
	Titre du document : Étude d'architecture du plug-in s'appuyant sur Mylyn et sur le plug-in Eclipse pour Codendi
	Référence : Tâche 3 Livrable 1
	Version du 01/06/11

# Étude d'architecture du plug-in s'appuyant sur Mylyn et sur le plug-in Eclipse pour Codendi


<b>Livrable du au titre du projet</b>	COCLICO
<b>Lot</b>	SP5
<b>Tache</b>	3.1
<b>Livrable</b>	Spécification

<b>Rédacteur(s)</b>	<b>Vérificateur(s)</b>	<b>Approbateur(s)</b>

	Titre du document : Étude d'architecture du plug-in s'appuyant sur Mylyn et sur le plug-in Eclipse pour Codendi
	Référence : Tâche 3 Livrable 1
	Version du 01/06/11


<b>Documents applicables</b>
Annexe technique au projet COCLICO

<b>Documents de références (pour information)</b>

	Titre du document : Étude d'architecture du plug-in s'appuyant sur Mylyn et sur le plug-in Eclipse pour Codendi
	Référence : Tâche 3 Livrable 1
	Version du 01/06/11


### Gestion des versions

N° de version	Date	Auteurs	Modification apportées
0.1	19/08/10	Houssam Fakhri	Document initial

	Titre du document : Étude d'architecture du plug-in s'appuyant sur Mylyn et sur le plug-in Eclipse pour Codendi
	Référence : Tâche 3 Livrable 1
	Version du 01/06/11

## Sommaire

1	Introduction.....	5
2	Méthodologie.....	6
3	Architecture du Plugin .....	7
4	Choix d'architecture.....	8
	Plugin côté eclipse.....	8
	Plugin forge.....	8
	Communication eclipse/fusionforge.....	9
5	Conclusion.....	10

	Titre du document : Étude d'architecture du plug-in s'appuyant sur Mylyn et sur le plug-in Eclipse pour Codendi
	Référence : Tâche 3 Livrable 1
	Version du 01/06/11

# 1 Introduction


Ce livrable étudie les choix de l'architecture pour concevoir un plug-in sur le poste de développement. Ce plug-in doit fournir aux développeurs une interface permettant d'accéder à la forge depuis le poste de travail.

Nous souhaitons également enrichir la forge d'un plug-in qui s'occupe du pilotage des tâches des développeurs.

Pour ce plug-in de pilotage, nous allons concevoir côté Eclipse les développements nécessaires pour sa mise en œuvre.

L'étude est basée sur :


- les travaux menés dans le cadre des tâches 1 (Convergence entre les différentes forges) et 2 (création d'un poste de développement de base qui comprend le plug-in Codendi)
- le retour de l'équipe Codendi de Xerox, ainsi que
- l'étude de l'existant et notamment les travaux menés dans le projet Mylyn largement adoptée parmi les développeurs et qui propose un plug-in orienté – tâches

	Titre du document : Étude d'architecture du plug-in s'appuyant sur Mylyn et sur le plug-in Eclipse pour Codendi
	Référence : Tâche 3 Livrable 1
	Version du 01/06/11

## 2 Méthodologie

Nos travaux concernent trois parties :

1. La première partie concerne le plug-in Eclipse
2. La deuxième partie concerne le plug-in Forge
3. La troisième partie concerne la communication entre le plug-in Eclipse et le plug-in Forge.

	Titre du document : Étude d'architecture du plug-in s'appuyant sur Mylyn et sur le plug-in Eclipse pour Codendi
	Référence : Tâche 3 Livrable 1
	Version du 01/06/11

### 3 Architecture du Plug-in

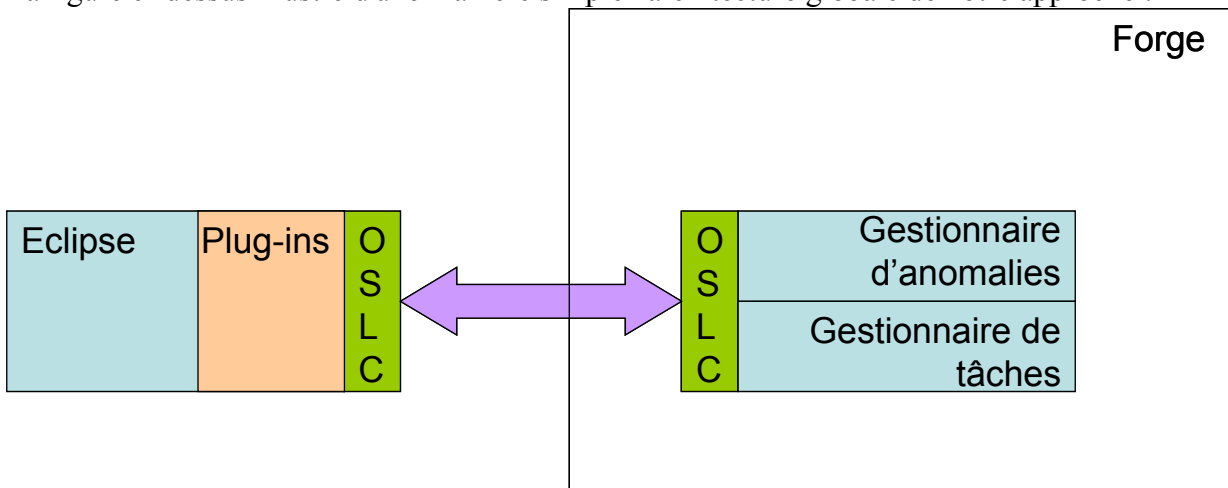
Nous présentons dans cette partie l'architecture du plug-in que nous souhaitons mettre en œuvre. Rappelons que pour nos expérimentations, nous allons utiliser Eclipse comme framework de développement et Novaforge comme forge logicielle.

Les technologies utilisées dans Novaforge sont le java et le PHP, en effet les développements propres à Novaforge sont en java et certains outils tel su Mantis sont basées sur le langage PHP. La technologie utilisée dans Eclipse est le langage Java.

Notre objectif est de proposer une solution générique et indépendante des spécificités de Novaforge et d'Eclipse. Afin que ce travail soit facilement adaptable à une autre forge.

Nous avons donc décidé d'utiliser le standard OSLC pour faire communiquer la forge et le poste de travail.

La figure ci-dessus illustre d'une manière simple l'architecture globale de notre approche :



Sur le poste de développement où se trouve Eclipse, nous allons concevoir des connecteurs OSLC pour le plug-in Mylyn d'Eclipse qui permet de gérer les fonctionnalités que nous souhaitons mettre à disposition de l'utilisateur depuis son Eclipse et qui sont accessibles depuis la forge. Nous allons également devoir implémenter des interfaces OSLC avec les outils de la forge qui gèrent ses artefacts sur la forge.

Entre la forge et le poste de développement, il y a une couche de communication. Cette couche de communication doit être abstraite dans le sens où le protocole de communication à mettre en place doit être indépendant et découplé aussi bien d'Eclipse que de la forge. C'est pourquoi, nous avons décidé d'implémenter le standard OSLC, de cette manière tous les outils exposant leurs services avec OSLC pourront utiliser le poste de travail.

	Titre du document : Étude d'architecture du plug-in s'appuyant sur Mylyn et sur le plug-in Eclipse pour Codendi
	Référence : Tâche 3 Livrable 1
	Version du 01/06/11

## 4 Choix d'architecture

### Plug-in Eclipse

Les travaux que nous menons dans ce WP au niveau du poste de développement sont proches des fonctionnalités proposées par Mylyn.

Rappelons que Mylyn est un plug-in Eclipse qui propose une perspective développeur orientée vue tâches. Il dialogue avec des trackers existants comme Mantis et autres pour permettre aux développeurs d'y accéder depuis leurs postes de développements. Mantis rencontre un grand succès auprès des développeurs et répond à leurs besoins. De plus, c'est le gestionnaire d'anomalies de NovaForge.

Au lieu de proposer une alternative à ce plug-in, nous avons choisi d'étendre les fonctionnalités de Mylyn. Nos travaux concernent principalement la notion de tâche de Mylyn et la réalisation d'un connecteur implémentant le standard OSLC-CM.

Nous nous basons donc sur l'architecture de Mylyn et les fonctionnalités offertes par ce plug-in pour la mise en œuvre des travaux de ce WP.

Rappelons que le plug-in Eclipse permet d'avoir accès à des fonctionnalités offertes par la forge depuis le poste de développeur ce qui a l'avantage de permettre la personnalisation du contenu à afficher suivant le profil de l'utilisateur mais aussi le contexte sur lequel il travaille. Ces fonctionnalités sont également accessibles par la forge depuis une page web.

### Plug-in forge


Nos travaux dans ce WP vont introduire la notion de tâche avec nos extensions pour Mylyn dans la forge. Ainsi, les gestionnaires de tâches et d'anomalies seront toujours synchronisés permettant ainsi à des profils non techniques comme les Chef de Projets ou les Directeurs de Projets qui n'utilisent pas Eclipse d'accéder directement à l'information.

A noter que les données pour la gestion de pilotage des tâches seront centralisées du côté de la forge et le plug-in Eclipse va envoyer des requêtes de mise à jour et de synchronisation au plug-in forge qui va s'occuper de la persistance des données.

### Communication eclipse/fusionforge

Nous avons vu que la couche de communication entre les deux plug-ins correspond à un modèle Client/Serveur et qu'il y a plusieurs possibilités pour la mise en œuvre de ce modèle comme les webservices avec plusieurs styles architecturaux comme XML/RPC, SOAP, REST, etc.

Nous avons étudié dans la tâche 1 de ce WP : « la convergence des différents travaux sur le poste de

	Titre du document : Étude d'architecture du plug-in s'appuyant sur Mylyn et sur le plug-in Eclipse pour Codendi
	Référence : Tâche 3 Livrable 1
	Version du 01/06/11

développeur ». Cela nous a permis d'avoir un retour intéressant sur le plug-in Eclipse de Codendi par Xerox.

Le plug-in Eclipse de Codendi étudié reprend les pages existantes et accessibles depuis la forge et les reproduit dans le poste de travail.

Le développeur a ainsi la possibilité d'accéder aux mêmes informations par deux canaux différents : la forge et le poste de développement. La communication entre le plug-in et la forge Codendi passe par des servicesweb.

Pour mesurer l'utilité de ce plug-in, nous avons mesuré le nombre d'accès à ces fonctionnalités ainsi que le canal d'accès utilisé par les développeurs.

Le recueil de ces informations a montré que les développeurs préfèrent passer par le canal web sur la forge pour accéder à ces fonctionnalités et n'utilisent pas le plug-in qui se trouve sur le poste de développement.

Cette préférence constatée est due principalement à la lourdeur de la communication mise en place entre Eclipse et la forge : problème de temps de réponse et de performance des servicesweb.


Ceci montre que l'idée de permettre aux utilisateurs d'accéder à des informations de pilotage depuis leur poste est intéressante mais sa mise en œuvre doit être performante pour qu'elle soit adoptée par l'utilisateur.

Ce retour a écarté le choix de passer par des webservices pour assurer la communication entre la forge et le poste du développement.

Étant donné que notre plug-in Eclipse se base sur Mylyn, nous avons étudié le choix de Mylyn d'implanter et de mettre en œuvre la spécification OSLC-CM pour communiquer avec les trackers.

Nous avons trouvé qu'il serait plus judicieux de s'aligner sur le choix de Mylyn parce que cela nous permet de se focaliser sur le contenu fonctionnel de plug-in de pilotage des tâches des développeurs et nous avons estimé que l'équipe de Mylyn a déjà une longue expérience et du fait a réalisé plus d'expérimentations sur les techniques de communication avec les trackers.

Le retour de l'équipe de Codendi/Xerox sur l'utilisation de leur plug-in a consolidé cette idée.

	Titre du document : Étude d'architecture du plug-in s'appuyant sur Mylyn et sur le plug-in Eclipse pour Codendi
	Référence : Tâche 3 Livrable 1
	Version du 01/06/11

## 5 Conclusion

Nous avons étudié dans ce document les choix à prendre pour la réalisation d'un plug-in Eclipse qui permet aux développeurs d'accéder à des fonctionnalités existantes sur la forge depuis le poste de développement ainsi que la réalisation d'un plug-in Forge pour la gestion de pilotage des tâches d'un développeur.

Notre travail concernait trois parties :

1. le plug-in sur le poste de travail,
2. le plug-in de la forge,
3. la communication entre le poste de travail et la forge.

Nos choix étaient principalement des choix stratégiques.

Nous avons donc décidé d'adopter le choix de Mylyn et pour mettre en œuvre la communication entre la forge et le poste de travail en implantant la spécification OSLC-CM côté Mylyn et côté Forge.