Using an MDA approach to model traceability within a modelling framework

J A Dalton, P W Norman, S Whittle* and T E Rajabally

Metamodelling for MDA,
Kings Manor, York
24 - 25 November, 2003

Presented by: John Dalton

Engineering Design Centre, University of Newcastle upon Tyne
*BAE Systems, Warton
Introduction

♦ Purpose of this presentation
  ♦ To describe how we modelled an approach to traceability using an MDA approach

♦ Traceability and the modelling framework
  ♦ The tracing methodology is specifically based on a modelling framework that we have been working on for 3 years
    this is part of an ongoing research project with the sponsors
  ♦ The original framework was modelled using UML, however the idea of MDA was not around when we started
    the early Java development of the framework was based on a UML approach

♦ Approach
  ♦ To properly describe the tracing methodology I will first outline the modelling framework
    i.e. the why, what & how
  ♦ Then the idea of traceability and how it fits into the overall picture
  ♦ Followed by the modelling approach used to design the tracing method
Outline of presentation

♦ **Modelling framework representation**
  ✷ The basics of how we designed and implemented a modelling framework information system
  ✷ Production of demonstration software

♦ **What we mean by traceability**
  ✷ What are we tracing and why
  ✷ How a trace is to be displayed

♦ **How we implement traceability using modelling tools**
  ✷ Modelling the process using an MDA approach
  ✷ Results obtained using this, and
  ✷ Generalisation of the outcome
Modelling framework representation

♦ Design of complex systems
  ♦ There is a need for a consistency in the management of data which an integrated approach to modelling can provide
  ♦ Large organisations involved in the design of complex systems have a wide range of models which operate independently
  ♦ Benefits can be derived if these can be integrated by use of a modelling framework to synthesise and examine a system at the design stage

♦ Advantages
  ♦ A means of reducing development costs by using system simulation to reduce testing by providing a rapid prototyping environment for the design engineer
  ♦ Expect this to improve the quality and reduce lead-time of designs by ensuring that all necessary information is available and within agreed specifications early in the design process
  ♦ Frameworks of models also enable system properties and other aspects such as design decisions to be traced and recorded
Modelling framework representation

- Designed an information system to integrate models
  - Information system created using UML

- Implemented as demonstration program
  - A cross-platform application using Java, tested on a range of platforms i.e. Windows, Unix, Linux.

- Features
  - Models represented by components within a framework
  - Frameworks collectively representing sub systems and systems
  - Information, in the form of properties and text descriptions can be exchanged between components and frameworks
  - Makes extensive use of XML
Conceptual view of modelling framework
Physical Domain

- Actual/design
- Fuel Pump
- i.e. design of fuel pump
- other physical domain information included in framework

Modelling Domain

- Computer Based Model
- Models Behaviour & Functionality
- modelling information represented by framework

Framework Component Representing Model

- Physical and Structural Information about Fuel Pump
- Goals/Design Definitions
- Representation of Functionality
- Representation of Behaviour
MFR
Class diagram

Framework

Component
  Title
  Current state

Information
  Project
  Author
  Date
  Description

CommonInterface

Model
  Type
  Location
  Author ...

State
  Description

Property
  Identifier
  Expression

GenericObject
  Type
  Title
  Prefix

Association

InterfaceObject
  Behaviour
  Function
  Goal
  Structure
Traceability within the modelling framework representation

♦ Types of models that traceability can be performed on
  ♦ information model
    entity relationship model, i.e. MFR
  ♦ process model
    describe system development
  ♦ documentation model
    reference
  ♦ enterprise model
    organisation structure

♦ Information model
  ♦ All tracing requires some form of relationship
    i.e. a syntactic relationship such as a function
    Force = \( \phi \) (mass, acceleration)
  ♦ We deal here with the tracing of properties
    the method, however is generic and can be applied
    to any linked objects, such as design decisions
Traceability within the modelling framework representation

♦ Advantages
  ♦ Incorporation of traceability between emerging high level properties and low level properties, both upwards and downwards
    for example: we can perform a downward trace on an engine sub-system to find out which parts are contributing most to the overall mass
  ♦ Allows designers to target their design effort
  ♦ Can act as a measure and test for sensitivity
    i.e. in identifying the required fidelity for a model
Tracing properties downwards

Result (r)

\[ \text{force} = o1.m \times o2.a \]

Object (o1)

\[ m = \text{rho} \times \text{vol} \]

Object (o2)

\[ a = (v2-v1)/o3.time \]

Object (o3)

\[ \text{time} = 5 \]
Model Driven Architecture (MDA) approach to traceability

- Register an interest in the MDA approach
  - Well suited to the type of work being done

- Executable UML models support the OMG initiative
  - Compose complete systems from models
  - Creates executable code directly from a set of UML models

- Physically, Executable UML comprises of a set of models
  - Consists of 3 fundamental projections
    - class model: identifies, classifies & abstracts the environment under study
    - state machines: how objects change during their lifecycles
    - states procedures: actions which carry out the actual computation of the system

Ref: (Mellor 02)
Overview of tracing process

- Framework data structure
  - Class diagrams
  - Tracing data structure
    - Populate data structure
      - Search data structure
        - Return a tree model
          - Display trace tree
            - Supply a property and its location
              - Class diagram
              - Sequence diagram
              - Activity diagram
Class diagram of the data structure
Tracing upwards

\[ \text{<property> } = \Phi(\text{param}_0, \text{param}_1, \ldots, \text{param}_n) \]
Sequence diagram: populating the data structure

1. **PropertyTrace**: Hashtable
   - **Component List**: ArrayList
   - **Component**:
   - **Object**:

2. **constructData()**
   - **getComp(i)**
   - **getObject()**

3. **TraceRecord**: Hashtable
   - **Comp title, Object title, Object prefix and ModelText**
   - **Strip&Parse**:
     - **ModelText**
     - **Obtain Property Name as KEY for TraceRecord and an ArrayList of all constituent properties.**
     - "alpha = beta + gamma"
     - Key = alpha, and ArrayList contains beta and gamma

4. **Collect an ArrayList of all links that arrive at this object, or depart (if double)**

5. **Add a list of the linked objects to the TraceRecord parameter list**

**Notes:**
- For each component on list
- For each object contained in component
- For each line in ModelText

**Put on PropertyTrace**
- **Key:** Component Title + Object Title + Object Prefix
- **Contents:** TraceRecord

**Put on TraceRecord**
- **Key:** Property name
- **Contents:** ArrayList of constituent properties
Activity diagram:
Search

get list of properties from TraceRecord

level ++

property not prefixed
property prefixed

get property object from TraceRecord based on key: property
get list of objectID's from TraceRecord

level = 1

objectID not in PropertyTrace?

property not in TraceRecord?

get property object from TraceRecord based on key: property
create a tree node then add to tree at a depth of level
Display is "property"

property is a function of at least one other property

is a leaf

level > 1

level = 1

level --

object ID has changed
object ID is same

for each property in list

terminate

return

doSearch(objectID, property, level)
Framework application view

dp = 2.384
Summary

♦ Modelling framework representation
  ♦ Designed and built a modelling framework representation to simulate systems and subsystems at an early stage in the design work

♦ Traceability
  ♦ Conceptualised traceability
    * i.e. derived the methodology related to properties & directionality
  ♦ Modelled the methodology

♦ Conclusions
  ♦ Using an MDA approach permitted a full understanding of the traceability method at the early modelling stage
  ♦ By viewing the produced models, a generalisation of the tracing method could be made
    - tracing of other objects, permitting this method to be used on other types of models, i.e. design decisions
  ♦ Made possible extensions to the tracing process
    - tracing upwards and downwards
    - the potential for traceability increased
      * i.e. what was nice to have is now a must have