THALES

EGF - Thales Corporate Services - EPM

EGF Tutorial
Reuse and Customization

Benoît Langlois – Thales/EPM

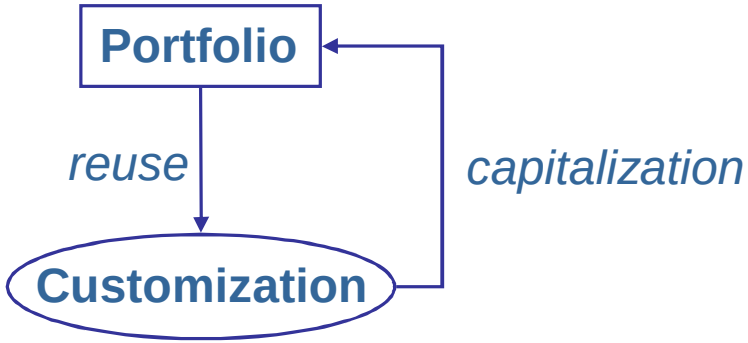
Agenda

- Introduction
- Pattern Extensibility

THALES

Need of Reuse and Customization

- **General needs:**
  - Need #1: Ability to **reuse a portfolio**, where a portfolio is a consistent set of off-the-shelf components
  - Need #2: Ability to **customize** an off-the-shelf component in order to fit specific project's expectations
  - Need #3: Ability to capitalize customized off-the-shelf components in order to reuse them as a **new portfolio**



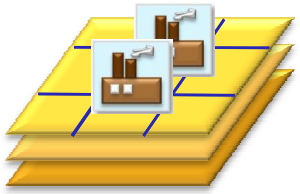
Customization in the EGF Context

- **EGF Vocabulary:**
  - Portfolio = consistent set of factory components
  - Factory component = factory component parameters + viewpoints + activity workflow
  - Pattern = formalism to express systematic behavior (Java and Jet supported today) executed by a pattern activity

- **Means of Customization with EGF :**
  - Means #1: **Parameterization** of factory component
  - Means #2: **Composition**: Creation of new factory component from factory components through a specific activity workflow
  - Means #3: **Pattern extension**: Ability to extend patterns from a reused factory component portfolio without any intrusion

Creation of Customized Portfolios

Initial Portfolio
Team #1

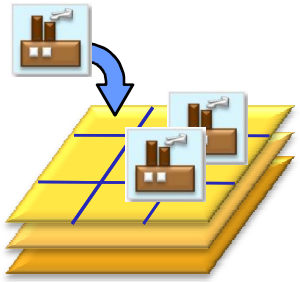


Portfolio

THALES

Creation of Customized Portfolios

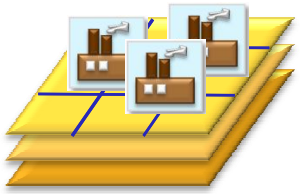
Capitalization



Portfolio

THALES

Creation of Customized Portfolios

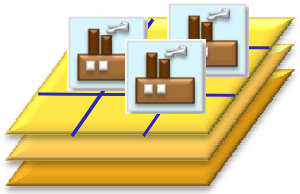


Portfolio

Off-The-Shelf solution

THALES

Creation of Customized Portfolios



Off-The-Shelf solution

Portfolio

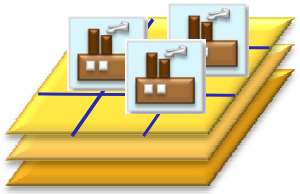
customization

Portfolio Adaptation Team #2

Variants

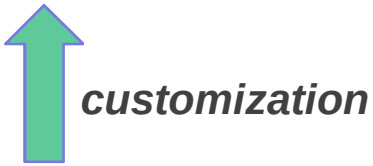
THALES

Creation of Customized Portfolios



Off-The-Shelf solution

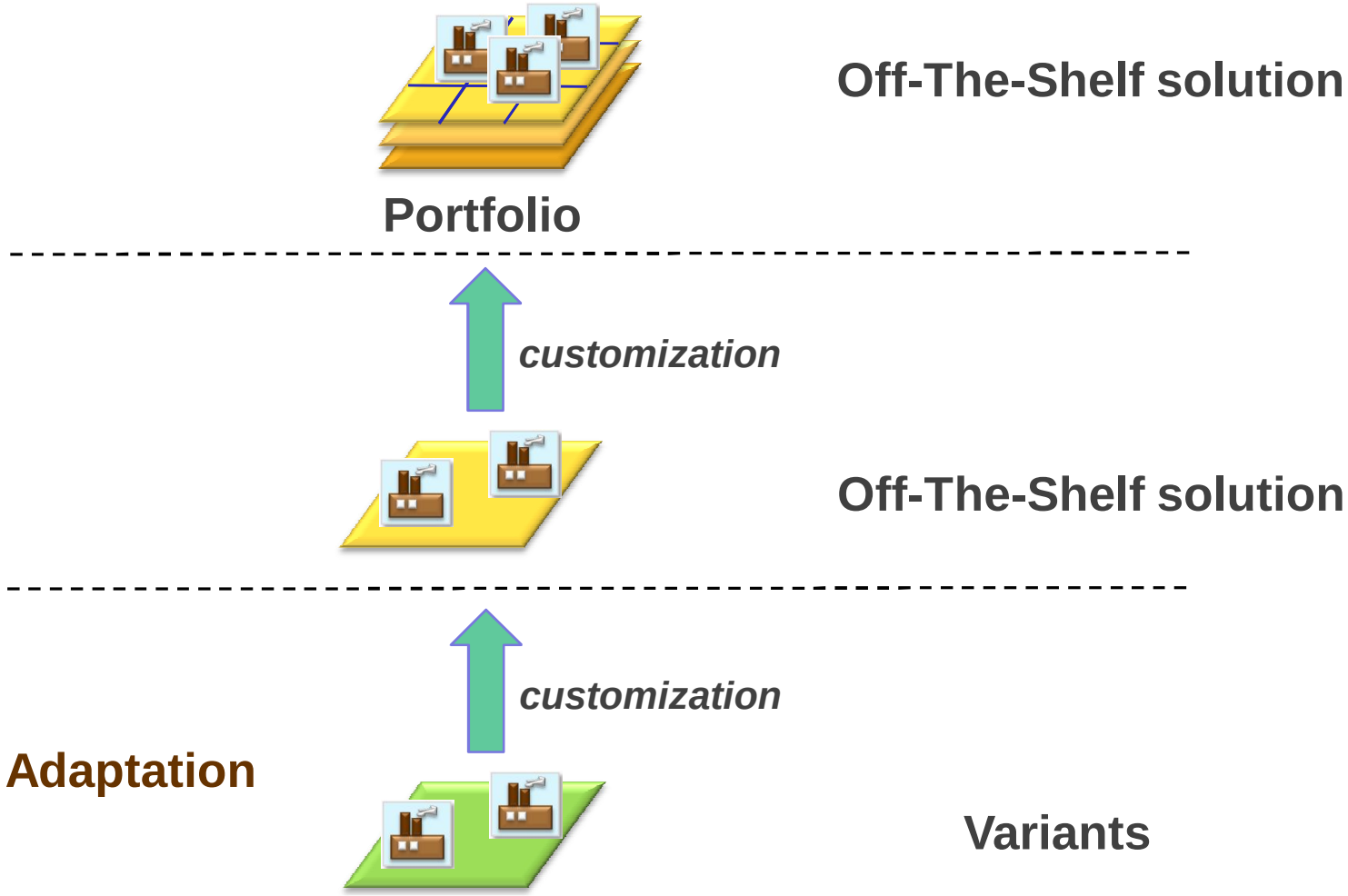
Portfolio



customization

Off-The-Shelf solution

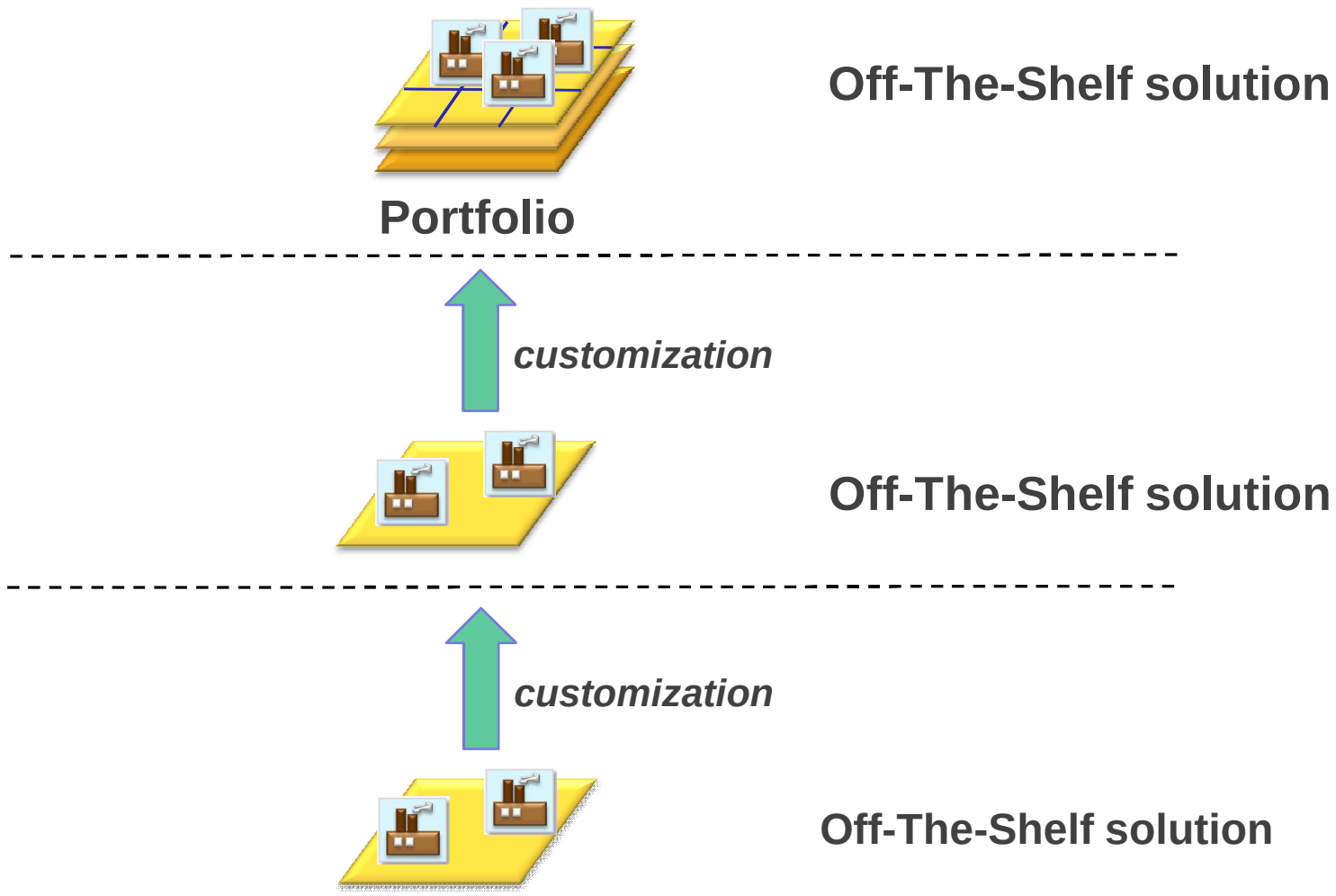
Creation of Customized Portfolios



Portfolio Adaptation Team #3

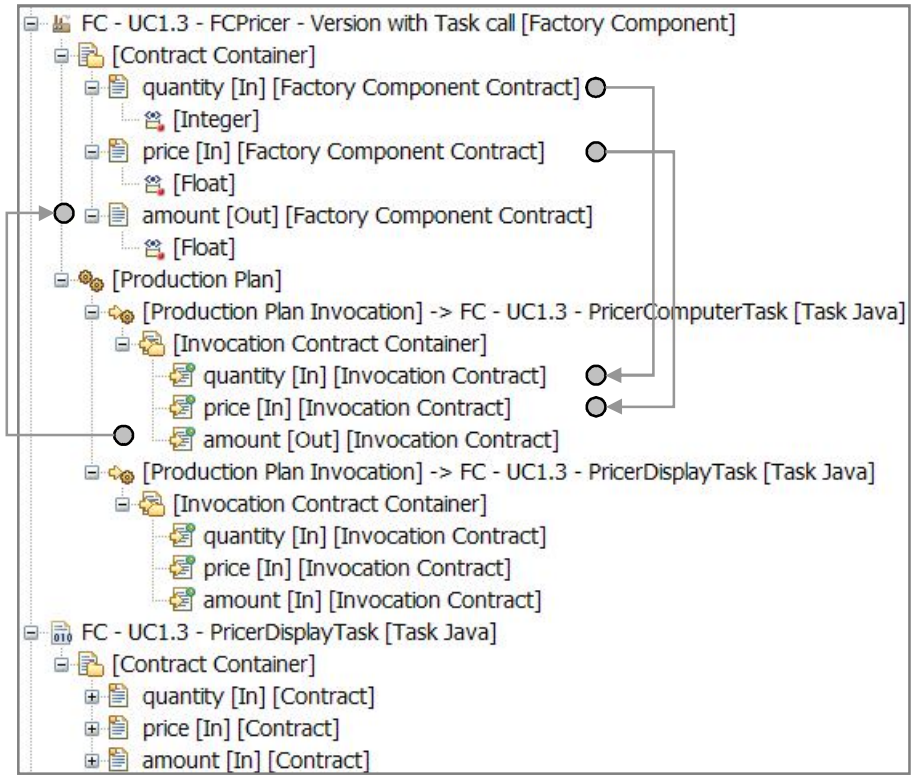
THALES

Creation of Customized Portfolios



THALES

Customization – Parameterization

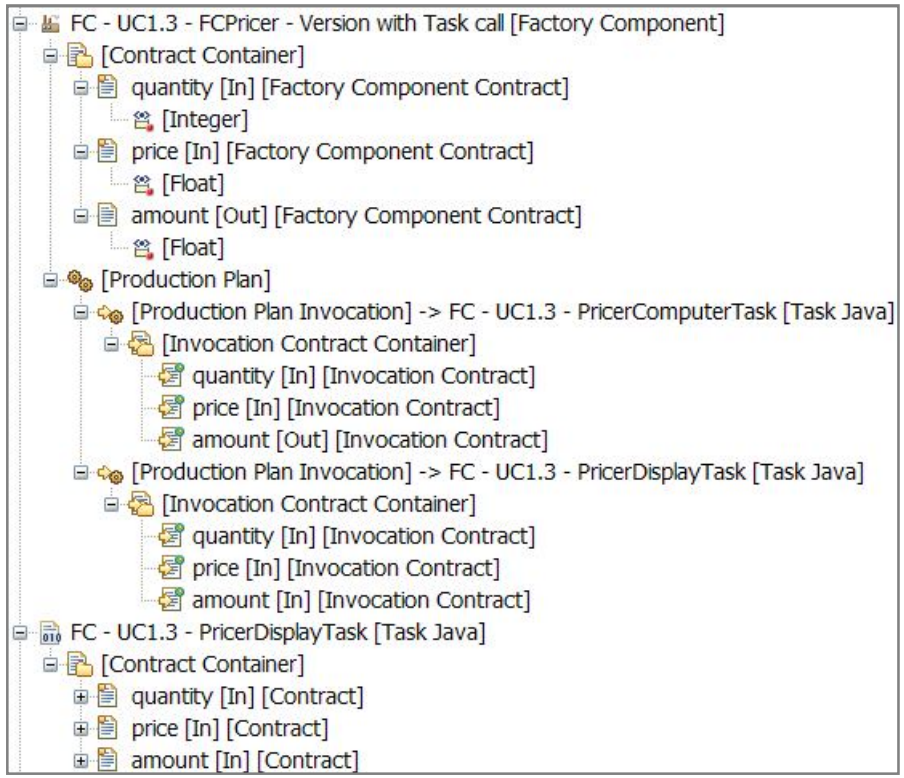


Parameters

Invocation with parameters passing for contextualization

THALES

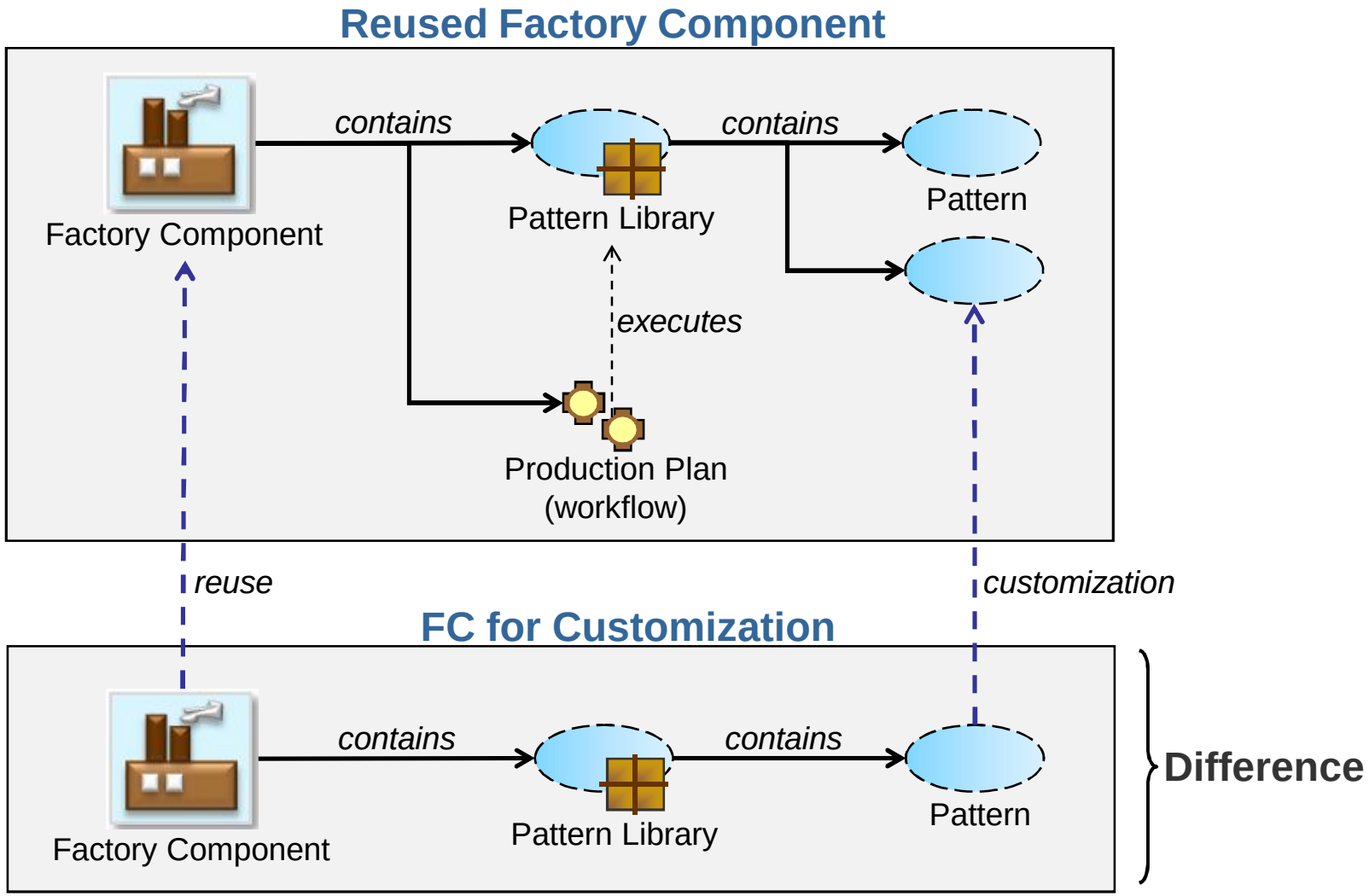
Customization – Composition



This factory component is the composition of activities (i.e., factory component, task) defined in an activity workflow

THALES

Customization – Pattern Extension



Effect: The new factory component has the same behavior than the reused factory component except the pattern customization

Agenda

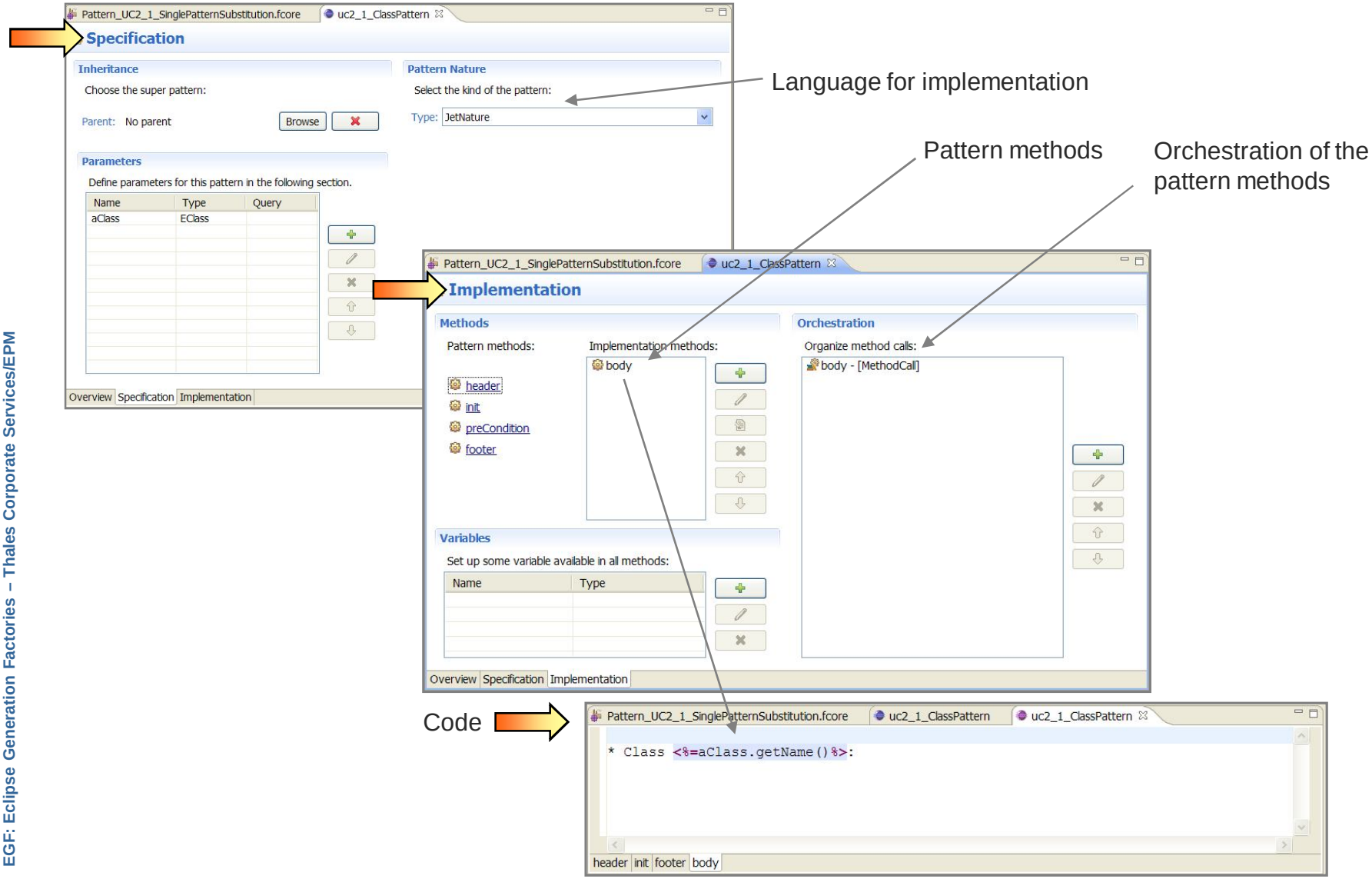
- Introduction
- Pattern Extensibility

THALES

- **Definition:**
  - Definition #1 – Rationale: A pattern is a solution to a recurrent problem
  - Definition #2 – Structural: A pattern is a formalism to express systematic behavior

- **Key points:**
  - Dissociating the specification (external view) from the implementation (internal view) of the behavior
  - Supporting multilingual patterns for the behavior implementation in order to use the best programming language for a given situation (e.g., programming language such as Java; M2T, M2M, T2M, T2T)

Pattern – Main Elements



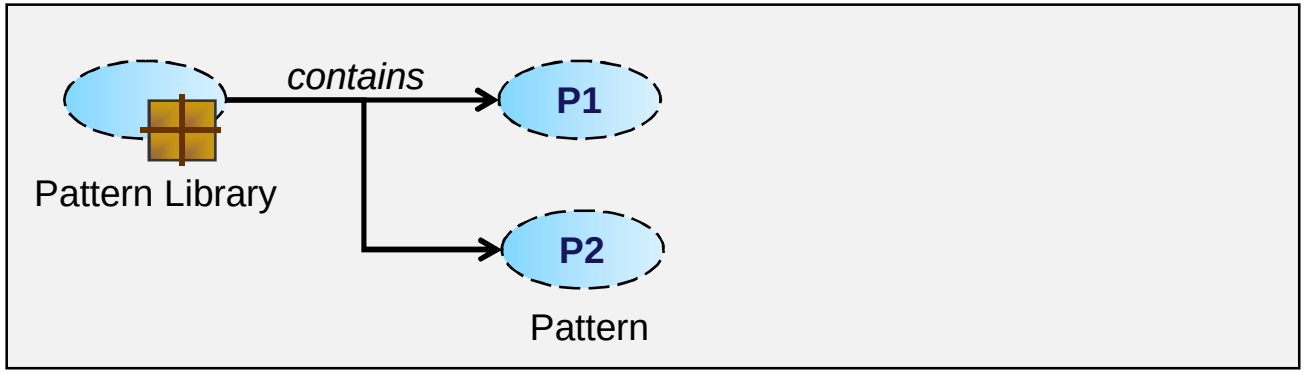
Pattern – Relationships

Relationship	Purpose	Defined in…
Inheritance	Inheriting properties from parent patterns	Specification View
Pattern Call	Behavioral delegation. In the orchestration, a pattern calls another pattern	Implementation View / Orchestration
Pattern Injection	Behavioral delegation. Determination of the call context at runtime	Implementation View / Orchestration
Multilingual Call	Property of the pattern call and pattern injection: a pattern implemented in a language calls a pattern implemented in another language	Implicit: depends on the pattern nature (implementation view) and the engine able to execute a pattern in a given language
Pattern Callback	Giving back the control to the pattern strategy which orchestrates the patterns to be applied over a resource (e.g., model)	Implementation View / Orchestration
Pattern Extension	Redefinition of pattern with a substitution mechanism	Activity invocation / substitution parameter values

Stage. Identification of Reused Factory Components

In the Reused Factory Component

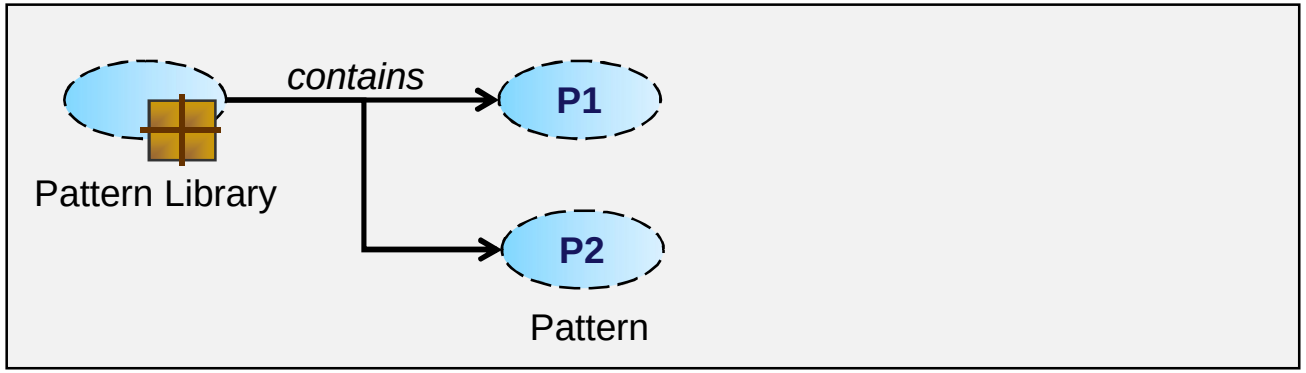
Initial Patterns



Stage. Pattern Definition – Customization Time

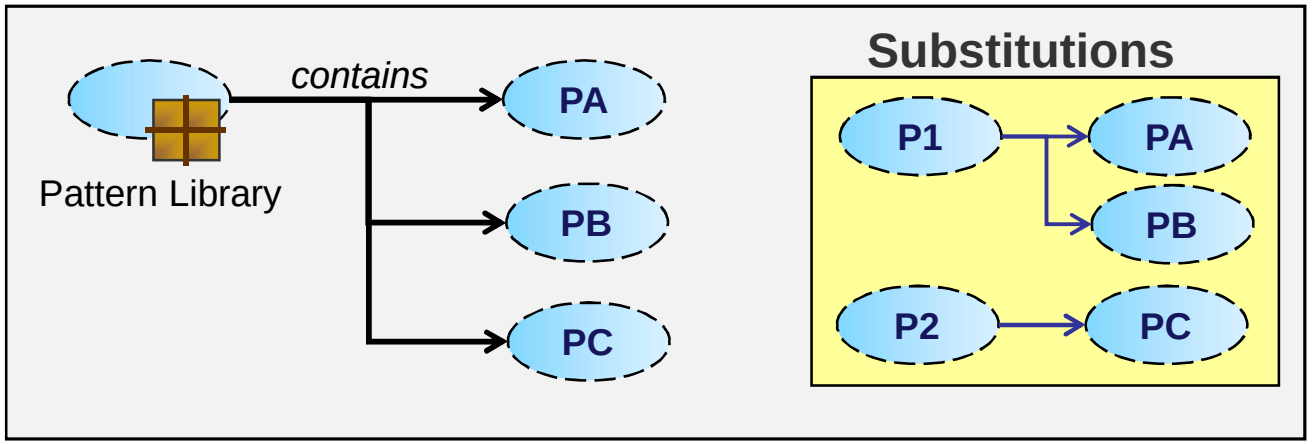
In the Reused Factory Component

Initial Patterns



In the Factory Component for Customization

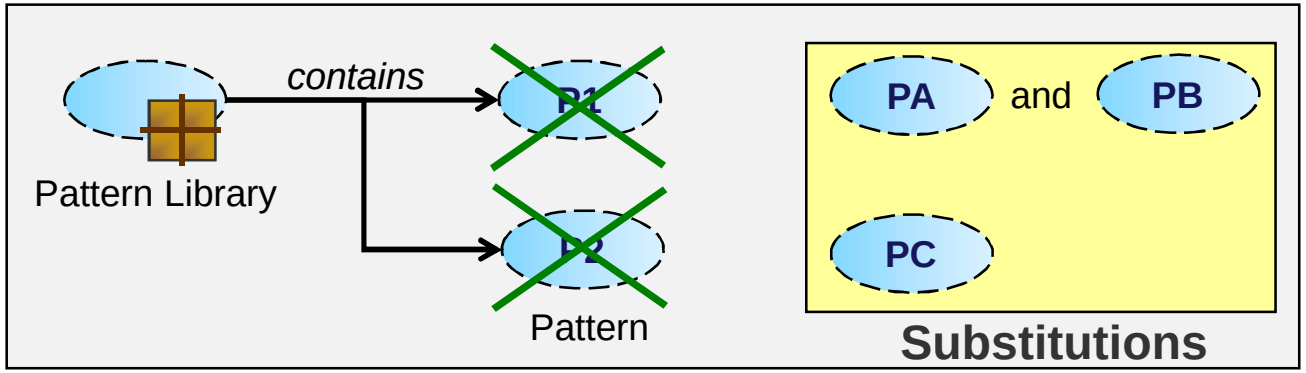
Patterns for substitution



Stage. At Runtime

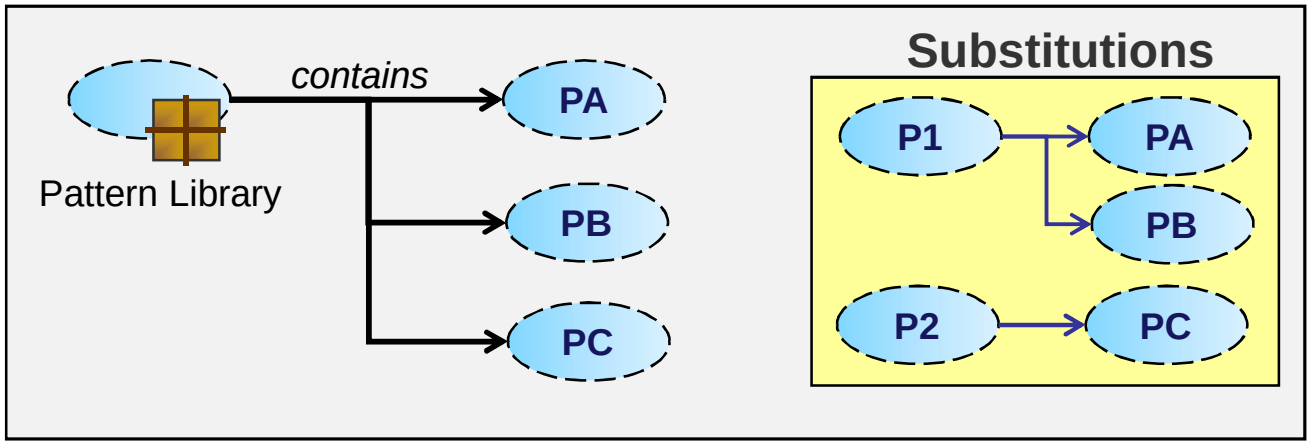
In the Reused Factory Component

Initial Patterns

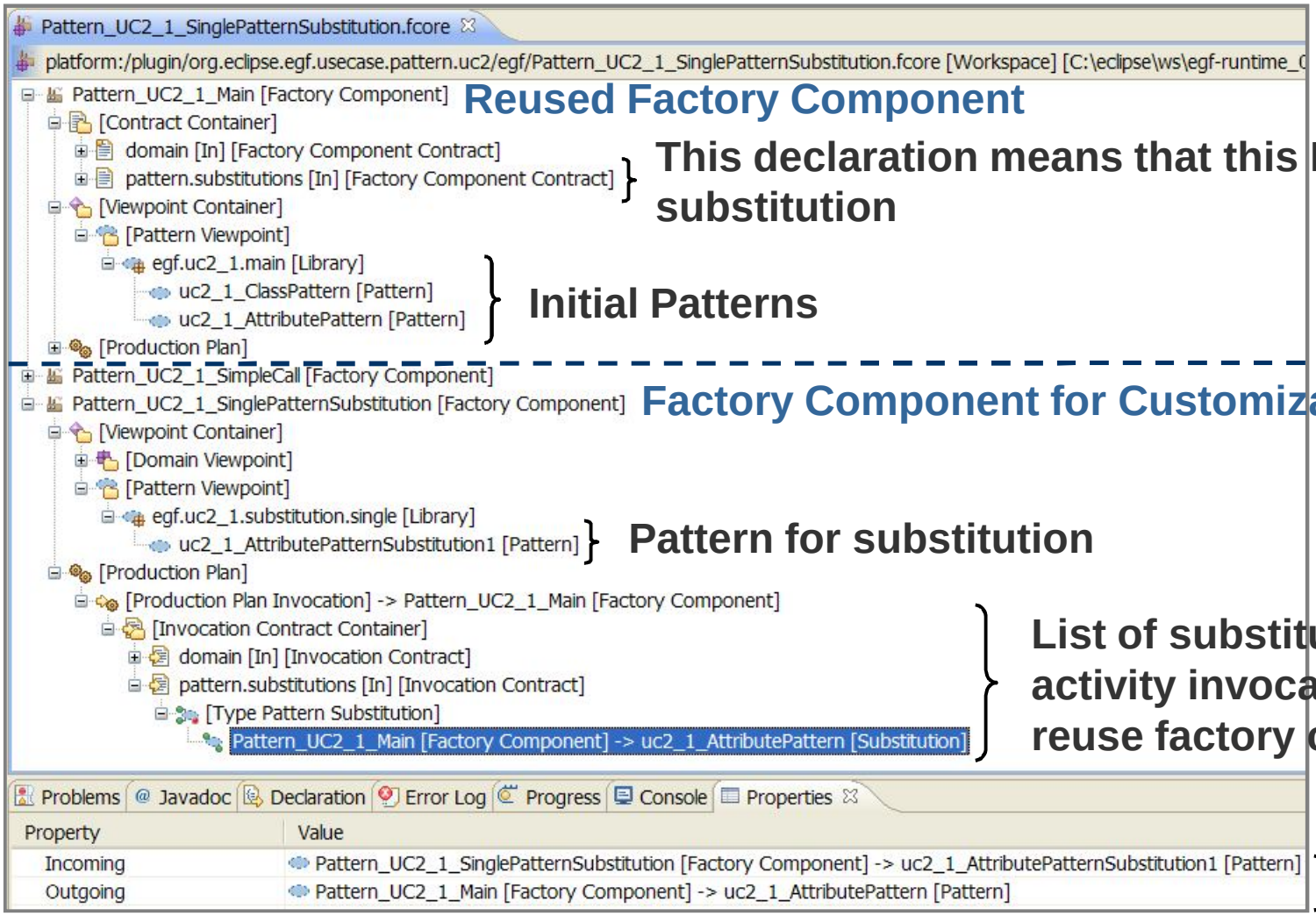


In the Factory Component for Customization

Patterns for substitution



Substitution – Notation



Reused Factory Component

This declaration means that this FC accepts substitution

Initial Patterns

Factory Component for Customization

Pattern for substitution

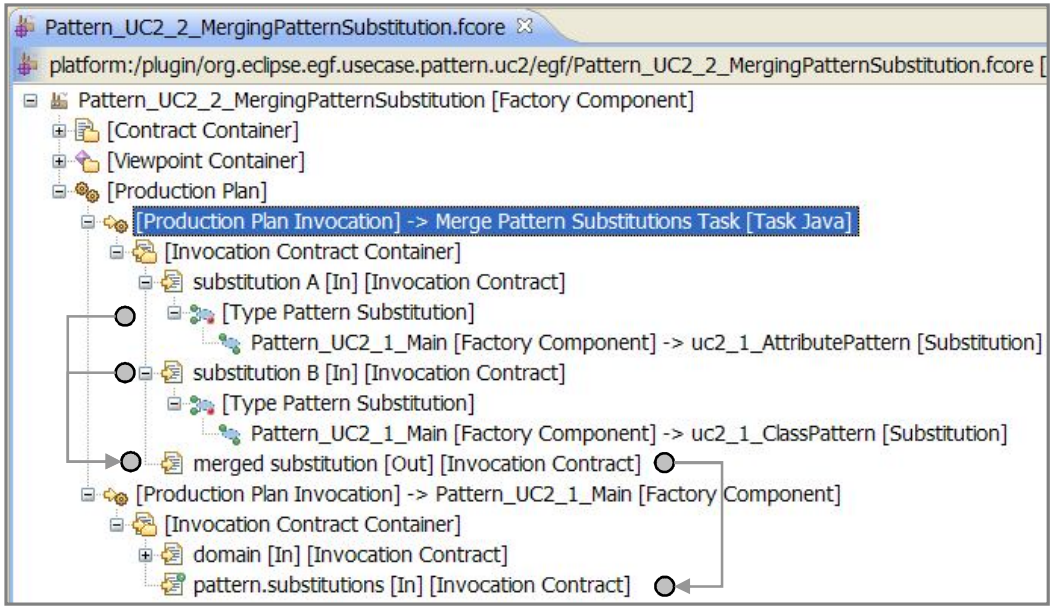
List of substitutions in an activity invocation of the reuse factory component

One substitution

THALES

Merge of Pattern Substitution

- When there is more than one substitution, a merge of pattern substitution is necessary
- A merge of pattern substitution consists in merging two lists of pattern substitution
- It is possible to merge pattern substitution in series



Two lists to be merged

Merge result