An MDA-Based Approach for Behaviour Modelling of Context-Aware Mobile Applications

Laura Maria Daniele

TTT 11 February 2009

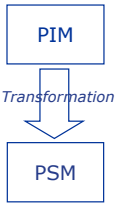
University of Twente
The Netherlands

Outline

- MDA
- Behaviour modelling in MDA
- Our approach for behaviour modelling
- Case study
- Conclusions
- Future Work

Model-Driven Architecture (MDA)

- MDA aims at facilitating (distributed) systems design through the separation of platform-independent and platform-specific concerns
- Common pattern for MDA:
  - Define a platform-independent model (PIM)
  - Apply transformations to this model (model transformation)
  - Obtain one or more platform-specific models (PSMs)



Behaviour modelling in MDA(1)

- The MDA community agrees on the need to consider behavioural aspects in model transformations
  - No agreement on how this should be done

Problem

- Much attention to structural aspects in PSMs and generating code
- Less attention to the PIM level and behaviour of the modelled applications

Behaviour modelling in MDA(2)

Consequence

- Application behaviour is not (well) defined at the PIM level
- Behavioural aspects have to be incorporated later in the development process
  - Hand-written code to PSMs or to implementation code skeletons

Behaviour modelling in MDA(3)

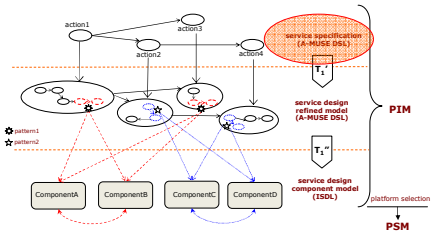
Our Solution

An MDA-based approach for behaviour modelling of context-aware mobile applications

- This approach integrates behavioural aspects of the modelled applications at the PIM level
  - The PIM level is decomposed in different models
  - Each model is a refinement of the previous one

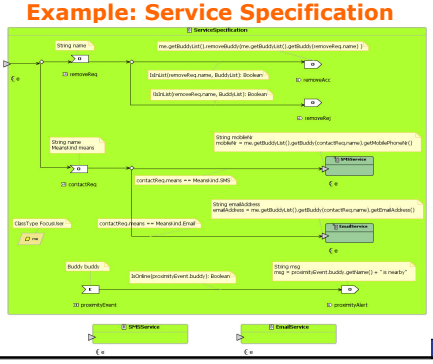
University of Twente
The Netherlands

Approach



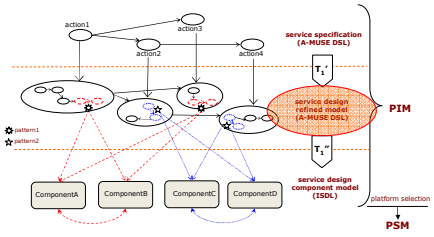
University of Twente
The Netherlands

Example: Service Specification



University of Twente
The Netherlands

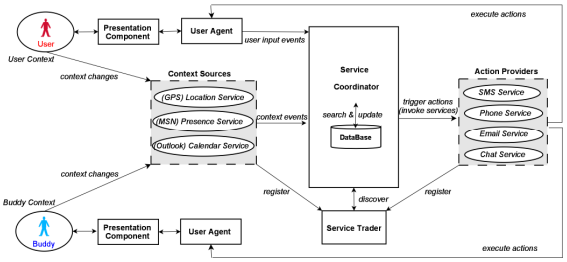
Approach



University of Twente
The Netherlands

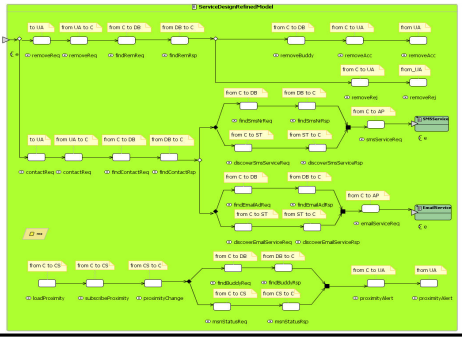
Example: Service Design Refined Model(1)

Live Contacts Architecture



University of Twente
The Netherlands

Example: Service Design Refined Model(2)



University of Twente
The Netherlands

Example: Service Design Refined Model(3)

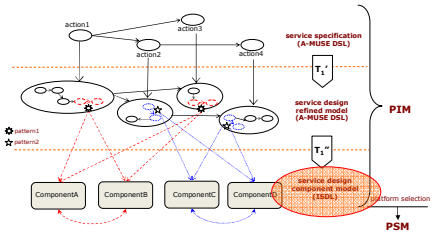
Interaction patterns

'Recurring sequence of actions performed by two or more interacting components'

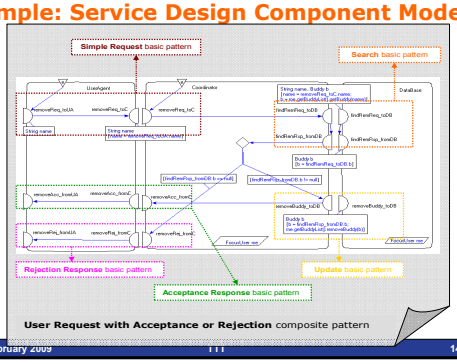
Basic interaction patterns occur between two interacting components

Composite interaction patterns occur between more than two components (combinations of basic patterns)

Approach



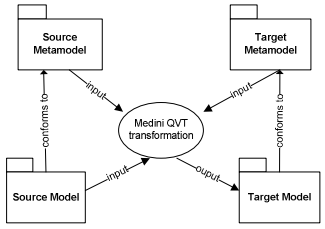
Example: Service Design Component Model

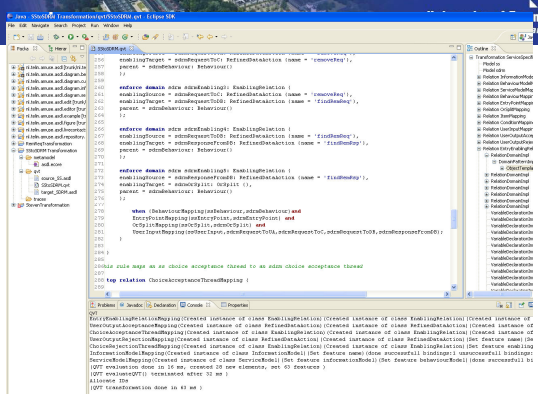


Transformation T_1' with Medini QVT

Source metamodel:
A-MUSE DSL metamodel
Target metamodel:
A-MUSE DSL metamodel

Source model:
A-MUSE DSL
service specification
Target model:
A-MUSE DSL
service design refined model






```
transformation ServiceSpecificationToServiceDesignRefinedModel (ss: asdl, sdrm: asdl)
{
  top relation BehaviourMapping {

     checkonly domain ss ssBehaviour: Behaviour {name = 'ServiceSpecification'};
     enforce domain sdrm sdrmBehaviour: Behaviour {name = 'ServiceDesignRefinedModel'};
  }

  top relation EntryPointMapping {

     entryPointName: String;

     checkonly domain ss ssEntryPoint: Entry {
     name = entryPointName,
     parent = ssBehaviour: Behaviour{}
     };
     enforce domain sdrm sdrmEntryPoint: Entry {
     name = entryPointName,
     parent = sdrmBehaviour: Behaviour{}
     };
       when {BehaviourMapping(ssBehaviour,sdrmBehaviour);}
   }
}
```

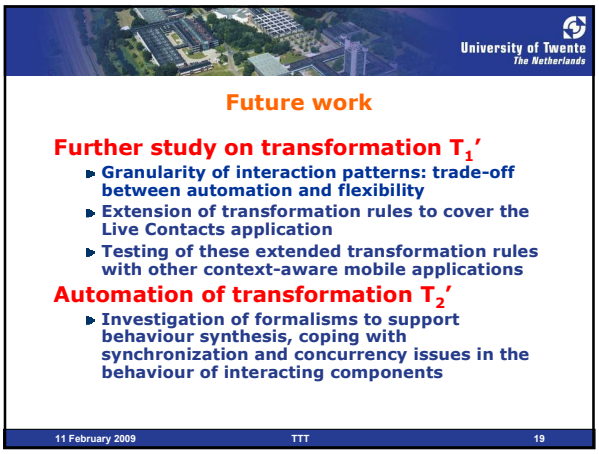

Conclusions

Our MDA-based approach divides the PIM level in tree models:

1. **Service specification**
2. **Service design refined model**
3. **Service design component model**

- **Two transformations**
  1. **Service specification -> Service design refined model**
  2. **Service design refined model -> Service design component model**

Towards the automation of this approach, we have realised the transformation T_1' with the Medini QVT tool


University of Twente
The Netherlands
Future work
Further study on transformation T_1'
Granularity of interaction patterns: trade-off between automation and flexibility
Extension of transformation rules to cover the Live Contacts application
Testing of these extended transformation rules with other context-aware mobile applications
Automation of transformation T_2'
Investigation of formalisms to support behaviour synthesis, coping with synchronization and concurrency issues in the behaviour of interacting components
11 February 2009
TTT
19