AI Level of Detail for Really Large Worlds
(the readme file concerning the CD example)

Cyril Brom, Tomáš Poch, Ondřej Šerý

Content

The CD contains:
- the IVE ("Intelligent Virtual Environment") simulator, a generic simulator of 2D grid-based worlds with LOD AI
- editor for authoring virtual worlds for IVE, including AI of virtual agents
- 2 example worlds (a pub world and a school world)
- documentations

Directory structure:
At the top level, there are separate directories for IVE Simulator, IVE Editor and example world DemoWorld. These directories follow the same structure. There is the `src` directory, containing java and xml sources, `resources` directory containing images and finally the `dist` directory containing the executable jar file. Besides, there is documentation and ant script which builds the dist directory from sources.

Pub Example

The pub example from the IVE simulator differs from the chapter's example in several aspects. Most importantly:

- There are 5 LODs, but they are organized a bit differently than in the chapter; for instance, there are no tables in the bar room at LOD 4 and there is no "beer barrel" there.
- The version on the CD does not work with expirations and information level (though these mechanisms have been implemented elsewhere).
- The general placing mechanism that groups objects based on their typical places of occurrence into floor-objects, table-objects etc., and that generates these objects randomly within constraints of the objects' categories has not been implemented.
- We have a waiter instead of a barman. The waiter is actually the only "story important" person meaning his view level (5) is larger than his existence level (3).
- Behavior trees of the NPCs feature two entities: processes and goals; these alternate in the hierarchy. Processes represent possibilities how to accomplish a goal; to accomplish a goal, one needs to execute successfully at least one process. If this process fails, one has to choose another process. Processes can be decomposed into subgoals. All of these subgoals (or most of them) must be executed to finish the process; some sobgoals may need to be executed repeatedly. Note that a process has been called a "task" in the chapter. In academic literature, the term "process" (or "plan") is often used when speaking about a particular implementation of an action selection mechanism while the term "task" when speaking more theoretically; however, the terminology is not settled. Note that the term "plan" is often used in the context of so-called BDI-based languages (BDI = Belief Desire Intention) and it is used there in a different manner than in the literature on classical planning (e.g. STRIPS).
- One level of detail is assigned to more levels of the behavioral hierarchy.

Installation & Quick Start

To run the IVE software, please make sure, that Java SDK is installed on your system and that its bin directory is in your path (you should be able to run java, javac and jar just by typing them in command line).

Running IVE Simulator

Go to the ive_core/dist directory and run ive_core.jar file.

```
cd ive_core/dist
java -Xmx512M -jar ive_core.jar
```

Main window of the simulator appears. There is not much to see until a world is loaded.

Loading DemoWorld

Run the IVE Simulator. In the File menu, select the Load World item. In the Open dialog, select the main xml file containing the world description. In the case of the DemoWorld example, select the demoworld/dist/DemoWorld.xml . Other files in the demoworld/dist directory describe parts of the world and are referenced by the main file. Do not open them individually.
When the world is loaded, run the simulation by pressing the play button in the toolbar. There are, however, no expanded locations at the beginning. At the right part of the screen, there is a tree of locations. At the beginning there is just the World location. To show the content of the location in the main part of the window, right-click on the location and select Open "Area A in Tab" option. To expand the location, select "Add Holdback" option which opens the dialog box. If default values in the dialog box are used, the location is expanded by one level, so that all sublocations remain atomic. In the DemoWorld, when one of cities is expanded, the waiter – story important person appears and automatically expands the locations in his neighbourhood to the level 5. You may find interesting to follow him. To speed up the simulation, use the Simulation/Real time ratio slider or the FastForward button. Further details are described in the user documentation - ive_core/UserDoc.pdf.

Running IVE Editor

Go to the editor/dist/lib directory and run the ive_editor.jar file. Since the editor uses relative paths, the actual directory must be the editor/dist/lib directory. Running the jar file by double-clicking on the icon should work well.

```
cd ive_editor/dist/lib
java -Xmx512M -jar ive_editor.jar
```

Main window of the simulator appears. To load the world, select the "Open Project" item from the File menu. In the Open dialog, select the directory containing the world description. To open the DemoWorld, select the editor/dist/demo/OriginalDemoWorld directory. The world description used by the editor is different to the description used by the simulator.
When the world is loaded into the editor, you may edit its content as described in editor/manual.pdf. When you are done, select the "Build World" item from the File menu, which builds the world description used by the simulator. Then, you may run the simulator directly from the editor by selecting "Run in Ive" item from the File menu.

Overview of source files and classes

Processes are described on two levels in the IVE simulator. First, the declarative part, describing preconditions, expansion to sub-goals, and participating objects and characters, is stored in an XML description. For an example of such description, see the directory "demoworld/xml". The scheme is described in the user documentation "ive_core/UserDoc.pdf", which also contains a simple step-by-step tutorial.
Second part of the process description is imperative and is written directly in Java. Technically, to provide such an imperative description, a programmer has to implement an interface `cz.ive.process.ProcessTemplate`. Since this interface is quite large, an abstract class `cz.ive.process.CommonProcessTemplate` is provided for convenience. Therefore, in majority of cases, it suffices to extend this method and implement three methods: `atomicLength`, `atomicCommit` and `atomicStop`. These three methods closely follow the ideas presented in the chapter. The `atomicLength` method has to compute expected duration of a task when executed as atomic. The `atomicCommit` method is executed at the end-event of the task to instantaneously compute its atomic result. Last, the responsibility of `atomicStop` is to compute the partial result of the task stub. Each method considers actual states of involved objects (e.g. it takes it less time to drink a half empty glass of beer than to drink a full glass). Examples of the imperative description can be found in the directory "demoworld/src/cz/ive/process/".

License

The IVE simulator is distributed under BSD license. The authors did their best to acknowledge every third-party work employed in IVE appropriately. If you find any incorrectness or omission, please let us know. We will remedy the problem as soon as possible.