Specifying a Web Service Ontology in First-Order Logic

March 28, 2006

Michael Gruninger and **Sheila McIlraith**

University of Toronto

(with the contributions of numerous others)

Presented by Sheila McIlraith

Take Home Message

Please look at:

- Semantic Web Services Framework (SWSF)

  http://www.daml.org/services/swsf/1.0/

  And specifically FLOWS (aka SWSO-FOL)

  http://www.daml.org/services/swsf/1.0/swso

- Process Specification Language (PSL)

  http://www.mel.nist.gov/psl/

- OWL-S

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

(Some of) what I hope you'll get from the talk

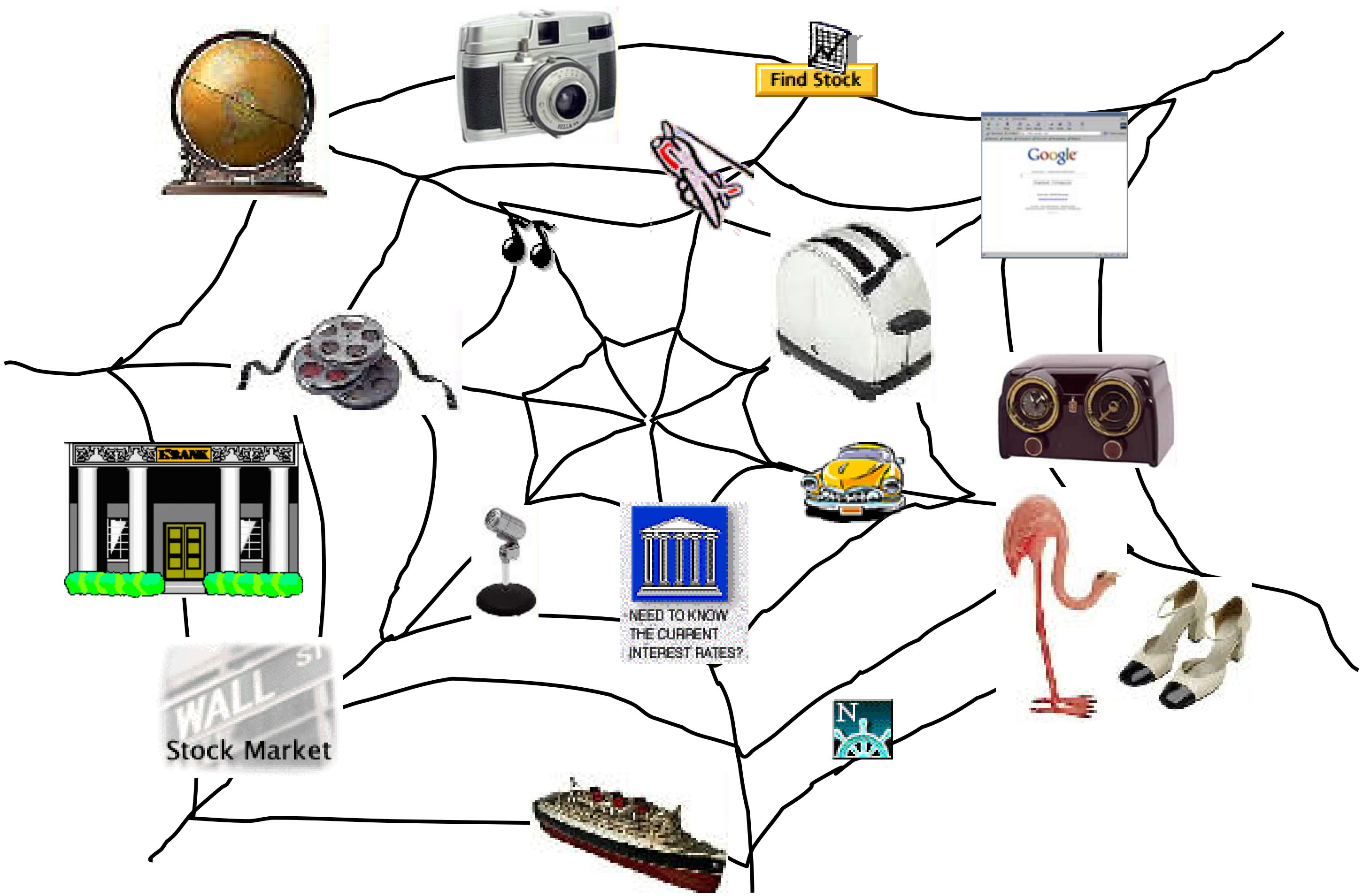
- Web Services are a rich domain for KR&R research.

- The landscape of ontologies for Web services:
  - OWL-S
  - FLOWS (SWSO) as part of SWSF
  - WSMO

- These ontologies (and particularly FLOWS/PSL) are examples of open repositories of domain-specific knowledge in the spirit of this symposium and its vision.

Web Services (WS)

Web Services are Web-accessible programs and devices.



Broad Objectives

1. **Self-describing Web Services:**

   Knowledge representation to enable automation by making service capabilities & user constraints *unambiguously computer interpretable* & *use-apparent.*

2. **Automation of Web Service Tasks:**

   Automated reasoning techniques that exploit KR to support automated Web service *discovery*, *invocation*, *composition and interoperation*.

Goal

Automation of:

- Web service discovery

  Find me a shipping service that will transport wine from San Francisco to Toronto.

- Web service invocation

  Buy me "Harry Potter and the Philosopher's Stone" at www.amazon.com

- Web service selection, composition and interoperation

  Make the travel arrangements for my KR2006 conference.

- Web service execution monitoring

  Has my book been shipped yet?

Web service simulation and verification

1. Self-describing Web Services:

Towards a declarative language for describing Web services ….

Modification of slide by James Snell (IBM)

Industry Activity: The Web Services Stack

Wire Protocols	Description	Discovery
SOAP Blocks	Agreements	
SOAP/XMLP	Process	
XML	WSDL Extensions	
HTTP/SMTP/BEEP	WSDL	Registry (UDDI)
TCP/IP	XML	Inspection

The Web Services Stack (cont.)

Wire Protocols

Description

Discovery

BPEL4WS (Microsoft, IBM, BEA,…)
W3C WS Choreography Group

WSCL (HP), BPML (Most but Microsoft)
WSCI (Sun, BEA, Yahoo, …)
XLANG (Microsoft), WSFL (IBM), …

Wire Protocols:
- SOAP Blocks
- SOAP/XMLP
- XML
- HTTP/SMTP/BEEP
- TCP/IP

Description:
- Agreements
- Process
- WSDL Extensions
- WSDL
- XML

Discovery:
- Registry (UDDI)
- Inspection

Industry Process Description Languages

Shortcomings:

- No well-defined semantics, despite origins in Petri Nets and Pi-calculus – thus the process model is not unambiguously computer interpretable.

- Lack the content necessary for SWS automation tasks (e.g., non-functional properties of services, effects of services, etc.)

- Describe process flow, but do not describe all the attributes of the process (e.g., that input-1 of process P is a book ISBN number and that book ISBN numbers have value restrictions and a 1-1 correspondence with a book, etc.) – thus can't reason about the entities being manipulated by the process model.

Modification of slide by James Snell (IBM)

SWS Languages Complement Industry Efforts

Wire Protocols

- SOAP Blocks
- SOAP/XMLP
- XML
- HTTP/SMTP/BEEP
- TCP/IP

Description

- Agreements
- Process
- WSDL Extensions
- WSDL
- XML

S W S

Automated **Discovery**
Invocation
Interoperation
Composition
Monitoring
Verification

- Registry (UDDI)
- Inspection

Outline

- OWL-S
- FLOWS: **F**irst-Order **L**ogic **O**ntology for **W**eb **S**ervices

OWL-S: A description-logic based SWS Lang.

OWL-S is an OWL (Ontology Web Language) ontology for WS

- Successor to DAML-S, a DAML+OIL ontology for WS
- All the merits of OWL, including:
  - expressiveness
  - well-defined semantics
  - decidable
  - declarative
  - supports compact rep'ntation, mapping, sharing, reuse, …
- Developed with the support of the DARPA DAML program.
- Developed by the coalition of researchers listed previously.

http://www.daml.org/services/

[DAML-S Coalition, 01, 02]

OWL-S Acknowledgements

OWL-S is the joint work of the **DAML-S Coalition**.

Members (old & new) include*:

BBN: Mark Burstein

CMU: Katia Sycara, Massimo Paolucci, Naveen Srinivasan, Anupriya Ankolekar,

De Montfort University: Monika Solanki

ICSI: Srini Narayanan

Maryland / College Park: Bijan Parsia, Evren Sirin

Nokia: Ora Lassila

SRI: David Martin

Stanford KSL: Deb McGuinness

Southampton: Terry Payne

Univ. of Toronto: Sheila McIlraith

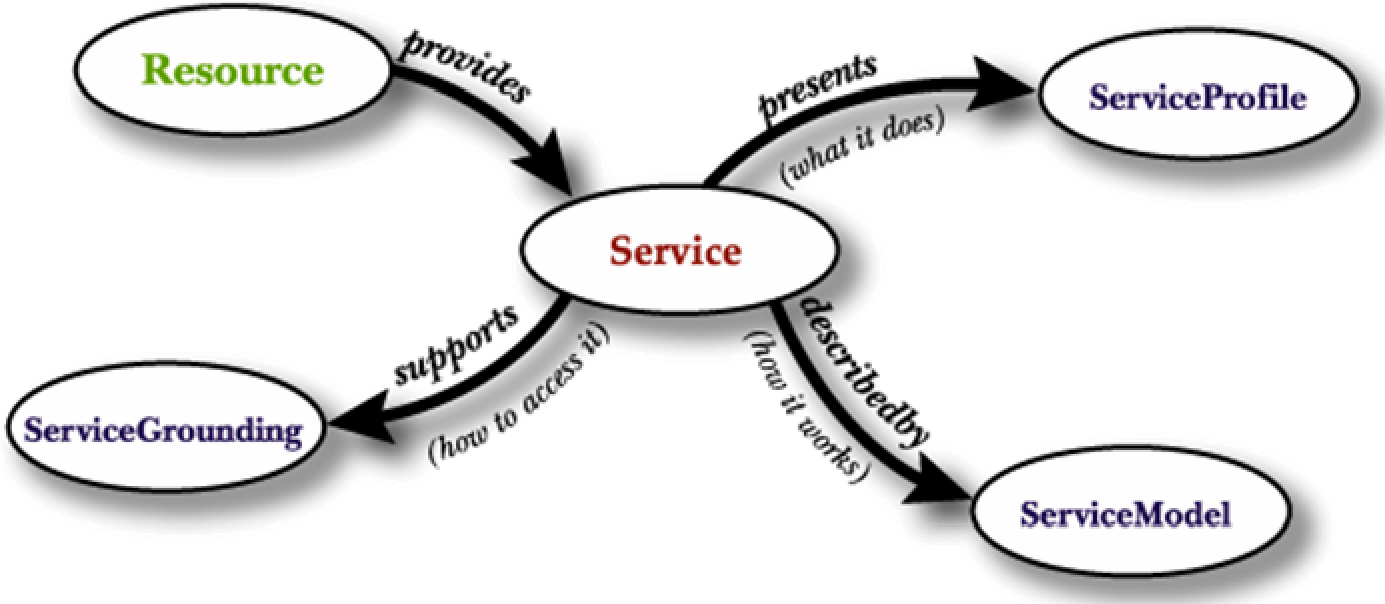
USC-ISI: Jerry Hobbs

Vrije Universiteit Amsterdam: Marta Sabou

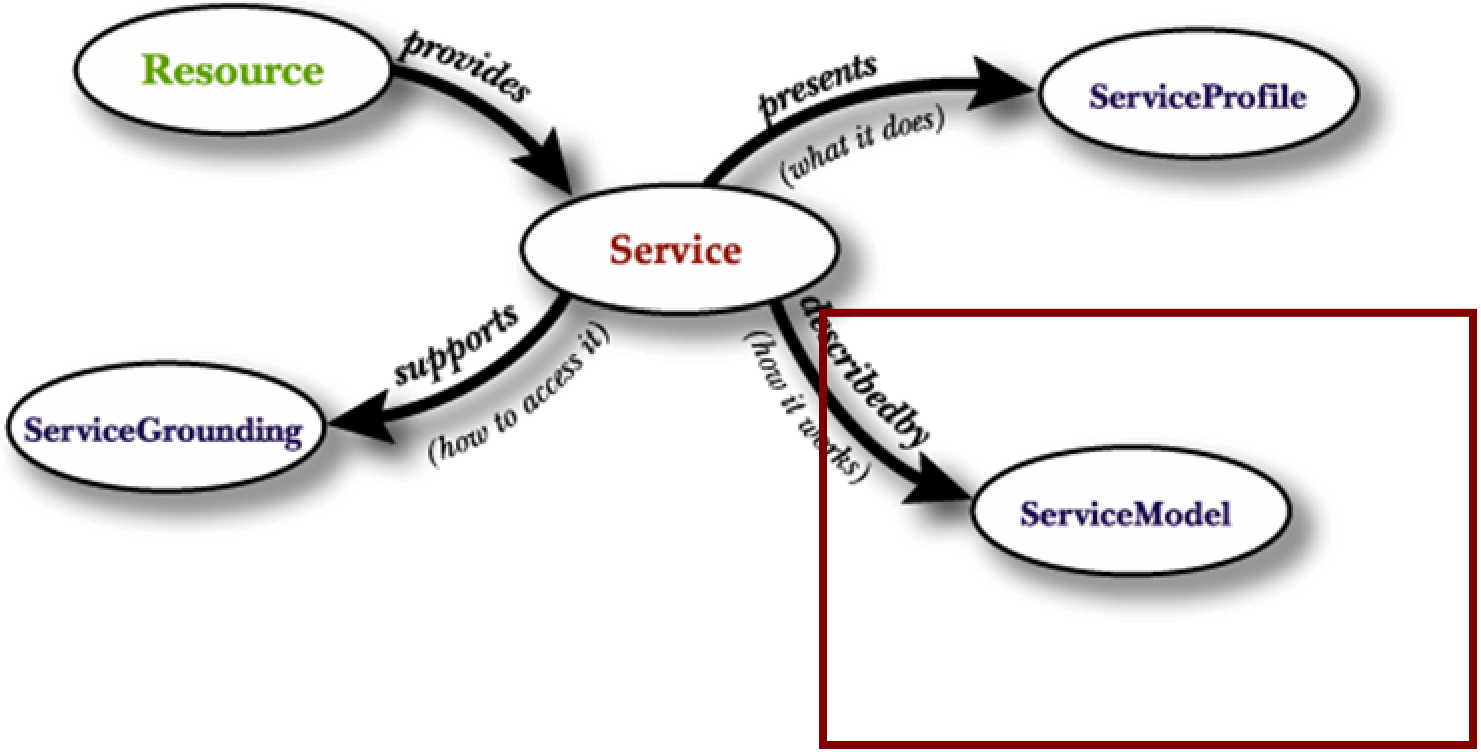
Yale: Drew McDermott

** Apologies to anyone I've missed*

Upper Ontology of OWL-S



The Service Process Model



Service Process Model

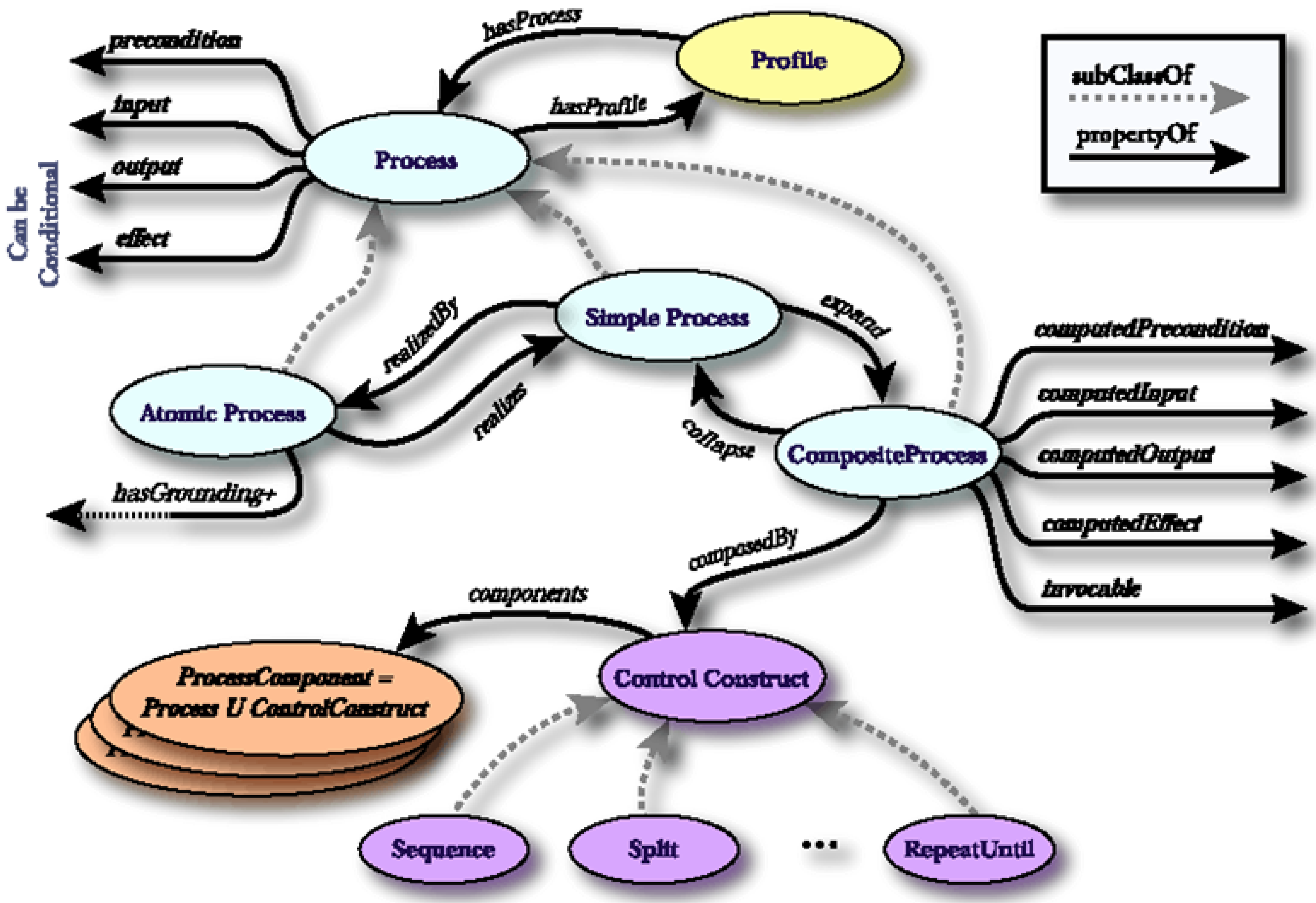
Detailed description of how the service works.

- Each service conceived as an atomic or composite process
- Associated w/ each service is a set of inputs, outputs, preconditions and effects **(function and action metaphor)**
- Composite processes are compositions of simple or other composite processes in terms of constructs such as sequence, if-then-else, fork,...
- Data flow and Control flow should be described for each composite service
- A black box, glass box or abstract views of services can be provided

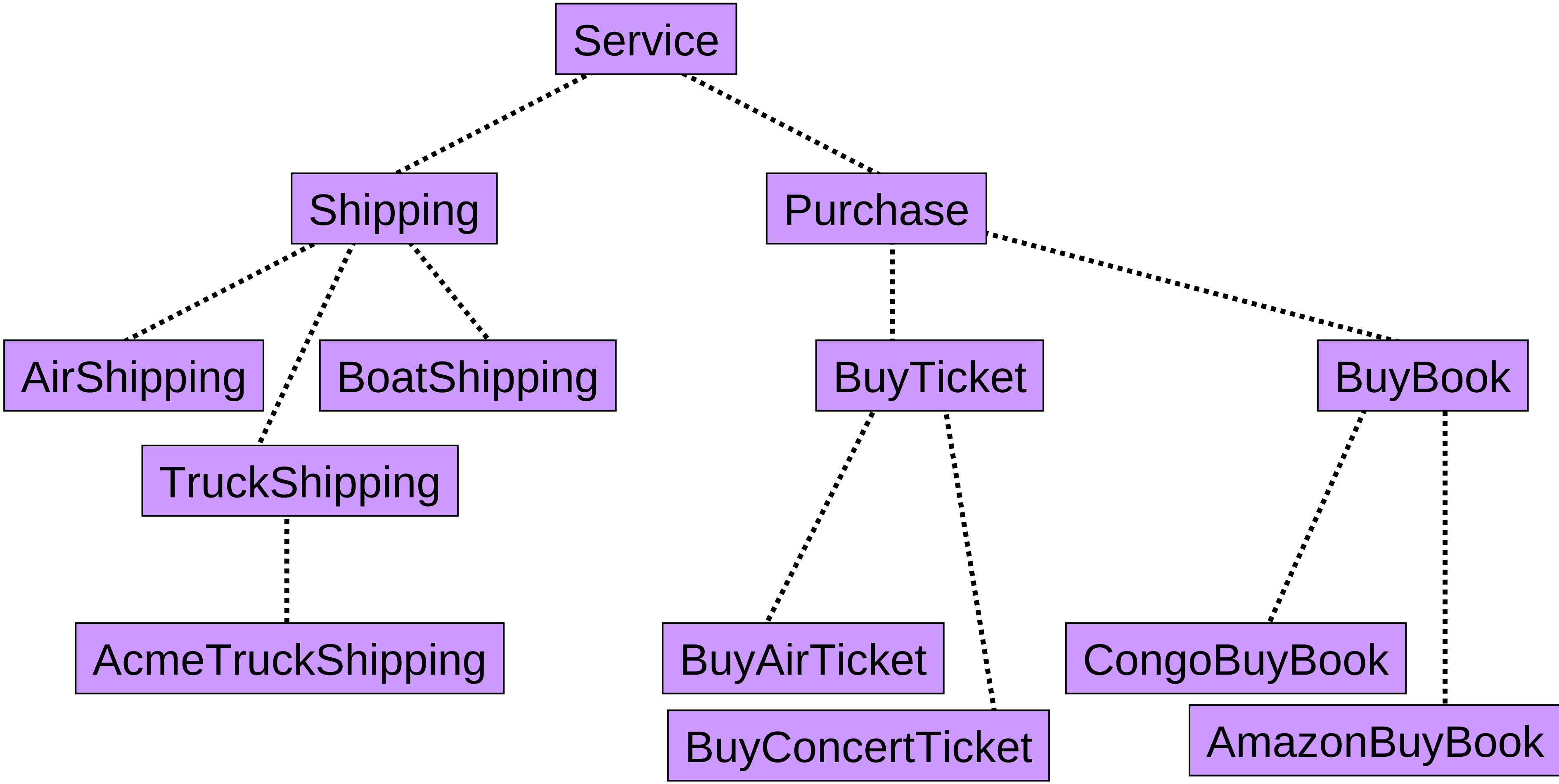
Common Usage:

- (automated) Web service invocation, composition, interoperation, monitoring.

Service Process Model



Ontologies of Services

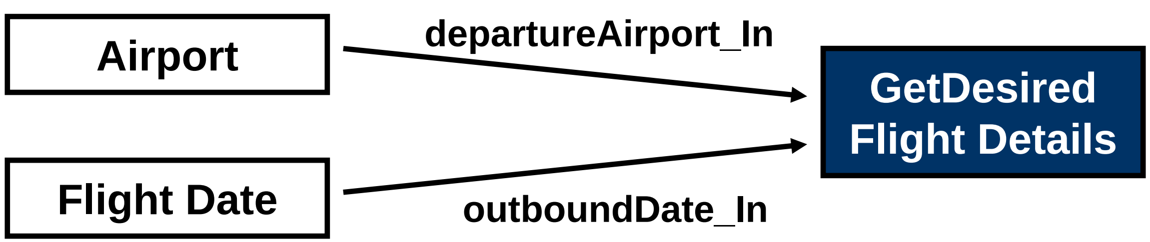


Atomic Process Example

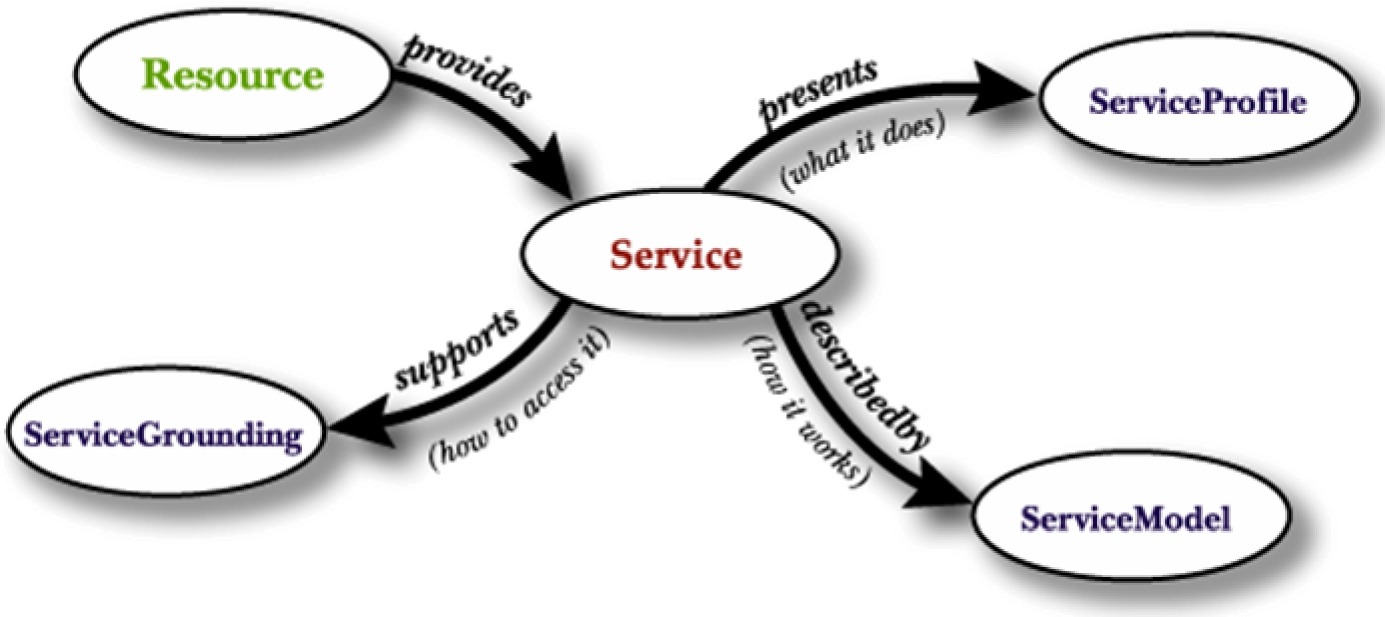
```
<!– Atomic Process Definition - GetDesiredFlightDetails   -->
  <rdfs:Class rdf:ID="GetDesiredFlightDetails">
    <rdfs:subClassOf rdf:resource="http://www.daml.org/Process#AtomicProcess" />
  </rdfs:Class>

<!– (sample) Inputs used by atomic process GetDesiredFlightDetails -->
  <rdf:Property rdf:ID="departureAirport_In">
    <rdfs:subPropertyOf rdf:resource="http://www.daml.org/Process#input" />
    <rdfs:domain rdf:resource="#GetDesiredFlightDetails" />
    <rdfs:range rdf:resource="http://www.daml.ri.cmu.edu/ont/
            DAML-S/concepts.daml#Airport" />
  </rdf:Property>

  <rdf:Property rdf:ID="outbounDate_In">
    <rdfs:subPropertyOf rdf:resource="http://www.daml.org/Process#input" />
    <rdfs:domain rdf:resource="#GetDesiredFlightDetails" />
    <rdfs:range rdf:resource="http://www.daml.ri.cmu.edu/ont/
            DAML-S/concepts.daml#FlightDate" />
  </rdf:Property>
```



The OWL-S Upper Ontology



Issue with OWL and OWL-S

1. Expressiveness & Semantics:
OWL has a well-defined semantics, but it is *not* sufficiently expressive to characterize all and only the intended interpretations of the OWL-S process model, and aspects of the rest of the ontology.

2. The role of the ontology
The process model is a "*description of the pieces of the process model*", rather than the process model itself. We need both.

Expressiveness & Semantics Issues

Syntax Bandaids:

1. A lot of time spent trying to invent solutions for the expressiveness we wanted e.g., formulas, connectives, variables.
2. DRS [McDermott & Dou, 02], Later version (2004)
   http://www.cs.yale.edu/homes/dvm/daml/DRSguide.pdf
   RDF encoding of rules language (predates and generalizes SWRL)

Expressiveness & Semantics Issues

Semantics Band-aids:

1. Distributed operational semantics via Petri Nets.
   [Narayanan & McIlraith, 2002]
2. Interleaving function-based operational semantics w/ subtype polymorphism. [Ankolekar et al., 2002]
3. Semantics via translation to (mostly) first-order logic (situation calculus).
   [Narayanan & McIlraith, 2002], [Gruninger, 2003]

Of course, these establish the semantics of the process model, but *not* within the language.

They enable mapping of OWL-S process models to other richer languages, but do not enable OWL-S itself to be unambiguously computer interpretable.

OWL-S Status

OWL-S is a member submission to the W3C

Version 1.1 is available on the Web

Version 1.2 (the “final” version) is in pre-release on the Web.

OWL-S: http://www.daml.org/services/

Lots of OWL and OWL-S tools available:
http://www.semwebcentral.org

Beyond OWL-S

SWSL (Semantic Web Services Language)

- Part of Joint EU-North American SW Services Initiative (SWSI)
- Committed to forward compatibility with OWL-S, while addressing limitations of the OWL language vis a vis Web services.

http://www.swsi.org/

WSML (Web Service Modeling Language)

- New European initiative (centred in DERI Ireland)

http://www.wsmo.org/

Outline

- OWL-S
- FLOWS: **F**irst-Order **L**ogic **O**ntology for **W**eb **S**ervices

Situating FLOWS

SWSI – Semantic Web Services Initiative

- http://www.swsi.org
- SWSA – SWS Architecture Committee
- SWSL – SWS Language Committee

Situating FLOWS

SWSI – Semantic Web Services Initiative

http://www.swsi.org

SWSA – SWS Architecture Committee

SWSL – SWS Language Committee

SWSF – SWS Framework

http://www.daml.org/services/swsf/

1) SWSO - Ontology

FLOWS – First-order Logic Ontology for Web Services (SWSO-FOL)

ROWS – Rules Ontology for Web Services (SWSO-Rules)

2) SWSL – Language

SWSL-Rules – Rules language

SWSL-FOL – First order language

3) Use Cases

Situating FLOWS

SWSI – Semantic Web Services Initiative

http://www.swsi.org

SWSA – SWS Architecture Committee

SWSL – SWS Language Committee

SWSF – SWS Framework

http://www.daml.org/services/swsf/

1) SWSO - Ontology

FLOWS – First-order Logic Ontology for Web Services (SWSO-FOL)

ROWS – Rules Ontology for Web Services (SWSO-Rules)

2) SWSL – Language

SWSL-Rules – Rules language

SWSL-FOL – First order language

3) Use Cases

Acknowledgements: SWSL Committee

- Steve Battle (Hewlett Packard)
- Abraham Bernstein (University of Zurich)
- Harold Boley (National Research Council of Canada)
- Benjamin Grosof (Massachusetts Institute of Technology)
- Michael Gruninger (University of Toronto)
- Richard Hull (Bell Labs Research, Lucent Technologies)
- Michael Kifer (State University of New York at Stony Brook)
- David Martin (SRI International)
- Sheila McIlraith (University of Toronto)
- Deborah McGuinness (Stanford University)
- Jianwen Su (University of California, Santa Barbara)
- Said Tabet (The RuleML Initiative)

What is FLOWS?

FLOWS is:

a First-order Logic Ontology for Web Services

FLOWS comprises:

- Service Descriptors
- Process Model

FLOWS Process Model

- FLOWS Process Model consists of
  - a subset of the PSL Ontology
  - extensions for service concepts

The bulk of this already exists and has been vetted.

… so here’s an overview of PSL….

PSL Acknowledgements

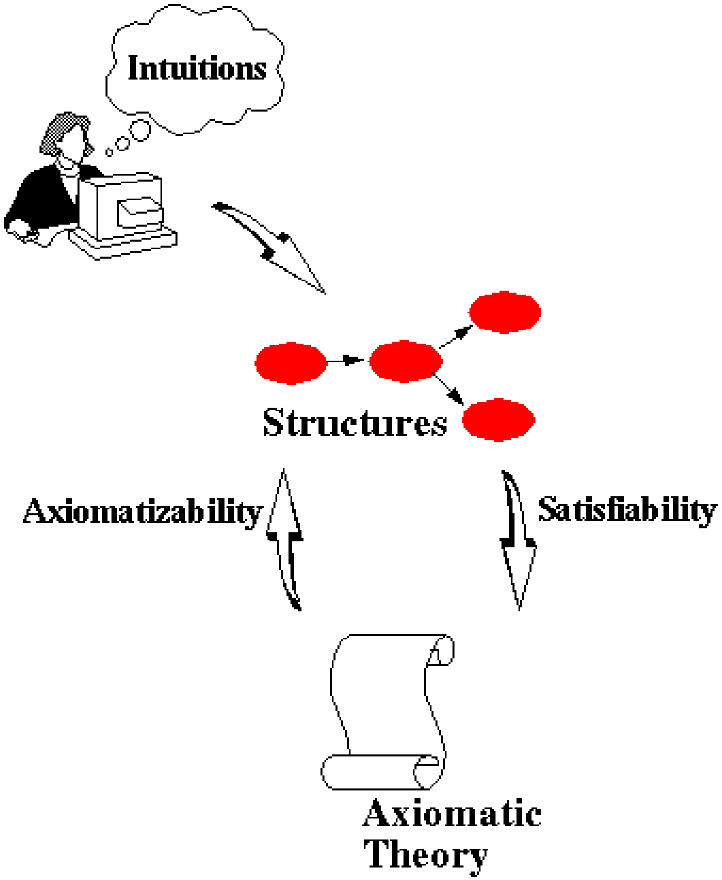
PSL is the joint work of many (old & new) including:

- Michael Gruninger (NIST - now UofT)
- Steve Ray (NIST)
- Craig Schlenoff (NIST)
- Conrad Bock (NIST)
- Josh Lubell (NIST)
- Austin Tate (Edinburgh)
- Steve Polyak (Edinburgh)
- Jintae Lee (Colorado)
- Chris Menzel (Texas A&M)
- Ron Fernandes (KBSI)
- Florence Tissot (KBSI)
- Line Pouchard (Oak Ridge National Labs)
- Anne-Francoise Cutting-Decelle (U. Savioe)
- Jean-Jacques Michel (Wanadoo)
- Bob Young (Loughborough University)
- Joe Kopena (Drexel)
- Kincho Law (Stanford)
- Arturo Sanchez (North Florida)

Process Specification Language

- PSL is a modular, extensible first-order logic ontology capturing concepts required for manufacturing and business process specification
  - PSL is an International Standard (ISO 18629)
  - There are currently 300 concepts across 50 extensions of a common core theory (PSL-Core), each with a set of first-order axioms written in Common Logic (ISO 24707)
  - The core theories of the PSL Ontology extend situation calculus
  - PSL is a verified ontology -- all models of the axioms are isomorphic to models that specify the intended semantics

Verified Ontologies



Formal Properties of PSL

- The meaning of terms in the ontology is characterized by models for first-order logic.
- The PSL Ontology has a first-order axiomatization of the class of models.
- Classes in the ontology arise from classification of the models with respect to invariants (properties of the models preserved by isomorphism).

Definitional Extensions

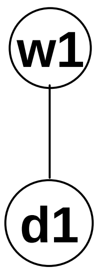
- Preserving semantics is equivalent to preserving models of the axioms.
  - *preserving models = isomorphism*

- Models are classified by using *invariants* (properties of models that are preserved by isomorphism).

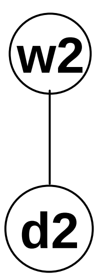
- Classes of activities and objects are specified using these invariants.

PSL Example

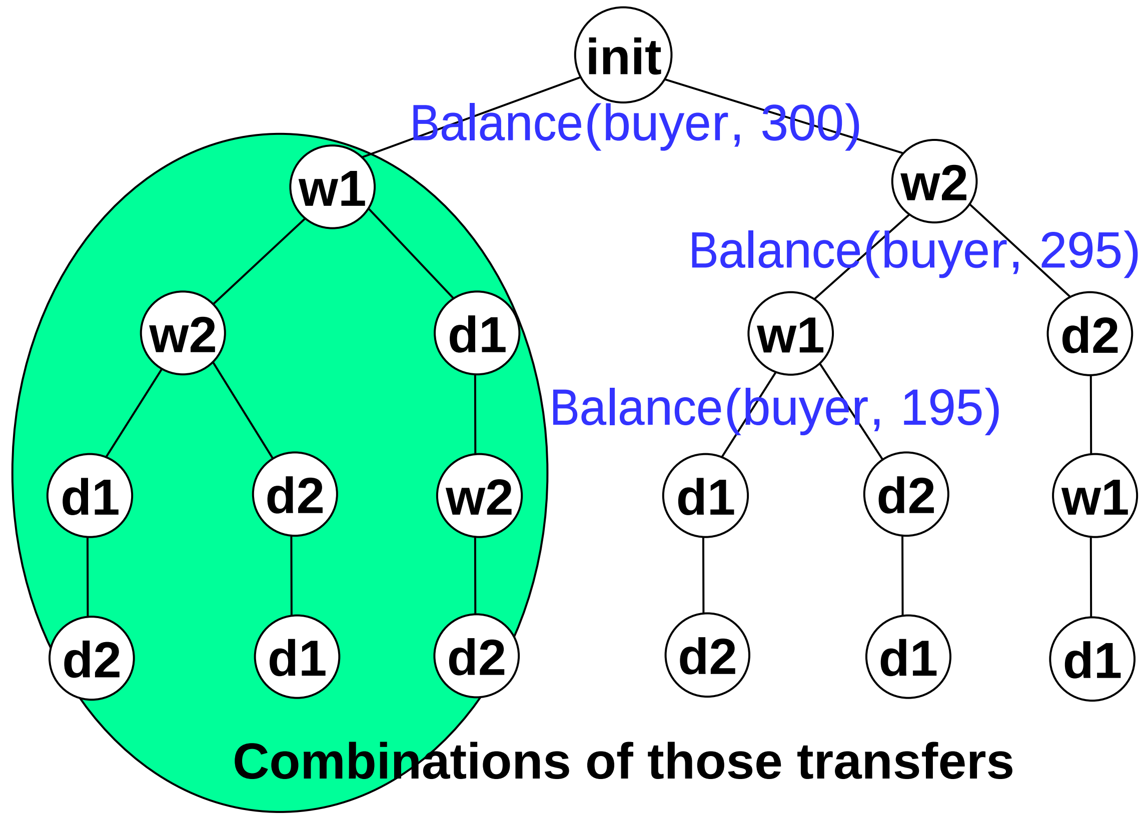
Atomic activities:
- **w1 = withdraw (100, buyer)**
- **d1 = deposit (100, seller)**
- **w2 = withdraw (5, buyer)**
- **d2 = deposit (5, broker)**



transfer(100, buyer, seller)



transfer(5, buyer, broker)



Combinations of those transfers

- Can add constraints, e.g., that w1 must precede w2

PSL Example (cont.)

To transfer money from Account1 to Account2, withdraw some amount from Account1 and deposit the amount in Account2.

```
(forall (?occ)
    (implies  (occurrence_of ?occ (transfer ?Amount ?Account1 ?Account2))
        (exists (?occ1 ?occ2 ?occ3)
            (and    (occurrence_of ?occ1 (withdraw ?Amount ?Account1))
                    (occurrence_of ?occ2 (deposit ?Amount ?Account2))
                    (subactivity_occurrence ?occ1 ?occ)
                    (subactivity_occurrence ?occ2 ?occ)
                    (leaf_occ ?occ3 ?occ1)
                    (min_precedes ?occ3 (root_occ ?occ2))))))
```

FLOWS Process Model

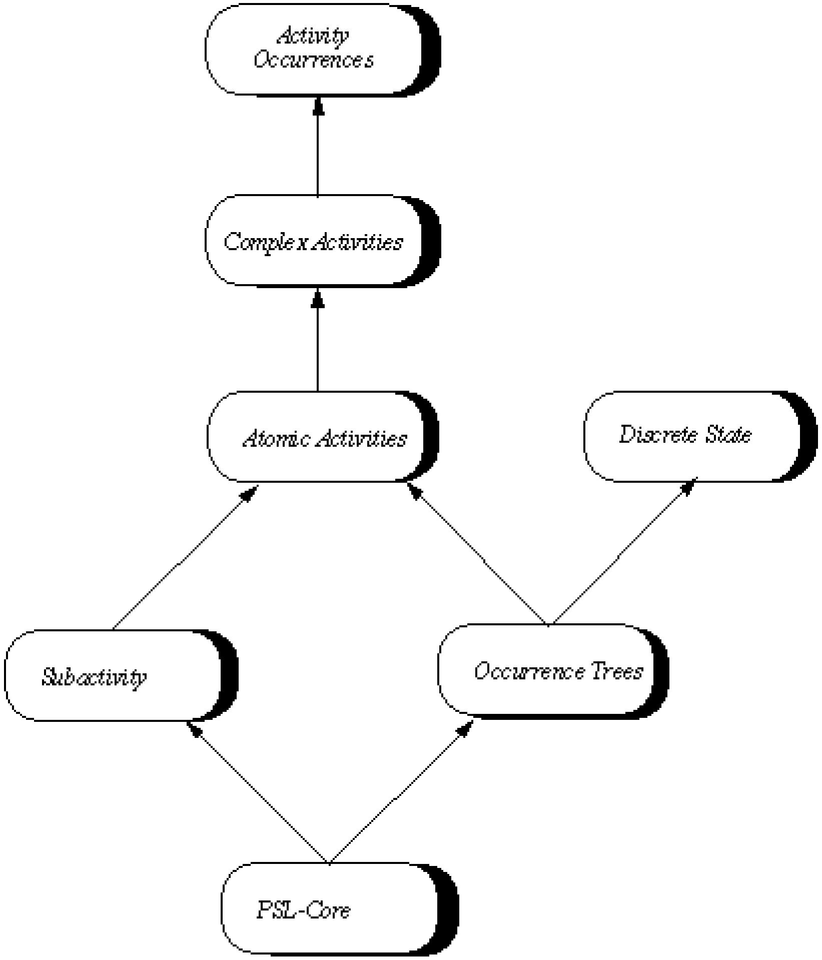
FLOWS-Core

- PSL-Core
- Service, AtomicProcess, composedOf, message, channel

FLOWS Extensions

- Control Constraints
  - Split, Sequence, Unordered, Choice, IfThenElse, Iterate, RepeatUntil
- Ordering Constraints
  - OrderedActivity
- Occurrence Constraints
  - OccActivity
- State Constraints
  - TriggeredActivity
- Exception Constraints
  - Exception

PSL Core Theories



FLOWS-core

- Web service
  - Named object
  - Has non-functional properties
  - Has a PSL activity (which describes the internal process of the service)
  - Can have multiple occurrences (instantiations of the service)
- AtomicProcess
  - Domain specific: analogous to OWL-S atomic processes; can impact “the real world”
  - Service specific: mainly for message handling
    - Create message (which can include place into a channel)
    - Read message
    - Destroy message
  - Also service-specific processes for channels
    - Create channel, destroy, add/delete source, add/delete target
- Messages
  - First-class objects that are created and destroyed, can be read
  - Can be placed on channels (as one mechanism to control data flow)

FLOWS Process Model

FLOWS-Core

- PSL-Core
- Service, AtomicProcess, composedOf, message, channel

FLOWS Extensions

- Control Constraints
  - Split, Sequence, Unordered, Choice, IfThenElse, Iterate, RepeatUntil
- Ordering Constraints
  - OrderedActivity
- Occurrence Constraints
  - OccActivity
- State Constraints
  - TriggeredActivity
- Exception Constraints
  - Exception

How to use FLOWS

1) Describe your web services in FLOWS.
2) Use FLOWS to define the semantics of your favourite modeling paradigm (e.g., UML, ASM, FSM, Petri nets).
   - *FLOWS provides an excellent SWS Framework for relating different WS/process modeling paradigms, ensuring semantic interoperation between different modeling paradigms*.

"How might the programmer-on-the-street describe web services in FLOWS?"

- In the current FLOWS ontology, the "Control Constructs" extension on top of FLOWS-Core provides a flowchart-style process model for the "programmer on the street"
- Other "procedural" models can be incorporated into FLOWS in an analogous manner

Driving home some points

“Reasoning in FOL is too hard.” FLOWS is an ontology. It provides an unambiguous (computer interpretable) specification of a process model. While our driving tasks are characterizable in FOL using entailment and consistency, we are not (necessarily) advocating that they be implemented using a full FOL reasoner. We anticipate the use of highly-optimized special-purpose reasoners.

“Reasoning in FOL is intractable” Problems are intractable, not languages.

“FLOWS/PSL is too hard to learn and write.” We don’t expect the average user to ever see or write in FLOWS . This is the assembly language that ensures everything works correctly. We anticipate 95% of the users working with a much less expressive high-level syntax that hides all these details.

“There’s too much detail in this language.” If you don’t need it, don’t use it, but it’s there if you do need it.

Take Home Message

Please look at:

- Semantic Web Services Framework (SWSF)

  http://www.daml.org/services/swsf/1.0/

  And specifically FLOWS (aka SWSO-FOL)

  http://www.daml.org/services/swsf/1.0/swso

- Process Specification Language (PSL)

  http://www.mel.nist.gov/psl/

- OWL-S

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