ISSGC'05

Introduction to Web Services

NeSC Training Team

www.eu-egee.org

National e-Science Centre

Information Society

Overview

Goals –

- An Appreciation of the role and context of
  - Web Services in general
  - Java-based tools
  - Structure of Standards

- **Structure –**
  - Web Services
  - Outline of JAVA tools and their use
  - Standards Structure

Evolution of Electronic inter-enterprise interaction

Web Services is the next step in the automation of inter-enterprise interaction

Web Browsing

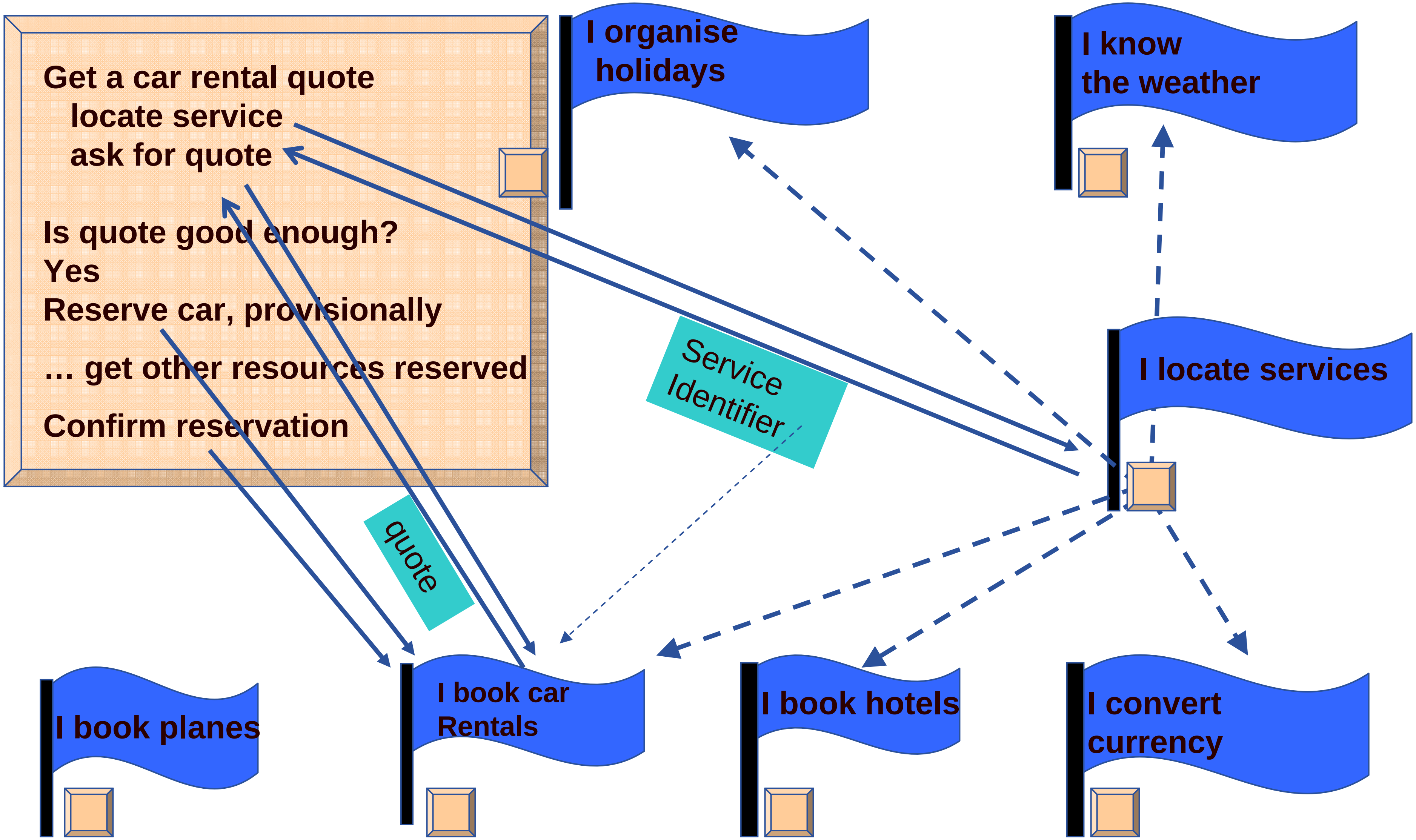
- Human travel agent provides “organise holiday” service by surfing the web to look for and invoking services – book a hotel; book a plane; book a car hire; ….; confirm bookings of best options to meet client needs.

Web Services

- The aspiration of Web services is to provide a framework that allows that same model to be used in writing an application –
- which is itself becomes an “organise a holiday” service, finding and using useful services

Mode	human intervention at –	
	service provider	service consumer
E-mail	Yes	Yes
Web browsing	No	Yes
Web Services	No	No

Service Interaction



Get a car rental quote
locate service
ask for quote

Is quote good enough?
Yes
Reserve car, provisionally

… get other resources reserved

Confirm reservation

Essential Requirements

- Need to achieve effective cooperation even though
  - the different services are
    - produced by different organisations,
    - without any design collaboration,
    - on different platforms
      - Requires “interoperability”
  - the services are autonomously evolving
- Requires Loose Coupling

Coupling

COUPLING – about intensity of communication

- Execution Coupling –
  - Frequency and extent of communication relative to processing
  - telephone conversation is tightly coupled, e-mail conversation is loosely coupled
  - For web services – very loose coupling
  - Interaction of order of a second
    - Whereas centralised object invocation – micro-seconds
  - Coarse granularity – do enough work in a service request to justify the time taken by the communication overhead
- Design Coupling –
  - How much design knowledge has to be communicated between the designers of the software at the two ends of an interaction
  - Which they then build into their software
  - The extent of statically shared knowledge between two ends of an interaction

Loose Design Coupling

- Loose (Design) coupling – minimum prior shared information between the designer of the two components of an interaction
  - Dynamically accessible Machine processable Meta data
    - Self-describing data in standard format – XML documents
    - Description of structure of communications – SCHEMAS (types)
    - Service description – WSDL, using SCHEMAS for message structure
    - Means for obtaining it – from a repository, using standard such as UDDI
    - Communication protocol that supports this – SOAP
  - Everything is a SCHEMA-described XML document – soap message, WSDL definition, schemas themselves (meta-schema)
  - Tolerance of partial understanding
    - Schemas allows extension points – one participant may have an older WSDL definition which accommodates extensions with additional information

WS vs O-O

- A service is a
  - S/W system designed to support interoperable machine-to-machine interaction over a network. (W3C Glossary)
- Has some of the characteristics of O-O architecture
- The O-O class roughly corresponds to a PortType (i.e. Interface) –
  - a collection of operations each with defined input and output data
- Object roughly corresponds to
  - a Service – an instantiation of a PortType
    - at a particular web location
    - using a particular communication protocol and message representation
- But
  - Less constrained than O-O model - interoperability
  - Focussed on very loose coupling
  - In O-O – new instances created dynamically by user request
    - Not true of basic WS – I Services
    - For that behaviour in Web Services use Resource Framework - WSRF

A Perspective on Web Services Model

- COUPLING – about intensity of communication
  - Degree of statically shared knowledge between two end of an interaction (knowledge which the programmer/designer has to know and build-in) – how much has to be communicated
  - Frequency and extent of communication relative to processing
- A scale of looser coupling (in both senses)
- Shared variable
  - interaction is
  - One end updating a variable; other end using it
- Object-Oriented
  - One end invoking method; other end being invoked
- Web Services
  - One end (service consumer) requesting a service
  - Other end (service provider) servicing the request
  - Quite similar to O-O (but might not be a reply!)

A Perspective on Web Services Model

- **Shared Variable Model - Close coupling**
  - The programmers of user side of an interaction know all about representation
  - Shared implementation
  - Suitable for single-programmer level
  - Interaction of order of nanosecond
  - Fine granuality
    - almost no work in a variable assignment
    - Simplest of tasks involves many interactions with variables
- **Object Oriented Model - Medium Coupling**
  - User side of interaction knows – what classes exist and their interface
    - But not their representation
  - Shared class design
  - Suitable for single-organisation level
  - Interaction of order of micro/milli-sec (possibly distributed objects)
  - Medium granuality – do some work in a method invocation – 20 lines of code
  - Within an object, typically use the Shared Variable model

A Perspective on Web Services Model

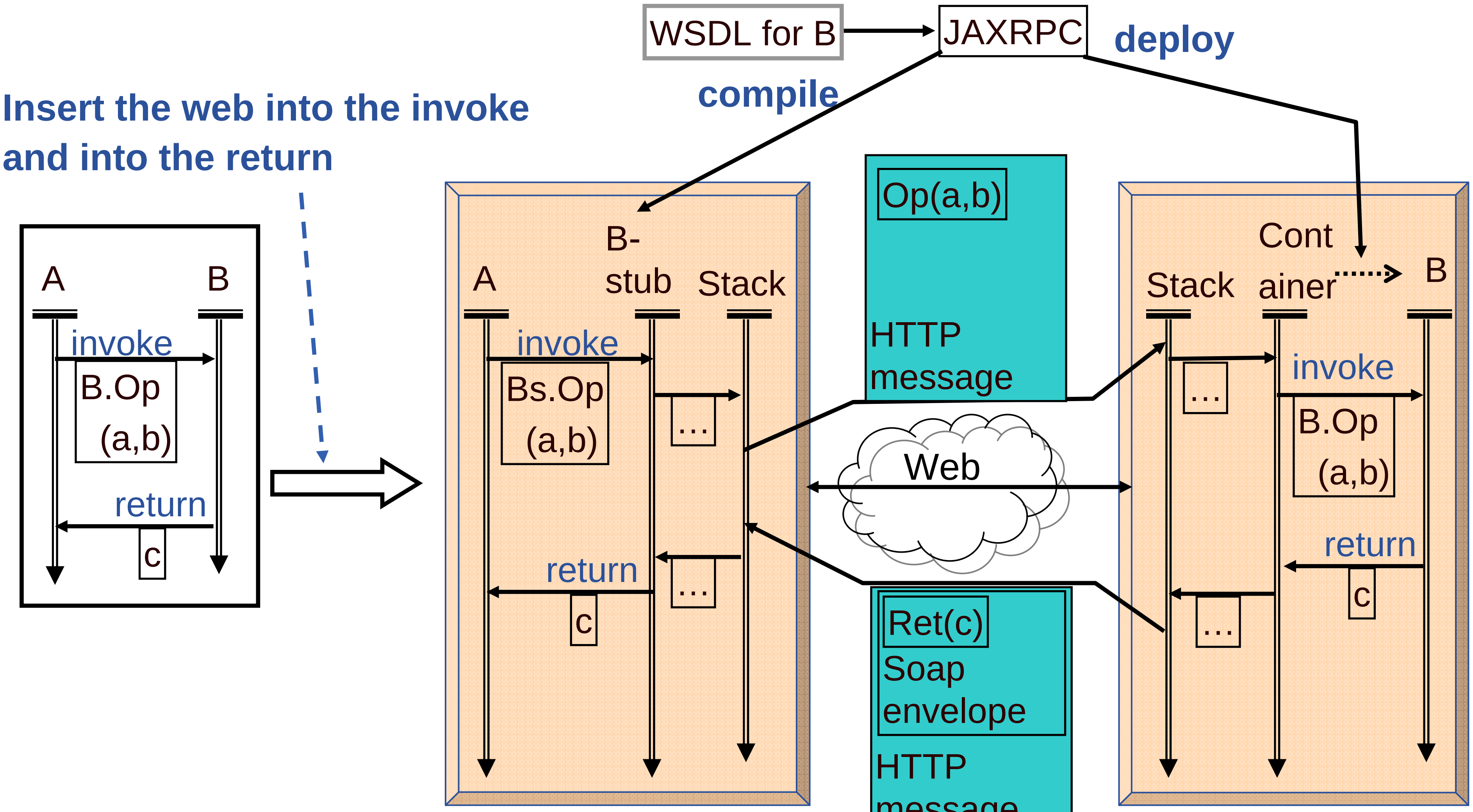
- **Shared Variable Model - Close coupling**
  - Shared implementation ; single-programmer ; nanosec interaction
  - nanosecond interaction; fine granuality;
- **Object-Oriented Model – Medium Coupling**
  - Shard Class Design ; single organisation ;
  - Micro/milli-sec interaction; medium granuality
- **Web Services - Loose coupling**
  - Programmers on user side knows how to programme the discovery of a service
  - Shared standards and knowledge of standard repository
  - Interaction of order of second
  - Coarse granuality – do enough work in a service request to justify the time taken by the communication overhead
  - Within a service, typically use the Object-oriented model – service request-response is mapped to method invocation-return
- **Progressively –**
  - looser coupling; more time-expensive interaction, coarser granuality
  - Each model builds on the previous one – uses it internally

Goals

- **Goals –**
  - An Appreciation of the role and context of
    - Web Services in general
    - The Web Services Resource Framework
    - Java-based tools
    - Structure of Standards
- **Structure**
  - Web Services
  - Outline of JAVA tools and their use
  - Standards Structure

O-O (JAVA) Web Services

Insert the web into the invoke and into the return



service consumer (client)

service provider (server)

WSDL – how and when

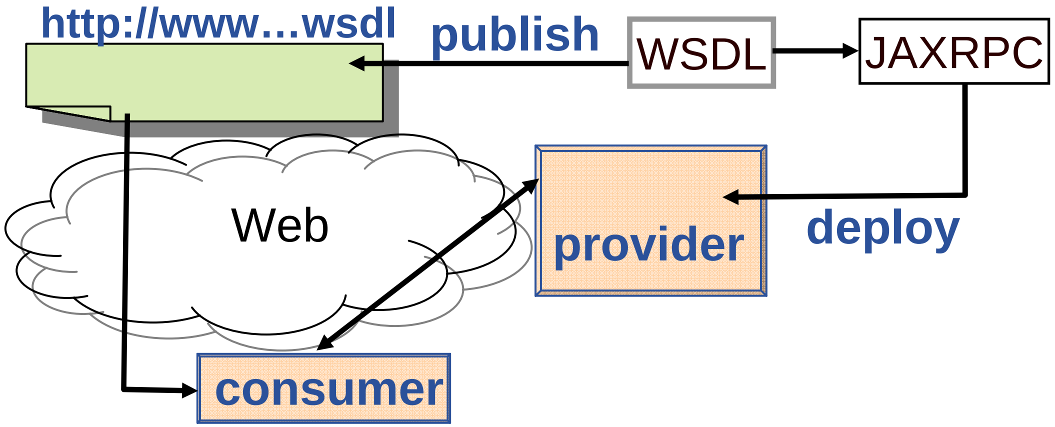
- **WSDL is keystone of web services –**
  - Defines a service’s **abstract** interface – operations and message structures
  - Defines the **physical** protocol and a data formants used to realise that abstract interface
  - The WSDL is published
  - Service consumer uses the WSDL to determine how to communicate with the service – stub generation
- **There are different approaches to how WSDL is**
  - **Produced**
    - Automatically
    - Manually
    - Semi-automatically
  - **Consumed**
    - Static Binding – access the WSDL at compile-time
    - Dynamic Binding – access the WSDL at run-time

WSDL Provider Options

Manual - Directly write WSDL

Unnatural for JAVA developer

WSDL is hard to write from scratch?



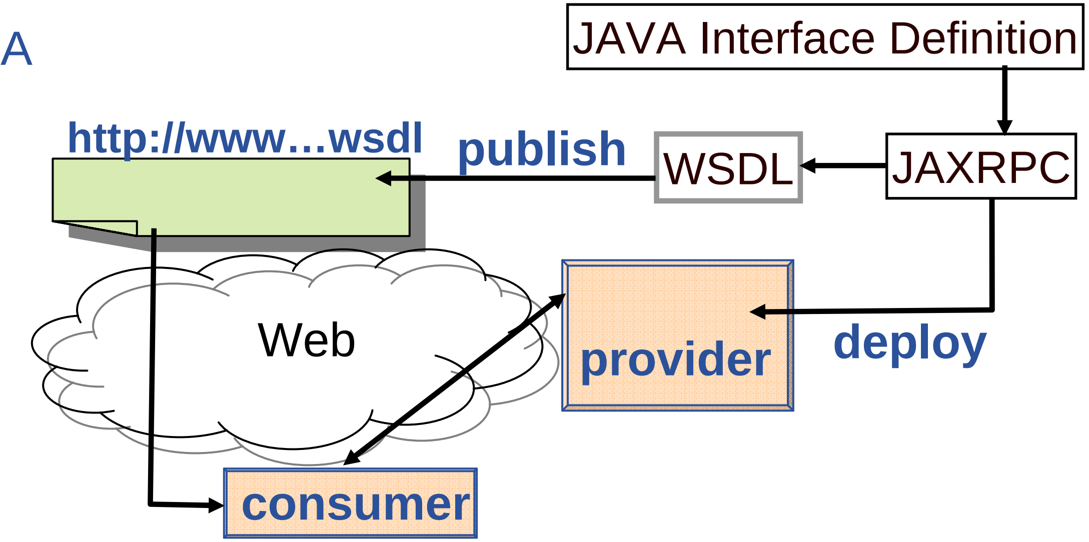
Automatic - Derive WSDL from JAVA

Natural for JAVA developer

May compromise interoperability

too JAVA specific

Won’t have extensibility



This is what you will do in the tutorial

WSDL Provider Options

Semi-Automatic –

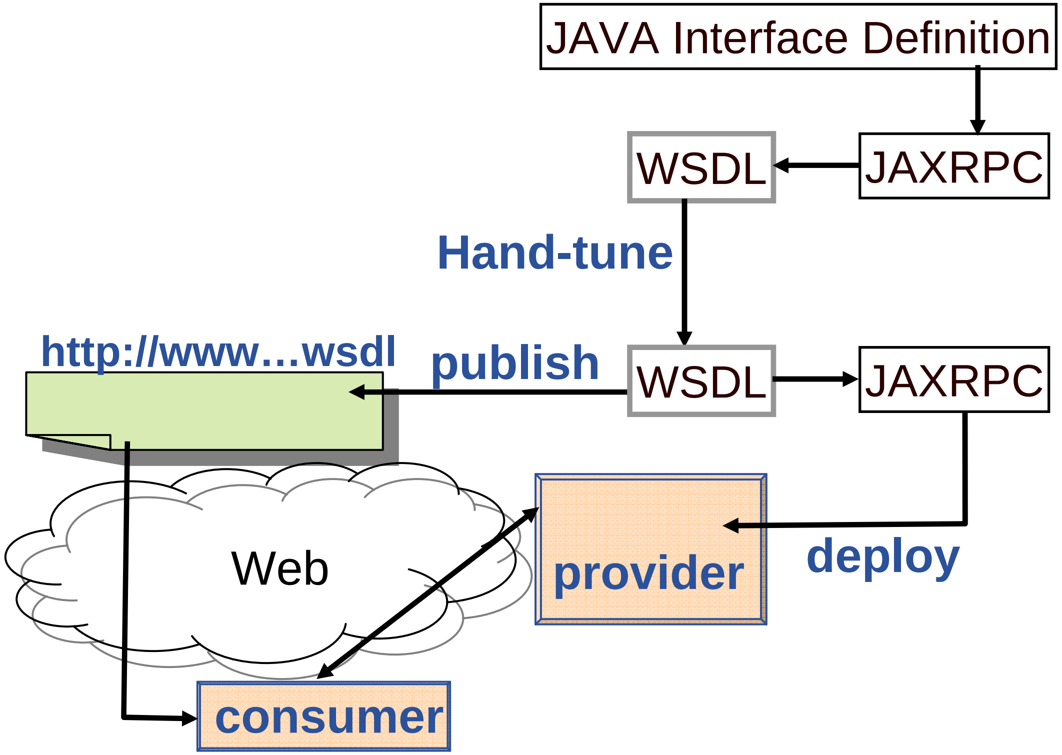
- Start with JAVA interface
- Derive initial WSDL
- Modify it for
  - interoperability
  - extensibility
- deploy (, test) and publish that

Natural for JAVA developer

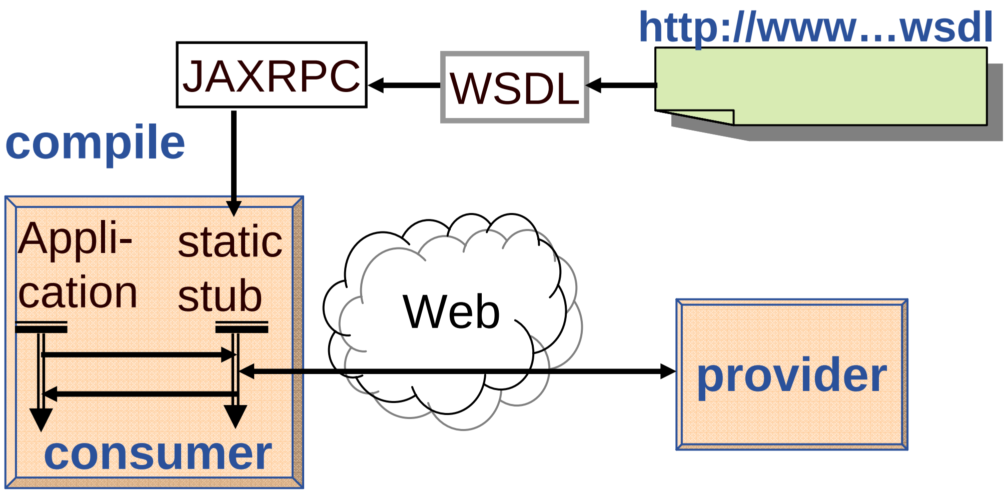
Addresses interoperability and extensibility

Development Issue

- For new version with extended interface, do you
  - start again from JAVA interface definition, and re-tune it all
  - work directly from the WSDL



WSDL – Consumer Options



Static Binding -

Manually obtain WSDL

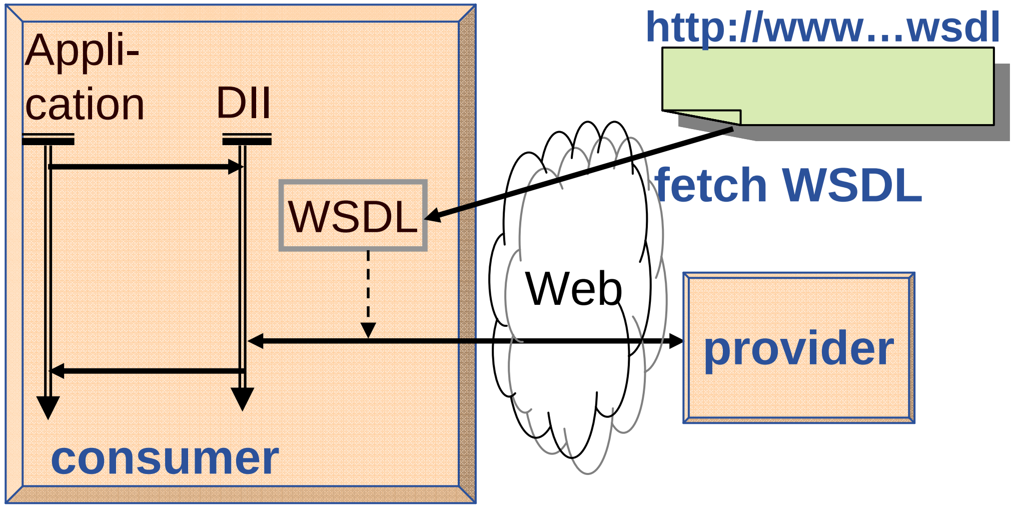
Use it to compile a stub

Which interacts with the service

Need to recompile if WSDL changes

Can test before going live

<u>This is what we will do in the Tutorial</u>



Dynamic Binding -

Via DII – Dynamic Invocation Interface

On each use of service,
get the WSDL

Use that to
construct message to the service
interpret the response message

Always using latest WSDL

But un-tested

Possibly inefficient

Goals

- **Goals –**
  - An Appreciation of the role and context of
    - Web Services in general
    - The Web Services Resource Framework
    - Java-based tools
    - Structure of Standards
- **Structure**
  - Web Services
  - Outline of JAVA tools and their use
  - Standards Structure

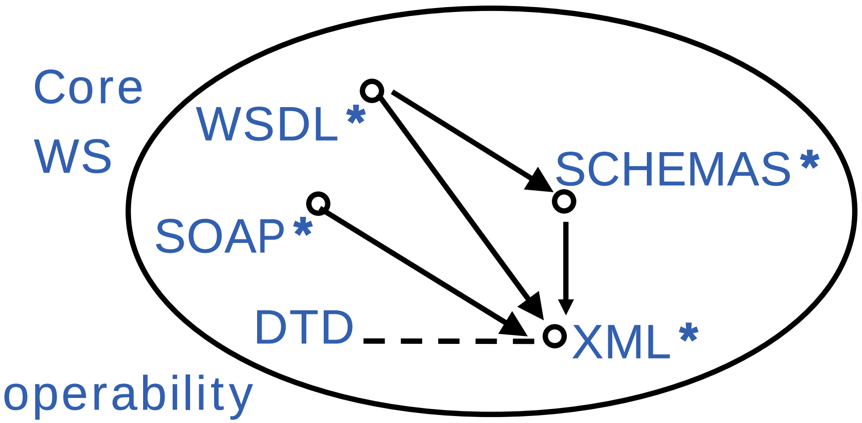
Flexible Standards

- Collaboration is on defining generic standards
- Two main standards bodies –
  - W3C – web community
    - actually produces “recommendations” not standards
  - OASIS – industry – IBM, Microsoft, Sun, ….
- These standards are factored to allow partial adoption and combination
  - The core standards
  - WS-I – clarifications to aid interoperability
  - Higher level standards built on them
- Take-up is pragmatic
- Standards have built-in extensibility
  - A standard typically defines some document –
    - E.g. The WSDL document for defining the interface to a service
    - The definition of the WSDL structure incorporates specific points where that structure can be extended

Core WS

- XML – the standard format for all information
- SCHEMA – the standard language for defining the structure (syntax/type) of a unit of information

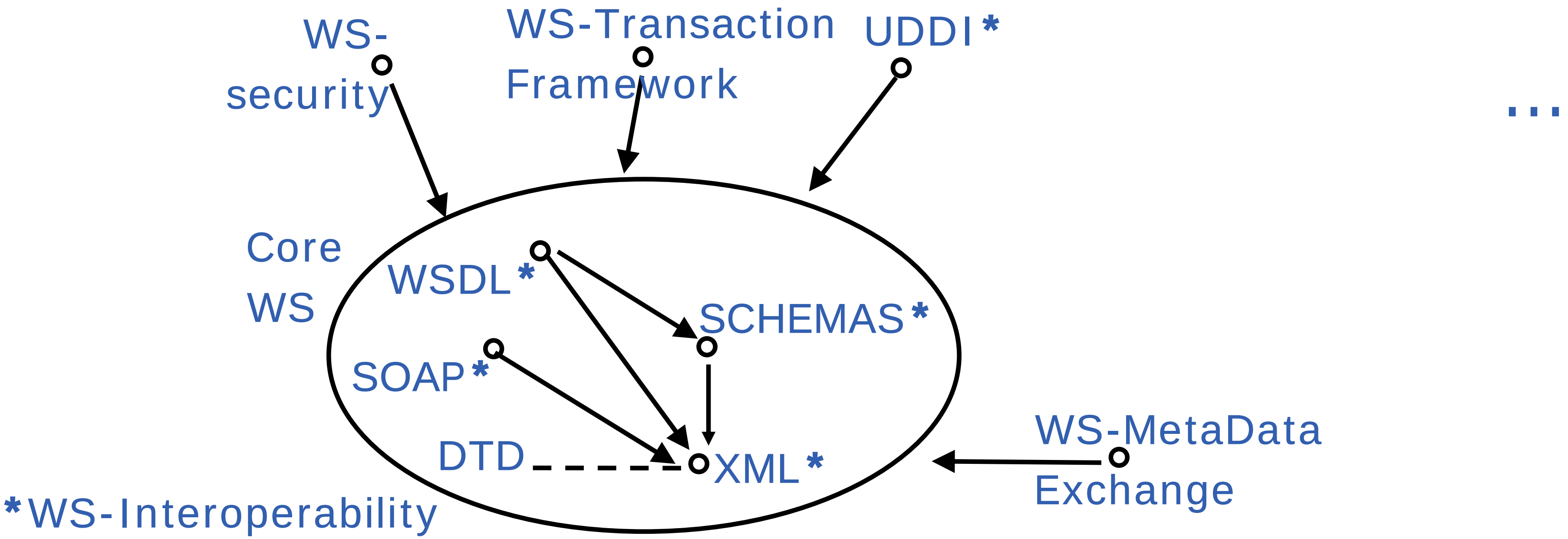
  DTD is a deprecated predecessor of Schemas
- SOAP – the standard message format
- WSDL – the language for defining a service –
  - Operations; Logical Message Structure; Bindings; locations



*WS-Interoperability

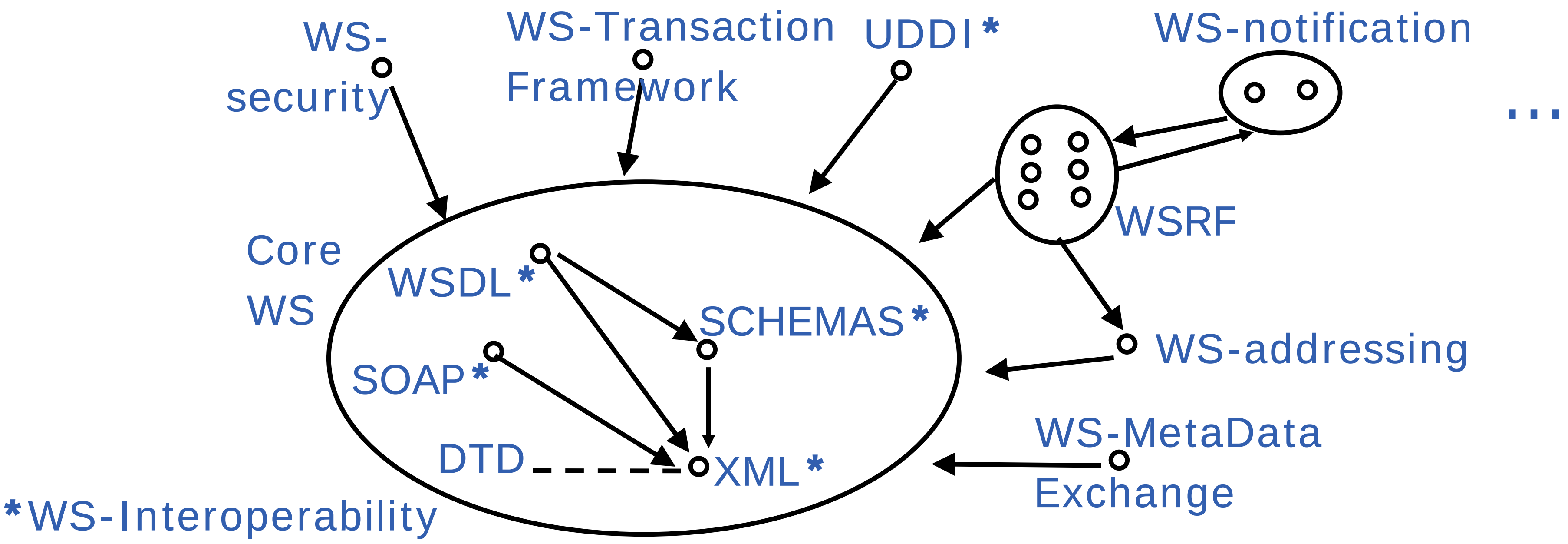
Some Further Standards

- **WS-Security – Framework for authentication and confidentiality**
- **WS-Transaction Framework – for robustness of correlated interactions, e.g. two phase – provisionally book everything, then confirm everything**
- **UDDI – standard repository interface (included in WS-I)**
- **WS-MetaDataExchange – how to communicate meta-data**
- **....**



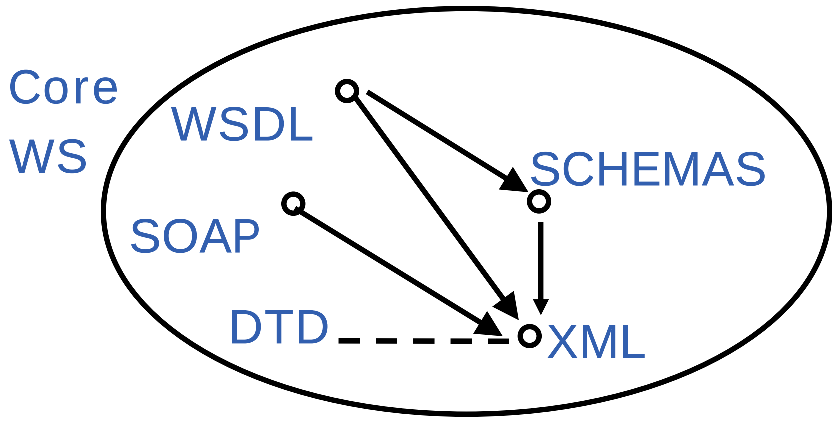
Some Further Standards

- WS-Addressing - For communication of identities between services
- WS-Notification - Framework of notification interaction – subscribe, publish
- WSRF – Web Services Resource Framework
  - Collection of standards concerning stateful dynamic resources –
    - E.g model a reservation as a stateful resource
    - Dynamically create new instance with extended lifetime



Remaining Talks

- **WSDL – the language for defining a service –**
  - This is our focus – this is what you will need to be able to read (and write)
  - This is what you will see in the practical
  - Will explain this in detail
- **Depends on**
  - SCHEMA – the standard language for defining the information structures
  - XML – the standard format for all information
  - SOAP – the standard message format – used in defining bindings
  - Will first explain these in outline – sufficient to understand a WSDL



The End

THE END