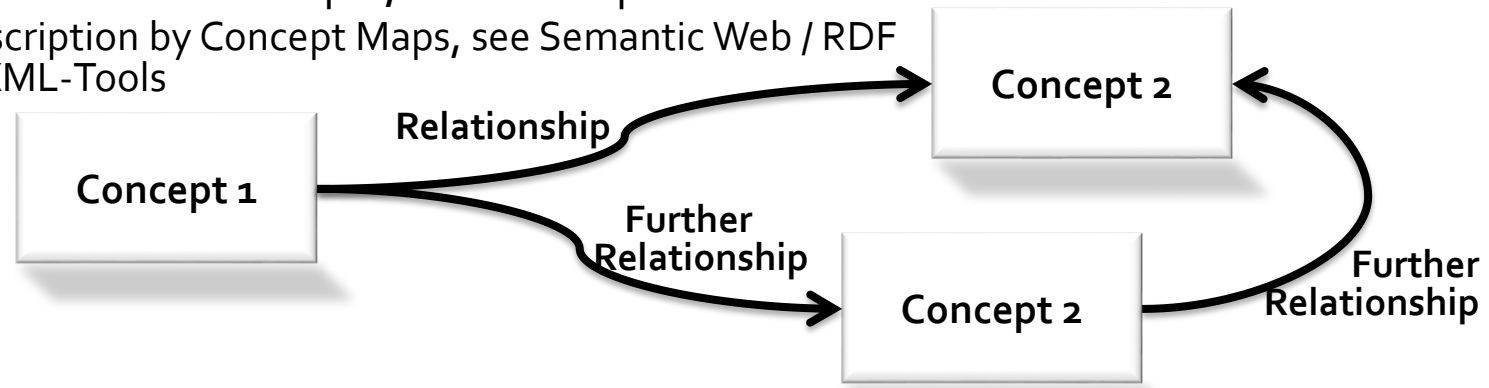
Patent disclosures relevant to this specification may be found on the Working Group's patent disclosure page.

Publication as a Working Group Note does not imply endorsement by the W3C Membership. This is a draft document and may be updated, replaced or obsoleted by other documents at any time. It is inappropriate to cite this document as other than work in progress. Other documents may supersede this document.

Table of Contents

SOA versus WSA

- **W3: https://www.w3.org/TR/ws-arch/**
- WSA is SOA with Web Services, Web Technology and Concepts
  - **Service Provider**
    - → Web Service
  - **Service Consumer**
    - → Client (Web Server, Browser, Application)
  - **Registry / Service Broker**
    - → Discovery Agent (z.B. UDDI, P2P mechanisms)
  - Message exchange **technology**
    - For message encoding: especially XML
    - For transmission: especially SOAP and HTTP
    - For EndPoint description: XML
  - **Semantics** of the concepts, relationships and models
    - Description by Concept Maps, see Semantic Web / RDF in XML-Tools



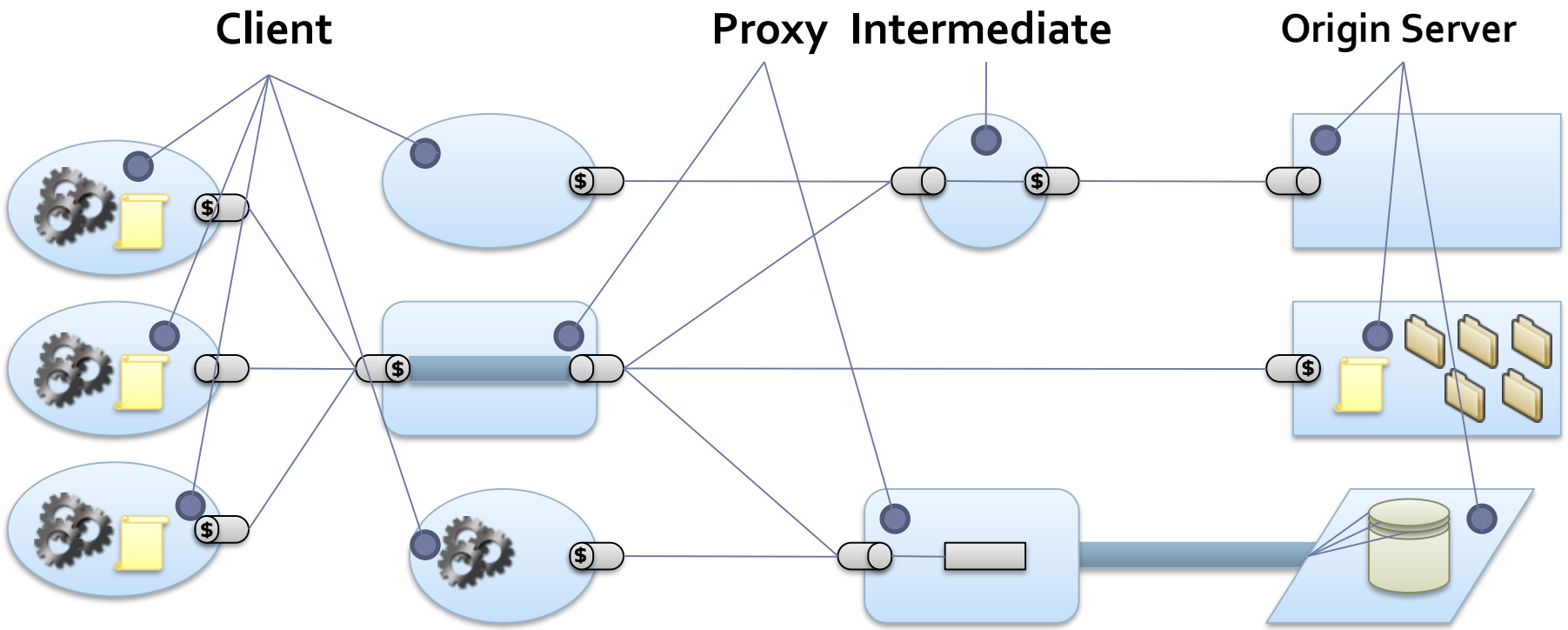
Classes of Web Services

- A *Web service* is a software system designed to support interoperable machine-to-machine interaction over a network

W3C Web Services Architecture
http://www.w3.org/TR/ws-arch/

- Classes of Web Services:
  - *REST-compliant Web services*, in which the primary purpose of the service is to manipulate (XML) representations of Web resources using a uniform set of "stateless" operations
  - *Arbitrary Web services*, in which the service may expose an arbitrary set of operations

Web of Web Services (1)



Web of Web Services (2)

- **Client**
  - Places an initial Request and is the final recipient of the Response
  - Responsible for rendering the resource's representation to an application-specific form

- **Origin Server**
  - Responsible component, which holds different representations of a resource
  - Final receiver of all Requests, which cause resource change

  - Which consequences does it have?
  - Components that have the representations cached cannot change the resources and have to forward the Request to the Origin Server

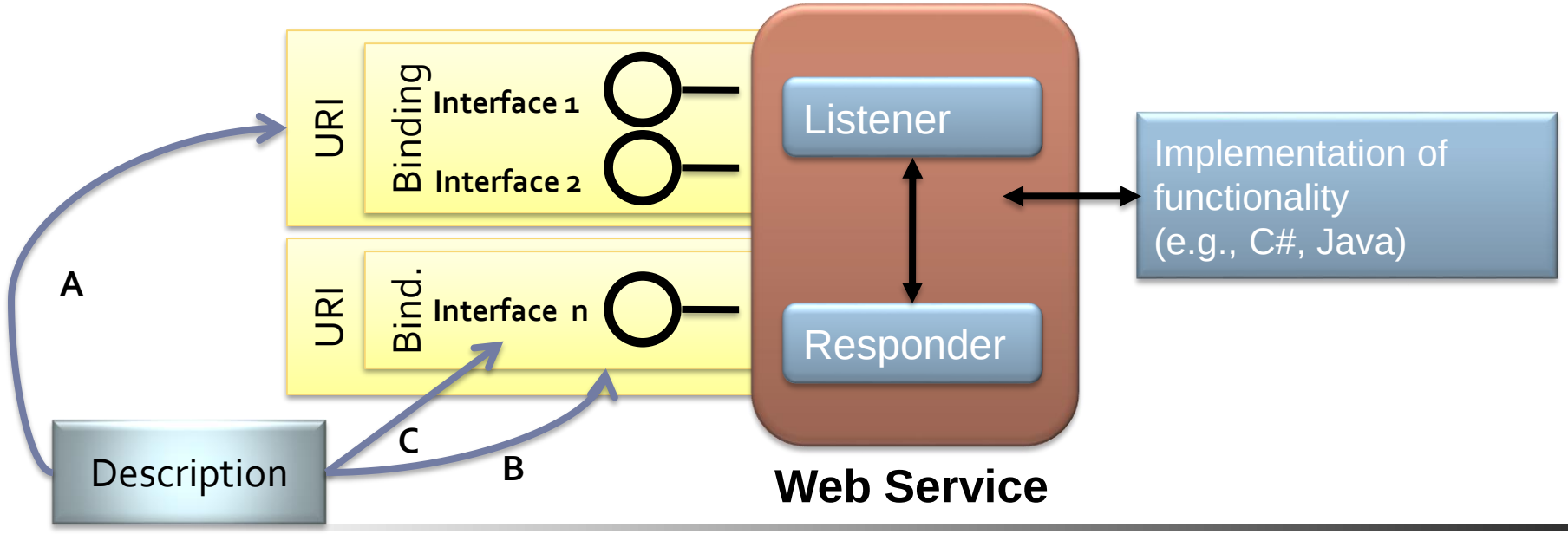
Web of Web Services (3)

- **Proxy**
  - Is approached by a Client and offers:
    - Encapsulation of further services
    - Data transformation (representation change)
    - Performance (Caching)
    - Security
- **Intermediate**
  - Is made available by the Origin Server and offers:
    - Encapsulation of further services
    - Data transformation
    - Performance
    - Security
- What is the difference?
  - The Client is free to choose which Proxy to use and whether to use one at all

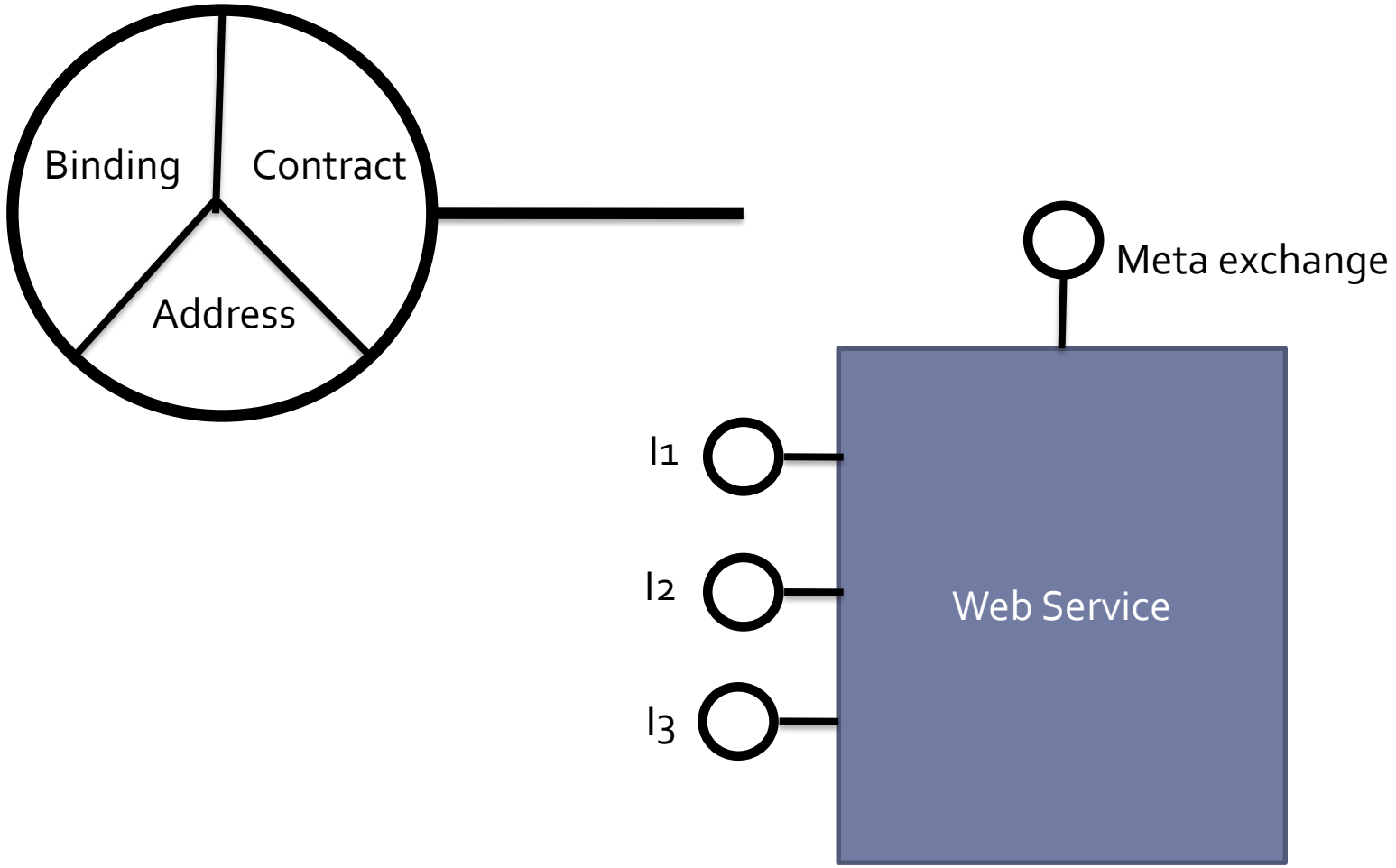
Chapter 2
THE SERVICE

Web Service: abstract representation

- A: **Address** URI for Interface/Endpoint
- B: **Binding** for message exchange, for example, via SOAP or HTTP
- C: **Contract** – Functionality description



Remember


Binding
Contract
Address
Meta exchange
I1
I2
I3
Web Service

Contract

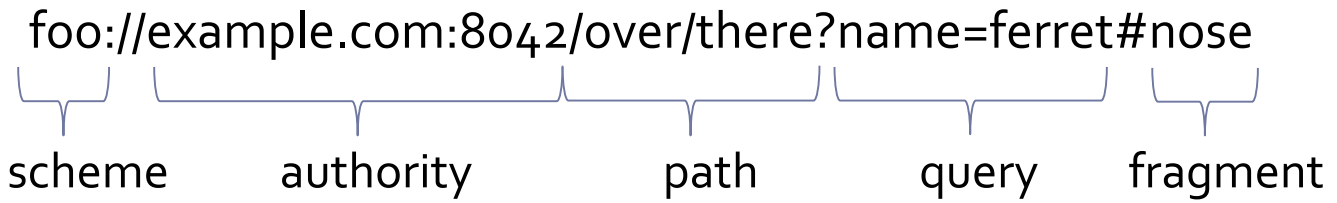
- Comes from the RPC world
- Specifies the functionality provided by a Web service
  - Set of available operations
  - Required input parameters and types of the operations available
  - Form of the messages

Binding

- Specifies how to communicate with a Web service
  - Transport protocol, e.g. via TCP, HTTP, SMTP, …
  - Encoding, e.g. UTF8, binary
  - Security

Addressing

- Specifies where to find the web service
- Should be unique
- E.g. URI (RFC 3986)
- Check IETF and W3C for standard documents



Chapter 3

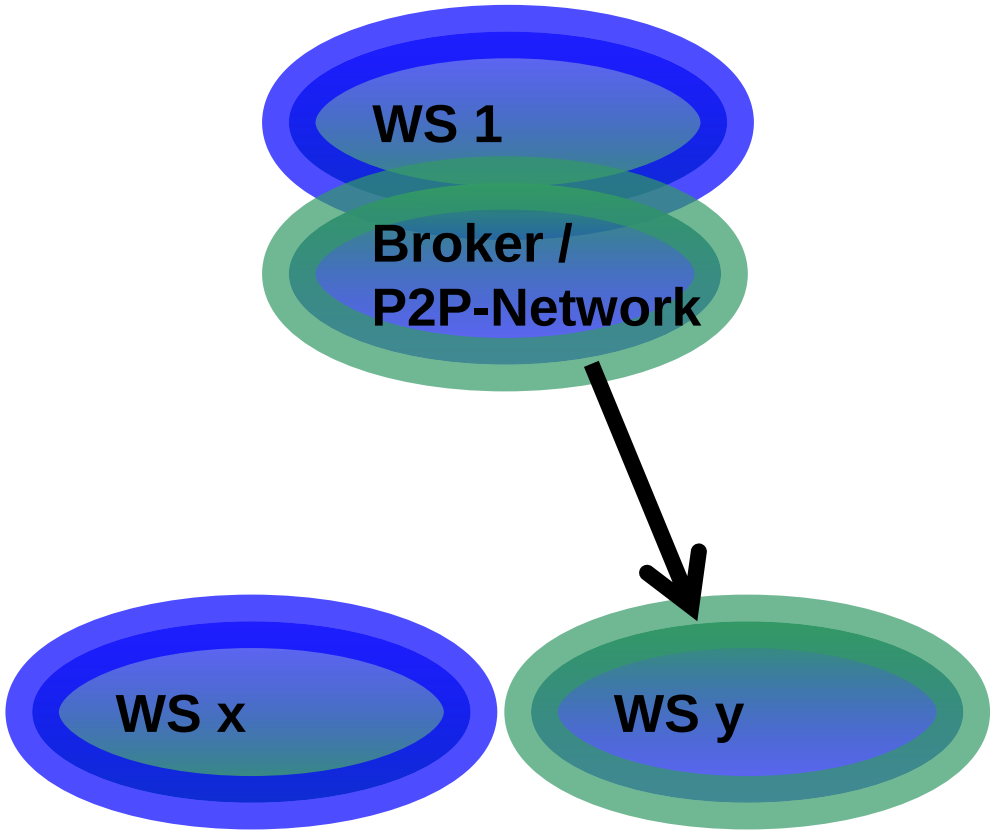
WEB SERVICE DISCOVERY

Broker and P2P Approaches

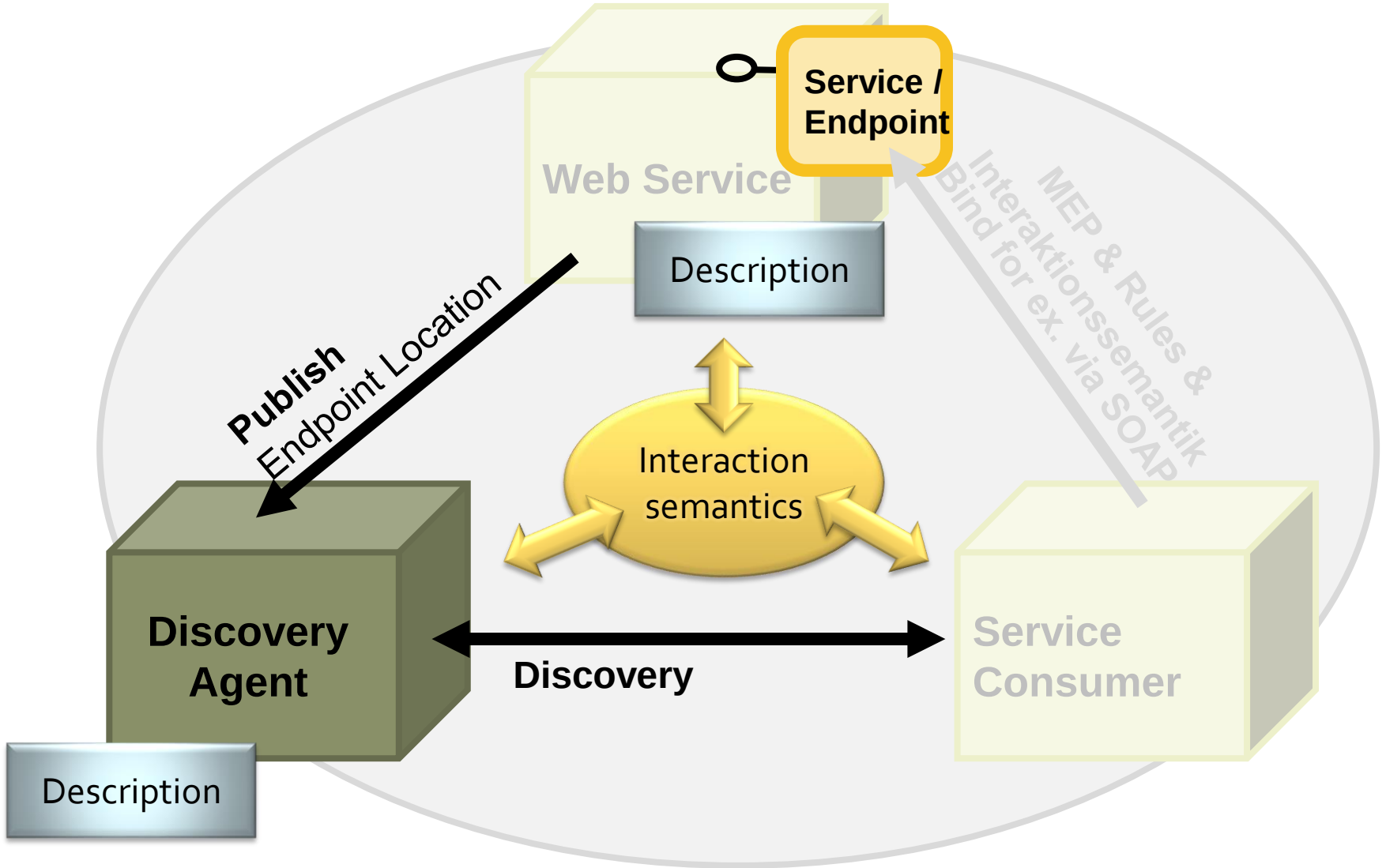
- Communicating processes may *be brokered by dedicated service (Broker) or an "intelligent" network (P2P network)*
  - Communications among business processes will often be requests for or provisions of Web Services
  - The consumer and provider likely do not have a priori knowledge of one another
  - Services will be **mediated** by **Brokers** or the **underlying P2P network**

Dynamic WS Discovery

- Web Service calls Web Service mediated by Broker (respectively P2P network)
  - Criteria may be quality, context, price, etc.
  - Requires classification system or metadata
- Broker could use UDDI automatically on request
- P2P discovery by content-based routing (e.g. for WSDL)



Introduction



Web Service Discovery

- **Discovery Service** – Is a backup service, which simplifies the process of Web service search.
  - It is a logical role, i.e. the task of Discovery Service can be performed by the Service Consumer, Web Service or another system
  - Discover process should facilitate search, i.e. it should support humans / machines with finding others
    - Manual Discovery – Discovery process is performed by a human
    - Autonomous Discovery – Discovery process is automated

Forms of Discovery Services

- Current form of Discovery Services' implementation:
  - **Registry Approach** – Authoritative, centrally controlled data storage
  - **Index Approach** – Index is a list of data, which is stored somewhere else
  - **Peer-to-Peer (P2P) Approach** – Does not use any centrally stored data, rather tries to facilitate Web Services in finding each other
  - **Federated Approaches** – Aggregates data from different Discovery Services of different operators
- Important in these considerations is not the technology, but who has the authority

DS Form: Registry Approach

- **Registry Approach** – Authoritative, centrally controlled data storage
  - Provider must be initially active (create an entry) for the data to be found
  - Registry owner has the power as to who can submit data
  - Registry owner determines what should be submitted
  - Example: Universal Description, Discovery, and Integration (UDDI), ProgrammableWeb.com

DS Form: Index Approach

- **Index Approach** – Index is a list of data, which is stored somewhere else
  - Publishing is passive, i.e. the Web Service description is made available and the index owners interested in the data use it without any further arrangements
  - Anyone can create an index, e.g. via Spider similarly to search engines
  - Further opportunities like advertising, e.g. influencing ranking, advertising
  - Example: Google

DS Form: P2P Approach

- **Peer-to-Peer (P2P) Approach** - Does not use any centrally stored data, rather tries to facilitate Web Services in finding each other
  - Web Service is a member of a network of equal partners (peers), which might be Web Services themselves (but not necessarily)
  - Search leads to requests being sent out to neighbors, which can, in turn, ask their own neighbors until the desired interface is found (or not)
  - Positive: Fail-safe, ideal for dynamic relations
  - Negative: Complexity (possibly, slow as well), no guarantee that the whole network would be searched, low and obstructive predictability

DS Forms: Federated Approaches

- **Federated Approaches** – Aggregates data from different Discovery Services of different operators
  - Example: Vacation booking: Interfaces for train, flight and car rental services
  - Discovery Services of different operators are queried separately or using an aggregated query, which is then partly resolved by each service
  - Positive: Very powerful
  - Negative: Complexity
  - Requires: Languages to support choreography or orchestration approaches