

OWL-S for Describing Artifacts

Rossella Rubino, **Ambra Molesini**, Enrico Denti
`ambra.molesini@unibo.it`

ALMA MATER STUDIORUM—Università di Bologna

Lisbon, 14–15 December 2006



- 1 Introduction
- 2 Artifacts
- 3 OWL-S
- 4 OWL-S and Artifacts
- 5 Conclusions and Future Work



Introduction

- Agents never live alone:
 - they coexist with other agents in a MAS,
 - within an *environment* where they act and interact.
- Artifacts are adopted as the basic building blocks to engineer the MAS environment (A&A metamodel).



Introduction

- Agents never live alone:
 - they coexist with other agents in a MAS,
 - within an *environment* where they act and interact.
- Artifacts are adopted as the basic building blocks to engineer the MAS environment (A&A metamodel).



Artifacts

- Artifacts take the form of objects or tools that agents *share* and *use* to
 - support their activities,
 - achieve their objectives.
- Artifacts allow to
 - model the environment as a first-class entity,
 - engineer the space of interaction among agents,
 - enrich MAS design with social and organisational structure, topological models, as well as (complex) security models.



Artifacts' Features

Usage Interface

The set of operations provided by an artifact.

The description of the procedure an agent has to follow to meaningfully interact with an artifact over time.

The description of the functionality provided by the artifact, which agents can use essentially for artifact selection.



Artifacts' Features

Usage Interface

The set of operations provided by an artifact.

Operating Instructions

The description of the procedure an agent has to follow to meaningfully interact with an artifact over time.

The description of the functionality provided by the artifact, which agents can use essentially for artifact selection.



Artifacts' Features

Usage Interface

The set of operations provided by an artifact.

Operating Instructions

The description of the procedure an agent has to follow to meaningfully interact with an artifact over time.

Function Description

The description of the functionality provided by the artifact, which agents can use essentially for artifact selection.



OWL-S

- OWL-S (Ontology Web Language for Services)
 - describes properties and capabilities of Web Services in an unambiguous way,
 - facilitates automated Web Service discovery, execution, composition and inter-operation.
- The discovery, selection and exploitation of a service calls for a suitable description of:
 - what the service does,
 - how it works,
 - how to access it.



Knowledge about a Service

Service Profile

The description of what can be accomplished by the service (limitation, quality of service, ...).

The description of the semantic content of requests and the conditions under which particular outcomes may occur.

The specification of how to access the service.



Knowledge about a Service

Service Profile

The description of what can be accomplished by the service (limitation, quality of service, ...).

Service Model

The description of the semantic content of requests and the conditions under which particular outcomes may occur.

The specification of how to access the service.



Knowledge about a Service

Service Profile

The description of what can be accomplished by the service (limitation, quality of service, ...).

Service Model

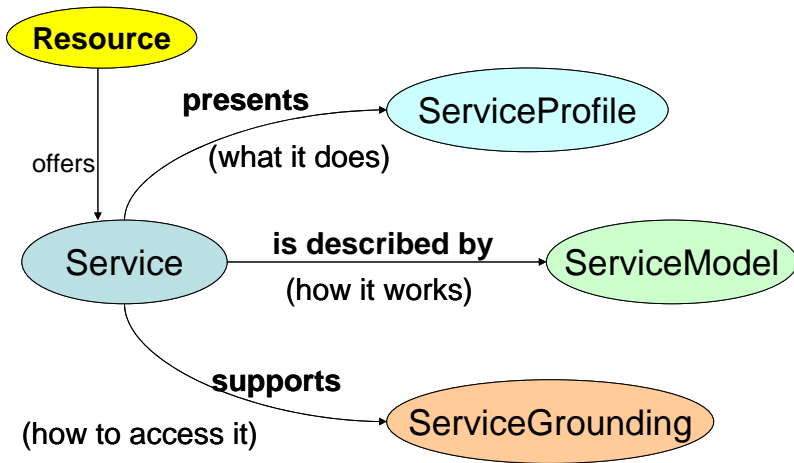
The description of the semantic content of requests and the conditions under which particular outcomes may occur.

Service Grounding

The specification of how to access the service.



OWL-S Service Ontology as RDF Graph



Which language for Artifact?

- To build open MASs where agents dynamically look for the most adequate artifact, artifacts should be expressed in some high-level language.
- Such a language should
 - make it possible for agents to adopt a uniform cognitive level for interaction,
 - be standard so as to support the MAS openness and agent mobility.
- So, heterogeneous agents join a MAS, discover and use in a simple way the services provided by artifacts.



Why OWL-S?

- OWL-S is standard.
- OWL-S allows to express artifacts' features in a natural way:

Function Description \Rightarrow Service Profile

Operating Instructions \Rightarrow Service Model

Usage Interface \Rightarrow Service Grounding

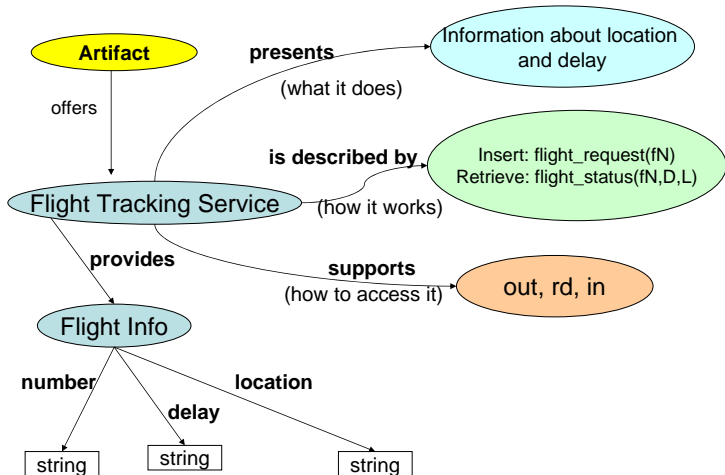


Example: flight tracking service

- Let us suppose that an artifact provides a flight tracking service. The service can be described through:
 - *function description*: information about the delay and the location of the selected flight;
 - *usage interface*: *out*, to insert a request; *rd*, to read the response; *in*, to read and remove /consume the response;
 - *operating instructions*: the user should first insert a request such as `flight_request(fN)`, then retrieve the response such as `flight_status(fN,D,L)`.



Example: flight tracking service



Example: Service Profile (1/2)

```
<profileHierarchy:Flight_Tracking
  rdf:ID="Profile_Flight_Tracking_Service">
  <service:presentedBy rdf:resource=
    "&flightService;#FlightTrackingService"/>
  <profile:has_process
    rdf:resource= "&flightProcess;
    #FlightTrackingProcessModel"/>
  <profile:serviceName>
    Flight_Tracking_Agent
  </profile:serviceName>
```



Example: Service Profile (2/2)

```
<profile:textDescription>
```

```
    This agentified service provides the  
    opportunity to request information  
    about the delay and the location of a  
    flight whose number is given as input.
```

```
</profile:textDescription>
```

```
</profileHierarchy:Flight_Tracking>
```



Example: Service Model (1/2)

```
<process:AtomicProcess rdf:ID="FlightTracking">
  <process:hasInput>
    <process:Input rdf:ID="FlightNumber">
      <process:parameterType rdf:resource="&xsd;#string"/>
    </process:Input>
  </process:hasInput>
  <process:hasPrecondition
    rdf:resource="#FlightNumberExists"/>
```



Example: Service Model (2/2)

```
<process:hasEffect>
  <process:ConditionalEffect
    rdf:ID="InfoRequestEffect">
    <process:ceCondition
      rdf:resource="#FlightNumberExists"/>
    <process:ceEffect rdf:resource="#InfoProvided"/>
  </process:ConditionalEffect>
</process:hasEffect>
</process:AtomicProcess>
```



OWL-S: Limits

- OWL-S describes services provided by an artifact and how it works in a (relatively) simple way.
- However, OWL-S is unable to describe the information related to the physical functionality of the service. So, typically WSDL is used together with OWL-S.
- WSDL is good in the Web Service context, but not-so-adequate for artifacts:
 - data and message type, how the message is going to be transmitted, where the service is located, ...
- On the artifact side, there isn't semantic language similar to WSDL to describe the implementation details of a service.



Conclusions and Future Work

- Exploited OWL-S for describing the services provided by artifacts
 - OWL-S easily expresses Function Description and Operating Instructions,
 - but cannot be directly used to describe Usage Interface
- Future work will be devoted to
 - explore languages for the description of Usage Interface,
 - compare OWL-S and WSMO for artifact description,
 - develop examples to validate this approach.



OWL-S for Describing Artifacts

Rossella Rubino, **Ambra Molesini**, Enrico Denti
`ambra.molesini@unibo.it`

ALMA MATER STUDIORUM—Università di Bologna

Lisbon, 14–15 December 2006

