Building an Application with EMF Models

S. A. Robenalt
steve@webcircuit.com
The Problem...

- Build an application to support operation of a manufacturing device
  - Control pressure, temperature, position, time
  - Execute a not-yet-finalized process
  - Monitor and control variables to be defined
- Use PLC, GPIB, Serial, USB interfaces
- Timing of sequences is critical
  - Target a 10 ms cycle time
Why use EMF?

• Take advantage of the EMF class libraries
  – XMI-based persistence of models is a key feature
• Generated code -> rapid iterative prototypes
  – This leaves room to experiment until model is right
• Low barrier to entry (e.g. Library model)
• Plenty of room for enhancements
  – Query, Validation, Transactions; OCL; GMF
Where to start...

• 5 Models, each for an aspect of the system
  – Develop each model in isolation (minimize coupling)
• Facility – machine, sensors, actuators, material
• Recipe – input variables for the process
• Process – sequences of steps to follow
• Operations – results of sequences and process
• Visualization – realtime display and controls
Complications...

• Late/unfinished definition of process
  – Resulted in a more abstract design (a good thing)

• Short development cycles
  – 1-2 weeks apart (focus on most-needed features)

• Frequent change of focus
  – Client requires new feature to meet this week's goal

• Geography
  – Client is remote, machines firewalled
What about EMF?

- EMF allows rapid development of the models
  - Generated code allows experimentation
  - Able to simulate operation and refine quickly

- EMF facilitates a declarative process model
  - Sequences of primitive steps to define a process
  - XMI model persistence allows customer to redefine processes on-site with text editor
  - Many new requirements met with existing primitives
  - More complicated requirements – new primitives
Some Primitive Steps

- **Simple Steps**
  - Synchronous Read/Write - of one or more variables
  - Timed Wait - for known period
  - Event Wait - for external signal

- **Complex Steps**
  - Condition Wait – until variable enters range
  - Retry - repeat portion of sequence if condition fails
  - Computed Wait – for calculation using variables
Example Sequences

Prepare Sequence
- SyncWrite WaterOn=1
- CondWait WaterLevel=1L ±0.05, 300 sec
- SyncWrite WaterTemp=95
- CondWait WaterTemp=95 ±2, 180 sec
- SyncWrite ReadyInd=1

Main Sequence
- SyncWrite HeaterPwr=1
- SyncWrite WaterFeed=1
- TimedWait 300 sec
- SyncWrite WaterFeed=0

Idle Sequence
- SyncWrite HeaterPwr=0
- SyncWrite WaterTemp=65

Abort Sequence
- SyncWrite WaterFeed=0
- SyncWrite HeaterPwr=0
A Runtime for Realtime Execution

- Execute a series of sequences
  - Each sequence consists of a set of primitive steps
  - Start time of each step is carefully controlled
  - Java 5 Concurrency constructs for threading

- Simple data binding model
  - Links control variables in recipe to step variables

- Monitor threads for sampling of variables values
  - Compute statistics, drive realtime graphical displays
The Real Benefit of EMF

- Use GMF to define a Graphical Editor
  - Sequence diagrams
  - Visualization layout
- Use OCL to specify Model Constraints
  - Validate recipes and sequences before a run fails
- Query, Validation, Transactions for Editing
  - Update all recipe values matching criteria
- Teneo for Model Persistence
A Minor Problem

- Collision of differing objects with same id field
  - Monitor (samples variables periodically)
  - Target (Siemens or GE PLC)
  - Both used integer id field with value of 1
  - Resulted in ClassCastException when one was substituted for the other

- One solution was to use GUIDs (ugly)

- Refactored to use names instead of ints
  - Enforced a naming convention to prevent collision
Future Direction

- Enhance code for Realtime Process Control
  - Support a wider variety of process types
  - Support parallel sequences (fork/join)
- Flesh out Hardware Interfaces
  - OSGi Declarative Services
  - Wider variety of interface types
- Additional models
  - Templates, Expressions, and Events
- Expand usage into Realtime simulations
Questions?

Thanks for your time!