Model-Driven Development of a Biosignal Analysis Framework: Benefits and Impacts on Processes

Nikolas Hofmann · 08.11.2012
Introduction

- Biosignal analysis has versatile applications in medicine
- Mobile devices allow new scenarios
- Used systems vary:
  - Hardware
  - Platform (OS, frameworks, ...)
  - Software tools, e.g. processing software
  - Programming languages
- But: processes and methods of biosignal analysis remain in general the same!
Introduction

• Features
  • Acquisition and processing of biosignals
  • Real-time: infinite signals
  • Support of different platforms, here Java and Matlab

• Quality attributes:
  • Scalability
  • Multi-platform support
  • Ease-of-use
Introduction

• The systems share main features
  • → they constitute an application family/ product line
• Reuse of abstract
  • Knowledge
  • Structures
• Application frameworks address this type of reuse
  • Provide domain-specific structures and functionalities
  • Allow wide range of applications
Model-driven software development

- Model-driven software development (MDSD)
  - Developing models from which a software system can (partly) be generated
  - Mapping the problem domain to the technology domain
- Architecture is splitted up
  - Application architecture
  - Domain architecture
    - Models
    - Platforms and their transformations
Model-driven software development

- Model
  - Intention of models in MDSD gets more prescriptive than descriptive
- Metamodel
  - Defines a domain-specific language (DSL)
- Meta-metamodel
  - Defines elements of the metamodel
  - Self-describing
Model-Driven software development

- Models in the Model-Driven Architecture (MDA) by the OMG
  - Computing independent model (CIM)
  - Platform independent model (PIM)
  - Platform specific model (PSM)
- Transformations
  - Link or combine models („mapping“)
  - Manually or automatically
Model-Driven software development

- Models in the Model-Driven Architecture (MDA) by the OMG
  - Computing independent model (CIM)
  - Platform independent model (PIM)
  - Platform specific model (PSM)
Development process

- Overview:

```
Domain architecture development

Requirements engineering

Modelling

Transformation / generation

Manual implementation
```

Feedback arrow between Domain architecture development and Modelling.
Development process

- Domain architecture tasks
- Roles:
  - Domain expert
  - Domain analyst
  - Domain architect
  - Application developer (customer)
Development process

- Rather agile development
  - Short iterations
  - Prototyping
  - Refactoring
- Advantages:
  - Considering feedbacks between processes
  - Supports learning of the different tools
Domain architecture: domains

- Decomposition of the system along two axis
  - Horizontal: technical domains (layering)
  - Vertical: problem/subject domain, „scopes“ (components)

- Example domain biosignal analysis:
  - Several steps: acquisition, transport, processing, visualisation
Domain architecture: domains

Acquisition → Processing → Visualisation

Load data → Show data table

- Biosignals
- Nodes and connectors
- Components
- Eclipse components

Analysis process
Domain architecture: prototyping

- First a prototype for one platform
  - Validation of requirements
  - Testing of technologies and domain concepts
  - Represents one member of the application family
- Afterwards implementation on another platform
Development process

- Domain analysis
- Reference model development
- Reference implementation
- Deriving transformations
Domain architecture: reference implementation

• Refined implementation of the first prototypes
• Aims at developing the domain architecture
• Used as reference
  • Results of domain architecture development continuously compared with
Domain architecture: retrieving models

• Models
  • Abstraction of all reference prototypes
• Platform Independent Models (PIMs):
  • Implemented here as textual DSL
  • Languages and models are developed alternately
• Comparison with reference model demands code generation
  • → Development of generators in parallel
Development process

Domain analysis

Reference model development

Reference implementation

Deriving transformations
Domain architecture: transformations

• More levels of abstraction are necessary
  • Separation of concerns (domains)
  • Reduction of complexity
• Biosignal data structures modelled as PIM
• Platform-specific details moved to PSMs
  • Allows multiple-platform support
  • Differences are e.g. the dealing with data sequences (arrays vs. matrices)
  • → reuse of PIMS (more abstract models)
Domain architecture: code generation

- Source code can be generated from models
- Model-to-text transformation
- → Generators
  - For each target language
Discussion

• Short iterations
  • Support interaction between different tools
  • Allow feedbacks → distribution of tasks to different roles or persons
  • „parallel“ development
  • Reduce risks
• Development of models and transformations for an additional platform can be done with less effort
• Initial effort is high
• Development strongly relies on tools
  • Which are often in an early development stage
  • Changing and incompatible versions
Thank you for your attention!