Rapport HADL
Université de Nantes
2010-2011

Rapport HADL

DEJEAN Charles, POTTIER Vincent

14 novembre 2010

Table des matières

1 Présentation succinte de l'outil HADL

1.1 Comment utiliser l'outil

Pour commencer, il faut récupérer l'archive du projet, et la dézipper. Le code source complet se trouve dans le directory 'src'. Il est possible d'importer ce projet en tant que projet Eclipse.

Une fois le projet ouvert dans un IDE, il suffit d'exécuter la classe *'DescriptionLangage'* située dans le package *'hadl.m0.descriptionLangage'*.

Une fois cela fait, un menu s'affiche dans la console, offrant des choix de gestion de configuration.

1.2 Architecture globale

1.2.1 Le modèle abstrait

Les classes du modèle abstrait sont dans les packages hadl.m2.*. Il y a le composant, la configuration et le connecteur.

1.2.2 L'implémentation concrète

L'implémentation concrète se trouve sous les packages hadl.m1.*. Elle contient le connecteur RPC, le composant Client, la configuration Serveur. Cette dernière comprend les connecteurs ClearenceRequest, SecurityQuery et SQLQuery, ainsi que les composants ConnectionManager, DataBase et SecurityDB.

1.2.3 Le langage de description

Il est contenu dans le package hadl.m0.descriptionLangage. Il contient le fichier .xml décrivant la configuration CS, le fichier .dtd qui permet de vérifier si le .xml de description est correct syntaxi-

quement correct.

1.3 Choix d'implémentation

1.3.1 Pattern Oberver

Nous avons utilisé le pattern oberver pour gérer l'envoie de message entre les différents éléments composant d'une configuration.

Un composant est Observable c'est-à-dire qu'il peut envoyer un message qui sera notifié à tous les objets Observateurs qui l'observent.

Une configuration, quant à elle, est à la fois Observable et Observateur.

Ainsi, quand un composant veut communiquer, il le fait savoir à la configuration parent qui prend alors en charge l'acheminement du message.

1.3.2 Pattern Composite

Nous utilisons un dérivé du pattern composite, au niveau du composant et de la configuration.

Une configuration et un composant simple dérivent de la classe composant, car ils ont les mêmes fonctionnalités basiques que le composant, c'est-à-dire d'avoir un nom, des contraintes et des propriétés.