Université d'Avignon
Master ILSEN 2016

Modélisation du logiciel

TP ILSEN : Analyse et Design du Séquenceur

1. Objectif du TP

L'objectif de ce TP est de réaliser l'implémentation de la bibliothèque « sequencer » permettant de réaliser des simulations à chronologie cohérente. Cette bibliothèque vous a été présentée en cours, et nous vous avons fourni un texte « État du problème : Séquenceur » qui rappelle ces explications.

La phase de UseCase a été réalisée en cours et nous avons obtenu une spécification de cette bibliothèque (transparent 29) et un mode d'emploi de son utilisation (transparent 30).

Le TP va se dérouler en quatre phases :

1) **Apprentissage de la méthode** : Le but de cette phase est de comprendre la méthode et d'apprendre l'utilisation de l'outil Papyrus, en refaisant l'analyse et le design de la simulation des pendules présentés en cours avec l'outil Papyrus. (CL 3h, AL 3h)
2) **Utilisation de la méthode** : Le but de cette phase est d'utiliser la méthode pour réaliser de manière autonome l'analyse et le design du séquenceur. (CL 7,5h, AL 7,5h)
3) **Examen de TP** : Pour les étudiants classiques uniquement un examen de TP de 3h. Le but de cet examen est d'évaluer votre capacité à utiliser la méthode pour résoudre un problème nouveau. Il est important que vous arriviez à cet examen avec un modèle Papyrus complet et pertinent.
4) **Réalisation de l'implémentation** : L'implémentation du séquenceur et de la simulation de test des pendules seront réalisées sous la forme d'un devoir à effectuer en dehors des séances de TP. Vous générerez le code Java à partir du modèle Papyrus rendu à l'examen de TP, vous compléterez et modifierez ce code et vous fournirez les diagrammes de classes en utilisant ObjectAid. L'examen de TP ne portera pas sur l'implémentation et n'utilisera que Papyrus.

2. Apprentissage de la méthode

Cet apprentissage va se faire sur la simulation des pendules.

2.1. Installation de l'environnement Eclipse

Vous devez avoir installé l'environnement Eclipse pour le TP avant la première séance de TP, en utilisant le document UML-Eclipse-installation.pdf.

2.2. Analyse

Vous devez rentrer les diagrammes du cours ce qui vous permettra de vous familiariser avec Papyrus. Nous vous donnons ici la marche à suivre, mais vous pouvez aussi utiliser la documentation de Papyrus disponible depuis le Help d'Eclipse :

1) **Créer un projet papyrus** : (File > New > Papyrus Project) nommer ce projet TP-ML-Papyrus-nom1-nom2 où nom1 et nom2 sont les noms des étudiants du binôme, choisir l'emplacement, (Next) choisir le langage UML, (Next) ne pas sélectionner de type de diagramme (Finish). Si vous ne l'avez pas, ajoutez la perspective Papyrus. Vous travaillerez dans cette perspective tout au long du TP.
2) **Créer la vue « logical view »** : Dans Project Explorer, vous voyez votre projet papyrus, pour l'ouvrir il faut cliquer 2 fois sur le fichier model, pour le fermer il suffit de fermer la fenêtre dans l'éditeur. Pour commencer à travailler sur votre modèle, une fois qu'il est ouvert, il faut sélectionner le Model Explorer. Dans le Model Explorer à partir du menu menu contextuel, vous allez créer un nouveau modèle (New Child > Model). Dans l'éditeur de « Properties », vous renommerez ce modèle « Logical view ». C'est dans cette vue que nous ferons l'analyse et le design. À partir du menu contextuel de « Logical view » importez le registred package « JavaPrimitiveTypes ».
3) **Créer les paquetages** : fr.univavignon.m1informatique.ml.sequencer (pour le séquenceur) , fr.univavignon.m1informatique.ml.sequencer.tests (pour les tests du séquenceur), fr.univavignon.m1informatique.ml.clocksimulation (pour la simulation des pendules), fr.univavignon.m1informatique.ml.clocksimulation.tests (pour les tests de la simulation) . Pour cela vous utilisez le menu contextuel sur « Logical view » (New Child > Package). ATTENTION, il faut créer tous les niveaux de package : fr, univavignon, m1informatique, ml, sequencer. Cette convention de nommage permet d'éviter les conflits.
4) **Créer un diagramme de classes :** sélectionnez le paquetage clocksimulation et créer un diagramme de classes « clocksimaulation-CD » à partir du menu contextuel (New Diagram > Class Diagram) .
5) **Construire le diagramme de classes du cours** : depuis le diagramme de classes, à l'aide de la palette d'outils créez les classes et les liens entre classes pour créer le digramme de classes du cours d'analyse (transparent 15).
6) **Créer le diagramme de séquences du cours** : même méthode que pour le diagramme de classes.
7) **Construire le diagramme de séquences** : même méthode que pour le diagramme de classes (transparent 17). ASTUCE pour mettre un message synchrone entre 2 « lifelines », il faut d'abord créer des « action execution specification » sur chaque lifeline.
8) **Compléter le diagramme de classes** : (transparent 18).

2.3. Design

Dans la simulation des pendules, il y a peu de problèmes de design excepté le problème de l'implémentation de « display » de « Clock » qui doit écrire dans un fichier sans ralentir la simulation. Ce cas est intéressant, car c'est une bonne illustration à la fois du masquage de l'analyse par le design et de l'aller-retour entre les phases de conception.Voici la marche à suivre pour réaliser la phase de design :

1) **Analyser le problème** : dans la documentation Java, regardez comment réaliser des écritures bufferisées.
2) **Modifier l'analyse** : modifier l'analyse pour pouvoir implémenter votre solution.
3) **Concrétiser les concepts** : reprendre le diagramme de l'analyse pour concrétiser les concepts (cf. cours de Design).
4) **Génération de code** : générer le code Java à partir du modèle UML. Le document « Devoir-Sequenceur.pdf » vous donne la marche à suivre pour le faire. Une fois le code généré, n'essayez pas de le corriger, vous le ferez dans le cadre de votre devoir. Par contre, construisez à partir des classes générées les diagrammes de classes avec ObjectAid, pour cela vous trouverez une aide dans le document « Devoir-Sequenceur.pdf ».

2.4. Fin de l'apprentissage

Vous devez à la fin de ce TP maîtriser les outils afin de réaliser le TP et le devoir en étant autonome. Si à la fin du premier TP, vous n'avez pas fini cet apprentissage, vous devez le poursuivre en dehors du TP afin qu'à partir de la deuxième séance, vous soyez autonome.

3. Utilisation de la méthode

Maintenant vous allez appliquer la méthode pour réaliser le séquenceur en respectant les deux phases : analyse, design.

3.1. Analyse

Cette analyse va porter sur la modélisation de la notion chronologie cohérente et celle du temps de simulation, en utilisant la méthode du cours. Nous vous conseillons de commencer par le contexte de chronologie cohérente, en considérant le temps réel. Une fois que ce contexte aura été analysé, vous passerez au problème du temps de la simulation.

3.2. Design

Vous essaierez d'identifier des problèmes de design. S'il n'en existe pas ou si vous les avez tous résolus, vous passerez à la concrétisation des concepts.

4. Examen de TP

Lors de cette séance, on vous donne un énoncé que vous devez faire en binôme. À la fin de la séance, vous déposez votre projet papyrus sur e-UAPV pour que nous puissions le corriger.

5. Réalisation de l'implémentation

Cette réalisation se fera sous la forme d'un devoir, le document Devoir-Sequencer.pdf vous donne des informations pour réaliser ce devoir.

Bon Travail ...