Model-driven development of a mediation service

Dick Quartel
Stanislav Pokraev
Telematica Instituut
The Netherlands

Rodrigo Mantovaneli Pessoa
Marten van Sinderen
University of Twente
The Netherlands

EDOC Conference - September 18, 2008
Agenda

1. Service mediation
2. Example scenario
3. Method
4. Demo
5. Concluding remarks
Service mediation

• How to achieve cooperation between non-interoperable systems?
  – Systems defined in terms of services
  – Systems can not be changed

Mismatches
• Data: different information models (vocabularies)
• Process: different (orderings of) messages
Service mediation

• Definition
  “to act as an intermediary in reconciling differences between the services of two or more systems”

![Diagram showing Service mediation concept]

- System A: S1, S2
- Mediator M
- System B: S3, S4, S5

requested service

provided service
Example scenario

POR = Purchase Order Request
POC = Purchase Order Confirmation

http://sws-challenge.org
Method

- Model-driven approach
- Existing services described at technology level
- Solution designed at abstract level

1. Abstract service PSMs to service PIMs
2. Semantic enrichment of the service PIMs
3. Design of the mediator PIM
4. Validation of the mediator PIM
5. Transformation to mediator PSM
Step 1: Abstract from PSMs to PIMs

- ‘Lift’ problem to design/model level
  - Abstract from technology aspects
  - Capture semantics of the integration problem
- Involvement of business domain experts
Step 1: Abstract from PSMs to PIMs

- **ISDL**
- **Grizzle**
- **WSDL**

(i) **operation call**
- User
- Provider
- Operation execution

(ii) **interpretation**
- Invoking
- Accepting
- Returning
- Catching
- Failing
Step 1: Abstract from PSMs to PIMs

- **ISDL**
- **Grizzle**
- **WSDL**

**Stereotype information**

```
invoke:
Pip34PurchaeOrderRequestType req
Return:
ReceiptAcknowledgmentType result
```

```
accept:
OrderType order
Reply:
long result
```

```
createNewOrder
```

```
addLineItem
```

```
CloseOrderType closeOrder
Invoke:
OrderLineItemConfirmationType req
Return:
String result
```

```
receiveConfirmation
```

```
poolType OMSrvcePortType;
namespaceURL nullcompany,
webapp on
```

```
```

```
```
Step 2: Semantic enrichment of PIMs

- Add semantics that could not be derived automatically
  - WSDL only defines syntax of messages/operations
- Information modelling
  - Use of ontology languages (e.g., OWL) and domain-specific ontologies (e.g., UDEF)
- Behaviour modelling
  - Use of formal process modelling languages (e.g., ISDL, Petri Nets)
Step 2: Semantic enrichment of PIMs
Step 3: Design of the mediator PIM

a) Derive the services/operations provided by and requested from the mediator
Step 3: Design of the mediator PIM

b) Design the mediation behaviour by relating the services/operations
   • ‘matching’ input and output parameters
Step 3: Design of the mediator PIM

c) Design the data mappings among the operation parameters
transformation blue2moon {
  namespaces {
    ..
  }
}

mapping por2search {
  target moon:SearchCustomerType SearchCustomer {
    contactName="searchString ";
  }
  source por:Pip3A4PurchaseOrderRequestType Pip3A4PurchaseOrderRequest {
    contactName= "fromRole/PartnerRoleDescription/..../businessName/FreeFormText";
  }
}

...
transformation blue2moon {
    namespaces {
        ..
    }
}

mapping productLineItems2poc {
    target poc:Pip3A4PurchaseOrderConfirmationType Pip3A4PurchaseOrderConfirmation {
        fromRole = "fromRole/PartnerRoleDescription";
        toRole = "toRole/PartnerRoleDescription";
        isDropShip = "PurchaseOrder/isDropShip";
        requestingDocumentDateTime = "requestingDocumentDateTime/DateTimeStamp";
        requestingDocumentIdentifier = "requestingDocumentIdentifier/ProprietaryDocumentIdentifier";
        thisDocumentGenerationDateTime = "thisDocumentGenerationDateTime/DateTimeStamp";
        thisDocumentIdentifier = "thisDocumentIdentifier/ProprietaryDocumentIdentifier";
        confs = "PurchaseOrder/ProductLineItem";
    }
}

source med:List confs {}

source por:Pip3A4PurchaseOrderRequestType por {
    fromRole = "toRole/PartnerRoleDescription";
    toRole = "fromRole/PartnerRoleDescription";
    isDropShip = "PurchaseOrder/isDropShip";
    requestingDocumentDateTime = "thisDocumentGenerationDateTime/DateTimeStamp";
    requestingDocumentIdentifier = "thisDocumentIdentifier/ProprietaryDocumentIdentifier";
}

expressions {
    thisDocumentGenerationDateTime = currentDateTimeStamp();
    thisDocumentIdentifier = generateDocumentId();
}
Step 4: Validation of the mediator PIM

- Interoperability analysis  [EDOC 2007]
- ‘Live’ simulation $\rightarrow$ demo
Step 5: Derivation of the mediator PSM

- Transformation to
  - Control flow manager – coordinates message exchange
  - Data manager – maintains state and performs mappings
Concluding remarks

- Method to guide development of a service mediator
- Techniques and tool support for each step
  - modelling, simulation, transformation, ...
- ... but, composition of the mediation solution is still mainly a manual process
Ongoing work

- Support automated reasoning
  - ontology language (OWL) for information modelling
  - integration in method and tool support
Ongoing work

• Semi-automated composition
  – matching input and output parameters
  – derivation of relations between operations
  – derivation of data mappings + GUI-based tool
  – integration of ISDL and OWL/SPARQL
The end

Questions?