Introduction to Model Driven Engineering using Eclipse Frameworks

Bruce Trask
Angel Roman
MDE Systems
Agenda

• Part I
  – What is Model Driven Engineering

• Part II
  – MDE, Eclipse in Action, Live Demo
  – Questions and Answers
Warning

The live demo in this tutorial is real and contains pictures of coding done without model driven development. Some of the scenes you will see may not be appropriate for children under the age of 13.
Complexity and Change
Abstraction Gap

Requirements
Software Engineering/Computer Science

- Raising the level of Abstraction
- Introducing layers of indirection to solve problems

1 Dennis DeBruler, Refactoring, Fowler et. al. Addison Wesley
Language and Platform

Components
Frameworks
Middleware
Libraries
APIs
OS
HW

Change

Complexity

TEAM

C++/Java
Assembly Language
OpCodes
Language and Platform – where we are now

Complexity

Change

TEAM

Components
Frameworks
Middleware
Libraries
APIs
OS
HW

Classic Product Line
Assets

C++/Java
Assembly Language
OpCodes

© Copyright MDE Systems 2008
Domain Specific vs. Domain Independent Code
Duplicate vs. Unique Code
What’s underneath the problems

• Language technology has not kept pace with platform technology\(^1\)
• Insufficient *linguistic power* to tackle platform and domain complexity
• Lack of *tools* to deal with increased complexity

1 Douglas C. Schmidt IEEE Computer Magazine February 2006
Solution

• Leverage recent critical innovations to provide a quantum leap of language technology and tools to overcome the complexity gap

• Eclipse provides the necessary tools to make this happen
Models - participants
Abstract(ion)

• “Thought of as apart from concrete realities, specific objects, or actual instances

• “Expressing a quality or characteristic apart from any specific object or instance”¹

• Derivation: abs—away or away from, tract—to pull or draw¹—relate to domain/product

• “Doing more with less” [Unknown]

Abstraction - Language
Groups of Abstractions

• Groups of abstractions are used to tackle complex problems
• Abstractions have particular relationships and can be combined in meaningful ways\(^1\)
• New names, new relationships
• Creation of a *vocabulary* and grammar = language\(^1\)

\(^1\)Software Factories, Greenfield et al. Wiley 2004
Model Driven Development

• Program with Models
• Create Models that can be processed by a machined to produce lower level abstraction artifacts
• Language - Editor - Generator
Model Driven Development

The Anatomy and Pattern
Purpose

• **Language**
  – define the abstract syntax, or grammar, upon which the editor, serialization concrete syntax, and in-memory concrete syntax are based.

• **Editor**
  – The purpose of the editor is to render, capture and express *design intent*

• **Generator**
  – Provides automatic ways to get to lower levels of more executable forms of code.
  – Provides the actual meaning or semantics to the higher levels of abstractions.
Synonyms

• Model Driven Development
• Domain Specific Language
Language

Vocabulary + Grammar = Language
Metamodel

Object oriented representation of the grammar of a language
Sidebar
Non Software Example “Domain Independent”
Non Software Example
“Domain Specific”
Non Software Example
“Domain Specific”
Domain Specific Language

- Model concepts found in a specific domain
- Contrast to General Purpose Language – encodes generic abstractions
Editor

- Textual – Syntax and highlighting, intellisense
- Graphical – Domain Specific Graphics
- Meant for the human
- Nominated to a first class MDD participant
- Constraints at modeling time
- See the design
- Program in terms of domain elements and relationships
- Declarative
Generator

- Where the rubber meets the road in MDD
- Correct by construction
- Best Practice and Design Pattern replication
- Programming Power
- Maintainability – two sides of the same coin when bug occurs
- Portability of design
- Opens the world up to new programmers
Framework

- Abstract architectures and classes that provide a partial solution to a family of problems.
DSLs, Models and Frameworks
Making it actually happen
Language Workbench

- Workbench Generator Framework
- Workbench Debugging Framework
- Workbench Graphical Modeling Framework
- Workbench Graphical Editor Framework
- Workbench Transformation Framework
- Workbench Constraint Framework
- Workbench Modeling Framework
- Workbench Testing Framework
Eclipse¹

http://www.martinfowler.com/articles/languageWorkbench.html
Eclipse – Language Workbench
Tasks
Frameworks and Projects
Eclipse as a Model Driven Development Engine

- Eclipse Modeling Framework
- Java Emitter Templates
- Eclipse Testing Framework
- Eclipse Debugging Framework
- Eclipse Constraint Framework
- Eclipse Transformation Framework (ATL)
- Domain Specific Generators
- Domain Specific Debuggers
- Graphical Modeling Framework
- Graphical Editor Framework
- Domain Specific Models
- Domain Specific Graphics
- Domain Specific Transformations
- Domain Specific Test
- Domain Specific Constraints
- Domain Specific Graphics Mapping
Part 2

Model Driven Development, Software Product Lines and Eclipse

in Action

Live Demo