Guidance on Building and Running Multi-agent Systems in the 3APL Interpreter

Student ID: 16522115

May 4, 2020

To start with, this interpreter is compiled and packaged into a single *.jar* file. The classes being packaged includes all classes that has been used and defined throughout the development of this interpreter including *tuProlog* classes. To run the interpreter, command "java -jar 3APL.jar" should be executed in a properly setup environment which provide JDK 11 and javaFx module. If the path of javaFx module is not included in the environment variables, it should be set using "–module-path" argument or by other methods.

To build a multi-agent system, servers and containers can be considered as general middleware that have already been pre-programmed and the only two components need to be programmed are agents and potentially an environment.

To program an agent, a plain text file should be created containing an agent program following the syntax described in the disseration. It is also notable that one file can contain one and only one agent program.

Environments should be programmed and compiled as Java classes extending the provided *Environment* abstract class. The details of that abstract class can be found in the dissertation. Normally, to program an environment, the programmer need to include the *.jar* file as a Java package in order to use the classes provided. Once the environment class is programmed, it need to be compiled into a *.class* file which will be used by the interpreter later. Note that as long as the *.class* file is in the same path as all other programmer defined classes that are used by the environment, it does not matter which path the *.class* file is in, the interpreter will allow users to choose the path for these files.

After all the necessary components are programmed, the interpreter will be used.

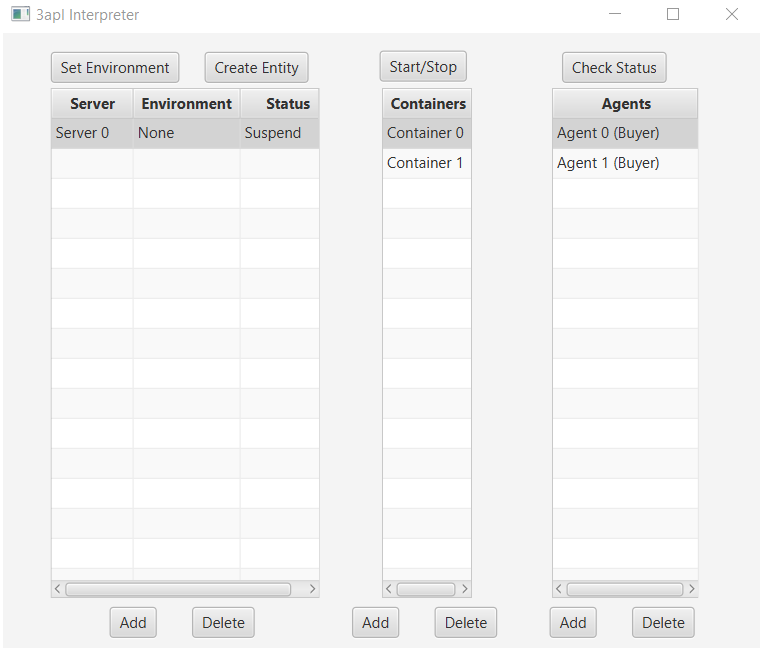

Figure 1: Main Page

As we have discussed in the dissertation, each server contains one and only one multi-agent system, and therefore, by adding (creating) a server in the interpreter, a multi-agent system is considered to be created. After a server is created and selected (single click on that row), containers can be created for it. To create an agent, a container should also be selected, the adding button will open a new file selector by which users should select the agent program file they want to compile. After the program file is selected, it will be compiled into an agent and its identifier and name will be displayed in the table.

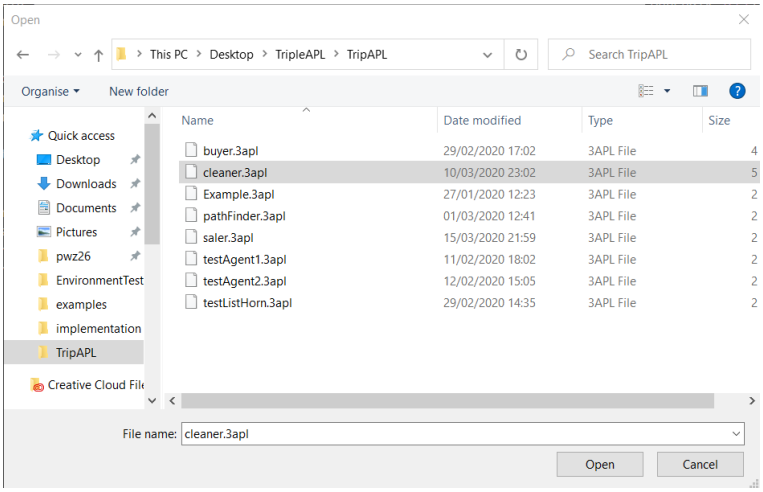

Figure 2: Agent Creation Page

To connect an environment to a multi-agent system, the "set environment" should be used to open a file selector using which an environment will be selected, loaded and connected to the previously selected server.

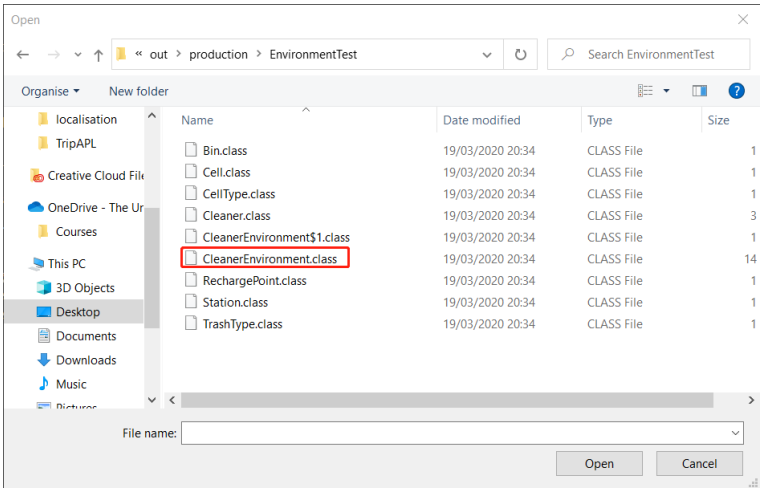

Figure 3: Environment Creation Page

Once an environment is connected, user and select an agent and use the "create entity" button to create a controllable entity and connect it to the agent. A new page will be prompted which lists all controllable entities provided by the environment and their arguments. By entering proper arguments and click the "create" button, an controllable entity will be created and connected to the agent. However, this function can also be used to make changes to the environment without creating any entities, more details can be found in the dissertation.

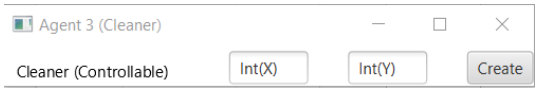

Figure 4: Entity Creation Page

Once the multi-agent system is completed, it can be run by the "start/stop" button. The runtime status of the selected server (multi-agent system) can be monitored by the status monitoring page. That page can be opened at any time by the "check status" button.

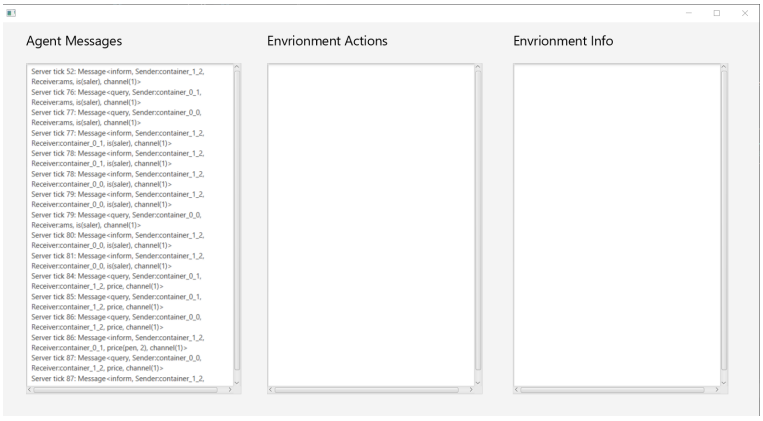

Figure 5: Status Monitor Page