Model-Driven Software Engineering

Model-Driven Development of SOA Applications II

Dr. Jochen Küster  (jku@zurich.ibm.com)
Agenda

- Process and Techniques for SOA Development
- From Service Realization to Service Implementation
- Summary and References
Process and Techniques for SOA Development
Overview of Methods for Developing SOA Applications

- Different methods exist for building SOA applications

- Service-oriented Modeling and Architecture (SOMA)
  - Method for developing service-oriented solutions
  - Developed by IBM
  - Can be used together with SoaML and IBM tooling

- Quasar Enterprise
  - Method for service-oriented engineering of application landscapes
  - Developed by cap gemini sd&m

- SOA Methodology
  - A set of principles for designing service-oriented solutions
  - Based on a book by T. Erl
  - Sometimes referred to as “Mainstream SOA Methodology”
Main Phases of SOMA

- **Identification phase**
  - Aims at identifying services
  - Various techniques can be used

- **Specification phase**
  - Aims at specifying services and components
  - Result is a service-oriented architecture

- **Realization phase**
  - Refine components

- **Service Implementation**
  - Construct, generate and assemble services
Activities for SOA Development (SOMA)

[Source:SOMA]
## Roles and Tools used in SOA Application Development

<table>
<thead>
<tr>
<th>Role</th>
<th>Task</th>
<th>Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business Executive</td>
<td>Convey business goals</td>
<td>IBM Rational Requirements Composer</td>
</tr>
<tr>
<td>Business Analyst</td>
<td>Analyze business requirements</td>
<td>IBM Rational Requirements Composer</td>
</tr>
<tr>
<td>Software Architect</td>
<td>Design the architecture of the solution</td>
<td>IBM Rational Software Architect</td>
</tr>
<tr>
<td>Solution Architect</td>
<td>Implement the solution</td>
<td>IBM Rational Application Developer</td>
</tr>
</tbody>
</table>
Identification of Services

- **Input of activity:**
  - Business requirements, business processes, business goals

- **Output of activity:**
  - Services architecture (SoaML model)

- **Roles:** Business Analyst, Software Architect

**Techniques:**

- **Goal-service modeling**
  - Identifies capabilities needed to realize business requirements such as goals and strategies

- **Domain decomposition**
  - Uses activities of business processes and other descriptions to identify needed capabilities

- **Existing asset analysis**
  - Mines capabilities from existing applications
Goal-Service Modeling

- Business goals are decomposed into subgoals until actionable goals are obtained
- Key Performance Indicators (KPIs) are identified
- Metrics are identified for measuring KPIs
- Candidate services are identified for reaching business goals
- Candidate services are then captured as Capabilities in SoaML
Goal-Service Modeling Example

<table>
<thead>
<tr>
<th>Goal and Subgoals</th>
<th>KPIs</th>
<th>Metrics</th>
<th>Services</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Attract and retain customers</td>
<td>Increase number of customers using banking products and services by 2% of the total customer base in the insurance and investment lines of business</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.1 Enable banking products and services through channels such as self-service portals and interactive voice response systems</td>
<td>Increase number of customers using banking products and services</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.1.1 Enable banking products such as checking, savings, money market accounts, CDs online</td>
<td>Increase number of customers using banking products</td>
<td>Number of accounts opened in banking products such as checking, savings, money market accounts, CDs</td>
<td>• Open account</td>
</tr>
<tr>
<td>1.1.1.1 Enable banking services such as bill payment, fund transfers, check status, and check reorders via self-service portal or IVR (integrated voice response)</td>
<td>Increase number of customers using banking services</td>
<td>Number of banking services through various channels</td>
<td>• Close account</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Get banking customer report</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Make payment</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Transfer funds</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Get account summary</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Get account activities</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Order checks</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Get check status</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Get statement</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Get payment history</td>
</tr>
</tbody>
</table>

[Source:SOMA]
Capabilities and Candidate Services

- Candidate Services can be captured as capabilities in SoaML
Specification of Services

- Input of activity:
  - Capabilities

- Output of activity:
  - Service Specification, Service Data Model

- Roles: Business Analyst, Software Architect

Key Activities:

- Specify Service Interfaces including operations and roles
Elements of a Service Specification

- Name of the service
- Provided and required interface, describing functional capabilities
  - Its name
  - Required or optional service data inputs or outputs
  - Preconditions, post-conditions, exceptions, fault conditions
- Any communication protocol or rules
- Capabilities that consumers have to provide to be able to use the service
- Requirements for service implementers
- Constraints
- Qualities of service that consumers should expect and that providers have to provide (cost, availability, performance, …)
- Policies such as security or transaction scopes
Service Specification Example

Operation or event reception

Service Interface (SoaML)

Roles (SoaML)

Communication protocol (SoaML)

- Note: The ShippingService realizes the Shipping interface and uses the required interface ScheduleProcessing
Service Specification Example
Service data is data exchanged between service consumers and providers

Note: This is not to be confused with a WSDL messages
Realization of Services

- **Input of activity:**
  - Service Specification, Service Data Model

- **Output of activity:**
  - Service Design Model

- **Roles:** Software Architect

**Key Activities:**

- **Service provisioning**
  - Decide which service providers provide which services
  - Design the service implementations

- **Service composition**
  - Assemble and connect service consumers and providers
Service Realization Example: Service Provisioning

- Participant Invoicer provides the InvoicingService service
- Participant Invoicer provides the Invoicing interface
- Participant Invoicer requires the InvoiceProcessing interface
- Participant Invoicer implements the two operations completePriceCalculation and initiatePriceCalculation
- The completePriceCalculation has to be consistent with the protocol
- The initiatePriceCalculation operation is implemented using Java
The OrderProcessor Participant provides the Purchasing Service and orchestrates the InvoicingService, SchedulingService and ShippingService
Service Composition

- The participant Manufacturer provides the complete service design model
From Service Realization to Implementation
Implementation of Services

- **Input of activity:**
  - Service Design Model
  - Service Data Model

- **Output of activity:**
  - Implementation of Service Design Model for a selected platform

- **Roles:** Software Architect

**Key Activities:**

- Transform Service Design Model using predefined transformations
- Add implementation relevant details manually
Models used for Service Implementation

- Interfaces
- Participants providing services
- Participants assembling services according to a ServicesArchitecture
Target Architecture: IBM SOA Programming Model

- Business Objects in XSD
- WSDL Interfaces
- Modules
- Module Assemblies in SCA
- Processes in BPEL4WS
- Java components
The UML to SOA transformation generates a WebSphere Integration Developer project from the SoaML models
- Library projects contain business objects, interfaces, module exports shared by other projects
- Module projects contain a module implementation for each participant in the UML services model
Generation using Rational Software Architect

- For each class and data type an XSD element is created
- For each UML interface a WSDL portType is created
- For each UML activity a BPEL process is created
- Each service provider component is transformed into a module
- Connections between participants in UML are implemented as bindings between module imports and exports in SCA
Module Assembly Example

- Compare to the OrderProcessor participant
Implementation of the Architecture

After generation, the implementation must be completed manually:

- Opaque behaviors must be implemented by adding Java code.
- Correlation for business processes needs to be specified. Correlation specifies information needed to identify instances of a component.
- User Interfaces for human tasks must be created.
Setting SoaML into MDA Perspective

- PIM in SoaML
- Service Data Model
- Service Design Model
- Automatic generation
- Business Objects in XSD
- SCA Modules
- BPEL Processes
- SCA Module Assemblies
- SCA Components
- Manual completion
- UI for Human tasks
- Java for Components
Summary of Lecture

- Introduction to SOA and SOA key concepts
- SoaML profile for modeling SOA applications
- Process and techniques for Model-Driven Development of SOA applications
- Generation of SOA applications from SoaML model

References:

Further References

- M. Huhns et al. Service-oriented computing: key concepts and principles, IEEE Internet Computing, Volume 9, Issue 1, Jan-Feb 2005, Page(s):75 - 81