Extraction d'histoire d'évolution de code source avec gestion des *branches* et *merges*

Stage Master Recherche 2011-2012

Informations générales

- *Encadrement*
  - Jean-Rémy Falleri, Maître de Conférences, ENSEIRB-Matmeca, falleri@labri.fr
  - Xavier Blanc, Professeur, Université Bordeaux 1, xblanc@labri.fr
- *Parcours conseillé : Génie Logiciel*
- *Financement : LaBRI*

Contexte

Le thème Sphere (équipe LSR) du LaBRI est spécialisé dans le domaine du génie logiciel. Il vise à développer des méthodes d'assistance à l'évolution d'applications internet. Pour cela, nous avons développé un modèle permettant de représenter l'évolution syntaxique du code source d'un logiciel, nommé VPraxis. Ce modèle est indépendant des différents langages de programmation. Il considère que tout code source peut être représenté sous forme de graphe. L'évolution est ensuite représentée comme la suite d'opérations d'éditions qu'il faut effectuer sur ce graphe pour modifier passer d'une version à une autre d'un code source donné. En outre des informations sur la contenu du code, VPraxis permet aussi de retenir les informations sur l'ordre des différentes actions d'éditions, ainsi que les auteurs les ayant effectuées. Ces informations sont extraites automatiquement à partir des système de gestion de version (VCS) du type Subversion.

Sujet du stage

A l'heure actuelle, la modélisation élaborée pour VPraxis ne permet de représenter de manière satisfaisante que l'évolution de code sources contenus sur des VCS centralisés, tels CVS ou Subversion. En effet notre modèle ne marche correctement qui si l'ordre entre les versions d'un code source est un ordre total. Malheureusement, la nouvelle génération de VCS distribués, comme Git ou Mercurial, ne respecte plus cette contrainte. En effet dans ces VCS il est possible qu'une version donnée soit issue de plusieurs versions différentes, ce que l'on appelle un *merge*. Notre modèle n'est donc pas adapté à la représentation de telles histoires.

Il sera donc demander au stagiaire de définir un nouveau modèle qui permette de pouvoir représenter ce type d'histoire. En outre il sera demandé au stagiaire de définir et d'implémenter un algorithme qui permette d'extraire de manière automatique cette histoire à partir d'un dépôt existant. Il sera enfin demandé au stagiaire de valider son approche (correction de l'histoire, temps d'exécution sur des logiciels volumineux) sur des dépôts logiciels existants.

Références

- X. Blanc, I. Mounier, A. Mougenot, T.Mens, Detecting model inconsistency through operation-based model construction. IEEE 30th International Conference on Software Engineering (ICSE), pp.511-520, Liepzig, Germany, April, 2008
- Mercurial Definitive Guide : http://hgbook.red-bean.com/