Lessons Learned in the Current Application of Model-driven Engineering

Stephanie E. August
saugust@lmu.edu

Christina Chiu
cchiu3@lion.lmu.edu

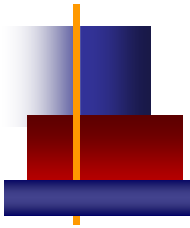
Steven R. Doran
sdoran@lion.lmu.edu

Sukhanya Kethuneni
skethune@lion.mu.edu

Matthew Shields
matthew.james.shields@gmail.com

Mesrop Simonyan
mesrop.simonyan@paramount.com

Loyola Marymount University



Assessing the Maturity of Model-driven Engineering

- Context: Graduate Seminar on Advanced Modeling of Software Systems
- Goal: Understand MDE significance to software engineering and its application to selected real-world systems
- Objectives:
  - Gain experience applying MDE, DSMLs to a variety of problem domains
  - Examine how the MDE facilitates domain-specific problem solving
  - Assess maturity of this paradigm

OMG Methodology

Computation independent model (CIM)

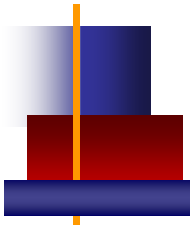
Domain-specific modeling language (DSML)

Platform independent model (PIM)

Platform specific model (PSM)

Implementation

Develop/conduct model transformations in selected problem domains



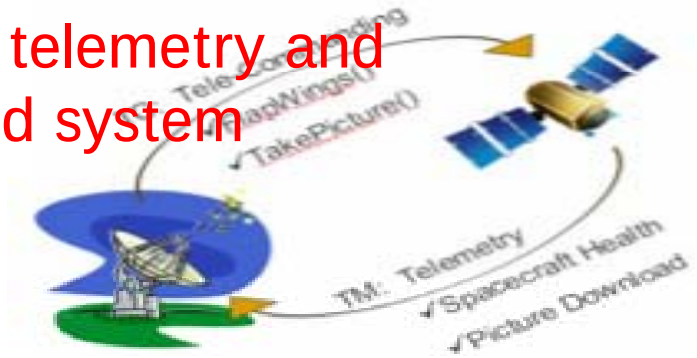
Problem Domains Areas

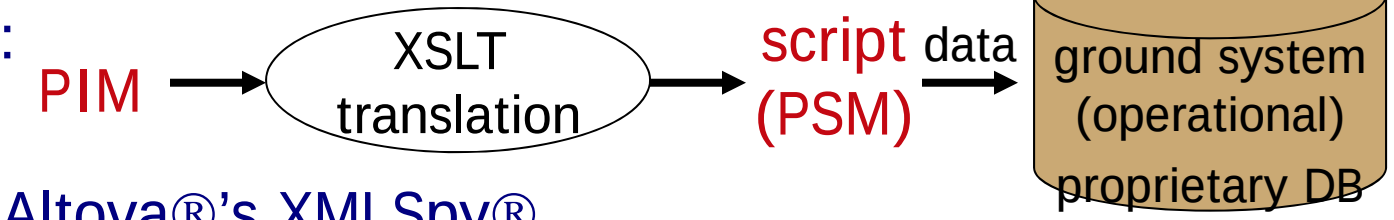
- Data Exchange in Spacecraft Telemetry and Control (Shields)
- Graphical User Interface Modeling (Chiu)
- Re-engineering a Monolithic Large Satellite as a Fractionated Spacecraft (Doran)
- Evaluation of a Virtual Engineering Science Learning Lab (Kethuneni)
- Developing a Domain Specific Model within an Agile Development Workflow Process (Simonyan)

Problem, Approach, Tools, Lessons/Observations

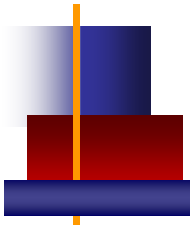
Spacecraft Telemetry and Control

- **Problem:** Transforming manufacturer telemetry and telecommanding data for use by ground system
- **Approach**:
  - Use XTCE and XSLT to transform contractor/manufacturer (problem-oriented) database → customer (operational) ground database
  - CIM: XTCE standard schema used to describe T&C domain
  - PIM: XTCE XML instance
  - PSM:





- **Tools**: Altova®'s XMLSpy®



Spacecraft Telemetry and Control

- **Lessons/Observations:**
  - CIM/DSL, PIM, PSM need not be UML-based
  - Application of XTCE in this domain was highly successful
    - Cost and schedule benefit
      - In managing changing data needs
      - Minimized duplication of effort between manufacturer and customer
    - PIM was useful for understanding data requirements
    - Can scale to map to multiple projects (product-line/reuse implications)
      - Managing complexity during model transformations still a challenge
    - XSLT transformations were straightforward because the XTCE was rich enough to permit PIM to PSM mappings
    - Implementation can support automated code-generation technologies from XML design
    - Model becomes the code (fix, test the model not code)
      - Moves away from code-focused development practices
      - Relies on good code generation tools—maturing

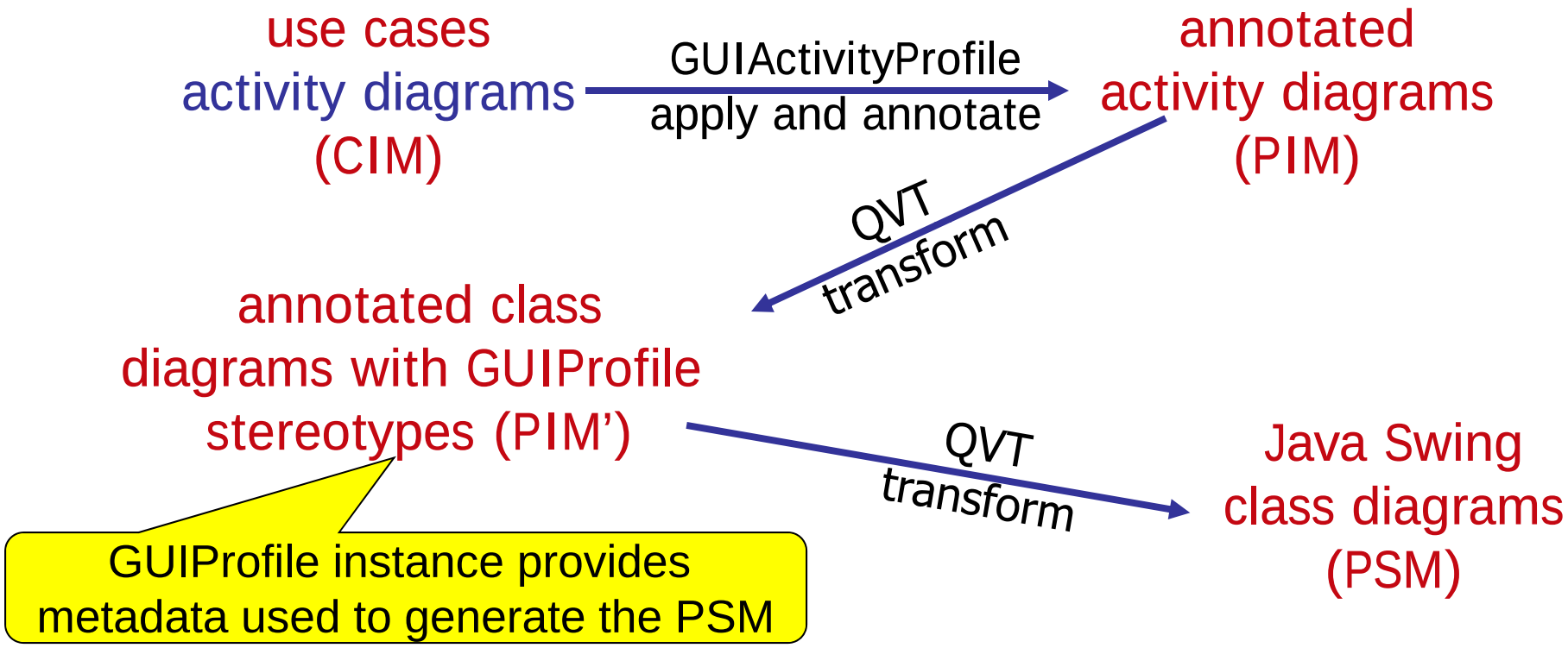
Graphical User Interface Modeling

- **Problem:** Develop a partially automated music-playing device to play user-selected songs from self-contained media.

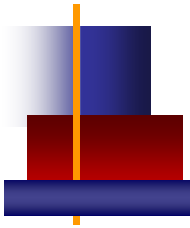
  How can you represent GUI architecture using MDE?

- **Approach:**
  - CIM describes structure, content, and function of user screens
    - Use cases and activity diagrams capture functional requirements and decision choices

Graphical User Interface Modeling



  - OCL used to specify model constraints
- **Tools:** Eclipse GMF, EMF, QVT) (Link, et al.)



Graphical User Interface Modeling

- **Lessons/Observations:**
  - MDE GUI modeling techniques evolving
  - MDE improved development and facilitated maintenance
  - Challenges:
    - High tool learning curve
    - Achieving model completeness
    - Debugging model representations, profiles
    - Defining model transformations
    - Paucity of MDE experts

Re-engineering Monolithic Large Satellites as Fractionated Spacecraft

- **Problem:** Can MDE reduce cost/schedule/complexity of a large monolithic satellite system by re-engineering it into a collection of fractionated spacecraft?
  - Decompose functions into wireless networked cluster of smaller mission microsatellites (graceful degradation)
  - Can MDE generate common flight code for each satellite?

- **Approach:**
  - CIM: High level state model in UML for flight software
  - Function mapping from CIM states to PIM managed as traceable relations
  - PIM: Use UML to capture requirements (use cases), functional behavior and structure
  - PSM: Use case model of PIM
    - exported into an XMI file, which is imported into an EMF meta model (ecore), which is used to generate code
    - translated into ER diagrams command telemetry for EMF code generation
- **Tools:** Gaphor (UML) EMF (model translation), MySQL Workbench (PSM ERD, database generation)

Re-engineering Monolithic Large Satellites as Fractionated Spacecraft

- **Lessons/Observations:**
  - Project experienced “pain” of integrating and transforming models in an open source, multi-vendor environment
    - *UML model interchange relies on XMI but vendor incompatibilities still exist (abstract model and diagram info)*
      - What gets exported and how can vary
    - Rose mdl format changed
      - Rose mdl support in open EMF relies on older mdl format
      - COTs support needed to handle newer formats
    - EMF ecore as a PSM has code generation support but may be too low level as a useful PSM (e.g. managing platform-specific meta-information)
    - Some EMF APIs have changed as eclipse has evolved
    - Gaphor UML tool is easy to use, but has fewer features and a few bugs--inadequate MDE tools, lack of text support

Re-engineering Monolithic Large Satellites as Fractionated Spacecraft

- **Lessons/Observations (cont.):**
  - Avoid older UML 1.x modeling tools/formats for new development
    - UML differences between 1.x and 2.0 are not transparent making interpreting models and translating models difficult.
  - *Seek common tool family suites to minimize incompatibilities*
  - Model transformation languages such a QVT implementations will continue to evolve
  - Auto-generation of SQL database from ERD is feasible, mature, and should be considered for complex databases
  - Eclipse project has many (often competing) modeling efforts

Evaluation of a Virtual Engineering Science Learning Lab

- **Problem:** Workflow Analysis: Apply MDE to instructor evaluation of student lab use
- **Approach:**
  - Use BPMN (CIM) to capture student/instructor's PIM workflow
  - Use cases developed to define processes, actors, classes to elaborate on BPMN concepts
  - PIM structure also developed as UML classes
  - Some automated ER diagram generation
  - Work still on-going
- **Tools:** Borland Together 2008

Evaluation of a Virtual Engineering Science Learning Lab

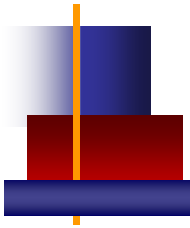
- **Lessons/Observations:**
  - No comprehensive tool support for MDE, but Together is excellent
  - Shortage of skilled help to incorporate MDE ideas
  - Tool mastery essential, but time-consuming
  - MDE training should be considered
  - Tool evolution can have development side-effects
    - Maturity of automated code generation, evolving Java, Eclipse dependencies

Developing a Domain Specific Model within an Agile Development Workflow Process

- **Problem:** Workflow in an agile environment of a film distribution system
  - MDE as a communication vehicle for non-technical people
- **Approach:**
  - Develop DSML terms using business-level process workflows (CIM)
  - Characterize the PIM as structural class diagrams (no methods)
  - PSM as an elaborated PIM
- **Tools:** DSL Tools plugin for MS Visual Studio

Developing a Domain Specific Model within an Agile Development Workflow Process

- **Lessons/Observations:**
  - Management commitment/perceived value of MDE can affect the degree of planned effort
  - When applying a DSL approach within an agile environment stakeholder uses of model, common terminology, and their technical skill set can vary significantly (business needs vs developer needs)
  - Domain experts can really help in the DSL
    - Avoid inventing new notation different from expert's



Conclusions

- There are many ways to accomplish MDE
- Seek interoperable approaches that facilitate model interchange
- XML based MDE approaches have achieved success
- Many of the MDE tool-sets have not fully matured, especially the open source tool sets.
  - Their evolution can affect dev environments
  - This will be an issue for some time
- MDE modeling tool limitations need to be thoroughly researched before their adoption in advanced transformation environments
  - Choose your MDE tools wisely!
  - Model interchange, UML versions, sysML vs UML+custom profiles...
- Tool support and training needed in order for MDE to be effective
- Management support of MDE approaches in an agile environment is essential
- Variation of stakeholder technical skill sets can affect model communication and its use

References (1)

- Altova®'s XMLSpy. XML Editor, Data Management, UML and Web Services tools. http://www.altova.com/ (last accessed 24 FEB 2010)
- Gaphor. http://gaphor.sourceforge.net/ (last accessed 24 FEB 2010)
- France, R. and Rumpe, B. 2007. Model-driven Development of Complex Software: A Research Roadmap. 2007 Future of Software Engineering (May 23 - 25, 2007). International Conference on Software Engineering. IEEE Computer Society, Washington, DC, 37-54. doi: http://dx.doi.org/10.1109/FOSE.2007.14.
- Link, S., Schuster, T., Hoyer, P., and Abeck, S. 2008. Focusing Graphical User Interfaces in Model-Driven Software Development. Proceedings of the First international Conference on Advances in Computer-Human Interaction (February 10 - 15, 2008). ACHI. IEEE Computer Society, Washington, DC, 3-8. doi: http://dx.doi.org/10.1109/ACHI.2008.16

References (2)

Margaria, Tiziana and Steffen, Bernhard. Continuous model-driven engineering. *Computer*. 42:10, Oct. 2009, pp. 106 - 109.

Pham, H. N., Mahmoud, Q. H., Ferworn, A., and Sadeghian, A. 2007. Applying Model-Driven Development to Pervasive System Engineering. *Proceedings of the 1st international Workshop on Software Engineering For Pervasive Computing Applications, Systems, and Environments* (May 20 - 26, 2007). International Conference on Software Engineering. IEEE Computer Society, Washington, DC, 7. doi: http://dx.doi.org/10.1109/SEPCASE.2007.2.

Schmidt, Douglas C. Guest Editor's Introduction: Model-Driven Engineering. *Computer*, vol. 39, no. 2, pp. 25-31, Feb. 2006, doi:10.1109/MC.2006.58.

Steffen, Bernhard. Continuous model-driven engineering. *14th IEEE International conference on Engineering of Complex Computer Systems*. 2-4 June 2009, pp. xi-xi.