

Scénarios de tests d'intégration/fonctionnels



Partenaire : Cellule OpenEmbeDD

Nom de l'outil : Démo basique OpenEmbeDD

Date test : 24/04/2009

Version de l'outil : OpenEmbeDD 1.0.0 Titan

Plateforme de test (OS + processeur) : Fedora Release 8 (noyau 2.6.25.14-69.fc8) Intel

Organisme testeur : OpenEmbeDD integration team

Nom du testeur : Christian Brunette

Récapitulatif des tests

Commentaires du testeur.....	1
Scénario « création » : ouverture du projet et vérification des ressources.....	2
Scénario « launch ATL » : transformation ATL du modèle UML en modèle relationnel.....	3
Scénario « launch Kermeta » : simulation du modèle relationnel avec Kermeta.....	4
Scénario « Help Inline » : vérification de l'aide en ligne OpenEmbeDD basic demo.....	5

Les scénarios de test de cette fiche suivent l'ordre chronologique de leur définition : c'est à dire ici « création », « launch ATL », « launch Kermeta », « Help Online »

Commentaires du testeur

Scénario « création » : ouverture du projet et vérification des ressources

Remarques : prérequis => l'exemple démo de OpenEmbeDD doit être installé

Action	Résultat attendu	OK	Commentaires
Dans le menu File → New → Example... sélectionner l'item « OpenEmbeDD basic demo » du répertoire OpenEmbeDD demo et puis cliquez sur « Finish »	- création du projet de la démo de base OpenEmbeDD dans le workspace courant - projet ouvert - pas de changement de perspective - nom du projet = org.openembedd.basic.uml2cwm.demo	Oui	
Déployer complètement le projet et vérifier la présence des ressources	présence des ressources spécifiées : - répertoire « atl » - fichier umlclass2cwmrelational.asm - fichier umlclass2cwmrelational.atl - répertoire « kermeta » - fichier cwm_simulator.kmt - fichier cwm.kmt - fichier rdb_utils.kmt - fichier simulator.kmt - répertoire « model » - fichier SalesRDB.relational - fichier SalesRDB.uml - fichier SalesRDB.umldi - fichier cwmSimulator.launch - fichier uml2cwm.launch - icône de l'éditeur pour chacun des fichiers : ATL, Browser, Kermeta, ecore, genmodel, uml, umldi, texte et xml	Oui	
Ouverture par défaut des éditeurs correspondant par double-clic ou par clic droit + « Open With »	- ouverture des éditeurs spécifiques à chaque ressource - sans erreur d'ouverture	Oui	

Scénario « launch ATL » : transformation ATL du modèle UML en modèle relationnel

Remarques : pré-requis => ATL engine

Action	Résultat attendu	OK	Commentaires
Sélectionner le fichier « SalesRDB.relational » du répertoire « model » situé dans le projet de la démo, puis supprimer le fichier	- présence du fichier - suppression du fichier	Oui	
Lancer le launch configurator de ATL en allant dans le menu Run → Open Run Dialog... et choisir la transformation ATL nommée « uml2cwm ».	- présence de la configuration « uml2cwm » - présence des paramètres de la configuration avec modèle d'entrée model/SalesRDB.uml et modèle de sortie model/SalesRDB.relational	Oui	
Lancer le launcher en cliquant sur « Run »	- création du fichier SalesRDB.relational dans le répertoire « model » du projet - édition du fichier possible avec l'éditeur réflexif EMF relational	Oui	
Vérifier le contenu du fichier SalesRDB.relational avec l'éditeur EMF	Arborescence : - catalog - schema SalesRDB - table customer - table Product - table Order - 3 types SQL - Integer - String - Boolean	Oui	
Editer le diagramme UML SalesRDB.uml et ajouter une Property en allant dans la palette à gauche de l'éditeur UML. Ajouter cette Property à la classe Customer puis cliquer droit sur cette Property et sélectionner Show Properties pour afficher la vue des propriétés. Dans cette vue et pour la propriété nouvellement ajoutée, saisir le nom « Age » pour le champ Name et sélectionner le Data Type « String » pour le champ Type. Sauver le modèle.	- nouveaux attributs pour la classe Customer : + Age : String - validation du modèle possible en cliquant sur le bouton « Validate the model » dans la toolbar au dessus du diagramme	Oui	
Supprimer à nouveau le fichier « SalesRDB.relational » du répertoire « model » et lancer le launcher de ATL en allant dans le menu Run → Open Run Dialog... puis en cliquant sur « Run »	création du fichier SalesRDB.relational dans le répertoire « model » du projet - édition du fichier possible avec l'éditeur réflexif EMF relational - présence de la nouvelle colonne « Age » dans la table customer du modèle relationnel	Oui	

Scénario « launch Kermeta » : simulation du modèle relationnel avec Kermeta

Remarques : pré-requis => Kermeta workbench

Action	Résultat attendu	OK	Commentaires
Lancer le launch configurator de Kermeta en allant dans le menu Run → Open Run Dialog... et choisir l'application Kermeta nommée « cwmSimulator ». Stopper l'exécution.	- présence de la configuration « cwmSimulator » avec le programme kermeta/cwm.kmt - la console Kermeta s'ouvre et lance la simulation	Oui	
Action similaire : cliquer droit sur le fichier Kermeta kermeta/cwm.kmt et choisir le menu Run As → Kermeta App. Stopper l'exécution.	- la console Kermeta s'ouvre et lance la simulation	Oui	
Relancer la simulation en cliquant sur « Run »	- activation de la console avec un menu général de gestion pour les tables SQL du modèle : 1 - manage schema: SalesRDB exit - Exit the simulation	Oui	
Dans la console : taper 1 + entrée	- affichage du menu des tables : 1 - manage table: Customer 2 - manage table: Product 3 - manage table: Order back - Return to the previous menu exit - Exit the simulation	Oui	
Dans la console : taper 1 + entrée	- affichage du menu d'une table : add - Add a new data row to the table back - Return to the previous menu del - Delete one of the existing rows exit - Exit the simulation print - Print content of table 'Customer' sel - select on a foreign key value	Oui	
Dans la console : add + entrée	- affiche : Enter value for column 'Name' -->	Oui	
Entrer une valeur texte « name1 » par exemple + entrée.	- affiche : Enter value for column 'Address' -->	Oui	
Entrer une valeur texte « adresse1 » par exemple + entrée.	- affiche : Enter value for column 'Age' -->	Oui	
Entrer une valeur texte « age1 » par exemple + entrée.	- affiche : Enter value for column 'PK_Customer_ID' -->	Oui	
Entrer une valeur texte « 1 » par exemple + entrée.	Retour sur le menu de la table Customer	Oui	
Dans la console : print + entrée	- affichage dans la console du résumé suivant: << Table 'Customer' >> ----- Name Address Age PK_Customer_ID ----- name1 adress1 age1 1 -----	Oui	
Dans la console : exit + entrée	- fin de l'application Kermeta	Oui	

Scénario « Help Inline » : vérification de l'aide en ligne OpenEmbeDD basic demo

Remarques :

Action	Résultat attendu	OK	Commentaires
Ouvrir l'aide en ligne de la basic demo : sélectionner le menu Help → Help Contents, choisir le lien « OpenEmbeDD basic demo » dans la colonne de gauche puis lire la doc	<p>Sommaire de l'aide en ligne attendue pour la basic demo:</p> <p>Table of Contents</p> <p>Introduction</p> <p>1. Preamble</p> <p>1. Installation of prerequisites and demo</p> <p>2. Prerequisite</p> <p>3. Goal of the demo</p> <p>3.1. Schema of the demo</p> <p>3.2. Specifications</p> <p>2. Run the Demo</p> <p>1. FIRST STEP : UML diagram (with TOPCASED)</p> <p>2. SECOND STEP : transformation from UML to CWM (with ATL)</p> <p>3. THIRD STEP : CWM simulator (with Kermeta)</p> <p>4. Beyond the present demo</p> <p>4.1. Play with the demo</p> <p>4.2. Some possible extensions of the chain</p> <p>A. Technical data</p> <p>1. ATL launch parameters</p>	Oui	La numérotation ne correspond pas. Mais les sections sont bien présentes.
Relire le document et vérifier la non-régression des actions décrites et des captures écrans.	- non régression	Oui	La documentation concernant l'installation de la démo n'est pas à jour. Elle correspond à une installation sur Eclipse 3.3. Certaines images sont de mauvaise qualité.