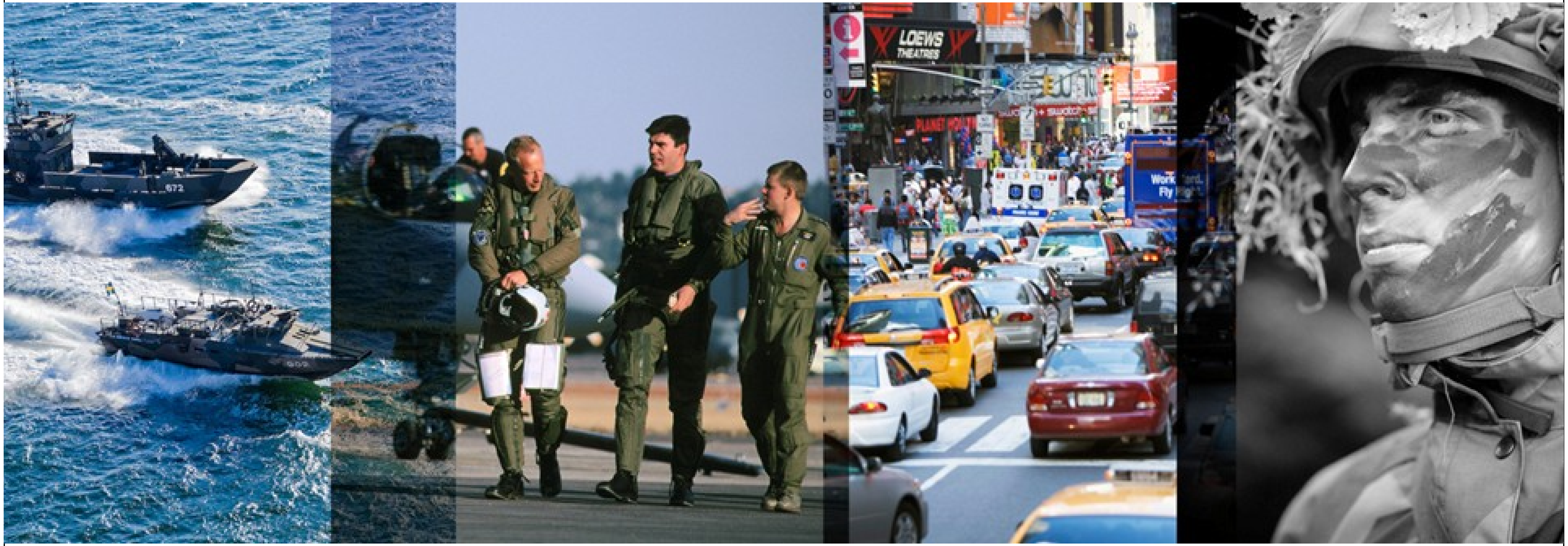
MODPROD 2011

xtUML in practice

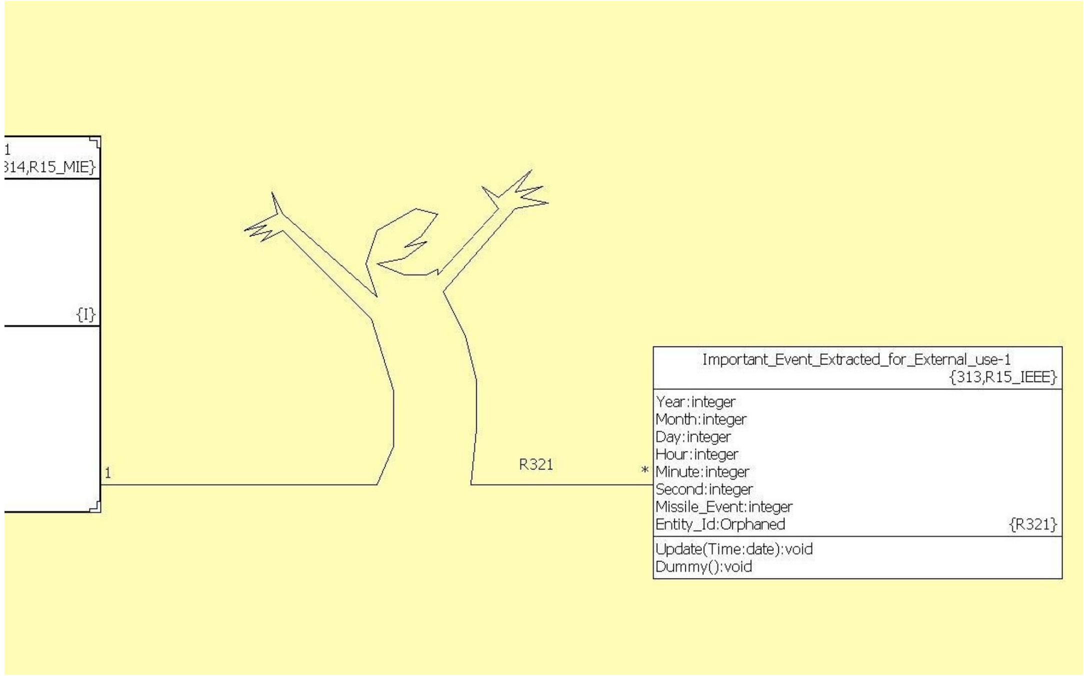


Martin Nilsson

Feb 9 2011

xtUML IN PRACTICE
Agenda

- Modelling background
- MDA and xtUML
- Experiences



Important_Event_Extracted_for_External_use-1
{313,R15_IEEE}

Year:integer
Month:integer
Day:integer
Hour:integer
Minute:integer
Second:integer
Missile_Event:integer
Entity_Id:Orphaned {R321}

Update(Time:date):void
Dummy():void

MODELLING BACKGROUND

DIFFERENT MODEL ORIENTED WAYS

- Sketches ~ Traditional development
  - informal capture of ideas and concepts
  - explanation images i traditional documents
  - coding may proceed directly
- Blue Prints ~ SysML/UML
  - specifies product structure architecture
  - close one-to-one mapping towards implementation
  - code structure / skeleton can be generated – code is added to the model in target language or via separate files
- Executable models ≡ Executable specifications ~ xtUML
  - separates the application from its implementation, i.e. platform independent parts from platform unique parts
  - the model does not need to have one-to-one mapping towards implementation

MODELLING BACKGROUND

- Saab has been Using xtUML in:
  - Flight demonstrator
  - Flight planning systems
  - Missiles
  - Missile training system
  - Demonstrator systems
  - Software for usage analysis and tactics/methods development at FM
- Translating models into:
  - C++
  - Ada95
  - PD80
  - SQL/XML for persistence
  - html for documentation/reviews
- Reusing models for translation into different languages and platforms for different projects

MDA AND xtUML

THE PROMISES OF MDA AND xtUML

- Executable models/specifications (with an abstract and executable action language)
  - Early validation of specifications. Execute the models while building them, without code generation
  - Raise the abstraction level to a suitable level for each subject matter
  - Reuse of applications across architectures and platforms
- Translation of executable models/specifications to implementations
  - Repartition functionality across a platform, e.g. between software and hardware
  - Retarget functionality to new/modified platforms
  - Reuse architectural patterns across applications
  - Consistency between models, documentation and code
- Integration of systems and software disciplines
  - Raise the productivity
  - Shorten the development time
  - Increase product quality
  - Increase flexibility in building systems

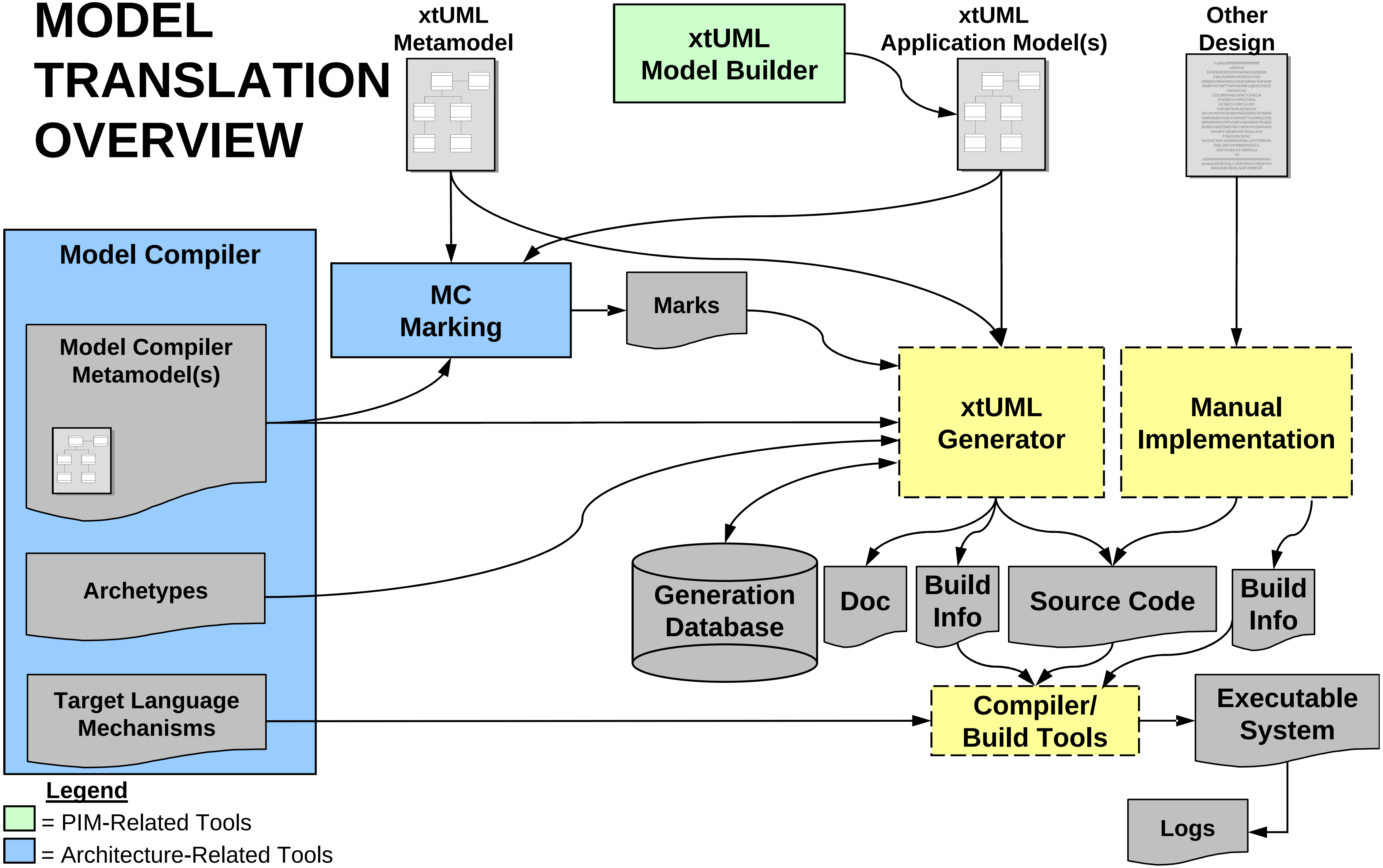
Courtesy of Saab Dynamics

USING xtUML

- xtUML is a subset of UML, extended with semantics
- xtUML models can be executed
- Platform independent models
- Open translation process – model compiler
- Application/model independent model compiler
- Model compiler independent models – fundamental for reuse and portability
- Reduce redundant documentation
- Based on meta-models

- Compatibility/transition, other standards

MODEL TRANSLATION OVERVIEW



xtUML Metamodel

xtUML Model Builder

xtUML Application Model(s)

Other Design

Model Compiler

Model Compiler Metamodel(s)

Archetypes

Target Language Mechanisms

MC Marking

Marks

xtUML Generator

Manual Implementation

Generation Database

Doc

Build Info

Source Code

Build Info

Compiler/ Build Tools

Executable System

Logs

Legend

🟩 = PIM-Related Tools

🟦 = Architecture-Related Tools

🟨 = Implementation and Build-Related Tools/Activities

Courtesy of Saab Dynamics

EXPERIENCES

EXPERIENCES - MODELLING

- The model shall contain a limited set of rules and shall be unambiguous
  - Capture rules in classes and associations (graphical) models instead of action code
- Raise the level of abstraction
  - Do not think code, not bit and bytes – Function oriented
  - The model might be instantiated for a specific system
  - The graphical view of the model shall be clear, readable and articulate, reduce need to read the actions of the model to understand the functionality
- A model is a solution to a problem or a family of problems
- Functional oriented interfaces instead of interface oriented functions
- Easier to perform big changes (function, architecture)
- Hard for traditional programmers to specify articulate models and to raise the abstraction level

EXPERIENCES – MODEL ENABLES

- The “cool” and “hard to understand” code solutions are removed from design. Efficient code will be used by the MC and in the architecture
- Consistency check of specification/model
- Seamless handover between SE and SW development
- Functions are disconnected from their implementation
- A view to discuss problems and their solutions
  - when you modify something you modify the solution
- Execution and test from specification down to implementation
- Defects in specification/design is found early in the process
- Earlier design decisions and identification of missing requirements

EXPERIENCES - ORGANIZATION

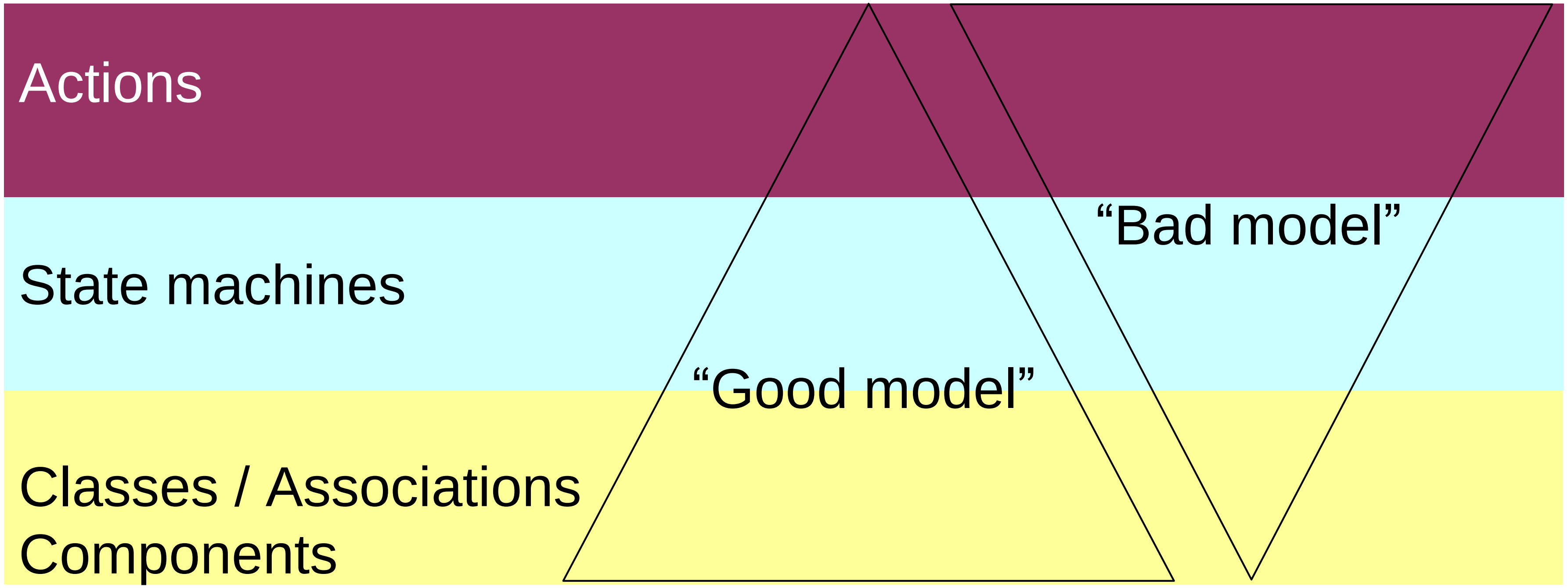
- Need to get cross-discipline and cross-functional teams and work process
- Use different modelling/development methods for different subject matters, within the same system/product. Let the methods use each other
- Big difference to base a project on a model based method compared to using model based methods of subsystems/applications within a project.
- Find the balance between model, legacy, build, test …
- Patience, it takes time. One (modeller) need to “fail” and see a better approach to understand
- Remember that the MC should be adapted for each project

EXPERIENCES - PROCESS

- Verification is still at object code level. Reuse of specification/model tests
- Functional review at model level
- Safety review at source code level
- Still lots of requirements in traditional documents, requirements at low level in model
- Parallel development

SUCCEEDING IN xtUML-MODELING
Creating Readable and Interpretable Models

Modeling Phase (Captured Rules and Logics)



SAABGROUP.COM