Model Interchange and Integration for Web Services

Robert J. Hall
bob-2modelxchg-@channels.research.att.com

AT&T Labs Research
Florham Park, NJ - USA

Andrea Zisman
a.zisman@soi.city.ac.uk

City University
London - UK

Outline

- Motivation
- OpenModel Modeling Language (OMML)
- OpenModel Validation Approach
- BookFind Example
  - Model translations
  - Model integration
  - Validation
- Demonstration
- Conclusion and Future Work

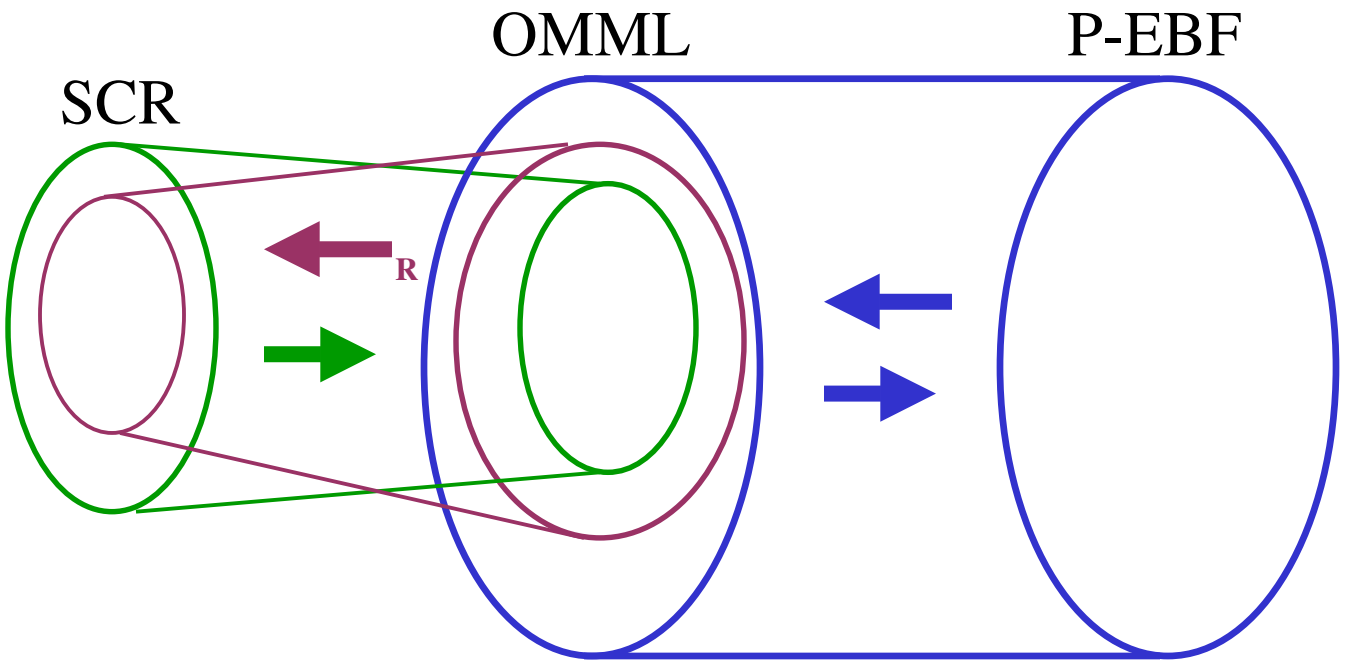
Motivation

- Web services are multi-stakeholder distributed systems (MSDS)
- Unrealistic to assume all stakeholders use the same behavioral specification (model) formalism
- This heterogeneity makes it difficult to share and integrate models to validate distributed systems
- Existing interface-based web service standards cannot support model-based validation

Therefore, model-based validation of web services fails for realistic web services

The Solution: OMML

- OpenModel Modeling Language (OMML)
- XML-based model interchange format
- Translation tools to/from other modeling languages



OMML General Design Goals

- *Function rich reactive system* language (functionality and control expressed procedurally in terms of function/ object theories, not limited to finite state models)
- *Executable specifications* of single nodes & compositions
  - abstract state model with computable abstraction map
  - explicit connector declaration
  - instance data declaration tolerating missing data
- Support for
  - *systematic validation methods* (symbolic execution and term simplification, specification coverage measurement, theorem proving, model checking)
  - *automated reasoning over* arbitrary *function rich theories* within shared domain-specific ontologies

Design of OMML

- Based on ISAT's Procedural Event-Based Formalism (P-EBF)
- Five different types of documents
  - **Theory**: declaration of domain-specific *types*, *functions* (signatures and body), and *reasoning axioms* governing their semantics
  - **Ontology**: group of related theories
  - **Specification**: synchronous executable behavior model
  - **Connection**: description of how outputs of one node relate to inputs of another node
  - **Model Instance**: instance information for one MSDS node (abstract state data, connections, specification)

Examples - Theory

```
<Theory TheoryName="LISTS" … >
 <IncludeDeclaration>
  <IncludeTheoryFrom
      OntologyName="OntoBasis"
      OntologyLocation="www.att.com">
   <IncludeTheory TheoryName="LOGICALS"/>
            …     </IncludeDeclaration>
 <TypeDeclaration>
   <Type TypeName="LIST">
    <SubTypeOf OntologyName="OntoBasis"
                  TheoryName="BASE"
                  SubTypeOfName="THING"/>
    <FunctionDefault
                FunctionName="EMPTY-LIST"/>
   </Type> </TypeDeclaration>
 <FunctionDeclaration>
  <Function FunctionName="EMPTY-LIST">
   <FunctionReturnValue
       FunctionReturnValueType="LIST"/> ...
 <Function FunctionName="MEMBER">
  <FunctionParameter OntologyName="OntoBasis"
          TheoryName="BASE"
          FunctionParameterType="THING"/>
  <FunctionParameter
          FunctionParameterType="LIST"/>
  <FunctionReturnValue OntologyName="OntoBasis"
           TheoryName="LOGICALS"
           FunctionReturnValueType="BOOLEAN"/>
  <FunctionBody LanguageType="LISP">
    <![CDATA[ DEFPURE LISTS.MEMBER (X Y )
     (DOLIST (YY Y (LOGICALS.FALSE)) (WHEN
     (IS-LOGICAL-TRUE ( LOGICALS.EQUAL X YY))
     (RETURN (LOGICALS.TRUE)))))]]>
  </FunctionBody> … </FunctionDeclaration>
<IteratorDeclaration> … </IteratorDeclaration>
<RewriteRuleDeclaration>
  <RewriteRule RewriteRuleType="DEFSIMPLIFIER"
          RewriteRuleName="MEMBER-CONS">
     … </RewriteRuleDeclaration>
</Theory>
```

Examples - Specification

```
<Model ModelName="WEBDELIVERY">
<IncludeDeclaration> . . .
 <IncludeTheory TheoryName="BOOK"/> . . .
<InputEventDeclaration>
<InputEvent InputEventName="DEL-BOOK">
<EventParameter
      EventParameterType="LIST"/> . . .
<StateRelationDeclaration>
 <StateRelation
    StateRelationName="BOOK-URL">
 <StateParameter
   StateParameterType="STRING"/>
 <StateValueParameter
  StateValueParameterType="STRING"/> . . .
<OutputEventDeclaration>
 <OutputEvent OutputEventName="MAIL">
  <EventParameter
      EventParameterType="MESSAGE"/>

   . . .
```

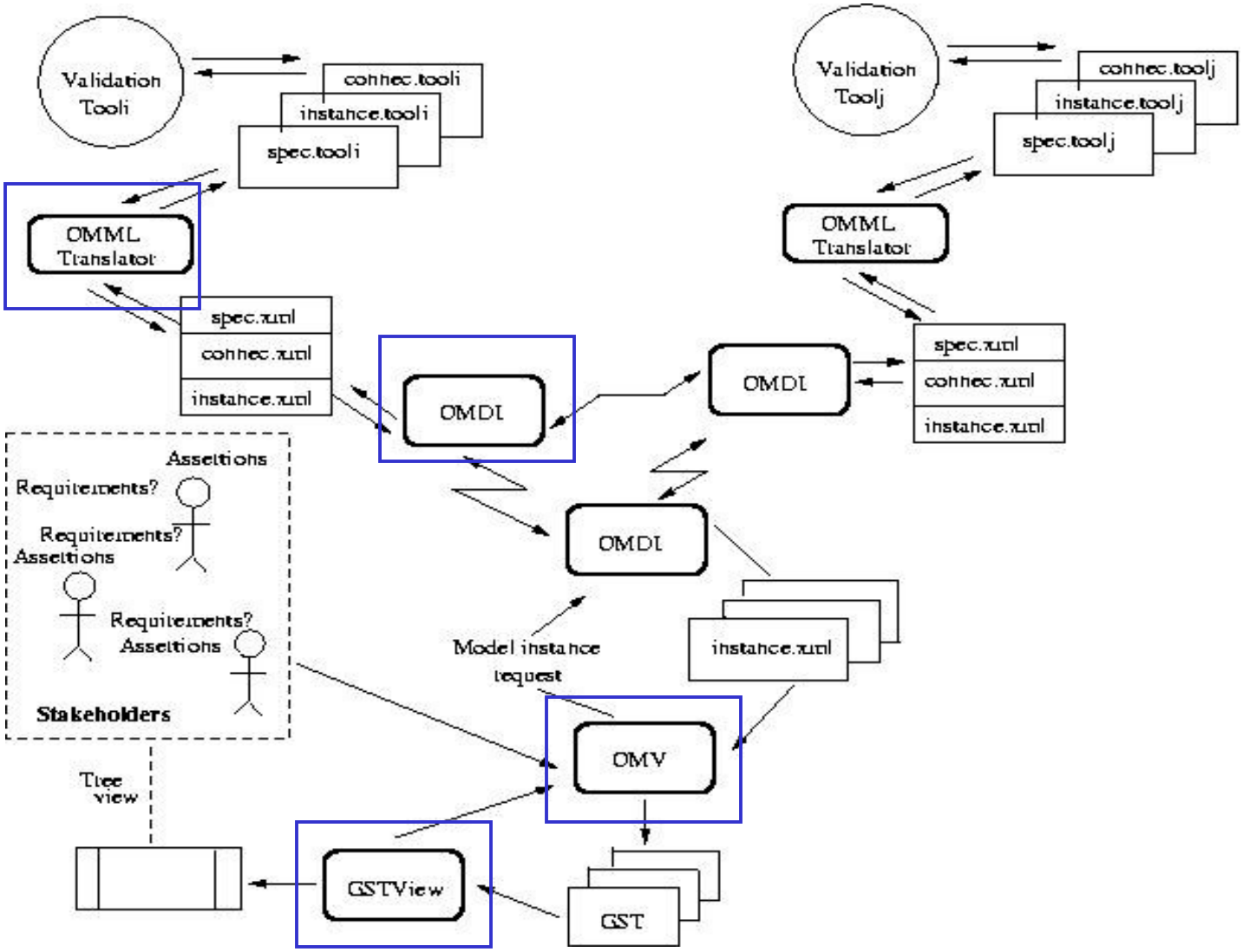
```
<InputEventStatement> <InputEventRef
    InputEventRefName="DEL-BOOK">
<Variable VariableName="BOOKINFO"/> . . .
<StatementList>
  <LETSTAR> <Bind>
   <Variable VariableName="TITLE"/> . . .
    <FunctionRef FunctionRefName="A-STRING">. . .
     <FunctionRef FunctionRefName="NTH"> . . .
      <Constant ConstantValue="3"/> . . .
       <Variable VariableName="BOOKINFO" /> . . .
   <IF><FunctionRef FunctionRefName="MEMBER">
    <Variable VariableName="TITLE"/> . . .
    <LOOKUP><StateRelationLookRef
     StateRelationLookRefName="VALID-TITLES"/>...
    <OutputEventRef OutputEventRefName="MAIL">
     <FunctionRef FunctionRefName="SMESSAGE">
      <Constant ConstantValue="WEBDELS.COM"/>...
      <Variable VariableName="EMAIL"/> . . .
      <StateRelationLookRef
        StateRelationLookRefName="BOOK-URL"> . . .
```

OMMLTranslator

- OMML Translator (OMT) maps models in different languages to/from OMML
  - OMML to P-EBF
  - P-EBF to OMML
  - SCR to OMML

  } total translation
  - OMML to SCR: partial translation (infinite state spaces and function rich theories)
- Status
  - OMT is currently implemented partially in Java and partially in LISP (ISAT)
  - Goal: standalone Java-based tool

OpenModel Overview



OpenModel Project

- OpenModel Modeling Language - ***OMML***
  - each node of an MSDS publishes behavioural model of itself based on shared domain specific ontologies at a certain level of abstraction
- OpenModel Validator - ***OMV***
  - incremental generation of behavior trees (GST)
  - dynamic deduction of MSDS nodes relevant to the task/requirements of interest
  - tolerance of partial information by using symbolic simulation
- OpenModel Distribution Infrastructure - ***OMDI***
  - protocol/API for model retrieval (planned)

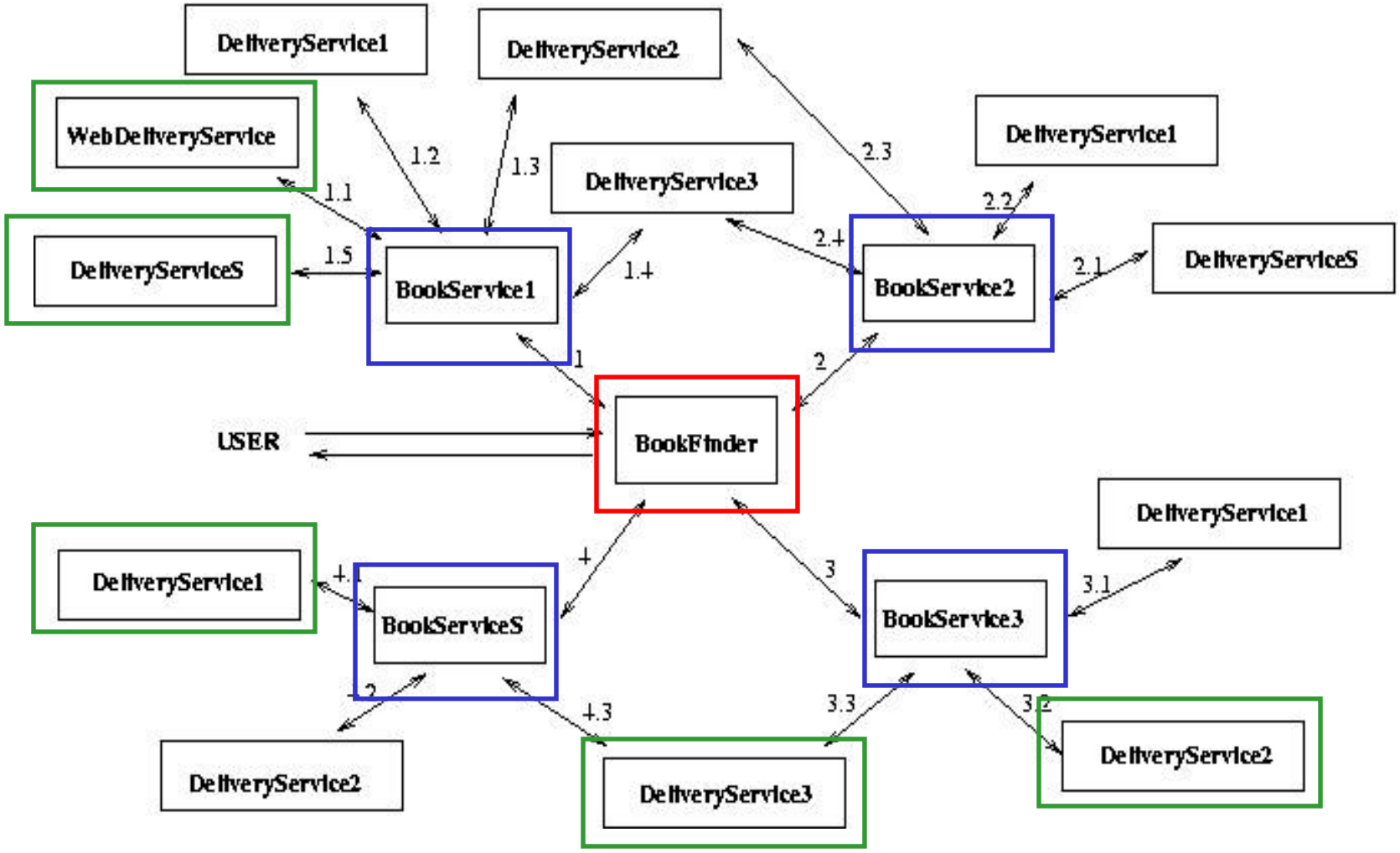
OpenModel Validation (GSTView)

Graphical interactive tool built on top of OMV to support validation of personal requirements via *assisted symbolic behavior browsing* methodology

- *visualisation* and *inspection* of symbolic behavior
  - graphical representation of message passing
  - detailed examination of events, state changes
- automated reasoning support for
  - *exact* property checkers
  - *approximate* property checkers
  - *noticers* of situations

} *ReqMons*

BookFind Example



BookFind Example

	BS1	BS2	BS3	BS-S	DS1	DS2	DS3	DS-S
Web delivery	yes	no	no	no	-	-	-	-
Spam messages (all)	yes	no	no	no	-	-	-	-
Spam messages (related)	-	no	yes	no	-	-	-	-
Hold personal info	yes	no	yes	no	-	-	-	-
Hold cc info	yes	yes	yes	no	-	-	-	-
Overnight delivery	-	-	-	-	yes	no	yes (EU)	no
P.O. Box address	-	-	-	-	no	yes	no	yes
Signature required	-	-	-	-	yes	no	no	no

Case Study Steps

- BookFind servers modeled as OMML, P-EBF, and SCR
- SCR and OMML models were translated into P-EBF; P-EBF models were translated into OMML and re-translated into P-EBF
- Eight scenarios were simulated, visualised and validated using GSTView

Demo will be presented soon

OMMLTranslation - Results

Documents	P-EBF size	SCR size	OMML size	Into OMML time	OMML-PEBF time
BookFinder.spec	9.11	-	38.6	3.7	<1.0
BookService1.spec	8.81	-	39.9	3.3	<1.0
BookServiceS.scr	27.4	11.0	121	30.4	<1.0
WebDelService.spec	1.64	-	8.62	<1.0	<1.0
DeliveryService1.spec	2.22	-	10.2	1.0	<1.0
DeliveryServiceS.scr	4.57	1.55	23.0	2.07	<1.0
Lists.theory	5.24	-	17.9	2.0	<1.0
DeliveryConfirm.conn	0.44	-	0.91	<1.0	<1.0

OMML = 4.5*P-EBF

GSTView Validation - Results 1/2

Requirements

R1: service node will send spam email to customers ?

R2: unacceptably high delivery cost ?

R3: vendor retains customer information after transaction ●

R4: signature is required on delivery --

R5: undesirable web delivery (hard-copy only) --

R6: urgent delivery requested to p.o.box address ●*

Input Constraints

I1: USA address, or outside

I2: p.o.box address, or not

I3: urgent delivery, or not

* approximate

GSTView Validation - Results 2/2

Requirements

R1: se

R2: un

R3: ve

R4: sig

R5: un

R6: ur

Scenario	Nodes	Violations	Time
I1,!I2,!I3	65	R1,R3-R5	0.7
I1, I2,!I3	63	R1,R3,R5	0.7
!I1,!I2,!I3	65	R1-R5	2.2
!I1, I2,!I3	63	R1,R3,R5	2.1
I1,!I2, I3	76	R1,R3-R5	2.2
I1, I2, I3	94	R1,R3,R5,R6	3.9
!I1,!I2, I3	76	R1-R5	2.7
!I1, I2, I3	94	R1,R3,R5,R6	3.6
Indefinite	144	R1-R6	5.6

?

?

ction ●

--

--

●*

I1: USA address, or outside

I2: p.o.box address, or not

I3: urgent delivery, or not

* approximate

Conclusion and Future Work

- OMML supports integration of heterogeneous models, allowing validation of realistic web services
- Future Work:
  - extending OMT to allow translations between OMML and other languages (e.g. Statecharts, Action Language)
  - investigating size reduction of OMML documents
  - implementing OMDI using UDDI

www.research.att.com/~hall/openmodel-project.html