Semantic Web Services: Promise, Progress, Challenges

David Martin

SRI International

Chair, OWL-S Coalition

Co-chair, Semantic Web Services
Language Committee

DARPA Distribution Statement “A”: Approved for Public Release, Distribution Unlimited

Convergence on Services

- eBusiness need & vision, vendor investment
  - Interoperability; virtual organizations
  - ***Intra***nets, not just *inter*nets
  - Market prediction: $11 Billion in 2007 (IDC study)
- Standards efforts at W3C, OASIS, etc.
- Semantic Web community
  - OWL-S, SWSL, WSMO & other research efforts
- Grid computing
- Ubiquitous computing (devices; smart environments)
  - Mobile access to services

➔ A remarkable opportunity

- Creating a Web with computation, goals, processes as 1st-class citizens
- Bringing *behavioral intelligence* to the Web

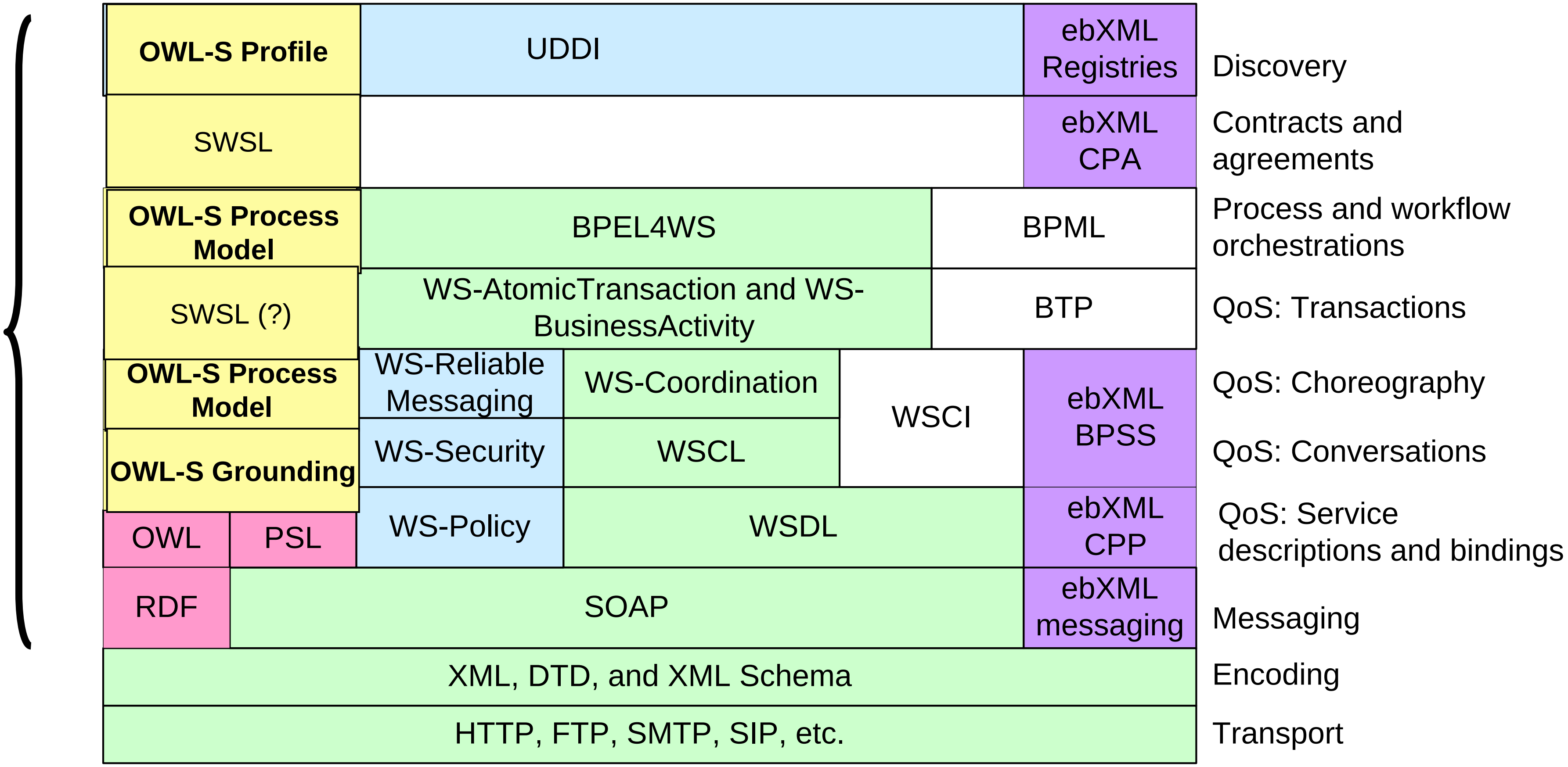
Convergence on Services

- eBusiness need & vision, vendor investment
  - Interoperability; virtual organizations
  - ***Intra***nets, not just *inter*nets
  - Market prediction: $11 Billion in 2007 (IDC study)
- Standards efforts at W3C, OASIS, etc.
- Semantic Web community
  - OWL-S, SWSL, WSMO & other research efforts
- Grid computing
- Ubiquitous computing (devices; smart environments)
  - Mobile access to services

➔ A remarkable opportunity

  - Creating a Web with computation, goals, processes as 1st-class citizens
  - Bringing *behavioral intelligence* to the Web

Towards a Service-enabled Web



OWL-S Profile | UDDI | ebXML Registries | Discovery

SWSL | ebXML CPA | Contracts and agreements

OWL-S Process Model | BPEL4WS | BPML | Process and workflow orchestrations

SWSL (?) | WS-AtomicTransaction and WS-BusinessActivity | BTP | QoS: Transactions

OWL-S Process Model | WS-Reliable Messaging | WS-Coordination | WSCI | ebXML BPSS | QoS: Choreography

OWL-S Grounding | WS-Security | WSCL | QoS: Conversations

OWL | PSL | WS-Policy | WSDL | ebXML CPP | QoS: Service descriptions and bindings

RDF | SOAP | ebXML messaging | Messaging

XML, DTD, and XML Schema | Encoding

HTTP, FTP, SMTP, SIP, etc. | Transport

SWS: towards an expressive, comprehensive, unified framework for reasoning about services

Derived From M. Singh and M. Huhns: *Service-Oriented Computing: Semantics, Processes, Agents*

Outline

- **Background & Vision ←**
  - **Web Services**
  - **Semantic Web**
  - **Semantic Web Services**
- OWL-S: Ontology-Based Semantics for WS
- SWSL: Building Out
- Service Management Tasks & Tools
- Applications
- Related Work
- Challenges & Next Steps

Warning:
Whirlwind
tour

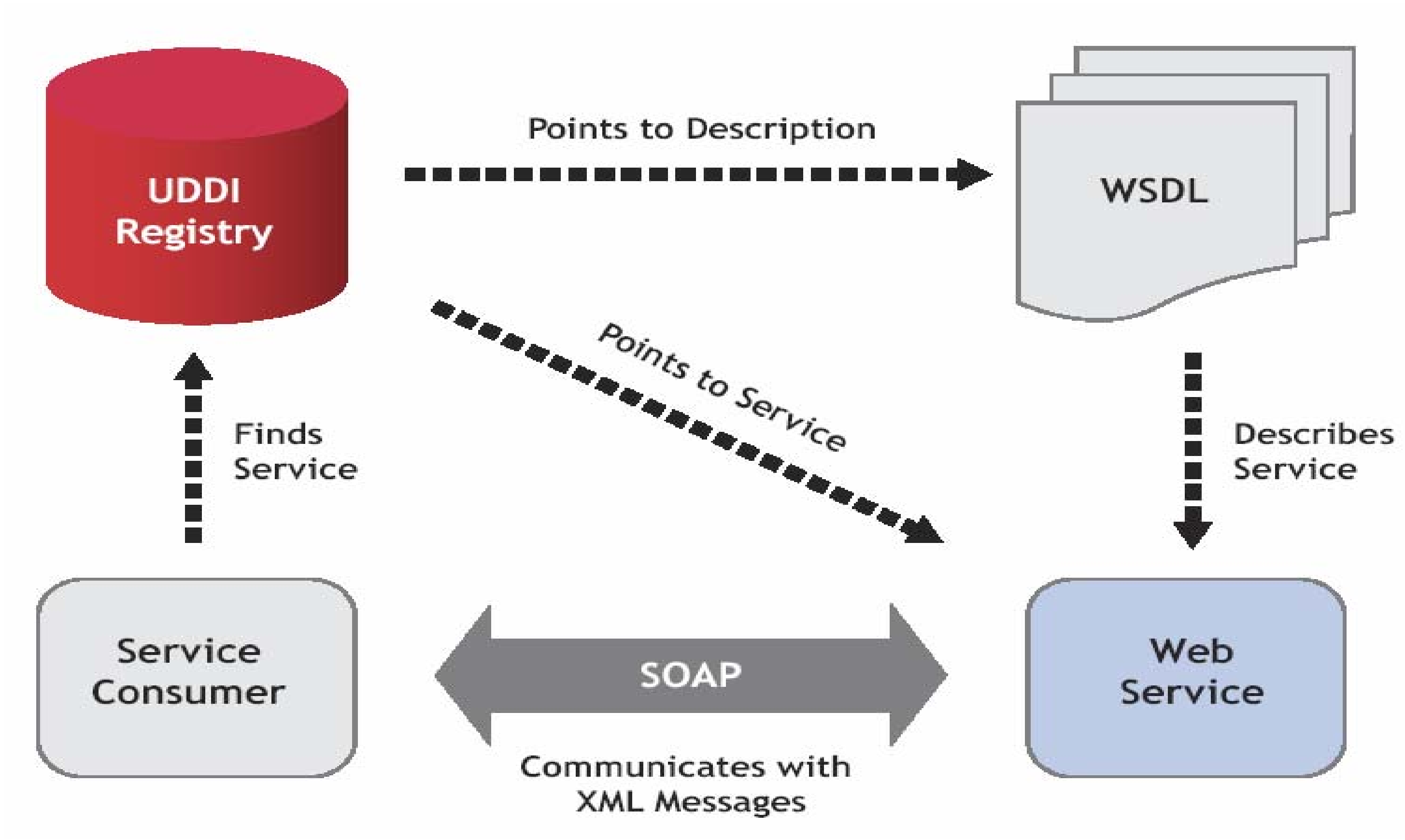
Web Services: The Essence

- "Loosely coupled software components that interact with one another dynamically via standard Internet technologies" (Gartner)
- Reliable, ubiquitous software interoperability
  - Across networks
  - Across organizations
  - Non-proprietary standards
  - Need: ~ half of IT activities and costs relate to integration
- Focus on communications; content exchange
  - Basic infrastructure & tools

WS: The Broader Vision

- Widely distributed, decentralized business services
- Accessible from a wide variety of platforms & devices
- The Internet as a global platform where organizations and individuals carry out commercial transactions
- Virtual organizations
- Dynamic enterprise; dynamic value chains
- Tools support the creation and enactment of individual services and collaborative processes
- Requesters have access to many providers; can quickly and easily find one and establish a working relationship
- Singh & Huhns: “When new techniques improve the reaction times of organizations and people from weeks to seconds, they change the very structure of business. This is not a mere quantitative change, but a major qualitative change.”

WS: Basic Building Blocks

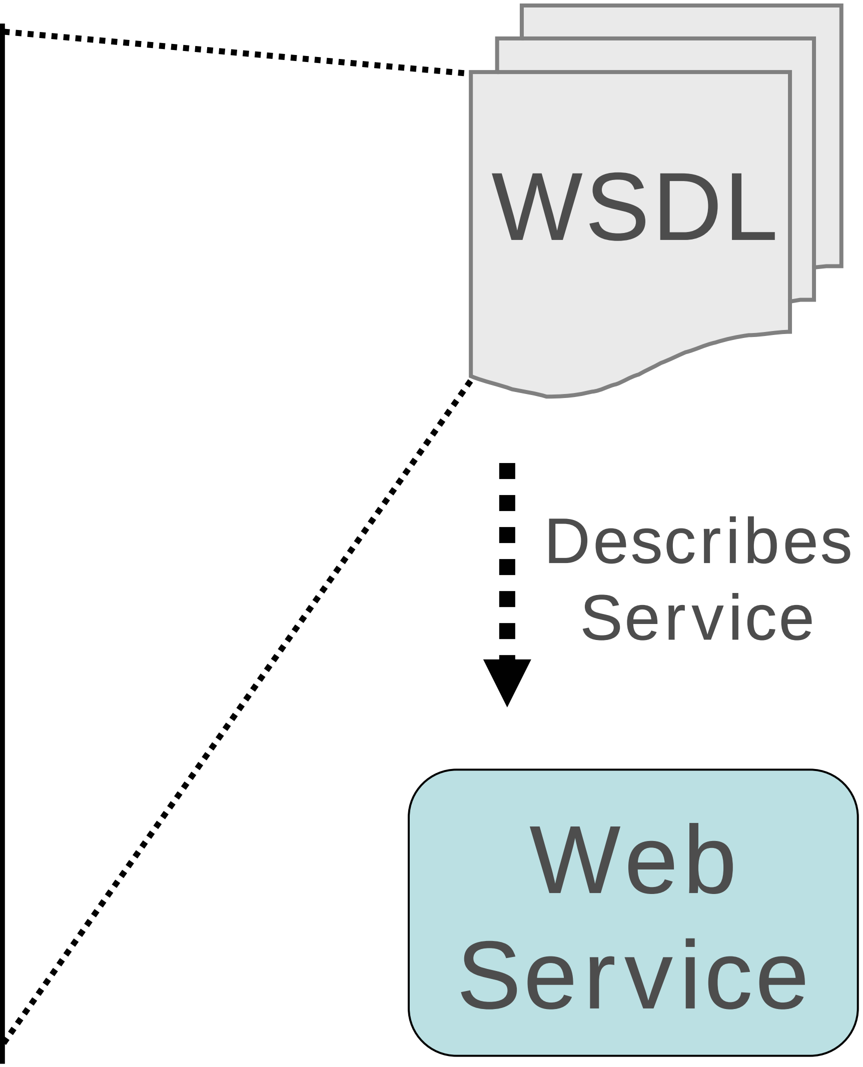


Web Services Description Language

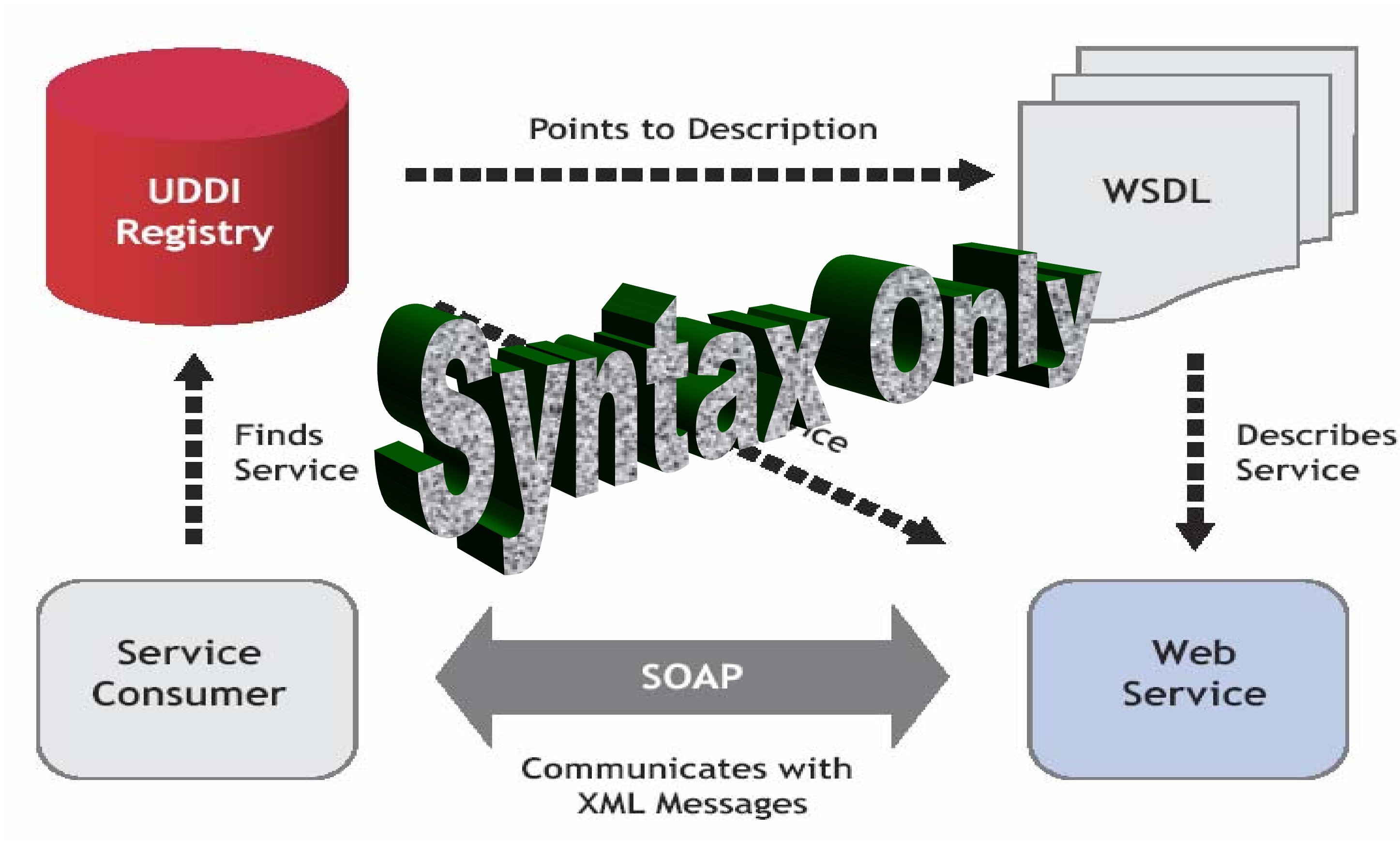
I can receive a message having this form …

And I will reply with a message having this form …

On port 5552, using HTTP transport, SOAP format



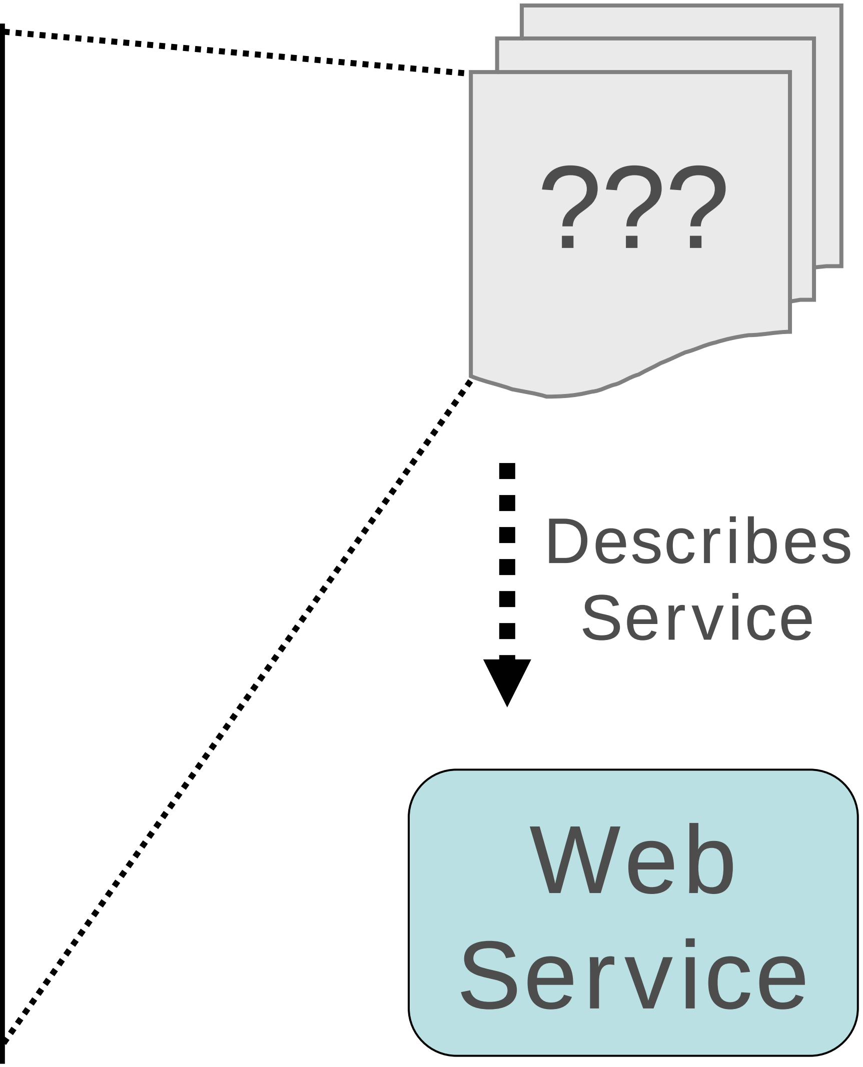
So What's the Problem?



Semantics Needed

To use this service you must be a member of AAA.

If you've been a member for 3 or more years, you get a 15% discount.

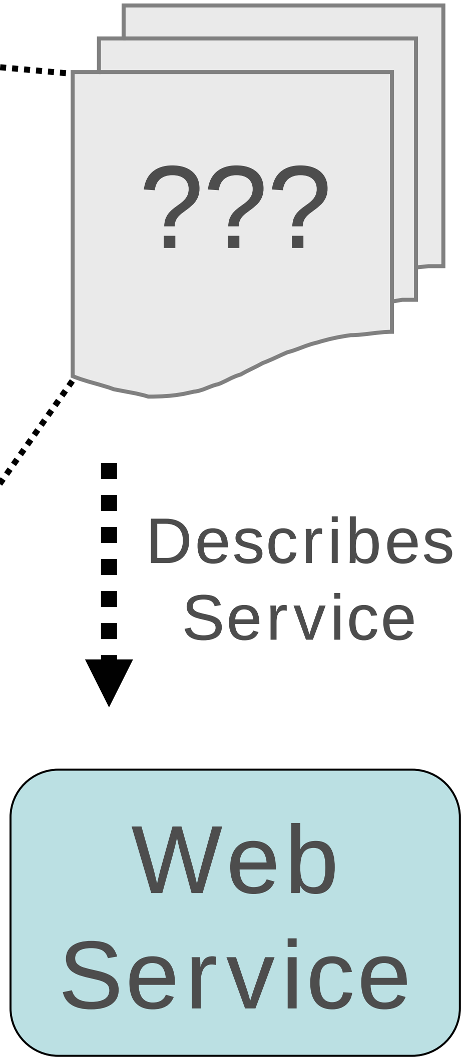


Semantics Needed

When you access this service, you may use TLS or WS-Security.

WS-Security is preferred.

Using TLS costs $9; using WS-Security $15.



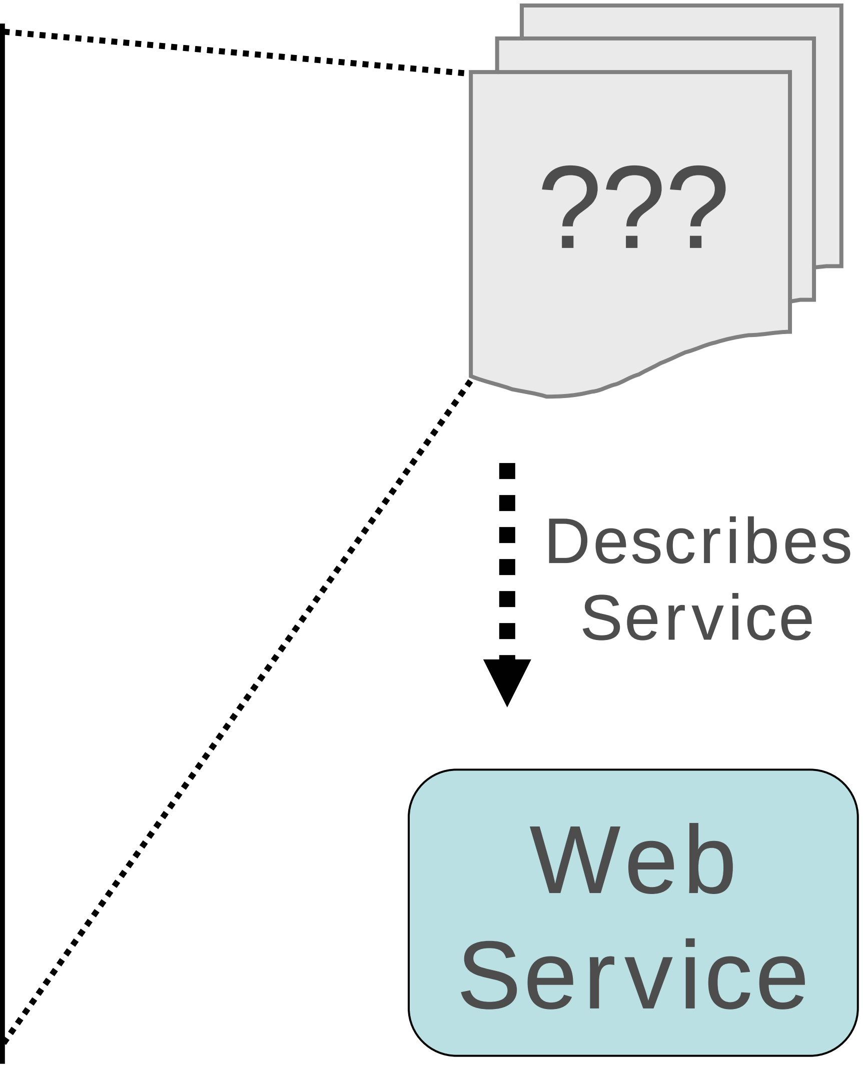
???

Describes Service

Web Service

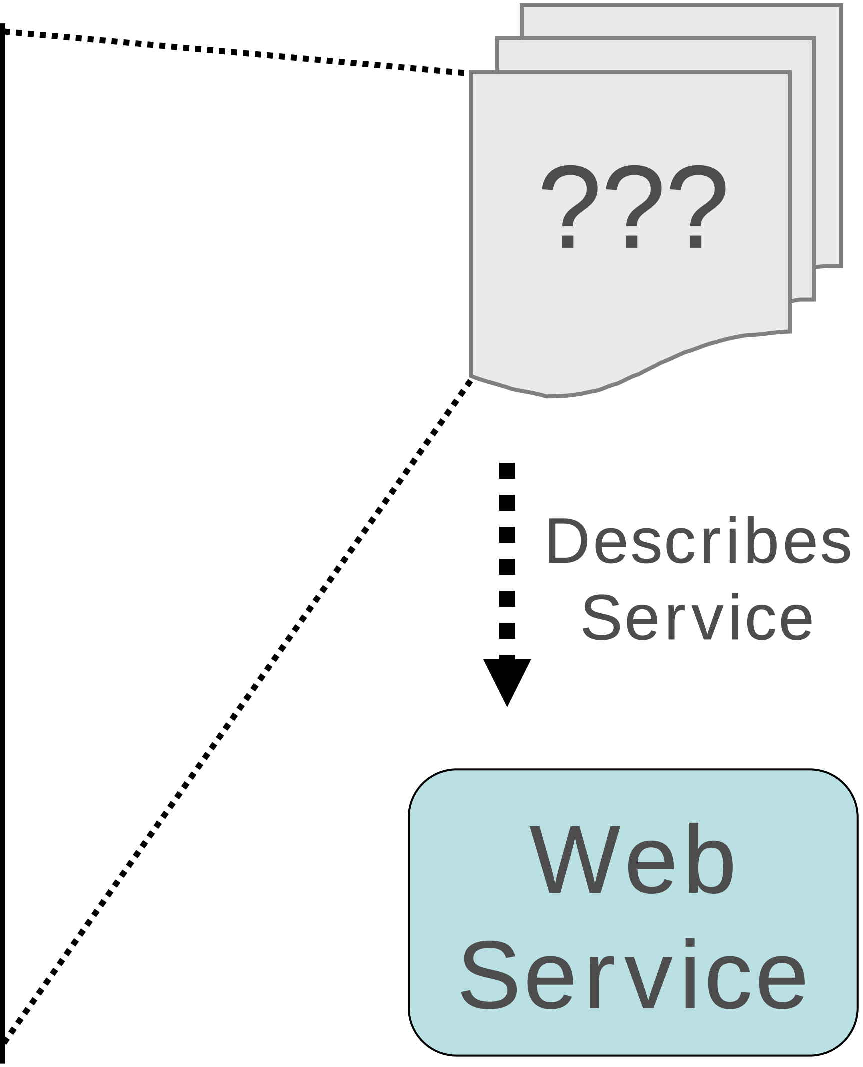
Semantics Needed

If I fail to deliver this item within 7 days, I will pay a 30% penalty.



Semantics Needed

You can only access this information if you agree to make changes to it freely available.

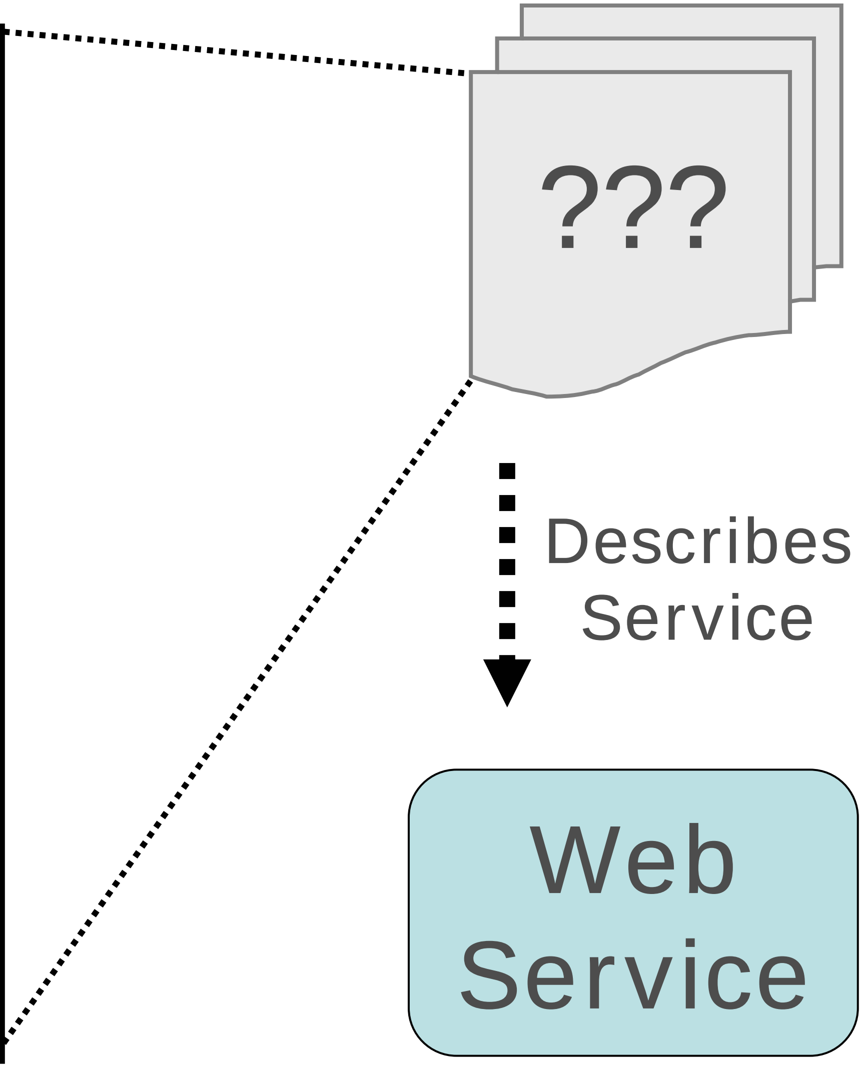


Semantics Needed

I will arrange for the requested book to be shipped to you

and

I will debit your credit card account for the listed price



What is the Semantic Web?

- A Vision
- A Research Area
- A Set of Standards Activities at the World Wide Web Consortium
- A Collection of Languages, Reasoners, and Tools
- A Growing Collection of *Ontologies* and Knowledge Bases (World-Wide)
- A Set of Shared Representations, Collaborative Activities and Communities Forming Around Them

What is the Semantic Web?

"The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation."

-- Tim Berners-Lee, James Hendler, Ora Lassila, The Semantic Web, Scientific American, May 2001

DAML In One Chart

Problem**:**

- **Computers cannot process most of the information stored on web pages**

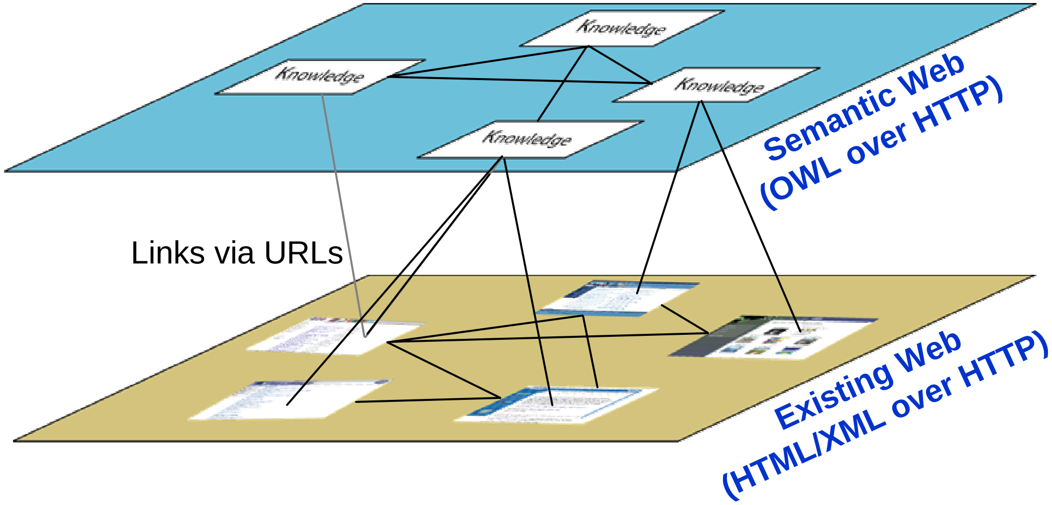
Solution**:**

- **Augment the web to link machine-readable knowledge to web pages**
  - Extend RDF with Description Logic
  - Use a frame-based language design
  - Create the first fully distributed web-scale knowledge base out of networks of hyperlinked facts and data

Approach**:**

- **Design a family of new web languages**
  - Basic knowledge representation (OWL)
  - Reasoning (SWRL, OWL/P, OWL/T)
  - Process representation (OWL/S)
- **Build definition and markup tools**
- **Link new knowledge to existing web page elements**
- **Test design approach in the Intelligence Community and others**
- **Standardize the new web languages**

Computers require explicit knowledge to reason with web pages



Knowledge

Knowledge

Knowledge

Knowledge

Semantic Web (OWL over HTTP)

Links via URLs

Existing Web (HTML/XML over HTTP)

People use implicit knowledge to reason with web pages

Courtesy of Mark Greaves, DARPA

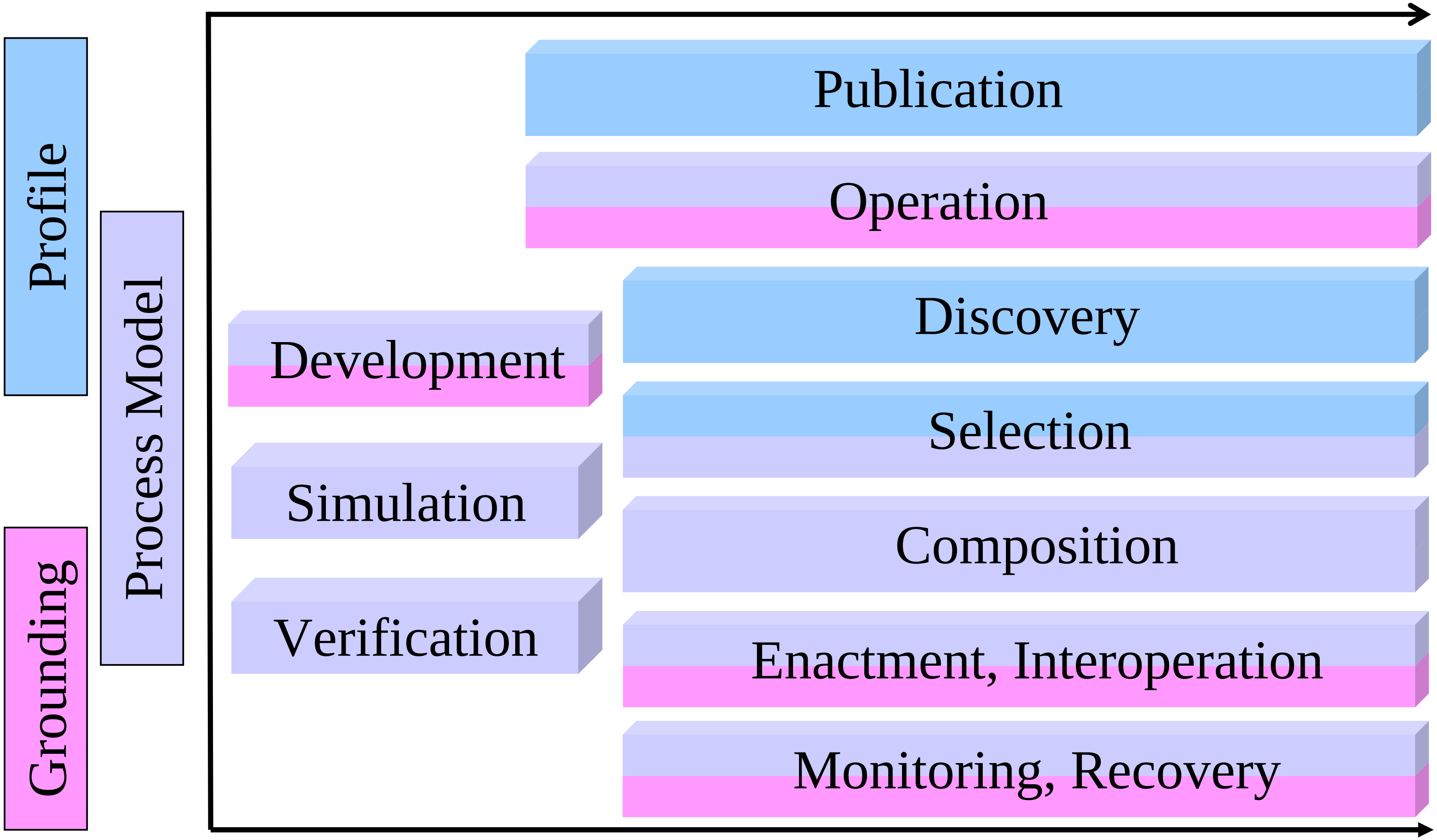
Why Semantic Web Services?

- Automation of service use by software agents
  - Ideal: full-fledged use of services never before encountered:
- Enable reasoning/planning about services
  - e.g., On-the-fly composition
- Comprehensive framework supporting the entire lifecycle of service management tasks
    - Discovery, selection, composition, invocation, monitoring, ..
- Integrated use with information resources
- Ease of use (for users and developers)
- Powerful tools

Automation Envisioned by SWS

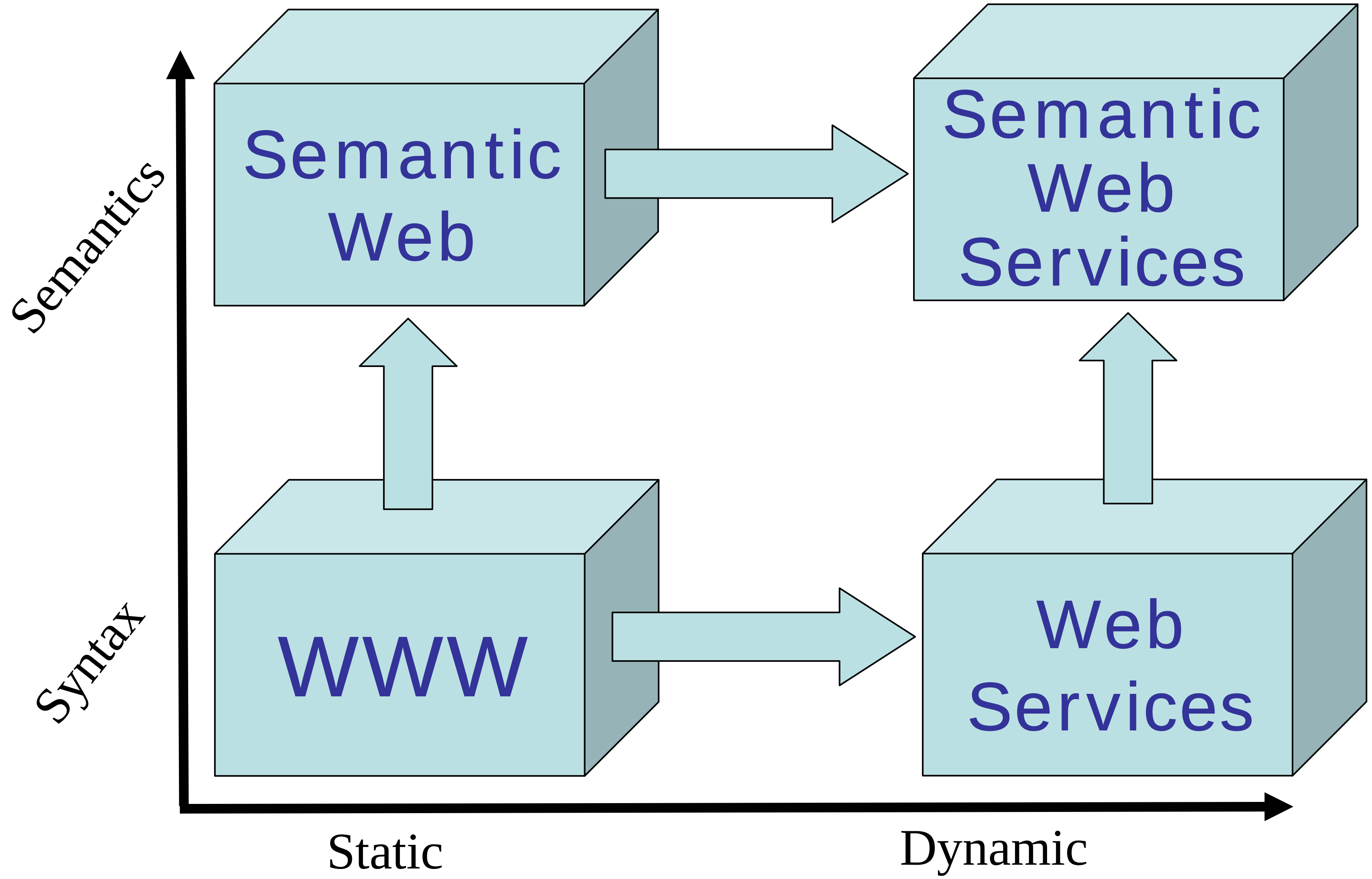
- Web service discovery
  - *Find me a shipping service that transports goods to Dubai.*
- Web service enactment
  - *Buy me 500 lbs. powdered milk from www.acmemoo.com*
- Web service selection & composition
  - *Arrange food for 500 people for 2 weeks in Duba*
- Web service execution monitoring
  - *Has the powdered milk been ordered and paid for yet?*

Web Service Mgmt. Tasks

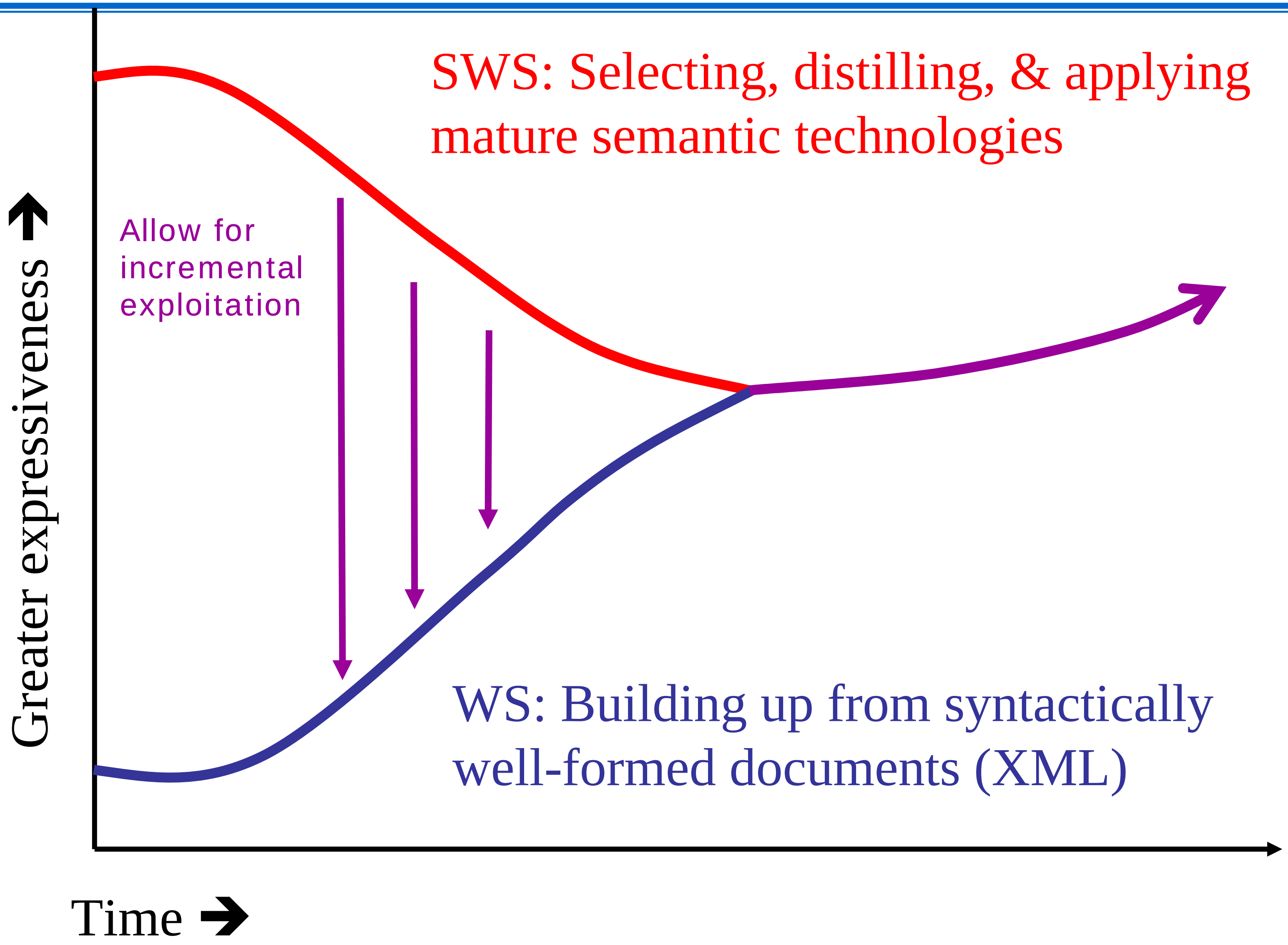


Development … Deployment … Use …

Evolution Towards Semantic Web Services



Convergence of WS and SWS



SWS: Selecting, distilling, & applying mature semantic technologies
Allow for incremental exploitation
Greater expressiveness →
WS: Building up from syntactically well-formed documents (XML)
Time →

Summary: WS + SW = SWS

- WS is providing the building blocks for
  - Constructing, publishing, finding, interoperating with behavioral building blocks --- mostly *manually*
  - Encapsulation, reuse, Web access
  - Lightweight, language- and vendor-neutral deployment
- *WS makes available a vast global repository of interoperable services / procedures / devices*
  - *But labor-intensive, expertise-intensive*
- SW provides infrastructure and technology for *reasoning* about this world of services
  - Using services more effectively
  - Providing better tools; building more robust services
  - Fuller *automation* of service use by software agents
  - Support for non-expert developers, end-users
- *WS + SW allows us to create a powerful Web of “behavioral intelligence”*

Outline

- Background & Vision
- **OWL-S: Ontology-Based Semantics for WS 🡐**
  - **Background & Motivation**
  - **Profile, Process, Grounding (subontologies)**
  - **Status**
- SWSL: Building Out
- Service Management Tasks & Tools
- Applications
- Related Work
- Challenges & Next Steps

Contributors to OWL-S
(partial list)

BBN: Mark Burstein

CMU: Katia Sycara, Massimo Paolucci, Naveen Srinivasan

De Montfort University: Monika Solanki

Maryland / College Park: Bijan Parsia, Evren Sirin

NIST: Craig Schlenoff

Nokia: Ora Lassila

SRI: David Martin

Stanford KSL: Deb McGuiness

Southampton: Terry Payne

Univ. of Toronto: Sheila McIlraith

USC-ISI: Jerry Hobbs

Yale: Drew McDermott

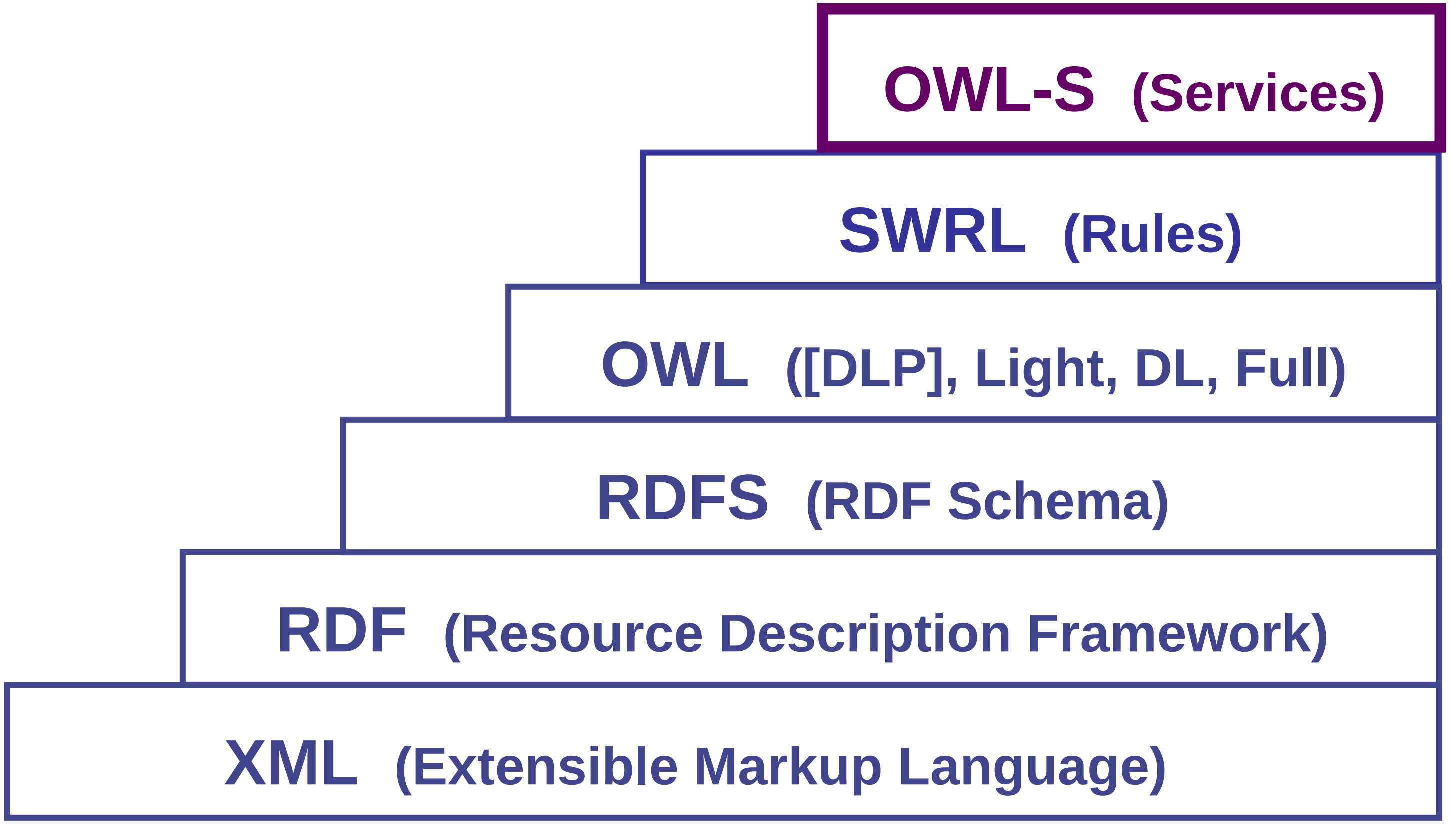
What is OWL-S?

- Ontology Web Language for Services
- An OWL ontology/language for (formally) describing properties and capabilities of Web services
- An approach that draws on many sources
  - Description logic
  - AI planning
  - Workflow
  - Formal process modeling
  - Agents
  - Web services

http://www.daml.org/services/owl-s

Layered Approach to Language Development

OWL-S: an ontology expressed in OWL and related languages



OWL-S (Services)

SWRL (Rules)

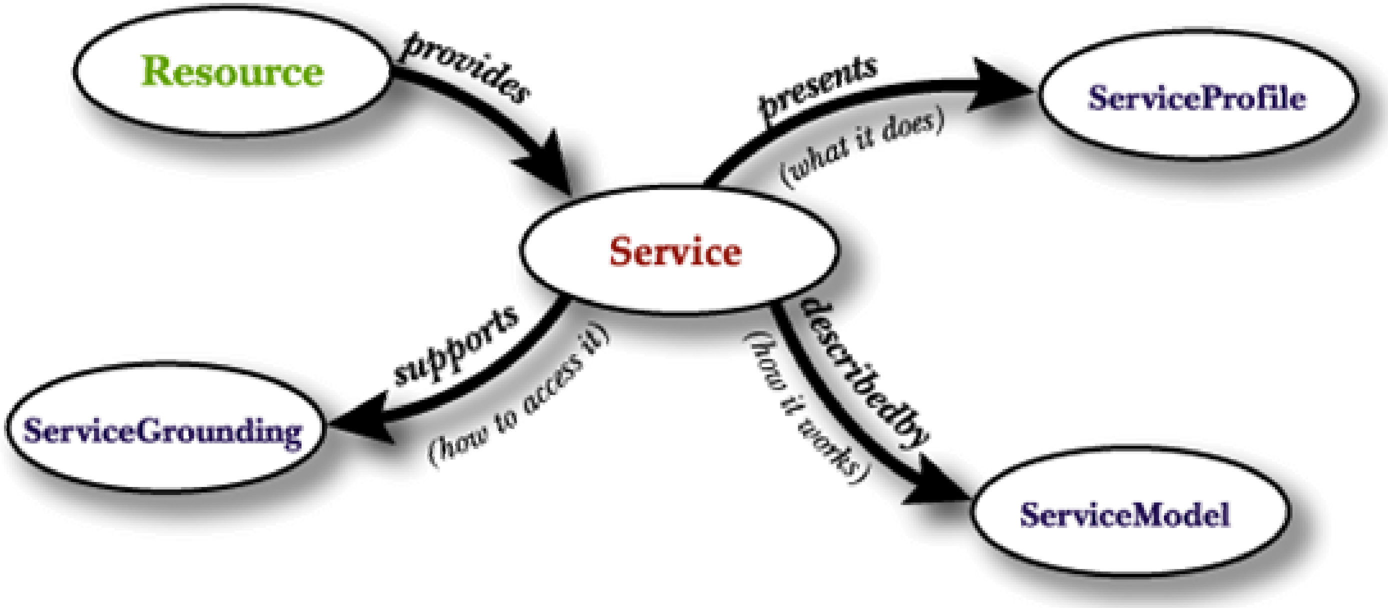
OWL ([DLP], Light, DL, Full)

RDFS (RDF Schema)

RDF (Resource Description Framework)

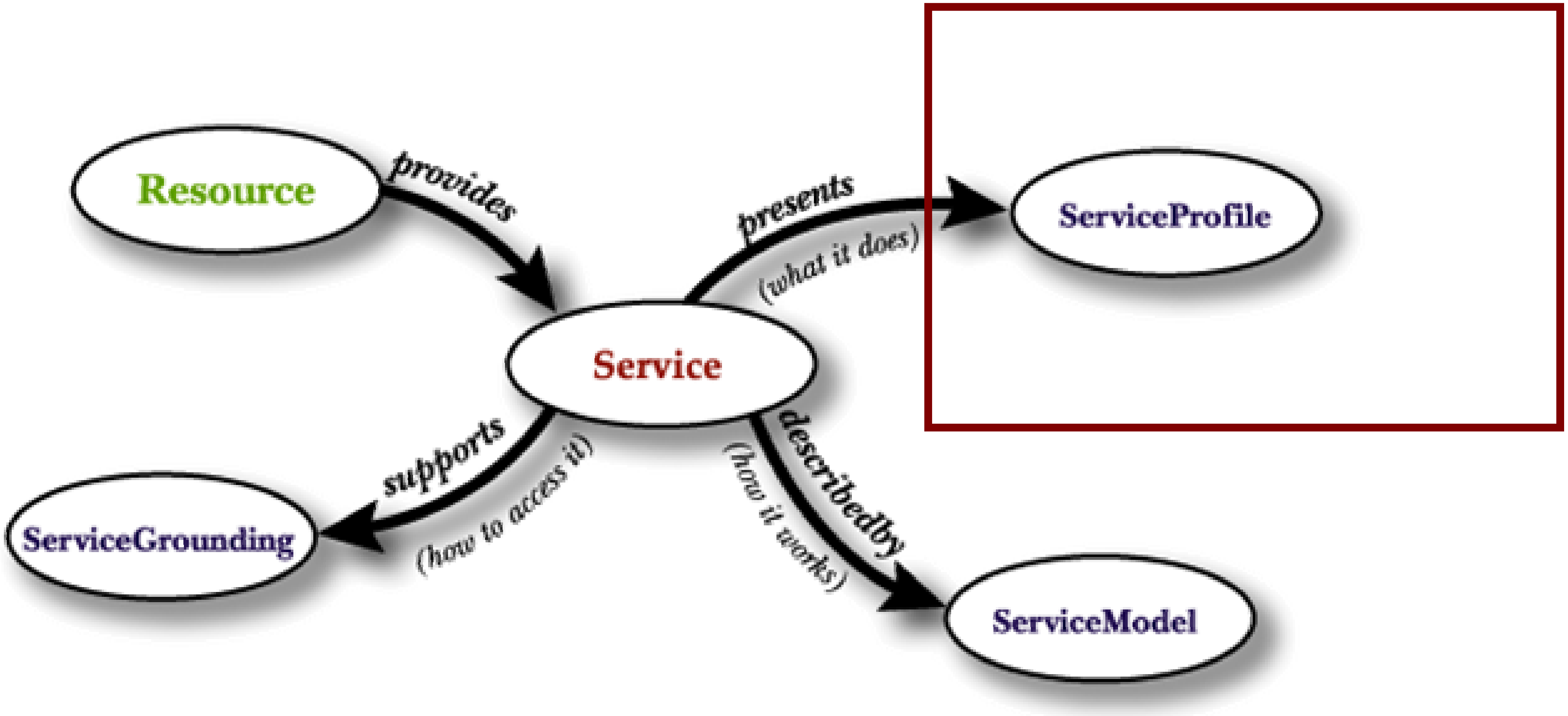
XML (Extensible Markup Language)

Upper Ontology of Services



Ontology images compliments of Terry Payne, University of Southampton

Upper Ontology of Services



Ontology images compliments of Terry Payne, University of Southampton

Service Profile: “What does it do?”

High-level characterization/summary of a service

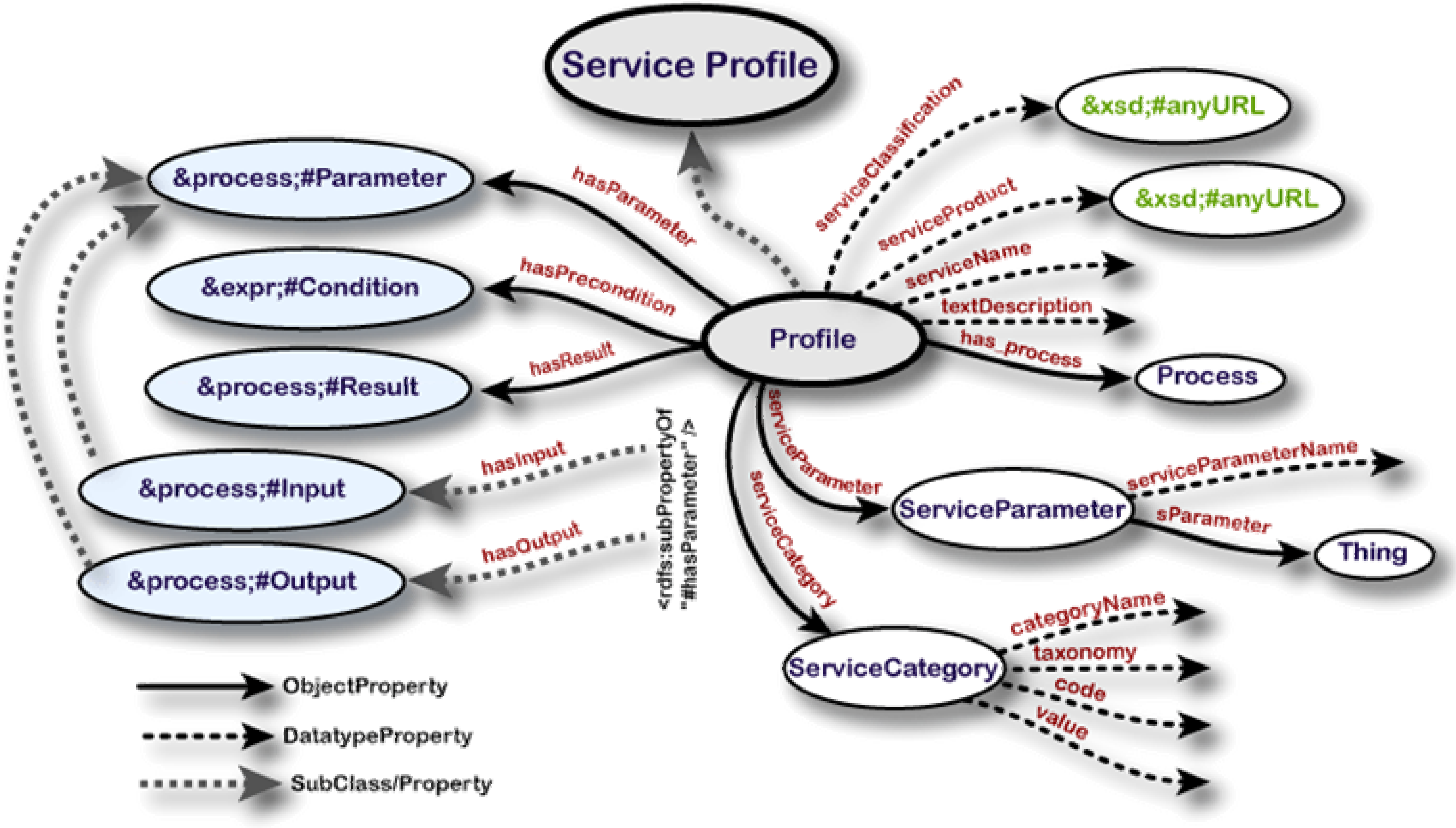
Used for

- Populating service registries
  - A service can have many profiles
- Automated service discovery
- Service selection (matchmaking)

One can derive:

- Service advertisements
- Service requests

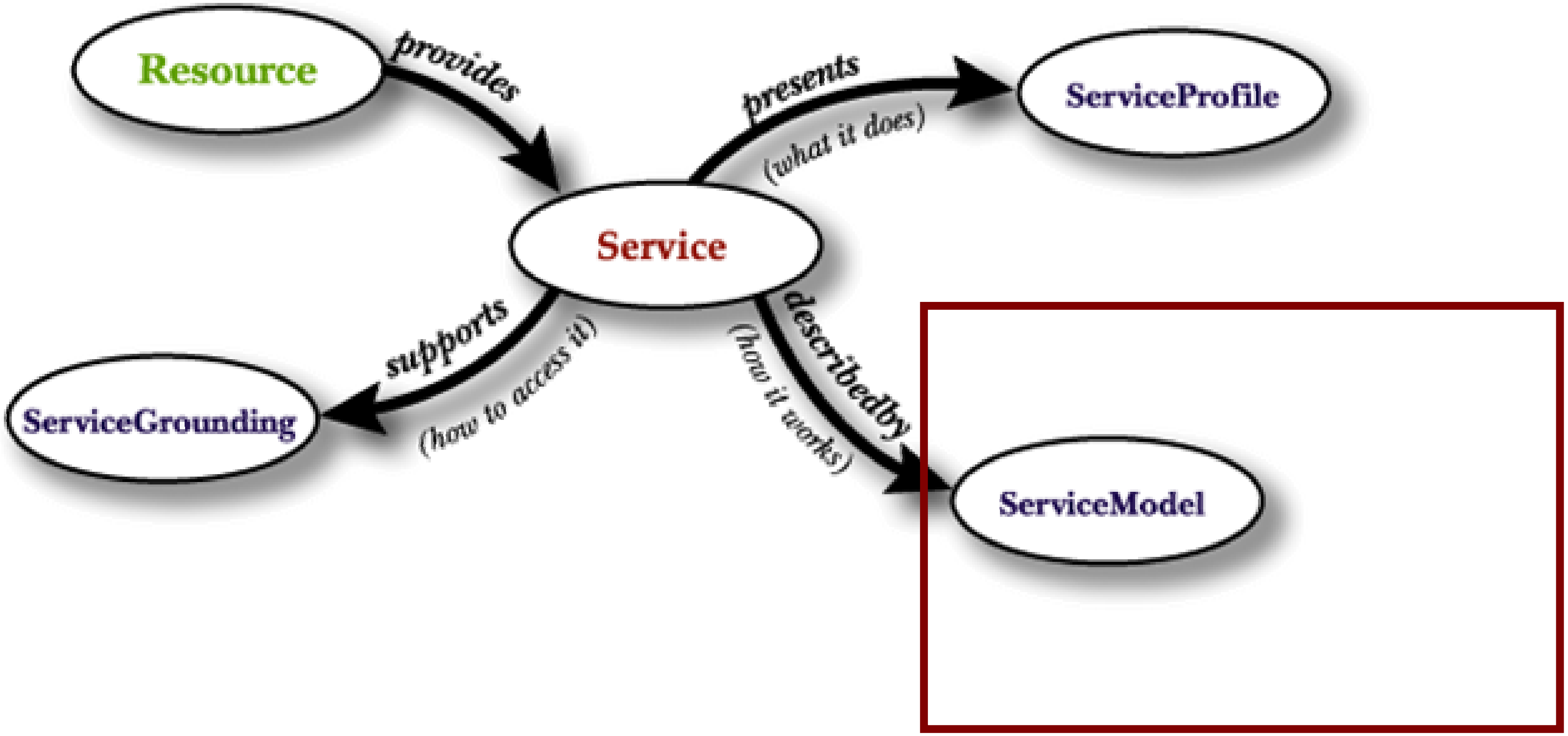
Service Profile



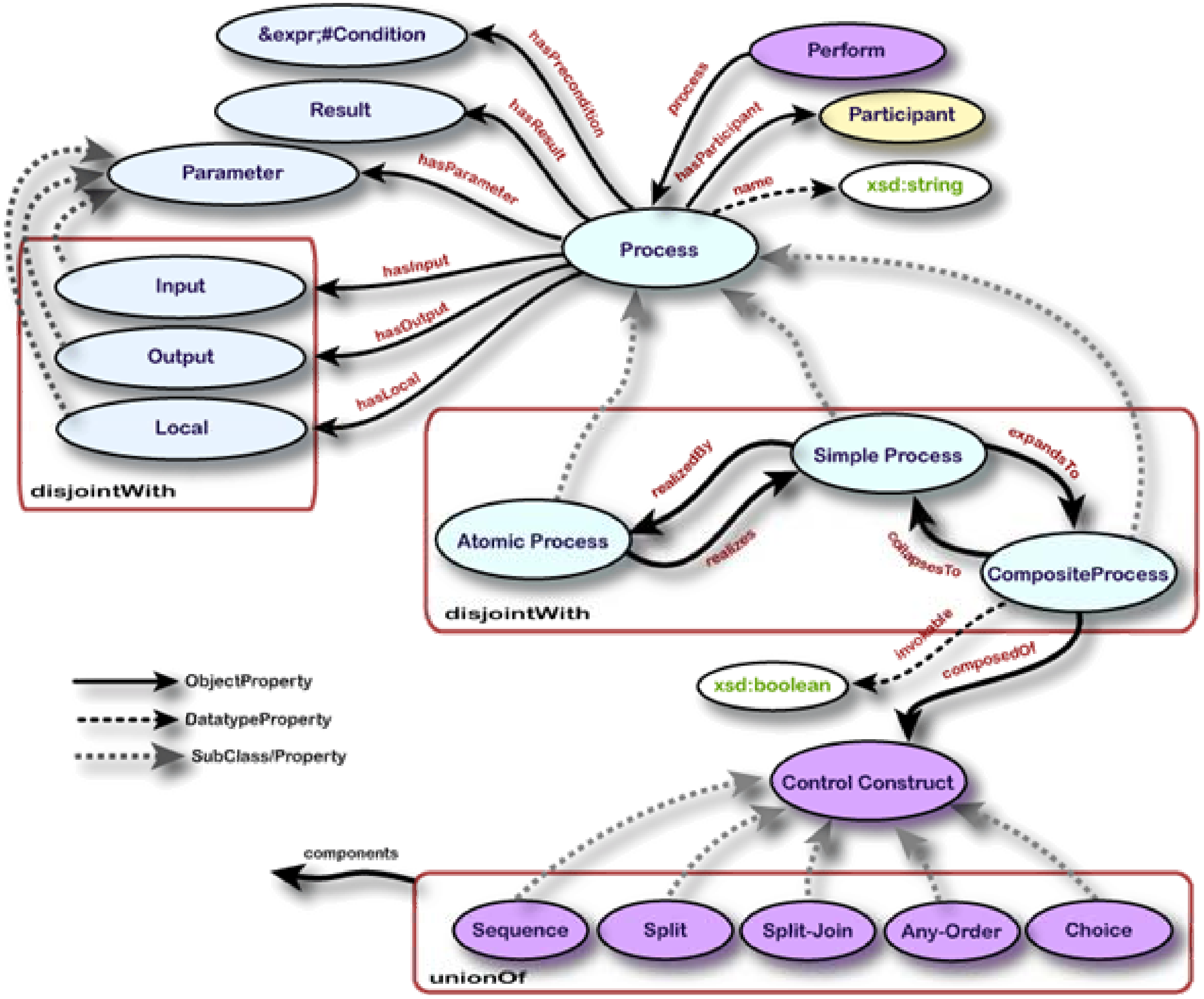
Service Profile: Styles of use

- Class hierarchical yellow pages
  - Implicit capability characterization
  - Arrangement of attributes on class hierarchy
  - Can use multiple inheritance
  - Relies primarily on “non-functional” properties
- Process summaries for planning purposes
  - More explicit
  - Inputs, outputs, preconditions, effects
  - Less reliance on formal hierarchical organization
  - Summarizes process model specs
  - Relies primarily on functional description

Upper Ontology of Services



Process Model



Process Model: “How does it work?”

Process

- Interpretable description of service provider’s behavior
- Tells service user how and when to interact (read/write messages)

& Process control

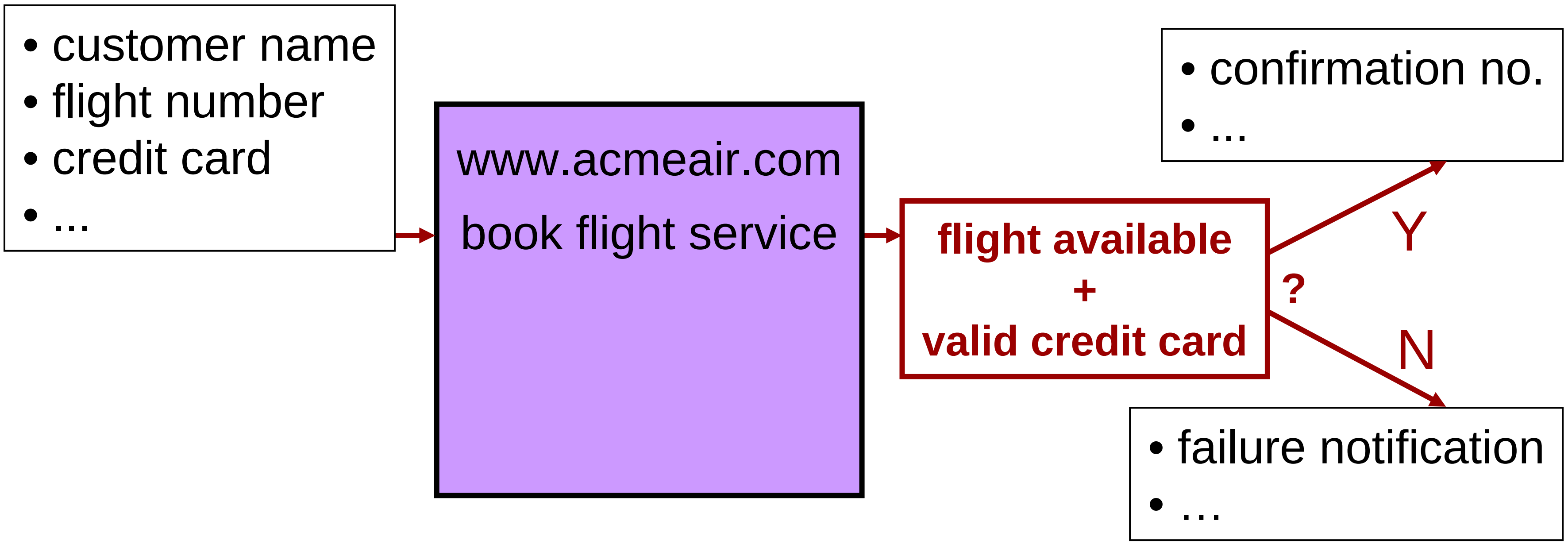
- Ontology of process state; supports status queries
- (stubbed out at present)

- Used for:
  - Service invocation, planning/composition, interoperation, monitoring
- All processes have
  - Inputs, outputs, preconditions and effects
  - Function/dataflow metaphor; action/process metaphor
- Composite processes
  - Control flow
  - Data flow

Function/Dataflow Metaphor

Input:

Output:



- customer name
- flight number
- credit card
- ...

www.acmeair.com

book flight service

flight available
+
valid credit card

?

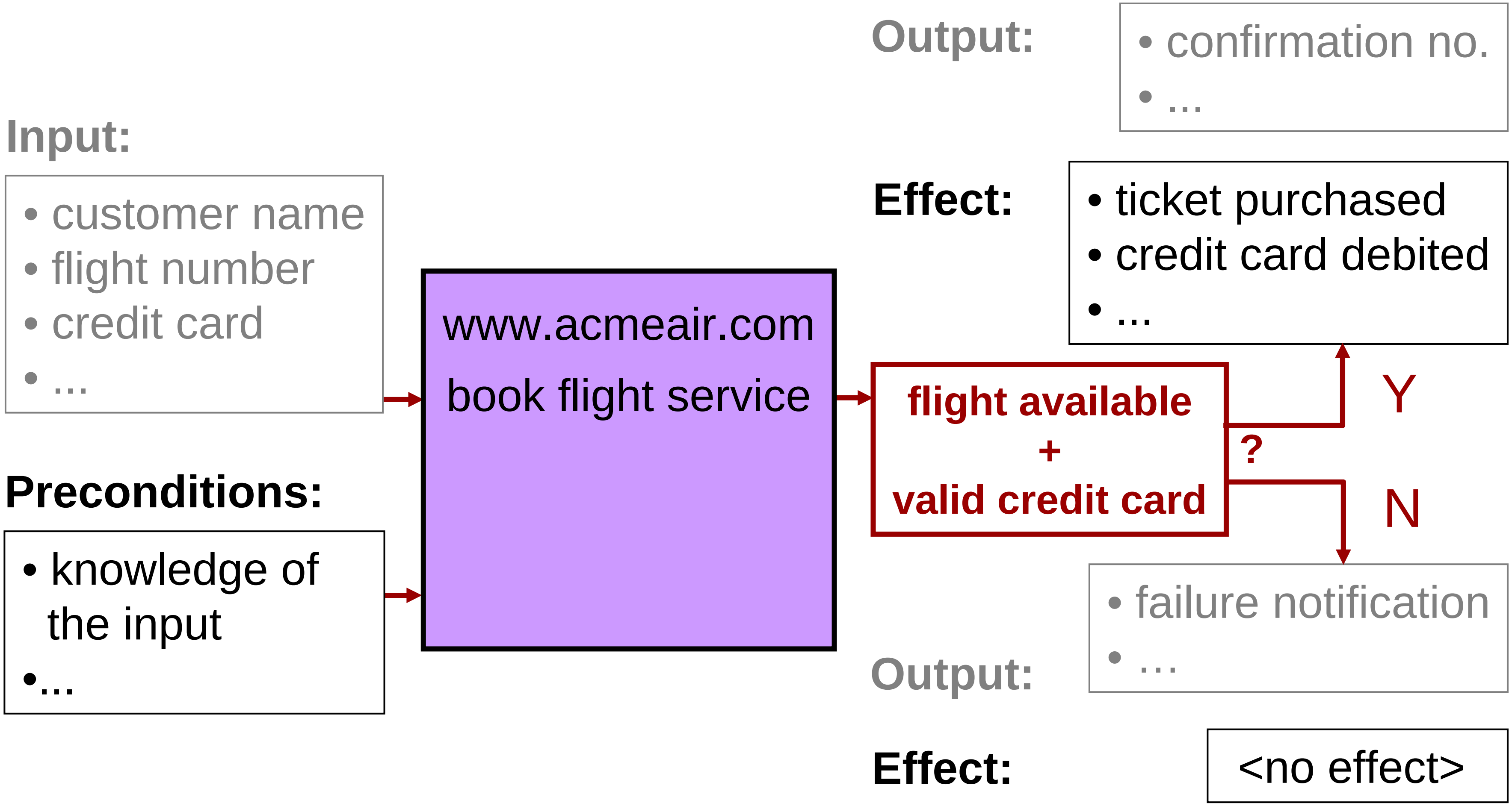
Y

- confirmation no.
- ...

N

- failure notification
- …

AI-inspired Action/Process Metaphor



Input:

- customer name
- flight number
- credit card
- ...

Preconditions:

- knowledge of the input
- ...

www.acmeair.com

book flight service

flight available

+

valid credit card

?

Y

Output:

- confirmation no.
- ...

Effect:

- ticket purchased
- credit card debited
- ...

N

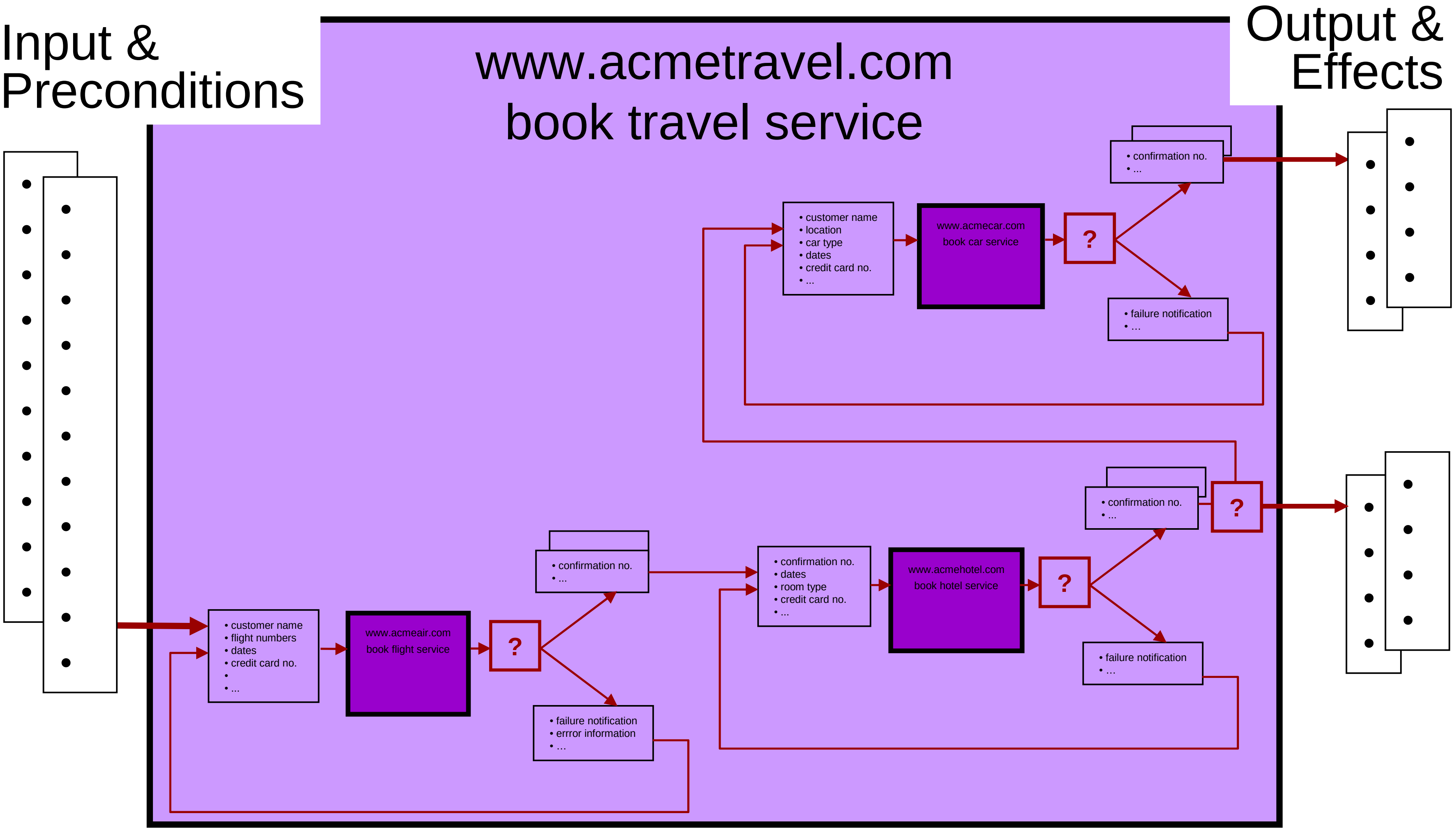
Output:

- failure notification
- ...

Effect:

<no effect>

Process of Processes



Input & Preconditions

www.acmetravel.com
book travel service

Output & Effects

- customer name
- flight numbers
- dates
- credit card no.
-
- ...

www.acmeair.com
book flight service

?

- confirmation no.
- ...

- failure notification
- errror information
- ...

- confirmation no.
- dates
- room type
- credit card no.
- ...

www.acmehotel.com
book hotel service

?

- confirmation no.
- ...

- failure notification
- ...

?

- customer name
- location
- car type
- dates
- credit card no.
- ...

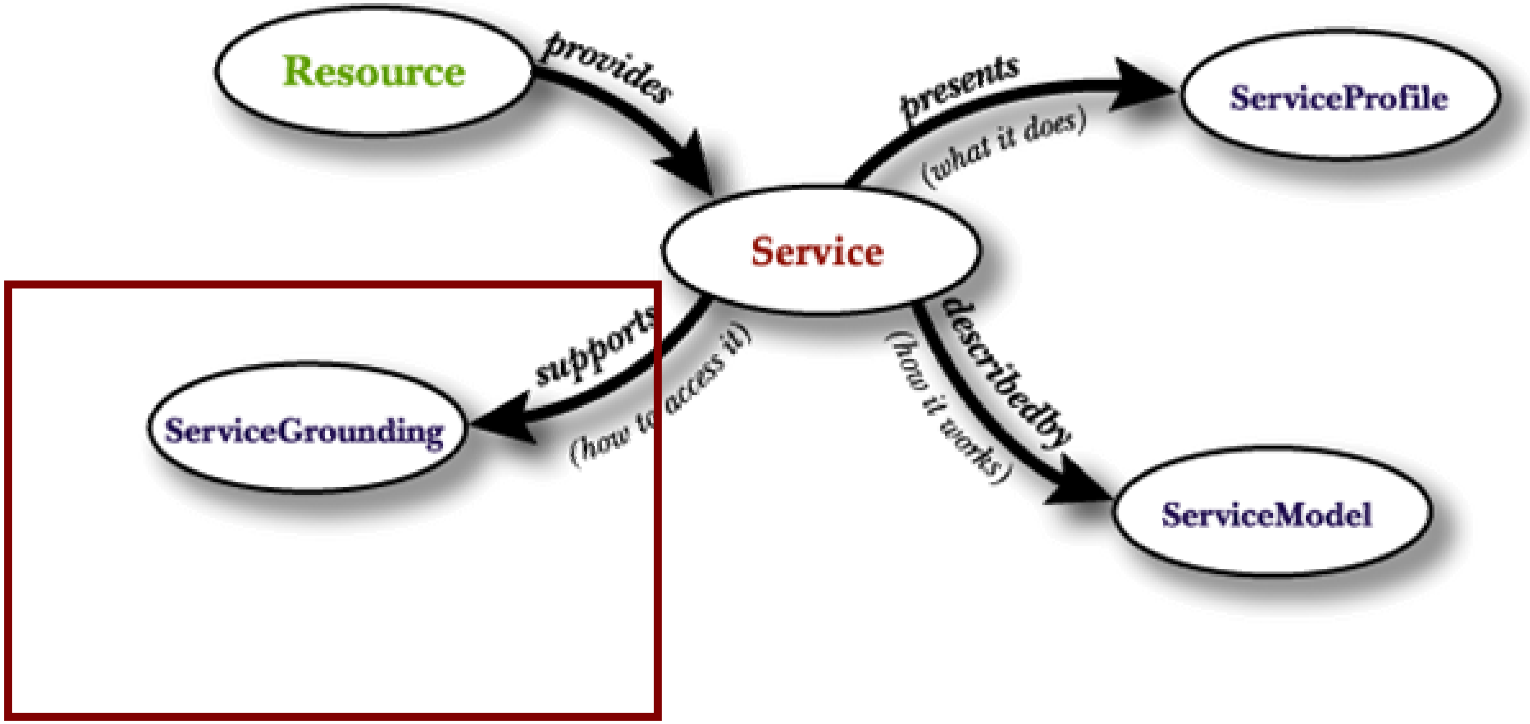
www.acmecar.com
book car service

?

- confirmation no.
- ...

- failure notification
- ...

Upper Ontology of Services

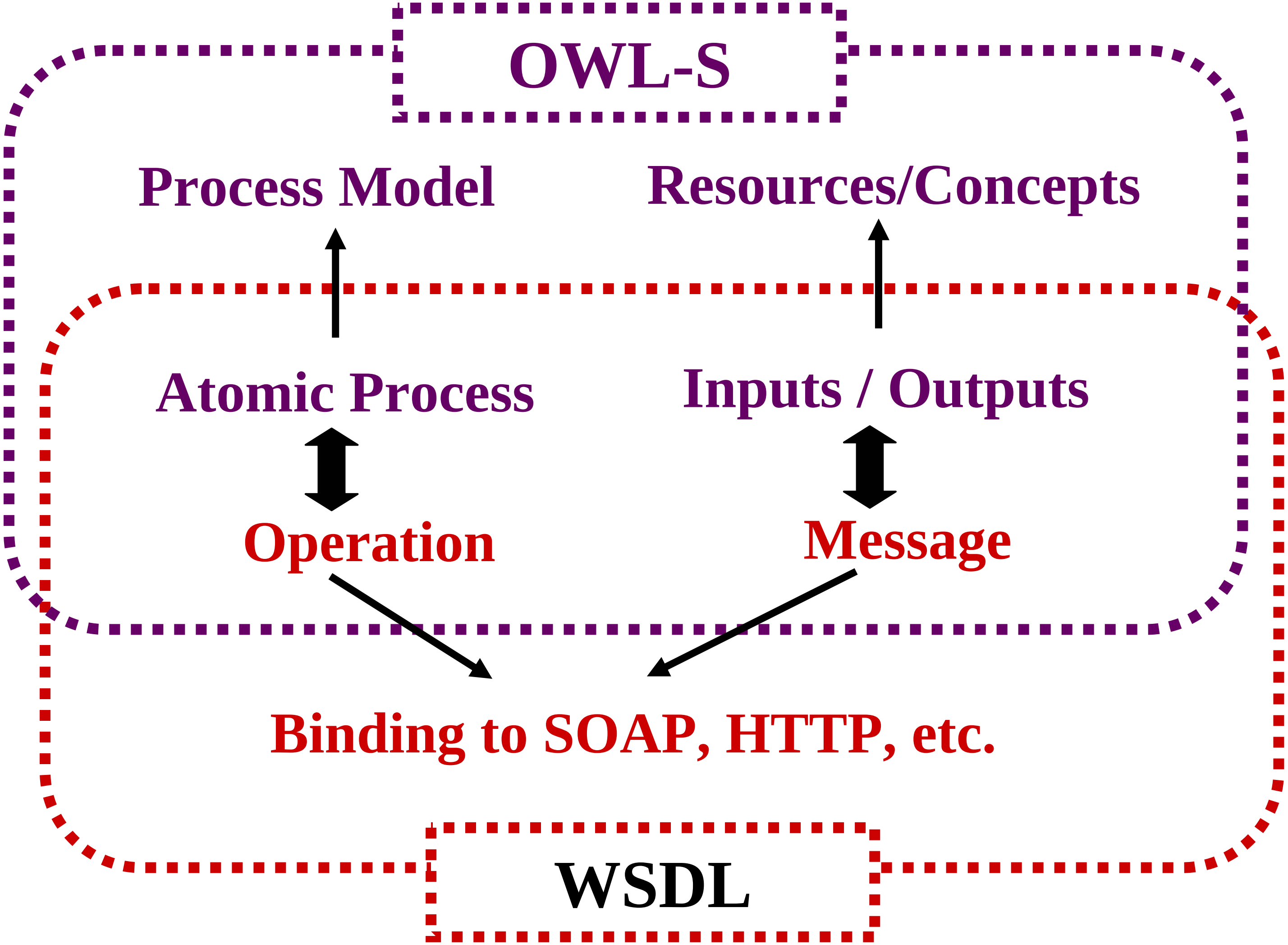


Ontology images compliments of Terry Payne, University of Southampton

Service Grounding: “How to access it”

- Implementation specific
- Message formatting, transport mechanisms, protocols, serializations of types
- Service Model + Grounding give everything needed for using the service
- Builds upon WSDL

OWL-S / WSDL Grounding



OWL-S

Process Model

Resources/Concepts

Atomic Process

Inputs / Outputs

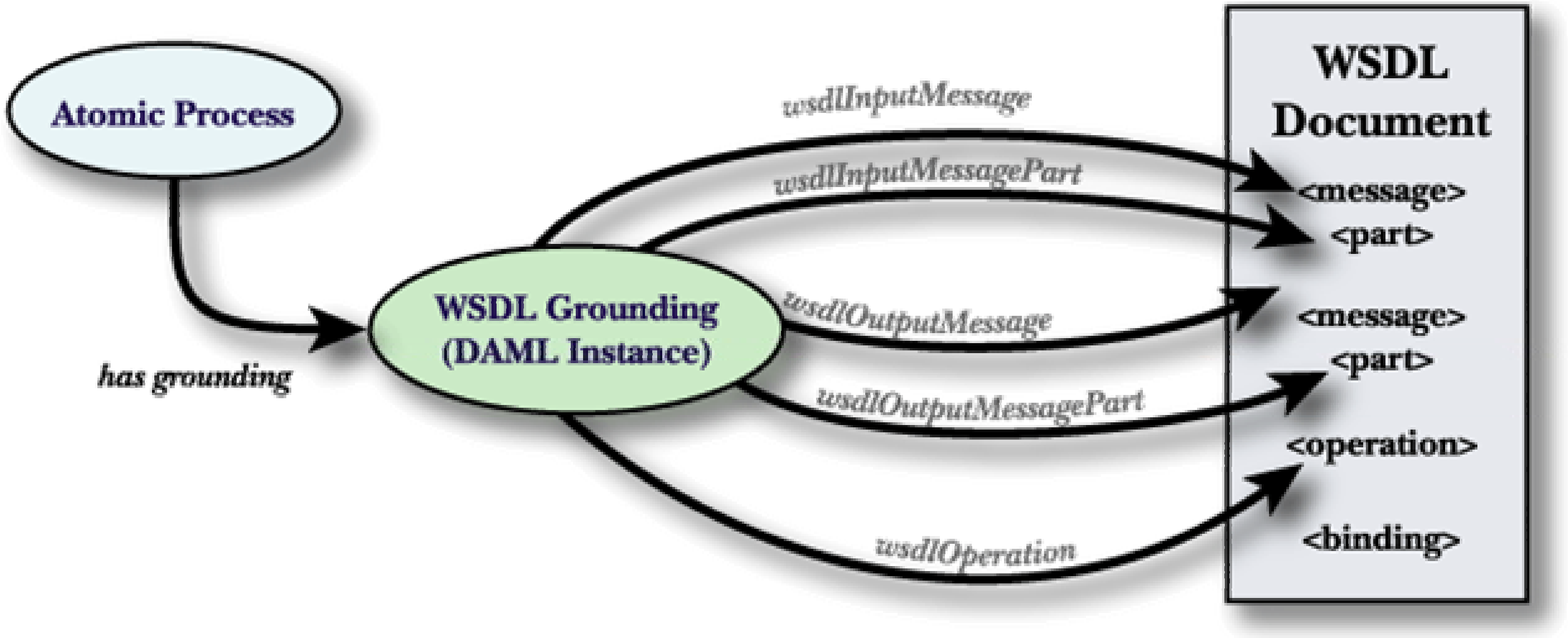
Operation

Message

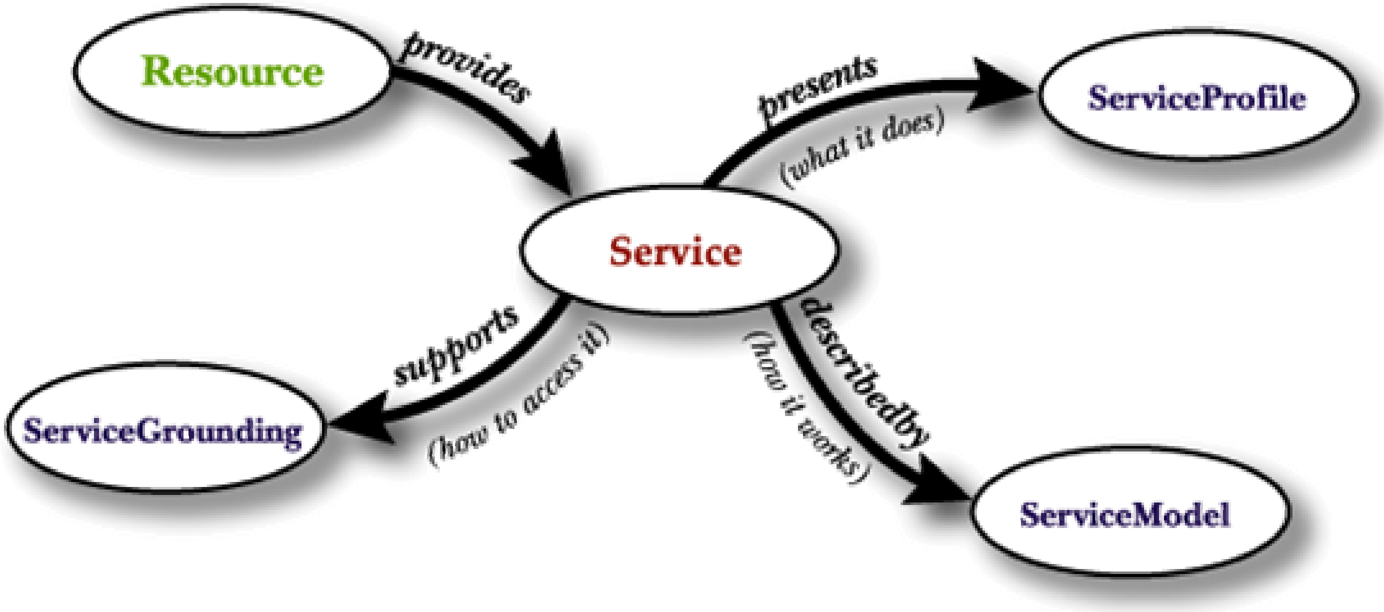
Binding to SOAP, HTTP, etc.

WSDL

OWL-S / WSDL Grounding (cont'd)



Upper Ontology of Services



Ontology images compliments of Terry Payne, University of Southampton

OWL-S: Summary & Status

- Describes “what it does”, “how it works”, “how to access it”
  - Profile, Process, Grounding subontologies
- Ties in naturally with WSDL, UDDI
- Additional semantics supports
  - Automation of various Web service tasks (later slides)
  - Varied applications (later slides)

- W3C member submission
  - http://www.w3.org/Submission/2004/07/
- 1.1 release finalized
- 1.2 release planned
- Publications, tools, examples
  - See http;//www.daml.org/services/owl-s/
  - ISWC, WWW conferences (and workshops)

Outline

- Background & Vision
- OWL-S: Ontology-Based Semantics for WS
- **SWSL: Building Out ←**
- Service Management Tasks & Tools
- Applications
- Related Work
- Challenges & Next Steps

SWSL Strategy

- Build out from OWL-S
  - to take advantage of more expressive languages
  - to extend the conceptual model
- Full-fledged use of FOL expressiveness
  - OWL-S can use SWRL and SWRL FOL in quoted contexts, in service descriptions (instances)
  - SWSL will use it throughout; both in ontology axioms and in all parts of service descriptions
- Leverage broad availability of LP-based languages, environments, tools, etc.
- Build on mature conceptual models
  - PSL, W3C architecture, Dublin core
- Maintain connections with the world of OWL
  - Layers of expressiveness

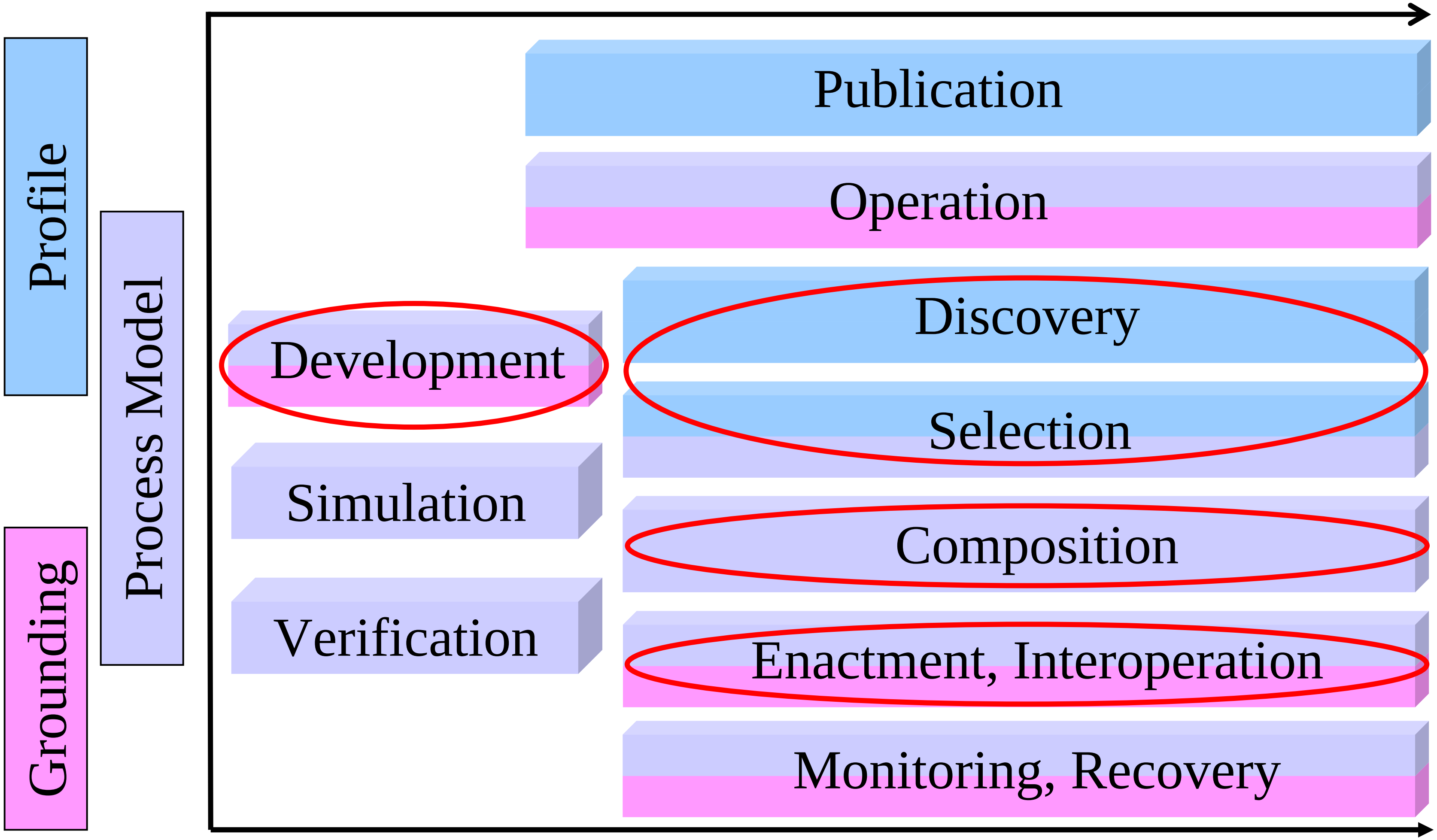
SWSL Components

- Conceptual Model
  - Build on OWL-S, PSL
- Language
  - SWSL FOL – can use frame syntax, Hilog extensions
  - SWSL Rules – LP with NAF; courteous LP, Hilog extensions
  - Shared presentation syntax; builds on F-Logic
  - Markup syntax – based on ruleML
- Ontology
  - Formal expression of conceptual model
  - Both in SWSL FOL and LP (as much as possible)
- Bridge
  - What can we provide to enable coordinated use of FOL and LP reasoners?
- Grounding
  - Like OWL-S Grounding, connects with WSDL

Outline

- Background & Vision
- OWL-S: Ontology-Based Semantics for WS
- SWSL: Building Out
- **Service Management Tasks & Tools 🡐**
  - **Development**
  - **Discovery & Selection**
  - **Enactment**
  - **Composition**
- Applications
- Related Work
- Challenges & Next Steps

SWS: Early "Sweet Spots"



Development … Deployment … Use …

Sweet Spot: Development

Some examples:

- OWL-S API (at least 2 instantiations)
- WSDL-to-OWL-S (at least 2)
- OWL-S Editor / IDE (at least 3)
- See also:
  - http://www.daml.org/services/owl-s/
    - Publications, Tools
  - http://www.semwebcentral.org

OWL-S Editor for Protégé (SRI)

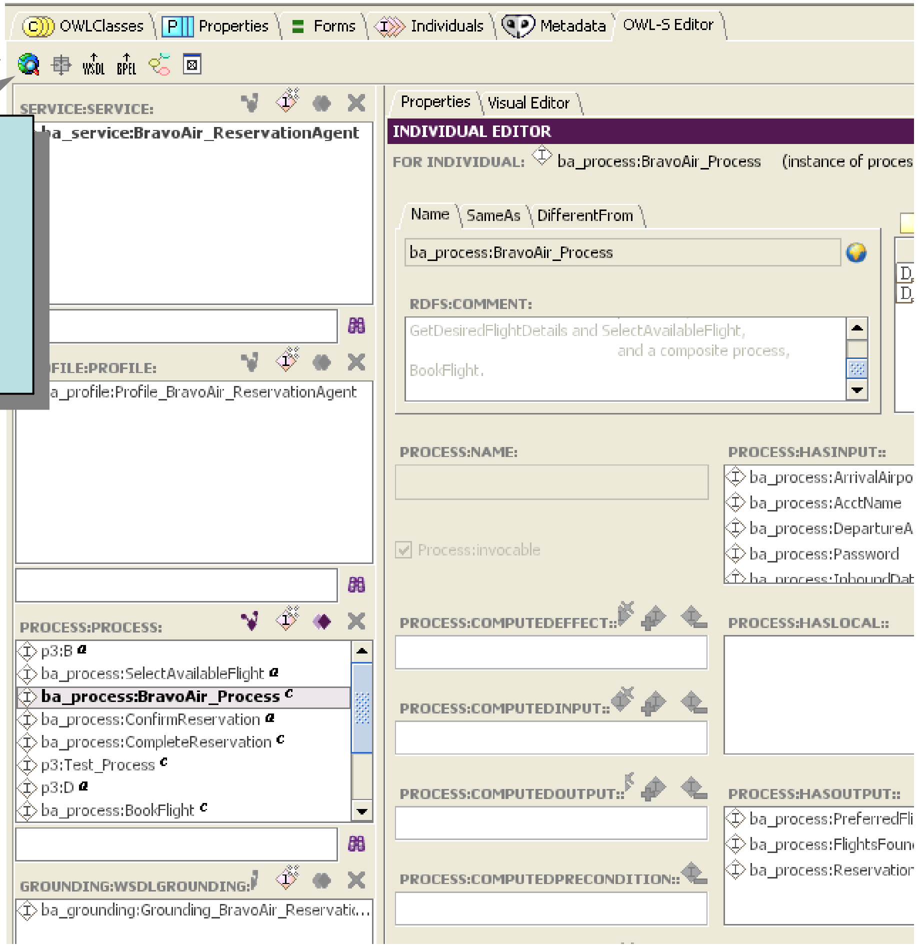
- Easy, intuitive OWL-S service development environment
- Based on popular Protégé/OWL ontology editor
- Open-source, with code available at

  http://owlseditor.semwebcentral.org
- *IOPR* Manager
  - *I*nput/*O*utput/*P*recondition/*R*esult
  - Maintain IOPR correspondences between OWL-S sub-ontologies
  - Perform consistency checks
- Auto-generated graphical process visualization
- Graph Overview
  - Visualize & navigate relationships between OWL-S sub-ontologies
- Generate & import skeletal OWL-S from WSDL
- Integrated with service discovery (planned)
- Graphical execution tracing (planned)

Sample Functionalities

Toolbar provides WSDL import, graphical overview, and more

Instance panes for Services, Profiles, Processes, and Groundings



SERVICE:SERVICE:
- ba_service:BravoAir_ReservationAgent

FILE:PROFILE:
- a_profile:Profile_BravoAir_ReservationAgent

PROCESS:PROCESS:
- p3:B a
- ba_process:SelectAvailableFlight a
- **ba_process:BravoAir_Process c**
- ba_process:ConfirmReservation a
- ba_process:CompleteReservation c
- p3:Test_Process c
- p3:D a
- ba_process:BookFlight c

GROUNDING:WSDLGROUNDING:
- ba_grounding:Grounding_BravoAir_Reservati...

FOR INDIVIDUAL: ba_process:BravoAir_Process (instance of proces

ba_process:BravoAir_Process

RDFS:COMMENT:
GetDesiredFlightDetails and SelectAvailableFlight,
and a composite process,
BookFlight.

PROCESS:NAME:

☑ Process:invocable

PROCESS:HASINPUT::
- ba_process:ArrivalAirpo
- ba_process:AcctName
- ba_process:DepartureA
- ba_process:Password
- ba_process:InboundDat

PROCESS:COMPUTEDEFFECT::

PROCESS:HASLOCAL::

PROCESS:COMPUTEDINPUT::

PROCESS:COMPUTEDOUTPUT::

PROCESS:HASOUTPUT::
- ba_process:PreferredFli
- ba_process:FlightsFoun
- ba_process:Reservatior

PROCESS:COMPUTEDPRECONDITION::

Full control of OWL-S properties with customized widgets

Thanks to SRI International

Other Features

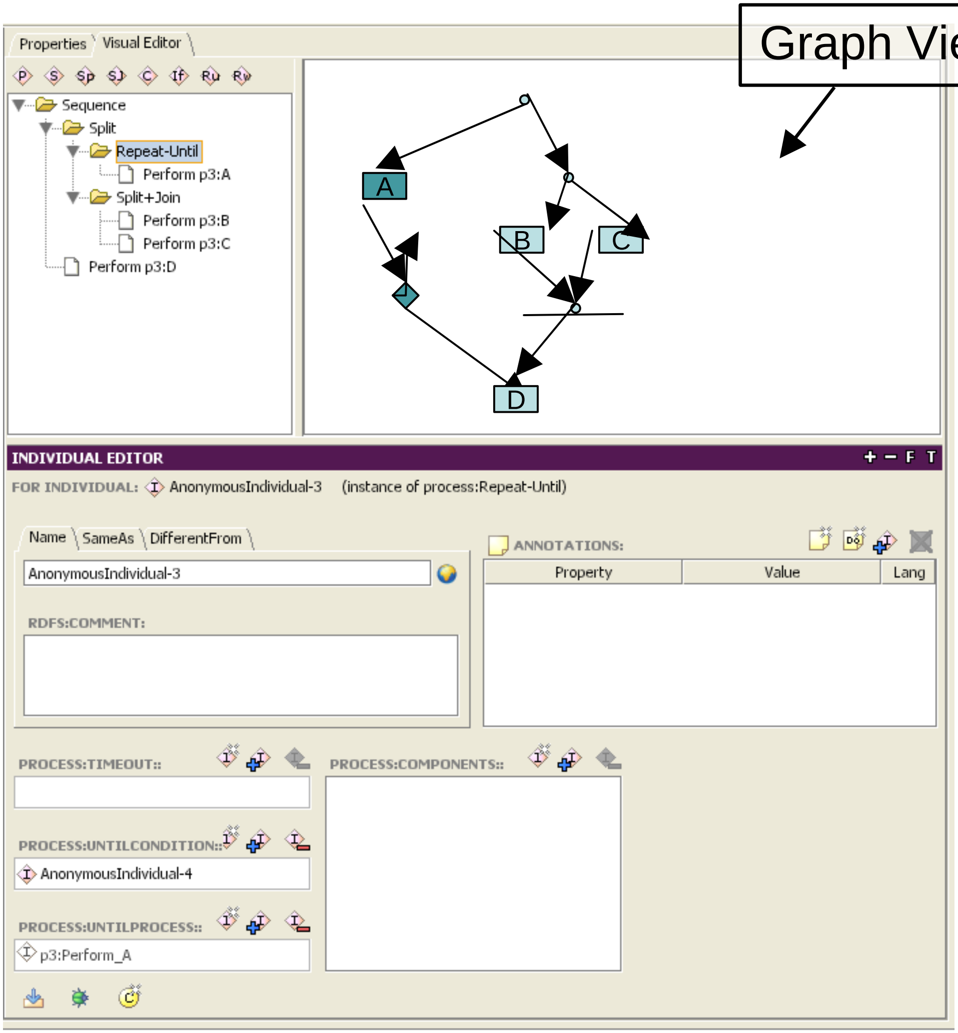
Tree View

- **Control Flow** (shown at right)
  - View and edit as a tree
  - Also visualize as a graph
- **Data Flow**

- **Work in progress**
  - Customized OWL-S code generation
  - Search the Semantic Web for OWL-S services
  - Visual tracing of execution

Edit details of control constructs

Graph View



- Sequence
  - Split
    - Repeat-Until
      - Perform p3:A
    - Split+Join
      - Perform p3:B
      - Perform p3:C
  - Perform p3:D

FOR INDIVIDUAL: AnonymousIndividual-3 (instance of process:Repeat-Until)

AnonymousIndividual-3

RDFS:COMMENT:

ANNOTATIONS:

PROCESS:TIMEOUT::

PROCESS:COMPONENTS::

PROCESS:UNTILCONDITION:: AnonymousIndividual-4

PROCESS:UNTILPROCESS:: p3:Perform_A

OWL-S IDE (CMU)

- An Eclipse-based tool that integrates the creation of OWL-S service descriptions with the generation of WS Java code

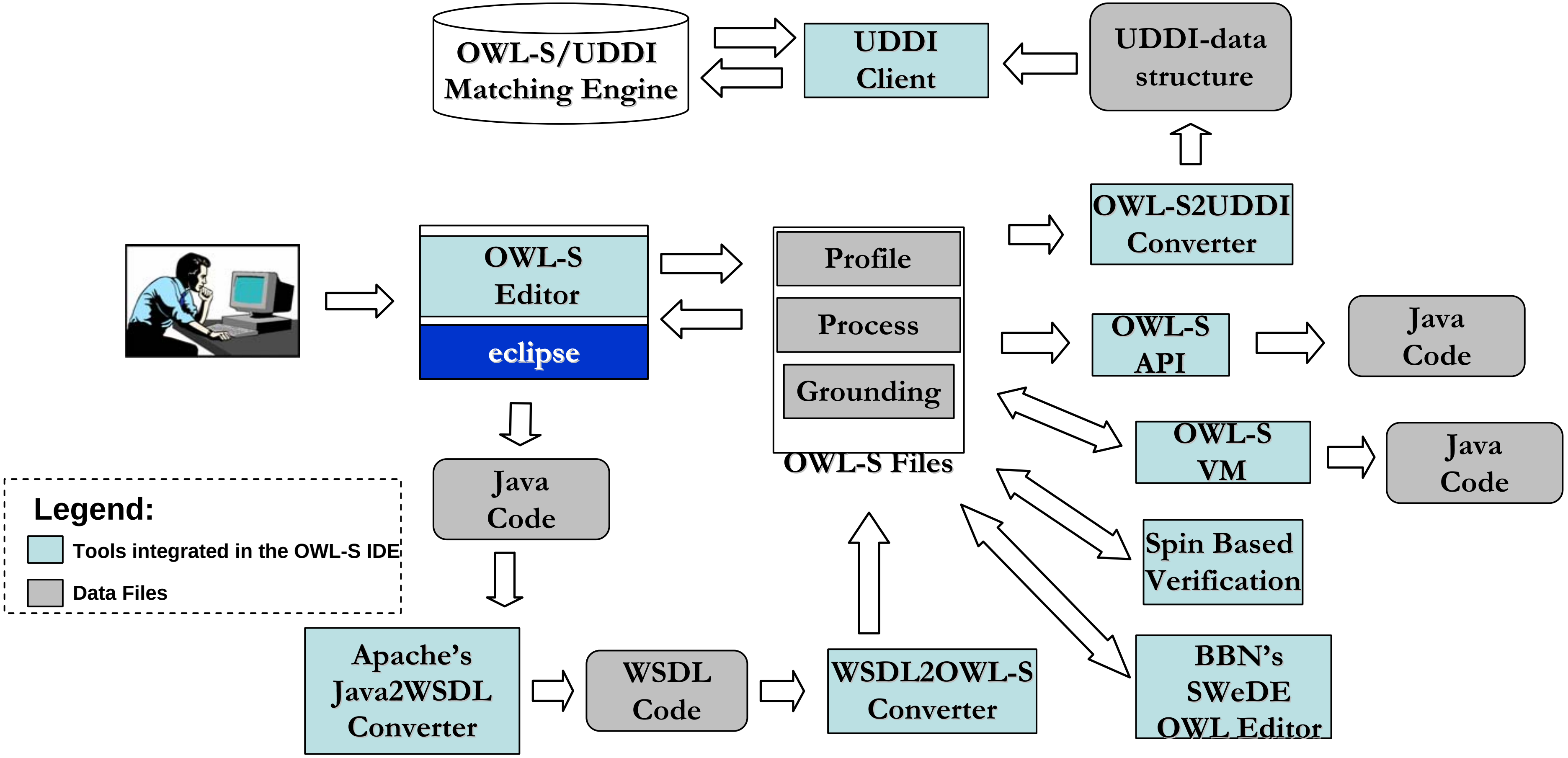
- Tools targeted to Web service developers
  - Main idea is to allow developers to generate their code and OWL-S description within the same environment

- http://projects.semwebcentral.org/projects/owl-sed/

OWL-S IDE Production Cycle

1. Developer creates Java code
2. IDE transforms Java into partial OWL description
    1. WSDL is generated as by-product
3. Easy to use OWL-S editor is used to complete the OWL-S description
4. UDDI client can be used for automatic advertisement in UDDI
5. Verification tools are available for correctness checking
6. Automatic client generation
7. Extension to SWeDE OWL Editor

OWL-S IDE Architecture



OWL-S/UDDI Matching Engine

UDDI Client

UDDI-data structure

OWL-S2UDDI Converter

OWL-S Editor

eclipse

Profile

Process

Grounding

OWL-S Files

OWL-S API

Java Code

OWL-S VM

Java Code

Spin Based Verification

BBN's SWeDE OWL Editor

Java Code

Apache's Java2WSDL Converter

WSDL Code

WSDL2OWL-S Converter

Legend:

🟦 Tools integrated in the OWL-S IDE

▨ Data Files

Sweet Spot: Discovery & Selection

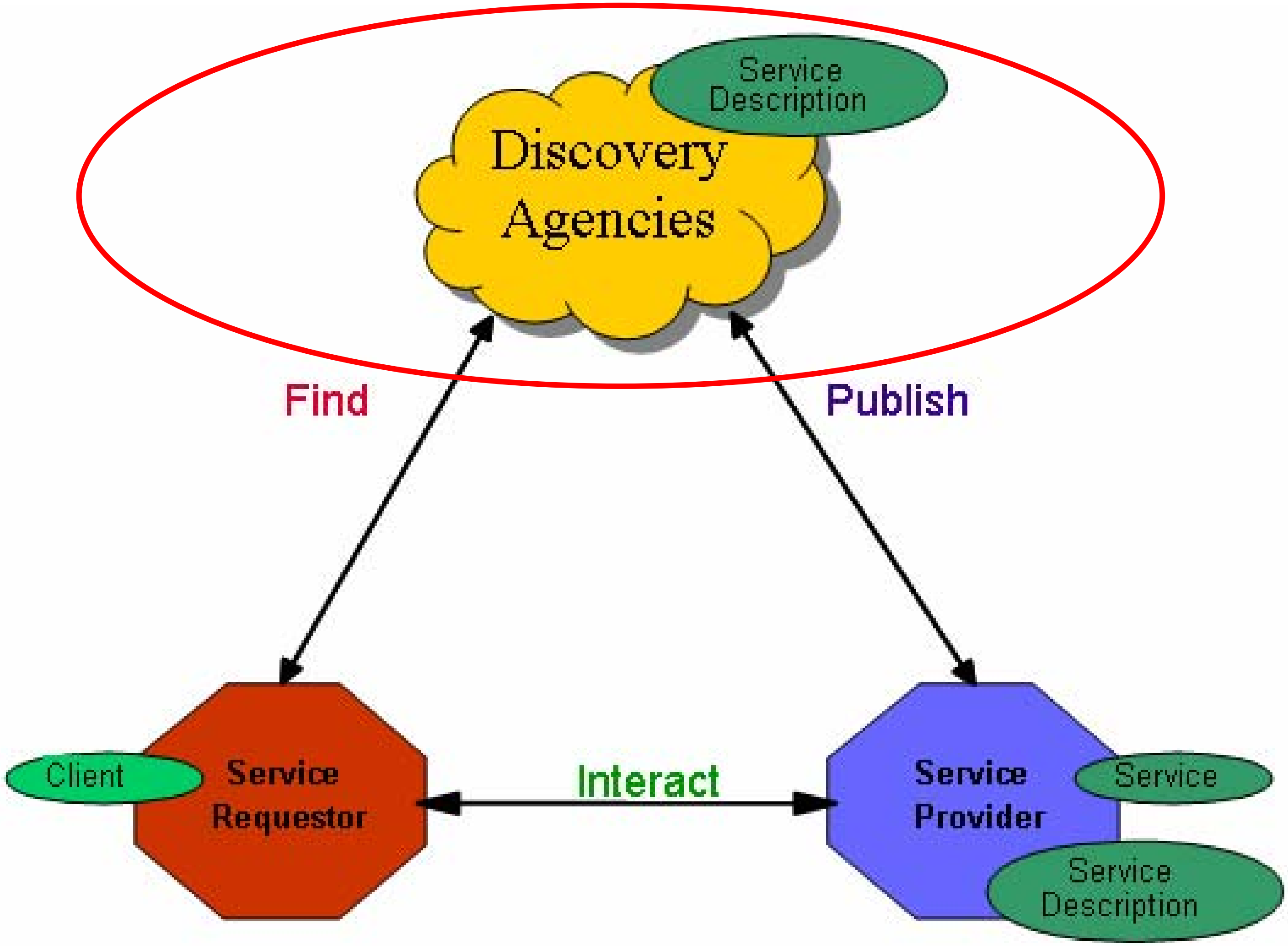


Diagram from “Web Services Architecture W3C Working Draft”
http://www.w3.org/TR/2002/WD-ws-arch-20021114/

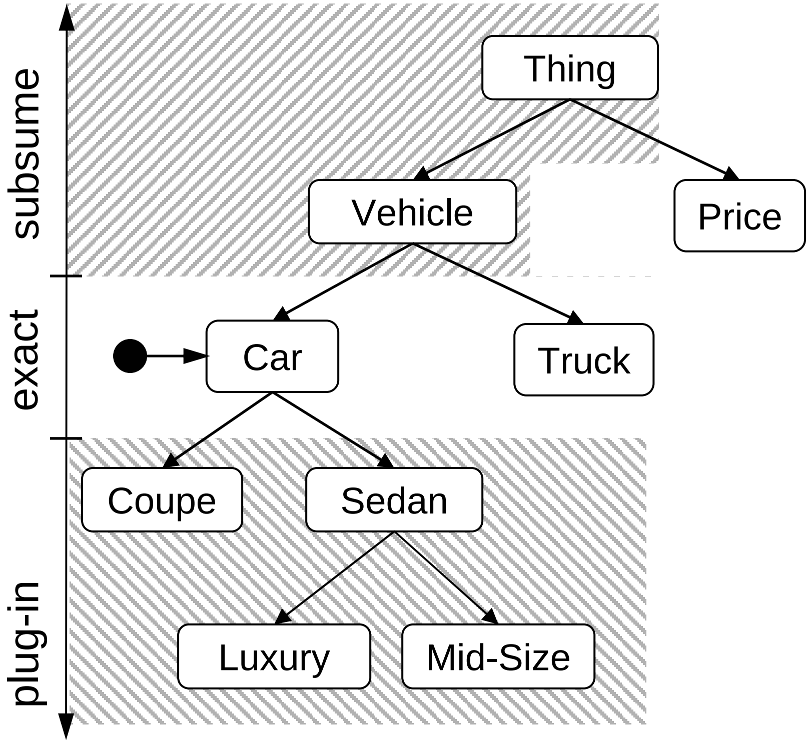
Expressing capabilities in OWL-S

- OWL-S *Profile* describes capabilities of Web services
- Three types of representations:
  1. Functional representation
     - Input/Output specify the information transformation produced by the Web service
     - Precondition/Effect specify the domain transformation produced by the Web service
  2. Non-functional properties
  3. Type of service and product information
- Many capability matching algorithms have been proposed.

Discovery with OWL-S

CMU’s Matchmaker

- Matching of I/O of the request with I/O of the advertisement
- **Efficient implementation** given correct indexing of advertisements
  - Match within ms
  - Linear complexity on the size of the query
- **Current work** aims at generalizing matching process to include preconditions/effects service and product types and service parameters



Discovery with OWL-S

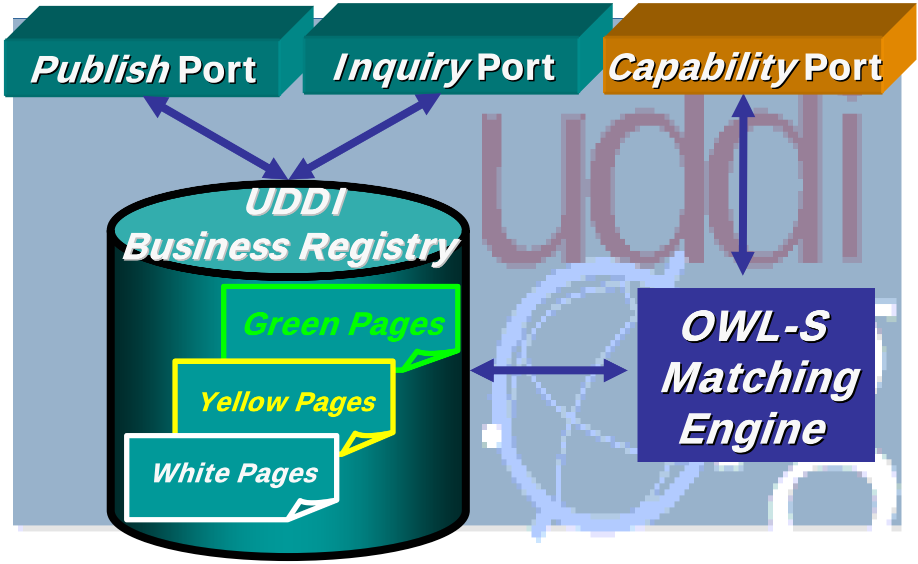
Using Subsumption

- Use subsumption relation between advertisement and request
- Five degrees of match
  - Exact
  - PlugIn $R \subseteq A$
  - Subsumed $A \subseteq R$
  - Intersection $\neg(A \sqcap R \subseteq \bot)$
  - Fail when disjoint $A \sqcap R \subseteq \bot$
- It shows that pure subsumption is inadequate for discovery in OWL-S
  - But problem is much deeper: subsumption is inadequate for discovery of Web services because
    - It is inherently difficult to specify partial descriptions of services which would allow the requester to say which are the features of the WS it really care about
    - Most of the matches reduce to intersection which is not really informative

Proposed by Li et al, WWW 2003

Discovery with OWL-S

Integration of OWL-S and UDDI



- OWL-S Profile has been mapped to UDDI data structure
  - OWL-S Web services can be advertised in UDDI as any other Web service

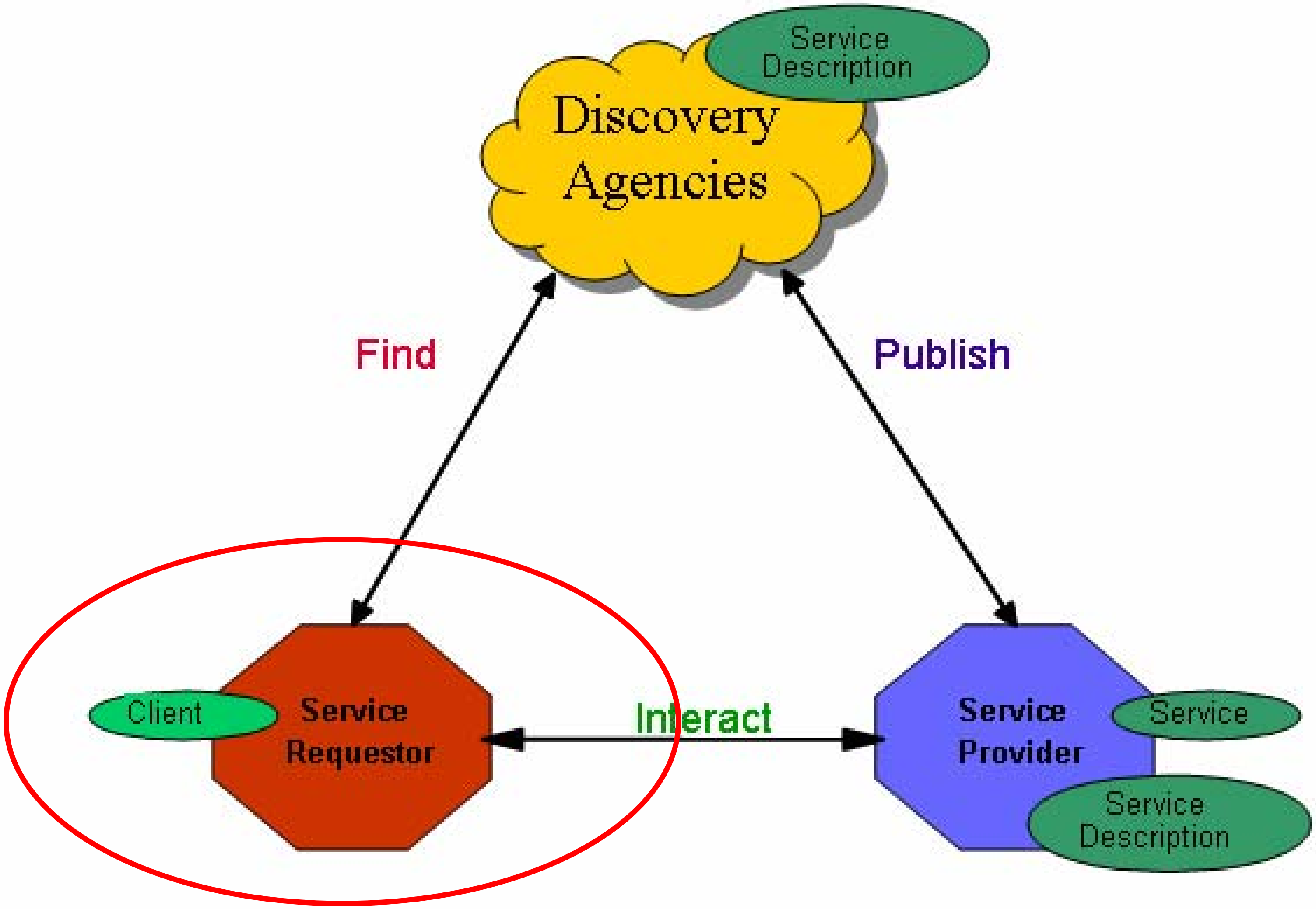
(see Paolucci et al 2002)

- CMU OWL-S Matching engine has been integrated within UDDI server
- CMU UDDI server provides
  - Normal UDDI Publish/Inquiry ports
    - Complete interoperability with any UDDI Client
  - Capability Port provides OWL-S based capability requests

(see Srinivasan et al 2004)

CMU UDDI is publicly available at
www.daml.ri.cmu.edu/matchmaker
or on SemWebCentral
www.semwebcentral.org
A variant of the CMU UDDI is in use at the NTT UDDI Business Registry (The main public UDDI in Japan) (see Kawamura et al 2003, 2004)

Sweet Spot: Enactment

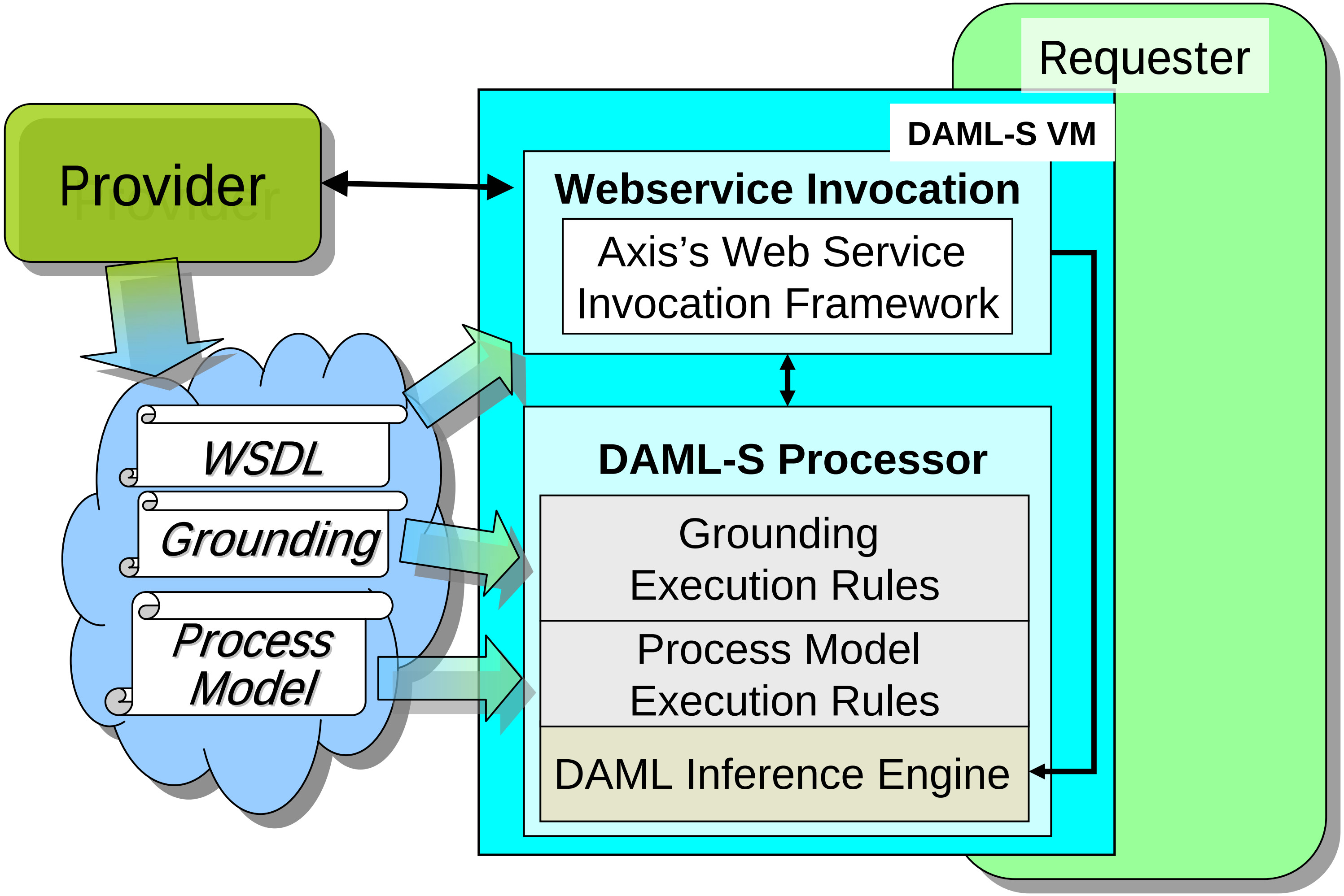


CMU’s OWL-S Virtual Machine

- Generic interpreter for OWL-S Process Model
  - It can interact with any OWL-S Web service
  - Based on the Process Model formal semantics (Ankolekar et al 2002)
  - Implement grounding mapping to WSDL
  - Exploits Web services technology such as Axis and WSIF for actual invocation and message exchange
  - Semantic Mediation

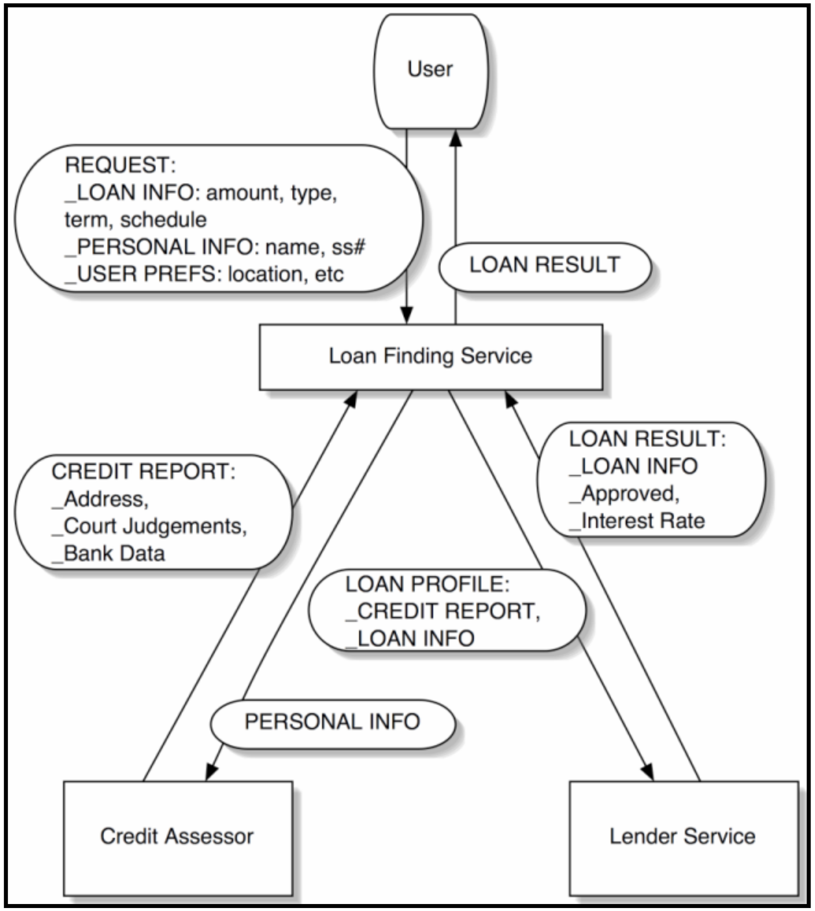
Enactment with OWL-S

CMU's OWL-S Virtual Machine



Semantic Discovery Service

- *Adapting BPEL4WS for the Semantic Web*
- Daniel J. Mandell
- Sheila A. McIlraith (was at Stanford KSL, now University of Toronto)



Shortcomings in BPEL

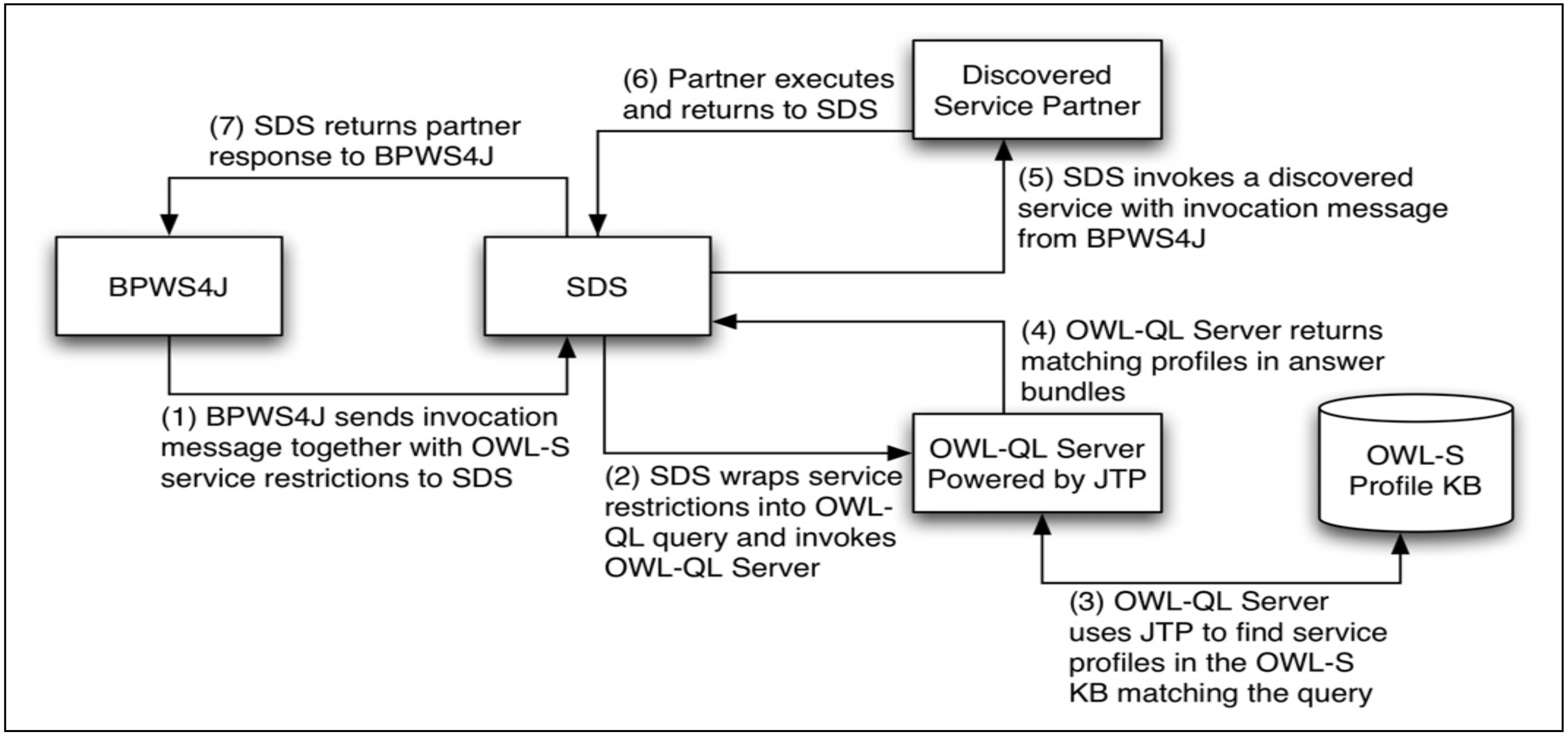
- In the current BPEL engine (BPWS4J): service partners defined *a priori*
  - What if user wants to use California-based lender to exploit in-state tax incentives?
  - BPEL engine cannot accommodate any such user-defined (runtime) customization b/c service partners are hard-coded
- In BPEL spec: reliance on XML, XMLSchema to describe service properties and messages
  - XMLSchema lacks expressivity to create/relate rich datatypes
  - Prevents integration of service partners with different syntax but compatible semantics (and converse)
  - What if: user from UK, and **assessor** generates `UKCreditReport` identical to `USCreditReport` required by **lender** except different representations of dates? BPEL fails integration

Semantic Discovery Service

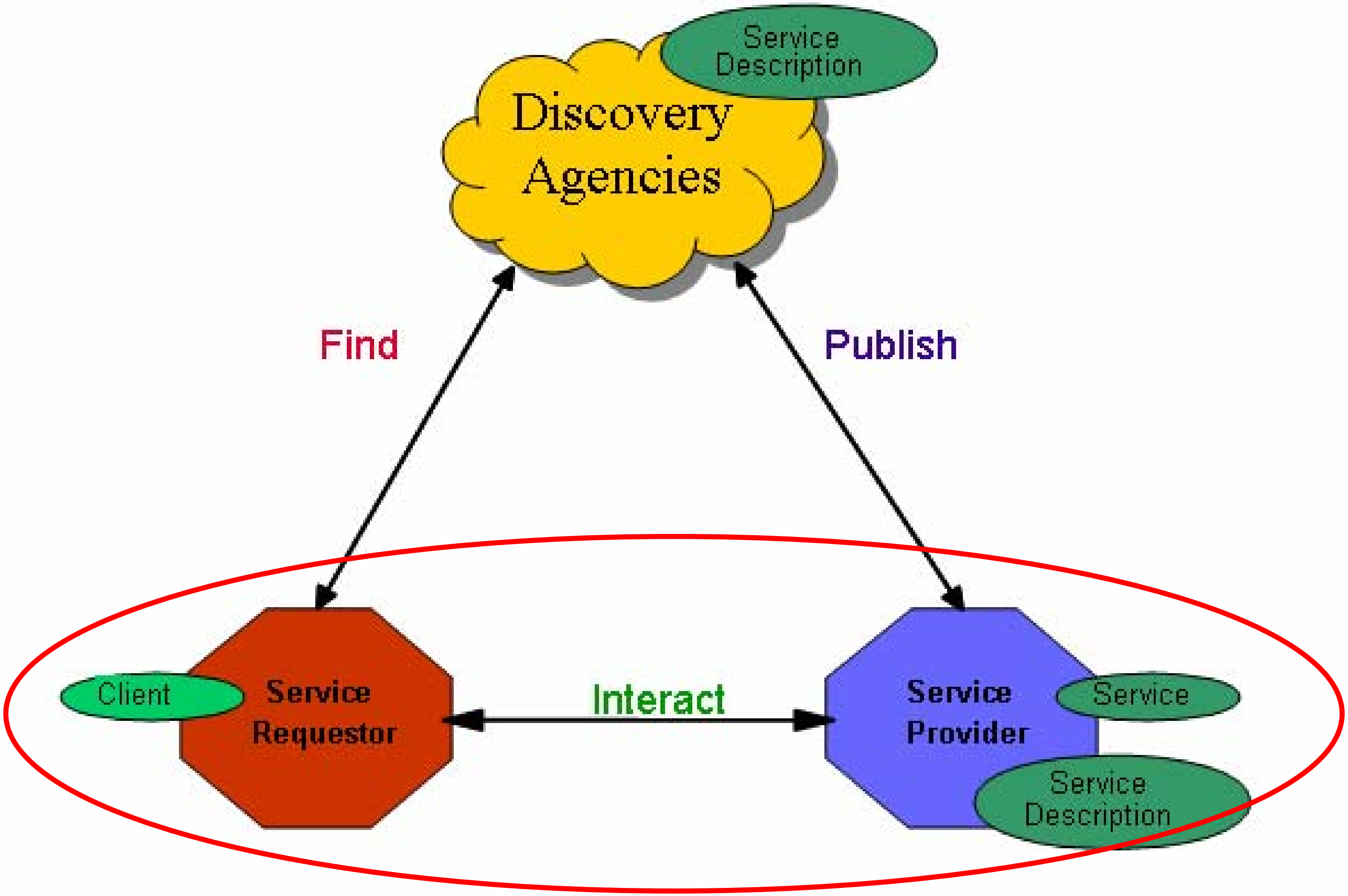
- Alleviates certain limitations of BPEL4WS
- Functionality gained:
  - Automated run-time binding of customized services to account for functional and user-defined constraints
  - Automated *semantic translations* to integrate services with messages of different syntax but equivalent semantics
- Supporting Semantic Web technologies
  - OWL-S: A well-defined ontology based on OWL, used to describe services
  - OWL Query Language (OWL-QL): Interfaces with *automated reasoner* operating over *knowledge base* (KB) of OWL-S profiles
  - Java Theorem Prover (JTP): Used as OWL-QL server's automated reasoner

Service Integration with SDS

- SDS serves as proxy between BPEL engine and potential service partners, discovering OWL-S profiles in KB meeting functional and user-defined constraints (automated customization) and translating semantically equivalent messages (semantic translations)


(6) Partner executes and returns to SDS
Discovered Service Partner
(7) SDS returns partner response to BPWS4J
(5) SDS invokes a discovered service with invocation message from BPWS4J
BPWS4J
SDS
(4) OWL-QL Server returns matching profiles in answer bundles
(1) BPWS4J sends invocation message together with OWL-S service restrictions to SDS
OWL-QL Server Powered by JTP
OWL-S Profile KB
(2) SDS wraps service restrictions into OWL-QL query and invokes OWL-QL Server
(3) OWL-QL Server uses JTP to find service profiles in the OWL-S KB matching the query

Sweet Spot: Composition



MindSwap’s Web Service Composer

- WS composition environment
  - Uses SHOP2, a well established planner
  - Contains an OWL-S execution environment
- Used for many applications of WS composition ranging from
  - Information gathering
  - Language translation
  - etc...
- Generates a composition that is directly executable through WSDL groundings.

in collaboration with FUJITSU

KSL Automated WS Composition Tool

Approach:

I. Plan a sequences of services that realize user's objective, using Golog & sit'n calculus .
(NP complete or worse)

II. Customize reusable generic procedures
- Define and archive reusable **generic procedures**
- Customize with **user's constraints**.
(NP complete or worse in a reduced search space)

Advantages: efficiency, ease of use, customization

CMU Composition Architecture

- It integrates discovery and composition
  - OWL-S/UDDI Matchmaker for discovery
  - Retsina planner to control the agent
    - Interleaving of planning and execution to allow communication while planning
  - OWL Reasoner
  - OWL-S Virtual Machine to communicate with other Web Services
- Used in a number of applications: travel domain, supply chain management
- Connection with autonomous agent technology

in collaboration with TOSHIBA

Outline

- Background & Vision
- OWL-S: Ontology-Based Semantics for WS
- SWSL: Building Out
- Service Management Tasks & Tools
- **Applications 🡐**
- Related Work
- Challenges & Next Steps

Some Applications of OWL-S

IBM
- Provide OWL-S API as part of SNOBASE Semantic Web tool
- http://www.alphaworks.ibm.com/tech/snobase
- Use OWL-S for enhanced semantic UDDI

SAP
- Use OWL-S for automatic composition of services to manage border control

Toshiba
- Use OWL-S in publicly available UDDI at NTT (Main Japanese UDDI)

Fujitsu
- OWL-S used in Task Computing Project; planned for production in 2005
- http://www.taskcomputing.org/

NIST, DCS, TARDEC
- Use OWL-S to describe capabilities of Autonomous Vehicles

MyGrid
- Use OWL-S to describe Bioinformatics Web services on the Grid
- http://www.mygrid.org.uk/

AgentCities
- OWL-S used for discovery of new agents
- http://www.agentcities.org/

Outline

- Background & Vision
- OWL-S: Ontology-Based Semantics for WS
- SWSL: Building Out
- Service Management Tasks & Tools
- Applications
- **Related Work 🡐**
- Challenges & Next Steps

Related Work (Precursors)

- Agent-Based Systems
- Knowledge-Based Software Engineering (KBSE)
  - Automated Software Engineering (ASE)
- AI Planning
- Programming Languages
- Workflow Systems
- Knowledge Representation
  - Situation Calculus
- Process Representation
  - Pi Calculus
  - Process Specification Language (PSL)

Directly Related Work

- **WSMO / WSML / WSMX**
  - http://www.wsmo.org/
- **OWL-P**
  - http://projects.semwebcentral.org/projects/owlp
- **METEOR-S**
  - http://lsdis.cs.uga.edu/Projects/METEOR-S/
- **Grid Services**
  - **OGSI → WSRF**
  - http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsrf
- **BPMN / BPML / BPQL**
  - http://www.bpmi.org/

Web Services Modeling Ontology

- WSMO / WSML / WSMX
- Under development at DERI and other organizations with European research funding
- Hi-level objectives & approaches similar to those of OWL-S
- But many interesting differences, too; e.g.
  - Focus on mediation
  - Different layers of language / expressiveness
- http://www.wsmo.org/
- Hear Chris Bussler tomorrow morning

OWL-P

- Processes = Protocols + Policies
- Protocols provide interaction-centric modeling, leaving policies to participants
- Commitment semantics yield flexible modeling and enactment
- Theory of protocols supports reusability, refinement, and aggregation of interactions
- Focus on *interaction*

METEOR-S

- Research at University of Georgia, led by Amit Sheth
- Focus on Web service lifecycle stages:
  - Semantic Annotation and Publication of Web Services
  - Abstract Process Creation
  - Semantic Discovery of Web Services
  - Orchestration/Composition of Web Services
- Lightweight approach: proposed enhancements to WSDL, UDDI, BPEL4WS

Grid Services

- http://www.globus.org/wsrf/
- submitted to OASIS standards group in March 2004
- Web Services Resource Framework
- http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsrf

Outline

- Background & Vision
- OWL-S: Ontology-Based Semantics for WS
- SWSL: Building Out
- Service Management Tasks & Tools
- Applications
- Related Work
- **Challenges & Next Steps ←**

Challenges

- Sociological: crossing the chasm
  - Infrastructure: getting to where the payoff exceeds the overhead (for significant numbers)
  - Allowing for small first steps in the meantime
  - Making it clear (to the right people at the right time) what’s the value-added
  - Comfort level with what’s under-the-hood
  - Getting consensus on approaches, ontologies
    - Competitive pressures
- Engineering
  - Knowledge acquisition
  - Scale
  - Heterogeneity, mediation

Challenges

- Technical
  - Determining how to scope, delimit, and deploy mature semantic technologies for use on the Web
    - Ultimately, the “marketplace” will make it happen
    - Complete generality not required
    - Full automation not required
    - “A little semantics goes a long ways”
  - Usability by non-logicians (Good tools are critical)
- Logical
  - Tractability
  - Bridging layers of expressiveness

Next Steps in SWS

- OWL-S version 1.2 (by end of May)
- SWSI report, submission (April)
- WSMO report, submission (April)
- W3C Workshop on Semantic Web Services
  - June 9-10, 2005
  - http://www.w3c.org/2005/01/ws-swsf-cfp.html
  - Recommendations to W3C on directions, activities
- Likely new activities at W3C around SWS
- SWSI, WSMO collaboration, merging
- In the meantime, evolution towards semantics in WS standards efforts
  - WSDL, UDDI, BPEL4WS, CDL, …

Conclusion

- The service paradigm will be a *crucial* and *integral* part of the Web of the future
- SWS aims to provide an expressive, comprehensive framework for reasoning about activities on the Web
  - Enable greater automation of *discovery, selection, invocation, composition, monitoring,* and other service management tasks
- Key challenge: distill mature technologies and show path(s) to usage that fit with (or can evolve from) current practice
- Many interesting tools & applications exist already