

TP1 : Création d'un simulateur

Objectifs:	Créer un simulateur et sa signature, construire et compiler pour une utilisation
	avec le moteur
Pré-requis:	TP0

Note: L'environnement de développement préconisé est Eclipse, accompagné de l'extension CDT. L'ensemble est disponible sur http://www.eclipse.org/downloads/ . En complément, un plugin OpenFLUID pour Eclipse a été développé pour accompagner la création d'un simulateur et générer un squelette du code source. Les développements s'appuient également sur la suite de compilation/construction/test/packaging CMake. CMake est disponible sur http://www.cmake.org .

1 Créer un simulateur

Il existe 2 façons de créer le code source d'un simulateur :

- Ecrire le code source "à la main", à partir d'un code vierge : long, fastidieux, source d'erreurs, …
- utiliser le plug-in OpenFLUID pour Eclipse : facile, assisté, intégré à l'environnement de développement, ...

Nous allons utiliser cette 2ème possibilité pour cet exercice.

1.1 Lancement d'Eclipse

Pour lancer Eclipse, cliquez sur l'icône Eclipse présente sur le Bureau. Au lancement d'Eclipse, il peut vous être demandé de choisir le chemin du workspace que vous souhaitez utiliser. Il est conseillé d'utiliser celui proposé par défaut (sous Linux : /home/openfluid/Bureau /formation/workspace). Le workspace est le répertoire qui contiendra l'ensemble des projets de simulateurs développés dans cet exercice ainsi que les suivants.

1.2 Création du squelette du simulateur

Nous allons utiliser le plug-in OpenFLUID pour Eclipse afin de générer le code source "vide" d'un simulateur. Aller dans le menu d'Eclipse *File > New > Other*... Dans la fenêtre qui s'ouvre, choisir *OpenFLUID > Simulator project* et cliquer sur *Next*.

😣 🗊 New				
Select a wizard				
Wizards:				
type filter text				×
🕨 🗁 Git				
🕨 🗁 Java				
🕨 🗁 JavaScript				
🖻 🗁 Maven				
🔻 🗁 OpenFLUID				
Simulator proj	ect			
PHP				
Plug-in Developn	hent			
PyDev				
?	< Back	Next >	Cancel	Finish

La fenêtre suivante est la 1ère étape de la création du simulateur. Cette première étape consiste à définir les caractéristiques du code source du simulateur.

Cliquer sur *Create new project* et indiquer le nom training.signal.prod. Choisir SignalSim.cpp comme nom de fichier (*Source file name*), SignalSimulator comme nom de classe C++ qui contiendra le simulateur (*c++ class name*). Laisser coché *Create Cmake build system files* ainsi que *Run Cmake on project creation* Cliquer sur *Next* une fois cette étape complétée.

80
OpenFLUID simulator
Sources files and build system
<u>Create new project:</u> training.signal.prod
O Use existing container: Browse
Sources
Source file name (.cpp): SignalSim.cpp
⊆++ class name: SignalSimulator
🥳 Create CMake build system files Build system Build subdirectory: _build
Install directory: /home/fabrejc/.openfluid/simulators
Enable build of documentation using sim2doc
🕢 Run CMake on project creation
(?) < Back Next > Cancel Finish

La deuxième étape consiste à renseigner les informations sur le simulateur, et notamment son identifiant (*Simulator ID*) par lequel il va être reconnu. Taper training.signal.prod comme identifiant de simulateur. Les autres champs sont facultatifs, vous pouvez y renseigner un nom plus détaillé du simulateur, votre nom et email. Cliquer sur *Next* une fois cette étape complétée.

80	
OpenFLUID simul	ator
Simulator ID and m	eta-information
Simulator ID:	training.signal.prod
Simulator name:	
Simulator version:	13.11
Simulator Domain:	
Description:	
Author's name:	
Author's email(s):	
?	< Back Next > Cancel Finish

La troisième et dernière étape consiste à renseigner les informations sur les variables/paramètres/attributs/évènements qui sont produits/utilisés/mis à jour par le simulateur.

Accribuces	variables	Events	Extra files	Scheduling	Spatial dynamic
init Descript	ion				
dd		Modify	y		Remove
	dd	nit Description	dd Modif	dd Modify	nit Description dd Modify

Aller dans l'onglet Scheduling et cocher l'option Scheduling uses the default DeltaT value.

80
OpenFLUID simulator
Data and dynamics
Parameters Attributes Variables Events Extra files Scheduling Ondefrned scheduling Scheduling uses the default DeltaT value Scheduling uses a fixed DeltaT value Scheduling uses a fixed DeltaT value Scheduling uses a DeltaT value varying from to
? < Back Next > Cancel Finish

Enfin cliquer sur *Finish*.

Si tout s'est bien passé, vous devriez obtenir un dossier training.signal.prod contenant 3 fichiers dans le *Project Explorer* d'Eclipse :

- SignalSim.cpp : fichier source du simulateur,
- CMakeLists.txt : fichier du système de construction du simulateur (à priori, à ne pas modifier) ,
- CMake.in.config : fichier de configuration de la construction du simulateur.

ainsi qu'un sous-répertoire _build qui contiendra les résultats de compilation/construction du simulateur.



2 Construire et installer le simulateur

La construction/installation du simulateur peut se faire de deux manières :

- Soit en utilisant la ligne de commande en exécutant la commande make install depuis le sous-répertoire _build
- Soit depuis Eclipse, dans le menu Project > Build Project ou cliquer sur l'icône Marteau de la barre de menu Outils. La méthode depuis Eclipse est à privilégier.



Note: Cette construction/installation est à exécuter à chaque fois que le code source du simulateur est modifié.

Pour vérifier que le simulateur a été correctement construit et installé, exécuter la commande suivante dans un terminal : openfluid -f

Le simulateur devrait alors apparaître dans la liste.

🙉 🗐 🛛 rabotin@lisah-ddaylewis: ~/000Lisah/1_projets/5_openfluid/9999_workspace/training.signal.prod/_build
Fichier Édition Affichage Rechercher Terminal Aide
Generating done Build files have been written to: /home/rabotin/000Lisah/1_projets/5_openfluid/9999_workspace/training.signal.prod/ build
rabotin@lisah-ddaylewis:~/000Lisah/1_projets/5_openfluid/9999_workspace/training.signal.prod/_build\$ openfluid -f
OpenFLUID v2.0.0~alphaS
software environment for Modelling Fluxes in Landscapes
LISAH, Montpellier, France
Available simulators: - training.signal.prod - examples.primitives.unitsA.up - examples.primitives.unitsA.prod
- examples.road.traffic - examples.trafficlight.state - examples.printives.unitsB.prod
rabotin@lisah-ddaylewis:~/000lisah/1_projets/5_openfluid/9999 workspace/training signal prod/_build\$ []