Decoration Service Architecture

Editor	Amine EL KOUHEN - CEA LIST / LIFL DaRT
Status	Draft
Version number	0.1
Date of preparation	23/05/11

© 2011 The Papyrus Consortium

Authors

Authors name (first/last name)	Company	E-mail	Initial
Amine EL KOUHEN	CEA LIST / INRIA DaRT	Amine.Elkouhen@cea.fr	AE

© 2011 The Papyrus Consortium

Revision chart and history log

Version	Date	Reasons
0.1	23/05/11	Initial contribution.

© 2011 The Papyrus Consortium

Table of contents

Decoration Service Architecture

© 2011 The Papyrus Consortium

1 Introduction

In the Papyrus project, we need decorate (on several occasions) the graphical elements of the editor, with icons, figures .. for mark them or simply distinguish them from each other. Hence the idea of creating a dynamic and centralized service independent of the target technology (element to decorate), to do that.

This document aims to describe the architecture of Decoration Service and plugins used to implement it.

This Tutorial is based on Eclipse Modeling Tools 'Indigo M5' version 3.7.0 of the Eclipse SDK, version 2.7.0 of EMF, version 1.4.0 of EMF Validation Framework and version 2.4.0 of GMF.

© 2011 The Papyrus Consortium

2 Goal

The goal of this service is to centralize the creation of decorations, on graphical elements of the editor, in order to mark them for different reasons :

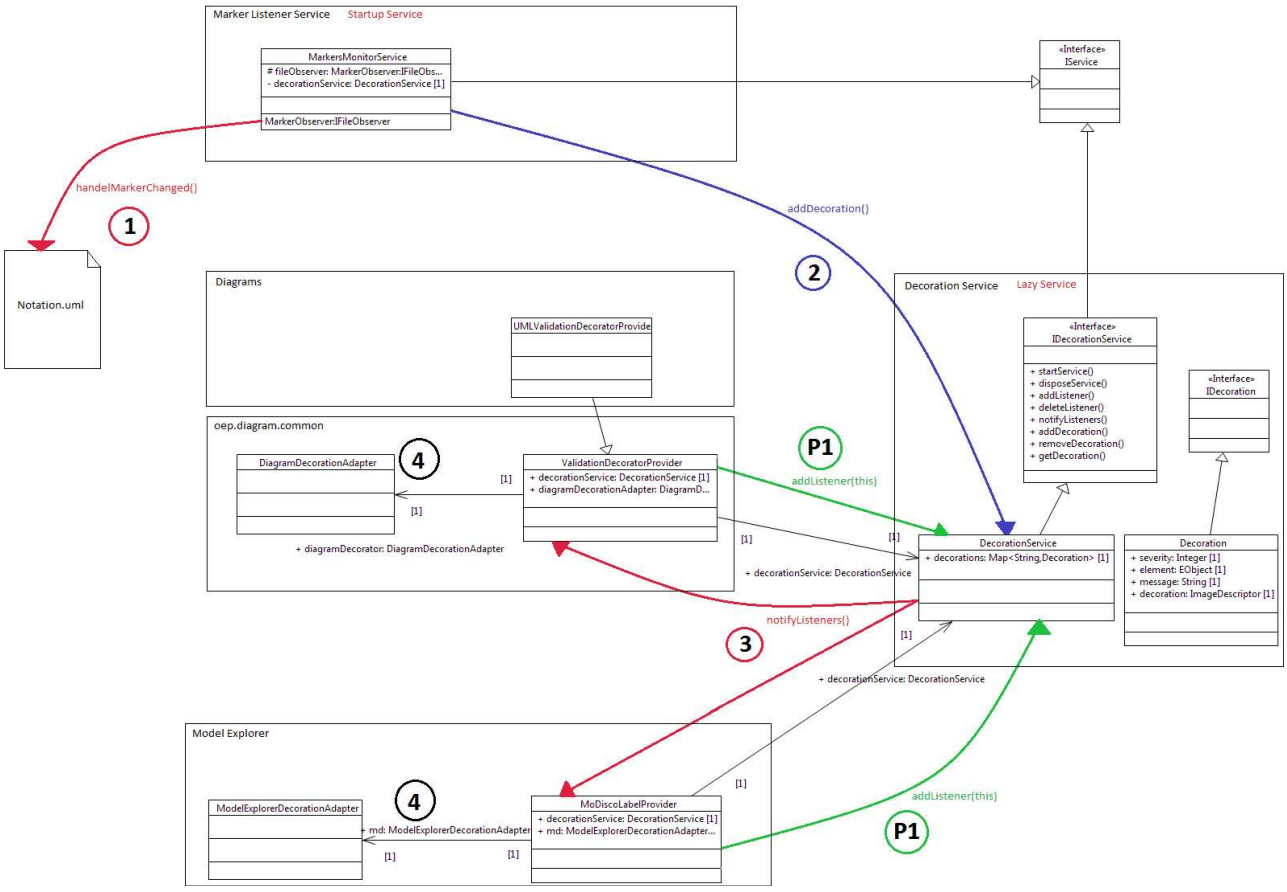
- Validation : Mark errors and warnings on UML elements
- Name checking : Mark errors on elements name
- OCL Checking : Mark elements with constraints
- Decorate the navigable elements (Navigation between diagrams)
- ...

Decoration service is required in various views in the Papyrus modeler.

The service provides a mechanism to register different views as a service customers :

- Model explorer
- Diagrams Editors

The purpose of this service is to create a decoration (2) whenever a marker was created (or changed) : (1) in the file NOTATION.UML. This decoration will then be sent (3) to different service listeners (already registered) (P1) and adjust (4) according to the type of decoration on the marked element in the notation file.



We combine the decoration Service on another service of file monitoring (MarkerMonitorService), this startup service aims to catch changes (1) in notation file (additions or changes of Markers) and add a decoration (2) to the Decoration service.

each client's decorating service must be registered as an listener at the launch of the application (Pretreatment 1), when the service receives a new decoration (2), it informs all registered listeners (3), thus decoration will be sent to service customers and adapted to the target technology (4).

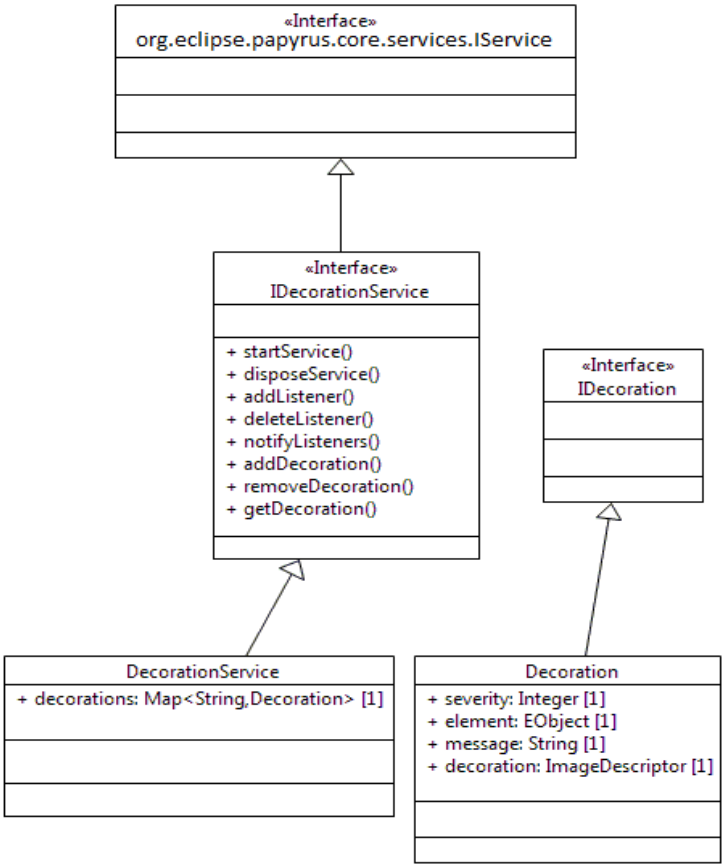
Tip : As a “lazy” service, the decoration service will be started when it receives a new decoration !

© 2011 The Papyrus Consortium

3 Architecture

This service rely on the backbone service system.

The service provides the interface : IDecorationService with the following features :

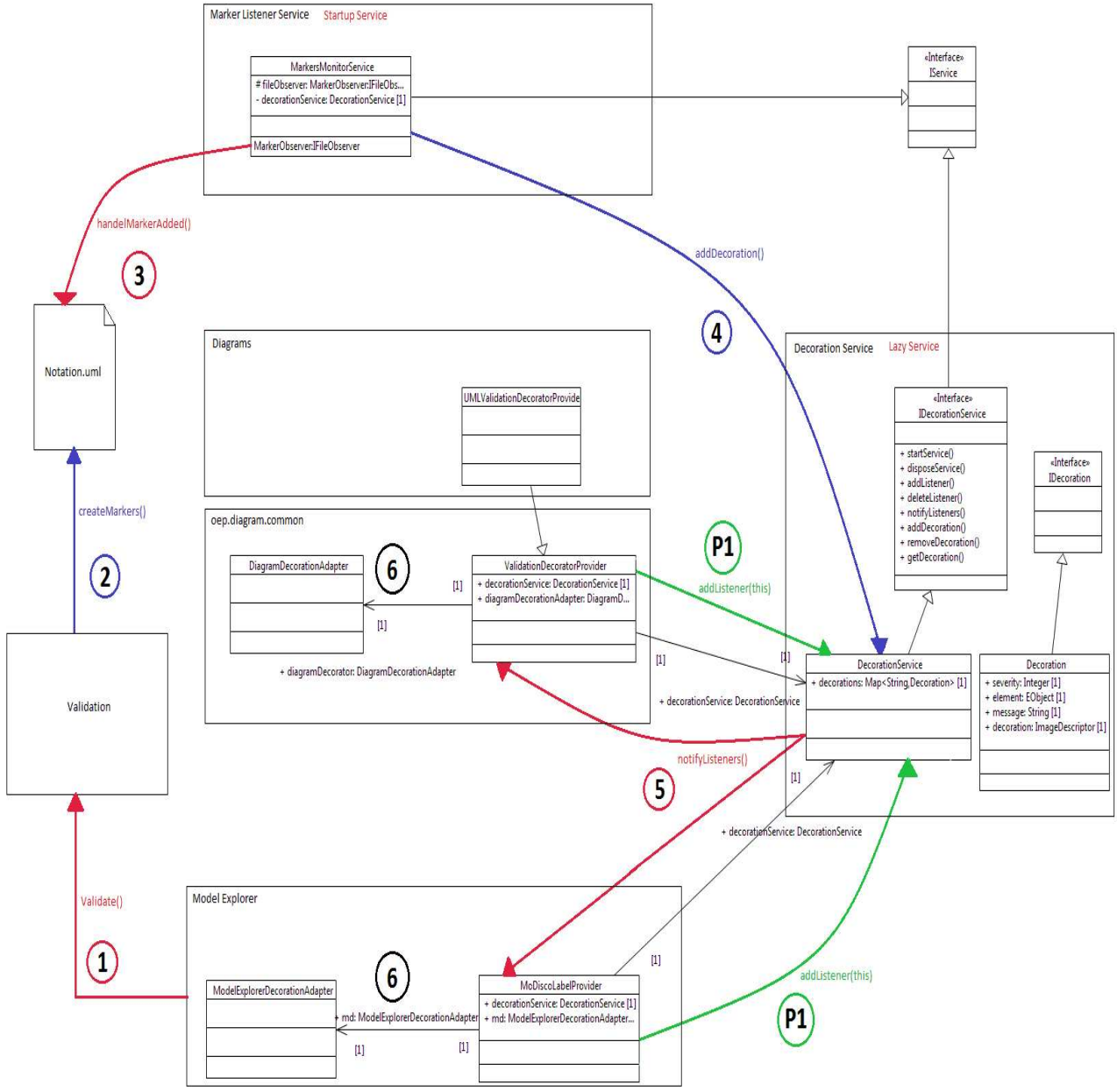


- startService() and disposeSerivce() : for the launch and the stop of the service,
- addListener(listener) : for add a new listener (service customer)
- deleteListener(listener) : for remove an existing listener
- notifyListeners() : notifies all registered listeners to refresh the views (informs them that a new decoration was added)
- addDecoration(..) : Add a new Decoration to a decorations Map with different ways :
  - key (the Marker ID), element To Decorate, severity of the Marker, messages of the Marker
  - key, element To Decorate, the image descriptor decoration, message to display
- removeDecoration(key) : remove a decoration from the decorations Map,
- getDecoration(element) : Return an Interface of decoration (Idecoration) for the element

© 2011 The Papyrus Consortium

4 Application

We take Validation process as an application of our service :



Now,the validation process is done in the following way :

- P1 (pretreatment 1): At startup, the consumers of the service, register as listeners,
- 1: The extension mechanism of the eclipse can set up a context menu in the Model Explorer view. We can run the validation from the model explorer or another location.
- 2: The validation plugin validate the selected item in the Model Explorer, this action aims to Marks the erroneous elements in the file notation.uml.
- 3: MarkerMonitorService catches all changes in the notation file.
- 4: When an inaccurate element has been detected by the MarkerMonitorService, a new decoration is created and added to the decoration service,

© 2011 The Papyrus Consortium

- 5 : When a decoration has been created and added, the service informs all listeners to refresh their views
- 6 : Different Listeners recuperates the decoration and adapts it according the displaying technology,

© 2011 The Papyrus Consortium

5 References

[1]

© 2011 The Papyrus Consortium