FFIA- PROPOSITION EN VUE DU RECRUTEMENT D'UN INGENIEUR ASSOCIE
Date : Février 2006

Auteur de la proposition : **Didier Parigot**

Projet ou service d'accueil : **Projet SmartTools, INRIA Sophia Antipolis**

Titre de la proposition : **Intégration de SmartTools dans Eclipse**

Résumé des objectifs :

Objectif de la proposition :

Fournir, à une large communauté de développeurs de plugins Eclipse, nos outils (également sous forme de plugins) pour automatiser le plus possible leur processus de développement. En nous appuyant sur la dynamique de cette plate-forme, permettre une diffusion plus large de notre approche de fabrique logicielle, par une instanciation sur le modèle d'environnement de programmation proposé par Eclipse.

Contexte de la proposition :

Avec l'essor grandissant de la société de l'information, il s'avère nécessaire de repenser fondamentalement le processus de développement logiciel pour une production rapide et fiable de familles d'applications (en particulier les applications ubiquitaires). Depuis quelques années, des nouveaux concepts comme la programmation par composants ou la notion d'architecture dirigée par les services (SOA), la programmation par séparation des préoccupations, la programmation dirigée par des modèles (MDA) ont été proposées pour répondre à ces nouveaux défis. Cela s'est accompagné de l'émergence d'une multitude de technologies logicielles, proposées et soutenues par des consortiums internationaux de standardisation (W3C, OMG...) ou des fondations/alliances (Apache, Eclipse, OSGA ...) ou des grands groupes industriels (MicroSoft, IBM, SUN, BEA...). Ces différents concepts peuvent être unifiés et regroupés dans une notion nouvelle de **fabrique logicielle** que nous avons concrétisé par **notre logiciel, SmartTools**. Ce concept de fabrique logicielle est défendu depuis 2004 par les plus grands groupes informatiques du domaine, comme étant l'un des axes importants pour la recherche et le développement pour le génie logiciel. On peut citer en particulier Microsoft avec leur nouvelle version de leur environnement de programmation avec les notions de DSLs[1] (« Domain-Specific Language »), IBM avec l'environnement de programmation Eclipse[2] et ses évolutions vers la notion de « Rich Client Plat-Form » (RCP) et enfin SUN avec l'environnement NetBeans.

Motivation pour une diffusion à travers des plugins Eclipse :

Même si nous avons validé notre approche grâce à notre propre utilisation, une diffusion dans sa forme actuelle (le logiciel SmartTools dans son intégralité) impose aux utilisateurs un saut qualificatif beaucoup trop important. Les industriels que nous avons pu contacter sont convaincus de la pertinence de notre démarche, mais attendent une mise en œuvre plus concrète et plus rapidement accessible.

Valorisation de la diffusion de notre logiciel :

L'instanciation sur le modèle Eclipse permet de proposer nos outils sur un processus de développement déjà largement diffusé et accepté. Les utilisateurs pourront plus facilement comprendre les avantages de ce processus de développement et l'instancier (totalement ou partiellement) sur leur développement sans avoir le besoin de changer radicalement leur méthode de travail.

Communauté d'utilisateurs visée :

La communauté des utilisateurs est essentiellement les développeurs de plugins Eclipse qui ne cesse d'augmenter. Il suffit de regarder sur le site de la fondation Eclipse pour voir le nombre de plugins proposés. Mais on vise dans un deuxième temps les développeurs d'applications du type RCP (assemblage de plusieurs plugins) qui pourront également profiter des points forts de notre approche. Pour se convaincre de la taille conséquente de cette communauté (pas de chiffre officiel sur le site

[1] msdn.microsoft.com/vstudio/teamsystem/workshop/sf/
[2] www.eclipse.org

d'Eclipse), il suffit de parcourir les divers sites Web : liste des plugins, liste des projets Eclipse, le programme de la conférence EclipseCon'06,…

Degré de fiabilité de la proposition :
Suite à notre examen précis des concepts et fonctionnalités d'Eclipse, une étude de faisabilité, dans le cadre d'un projet d'ingénieur, a déjà été réalisée (terminée fin février 2006). Les premiers résultats (réalisation sous la forme d'une maquette) sont très prometteurs et montrent bien la complémentarité des deux approches, Eclipse et SmartTools. Puis surtout cette étude montre bien que cette intégration ne demande en aucun cas une réécriture du code source de nos composants.

Avantage de notre approche par rapport à la concurrence :
L'avantage de notre approche est de proposer un vrai processus de développement par fabrique logicielle. En effet, d'après notre première étude de faisabilité, le processus de développement dans Eclipse reste avant tous un processus de programmation par objets (notamment pour le développement des plugins). Pour l'instant, les développements sous Eclipse sont extrêmement dépendants des diverses interfaces (API). Après une étude précise des différents projets (voir le site des « projet Eclipse ») et des différents plugins, nous pensons qu'il n'y a pas actuellement de solution vraiment concurrente à notre proposition (en particulier, les contributions actuelles ne proposent que des solutions partielles qui doivent impérativement être mises en commun manuellement).

Organisation et déroulement focalisés sur la promotion et la diffusion des résultats :
Pour mieux maîtriser le déroulement du travail, nous pensons procéder en quatre phases progressives.
Une première phase se focalisera sur l'instanciation de notre démarche dans le cadre d'une utilisation classique (génération de plugins). On ne cherchera à intégrer que les fonctionnalités de notre logiciel (un sous-ensemble de nos DSLs) utiles pour la génération automatique de plugins. Une seconde phase testera, par une auto utilisation sur nos propres plugins, cette intégration (en d'autres termes nous utiliserons nos propres plugins pour s'auto générer) et devrait intervenir le plus rapidement possible. La troisième phase mettra en place les divers moyens de promotion de notre solution (documentation, tutoriaux, démonstrations, etc.). En effet, les moyens techniques de la plate-forme Eclipse, les divers événements/conférences soutenus par la fondation Eclipse et enfin la large communauté de développeurs, seront de notre point de vue, des atouts pour réussir cette phase de promotion. Nous souhaitons que cette phase soit au premier plan de l'activité de l'ingénieur tout au long de sa mission de développement. Enfin une dernière phase se focalisera sur une utilisation plus poussée de notre approche, dans le cadre de la notion de RCP (assemblage de plugins). Les points forts de notre approche permettent de faciliter ces assemblages de plugins dans le cadre particulier d'applications ubiquitaires (sur le web).

Résumé de l'activité de l'ingénieur :

Description des fonctionnalités du logiciel, SmartTools :

A l'heure actuelle, **notre fabrique logicielle est composée d'une dizaine de DSLs avec leurs outils associés.** Ceci représente environ 100 000 lignes de code java écrites pour un logiciel comportant finalement environ 1 000 000 de lignes de code après génération automatique de code (e.g., structures de données, parseurs, vues graphiques, conteneurs de composants, etc.). Cette génération de code se fait par les différents générateurs de composants de SmartTools lui-même.
Nos axes de recherches (les points fort de notre démarche de fabrique logicielle) s'articulent essentiellement autour des trois concepts suivants :

- un développement dirigé par la spécification de langages dédiés (une modélisation à l'aide de DSLs) ;
- une notion de séparation des préoccupations définies directement sur ces langages dédiés, associés au métier sous-jacent ;
- une architecture dirigée par les services basée sur des composants qui doivent pouvoir étendre leur interface (services) dynamiquement.

Notre logiciel s'appuie sur nos travaux de recherche. Il a la capacité de s'auto générer. En effet, le résultat immédiat de cette auto-utilisation est que **plus de 90% du code source des applications ou**

des outils que nous avons réalisés est produit automatiquement par la fabrique. Étant donné que notre développement s'appuie fortement sur les technologies XML et la programmation en Java, notre fabrique se caractérise par les points techniques suivants :

- l'utilisation de java nous permet de profiter des divers efforts de développement du monde du logiciel libre et permet une complémentarité et une intégration dans des environnements de développement soutenus par les grands de ce secteur, comme l'environnement Eclipse d'IBM (expérience en cours) ou NetBeans pour SUN ;
- l'utilisation des technologies XML facilite l'intégration ou la complémentarité avec les divers outils du WEB développés par divers consortiums du domaine ;
- notre architecture dirigée par les services, basée sur la notion de composants hautement adaptables, permet de concevoir des applications par assemblage dynamique et facilite l'intégration dans d'autres plates-formes ;
- nos techniques de composants de visualisation (affichage/édition) et de communication d'information entre nos composants (à base de transformation de donnée en format XML) permettent un déploiement aisé sur la toile (Internet).

Description précise du travail à accomplir :

L'activité de l'ingénieur commencera par une phase préliminaire de mise en route pour bien comprendre les objectifs de l'opération et elle s'appuiera sur l'examen de notre étude de faisabilité.
La première phase consistera en une immersion des fonctionnalités de base de SmartTools pour automatiser le plus possible le développement de plugins (typiquement, lors de cette phase, nous ne chercherons pas à instancier notre modèle de communication entre nos composants, ou encore notre modèle d'architecture dirigée par les services). Plus précisément, cette phase s'occupera de l'immersion dans Eclipse des seuls trois composants basiques suivants : absynt, cosynt et xprofile. L'intégration des fonctionnalités plus complexes s'effectuera dans un deuxième temps, après une phase de promotions de ces premiers résultats.
La deuxième phase de tests aura pour objectif de valider au plus tôt la première phase « immersion » par une auto utilisation de notre démarche sur ces trois plugins (i.e., absynt, cosynt et xprofile).
La troisième phase de diffusion et promotion permettra, à l'aide des moyens techniques offerts par la plate-forme Eclipse, de mettre en œuvre une diffusion active de notre approche de génération automatique de plugins.
Et enfin, une quatrième phase aura pour but d'immerger dans Eclipse nos techniques ou concepts pour la construction d'applications plus complexes (par assemblage dynamique de plugins). En effet, avec la notion de RCP d'Eclipse, il est possible de construire une application de A à Z qui n'a plus rien à voir avec l'application Eclipse proprement dite. Avec les plugins, on enrichit l'environnement Eclipse considéré comme l'application de base. Par contre avec la notion de RCP, on peut concevoir une application autonome à partir d'un assemblage particulier de plugins. Dans ce contexte, l'atout principal de notre solution logicielle SmartTools est de produire des plugins qui sont prévus pour être assemblés sur la toile (à travers le réseau), grâce en particulier aux moyens de communication entre composants et aux moyens d'extension de services et d'exportation des vues graphiques.

Cahier des charges organisé en 4 phases :

Phase 1 d'intégration des composants de base de SmartTools :

Cette phase consiste essentiellement à immerger les techniques, les codes générés et les concepts de SmartTools, dans le modèle Eclipse.

- Intégration de la base de SmartTools sous forme de plugins.
- Intégration de la génération de structures de données (composant absynt).
- Intégration de la génération de parseurs (composant cosynt).
- Gestion de la notion de partitionnement et appel incrémental des parseurs.
- Intégration de la génération des parties sémantiques (composant xprofile).
- Gestion de la communication entre la structure de donnée et l'environnement Eclipse.
- Intégration de la génération de vues graphiques (composant cosynt).

Phase 2 de tests :
Cette phase consistera à mettre en place des bases de tests pertinents, mais surtout, à prendre le temps de tester des cas d'utilisation réels (téléchargement des plugins, déroulement de schéma de génération, etc.).

- Mise en place d'une base de tests pour les parties purement environnement Eclipse.
- Test des chargements de plugins sur divers OS (linux et Windows).
- Test des phases de génération.
- Test des diverses interfaces et environnements.

Phase 3 de diffusion et promotion :
Cette phase s'appuiera sur les outils disponibles sous l'environnement Eclipse, les moyens de promotion mis en place par la Fondation Eclipse (conférences, news-groups, événements, concours, etc.) et surtout sur le dynamisme de cette communauté, pour mettre en place une diffusion et une promotion actives de nos résultats.

- Mise en forme des divers plugins qui composeront la distribution de SmartTools et leur mode de distribution. On ne distribuera SmartTools qu'à travers des plugins Eclipse.
- Mise en forme des environnements associés à nos composants, avec la documentation en ligne, menus conceptuels, messages d'erreurs explicites, etc.
- Mise en forme de plusieurs exemples de création de plugins avec notre approche, à l'aide des composants déjà en place.
- Mise en forme du site Web SmartTools sous le site GForge de l'INRIA, et mise en place de mail-list, bug-report etc.
- Contribution sur le site Web des plugins Eclipse : www.eclipe-plugins.info.
- Mise en forme de démonstrations exécutables et téléchargeables à travers l'environnement Eclipse.
- Mise en place de FAQ, de tutoriaux, etc.
- Prise en charge des bugs reports et des mailing-lists.
- Prise de contact avec des industriels et préparation des démonstrations associées.
- Préparation et Participation à des événements de promotion (par exemple EclipseCon ou ObjectWebCon).
- Montage de journées de formations (destinées principalement aux industriels) sur l'approche.
- Montage de proposition ou appel à des projets de développement (sous forme de projets RNTL, projets Eclipse, etc.).

Phase 4 d'approfondissement vers l'approche RCP :
L'objectif de cette phase est de fabriquer des assemblages de composants (plugins) plus complexes, en vue de faciliter la construction de familles d'applications ubiquitaires (sur le web).

- Permettre un assemblage de plugins plus dynamiques et une extension dynamique de leurs interfaces grâce à nos techniques de composants hautement adaptables.
- Permettre une exportation sur le réseau des composants (plugins) d'interface utilisateurs grâce à nos techniques de génération de vue graphiques.
- Permettre une extension dynamique du comportement des composants grâce à nos techniques de programmation par aspects.

Durée et échéancier temporel :
Les phases présentées plus haut ne sont pas strictement chronologiques et s'effectueront en partie parallèlement les unes aux autres. La phase 1 durera entre 4 à 5 mois et se terminera à t0 + 6 mois. La phase 2 de tests sera échelonnée tout au long de l'opération et prendra entre 2 et 3 mois. La phase 3 commencera à t0 + 3 mois et durera de 6 à 8 mois. La phase 4 commencera à t0 + 9 mois (après l'évaluation de la première année) et durera de 6 à 8 mois. Un planning plus détaillé est fourni en annexe.

Environnement :

Organisation du travail :

Vu la variété des concepts sous-jacents à notre approche et l'ampleur et le dynamisme de cette communauté Eclipse, il sera nécessaire d'établir un suivi régulier (hebdomadaire) de la progression du travail de l'ingénieur. Pour cela, la durée de chaque action ne devrait pas excéder plus de 2 à 3 semaines (hormis les tâches de fond). Le suivi par un ingénieur DREAM de Sophia (service de développement logiciel) est systématiquement mis en place pour les ingénieurs associés et les ingénieurs ODL, avec une fréquence hebdomadaire.

Méthodes et outils de développement utilisés :

Comme notre approche est basée fortement sur un développement en Java et une programmation par composants (plugins), l'environnement Eclipse (un IDE très complet pour Java) sera naturellement notre outil de développement**.** De plus, les outils de distribution de plugins, de gestion de configurations, de gestion de projet et de gestion de versions (CVS), présents dans l'environnement Eclipse et que nous utilisons déjà, nous offrent gratuitement un cadre idéal pour ce type de développement.

Etat du logiciel au démarrage de l'opération et son état espéré à la fin :

Notre outil a été testé sur un ensemble cohérent de situations (testant l'ensemble des fonctionnalités de base), mais il reste certainement encore des imperfections. Une distribution sous forme de composants exécutables (fichiers Jar) et une mise à disposition des sources sous forme d'une base CVS sont déjà disponibles sur le site de SmartTools (www-sop.inria.fr/smartool/distrib). Le logiciel a été déposé à l'APP. Une licence LGPL et une base de donnée des chargements (une dizaine par mois depuis deux ans) sont en place. Des efforts de promotion et de diffusion ont été entrepris mais les retombées restent faibles pour l'instant :

- un Café Logiciel à Sophia, en juillet 2004 ;
- un contrat de recherche (direct) avec EDF R&D ;
- démonstrations à la conférence OOPSLA, en octobre 2004 ;
- lauréat de la journée « Tremplin Recherche », organisée par le Sénat, le 21 février 2006 ;
- plusieurs prises de contact avec des industriels (FT, Thales, EDF, …).

A la fin de l'opération logicielle, SmartTools sera disponible et distribué sous forme de plugins d'Eclipse.

Les partenaires éventuels :

Les objectifs de cette mission pourront intéresser plusieurs projets du thème « systèmes communicants » et en particulier le consortium ObjectWeb qui est en relation avec plusieurs projets de la fondation Eclipse (WebTool, par exemple). De plus, nous avons plusieurs contacts avec des industriels qui sont intéressés par cette approche, mais attendent une certaine maturité de celle-ci pour se lancer dans l'aventure. L'intégration dans l'environnement Eclipse sera certainement un atout pour lever cette réticence.

Positionnement par rapport aux défis scientifiques de l'INRIA :

Cette mission participe aux trois défis scientifiques de l'INRIA suivants :

- Concevoir et maîtriser les futures infrastructures des réseaux et des services de communication ; en effet notre architecture dirigée par les services, jouera certainement un grand rôle dans la conception et la maîtrise des applications sur Internet.
- Développer le traitement des informations et données multimédia ; notre outil s'appuie fortement sur les technologies XML.
- Garantir la fiabilité et la sécurité des systèmes à logiciel prépondérant ; notre approche à base de DSLs nous semble le passage obligé pour garantir une fiabilité des systèmes à logiciel prépondérant.

Responsable de l'encadrement : Didier Parigot

Profil de poste : On s'oriente plus vers le choix d'un ingénieur associé.
Compétences recherchées : Des connaissances approfondies sur la programmation Java, les techniques de compilation et la programmation distribuée seront incontournables. Une aptitude forte pour l'abstraction sera demandée.
Expérience souhaites : Une connaissance de la plateforme Eclipse (dont le développement de plugins) serait un atout supplémentaire.

Expliquer en quoi cette expérience est formatrice :

Etant donnée les technologies utilisées (Java, XML, Composant, Eclipse etc.) et les concepts manipulés, cette expérience ne pourra qu'être formatrice pour l'ingénieur. Apres ces deux années, l'ingénieur aura acquis une réelle compétence dans ce nouveau domaine de conception d'application à travers l'élaboration d'une fabrique logicielle.

Proposer des dispositions de formation et d'accompagnement :

L'ingénieur sera certainement solliciter pour participer aux nombreux lieux d'échange (conférences, user group etc.) autour de la plateforme Eclipse et de ces domaines R&D extrêmement actifs.

Analyser les perspectives de débouchés professionnels :

Les débouchés professionnels d'une telle mission sont sans appels vu l'impact de la plate-forme Eclipse dans le monde industriel. En effet, il suffit de parcourir le site d'Eclipse pour se rendre compte de l'intérêt que porte le monde industriel pour une telle démarche (développement libre). Ils sont souvent les porteurs des divers projets (proposals).

Annexe

Une proposition d'un planning des tâches avec [durée, phase]:
Pour la première année :

1. Apprentissage de l'approche SmartTools; [1 mois, phase 0]
2. Intégration basique de nos composants pour rendre l'ensemble de nos services disponibles à travers l'environnement Eclipse. S'assurer que les services de génération de notre composant « absynt » s'intégrer bien à la plate-forme Eclipse avec les parties de traitements sémantiques « xprofile »; [1 mois, phase 1]
3. Approfondir l'utilisation des services de notre composant « cosynt » (génération de parseur/scanner) pour d'obtenir de vrai éditeur incrémentale « à la Eclipse »; [1 mois, phase 1]
4. test sur nos trois composants [15 jours, phase 2]
5. Intégration de notre démarche de construction automatique de plugins en utilisant le plus possible les divers services (éditions, message d'erreur etc..) de la plate-forme Eclipse; [1 mois, phase 3]
6. phase de distributions des trois plugins de base [15 jours, phase 2 et 3]
7. Intégration des services de notre composant « cosynt » en terme de construction automatique de vue graphique; [1 mois, phase 1]
8. Documentations et instrumentations de nos trois plugins et de notre démarche de fabrique logicielle; [3 mois, phase 2 et 3]
9. Effort de diffusion de notre approche à travers des présentations/démonstrations. [3 mois, phase 3]

Pour la deuxième année : en poursuivant les tâches 8 et 9

1. Approfondir nos plugins en terme d'extension de services pour permettre d'étendre nos composants graphiques; [1 mois, phase 4]
2. Utilisation de l'architecture par composants dirigée par les services pour une plus grande souplesse; [3 mois]
3. Utilisation de la génération automatique de vue graphique pour une exportation de composant (plugins) à travers le réseau; [2 mois]
4. Utilisation de notre approche de programmation par aspects pour une adaptation dynamique des composants. [2 mois]

Les tâches 8 et 9 et celles de la deuxième année devront être précisées et découpées en sous tâches.