Multi-agent and Semantic Web Systems: Web Services: Part 2

Fiona McNeill

School of Informatics

11th March 2013

Recap

- Web Services (WS) can be thought of as Remote Procedure Calls.

Recap

- Web Services (WS) can be thought of as Remote Procedure Calls.

- Messages from a client will specify the operation to be called, and will supply arguments for the operation.

Recap

- Web Services (WS) can be thought of as Remote Procedure Calls.
- Messages from a client will specify the operation to be called, and will supply arguments for the operation.
- The services responds (typically) with the result of the operation on those arguments.

Recap

- Web Services (WS) can be thought of as Remote Procedure Calls.

- Messages from a client will specify the operation to be called, and will supply arguments for the operation.

- The services responds (typically) with the result of the operation on those arguments.

- The messages are standardly sent over HTTP as the body of a SOAP document; the SOAP header contains addressing information.

Recap

- Web Services (WS) can be thought of as Remote Procedure Calls.
- Messages from a client will specify the operation to be called, and will supply arguments for the operation.
- The services responds (typically) with the result of the operation on those arguments.
- The messages are standardly sent over HTTP as the body of a SOAP document; the SOAP header contains addressing information.
- Services are standardly described using WSDL. This specifies
    - types;

Recap

- Web Services (WS) can be thought of as Remote Procedure Calls.
- Messages from a client will specify the operation to be called, and will supply arguments for the operation.
- The services responds (typically) with the result of the operation on those arguments.
- The messages are standardly sent over HTTP as the body of a SOAP document; the SOAP header contains addressing information.
- Services are standardly described using WSDL. This specifies
  - types;
  - operations and their inputs and outputs;

Recap

- Web Services (WS) can be thought of as Remote Procedure Calls.

- Messages from a client will specify the operation to be called, and will supply arguments for the operation.

- The services responds (typically) with the result of the operation on those arguments.

- The messages are standardly sent over HTTP as the body of a SOAP document; the SOAP header contains addressing information.

- Services are standardly described using WSDL. This specifies
  - types;
  - operations and their inputs and outputs;
  - a binding for each operation which specifies the allowed protocol and the service endpoints.

Motivation for Semantic Web Services

- Standard Web Service technology provides virtualisation for distributed computing:

Motivation for Semantic Web Services

- Standard Web Service technology provides virtualisation for distributed computing:
    - abstraction from specific platforms and programming languages;

Motivation for Semantic Web Services

- Standard Web Service technology provides virtualisation for distributed computing:
  - abstraction from specific platforms and programming languages;
  - promotes interoperability of diverse service implementations.

Motivation for Semantic Web Services

- Standard Web Service technology provides virtualisation for distributed computing:
    - abstraction from specific platforms and programming languages;
    - promotes interoperability of diverse service implementations.
- But foundation for automating Web Services still lacking.

Motivation for Semantic Web Services

- Standard Web Service technology provides virtualisation for distributed computing:
  - abstraction from specific platforms and programming languages;
  - promotes interoperability of diverse service implementations.
- But foundation for automating Web Services still lacking.
- Semantic WS intended to supplement standard WS.

Motivation for Semantic Web Services

- Standard Web Service technology provides virtualisation for distributed computing:
  - abstraction from specific platforms and programming languages;
  - promotes interoperability of diverse service implementations.
- But foundation for automating Web Services still lacking.
- Semantic WS intended to supplement standard WS.
- By providing semantically explicit metadata for WS:

Motivation for Semantic Web Services

- Standard Web Service technology provides virtualisation for distributed computing:
  - abstraction from specific platforms and programming languages;
  - promotes interoperability of diverse service implementations.
- But foundation for automating Web Services still lacking.
- Semantic WS intended to supplement standard WS.
- By providing semantically explicit metadata for WS:
  - software can interpret descriptions of unfamiliar WS, and

Motivation for Semantic Web Services

- Standard Web Service technology provides virtualisation for distributed computing:
  - abstraction from specific platforms and programming languages;
  - promotes interoperability of diverse service implementations.
- But foundation for automating Web Services still lacking.
- Semantic WS intended to supplement standard WS.
- By providing semantically explicit metadata for WS:
  - software can interpret descriptions of unfamiliar WS, and
  - carry out discovery, composition, etc.

Motivation for Semantic Web Services

- Standard Web Service technology provides virtualisation for distributed computing:
    - abstraction from specific platforms and programming languages;
    - promotes interoperability of diverse service implementations.
- But foundation for automating Web Services still lacking.
- Semantic WS intended to supplement standard WS.
- By providing semantically explicit metadata for WS:
    - software can interpret descriptions of unfamiliar WS, and
    - carry out discovery, composition, etc.
- OWL-S builds on OWL to provide OWL descriptions of Services.

OWL Digression

- RDFS allows us to build simple class hierarchies for describing ontological structure.

OWL Digression

- RDFS allows us to build simple class hierarchies for describing ontological structure.

- OWL (Web Ontology Language) gives us a richer framework:

OWL Digression

- RDFS allows us to build simple class hierarchies for describing ontological structure.
- OWL (Web Ontology Language) gives us a richer framework:
  - syntactically layered on RDF

OWL Digression

- RDFS allows us to build simple class hierarchies for describing ontological structure.
- OWL (Web Ontology Language) gives us a richer framework:
  - syntactically layered on RDF
  - uses theoretical framework of Description Logic (decidable fragment of First Order Logic);

OWL Digression

- RDFS allows us to build simple class hierarchies for describing ontological structure.
- OWL (Web Ontology Language) gives us a richer framework:
  - syntactically layered on RDF
  - uses theoretical framework of Description Logic (decidable fragment of First Order Logic);
  - a language for describing 'concepts' (classes of instances).

OWL Digression

- RDFS allows us to build simple class hierarchies for describing ontological structure.
- OWL (Web Ontology Language) gives us a richer framework:
  - syntactically layered on RDF
  - uses theoretical framework of Description Logic (decidable fragment of First Order Logic);
  - a language for describing 'concepts' (classes of instances).
  - Provides negation, and standard notion of logical consistency;

OWL Digression

- RDFS allows us to build simple class hierarchies for describing ontological structure.

- OWL (Web Ontology Language) gives us a richer framework:
  - syntactically layered on RDF
  - uses theoretical framework of Description Logic (decidable fragment of First Order Logic);
  - a language for describing 'concepts' (classes of instances).
  - Provides negation, and standard notion of logical consistency;
  - provides operators for defining classes as well as introducing primitive classes;

OWL Digression

- RDFS allows us to build simple class hierarchies for describing ontological structure.

- OWL (Web Ontology Language) gives us a richer framework:
  - syntactically layered on RDF
  - uses theoretical framework of Description Logic (decidable fragment of First Order Logic);
  - a language for describing 'concepts' (classes of instances).
  - Provides negation, and standard notion of logical consistency;
  - provides operators for defining classes as well as introducing primitive classes;
  - provides a limited form of quantification.

Syntax and Semantics of DL Concepts

Simple Concepts

Giraffe	$\{x \mid \text{Giraffe}(x)\}$

Syntax and Semantics of DL Concepts

Simple Concepts

Giraffe	$\{x \mid \text{Giraffe}(x)\}$

Composed Concepts

Brother $\sqcup$ Sister	$\{x \mid \text{Brother}(x) \vee \text{Sister}(x)\}$
Adult $\sqcap$ Male	$\{x \mid \text{Adult}(x) \wedge \text{Male}(x)\}$
$\neg$ Married	$\{x \mid \neg\text{Married}(x)\}$

Syntax and Semantics of DL Concepts

Simple Concepts

Giraffe	$\{x \mid \text{Giraffe}(x)\}$

Composed Concepts

Brother $\sqcup$ Sister	$\{x \mid \text{Brother}(x) \vee \text{Sister}(x)\}$
Adult $\sqcap$ Male	$\{x \mid \text{Adult}(x) \wedge \text{Male}(x)\}$
$\neg$ Married	$\{x \mid \neg\text{Married}(x)\}$

Subsumption

Giraffe $\sqsubset$ Mammal	$\forall x(\text{Giraffe}(x) \rightarrow \text{Mammal}(x))$

Syntax and Semantics of DL Concepts

Simple Concepts

Giraffe	$\{x \mid \text{Giraffe}(x)\}$

Composed Concepts

$\text{Brother} \sqcup \text{Sister}$	$\{x \mid \text{Brother}(x) \vee \text{Sister}(x)\}$
$\text{Adult} \sqcap \text{Male}$	$\{x \mid \text{Adult}(x) \wedge \text{Male}(x)\}$
$\neg\,\text{Married}$	$\{x \mid \neg\text{Married}(x)\}$

Subsumption

$\text{Giraffe} \sqsubseteq \text{Mammal}$	$\forall x(\text{Giraffe}(x) \rightarrow \text{Mammal}(x))$

Definitional Equivalence

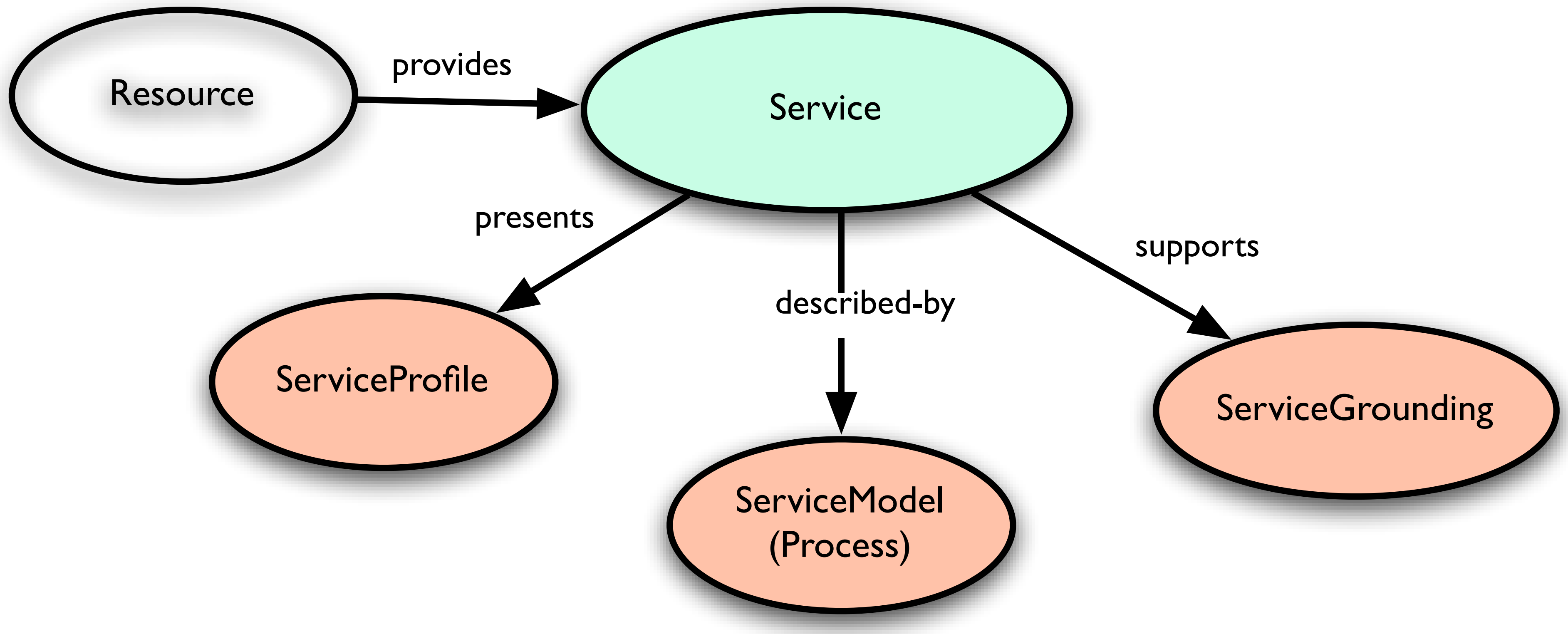
$\text{Sibling} \doteq \text{Brother} \sqcup \text{Sister}$	$\forall x(\text{Sibling}(x) \leftrightarrow \text{Brother}(x) \vee \text{Sister}(x))$

OWL-S View of Services

- Based on DAML (Darpa Agent Markup Language) and DAML-S.
- Provides an ontology for web services that consists of three sub-ontologies.

Service Profile:	How the service presents itself to the external world.
Service Model:	What the service does, and how the client interacts with it.
Service Grounding:	How the service is realised — analogous to WSDL binding.

OWL-S Service Ontology



Service Model: Inputs and Outputs

- OWL-S functional description of services very similar to WSDL.

Service Model: Inputs and Outputs

- OWL-S functional description of services very similar to WSDL.
- Inputs and outputs specify the data transformation produced by the process.

Service Model: Inputs and Outputs

- OWL-S functional description of services very similar to WSDL.
- Inputs and outputs specify the data transformation produced by the process.
- General notion of `Parameter`;

Service Model: Inputs and Outputs

- OWL-S functional description of services very similar to WSDL.
- Inputs and outputs specify the data transformation produced by the process.
- General notion of `Parameter`;
- The type of (values of) the `Parameter` is specified with a URI.

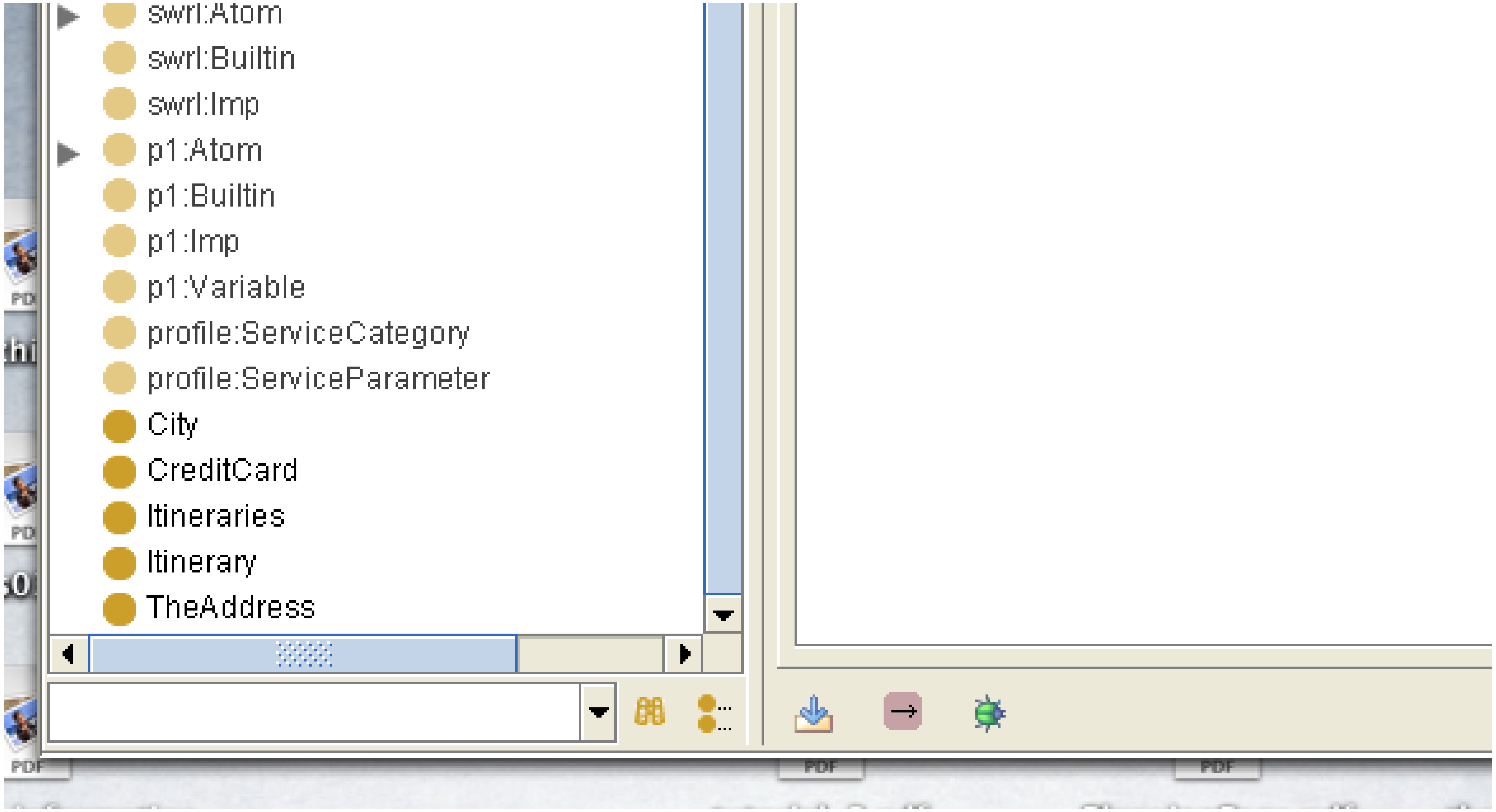
Service Model: Inputs and Outputs

- OWL-S functional description of services very similar to WSDL.
- Inputs and outputs specify the data transformation produced by the process.
- General notion of `Parameter`;
- The type of (values of) the `Parameter` is specified with a URI.
- Typically, this will be a pointer to an OWL class in a domain ontology.
  - `Input`,`Output` $\sqsubset$ `Parameter`

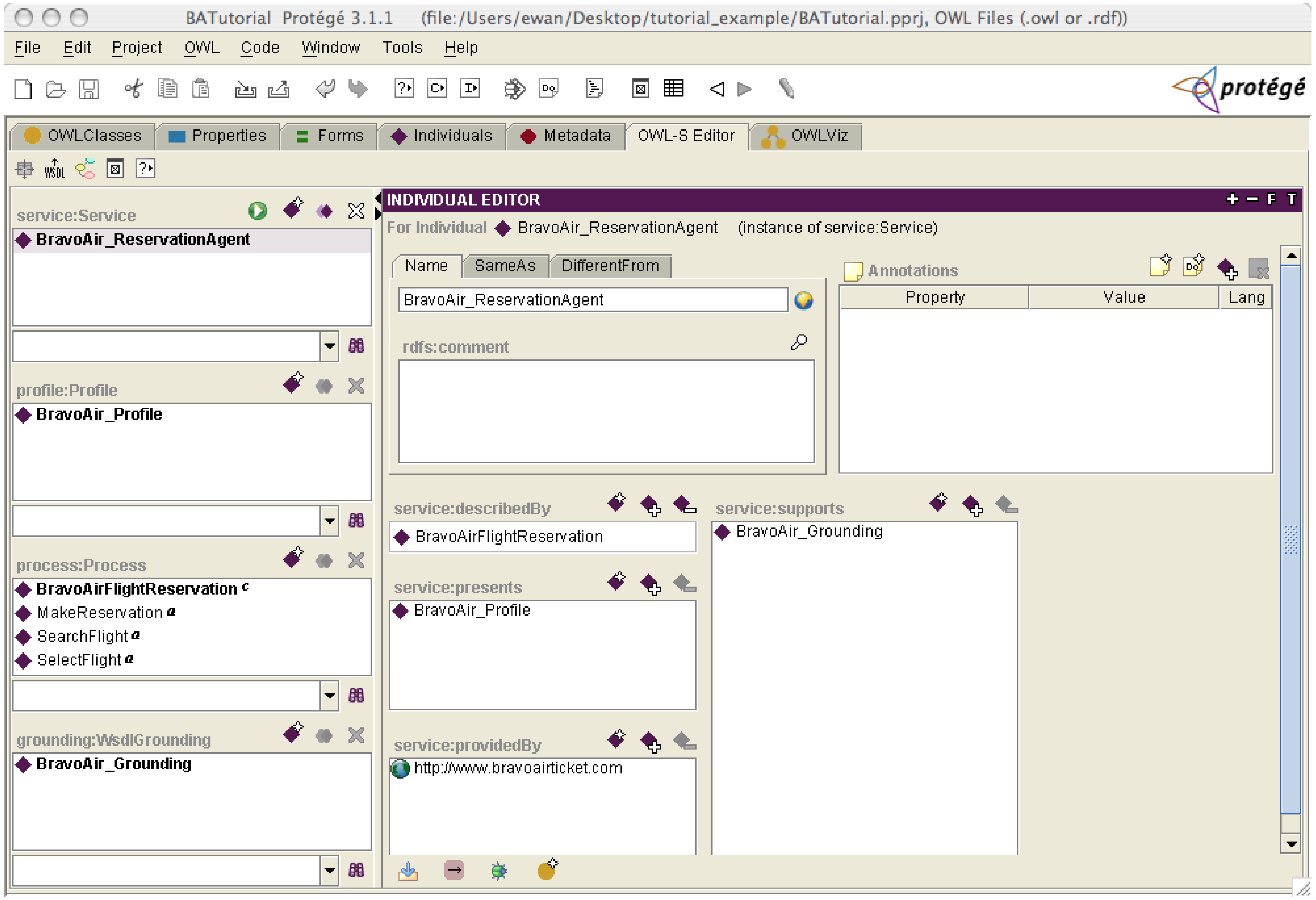
Service Model: Inputs and Outputs

- OWL-S functional description of services very similar to WSDL.
- Inputs and outputs specify the data transformation produced by the process.
- General notion of `Parameter`;
- The type of (values of) the `Parameter` is specified with a URI.
- Typically, this will be a pointer to an OWL class in a domain ontology.
  - `Input`,`Output` $\sqsubset$ `Parameter`
- Parameters are associated with services via property `hasParameter`:
  - `hasInput`, `hasOutput` sub-properties of `hasParameter`

OWL-S Plugin for Protégé: Domain Ontology


swrl:Atom
swrl:Builtin
swrl:Imp
p1:Atom
p1:Builtin
p1:Imp
p1:Variable
profile:ServiceCategory
profile:ServiceParameter
City
CreditCard
Itineraries
Itinerary
TheAddress

OWL-S Plugin for Protégé: OWL-S Service


BATutorial Protégé 3.1.1 (file:/Users/ewan/Desktop/tutorial_example/BATutorial.pprj, OWL Files (.owl or .rdf))
File Edit Project OWL Code Window Tools Help
protégé
OWLClasses Properties Forms Individuals Metadata OWL-S Editor OWLViz
INDIVIDUAL EDITOR
Name SameAs DifferentFrom
Annotations
Property Value Lang
rdfs:comment

service:Service
- BravoAir_ReservationAgent

profile:Profile
- BravoAir_Profile

process:Process
- BravoAirFlightReservation c
- MakeReservation a
- SearchFlight a
- SelectFlight a

grounding:WsdlGrounding
- BravoAir_Grounding

For Individual BravoAir_ReservationAgent (instance of service:Service)

Name: BravoAir_ReservationAgent

service:describedBy: BravoAirFlightReservation

service:presents: BravoAir_Profile

service:providedBy: http://www.bravoairticket.com

service:supports: BravoAir_Grounding

Service Model: Participants

- A process involves two or more agents.

Service Model: Participants

- A process involves two or more agents.
- Required agents:
  - `TheClient` — the service is described from the point of view of the client.

Service Model: Participants

- A process involves two or more agents.
- Required agents:
  - `TheClient` — the service is described from the point of view of the client.
  - `TheServer`—principal element of the service that the client deals with.

State Transformations

Question: Can Web Services change the world?

State Transformations

Question: Can Web Services change the world?

Answer: Yes, if it involves shunting bits rather than atoms.

State Transformations

Question: Can Web Services change the world?

Answer: Yes, if it involves shunting bits rather than atoms.

Changing the world with WS

Before invoking Amazon: your net assets are £999.00.

State Transformations

Question: Can Web Services change the world?

Answer: Yes, if it involves shunting bits rather than atoms.

Changing the world with WS

Before invoking Amazon: your net assets are £999.00.
After invoking Amazon: your net assets are £000.00, but you are now the proud owner of a Widescreen Plasma TV.

Preconditions and Effects

OWL-S distinguishes two aspects of WS:

Preconditions and Effects

OWL-S distinguishes two aspects of WS:

1. transforming information — inputs and outputs

Preconditions and Effects

OWL-S distinguishes two aspects of WS:

1. transforming information — inputs and outputs
2. transforming the world — preconditions and effects

Preconditions and Effects

OWL-S distinguishes two aspects of WS:

1. transforming information — inputs and outputs
2. transforming the world — preconditions and effects

Example Preconditions

$\texttt{valid(creditcard, t0)} \wedge \texttt{limit(creditcard)} \geq \texttt{£999.00}$

Preconditions and Effects

OWL-S distinguishes two aspects of WS:

1. transforming information — inputs and outputs
2. transforming the world — preconditions and effects

Example Preconditions

$\texttt{valid(creditcard, t0)} \wedge \texttt{limit(creditcard)} \geq \texttt{£999.00}$

Example Effect

$(\texttt{debt(creditcard, t1) =}$
$\texttt{debt(creditcard, t0)} - \texttt{£999.00})$
$\wedge\ \texttt{own(i, TV, t1)}$

Preconditions and Effects

OWL-S distinguishes two aspects of WS:

1. transforming information — inputs and outputs
2. transforming the world — preconditions and effects

Example Preconditions

$\texttt{valid(creditcard, t0)} \wedge \texttt{limit(creditcard)} \geq \texttt{£999.00}$

Example Effect

$(\texttt{debt(creditcard, t1) =}$
$\texttt{debt(creditcard, t0)} - \texttt{£999.00})$
$\wedge\ \texttt{own(i, TV, t1)}$

IOPEs

IOPE = Input, Output, Precondition and Effect.

Expressing Preconditions and Effects

Expressing Truths about the World

Preconditions and effects need to be stated in terms of a reasonably expressive logical language. By themselves, RDF and OWL do not provide a good basis for such a language. (Why?)

Embedding Logic in OWL-S

- Logic and the Semantic Web — rather a mess!
    - `http://www.w3.org/DesignIssues/Logic`
    - Fensel & van Harmelen (2007)

Embedding Logic in OWL-S

- Logic and the Semantic Web — rather a mess!
    - `http://www.w3.org/DesignIssues/Logic`
    - Fensel & van Harmelen (2007)

- OWL-S tries to be non-committal about choice of logical language, makes a number of suggestions:

Embedding Logic in OWL-S

- Logic and the Semantic Web — rather a mess!
  - `http://www.w3.org/DesignIssues/Logic`
  - Fensel & van Harmelen (2007)

- OWL-S tries to be non-committal about choice of logical language, makes a number of suggestions:

N3	Extensions beyond RDF for expressing logical rules.

Embedding Logic in OWL-S

- Logic and the Semantic Web — rather a mess!
  - `http://www.w3.org/DesignIssues/Logic`
  - Fensel & van Harmelen (2007)

- OWL-S tries to be non-committal about choice of logical language, makes a number of suggestions:

N3	Extensions beyond RDF for expressing logical rules.
RuleML	`http://www.ruleml.org/` — broader than deductive logic; XML-based; somewhat orthogonal to other efforts.

Embedding Logic in OWL-S

- Logic and the Semantic Web — rather a mess!
  - `http://www.w3.org/DesignIssues/Logic`
  - Fensel & van Harmelen (2007)

- OWL-S tries to be non-committal about choice of logical language, makes a number of suggestions:

N3	Extensions beyond RDF for expressing logical rules.
RuleML	`http://www.ruleml.org/` — broader than deductive logic; XML-based; somewhat orthogonal to other efforts.
SWRL	(Semantic Web Rule Language) `http://www.w3.org/Submission/SWRL/` — embeds OWL assertions in Horn-clause rules.

Embedding Logic in OWL-S

- Logic and the Semantic Web — rather a mess!
  - `http://www.w3.org/DesignIssues/Logic`
  - Fensel & van Harmelen (2007)

- OWL-S tries to be non-committal about choice of logical language, makes a number of suggestions:

N3	Extensions beyond RDF for expressing logical rules.
RuleML	`http://www.ruleml.org/` — broader than deductive logic; XML-based; somewhat orthogonal to other efforts.
SWRL	(Semantic Web Rule Language) `http://www.w3.org/Submission/SWRL/` — embeds OWL assertions in Horn-clause rules.
SWRL-FOL	`http://www.daml.org/2004/11/fol/proposal` — extension of SWRL to arbitrary FOL formulas

Embedding Logic in OWL-S

- Logic and the Semantic Web — rather a mess!
  - `http://www.w3.org/DesignIssues/Logic`
  - Fensel & van Harmelen (2007)

- OWL-S tries to be non-committal about choice of logical language, makes a number of suggestions:

N3	Extensions beyond RDF for expressing logical rules.
RuleML	`http://www.ruleml.org/` — broader than deductive logic; XML-based; somewhat orthogonal to other efforts.
SWRL	(Semantic Web Rule Language) `http://www.w3.org/Submission/SWRL/` — embeds OWL assertions in Horn-clause rules.
SWRL-FOL	`http://www.daml.org/2004/11/fol/proposal` — extension of SWRL to arbitrary FOL formulas
SPARQL	Partial specification of entailment over RDF(S) graphs.

Embedding Logic in OWL-S

- Logic and the Semantic Web — rather a mess!
  - `http://www.w3.org/DesignIssues/Logic`
  - Fensel & van Harmelen (2007)

- OWL-S tries to be non-committal about choice of logical language, makes a number of suggestions:

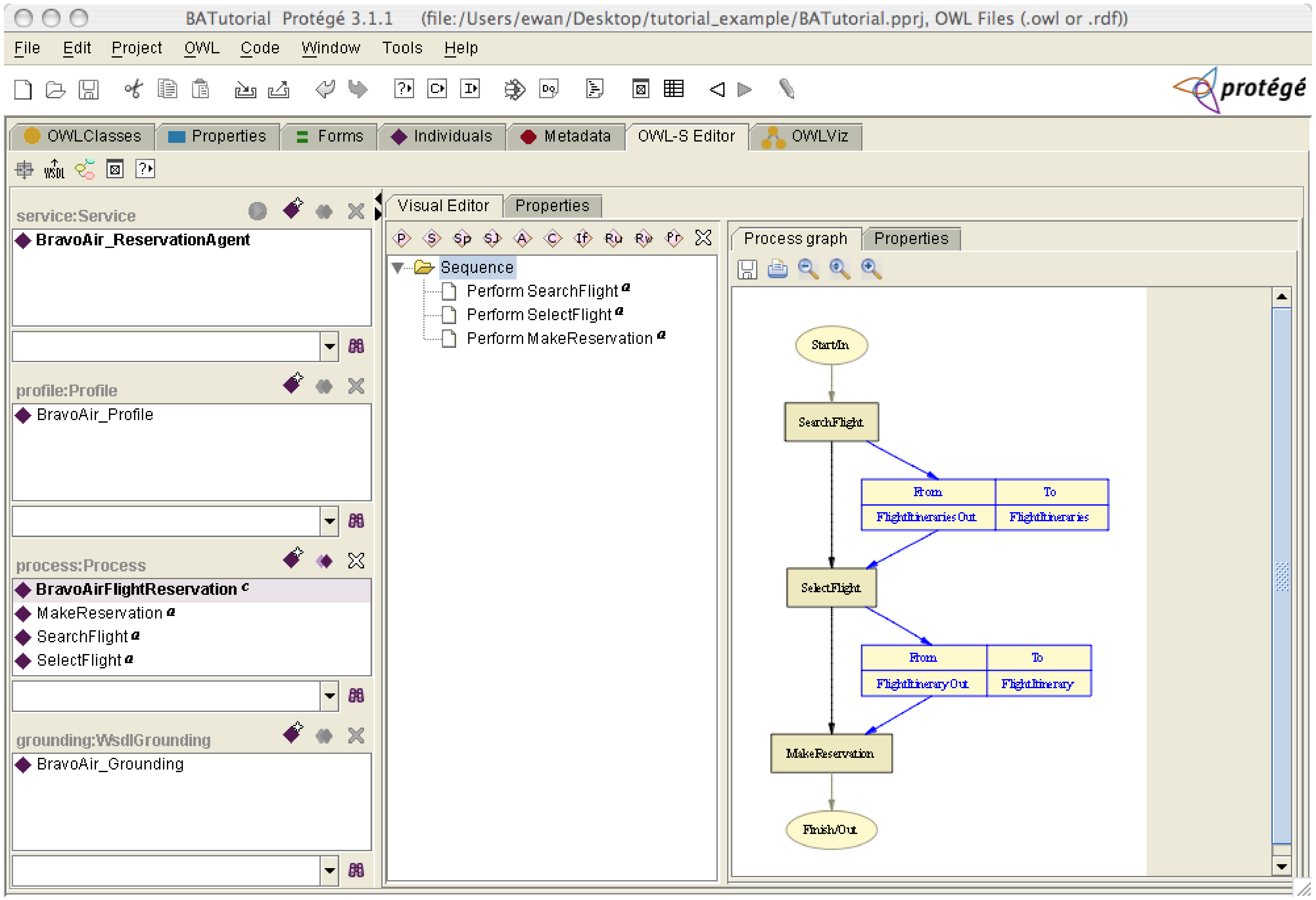
N3	Extensions beyond RDF for expressing logical rules.
RuleML	`http://www.ruleml.org/` — broader than deductive logic; XML-based; somewhat orthogonal to other efforts.
SWRL	(Semantic Web Rule Language) `http://www.w3.org/Submission/SWRL/` — embeds OWL assertions in Horn-clause rules.
SWRL-FOL	`http://www.daml.org/2004/11/fol/proposal` — extension of SWRL to arbitrary FOL formulas
SPARQL	Partial specification of entailment over RDF(S) graphs.

- In OWL-S, expressions from these languages can be embedded as RDF literals.

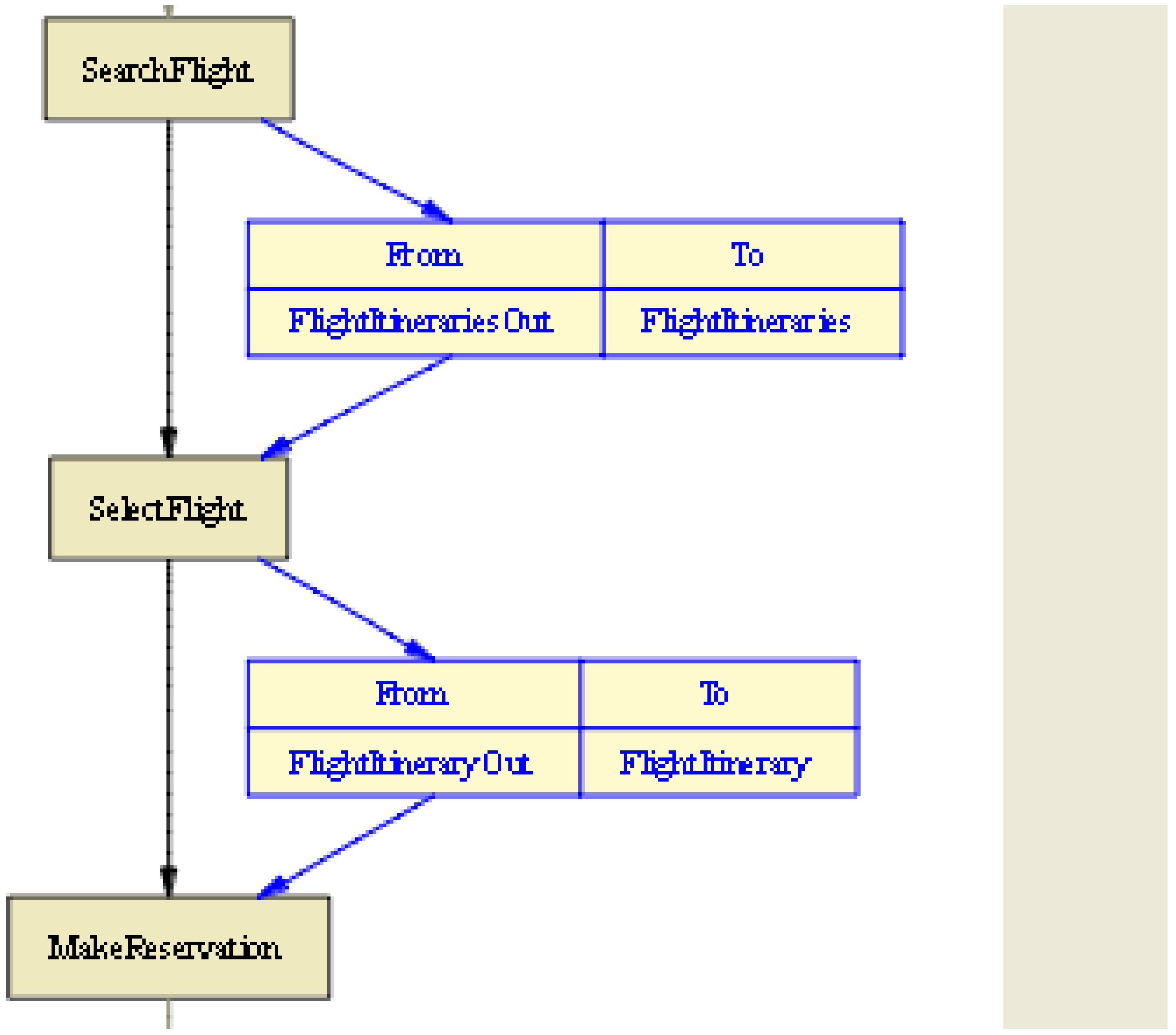
The Process Ontology

- OWL-S divides processes into
  - atomic, and
  - composite.
- Various constructors are provided for assembling composite processes out of component ones, e.g.,
  - sequence,
  - choice,
  - iterate, etc.
- A composite process represents behaviour a client can perform by sending and receiving messages.
- Inputs of an standalone atomic process must come directly from client;
- Inputs of components of a composite process may come from preceding steps.

OWL-S Plugin for Protégé: Process I


BATutorial Protégé 3.1.1 (file:/Users/ewan/Desktop/tutorial_example/BATutorial.pprj, OWL Files (.owl or .rdf))
File Edit Project OWL Code Window Tools Help
OWLClasses Properties Forms Individuals Metadata OWL-S Editor OWLViz
service:Service
BravoAir_ReservationAgent
profile:Profile
BravoAir_Profile
process:Process
BravoAirFlightReservation
MakeReservation
SearchFlight
SelectFlight
grounding:WsdlGrounding
BravoAir_Grounding
Visual Editor Properties
Sequence
Perform SearchFlight
Perform SelectFlight
Perform MakeReservation
Process graph Properties
Start/In
SearchFlight
From To
FlightItinerariesOut FlightItineraries
SelectFlight
From To
FlightItineraryOut FlightItinerary
MakeReservation
Finish/Out

OWL-S Plugin for Protégé: Process 2



Abstracting over Composite Processes

- Composite processes can be viewed at a higher level of abstraction, as simple processes.
- Allows layering, i.e. composite processes can be incorporated as simple processes into further composites.

Service Profile

- Description of the service that can be used by registry or broker.
- Once a client has chosen to engage with a service, uses the Service Model, not the Profile.
- By default, Profile uses same IOPEs as the Model, but this is not mandatory.
- Can also include information such as Service Category and Quality of Service (QoS).

Grounding

- Mapping from abstract specification to a concrete specification of service;

Grounding

- Mapping from abstract specification to a concrete specification of service;
- particularly, those service elements required for interaction.

Grounding

- Mapping from abstract specification to a concrete specification of service;
- particularly, those service elements required for interaction.
- For OWL-S, main issue is relating inputs and outputs of atomic process to the input and outputs of a WSDL operation.

Grounding

- Mapping from abstract specification to a concrete specification of service;
- particularly, those service elements required for interaction.
- For OWL-S, main issue is relating inputs and outputs of atomic process to the input and outputs of a WSDL operation.
- WSDL by default specifies types using XML Schema,

Grounding

- Mapping from abstract specification to a concrete specification of service;

- particularly, those service elements required for interaction.

- For OWL-S, main issue is relating inputs and outputs of atomic process to the input and outputs of a WSDL operation.

- WSDL by default specifies types using XML Schema,

- but OWL classes could be defined (using OWL namespace) in `types` section, or

Grounding

- Mapping from abstract specification to a concrete specification of service;
- particularly, those service elements required for interaction.
- For OWL-S, main issue is relating inputs and outputs of atomic process to the input and outputs of a WSDL operation.
- WSDL by default specifies types using XML Schema,
- but OWL classes could be defined (using OWL namespace) in `types` section, or
- referenced from within a WSDL `operation` definition using an `owl-s-parameter` attribute.

Reading

- http://www.w3.org/Submission/OWL-S
- http://www.daml.org/services/owl-s/1.0/

Bringing Semantics to Web Services with OWL-S
David Martin et al. (2007) *2007 World Wide Web Journal*, Volume 10, Number 3, pp. 243–277.

Unifying Reasoning and Search to Web Scale
Dieter Fensel and Frank van Harmelen (2007) *Internet Computing*, IEEE Volume 11, Issue 2, March-April, pp. 96–95

Summary

- OWL-S provides an upper ontology for web services:
  - Profile,
  - Process, and
  - Grounding.

Summary

- OWL-S provides an upper ontology for web services:
  - Profile,
  - Process, and
  - Grounding.

- OWL-S allows service inputs and outputs to be typed in terms of OWL classes.

Summary

- OWL-S provides an upper ontology for web services:
  - Profile,
  - Process, and
  - Grounding.

- OWL-S allows service inputs and outputs to be typed in terms of OWL classes.

- Latter are typically drawn from a domain ontology.

Summary

- OWL-S provides an upper ontology for web services:
  - Profile,
  - Process, and
  - Grounding.

- OWL-S allows service inputs and outputs to be typed in terms of OWL classes.

- Latter are typically drawn from a domain ontology.

- OWL-S supplements functional descriptions with preconditions and effects.

Summary

- OWL-S provides an upper ontology for web services:
  - Profile,
  - Process, and
  - Grounding.

- OWL-S allows service inputs and outputs to be typed in terms of OWL classes.

- Latter are typically drawn from a domain ontology.

- OWL-S supplements functional descriptions with preconditions and effects.

Summary

- OWL-S provides an upper ontology for web services:
  - Profile,
  - Process, and
  - Grounding.

- OWL-S allows service inputs and outputs to be typed in terms of OWL classes.

- Latter are typically drawn from a domain ontology.

- OWL-S supplements functional descriptions with preconditions and effects.

- The logic for these is embedded as RDF literals.

Summary

- OWL-S provides an upper ontology for web services:
  - Profile,
  - Process, and
  - Grounding.

- OWL-S allows service inputs and outputs to be typed in terms of OWL classes.

- Latter are typically drawn from a domain ontology.

- OWL-S supplements functional descriptions with preconditions and effects.

- The logic for these is embedded as RDF literals.

- Service Grounding is realised in terms of a mapping to WSDL.