UML Function Transformations

René Ladan

Thiel Chang

Jelle Gerbrandy

TU/e

SAN

Getronics PinkRoccade

UML Function Transformations

- Why this Subject
- Research Questions
- Proposed Solution
- Conclusions

TU/e

SAN

Getronics PinkRoccade

Why this Subject - Model Driven Architecture

- MDA is a new approach to develop large software systems.
- CIM → PIM → PSM → code
- UML Component models should become PIMs, this way the model can be transformed using MDA.

Why this Subject – The Problem

- UML Component Method and AOSD
  - provides potential for
    - improved structuring of design
    - reuse of components and aspects in different designs
  - are incompatible with MDA tools
    - AOSD models need weaving
    - UML Component models need transformations
  - as model transformations?

TU/e

SAN

Getronics PinkRoccade

Why this Subject – Causes

- There is a gap between the UML template specification and contemporary tool implementation.
- QVT is
  - a proposal of the OMG to define transformations
  - not yet an implemented standard
- It would be nice if UML templates could be used as an aid to transform models when using MDA tools.

TU/e

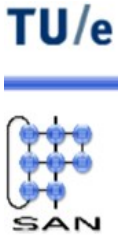
SAN

Getronics PinkRoccade

Why this Subject - Goal

Develop a method to automatically transform UML Component models

- use this method to describe automations
- must be applicable from inside the model



Getronics PinkRoccade

UML Function Transformations

- ✔ Why this Subject
- Research Questions
- Proposed Solution
- Conclusions

TU/e

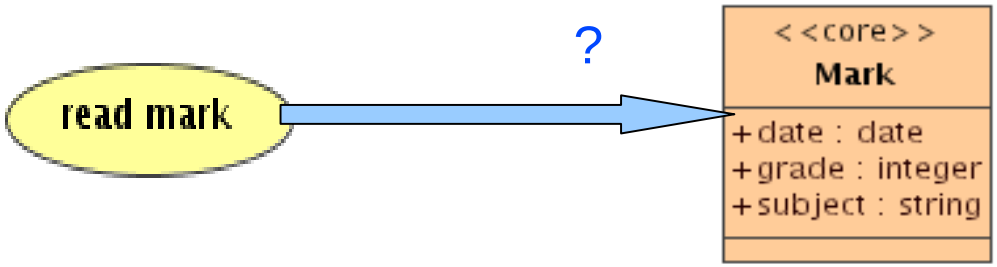
SAN

Getronics PinkRoccade

Research Questions - How To Transform Models

The main research questions are

- Can we transform UML Component models?
- Do we need some metamodel for these transformations?
  - If yes, how does it look?
  - If no, why not?



TU/e

SAN

Getronics PinkRoccade

Research Questions - Required Transformations

- We need transformations to automate
  - the entire UML Component Method as defined in [1]
  - the fundamentals of AOSD [2]
- The transformations should replace the manual transformations.
- References
  - [1] “UML Components”, Cheesman / Daniels
  - [2] “AOSD with Use Cases”, Jacobson / Ng

TU/e

SAN

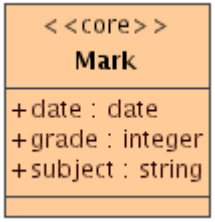
Getronics PinkRoccade

Research Questions - Functions

Transformations are equal to functions

- domain: parts of UML models

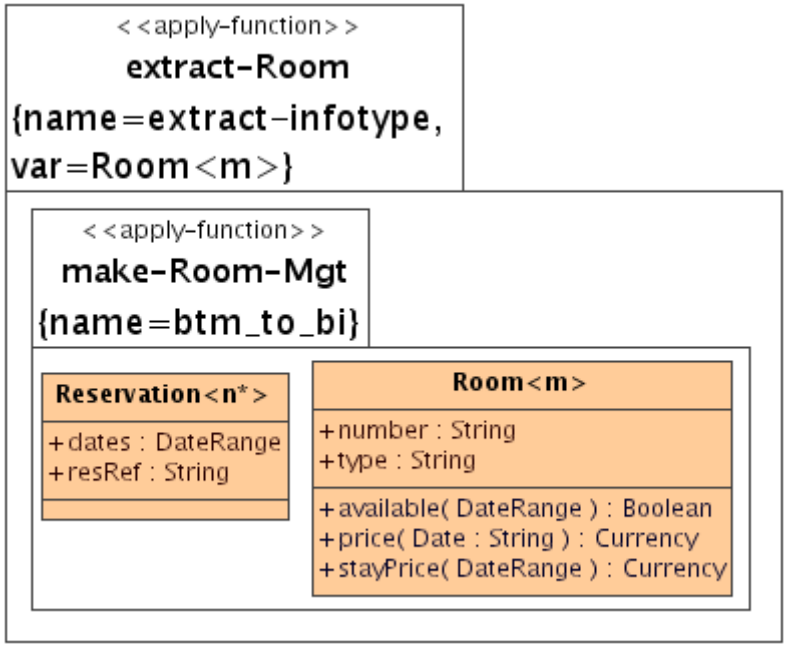
- co- domain: parts of UML class diagrams



Getronics PinkRoccade

Research Questions - Nested Function Applications

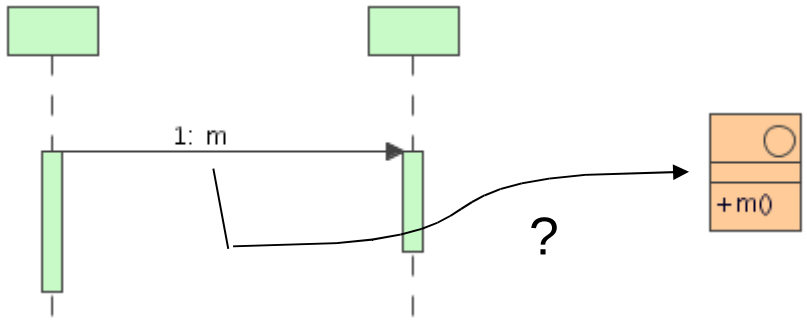
It is desirable to be able to nest UML function applications, just like mathematical function applications (e.g. f(g(...))). This way we can reuse UML functions to keep the number of UML functions manageable.



Research Questions - Combining Diagram Types

A real-world question which comes to mind when applying UML methods:

"Assume we want to transform a message of a sequence diagram into an interface method of a class diagram. UML forbids us to directly draw the sequence chart and the interface into one diagram."



TU/e

SAN

Getronics PinkRoccade

UML Function Transformations

- ✔ Why this Subject
- ✔ Research Questions
- Proposed Solution
- Conclusions

TU/e

SAN

Getronics PinkRoccade

Proposed Solution - General Idea

Implement the UML transformations as a set of functions and scripts. A function is only a recipe, a script is something which executes functions. Use these scripts to automatically apply the functions to the UML models.

TU/e

SAN

Getronics PinkRoccade

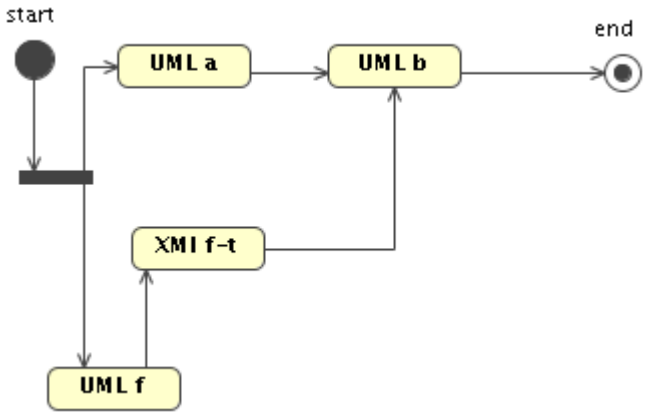
Proposed Solution – Ideal Scheme

UML a: input model

UML b: output model

UML f: UML Template model (function)

XMI f- t: templated XMI version of **UML f** (reference point)



TU/e

SAN

Getronics PinkRoccade

Proposed Solution - Used Scheme

UML a: input model

UML b: output model

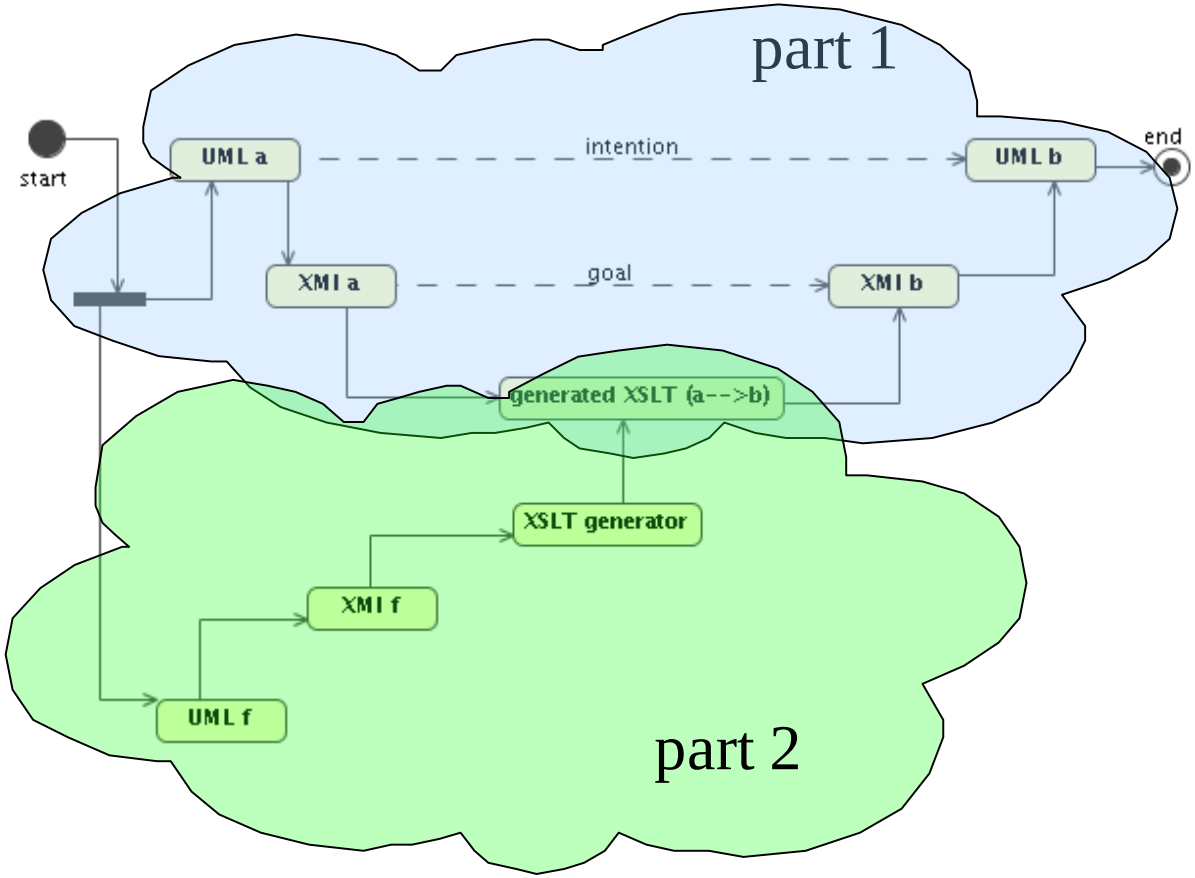
UML f: UML function model

XMI a: XMI version of **UML a**

XMI b: XMI version of **UML b**

XMI f: XMI version of **UML f**

generated XSLT: XSLT generated by **XSLT generator** from **XMI f**



TU/e

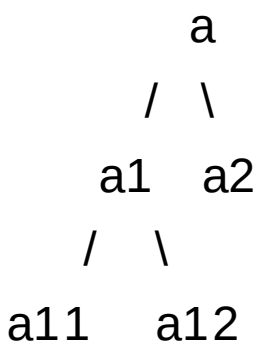
SAN

Getronics PinkRoccade

Proposed Solution – Metamodel

The metamodel for UML functions is based on λ- calculus

- more natural for this research than OMG templates, as they better describe the UML function characteristics
- work with lists, here nested lists of UML elements
- properties of UML elements are denoted as *e.p*, where *e* is the element identifier and *p* is a property of *e* (e.g. *name*, *stereotypes*)



==

[a, [a1, [a11, a12], a2]]

Proposed Solution - Rules

- UML functions need rules (pre- and postconditions)
- Described in thesis chapter 4.3
- λ-calculus-based rules accompanied by textual descriptions.
- Rules for
  - function definition (<<function>>)
  - function application (<<apply-function>>)
  - parameters

TU/e

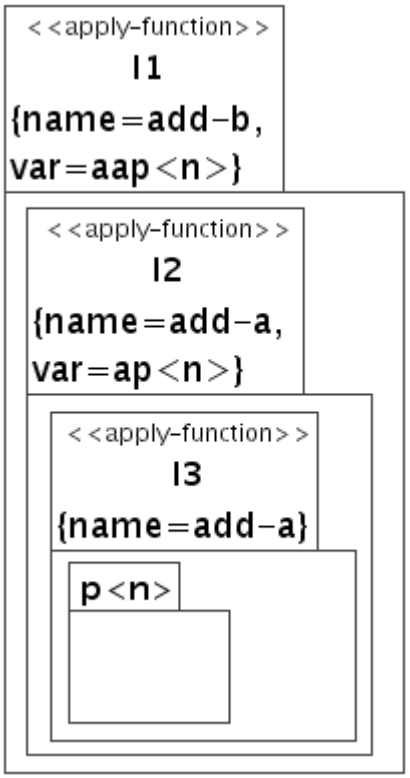
SAN

Getronics PinkRoccade

Proposed Solution – Nested Function Application Idea

To nest one function application inside another
- nest the <<apply-function>> packages
- use the tag 'var' to indicate parameter coupling of the resulting elements

- The example shows the nested application add-b(add-a(add-a(p)))



TU/e

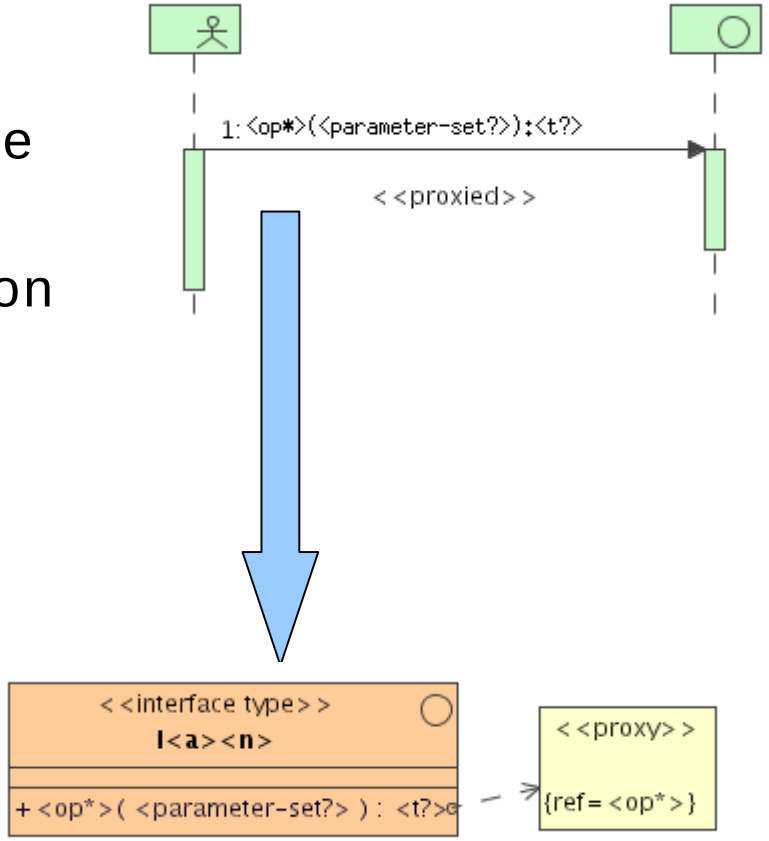
SAN

Getronics PinkRoccade

Proposed Solution – Proxy Idea

Transform <op*> calls into <op*> methods

- these calls are referenced by the <<proxy>> object
- the resulting methods depend on this <<proxy>> object



Proposed Solution - Script Generator

- Design and implement a generator in XSLT
  - translate the UML functions into XSLT using this generator
  - this generator must handle both the UML Component Method and AOSD
- XSLT was chosen because it best fits the existing development practices.

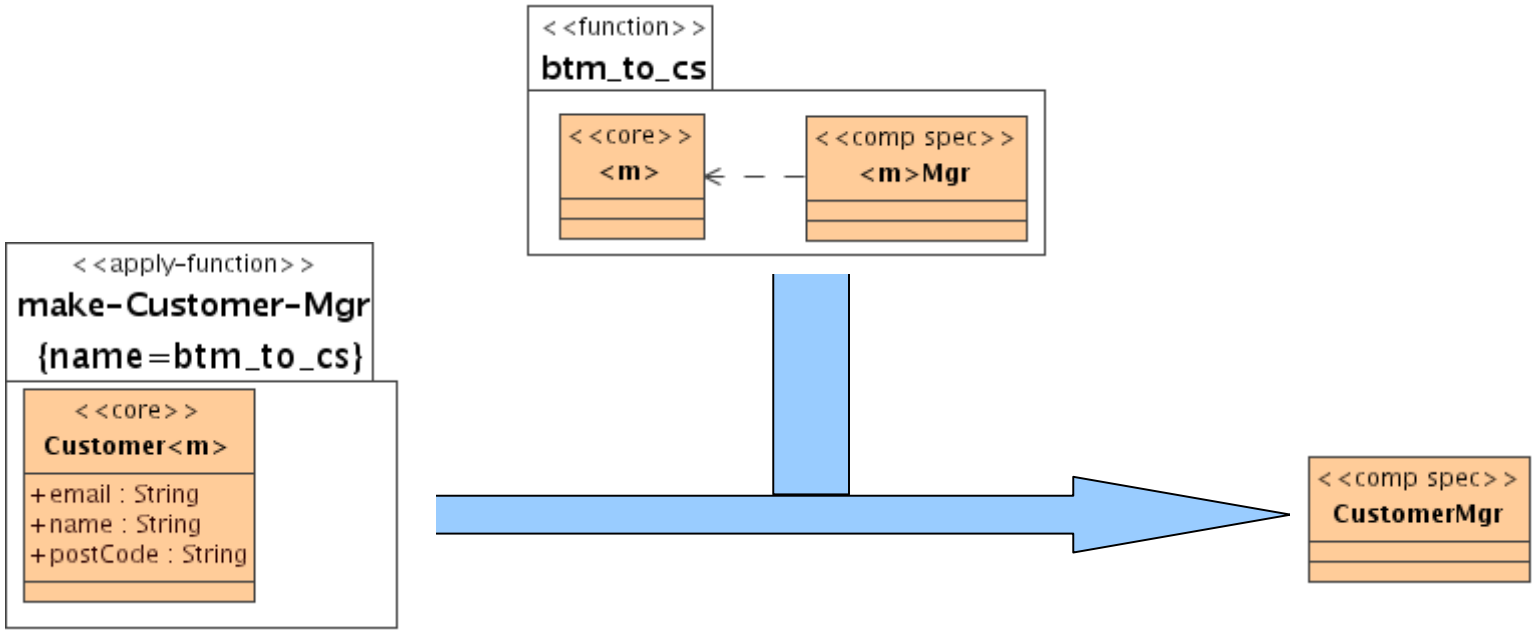
TU/e

SAN

Getronics PinkRoccade

Proposed Solution – Example 1/2

Description of function *btm_to_cs(<m> : UML_Class) : UML_Class*



TU/e

SAN

Getronics PinkRoccade

Proposed Solution – Example 2/2

btm_to_cs =
λN.λm : UML_Class.(N – m + m' : UML_Class[m'.name = m.name ++ 'Mgr' ∧ m'.stereotypes = m.stereotypes - {<<core>>} + {<<comp spec>>}]

btm_to_cs(c : UML_Class[c.name = 'Customer' ∧ c.stereotypes = {<<core>>} ∧ c.attributes = ['email' : string, 'name' : string, 'postCode' : string])
=
c' : UML_Class[c.name = 'CustomerMgr' ∧ c.stereotypes = {<<comp spec>>}]

TU/e

SAN

Getronics PinkRoccade

UML Function Transformations

- ✔ Why this Subject
- ✔ Research Questions
- ✔ Proposed Solution
- Conclusions

TU/e

SAN

Getronics PinkRoccade

Conclusions - Summary

A method has been developed to automate transformations for the UML Component Method and AOSD, using a set of UML functions which are based on λ-calculus and a generator which translates these functions to XSLT code.

TU/e

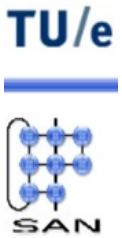
SAN

Getronics PinkRoccade

Conclusions - Deliverables

The deliverables of this project are available at
http://home.tiscali.nl/rladan/tupt/

- thesis
- tool
- user manual for the tool
- models containing
  - all UML functions
  - examples
- this presentation



Getronics PinkRoccade

Conclusions - Questions

?????
?? ??
  ??
 ??
??

??

TU/e

SAN

Getronics PinkRoccade