Complex Generations with EGF

Benoît Langlois / Thales Global Services
Agenda

1. Presentation
2. Demo

Eclipse (EMFT) EGF | © 2011 by Thales; made available under the EPL v1.0
Agenda

- Introduction
- Needs
- EGF Solution
- Summary
- Demo
A global company with 68,000 employees and €13.1 billion in revenues

- We help our customers to:
  - Provide reliable and secure solutions
  - Monitor and control
  - Protect and defend
- In two major sectors

Aerospace and Transport 40%
Defence and Security 60%

Thales: a reliable, long-term partner with operations in 50 countries
Input → Generator → Output

model
Integration of heterogeneous kinds of know-how

Different types of input
Orchestration
Different types of output

Input → Generator → Output

Different languages & Tools

model
File
Plug-in

Variability

Framework

Different types of output

But…
How to deal with Complex Generations?
What Drivers?

- Generation scope?
- Generation reusability?
- Generation customization?
- Variability? Product lines?
- Generation orchestration?
- Generation workflow?
- Generation data, which ones, where?
- Combining [model|text|dsl]-to-[Model|text|dsl]?
- Multiplicity of languages and engines?
- Integration of a new language?
- How to develop & test?
- Executability? Distribution?
- What target-platform?
- Performance, scalability?
- One-click generation solution?
- Best practices, guidance?
- Update strategy of the produced artifacts?
- Merging Generation?
One Solution
To deal with Complex Generations
Integrated, Extensible
Generation framework
**EGF (Eclipse Generation Factories)**

is an EMFT Component project (incubation)

**Integrated to Eclipse Indigo and Juno**
EGF (Eclipse Generation Factories) is an EMFT Component project (incubation)

Integrated to Eclipse Indigo and Juno

10,000 downloads since the Indigo release (June 2011)
What is EGF?

- Complex Generation
- Extensible Framework
- Generation Portfolio
Agenda

- Presentation
  - Introduction
  - Needs
  - EGF Solution
  - Summary
  - Demo

Eclipse (EMFT) EGF | © 2011 by Thales; made available under the EPL v1.0
Complex Generations – Need #1

Generation in series

Need

Ability to endlessly repeat the same generation
Generation in series

**Solution**

Using transformation engines, such as model-to-text tools (e.g., Acceleo, Jet, xPand)
Complex Generations – Need #1

Generation in series

1. Invoking model-to-text tools
   - Acceleo
   - Jet
   - ...

2. Using EGF Patterns
   - Pattern
Generation in series

Pattern

Systematic behavior expressed in a language (e.g., Java, Jet)
Systematic behavior expressed in a language (e.g., Java, Jet)
Generation in series

Systematic behavior expressed in a language (e.g., Java, Jet)

Pattern

Pattern Combination (e.g., call, inheritance, substitution)
Complex Generations – Need #2

Factory Component

Need

Ability to represent a generation as a component
Complex Generations – Need #2

Factory Component

- Encapsulation
- Interface

Contract container

Contract
Factory Component

Substitution

Version 2
Generation Composition

Need

Ability to compose generations
Complex Generations – Need #3

Generation Composition

- Activity
- Factory Component
- Task
- Composite
- Leaf
Generation Composition

Activity

Factory Component

Composite

Task

Leaf

Implementation
Complex Generations – Need #3

Generation Composition

- Activity
  - Factory Component
    - Composite
  - Task
    - Leaf
    - Implementation
      - In a language

Languages: Java, Jython, Ruby, Acceleo, Jet, ATL

Eclipse (EMFT) EGF © 2011 by Thales; made available under the EPL v1.0
Complex Generations – Need #3

Generation Composition

Activity

Factory Component

Task

Implementation

Composite

Leaf

For a formalism

EGF Pattern

Eclipse Generation Factories

Thales
Generation Orchestration

**Need**

Ability to conduct a generation
Heterogeneity

Need

Ability to apply heterogeneous languages and tools during a generation
Composition

Composition in a workflow

Factory Component -> Production Plan (Activity workflow)

Data exchange between heterogeneous activities

Language Task: Java, Jython, Ruby

Tool Task: Jet, Acceleo, ATL

Composite Activity Invocation

Factory Component
Complex Generations – Need #4

Composition

Orchestration

Generation Steps

Factory Component → Production Plan (Activity workflow) → Data exchange between heterogeneous activities → Factory Component

Language Task: Java, Jython, Ruby

Tool Task: Jet, Acceleo, ATL

Composite Activity Invocation

Only Sequential Today

Eclipse (EMFT) EGF © 2011 by Thales; made available under the EPL v1.0
Composition

Orchestration

Heterogeneity

Diversity

Data exchange between heterogeneous activities

Factory Component

Production Plan (Activity workflow)

Language Task

Java

Jython

Ruby

Tool Task

Jet

Acceleo

ATL

Composite Activity Invocation
Generation Portfolio

Need

Ability to provide Factories Off-The-Shelf (FOTS)
Complex Generations – Need #6

Generation Portfolio

Factory Component

Composition

Factory Component

Factory Component

Tool Task

Composition

Tool Task

Language Task
Generation Portfolio

Individual Activities

- Factory Component
- Composition
- Factory Component
- Tool Task
- Composition
- Tool Task
- Language Task

Portfolio

Eclipse (EMFT) EGF | © 2011 by Thales; made available under the EPL v1.0
Generation Portfolio

Portfolio focuses on a Generation Topic

Eclipse (EMFT) EGF | © 2011 by Thales; made available under the EPL v1.0
Generation Portfolio

Portfolio focuses on a Generation Topic

Portfolio = Set of Factory Components
Generation Portfolio

Portfolio focuses on a Generation Topic

Portfolio = Set of Factory Components

Simple to Sophisticated Generation Portfolios

Eclipse (EMFT) EGF | © 2011 by Thales; made available under the EPL v1.0
Complex Generations – Need #6

Generation Portfolio

Enabler for Common
development Practice

Portfolio

Portfolio Developer
Developer
End User
Complex Generations – Need #6

Generation Portfolio

Provided With EGF

Enhanced EMF Generation

Build modeler and generator
For continuous integration
(Hudson/Jenkins, Buckminster today)
Generation Variants

Need

Ability to support generation variants
Generation Variants

Variability model (for product lines), not supported yet
Complex Generations – Need #7

Generation Variants

Example

Code/textual generation for my organization

Specific generation for my project

Several levels of Customization
Generation Variants

Supported by EGF Patterns
With the pattern substitution mechanism

Several levels of Customization
Generation DSL

Need

Ability to describe generation with a DSL
Generation DSL

Generation Chain

Definition of generation at a high level of description

Purpose: to easily define and maintain complex generations
Setting main features for each selected ecore models
Setting main features for each selected ecore models
Setting main features for each selected ecore models

A generation chain model is automatically created
Generation DSL – Applied to the EMF Generation

**Design**

Concrete Syntax

**DSL**

Abstract Syntax

**Graphical Representation**

**Textual Representation**

[Other Form of] Representation

---

**Generation Chain**
Generation DSL – Applied to the EMF Generation

Design

DSL

Concrete Syntax

Graphical Representation

Textual Representation

[Other Form of] Representation

Generation Chain

Abstract Syntax

Implementation

GenModel

Factories

Automated translation

Customization

Execution

Generated EMF Code
Generation DSL – Applied to the EMF Generation

1. **Design**
   - Abstract Syntax
   - Concrete Syntax
   - DSL

2. **In Eclipse**
   - GenModel
   - Factories
   - Execution

3. **Implementation**
   - Automated translation
   - Generated EMF Code

**Generation Chain**

---

**Eclipse (EMFT) EGF | © 2011 by Thales; made available under the EPL v1.0**
Input Diversity

**Need**

Ability to apply generations from various resource types
Input Diversity

Supported today with patterns (Jet, Java)

- Ecore model
- File
- Eclipse Workspace Domain
- Plug-in
  - Any Tree-structure artefact

Potentially...

Eclipse (EMFT) EGF | © 2011 by Thales; made available under the EPL v1.0
Generation Strategy

Need

Ability to apply different strategies of generation on the same resources
Generation Strategy

Pattern Strategy

Way to apply patterns over a resource
Complex Generations – Need #10

Generation Strategy

Controller

Pattern Strategy
Way to apply patterns and a resource together

View

Processing
List of patterns to be applied
For pattern execution, use of the language engine matching the pattern language, e.g. Jet, Java

Model

Resource
For each pattern, query over a resource, e.g. Model

Result

Eclipse (EMFT) EGF | © 2011 by Thales; made available under the EPL v1.0
Generation Strategy

Examples of Pattern Strategies

*Domain-driven pattern strategy*: in-depth navigation over a resource (e.g., model, file); for each model element, applying a set of patterns

*Pattern-driven strategy*: for each pattern, applying the pattern to each resource element
**Generation Strategy**

**Examples of Pattern Strategy Options**

*Resource visitor*: When navigating over a resource, the visitor function specifies how to continue this navigation. Example: considering the sub-classes of the current resource element.

*Reporter*: Specification of the responsible for reporting a model-to-text transformation.
Post Processing

In progress
EGF can be used in other concerns than Generation
Ex: Validation, Transformation

Portfolio & Reusability contribute to define Generation Framework

With maturity, “Factory” has more and more the meaning of “Automation”
Summary of the main EGF concepts

Factory Component
Composite generation unit with an activity workflow

Task
Leaf generation unit written in a language (e.g., Java, Ruby)

Portfolio
Capitalization on a specific generation topic

Generation Chain
High generation view to define complex generations

EGF Pattern
- Description of systematic behavior
- For definition of code generation families
Architecture

**EGF Portfolio**

<table>
<thead>
<tr>
<th><strong>EGF</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Engine Extensions</td>
</tr>
<tr>
<td>EGF Engine</td>
</tr>
</tbody>
</table>

*Eclipse (EMFT) EGF | © 2011 by Thales; made available under the EPL v1.0*
Extensibility

All metamodels are extensible
Possibility to define new Activity type for new paradigms
Possibility to add new pattern languages

Generally, costly at this level

Provides basic metamodels and behaviors to automate software production

EGF Metamodel
Basic behaviors, dynamic execution

Factory component, task

Pattern
Extensibility

Possibility to add new languages and tools as tasks
Possibility to add new pattern strategies

Meets specific software production needs
Language & tools interoperability
New types of generation formalisms
EGF Extensibility

EGF Portfolio

EGF

Engine Extensions

EGF Engine

Examples of Portfolios provided with EGF

Enhancement of the EMF Generation

Build modeler and generator

Portfolio = generation topic

Simple to off-the-shelf software factories

Extensibility

Generation redefinition with pattern substitution

Task supports inheritance, which enables property and behavior redefinition

Development of (new types of) portfolios
Project Organization

Host: Infrastructure, Policies, Process

Core Project: EGF

Additional Portfolios: Not EPL License

Community

Committer
Contributor
Owner
Committer
Contributor

Eclipse (EMFT) EGF | © 2011 by Thales; made available under the EPL v1.0
Maturity of Software Production

- Hand-crafted
- Automated
- Industrial

Variability

Complementarity & Assembly, e.g.
Generation / DSL, COTS/FOTS

Generation Framework
FOTS
Extensible Generation Framework
Integrated generation tool

Generation tool in silo

Hand-written Code
Download

by update site from Indigo
by update site from Amalgam

http://wiki.eclipse.org/EGF_Installation
General info about EGF

Project page: [http://www.eclipse.org/egf](http://www.eclipse.org/egf)
Twitter: @LangloisBenoit