



www.thalesgroup.com

Contributive XHTML DocGen

Kitalpha – Capella Studio

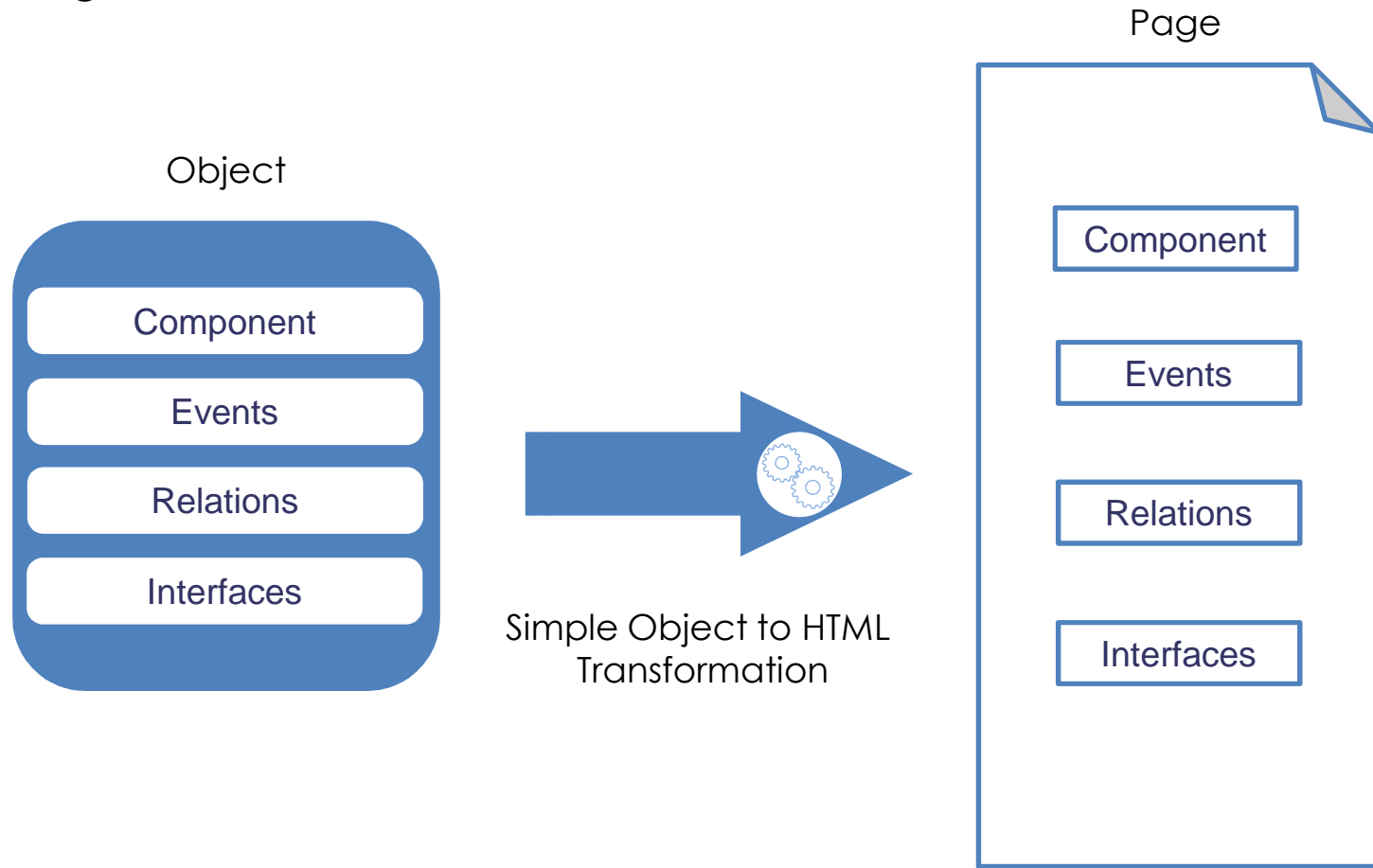
OPEN

THALES

Contributive XHTML DocGen

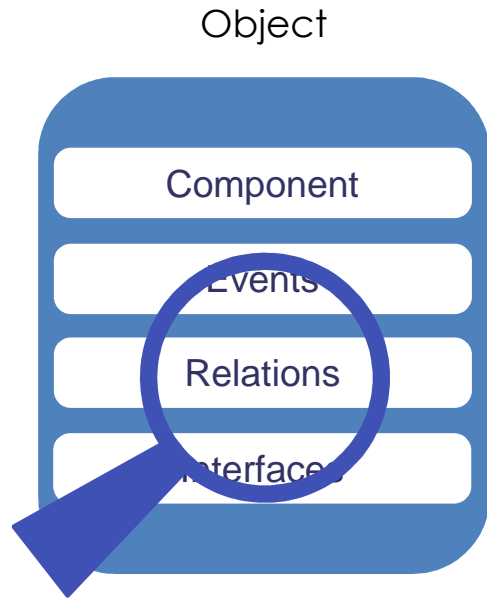
- ◆ General Concept
- ◆ Contributive XHTML DocGen for Ecore
- ◆ Contributive XHTML DocGen for Capella
- ◆ Contributive XHTML with Extension point
- ◆ Types of competition when multiples contributions
- ◆ Pattern substitution principle

XHTML Page Structure

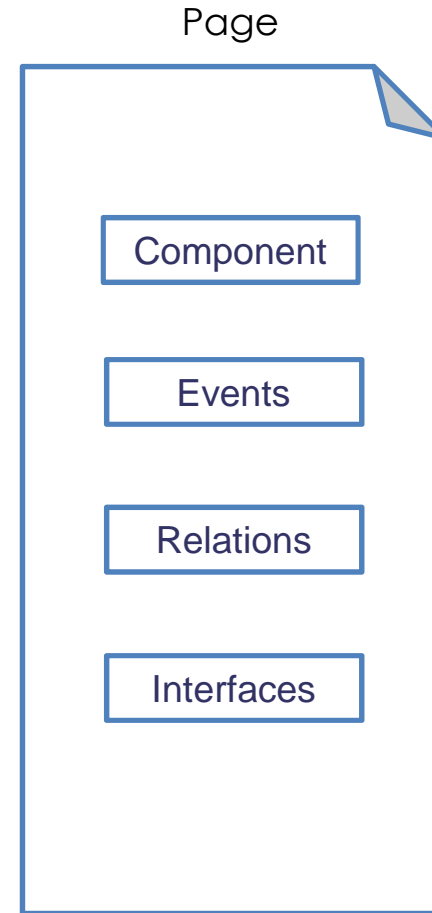


Piece of XHTML Page

XHTML Page Structure

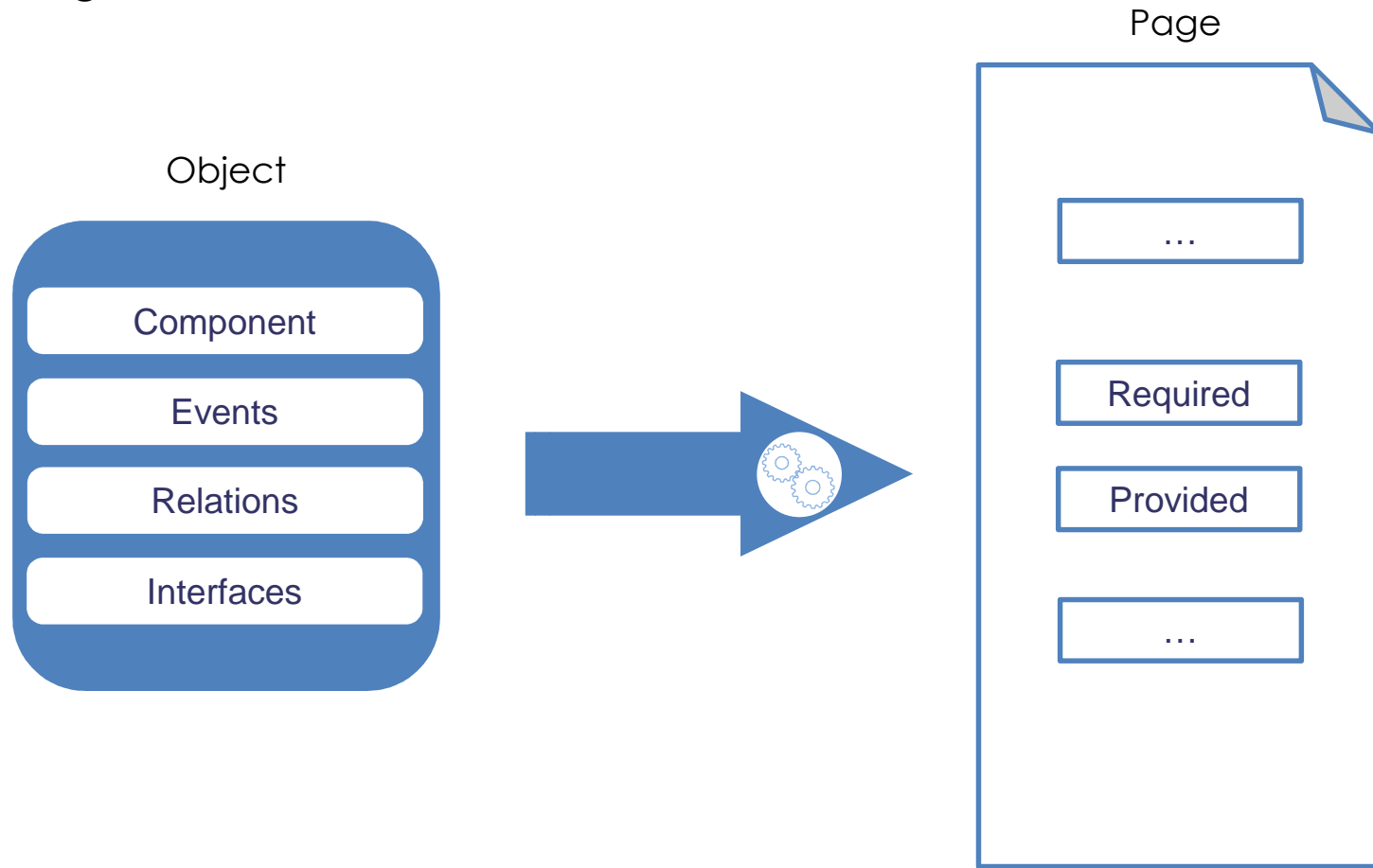


How to split Required & Provided References in generated page



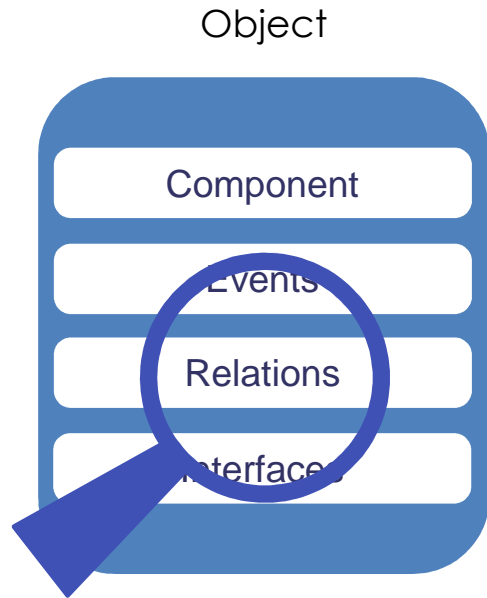
Piece of XHTML Page

XHTML Page Structure

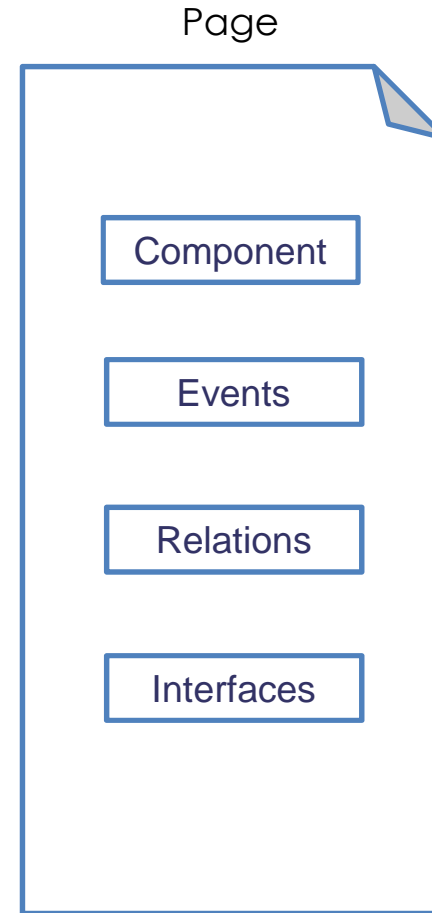


Piece of XHTML Page

XHTML Page Structure

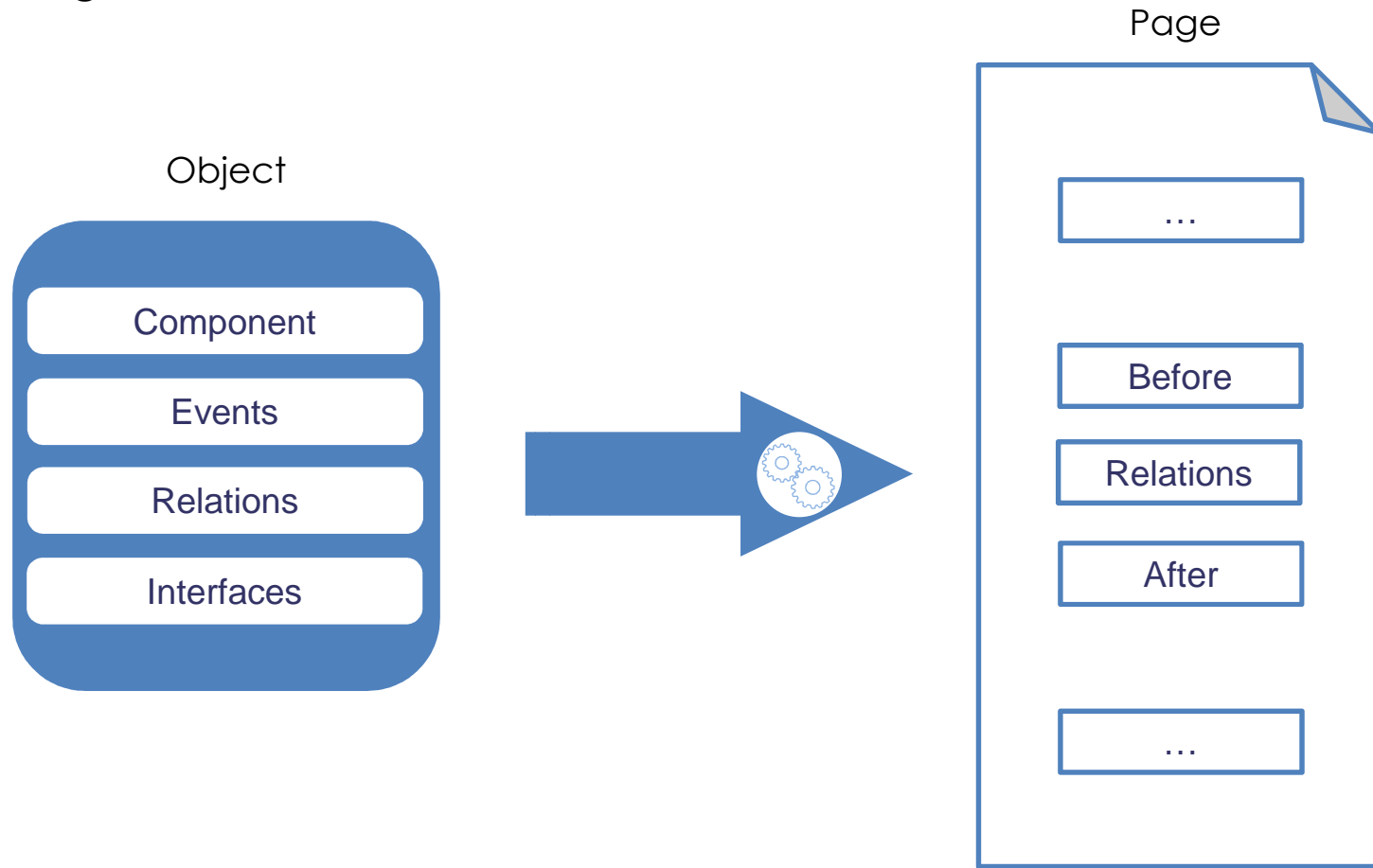


How to add information around relations in generated page



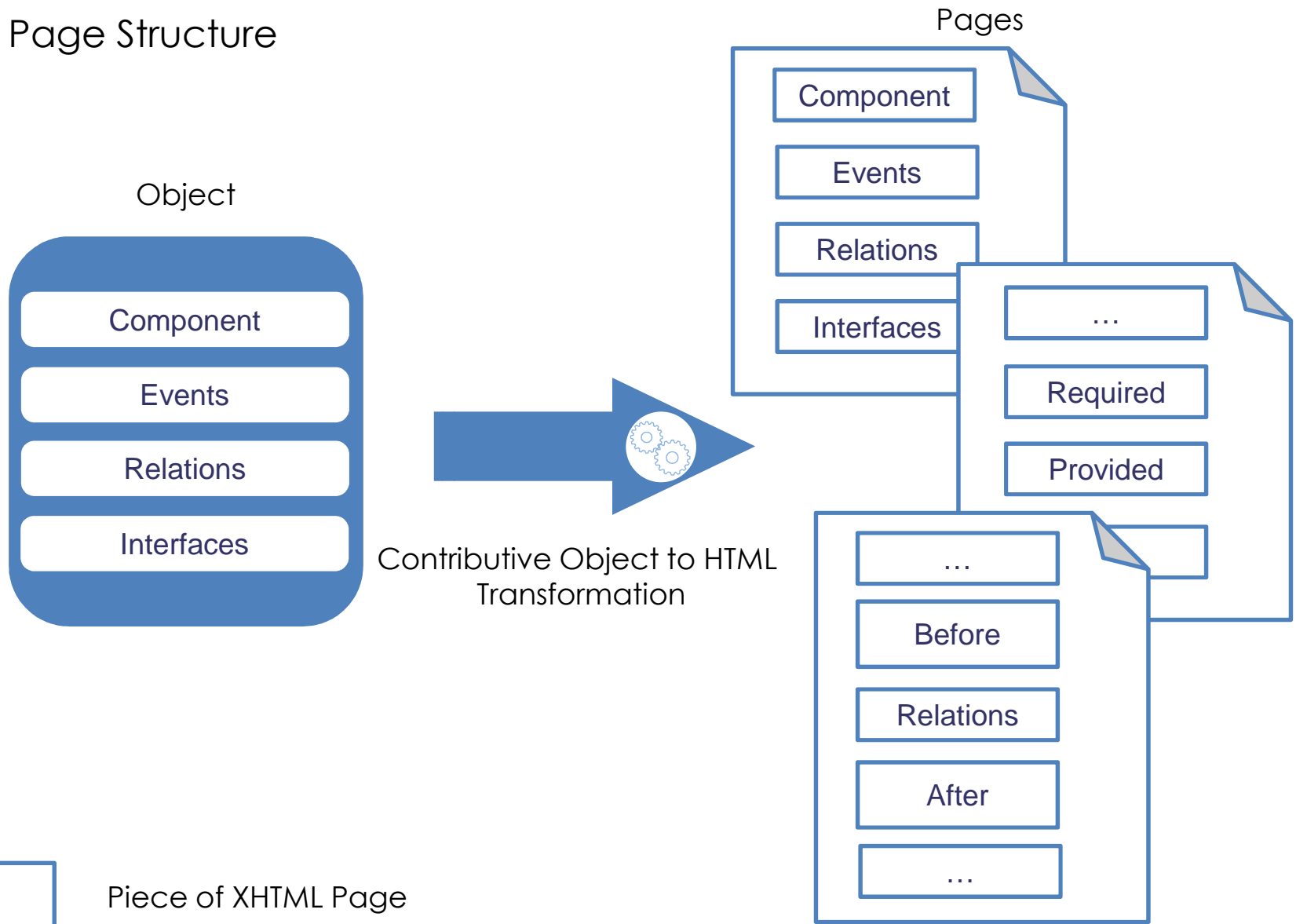
Piece of XHTML Page

XHTML Page Structure



Piece of XHTML Page

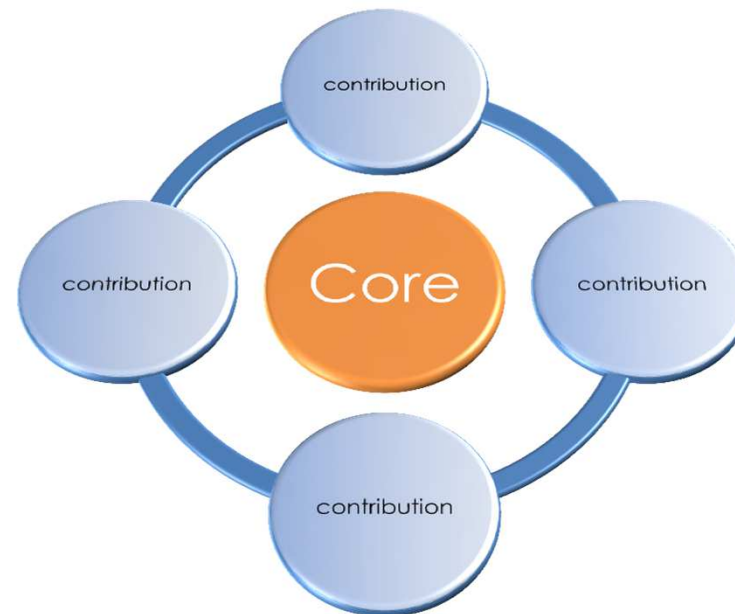
XHTML Page Structure



Piece of XHTML Page

Contribution

- Customize functionalities
- Remove functionalities



Contributive XHTML DocGen for Ecore

- ◆ Purpose
- ◆ Concept – Contributive XHTML DocGen based Pattern call
- ◆ Example: EDataTypes & EClass EAnnotation

Context

- Models are complex and Basic documentation is insufficient
- Existing Ecore XHTML DocGen is not contributive

Need

- Need to extend XHTML DocGen for customizing and adding or remove information in generated documentation from models

Objective

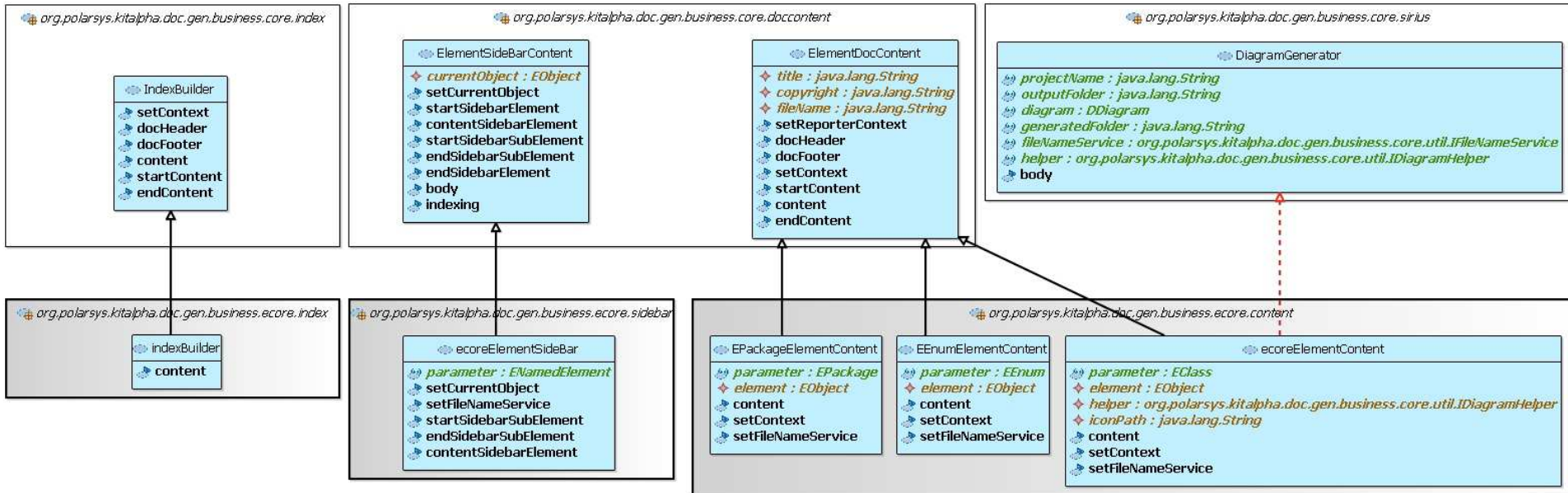
- Rework existing Ecore XHTML DocGen framework to be contributive

Objective



Action

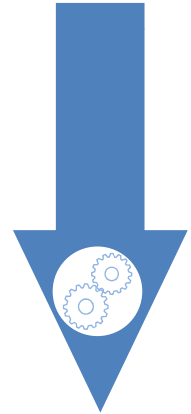
- Rework existing Ecore XHTML DocGen framework to be contributive by pattern calls and substitution mechanism
- Rework existing Ecore XHTML DocGen framework to add pattern call and substitution mechanism



sclosed to any third party

This document is not to be reproduced without the prior written permission of

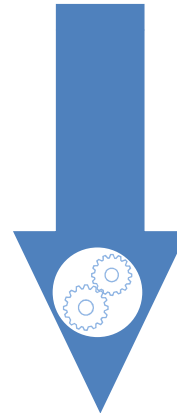
- platform:/resource/a/m/Example.ecore
 - Example
 - ClassA
 - Documentation
 - Type -> EClassA is Concept of Example.ecore
 - Goal -> To Show How Contributive GenDoc for Ecore works
 - attributeA : EByte
 - (.) EByte




Copyright © 2014 The Eclipse Foundation. All Rights Reserved.

OPEN

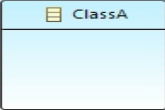
- platform:/resource/a/m/Example.ecore
 - Example
 - ClassA
 - Documentation
 - Type -> EClassA is Concept of Example.ecore
 - Goal -> To Show How Contributive GenDoc for Ecore works
 - attributeA : EByte
 - (.) EByte



ClassA



ClassA - Extensibility

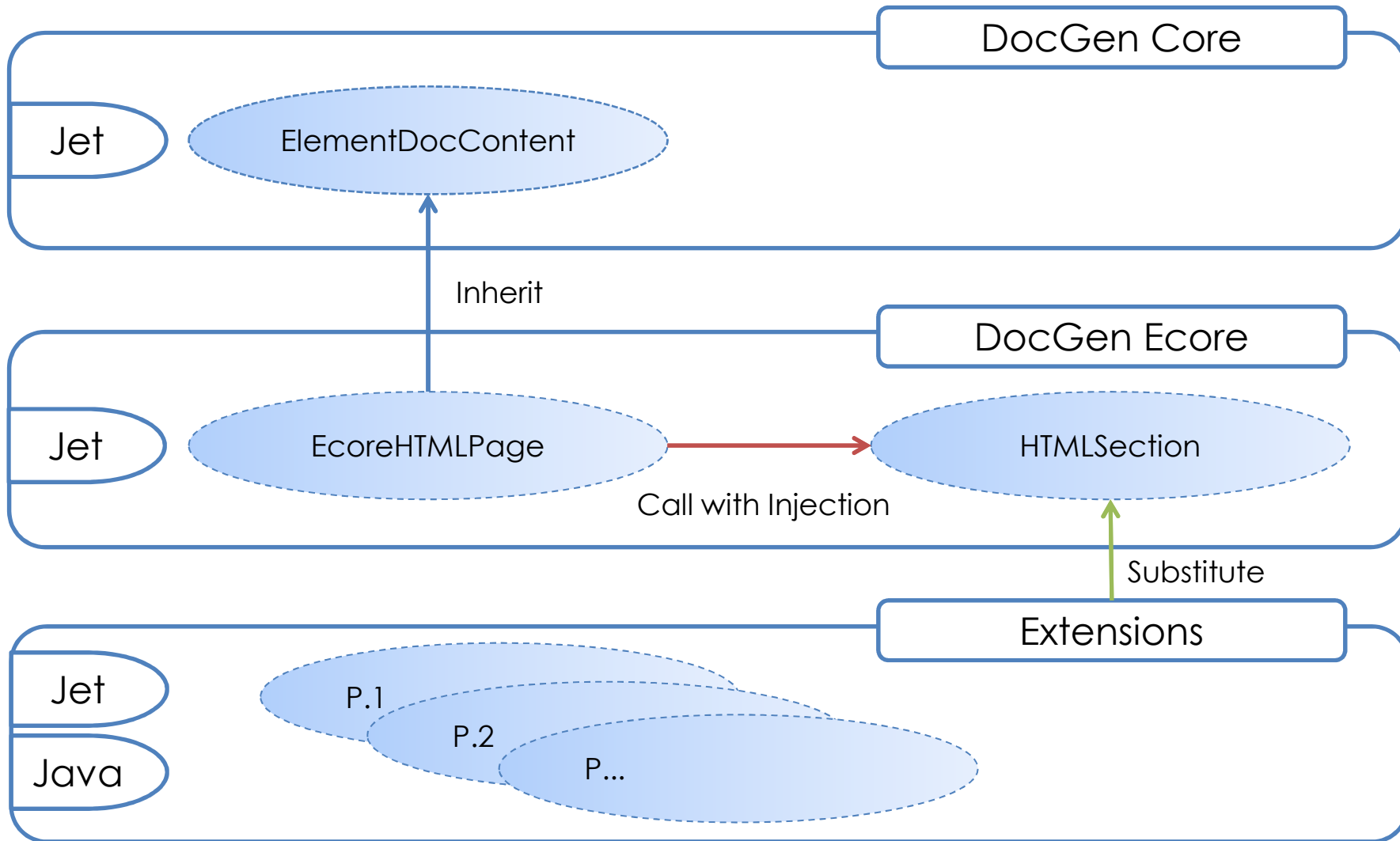


Class Annotations

Source: Documentation

Type	EClassA is Concept of Example.ecore
Goal	To Show How Contributive GenDoc for Ecore works

This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. ©THALES 2013 – All rights reserved.



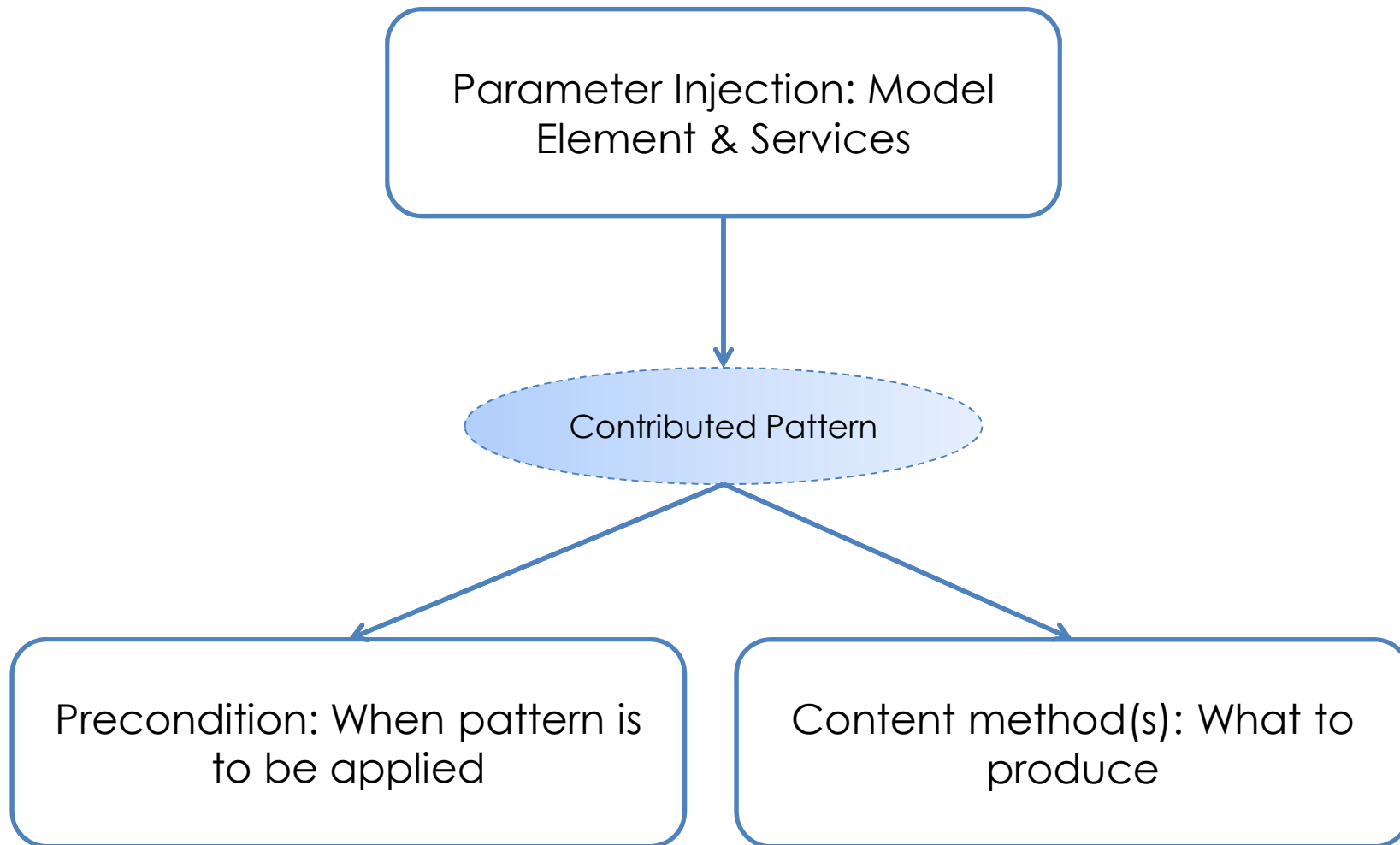
This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. ©THALES 2013 - All rights reserved.

EcoreHTMLPage

- Contribute DocGen Core
- Call “HTMLSection” pattern
- Work on EPackages, EClasses and EDatatypes
- Inject the current model element to “EcoreHTMLPattern”

HTMLSection

- Empty pattern
- Will be substituted by the contributions

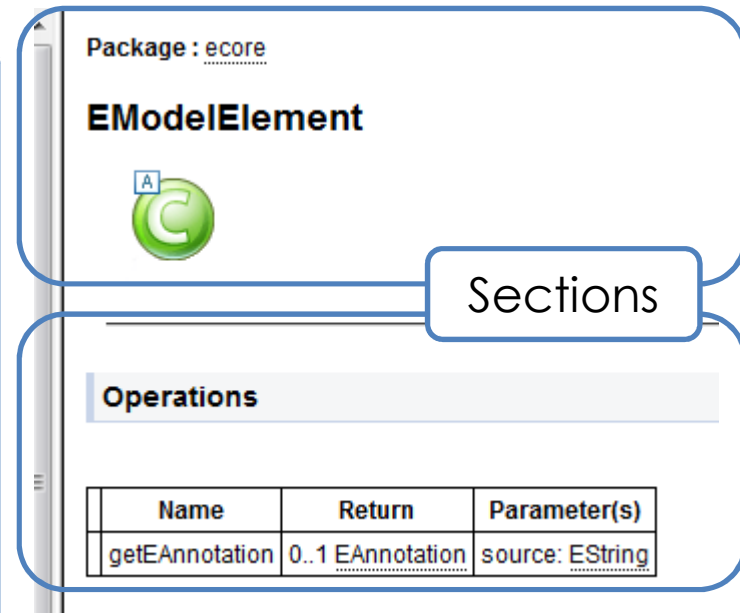


Parameter Injection

- Injection is done in Pattern parameters
- Available queries
 - Ecore DocGen Injected Element: To retrieve the model element
 - Inject Diagram Helper: Inject the diagram helper to manage relations between representation and semantic element
 - Inject File Name Service: Allows to compute file names and links

Ecore DocGen Patterns

- **EPackage**
 - EPackageHTMLSectionHeader
 - EPackageHTMLSubPackagesSection
 - EPackageDataTypeHTMLSection
- **EClass**
 - EClassHTMLHeader
 - EClassHTMLDescription
 - EClassFocusDiagram
 - EClassSuperType
 - EClassAttributes
 - EClassOperations
 - EClassReferences
 - EClassRepresentation
- **Enumeration**
 - EEnumHTMLHeader
 - EEnumHTMLContent



This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2013 – All rights reserved.

Examples

Objective

- Add EDataType page creation by JET Pattern nature
- Format:
 - EDataType Name [EDataType Instance Class]
 - Name: EDataType Name
 - Instance Class: Instance Class Name

Result

EDataType

EByteObject [java.lang.Byte]

- Name : EByteObject
- Instance Class: java.lang.Byte

- Create a new EGF Factory Component
- Create a Viewpoint Container
- Create a Pattern Viewpoint
- Create a Pattern Library
- Create a new JET or Java Pattern (e.g., EDataTypeSection_JET)
- Create a method(s) (e.g., content) and add it to the orchestration
- Implement the business of the method
- Add preCondition to the pattern to be applied only on EDataTypes
- Add Product Plan to Factory Component
- Invoke EcoreDocGenLauncher
 - Fill invocation contracts
 - ecoreFilePath
 - outputDirectoryPath
 - pattern.substitutions
 - Substitute « HTMLSection » pattern by: EDataTypeSection_JET and HTMLSection
- Run the Factory Component

EDataTypeHTMLSection_JET

Specification

Inheritance
Choose the super pattern:
Parent: No parent
Browse [X]

Pattern Nature
Select the kind of the pattern:
Type: JetNature

Parameters
Define parameters for this pattern in the following section.

Name	Type	Query
parameter	EDataType	Ecore DocGen Injected ...

Overview Specification Implementation

Query to retrieve injected model Element

EDataTypeHTMLSection_JET

EDataTypeHTMLSection_JET

Pattern methods:

- [header](#)
- [init](#)
- [preCondition](#)
- [footer](#)

Implementation methods:

content

+
✎
📄
🗑️
✖
↑
↓

Organize method calls:

content - [MethodCall]

+
✎
✖
↑
↓

Variables

Set up some variable available in all methods:

Name	Type	
		+
		✎
		✖

Overview
Specification
Implementation

This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2013 – All rights reserved.

2 Example – EDataTypeHTMLSection content and preCondition methods

Content method

```
EDatatypeHTMLSection_JET  EDatatypeHTMLSection_JET ✕
1 |k%
2 String text = LabelProviderHelper.getText(parameter);
3 %>
4
5 <% if (text != null && !text.isEmpty()) { %>
6 <h2>EDataType</h2>
7 <%= text %>
8 <br/>
9 <ul>
10 <li>Name : <%= parameter.getName() %></li>
11 <li>Instance Class: <%= parameter.getInstanceClass().getCanonicalName() %></li>
12 </ul>
13 <%}%>
14
```

header init preCondition content footer

Model Element from pattern parameter

preCondition method

```
EDatatypeHTMLSection_JET  EDatatypeHTMLSection_JET ✕
1 |return (parameter instanceof EEnum);
```

header init preCondition content footer

Objective

- Add “Class Annotations” section to EClass pages by Java Pattern nature
- Format:
 - Source: annotation source

Key 1	Value 1
Key 2	Value 2

Result

Class Annotations	
Source: http://www.eclipse.org/emf/2002/Ecore	
constraints	WellFormedName

- Steps are the same as in EDataType contribution example
- Name of pattern: EAnnotationHTMLSection_JAVA
- Add substitution pattern in factory component:
 - Substitute « HTMLSection » pattern by:
EAnnotationHTMLPattern_JAVA and HTMLSection patterns

EAnnotationHTMLSection_Java

DocGenExtensions.fcore EAnnotationHTMLSection_JAVA

Specification

Inheritance
Choose the super pattern:
Parent: No parent
Browse X

Pattern Nature
Select the kind of the pattern:
Type: JavaNature

Parameters
Define parameters for this pattern in the following section.

Name	Type	Query
parameter	EClass	Ecore DocGen Injected El...

Overview Specification Implementation

EAnnotationHTMLSection_Java

Implementation

Methods

Pattern methods:

- header
- init
- preCondition
- footer

Implementation methods:

- content

Variables

Set up some variable available in all methods:

Name	Type

Orchestration

Organize method calls:

- content - [MethodCall]

Overview | Specification | Implementation

Content method

```
DocGenExtensions.fcore  EAnnotationHTMLSection_JAVA  EAnnotationHTMLSection_JAVA  ⌵
1 EList<EAnnotation> eAnnotations = parameter.getEAnnotations();
2
3 out.append("<h2>Class Annotations</h2>");
4 for (EAnnotation eAnnotation : eAnnotations) {
5
6     EMap<String, String> details = eAnnotation.getDetails();
7     String source = eAnnotation.getSource();
8
9     out.append("<b>Source: <i>").append(source).append("</i></b>");
10    out.append("<table>");
11    for (Entry<String, String> entry : details) {
12        out.append("<tr>");
13        out.append("<td>").append(entry.getKey()).append("</td><td>").append(entry.getValue()).append("</td>");
14        out.append("</tr>");
15    }
16    out.append("</table>");
17 }
```

header init precondition content footer

preCondition method

```
DocGenExtensions.fcore  EAnnotationHTMLSection_JAVA  EAnnotationHTMLSection_JAVA  ⌵
1 return !(parameter.getEAnnotations().isEmpty());
```

header init precondition content footer

Contributive XHTML DocGen for Capella

- ◆ Purpose
- ◆ Introduction
- ◆ Concept – Contributive Capella XHTML DocGen
- ◆ Contributive XHTML with Extension point
- ◆ Example: Basic Viewpoints

Context

- Capella Models are complex, extensible and Basic documentation is insufficient
- Existing Capella XHTML DocGen is not contributive

Need

- Need to extend Capella XHTML DocGen for customizing, adding or remove information in generated documentation from models

Objective

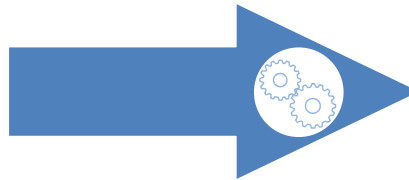
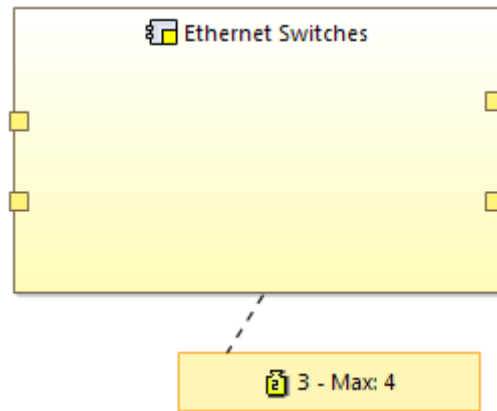
- Allowing selection of multiple extensions when launching a documentation generation

Objective



Action

- Make Capella XHTML DocGen to be contributive by pattern calls and substitution mechanism
- Adding pattern calls and patterns substitutions to make the Capella Docgen Contributive



No description.

Ports

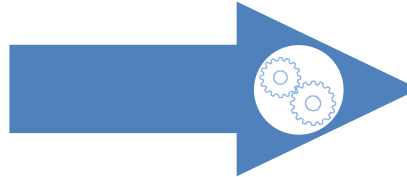
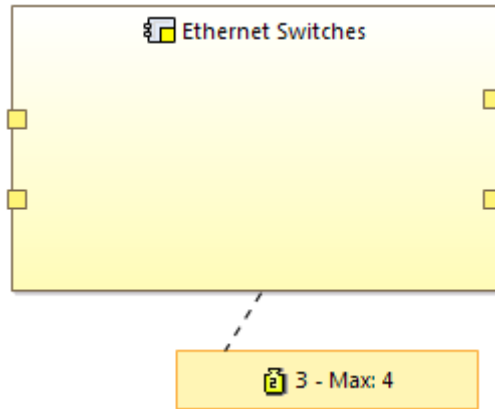
- PP2
- PP4
- PP5
- PP7

Diagrams displaying "Ethernet Switches"

- [PCBD] Implementation Components
- [EAB] Configuration Items and Realized Artefacts
- [FAB] [BUILD] All PCs, PFs, FEs
- [FAB] Focus on Network Setup, Configuration and Tests
- [FAB] Implementation and Behaviour Components
- [FAB] [BUILD] Template
- [FAB] Focus on Network Transport

Copyright © 2014 The Eclipse Foundation. All Rights Reserved.

Page without contributions



- PP 6
- PP 7

Diagrams displaying "Ethernet Switches"

- [PAB] Focus on Network Setup, Configuration and Tests
- [EAB] Configuration Items and Realized Artefacts
- [PAB] [BUILD] All PCs, PFs, FEs
- [PCBD] Implementation Components
- [PAB] [BUILD] Template
- [PAB] Focus on Network Transport
- [PAB] Implementation and Behaviour Components

Mass Viewpoint

Current mass: 3
Max mass: 2;

Price Viewpoint

Price: 18
Max price: 30

Copyright © 2014 The Eclipse Foundation. All Rights Reserved.

Page with viewpoint contributions

This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2013 – All rights reserved.

Mechanism

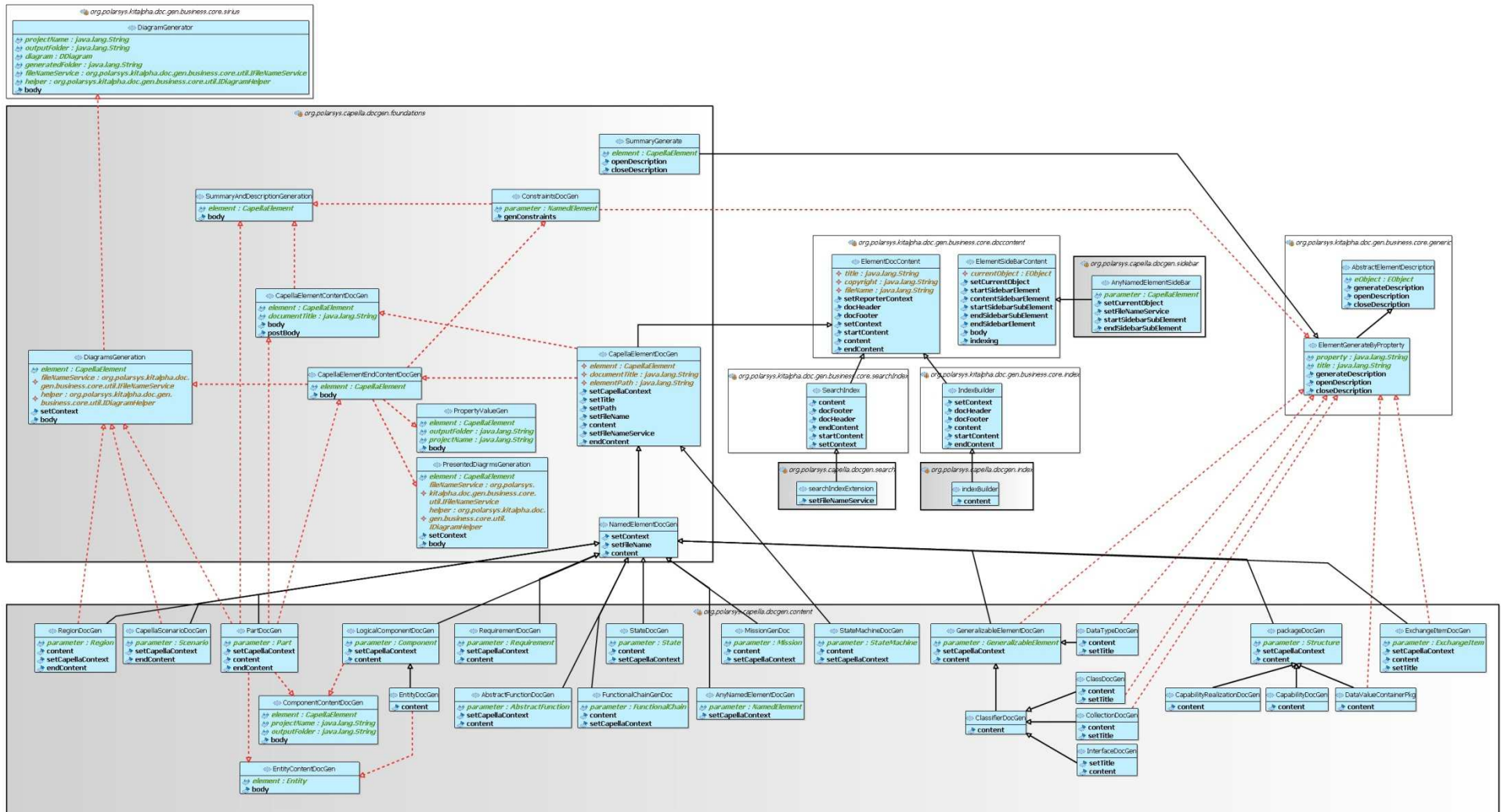
Definition of Pattern extension points where contributions can come with their functionalities



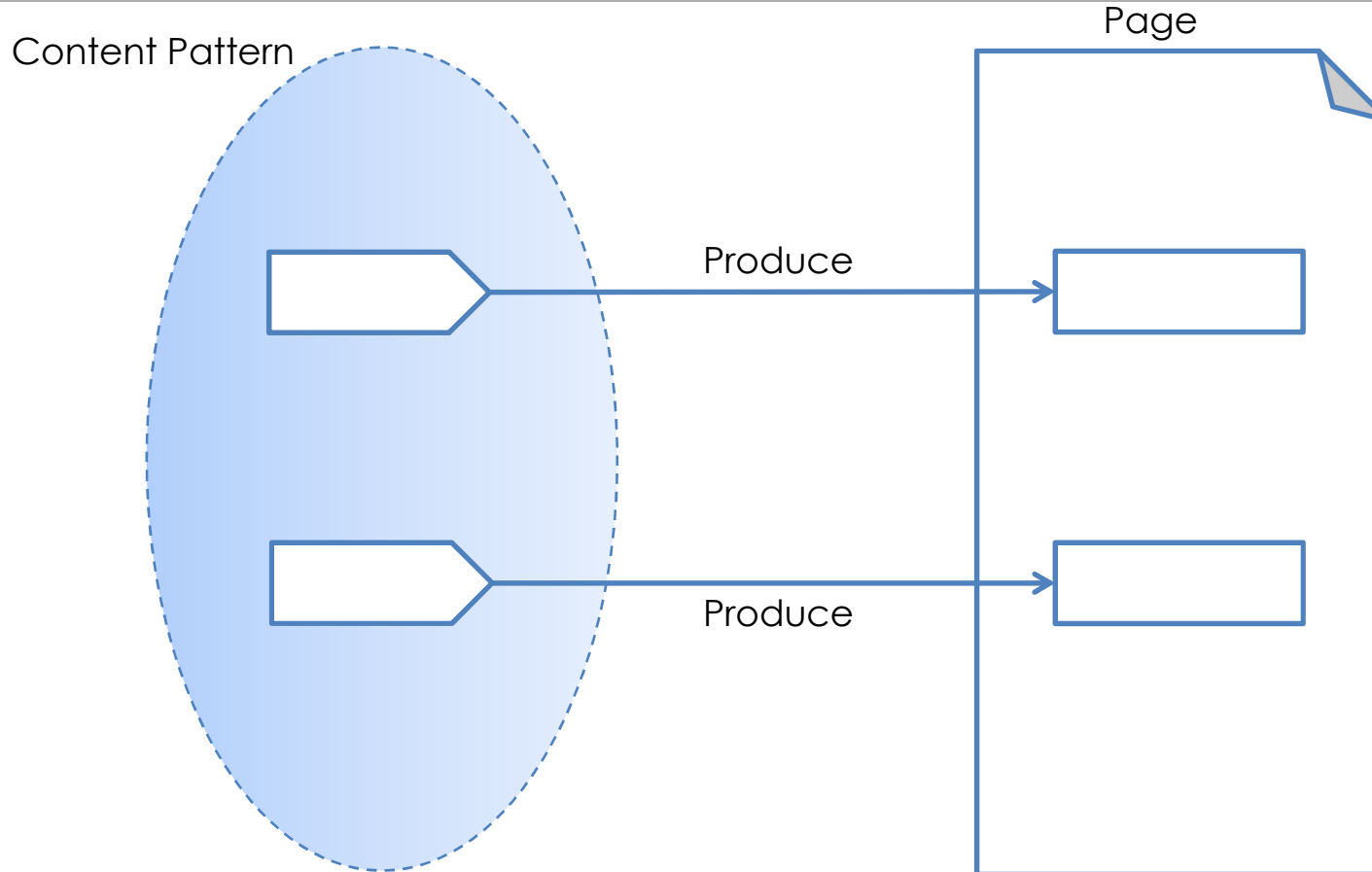
Extensions which aimed to add, customize, replace or extends functionalities



Pattern Extension Point



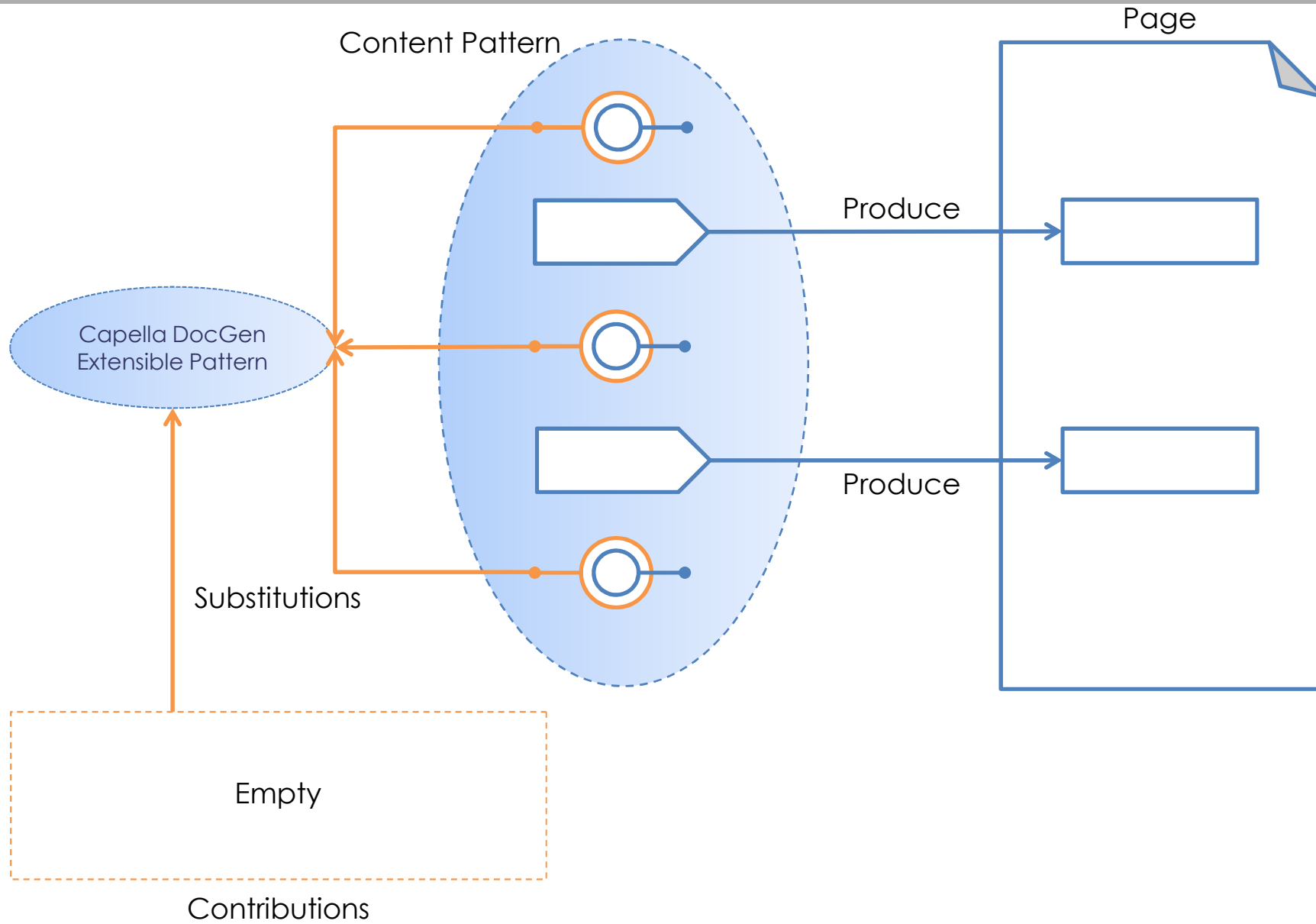
This do without



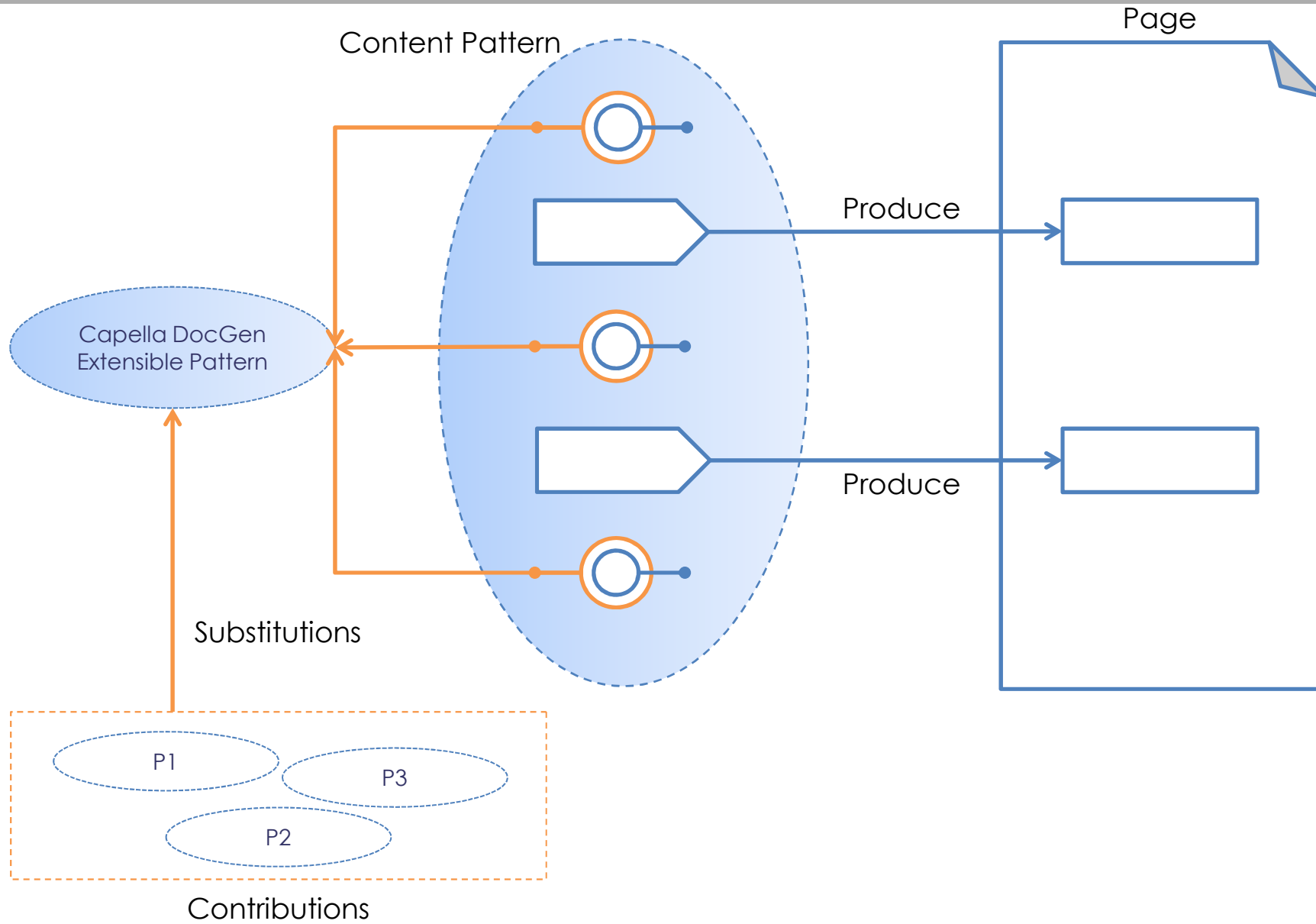
Piece of generation code which produce piece of XHTML page



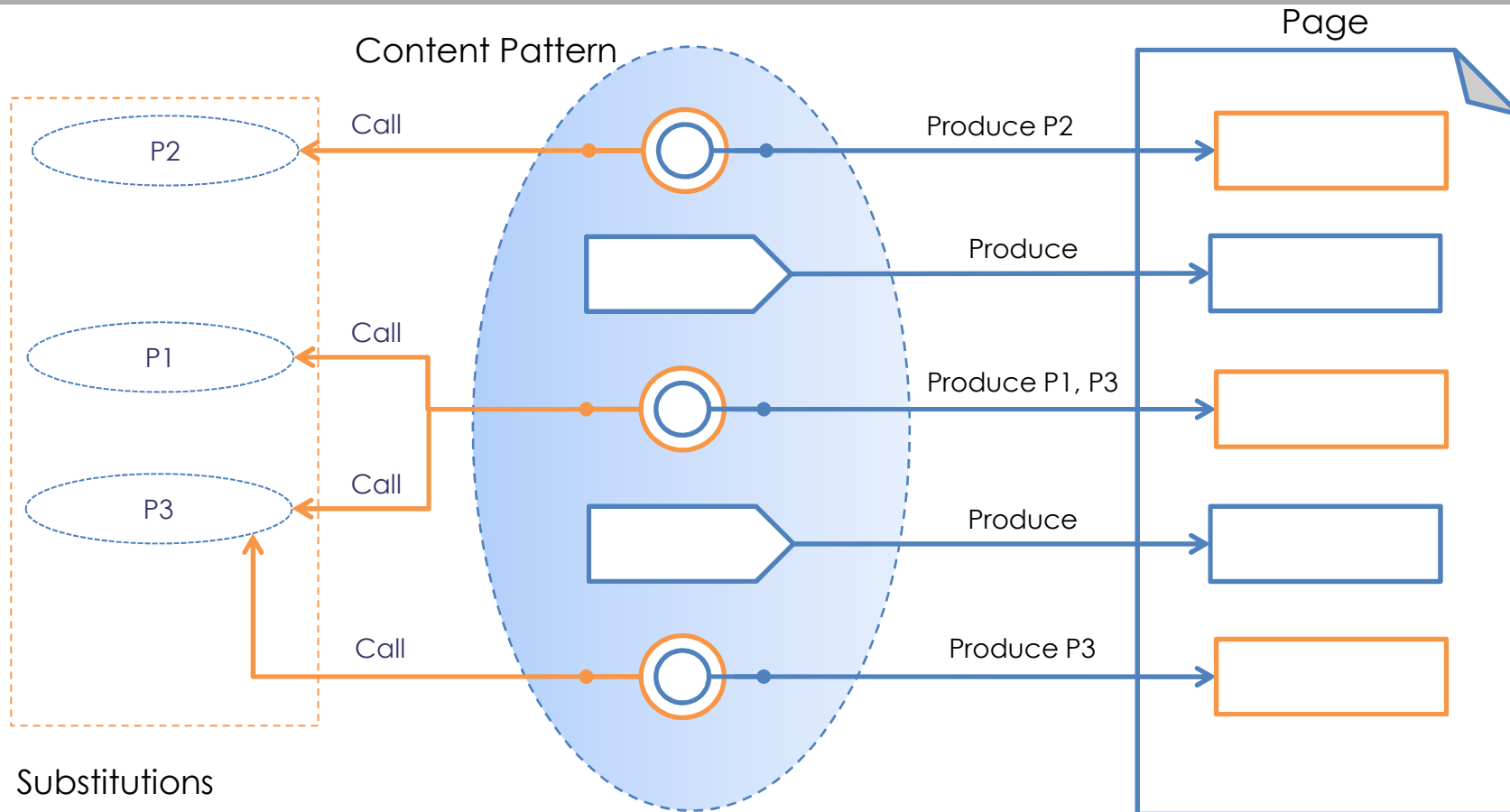
Piece of XHTML Page



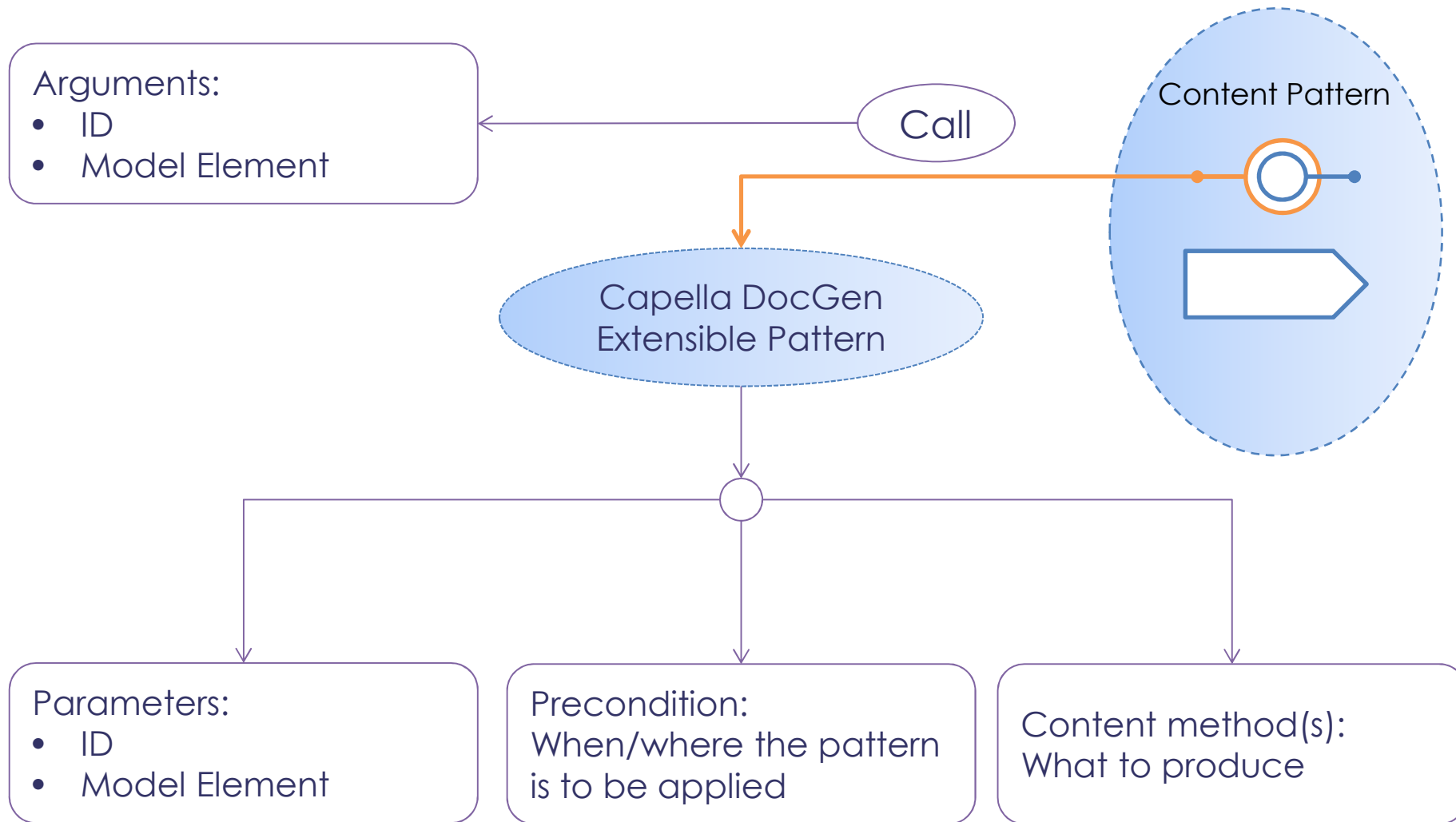
This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2013 – All rights reserved.



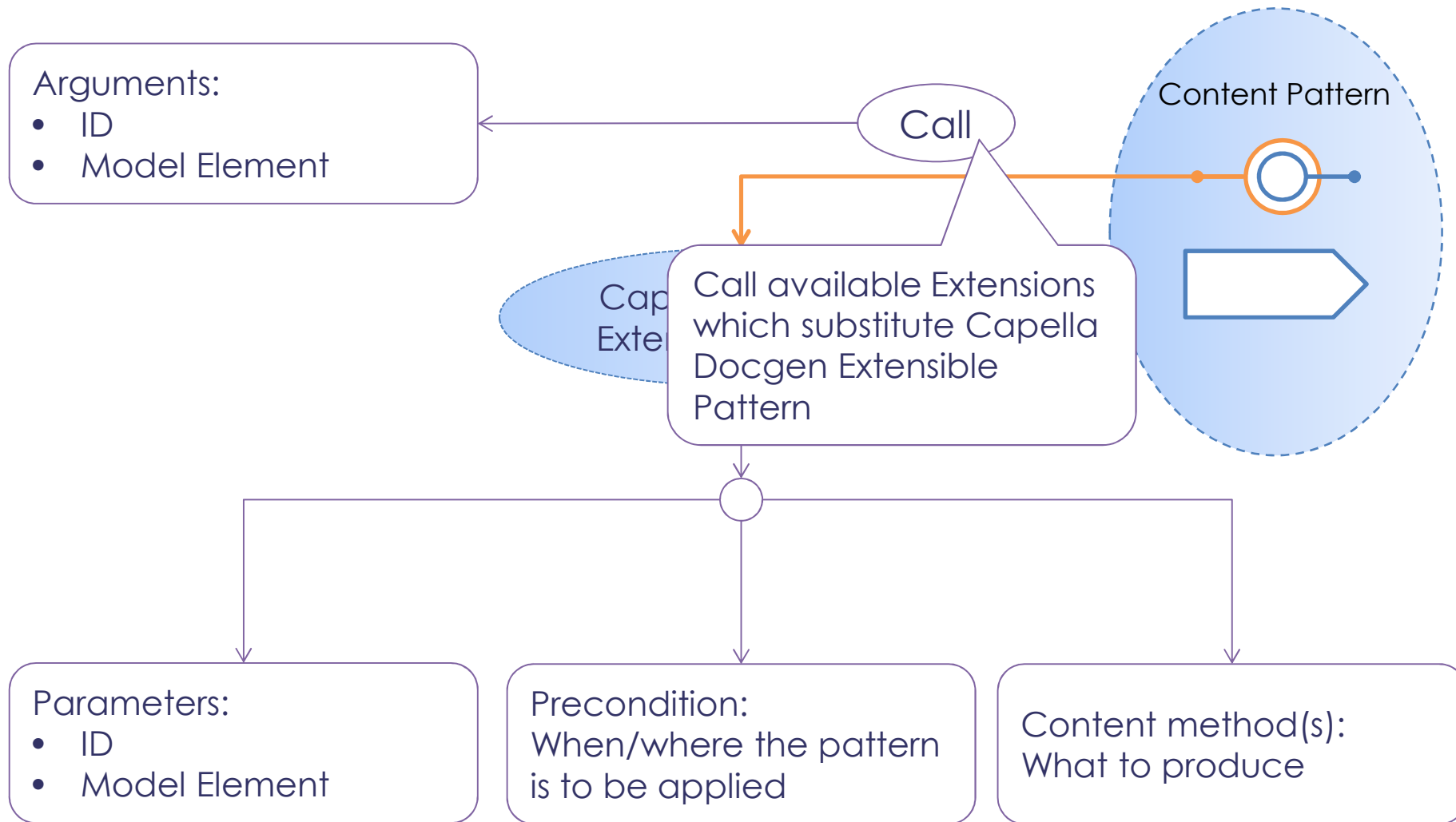
This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2013 – All rights reserved.



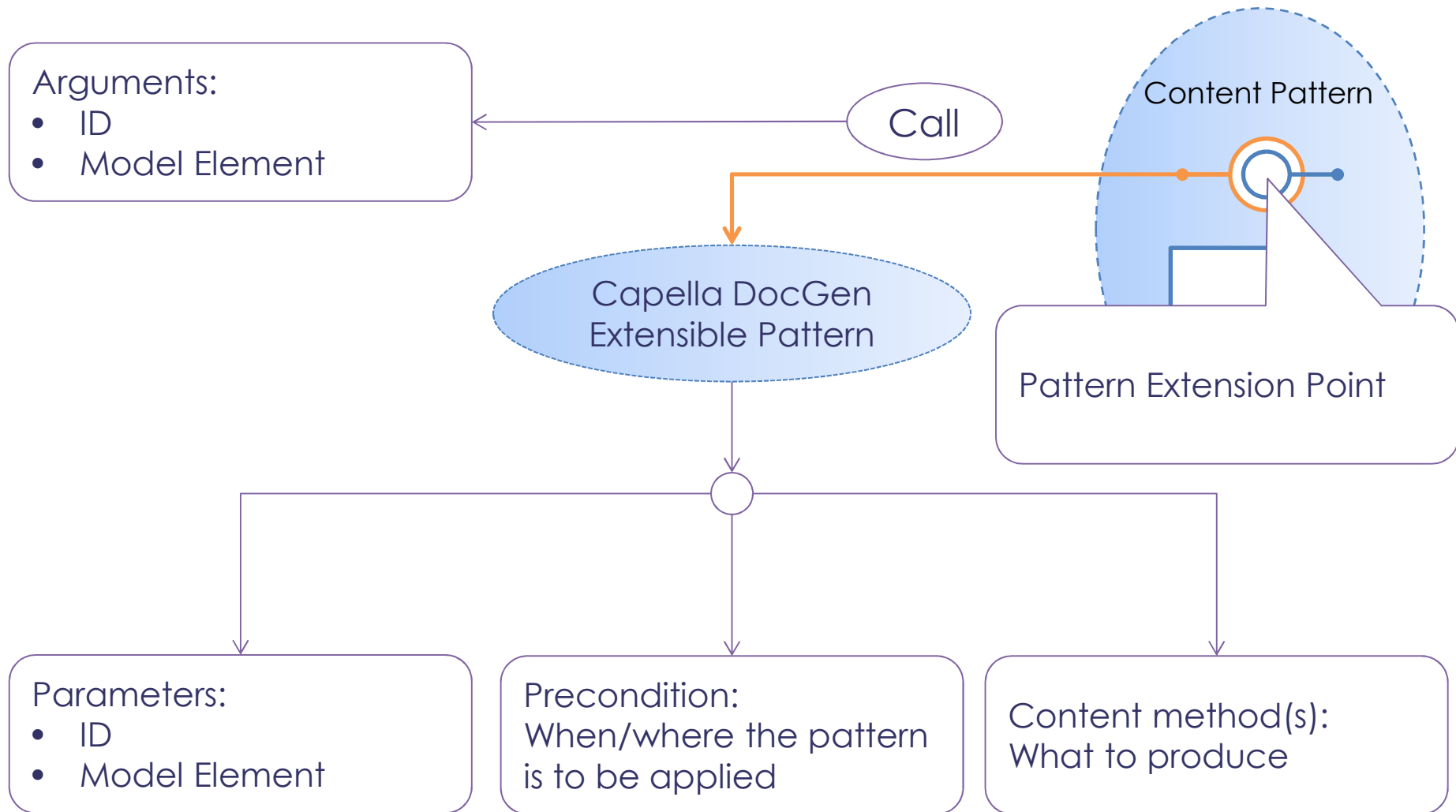
This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. ©THALES 2013 – All rights reserved.



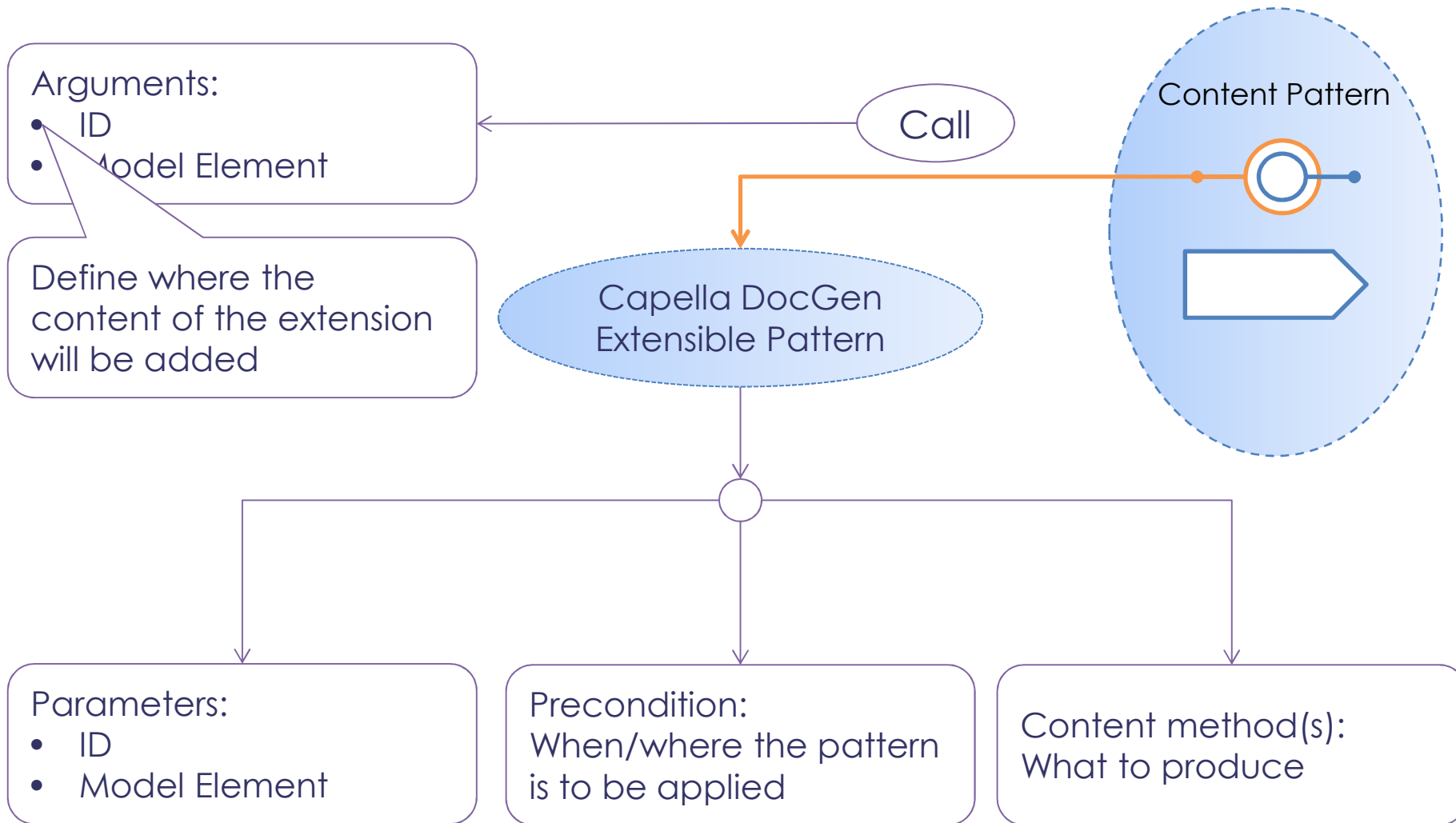
This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. ©THALES 2013 – All rights reserved.



This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. ©THALES 2013 – All rights reserved.



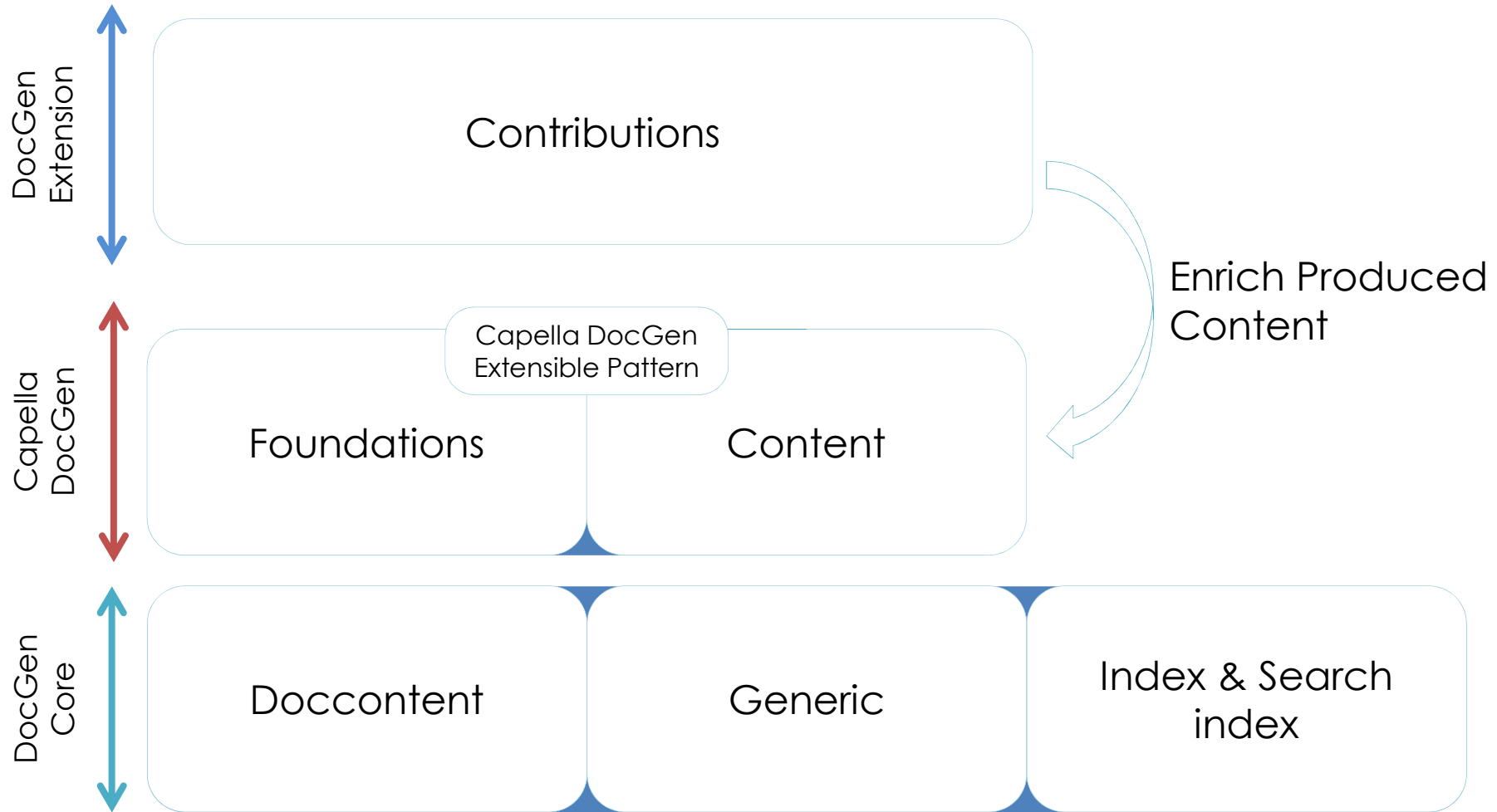
This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. ©THALES 2013 – All rights reserved.



This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2013 – All rights reserved.

IDs are defined in: org.polarsys.capella.docgen.extensions.ID

```
1 package org.polarsys.capella.docgen.extensions;
2
3 public interface ID {
4
5     //Parts IDs
6     public final String PART_BEFORE_HEADER = "org.polarsys.capella.docgen.content.PartDocGen.part.before.header";
7     /**
8      * After header & before description
9      */
10    public final String PART_AFTER_HEADER = "org.polarsys.capella.docgen.content.PartDocGen.part.after.header";
11    /**
12     * After Description & before Diagram generation
13     */
14    public final String PART_AFTER_DESCRIPTION = "org.polarsys.capella.docgen.content.PartDocGen.part.after.description";
15    /**
16     * After Diagram Generation & before type
17     */
18    public final String PART_AFTER_DIAGRAM_GENERATION = "org.polarsys.capella.docgen.content.PartDocGen.part.after.diagram.generation";
19    /**
20     * before type Content
21     */
22    public final String PART_BEFORE_TYPE_CONTENT = "org.polarsys.capella.docgen.content.PartDocGen.part.before.type.content";
23    /**
24     * After type content and before component content
25     */
26 }
```



This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. ©THALES 2013 – All rights reserved.



www.thalesgroup.com

Contributive XHTML with Extension point

OPEN

THALES

Context

- Models are complex and Basic documentation is insufficient
- Existing Core XHTML DocGen is not contributive

Need

- Need to extend XHTML DocGen for customizing and adding or remove information in generated documentation from models

Objective

- Rework existing Core XHTML DocGen framework to be contributive

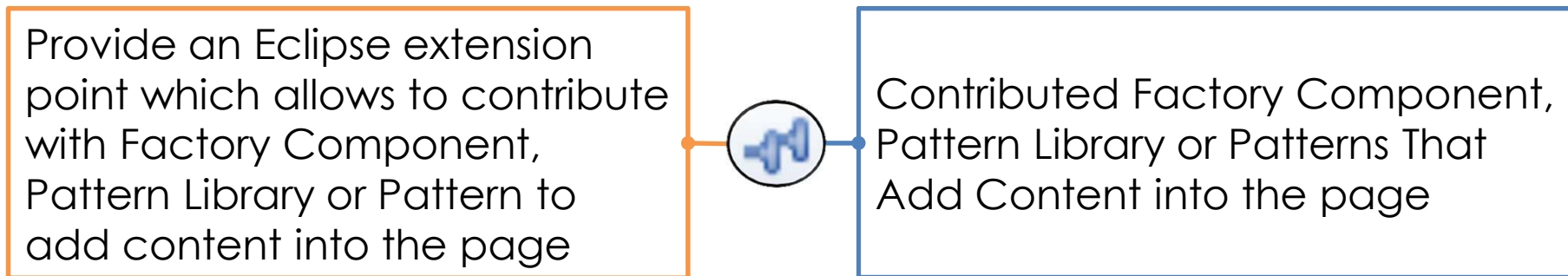
Objective



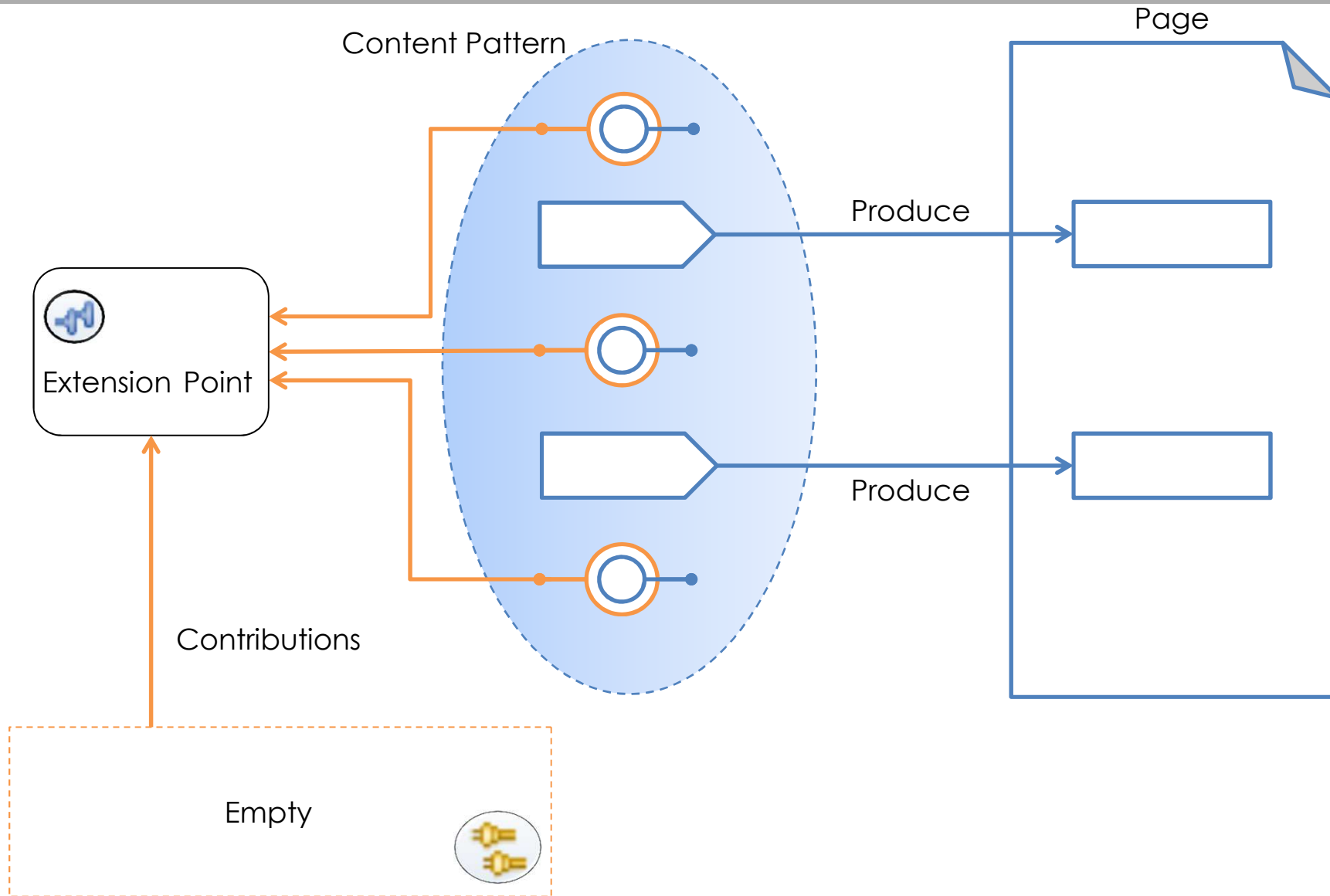
Action

- Work on existing Core XHTML DocGen framework to be contributive by Eclipse-base extension point
- Add to existing Core XHTML DocGen framework to be contributive by Eclipse-base extension point

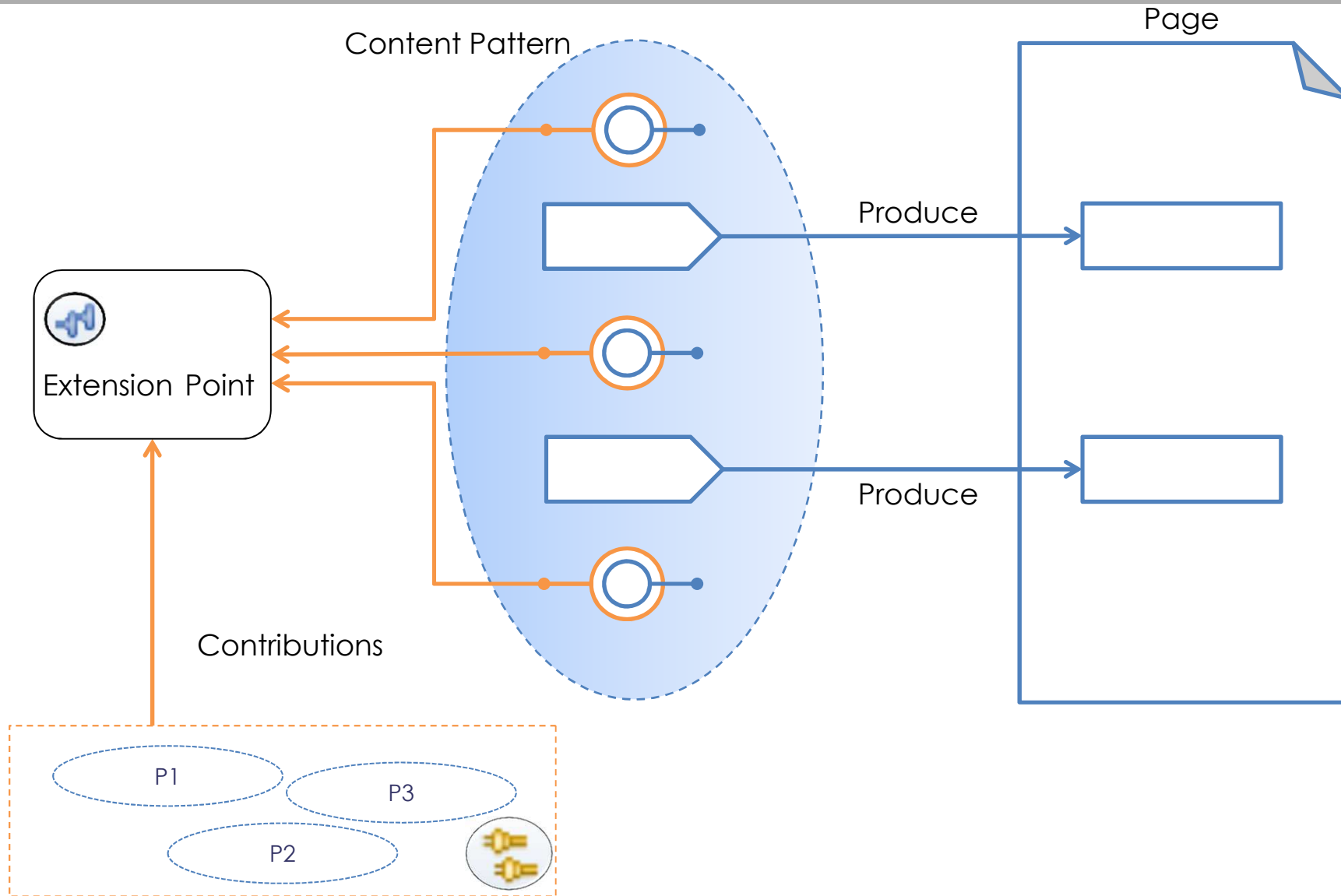
Mechanism



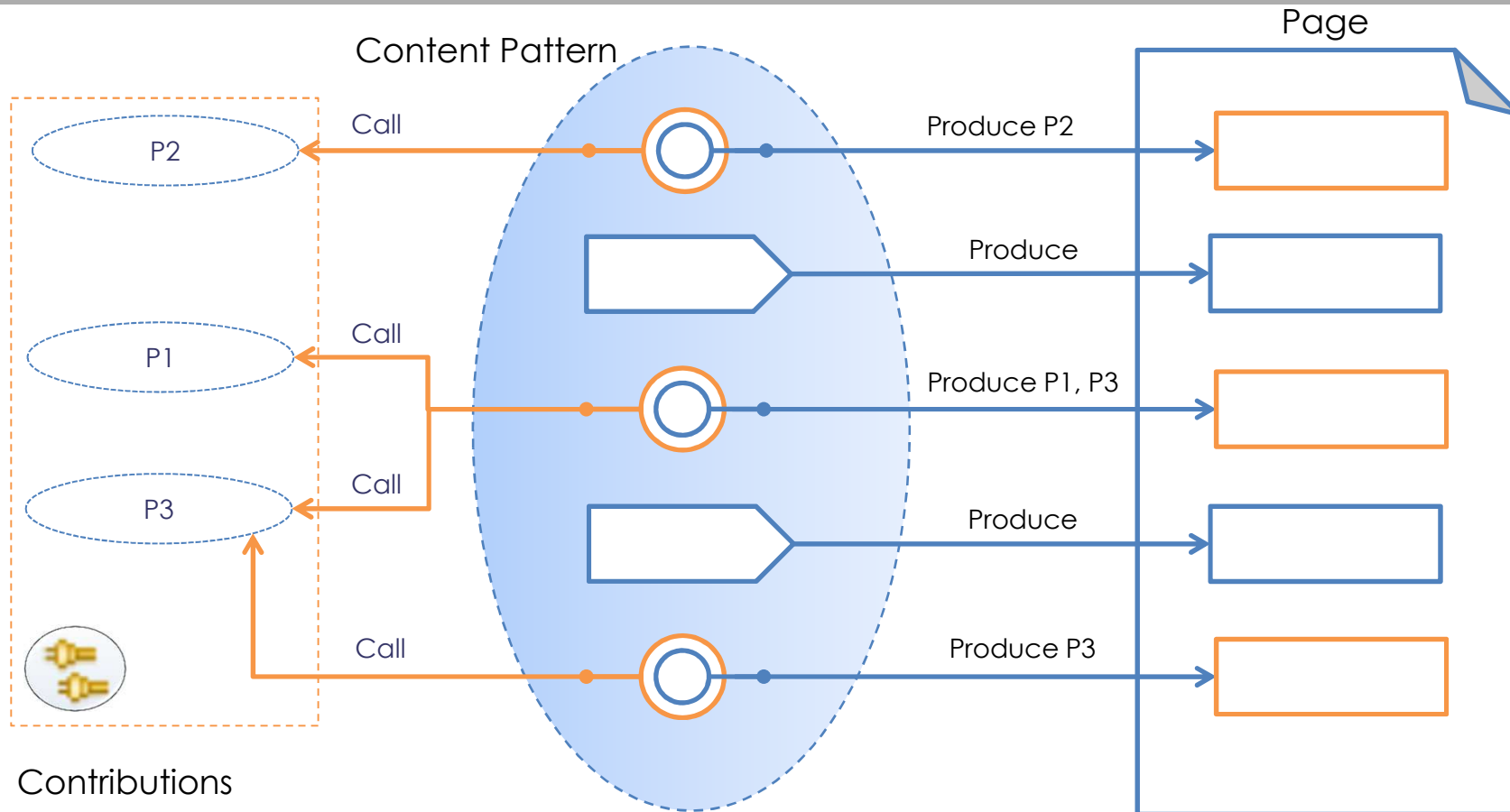
Eclipse Extension Point



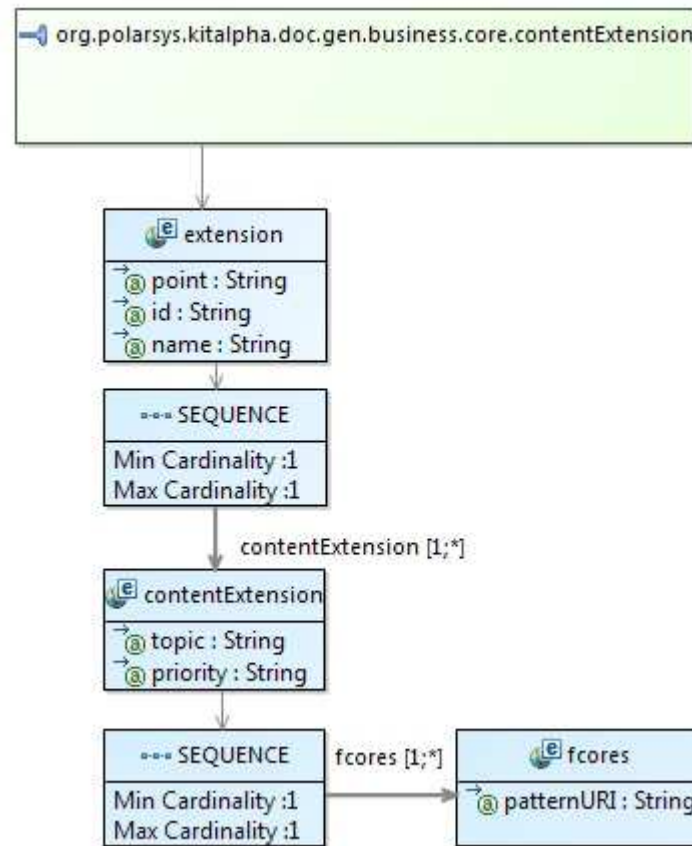
This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2013 – All rights reserved.



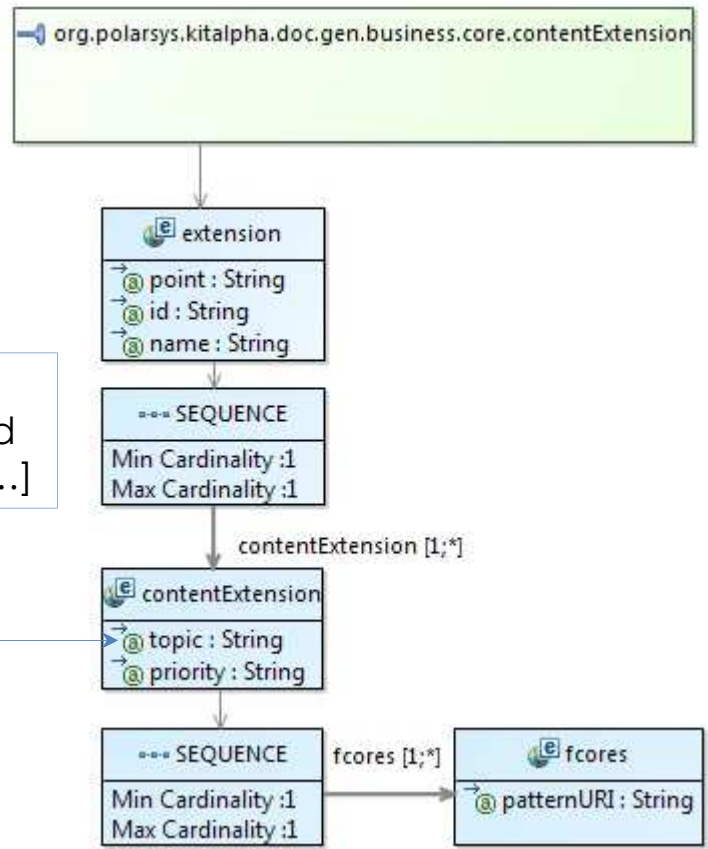
This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2013 – All rights reserved.



This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2013 – All rights reserved.

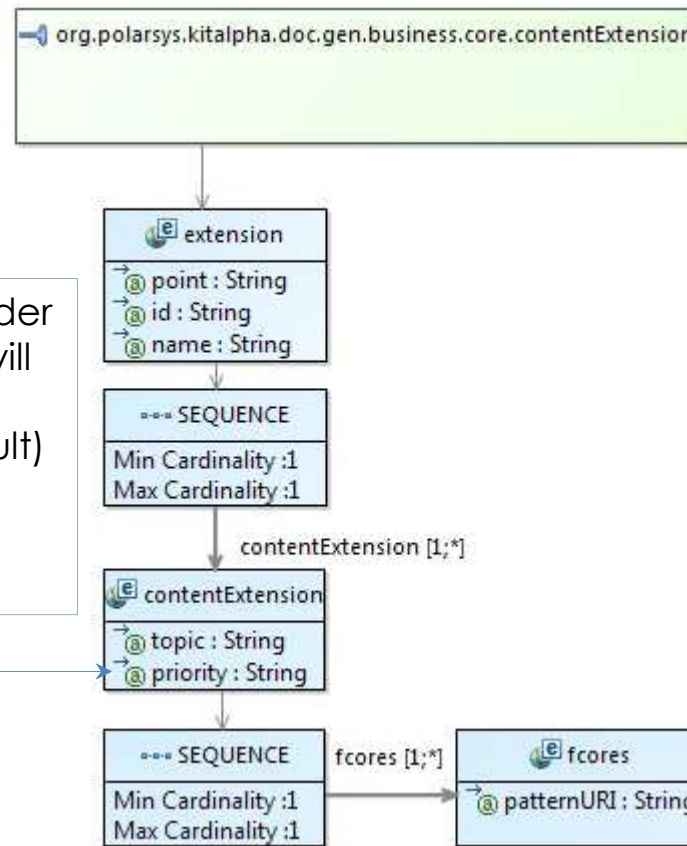


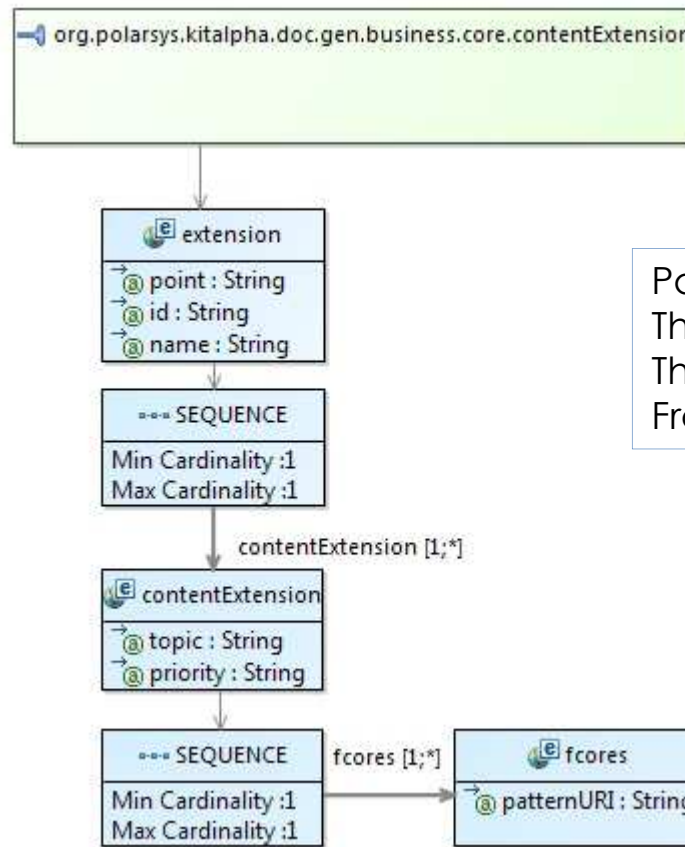
Topic indicate where the Contributions will be applied
Format: TOPIC_1 [, TOPIC2,...]



This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2013 – All rights reserved.

Priority indicates in which order
The contributions to topics will
Be executed.
It's an integer from -1 (default)
To n.
-1 : is the lowest priority
0 : is the highest priority





PatternURI indicates the path to the patterns.
The path must contain the fragment

This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2013 – All rights reserved.

- **Topic**
 - Declaration of execution point where a set of patterns will be applied
 - Dispatching a set of patterns on execution points on which they will be applied
- **Execution point**
 - Point in the flow execution where external behavior will be applied
- **Dispatching**
 - Association of a set of execution with a set of patterns
 - The patterns have a priority (lowest by default)

- Services to declare execution point are defined in:
org.polarsys.kitalpha.doc.gen.business.core.extension.intf.IDocGenExtensionEngine

- Example to declare Execution point “MY_EXECUTION_POINT”

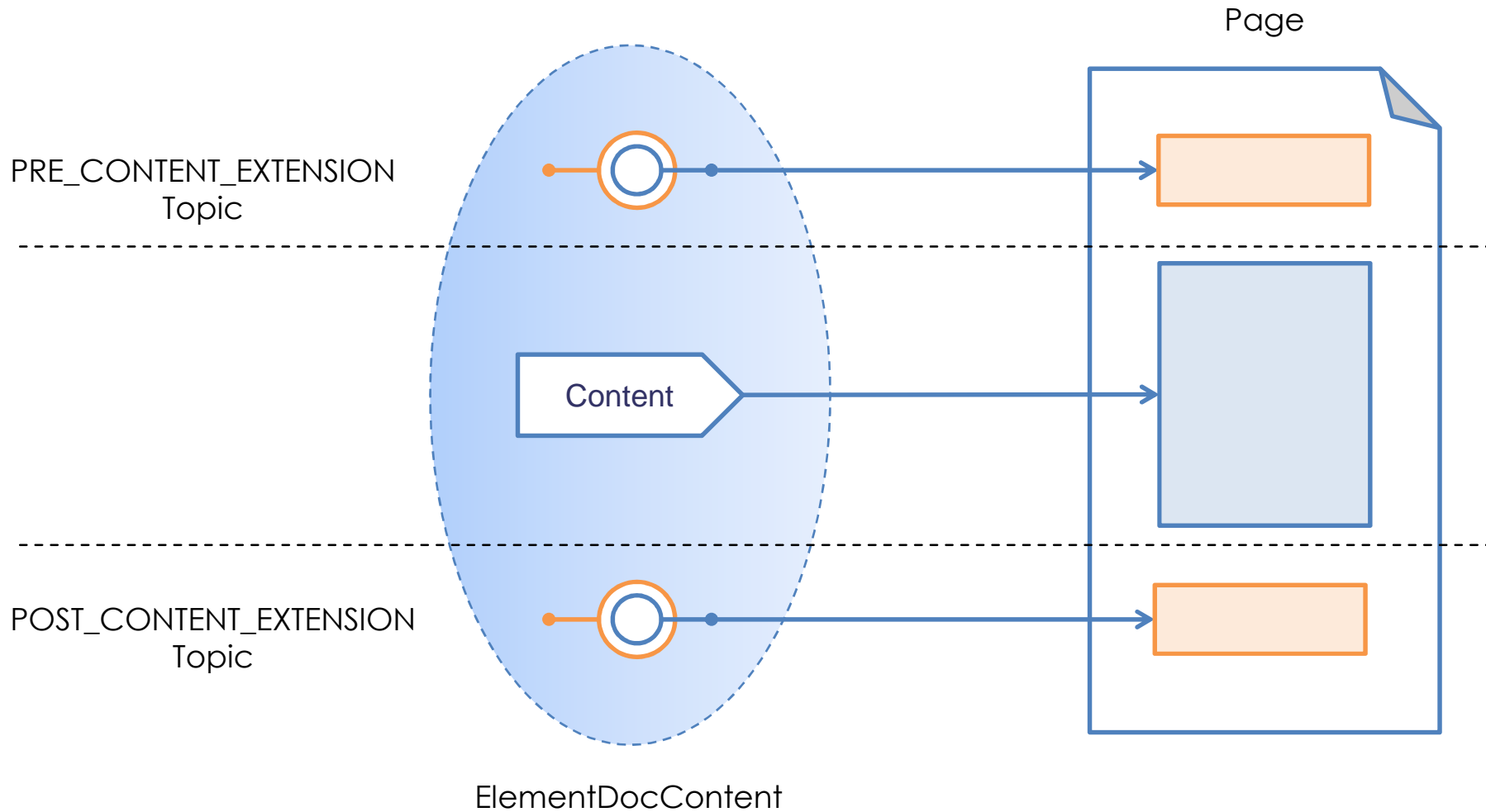
```
DocGenExtensionFactory.newDocGenExtensionEngine().apply(  
    « MY_EXECUTION_POINT”, //the execution point  
    ctx, //the context of the pattern  
    getParameters(), //the parameters of the pattern  
    stringBuffer); //a buffer
```

- Example to declare generic execution point

```
DocGenExtensionFactory.newDocGenExtensionEngine().execute(  
    ctx, //the context of the pattern  
    getParameters()); //the parameters of the pattern
```


- **Dispatching**
 - Contribution to: org.polarsys.kitalpha.doc.gen.business.core.extensionContent
- Example

The screenshot shows the Eclipse IDE's 'Extensions' view. The left pane, titled 'All Extensions', contains a tree view of extensions. The root node is 'org.polarsys.kitalpha.doc.gen.business.core.contentExtension', which has a sub-node '(contentExtension)'. This sub-node has two children: 'platform:/plugin/org.polarsys.capella.docgen.extension' and 'platform:/plugin/org.polarsys.capella.docgen.extension'. To the right of the tree are buttons for 'Add...', 'Remove', 'Up', and 'Down'. Below the tree is a search filter box labeled 'type filter text'. The right pane, titled 'Extension Element Details', shows the properties for the selected extension. It includes a 'topic*' field with the value 'POST_CONTENT_EXTENSION' and a 'priority' field with the value '0'. The bottom of the screenshot shows a series of tabs: 'Overview', 'Dependencies', 'Runtime', 'Extensions', 'Extension Points', 'Build', 'MANIFEST.MF', 'plugin.xml', and 'build.properties'. The 'Extensions' tab is currently selected.

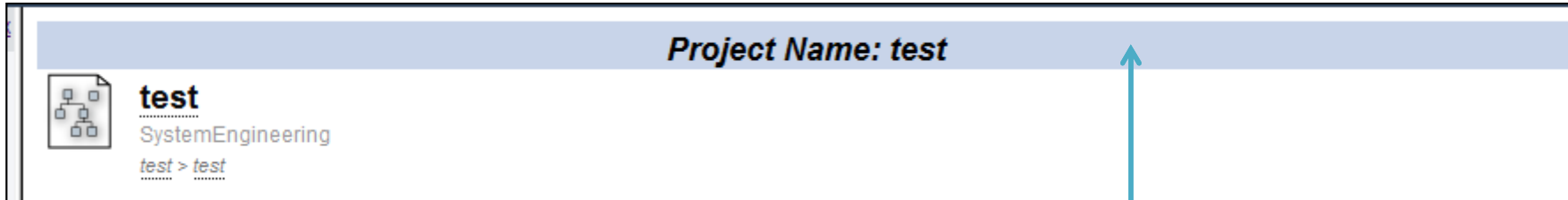


This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2013 – All rights reserved.

Examples

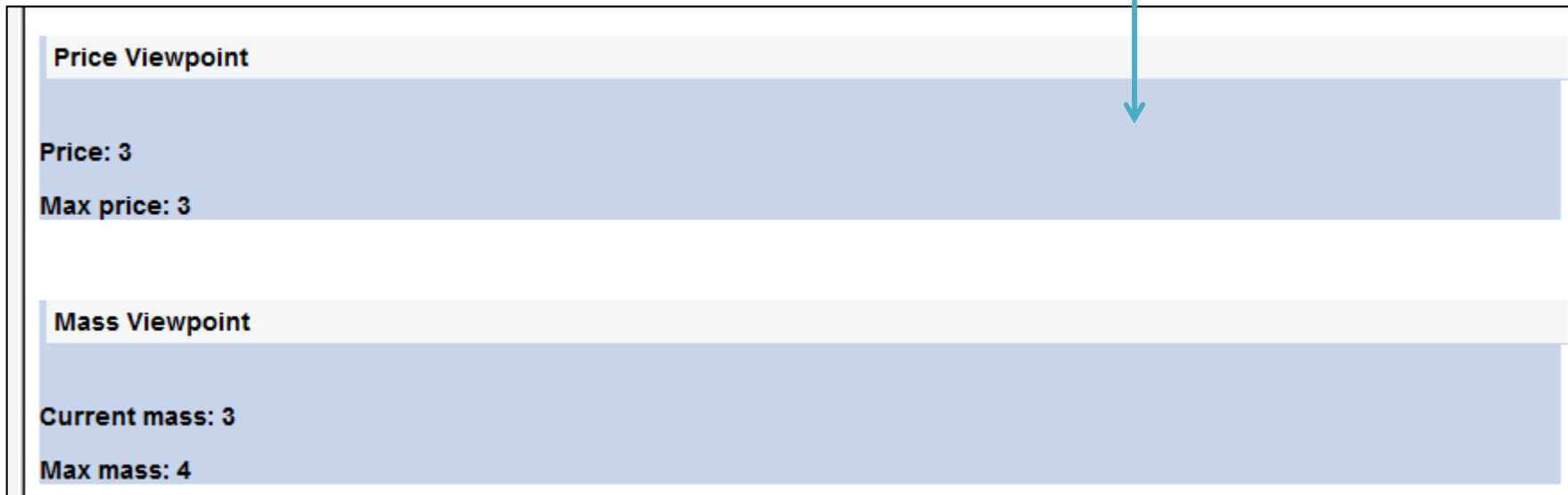
- Use case
 - PRE_CONTENT_EXTENSION
 - Display the project name in center with changing the background color on all pages
 - POST_CONTENT_EXTENSION
 - Display the information of Price and Mass viewpoints in that order and apply the same background color as project name
 - The background color settings must be contributed

PRE_CONTENT_EXTENSION



The screenshot shows a software interface with a light blue header bar containing the text "Project Name: test". Below the header, on the left, is a tree icon and the text "test", "SystemEngineering", and "test > test".

POST_CONTENT_EXTENSION



The screenshot shows a software interface with two sections. The first section is titled "Price Viewpoint" and contains the text "Price: 3" and "Max price: 3". The second section is titled "Mass Viewpoint" and contains the text "Current mass: 3" and "Max mass: 4".

Same background



This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. ©THALES 2013 – All rights reserved.

Steps to contribute to PRE_CONTENT_EXTENSION execution point

1. Create fcore with pattern named ProjectName
2. Add parameter named "parameter" typed Eobject
3. Add precondition which check if the root is Capella project
4. Implement and orchestrate the body method (cf. code after)
5. Contribute to dispatch the pattern on PRE_CONTENT_EXTENSION with priority 0

Pattern code

```
If (parameter != null){
  Map<String, Object> callParameters = new HashMap<String, Object>();
  callParameters.put("parameter", this.parameter);
  DocGenExtensionFactory.newDocGenExtensionEngine().apply("BEFORE_PROCESSING_CONTENT",
                                                         ctx, callParameters, out);

  String name = ((Project) parameter).getName();
  out.append("<div style='color:black'><center><h1><i>Project Name: ").
    append(name).
    append("</i></h1></center></div>");
  DocGenExtensionFactory.newDocGenExtensionEngine().apply("AFTER_PROCESSING_CONTENT",
                                                         ctx, callParameters, out);
}
```

Notice this contribution declares 2 execution point named BEFORE_PROCESSING_CONTENT and AFTER_PROCESSING_CONTENT. These execution point will served to apply a background color

Steps to contribute to POST_CONTENT_EXTENSION execution point

1. Create fcore with pattern named pricevp
2. Add parameter named "parameter" typed EObject
3. Add precondition which check if the parameter is part
4. Implement and orchestrate the body method (cf. code next slide)
5. Contribute to dispatch the pattern on POST_CONTENT_EXTENSION with Priority 0

Pattern code

```

if (parameter != null && ((Part) parameter).getAbstractType() instanceof PhysicalComponent) {
    PhysicalComponent p = (PhysicalComponent) ((Part) parameter).getAbstractType();
    EList<AbstractTypedElement> abstractTypedElements = p.getAbstractTypedElements();
    for (AbstractTypedElement ac : abstractTypedElements) {
        EList<ElementExtension> ownedExtensions = ac.getOwnedExtensions();
        for (ElementExtension e : ownedExtensions) {
            if (e instanceof PartPrice) {
                Map<String, Object> callParameters = new HashMap<String, Object>();
                callParameters.put("parameter", this.parameter);
                DocGenExtensionFactory.newDocGenExtensionEngine()
                    .apply("BEFORE_PROCESSING_CONTENT", ctx, callParameters, out);
                PartPrice price = ((PartPrice) e);
                int value = price.getValue();
                int maxValue = price.getMaxValue();
                out.append("<h2>Price Viewpoint</h2>");
                out.append("<h3>Price: ").append(value).append("<br/></h3>\n");
                out.append("<h3>Max price: ").append(maxValue).append("</h3>");
                DocGenExtensionFactory.newDocGenExtensionEngine()
                    .apply("AFTER_PROCESSING_CONTENT", ctx, callParameters, out);
            }
        }
    }
}

```

Notice this contribution declares 2 execution point named BEFORE_PROCESSING_CONTENT and AFTER_PROCESSING_CONTENT. These execution point will served to apply a background color

Steps to contribute to POST_CONTENT_EXTENSION execution point

1. Create fcore with pattern named massvp
2. Add parameter named "parameter" typed EObject
3. Add precondition which check if the parameter is part
4. Implement and orchestrate the body method (cf. code next slide)
5. Contribute to dispatch the pattern on POST_CONTENT_EXTENSION with Priority 1

Pattern code

```

if (parameter != null && ((Part) parameter).getAbstractType() instanceof PhysicalComponent) {
    PhysicalComponent p = (PhysicalComponent) ((Part) parameter).getAbstractType();
    EList<AbstractTypedElement> abstractTypedElements = p.getAbstractTypedElements();
    for (AbstractTypedElement ac : abstractTypedElements) {
        EList<ElementExtension> ownedExtensions = ac.getOwnedExtensions();
        for (ElementExtension e : ownedExtensions) {
            if (e instanceof PartMass) {
                Map<String, Object> callParameters = new HashMap<String, Object>();
                callParameters.put("parameter", this.parameter);
                DocGenExtensionFactory.newDocGenExtensionEngine()
                    .apply("BEFORE_PROCESSING_CONTENT", ctx, callParameters, stringBuffer);
                PartMass p1 = (PartMass) e;
                currentMass = p1.getValue();
                maxValue = p1.getMaxValue();

                stringBuffer.append("<h2>Mass Viewpoint</h2>");
                stringBuffer.append(currentMass);
                stringBuffer.append("<h3>Current mass: ");
                stringBuffer.append(maxValue).append("<br/></h3><h3>");
                stringBuffer.append("<h3>Max mass: ").append("</h3>");

                DocGenExtensionFactory.newDocGenExtensionEngine()
                    .apply("AFTER_PROCESSING_CONTENT", ctx, callParameters, stringBuffer);
            }
        }
    }
}

```

Notice this contribution declares 2 execution point named BEFORE_PROCESSING_CONTENT and AFTER_PROCESSING_CONTENT. These execution point will served to apply a background color

Steps to contribute to BEFORE_PROCESSING_CONTENT execution point

1. Create fcore with pattern named backgroundBefore
2. Add parameter named "parameter" typed EObject
3. Implement and orchestrate the body method (cf. code after)
4. Contribute to dispatch the pattern on BEFORE_PROCESSING_CONTENT with Priority 1

Pattern code (Jet pattern)

```
<div style='background-color:#C8D4E9;color:black;'>
```

Steps to contribute to AFTER_PROCESSING_CONTENT execution point

1. Create fcore with pattern named backgroundAFTER
2. Add parameter named "parameter" typed EObject
3. Implement and orchestrate the body method (cf. code after)
4. Contribute to dispatch the pattern on AFTER_PROCESSING_CONTENT with Priority 1

Pattern code (Jet pattern)

```
</div>
```

Pattern substitution principle

Capella DocGen Extensible Pattern Implementation details

Capella DocGen
Extensible Pattern

CapellaDocGenExtensible

Specification

Inheritance

Choose the super pattern:

Parent: No parent

Pattern Nature

Select the kind of the pattern:

Type:

Parameters

Define parameters for this pattern in the following section.

Name	Type	Query
id	String	
element	EObject	

Overview
Specification
Implementation

Capella DocGen
Extensible Pattern

Methods

Pattern methods:

- header
- init
- preCondition
- footer

Implementation methods:

- content

Variables

Set up some variable available in all methods:

Name	Type

Orchestration

Organize method calls:

- content - [MethodCall]

Overview Specification Implementation

This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2013 – All rights reserved.

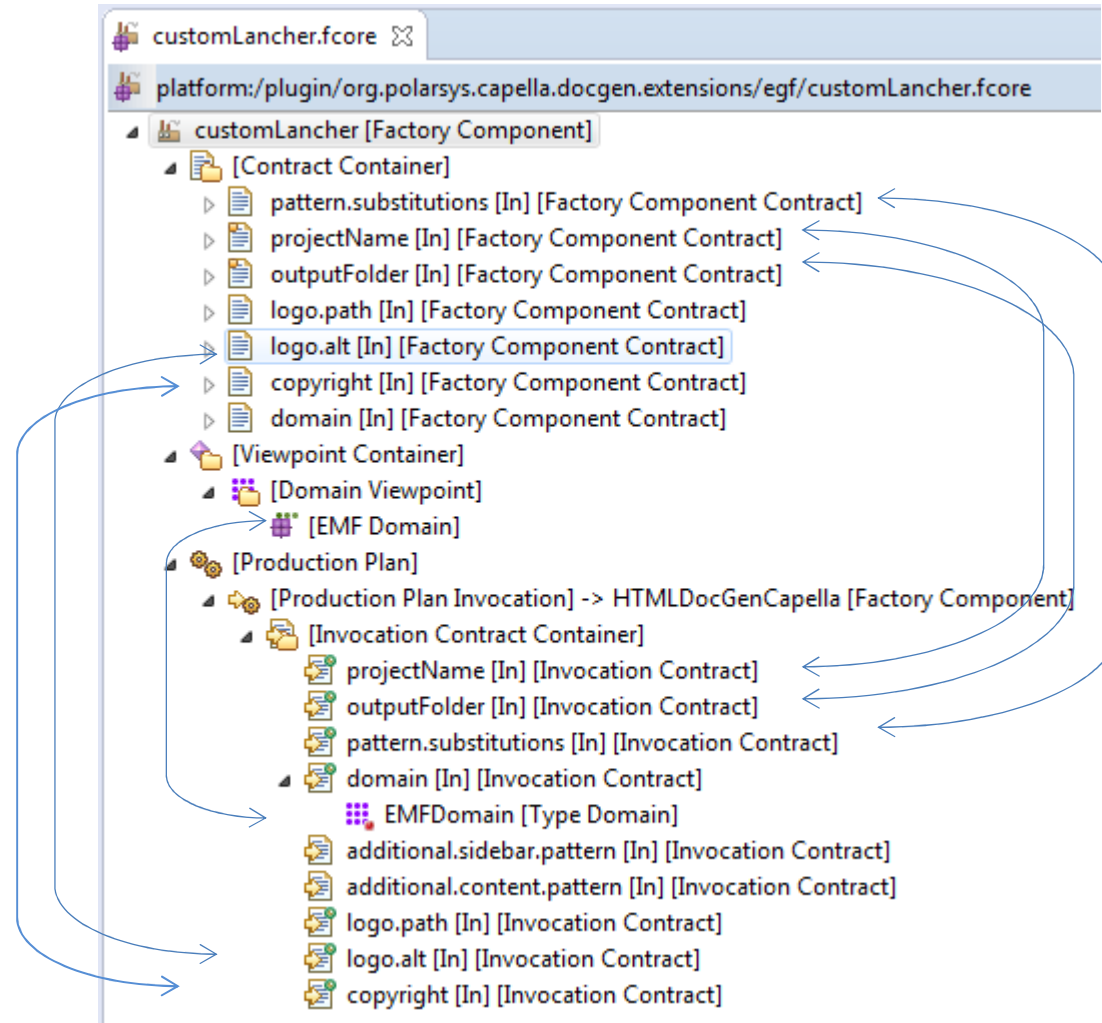
Capella DocGen – Contribution steps

Step 1: Content Pattern Specification

- Create a new EGF Factory Component
- Create a Viewpoint Container
- Create a Pattern Viewpoint
- Create a Pattern Library
- Create a new JET or Java Pattern (e.g., PartMassViewpoint)
- Create a method(s) (e.g., content) and add it to the orchestration
- Implement the business of the methods
- Add preCondition specifying where/when to contribute

Step 2: Launcher Definition

- Create a new EGF Factory Component (e.g, CustomLauncher)
- Create contract Container
- Create Factory Component Contracts, at least for:
 - Project Name (projectName)
 - Output Folder (outputFolder)
 - Pattern substitutions (pattern.substitutions)
- Substitute CapellaDocGenExtensible by pattern previously created and with itself
- Create a Viewpoint Container
- Create a Domain Viewpoint
- Create a EMF Domain
- Add Production Plan
- Invoke HTMLDocGenCapella factory
- Connect contracts of HTMLDocGenCapella with launcher's contracts
- Contribute to “org.polarsys.capella.docgen.launcher” to register the new launcher



This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2013 – All rights reserved.

Examples

Objective

- Add Basic Mass and Basic Price to Physical Component

Result

Mass Viewpoint
Current mass: 3 Max mass: 2
Price Viewpoint
Price: 18 Max price: 30

This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2013 – All rights reserved.

PartMassViewpoint

Specification

Inheritance
Choose the super pattern:
Parent: No parent
Browse [X]

Pattern Nature
Select the kind of the pattern:
Type: JetNature

Parameters
Define parameters for this pattern in the following section.

Name	Type	Query
id	String	
element	EObject	

Mandatory Parameters

Overview Specification Implementation

Implementation

Methods

Pattern methods:

- header
- init
- preCondition
- footer

Implementation methods:

- content

Variables

Set up some variable available in all methods:

Name	Type

Orchestration

Organize method calls:

- content - [MethodCall]

Overview | Specification | Implementation

preCondition method

```

customLancher.fcore  massvp.fcore  PartMassViewpoint  PartMassViewpoint
1 return ID.PART_AFTER_END_CONTENT.equals(id);
header  init  preCondition  content  footer

```

Content method

```

PriceViewpoint  PriceViewpoint  PartMassView...  org.polarsy...  PartMassView...
1 <%
2 int currentMass = 0;
3 int maxValue = 0;
4 if (element instanceof PhysicalComponent) {
5     PhysicalComponent p = (PhysicalComponent) element;
6     EList<AbstractTypedElement> abstractTypedElements = p.getAbstractTypedElements();
7     for (AbstractTypedElement ac : abstractTypedElements) {
8         EList<ElementExtension> ownedExtensions = ac.getOwnedExtensions();
9         for (ElementExtension e : ownedExtensions) {
10            if (e instanceof PartMass) {
11                PartMass p1 = (PartMass) e;
12                currentMass = p1.getValue();
13                maxValue = p1.getMaxValue();
14            }
15            <h2>Mass Viewpoint</h2>
16            Current mass: <%=currentMass%><br/>
17            Max mass: <%=maxValue%>;
18        }
19    }
20 }
21 }
22 %>
header  init  preCondition  content  footer

```


PriceViewpoint

Specification

Inheritance

Choose the super pattern:
Parent: No parent

Browse X

Pattern Nature

Select the kind of the pattern:
Type: JavaNature

Parameters

Define parameters for this pattern in the following section.

Name	Type	Query
id	String	
element	EObject	

Mandatory Parameters

Overview Specification Implementation

The screenshot shows a software development tool interface for implementing a Price Viewpoint pattern. The interface is divided into three main sections: Methods, Variables, and Orchestration.

Methods: This section is split into two columns. The left column, labeled "Pattern methods:", contains a list of methods: `header`, `init`, `preCondition`, and `footer`. The right column, labeled "Implementation methods:", contains a list with one method: `content`. To the right of the implementation methods list are several control buttons: a green plus sign (+), a pencil icon, a hand icon, a printer icon, a red X icon, an up arrow icon, and a down arrow icon.

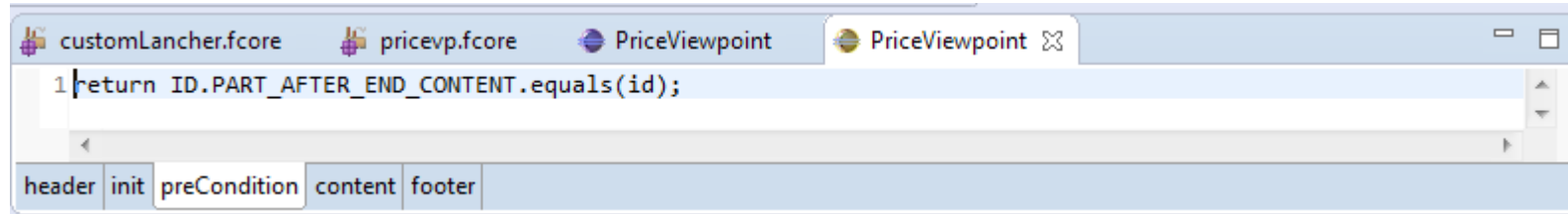
Variables: This section is titled "Variables" and contains the instruction "Set up some variable available in all methods:". Below this is a table with two columns: "Name" and "Type". The table is currently empty. To the right of the table are three control buttons: a green plus sign (+), a pencil icon, and a red X icon.

Orchestration: This section is titled "Orchestration" and contains the instruction "Organize method calls:". Below this is a large text area containing the text `content - [MethodCall]`. To the right of this area are five control buttons: a green plus sign (+), a pencil icon, a red X icon, an up arrow icon, and a down arrow icon.

At the bottom of the interface, there is a navigation bar with three tabs: "Overview", "Specification", and "Implementation". The "Implementation" tab is currently selected.

This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2013 – All rights reserved.

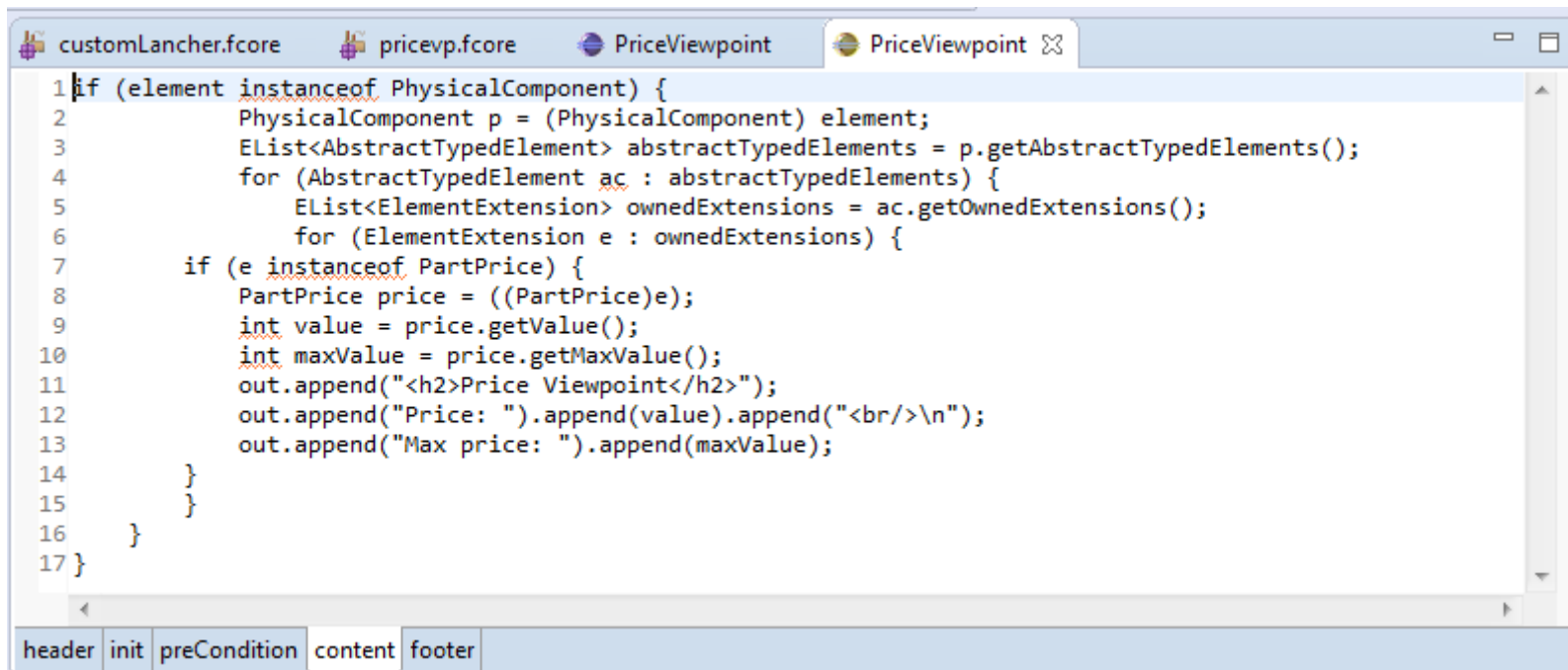
preCondition method



```
1 return ID.PART_AFTER_END_CONTENT.equals(id);
```

The screenshot shows an IDE window titled 'PriceViewpoint' with a tab for 'pricevp.fcore'. The code editor displays a single line of code: `1 return ID.PART_AFTER_END_CONTENT.equals(id);`. Below the editor is a breadcrumb navigation bar with tabs for 'header', 'init', 'preCondition', 'content', and 'footer', where 'preCondition' is currently selected.

Content method



```
1 if (element instanceof PhysicalComponent) {
2     PhysicalComponent p = (PhysicalComponent) element;
3     EList<AbstractTypedElement> abstractTypedElements = p.getAbstractTypedElements();
4     for (AbstractTypedElement ac : abstractTypedElements) {
5         EList<ElementExtension> ownedExtensions = ac.getOwnedExtensions();
6         for (ElementExtension e : ownedExtensions) {
7             if (e instanceof PartPrice) {
8                 PartPrice price = ((PartPrice)e);
9                 int value = price.getValue();
10                int maxValue = price.getMaxValue();
11                out.append("<h2>Price Viewpoint</h2>");
12                out.append("Price: ").append(value).append("<br/>\n");
13                out.append("Max price: ").append(maxValue);
14            }
15        }
16    }
17 }
```

The screenshot shows an IDE window titled 'PriceViewpoint' with a tab for 'pricevp.fcore'. The code editor displays the following code:

```
1 if (element instanceof PhysicalComponent) {
2     PhysicalComponent p = (PhysicalComponent) element;
3     EList<AbstractTypedElement> abstractTypedElements = p.getAbstractTypedElements();
4     for (AbstractTypedElement ac : abstractTypedElements) {
5         EList<ElementExtension> ownedExtensions = ac.getOwnedExtensions();
6         for (ElementExtension e : ownedExtensions) {
7             if (e instanceof PartPrice) {
8                 PartPrice price = ((PartPrice)e);
9                 int value = price.getValue();
10                int maxValue = price.getMaxValue();
11                out.append("<h2>Price Viewpoint</h2>");
12                out.append("Price: ").append(value).append("<br/>\n");
13                out.append("Max price: ").append(maxValue);
14            }
15        }
16    }
17 }
```

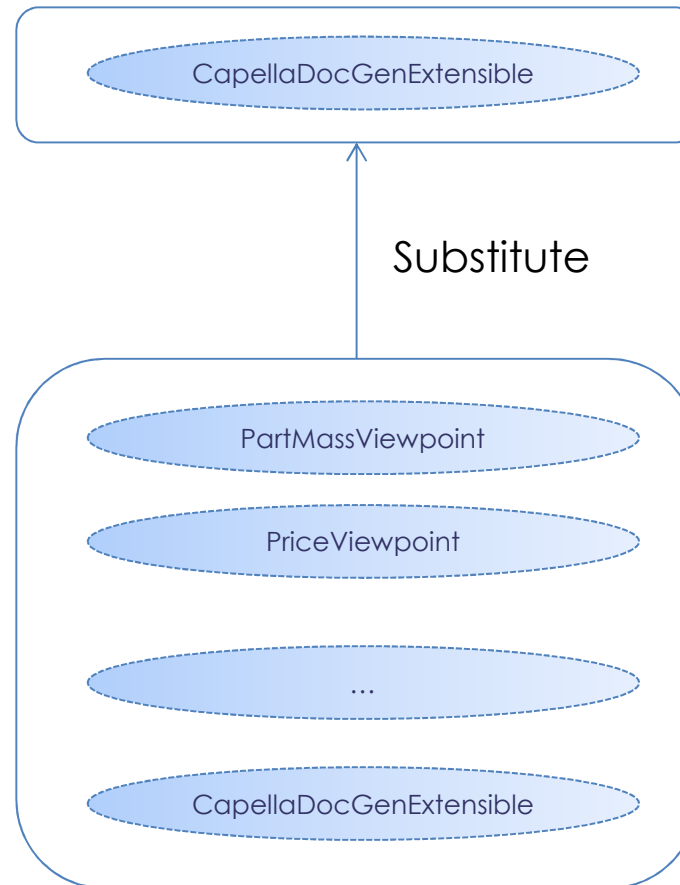
 Below the editor is a breadcrumb navigation bar with tabs for 'header', 'init', 'preCondition', 'content', and 'footer', where 'content' is currently selected.

Substitution of CapellaDocGenExtensible Pattern

The screenshot shows the Capella IDE interface with the following configuration for the custom launcher:

- customLancher [Factory Component]
 - [Contract Container]
 - pattern.substitutions [In] [Factory Component Contract]
 - [Type Pattern Substitution]
 - CapellaDocGenExtensible <- org.polarsys.capella.docgen.extensible.pattern [Library] <- HTMLDocGenCapella [Factory Component] [Substitution]
 - CapellaDocGenExtensible <- org.polarsys.capella.docgen.extensible.pattern [Library] <- HTMLDocGenCapella [Factory Component] [Substitution]
 - CapellaDocGenExtensible <- org.polarsys.capella.docgen.extensible.pattern [Library] <- HTMLDocGenCapella [Factory Component] [Substitution]
 - projectName [In] [Factory Component Contract]
 - outputFolder [In] [Factory Component Contract]
 - logo.path [In] [Factory Component Contract]
 - logo.alt [In] [Factory Component Contract]
 - copyright [In] [Factory Component Contract]
 - domain [In] [Factory Component Contract]
 - [Viewpoint Container]
 - [Domain Viewpoint]
 - [EMF Domain]
 - [Production Plan]
 - [Production Plan Invocation] -> HTMLDocGenCapella [Factory Component]

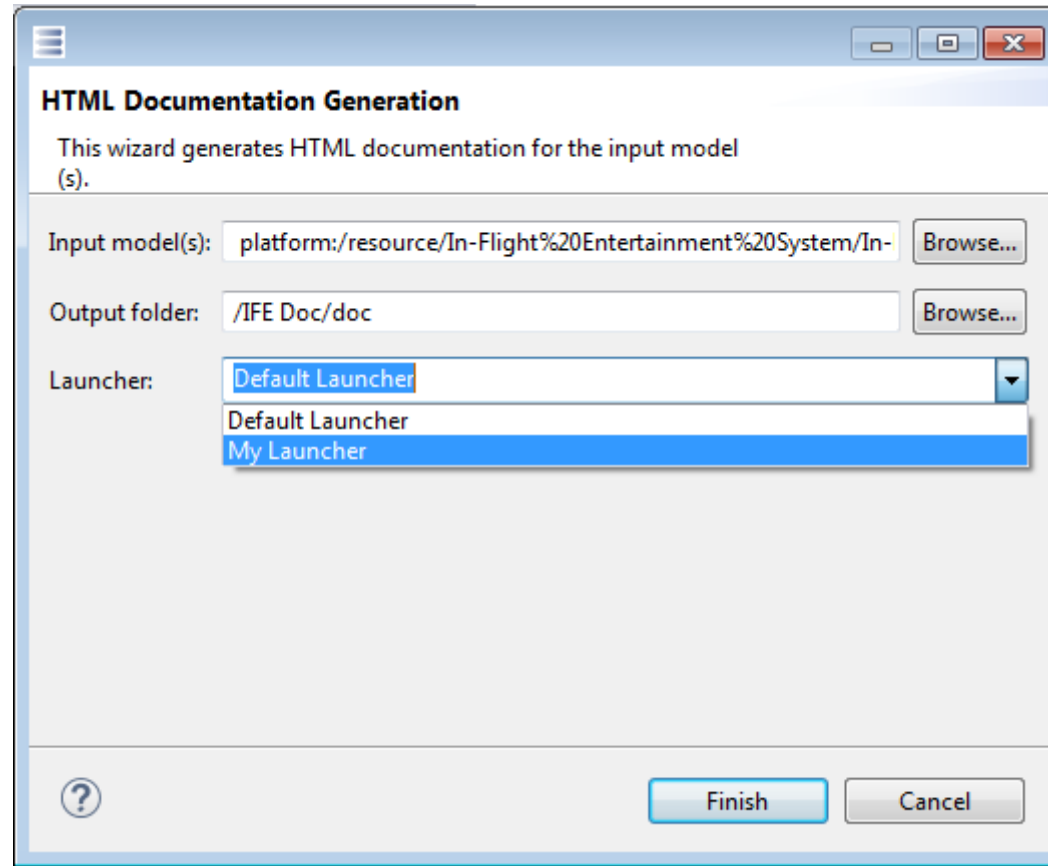
Substitution of CapellaDocGenExtensible Pattern



Contribution to Capella docGen Launcher

The screenshot shows the Eclipse IDE's 'Extensions' dialog box. The 'All Extensions' section on the left lists the installed extensions, including 'org.eclipse.egf.core.fcore' and 'org.polarsys.capella.docgen.ui.launcher'. Under 'org.polarsys.capella.docgen.ui.launcher', the extension 'My Launcher (launcher)' is selected. The 'Extension Element Details' section on the right shows the configuration for the 'launcher' extension. The 'uri*' field is set to 'platform:/plugin/org.polarsys.capella.docgen.extensions/egf/customLancher.fcore#_hJdNODO-Eea_XMcxmay1Q' and the 'name' field is set to 'My Launcher'. The 'uri*' field is marked as required with an asterisk.

This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2013 – All rights reserved.



Disjunction

Two contributions work on different purposes for different outputs.
Example: Class and Package contributions

Complementarity

Two contributions work on the different purposes for the same output.
Example: Class Attributes and Class Operations contributions

Concurrently

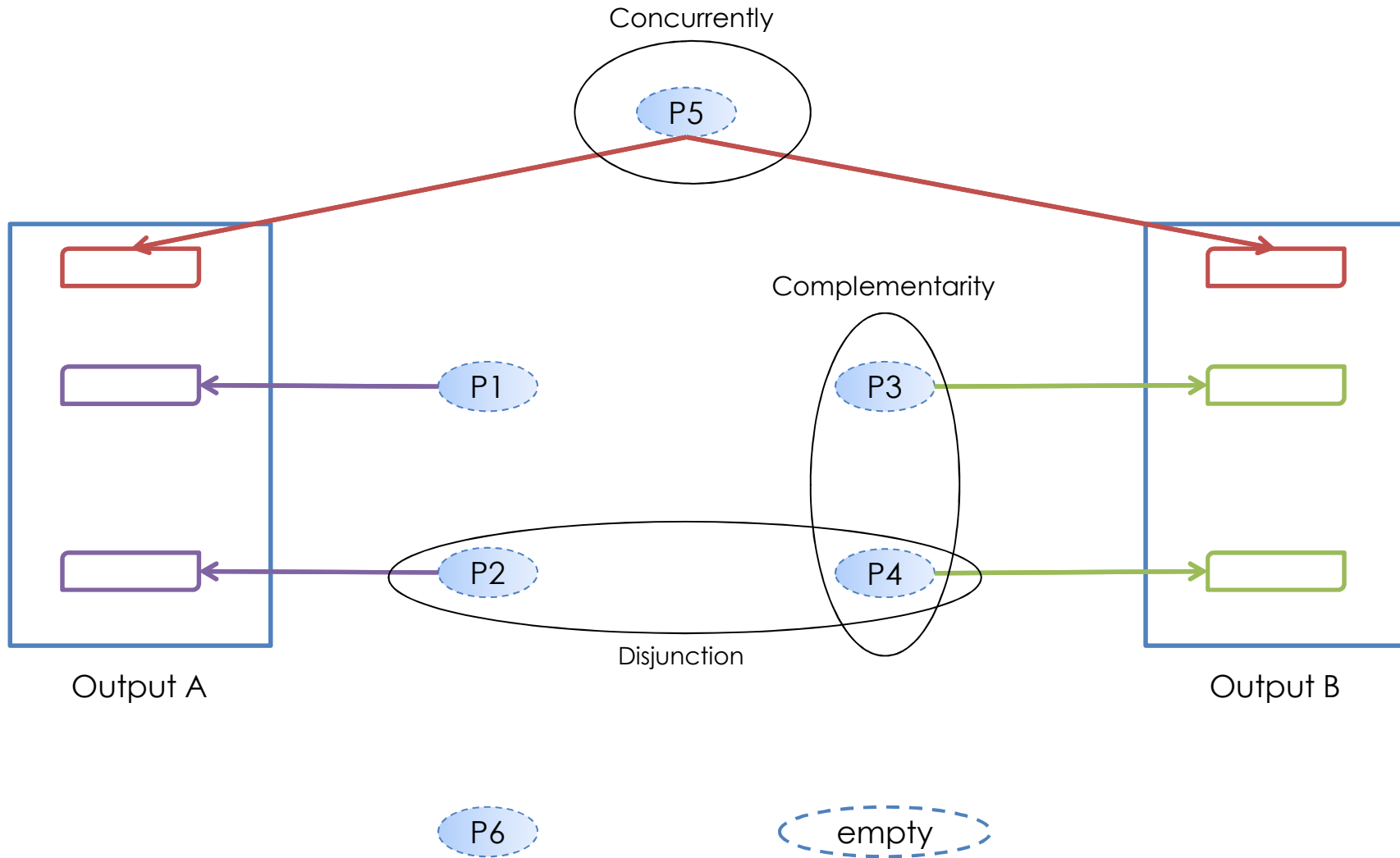
A Contribution works on a purpose for subset or whole of output.
Example: adding copyright before header of all output pages

Replacement

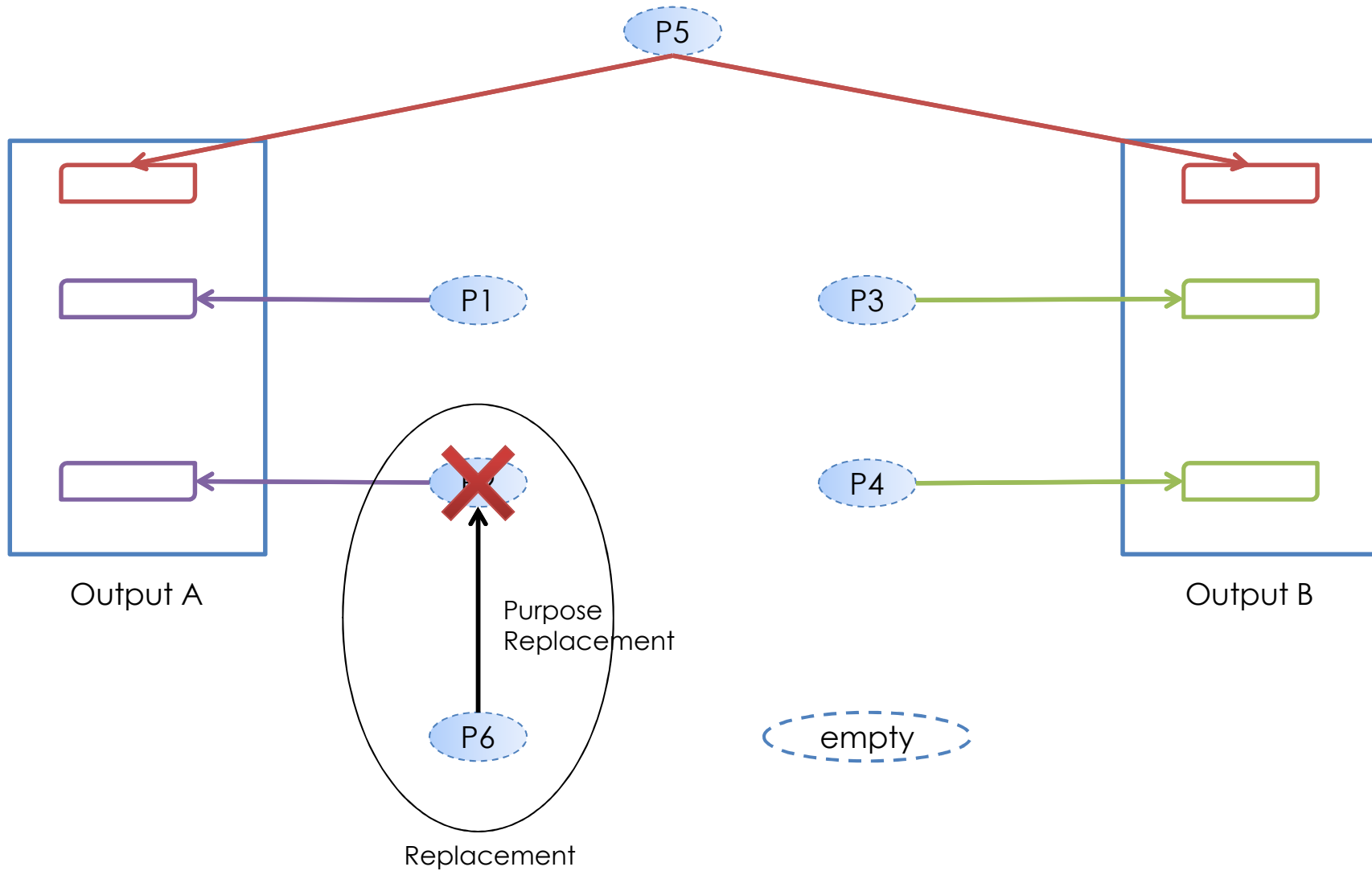
A Contribution replaces another one. Replacement may be the order of the production or the purpose of the production
Example: adding copyright after last section of Class pages

Annihilation

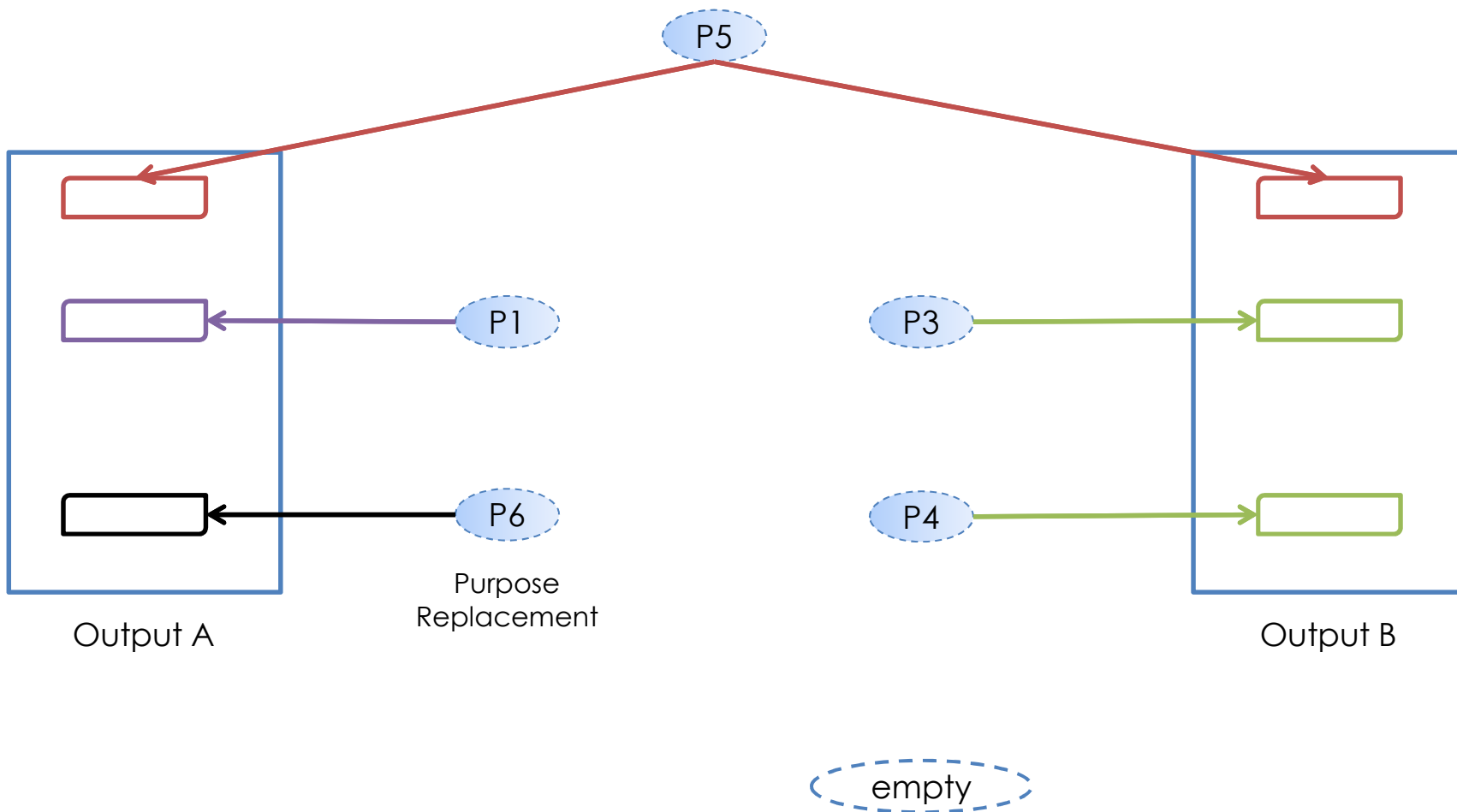
A contributions erases the effect of another one.
Example : the body of a Section is empty



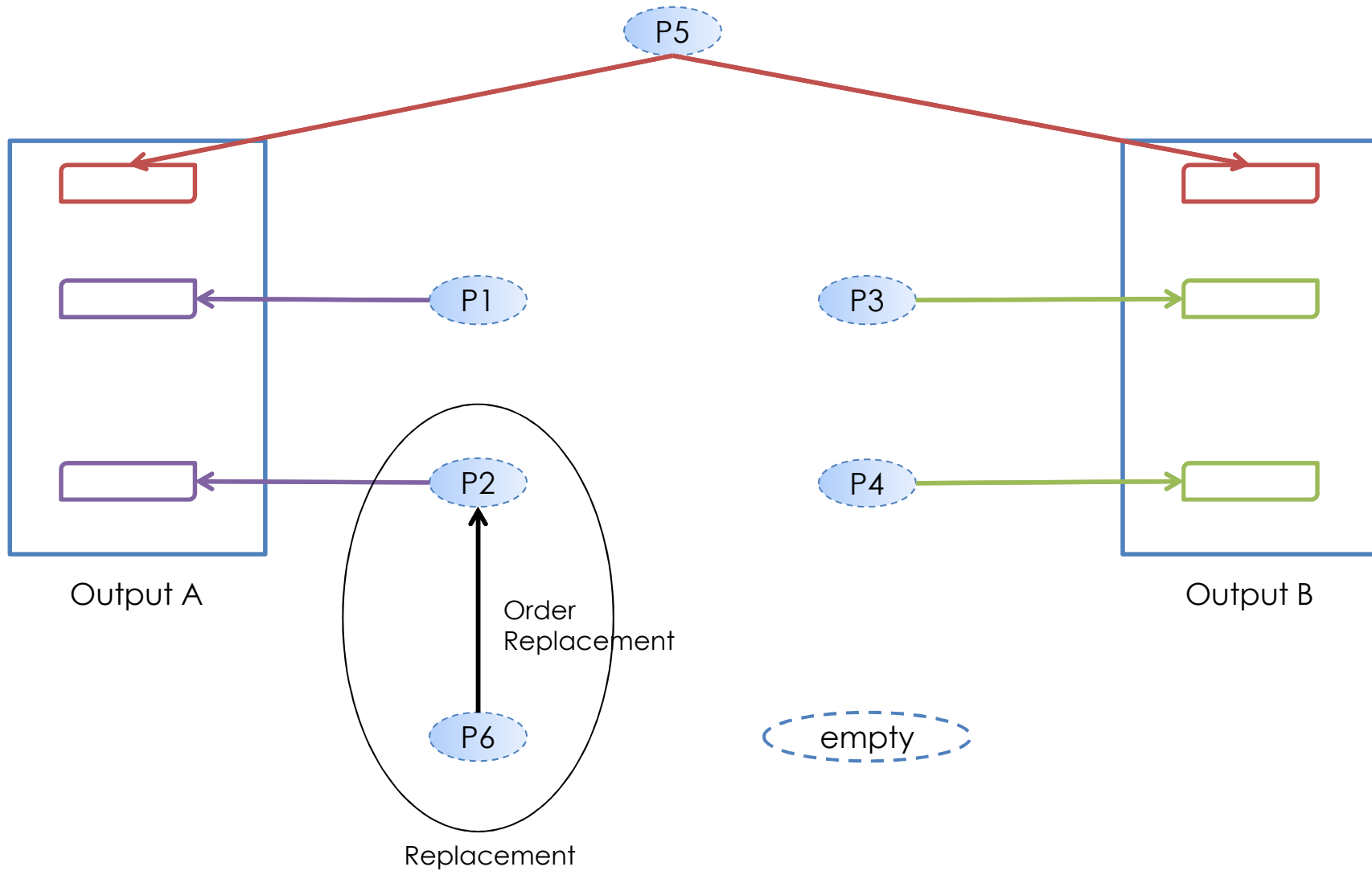
This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2013 – All rights reserved.



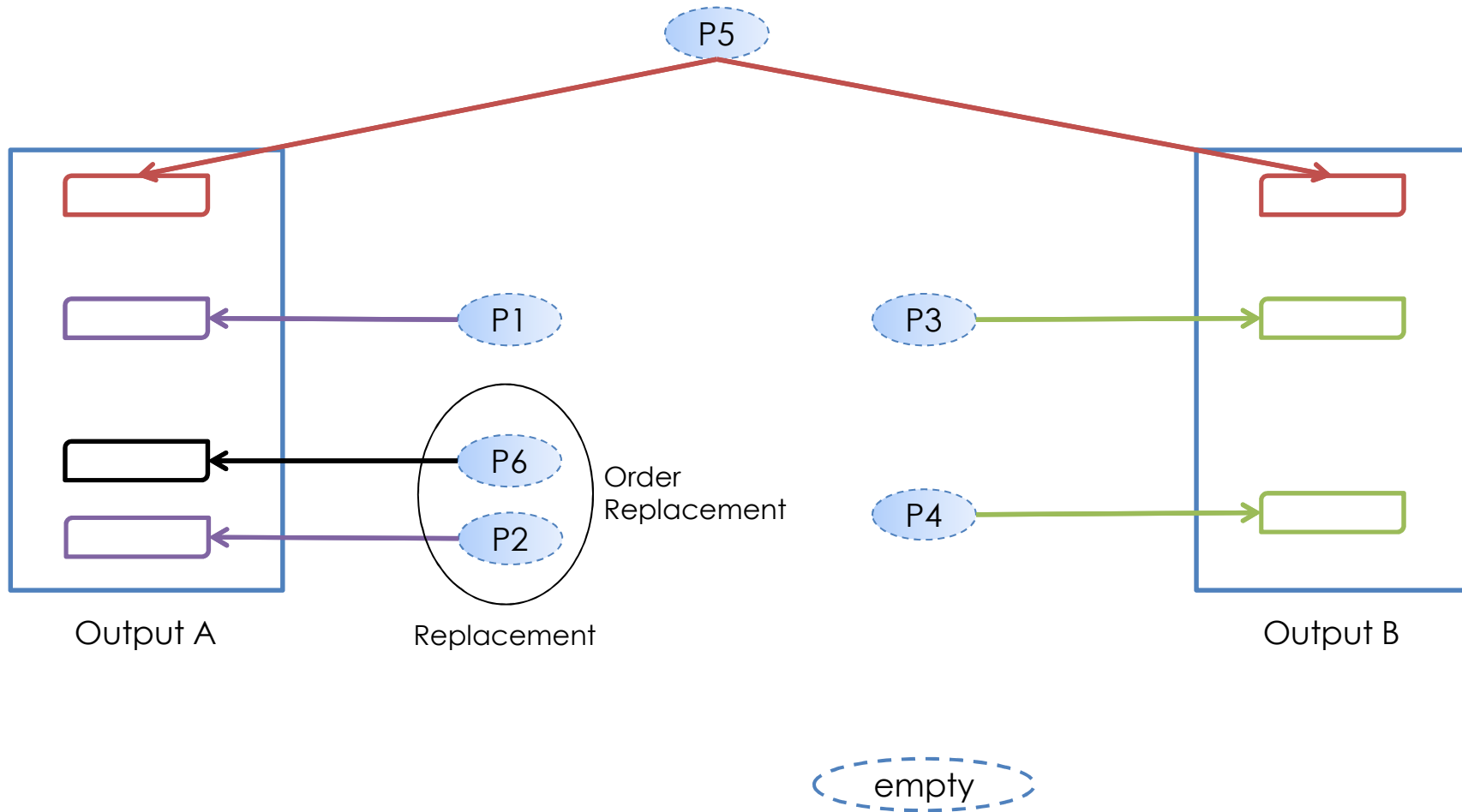
This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. ©THALES 2013 – All rights reserved.



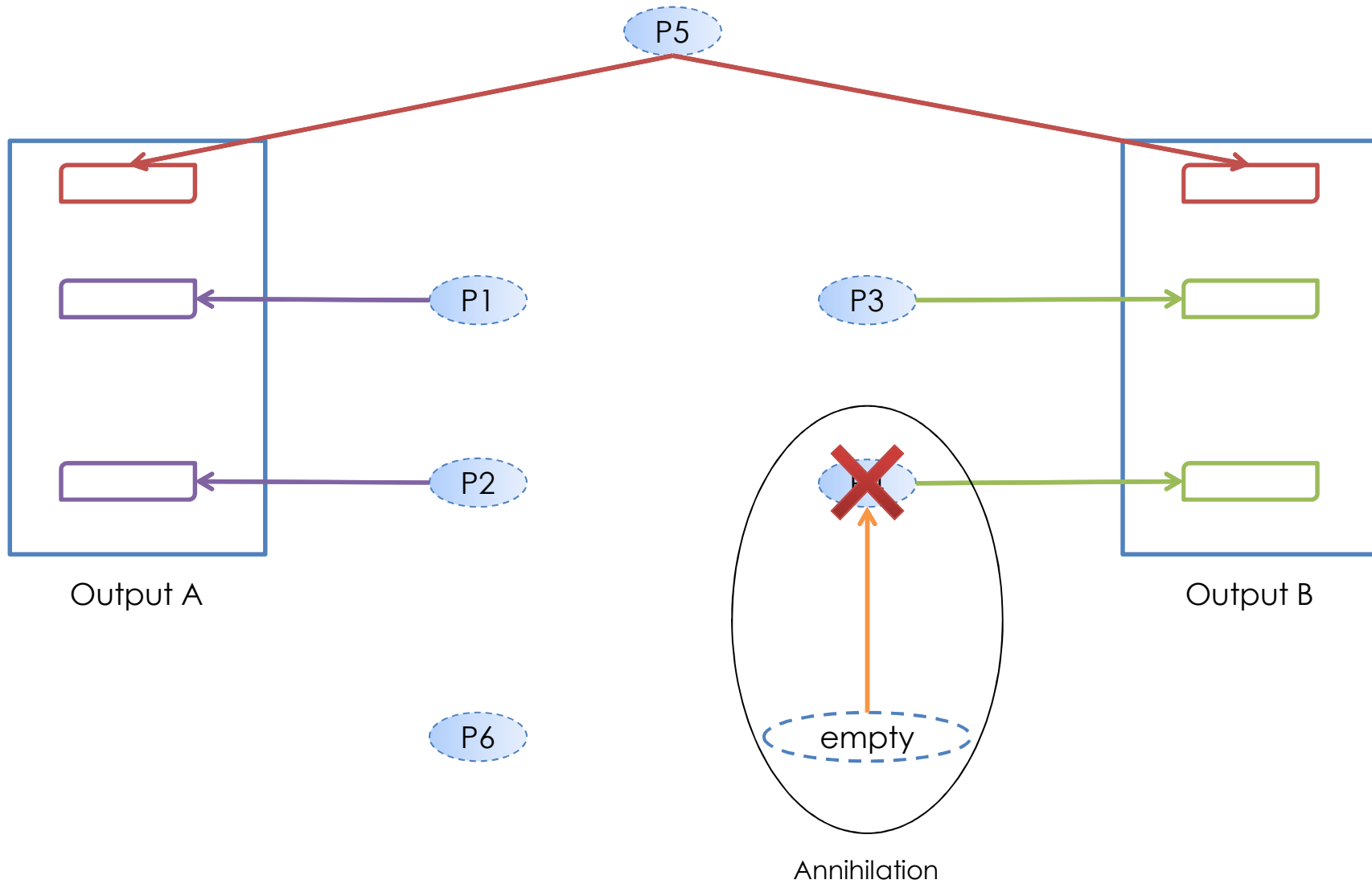
This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2013 – All rights reserved.



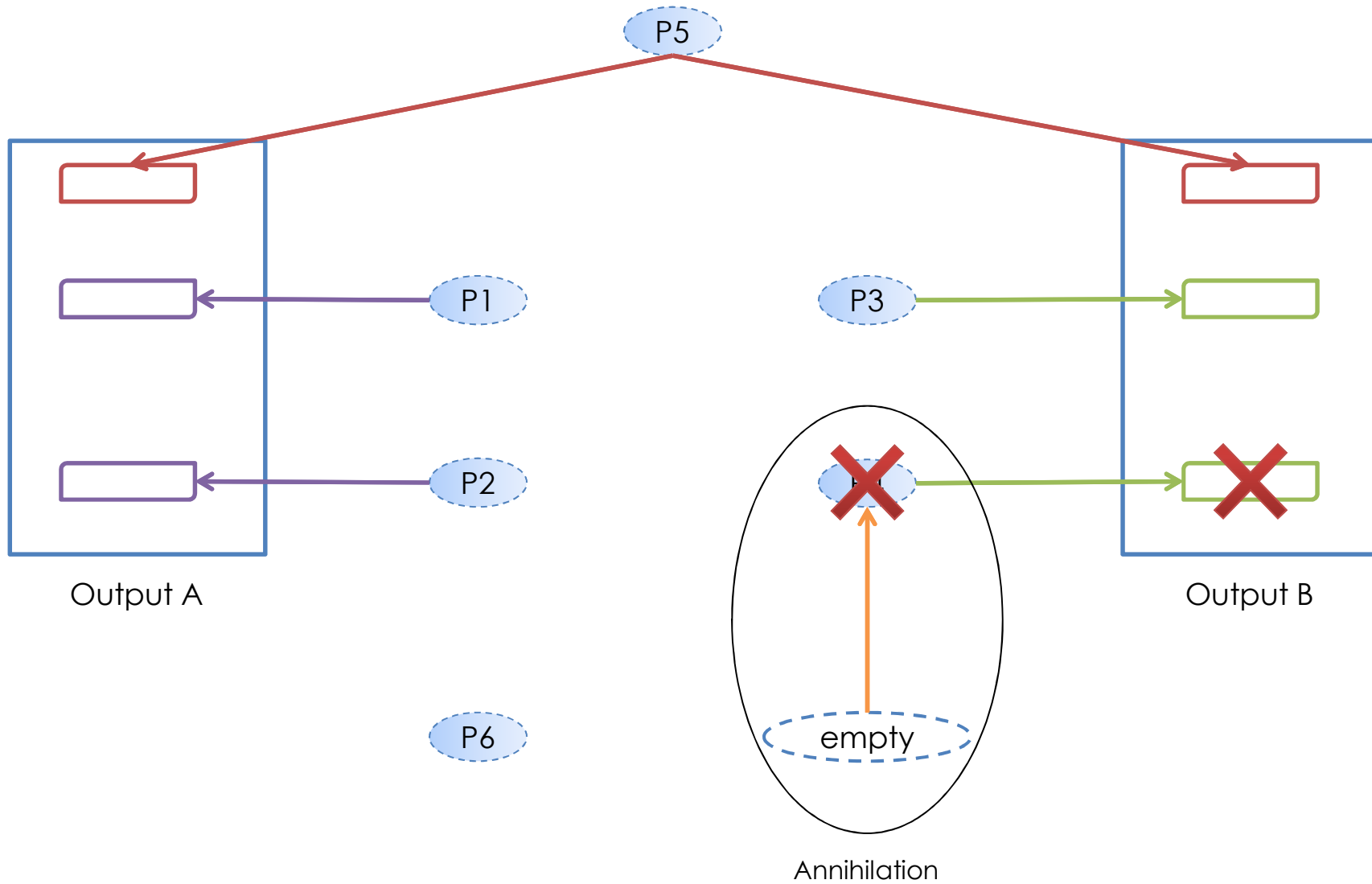
This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2013 – All rights reserved.



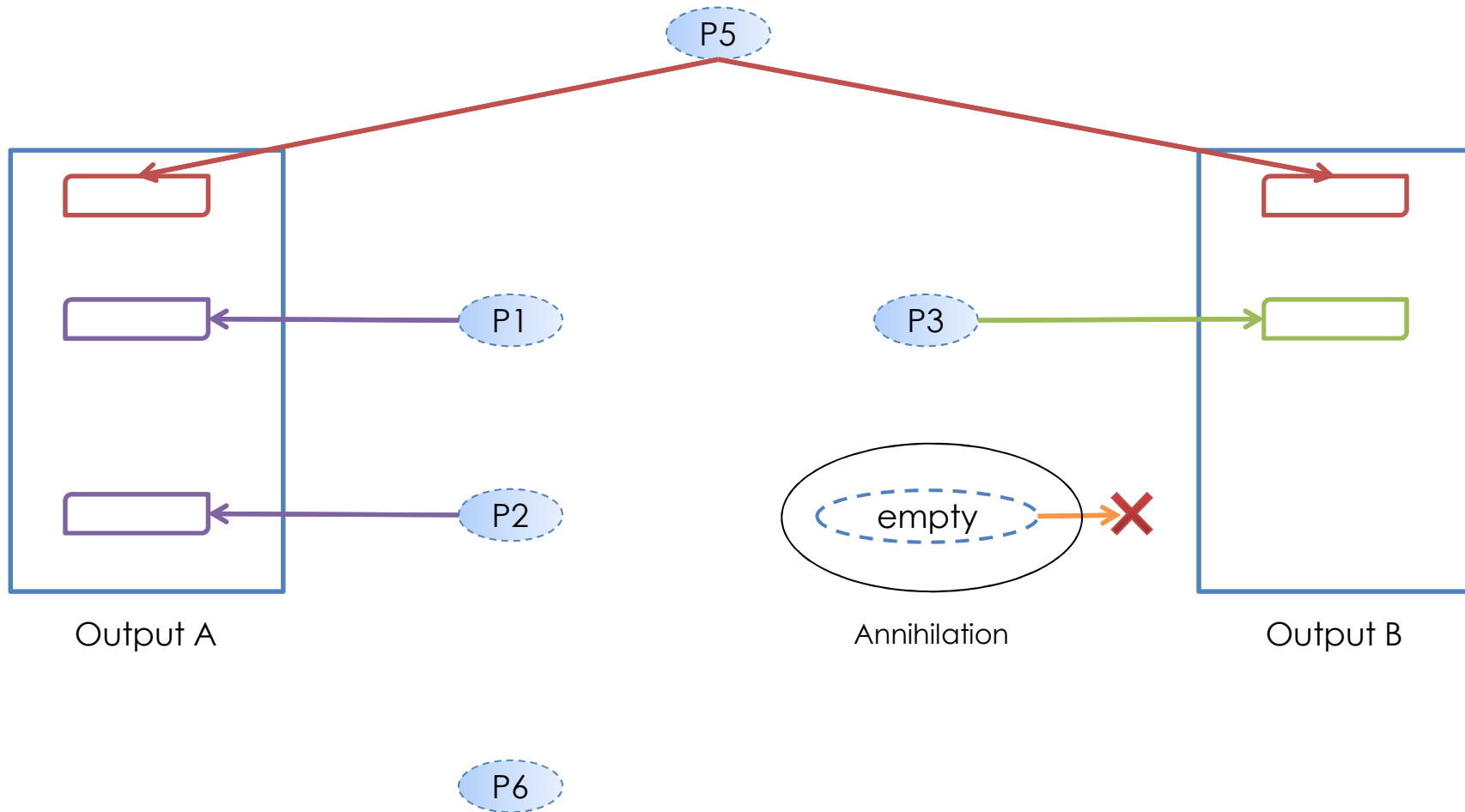
This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2013 – All rights reserved.



This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2013 – All rights reserved.

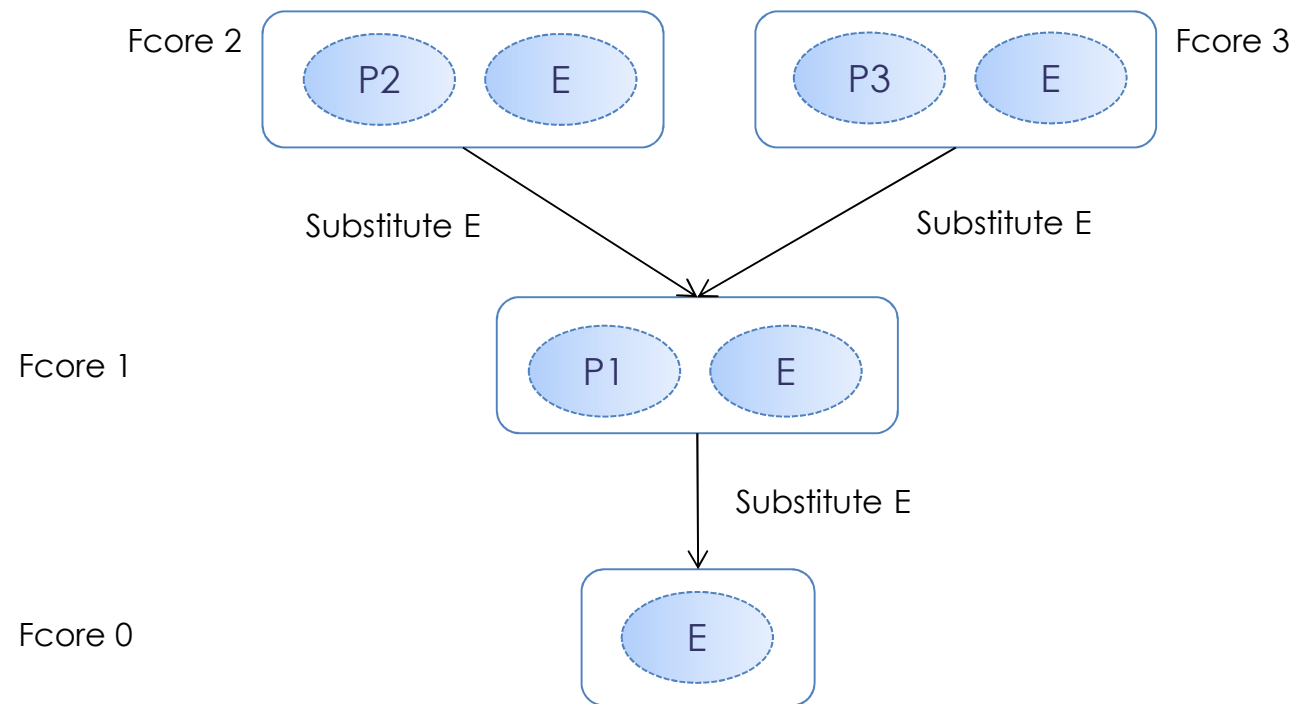


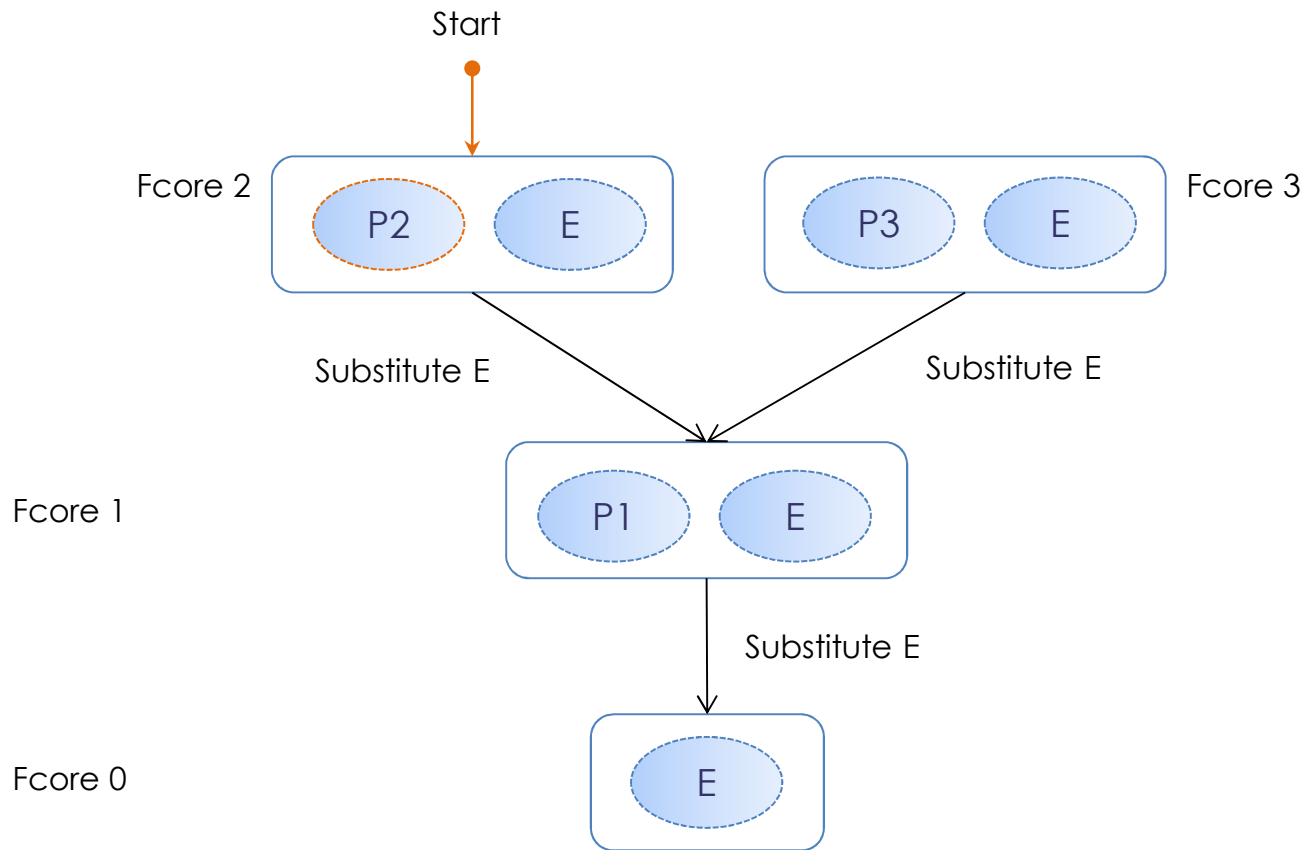
This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2013 – All rights reserved.



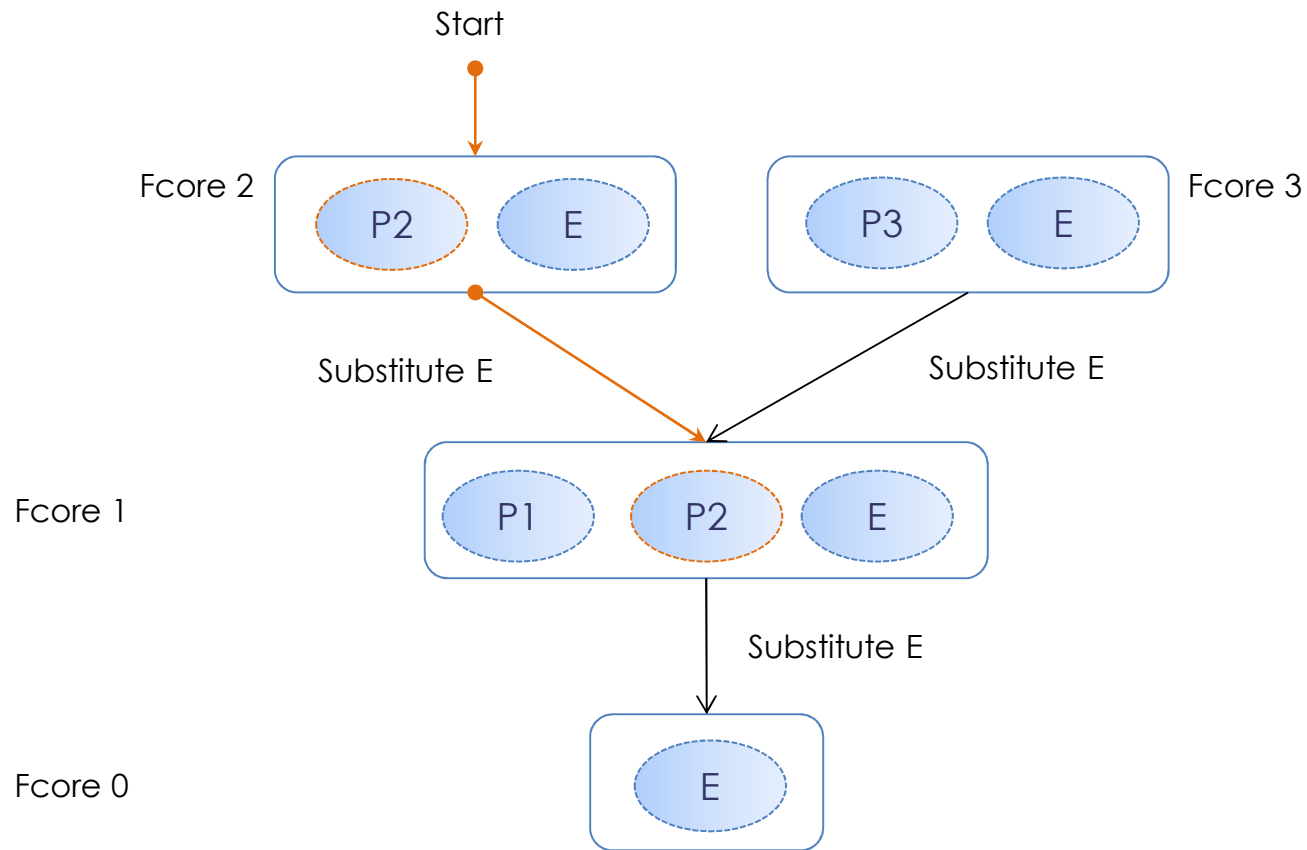
This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2013 – All rights reserved.

- Add always the empty pattern in the list of replacement patterns

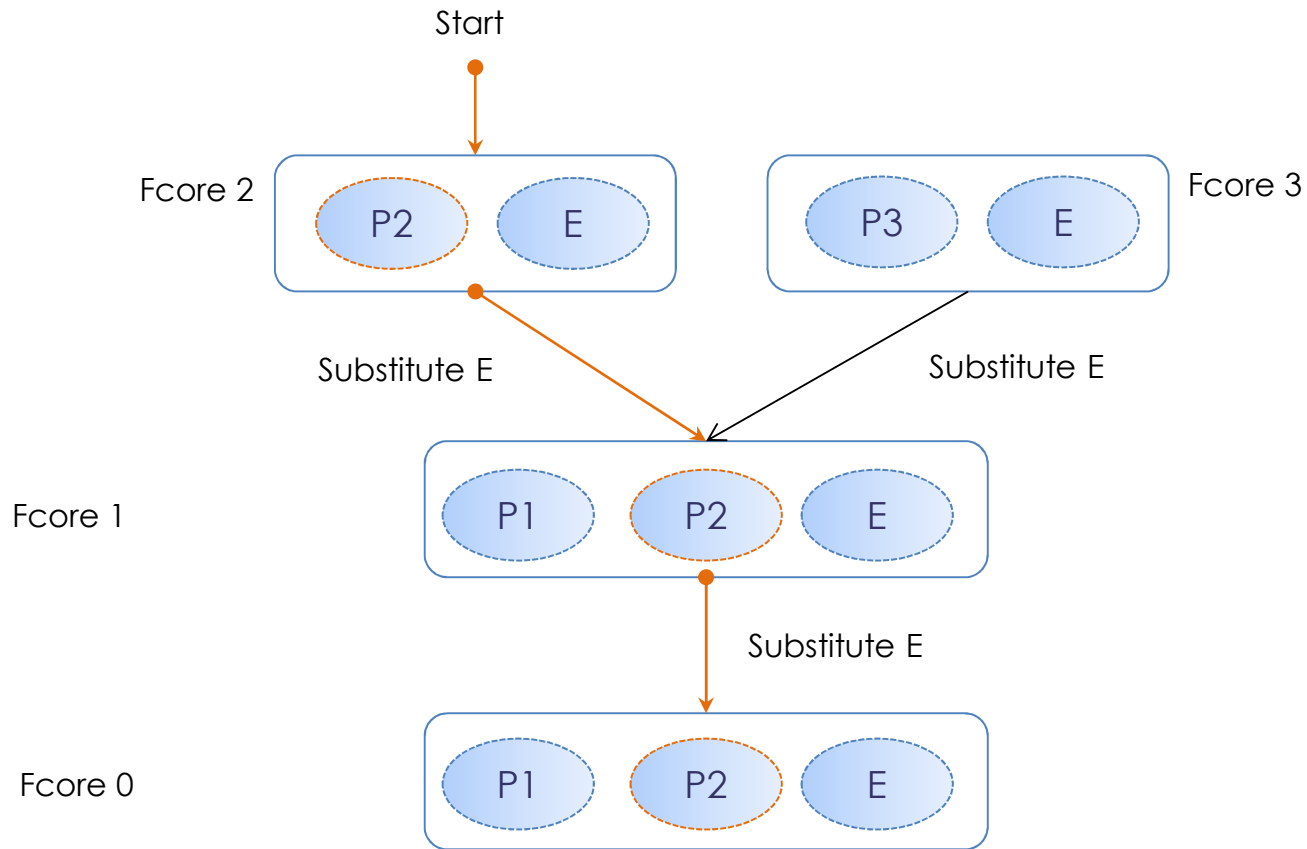




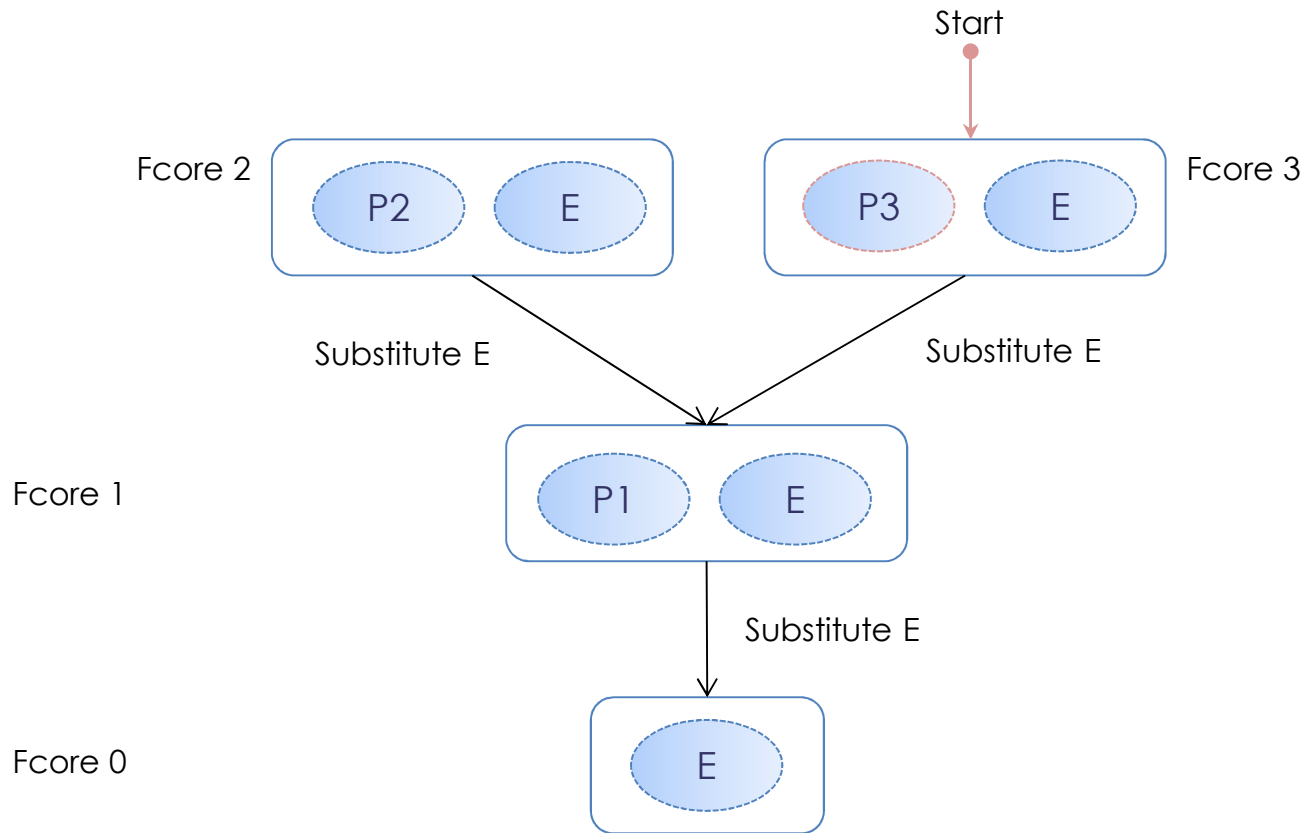
This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. ©THALES 2013 – All rights reserved.



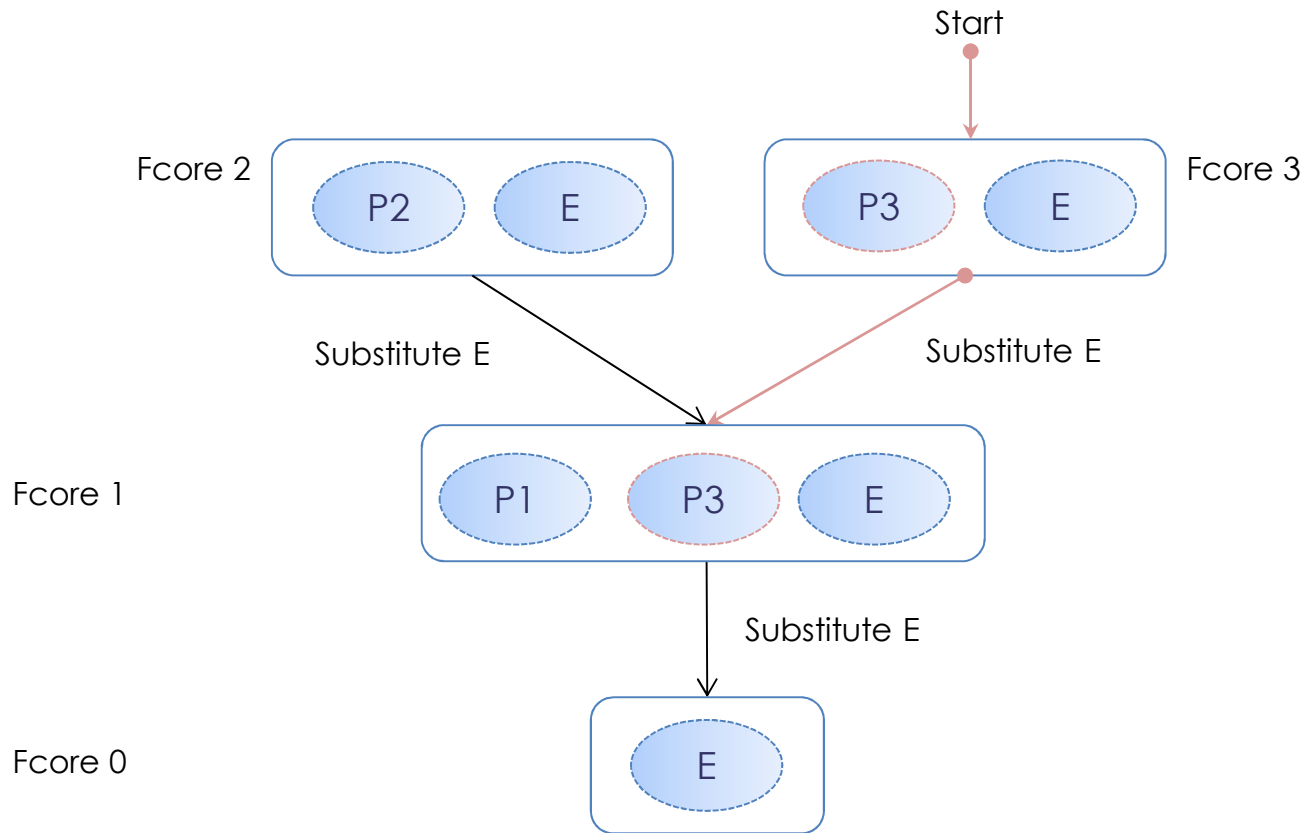
This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. ©THALES 2013 – All rights reserved.



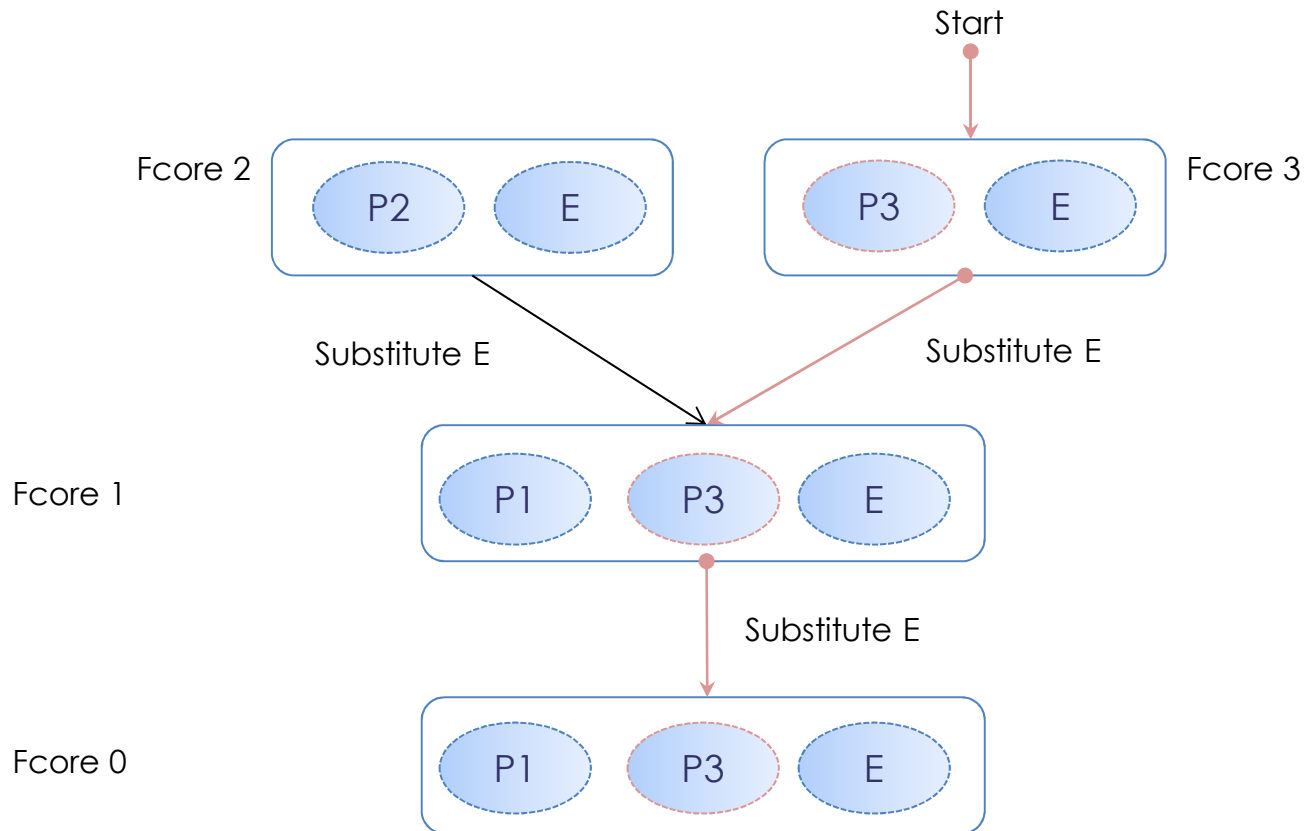
This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2013 – All rights reserved.



This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2013 – All rights reserved.



This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2013 – All rights reserved.



This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. ©THALES 2013 – All rights reserved.



Initiative supported by **Clarity**,
a French collaborative project
<http://www.clarity-se.org/>



<http://polarsys.org/kitalpha/>

<http://polarsys.org/capella/>

This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2013 – All rights reserved.