Classification of SOA Contract Specification Languages

Joseph C. Okika, Anders P. Ravn
{ojc, apr}@cs.aau.dk

Department of Computer Science
Aalborg University, Denmark

Why SOA Contract?

We should integrate our systems using Web Service Technologies …
Standards

- ebXML
- WS-Attachments
- SOAP MTOM
- WS-Reliability
- WS-ReliableMessaging
- WS-Acknowledgment
- WSDL
- ASAP
- SOAP
- SPML
- WSDM
- WS-Addressing
- WS-Referral
- WS-Enumeration
- WS-Transfer
- WS-Referral
- OWL Fat
- OWL Light
- SOAP MTOM
- WS-Notification
- WS-Routing
- WSDL
- WS-CallBack
- WS-MessageData
- WS-NonRepudiation
- WS-Federation
- WS-SecureConversation
- WS-Trust
- WS-Security
- SAML
- OWL-S
- WS-MetadataExchange
- WS-SecurityPolicy
- XACML
- WS-PolicyAssertions
- WS-PolicyAttachment
- WS-PolicyFramework
- WS-Topics
- WS-Eventing
- WS-Notification
- WS-Manageability
- WS-Resource
- WS-Coordination
- BPEL4WS
- BTP
- WSCI
- WSDL
- WS-BusinessActivity
- WS-Transaction
- WS-AtomicTransactions
- Cool! but, what Standards should I use?

Ref: Benatallah "Interoperability Specifications", May 06, IEEE Computer, by Motahari, Benatallah, Casati and Toumani
Surveys

- [Lippe et al 2005]
  - Cross-Organisational Business processes
  - concepts
  - coverage of development process

- [Beek et al 2006]
  - semantics of languages
  - mappings between languages
  - applicability
Classification – Compatible notations

BPEL4WS  WSDL
WS-Coordination  WSCDL
WS-Addressing  WS-SLA
WS-Routing  WS-Referral
WS-Trust  WS-Security
WS-PolicyFramework  WS-Reliability
WS-Transaction  WS-Notification
WS-MessageData  WS-Eventing

SOAP MTOM
ASAP
SOAP

ebXML  ebCPA
ebCPP  ebBPSS

Cool! but, what Standards are compatible?

OWL Fat  OWL Light

OWL-S  WMSO
What are SOA Contracts?

Defining contract:

- A contract is about certain aspects of a service:
  - interfaces.
  - behaviour
  - functionality
  - quality
- A contract language expresses properties of some aspects
- Properties define proper service collaborations and/or interactions
Contract Aspects

- **Interface:**
  - defines syntax of communication

- **Functionality:**
  - operations and their properties

- **Protocol:**
  - behaviour: sequencing of events, signals, messages, etc.

- **Security:**
  - techniques and practices that ensure confidentiality

- **Extra functional properties:**
  - constraints on a concrete service property: performance, availability, reliability, accessibility, etc.
Tools for Analysis I

- based on:
  - Automata
  - Petri nets
  - Abstract State Machines
  - Process Algebra
  - etc.
Tools for Analysis II

- **Interface:**
  - compilers

- **Functionality:**
  - test tools

- **Protocol:**
  - monitoring
  - model checking

- **Security:**
  - monitoring, model checking

- **Extra functional properties:**
  - simulation, monitoring

---

[Beek et al 2006]
## Classification - Aspect and Family

<table>
<thead>
<tr>
<th>Aspects</th>
<th>Contracts Languages/Approaches</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Web Services (WS-*)</td>
</tr>
<tr>
<td>Interface</td>
<td>WSDL</td>
</tr>
<tr>
<td>Functionality</td>
<td>WS-BPEL, WSOL</td>
</tr>
<tr>
<td>Protocol</td>
<td>WS-BPEL, WS-CDL</td>
</tr>
<tr>
<td>Security</td>
<td>WS-Security</td>
</tr>
<tr>
<td>Quality</td>
<td>policy</td>
</tr>
<tr>
<td></td>
<td>trust</td>
</tr>
<tr>
<td></td>
<td>availability</td>
</tr>
<tr>
<td></td>
<td>performance</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Using the Classification

<table>
<thead>
<tr>
<th>Aspects</th>
<th>WS - *</th>
<th>Tool Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface</td>
<td>WSDL</td>
<td>Compilers, XML-parsers</td>
</tr>
<tr>
<td>Functionality</td>
<td>WS-BPEL</td>
<td>Model checking ? Proof techniques (cf. JML)</td>
</tr>
<tr>
<td>Protocol</td>
<td>WS-BPEL, WS-CDL</td>
<td>Model checking</td>
</tr>
<tr>
<td>Security</td>
<td>WS-Security</td>
<td>Model checking</td>
</tr>
<tr>
<td>Quality</td>
<td>WSLA</td>
<td>Simulation</td>
</tr>
</tbody>
</table>
Example I: WSDL

- Describes the public interface to a particular web service
  - Data Types <types>
  - Messages <message>
  - Operations <portType>
  - Communication Protocols <binding>
- Serves as a (syntactic) contract between service providers and service consumers
- WSDL tells a client what functions are available

```xml
<?xml version="1.0" encoding="utf-8" ?>
<definitions xmlns:xs="http://www.w3.org/2001/XMLSchema"
 xmlns:wsdl="http://schemas.xmlsoap.org/soap/soap/">
  <portType name="EmployeeTravelStatusPT">
    <operation name="EmployeeTravelStatus">
      <input message="tns:EmployeeTravelStatusRequestMessage"/>
      <output message="tns:EmployeeTravelStatusResponseMessage" />
    </operation>
  </portType>

  <message name="EmployeeTravelStatusRequestMessage">
    <part name="employee" type="tns:EmployeeType"/>
  </message>

  <message name="EmployeeTravelStatusResponseMessage">
    <part name="travelClass" type="tns:TravelClassType"/>
  </message>
</definitions>
```
Example II - BPEL

- Specifies behavioural aspects of service contract
- Uses partnerlinks and activities to model the service interaction

```xml
<?xml version="1.0" encoding="utf-8" ?>
<process name="Travel" xmlns="http://schemas.xmlsoap.org/ws/2003/03/"
business-process=""
...
<partnerLinks>
  <partnerLink name="client"
    partnerLinkType="trv:travelLT"
    myRole="travelService">... ...
  <partnerLink name="employeeTravelStatus"
    partnerLinkType="emp:employeeLT"
    partnerRole="employeeTravelStatusService"/>
...
```

```xml
...<partnerLink name="AmericanAirlines"
  partnerLinkType="aln:flightLT"
  myRole="airlineCustomer"
  partnerRole="airlineService"/>
  <partnerLink name="DeltaAirlines"
    partnerLinkType="aln:flightLT"
    myRole="airlineCustomer"
    partnerRole="airlineService"/>
</partnerLinks>
<!-- Variables are declared here-->
<sequence>
  <receive partnerLink="client"
    portType="trv:TravelApprovalPT"
    operation="TravelApproval"
    variable="TravelRequest"
    createInstance="yes"/>
```
Why do we do this?
A Tool Framework for BPEL Analysis

[Ongoing work with Cambronero]
A Tool Framework for Behavioural Aspect

[Cambronero 2007]

Are the translation function reasonable?
What about the semantics?
A Tool Framework for Behavioural Aspect
A Tool Framework for BPEL Analysis
The Difficult issues in Semantics

- BPEL compensation, faults and flow
- UppAal model of a compensation manager.
- Completion condition of concurrent activities
- Data dependent behaviour
What Properties should a service have?

- No deadlock
- Completion - reaching an end point
- Inter-operability/Consistency with choreography
- ?
Conclusion

WS-* family covers the major aspects of SOA contract

The classification gives a blue print for:

- Web Services specification language families
- integrated analysis tools
* Eventhandlers * Activity

and Onalarm = Onalarm of For_Or_Until * AScope

**datatype** Extension2 = Extension of Import * Partnerlinks * Messageexchanges * Variabl
* Correlationsets * Faulthandlers * Eventhandlers * Activity
* Extensionattr

**type** Extension = string

**datatype** BPEL_Process = bpelprocess of BPA * Extension * Import list * Partnerlinks
* Messageexchanges * Variables * Correlationsets
* Faulthandlers list * Eventhandlers list
* Activity
con Activity1 = Activity1 : Onmessage
con Onalarm_P = fn : (Expr_Attrs * Expr_Attrs * Expr_Attrs) * Activity1 -> Onalarm_P
con Faulthandlers = fn : CatchAttrs * Catchall -> Faulthandlers
con Eventhandlers = fn : Onevent * Onalarm -> Eventhandlers
con AScope = fn :
Scope_Attrs * PLA list * Messageexchange_Attrs list * V_Attrs list *
CRS_Attrs list * Faulthandlers * Compensationhandler * Terminationhandler *
Eventhandlers * Activity1 -> AScope
con Onalarm = fn : (Expr_Attrs * Expr_Attrs * Expr_Attrs) * AScope -> Onalarm
con Extension = fn :
Import_Attrs * PLA list * Messageexchange_Attrs list * V_Attrs list *
CRS_Attrs list * Faulthandlers * Eventhandlers * Activity * Extensionattr ->
Extension2
con bpelprocess = fn :
{name : string, parallel : bool, supressJoinFailure : bool,
targetNamespace : string, xmlns : string, xmlns : string} * string *
Import_Attrs list * PLA list * Messageexchange_Attrs list * V_Attrs list *
CRS_Attrs list * Faulthandlers list * Eventhandlers list * Activity1 ->
BPEL_Process