Chapter 7

JSystem Agent

➢ **In this chapter…**

7.1 JSystem Agent

JSystem 5.0 represents the first installment in the conversion of the JSystem application Framework from a desktop application into a full client server application.

The JSystem JRunner has undergone an infrastructural alteration in order to enable the separating of the execution engine and JRunner UI.

7.2 Installing and Running the JSystem Agent

In order to install and activate the JRunner Agent, run the "**runAgent.bat**" batch file.

- **runAgent.bat (runAgent for Linux)** – activates the JRunner agent. The JRunner agent is a console application (no UI). The agent can be activated and controlled via the JRunner UI or by native java JConsole application.
- **runBase.bat (runBase for Linux)** – base script file used by the run and runAgent. (Runs in the background on the machine).
- **InstallJSystemAgent-NT.bat** – script installs the JRunner agent as a service on a windows machine.
  Issues to note when working with JSystem's JAgent:
  - In order to install the service on Windows Vista the script has to be activated by the system administrator.
  - In order for the service to work properly, the JAVA_HOME environment variable must be defined.
  - Activating the JRunner agent as service on Linux is presently not supported.

Note: When working with the JRunner agent, make sure that the installation path does not have spaces in it (for example, don't install JSystem on c:\Program Files\...).

7.2.1 Configuring the JSystem Agent

The JSystem agent supports all JSystem properties that are related to test executions for example, run.mode and test.vm.params. These properties can be altered by changing the "**jsystem.properties**" file.

Relevant agent parameters:

- **RMI port** – The (Remote Method Invocation) RMI is the main protocol used for connecting and activating the agent. The RMI server port can be configured in the runAgent script file or in the "**wrapper.conf**" file for the agent service. The default port is 8999.
- **Web port –** The JRunner agent has an embedded tomcat servlet that enables accessing HTML reports remotely. The HTTP port can be configured by setting the property "**agent.server.web.port**" in the "**jsystem**.**properties**" file. The default port is 8383.
- **FTP port** – The JRunner agent has an FTP server embedded in it that enables the transfer of automation projects from remote clients to the agent. The FTP port is configured by updating the property "**agent.server.ftp.port**" in the "**jsystem.properties**" file. The default port is 2121.

7.3 Working with the JSystem Agent

1. Run the JRunner agent on machine "**a**", either by activating the "**runAgent**" script or by registering the service and starting it.
2. Now, run the JRunner UI on another machine by activating the run script.

Note: The JRunner agent and the UI can be activated on the same machine, but they need different installations.

3. Open the JRunner and select a project then create and or edit a scenario, then run the scenario, making sure that it is stable.
4. Select the "**View→Toolbars→Agent**" Toolbar.

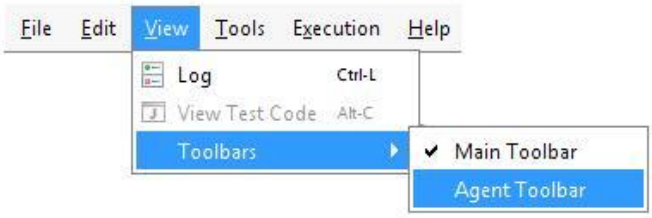

Figure 1: Setting the JSystem Agent

5. The Agent toolbar appears in the JSystem interface toolbar.

Figure 2: JSystem Agent Tools

7.3.1 JRunner Agent Toolbar Items

- **Agent List** – Lists recent agent connections between the JRunner GUI and the agents. The first agent in the list is the last agent that was connected to the JRunner. By default the JRunner starts with the "**local**" agent selected in the agents list. Local refers to the JRunner operates normally.
- **Connect -** Signals the GUI to connect to the agent currently selected in the agents list. The agent list combo box is editable, in order to connect to a new agent write "**agent_ip:rmi_port**" and press on the connect button.
- **Status -** shows the connection status:
  - **Green** - Agent correctly connected.
  - **Red** – There is a Problem with the agent connection.
  - **Grey** – No connection exists.

When the connection to agent is lost, the UI starts to monitor the connection. Once the agent can be reached the status button returns to the green icon.

- When the JRunner UI is connected to the agent the user can either continue to configure the scenario or press on play button.

By pressing the play button the JRunner performs two functions:

1. The UI signals the agent to switch to the automation project that is currently active in the client. The project is identified by the name of class folders parent.
2. The JRunner asks the user whether to synchronize the agent with the automation project, if "**yes**" is selected the automation project is copied from the client to the agent (using the FTP server on the agent).

Note: In order to initiate the first run press the "yes" button to synchronize.

In order to implement a change in the scenario, press the "**yes**" button to synchronize the JRunner.

Automation content that is transferred to the agent includes the following:

- Class's folder (includes compiled tests, scenarios and SUT files).
- "**lib**" folder (includes system objects jars).
- "**resources**" folder including additional resources that are needed by the automation project.
    - The UI signals the agent to switch to the currently active scenario and currently active SUT file
    - The JRunner takes relevant "**jsystem.properties**" on the client side and updates the agent with the properties values.
    - The agent signals the run scenario.
    - The execution cannot be stopped while the agent is synchronizing with the client. The reporter tab is updated with remote activation progress.
- Once the execution has started, stop, graceful stop, pause, repeat work as usual.
- By pressing the "**log**" button, the browser is opened with a URL that points to the "**index.html**" on the remote agent.
- The following functionalities do not work when working on remote agent:
    - The jar list shows a list on the client and not on the remote agent.
    - The JRunner publishes through the UI, and works only on local agent.
    - Database properties updated manually on agent.

7.4 Distributed Execution

The JSystem Automation Framework version 5.1 supports a distributed execution of a JSystem scenario. The components that comprise a distributed setup are one or more JRunner agents and a JSystem execution station.

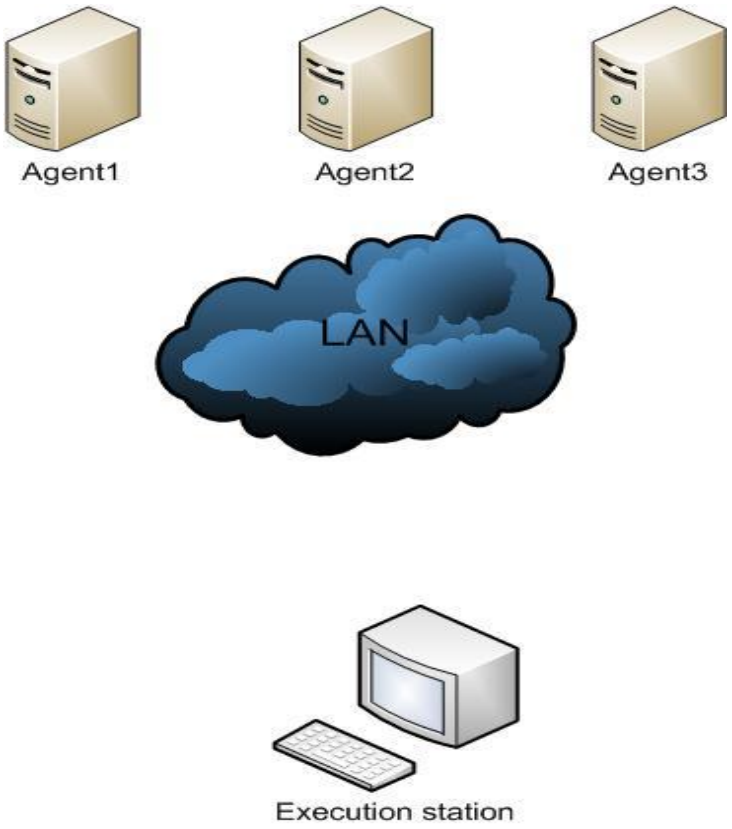

Figure 3: Distribution Execution Diagram

7.4.1 General Workflow of Distributed Environments

The following list details the basic steps required for working with this "**Agent Distribution**".

1. Installs agents.
2. Add agents to the JRunner in the execution station.
3. Design the scenario.
4. Assign agents to sub-scenarios/tests.
5. Run the scenario.
6. Analyze the results.

Note: In order to get and handle events that come from the executing agent correctly, the version of the remote agent should be identical to the version of the JRunner in the execution station, in addition, all agents must be synchronized with the execution station automation project

7.4.2 Working with a Distributed Setup

Once agents are installed in the setup, the JRunner executing station must be configured to "**recognize**" the agents; the user can then associate scenarios and or tests with an agent and then run the scenario.

7.4.2.1 Managing Agents

In order to manage agents in the JRunner, select the Tools pull down menu and select the "**Agents List**" menu item.

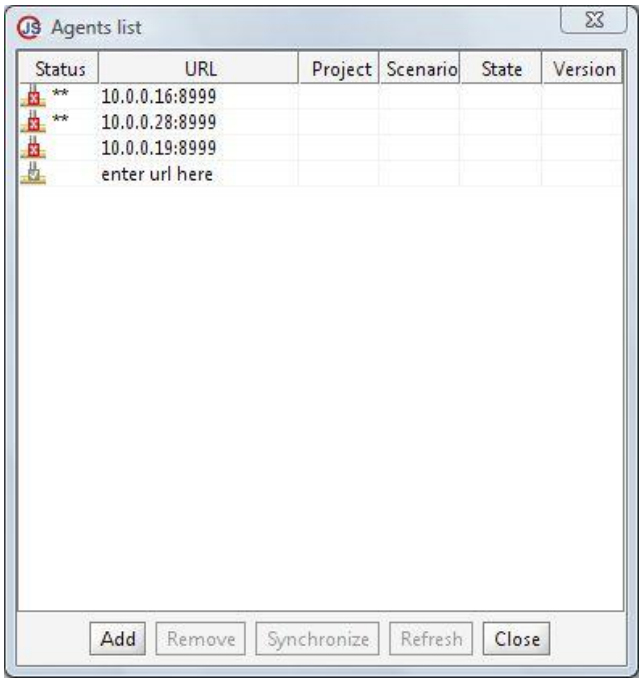

Figure 4: Agent List Window

7.4.2.2 Agents Table

The Agents list dialog is divided into the following columns.

Status- The connection status to the agent, when the icon is red it means that either the agent is down or there is a network problem. When an agent is marked with ** it means that one or more sub-scenarios and or tests in the currently selected root scenario are configured to run on the agent.

URL - The URL of the agent, the "**host:port**" the URL that is used by the system as a unique ID to identify the agent.

Project - The name of the automation project that is currently active in the agent.

Scenario - The name of the root scenario in the agent.

State - Execution state (running/idle).

Version - Agent version.

7.4.2.3 Operations

Add - By pressing the "**Add**" button a new line is added to the table.

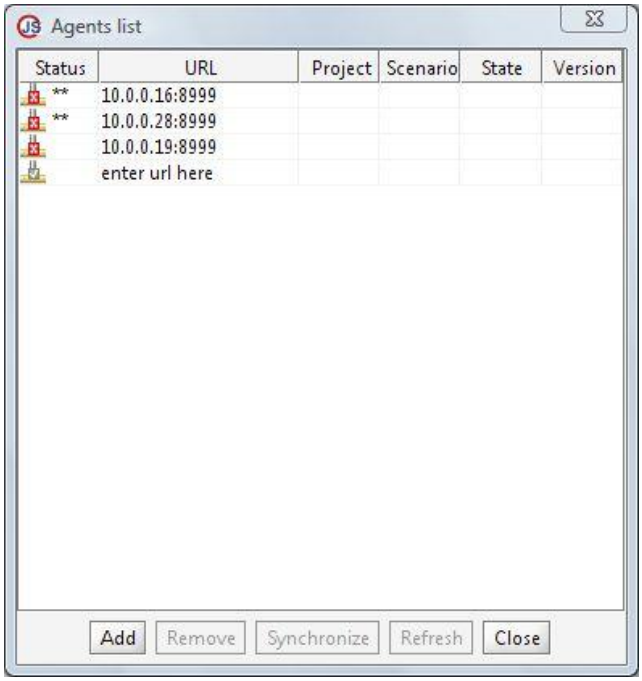

Figure 5: Adding an Operation to the Agent List

Select the "**Enter url**" input field and enter the agent's URL, then press the "**enter**" button. The manager will then add the agent and will attempt to make a connection.

- **Remove –** In order to remove an agent from the managed agents list select one or more agents and press on the "**Remove**" button. Once an agent is removed from the list, the user can not assign it to a test/scenario, and events dispatched from the agent will not be shown by the JRunner.
- **Synchronize** – In order to synchronize the agent with the local project, select the agents to be synchronized, and press on the "**Synchronize**" button.

By pressing the "**Synchronize**" button the following occurs.

1. The currently active automation project in the execution station is zipped.
2. The zip is then transferred, using FTP protocol to the selected agents.
3. Agents are then signaled to extract the project.
4. The currently active SUT file and scenarios are set in the agents.
5. During the process a progress panel is displayed to the user.
    - **Refresh** - When pressed, the JRunner tries to reconnect to selected agents and fetch the agent information.
    - **Close** -Closes the management dialog.

7.4.3 Assigning an Agent to a Scenario and or Test

In order to associate a test and or scenario with an agent perform the following steps.

1. Select the test/scenario in the scenario tree.
2. Go to the "**Test Info**" tab.
3. In the "**Test Info**" tab, select the "**JSystem-Hosts**" sub-tab.

Figure 6: Assigning an Agent to the Host

4. Select the "**Value**" input field and the "**file selection**" button will appear.

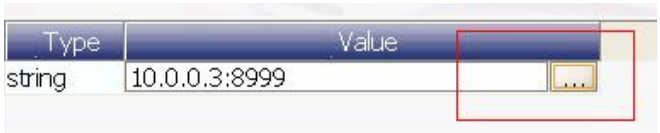

Figure 7: Setting the Value of the Host

5. After pressing on the "**Select File**" button, a selection dialog opens, now select the hosts that the scenario/test will run on and press the "**Save**" button.

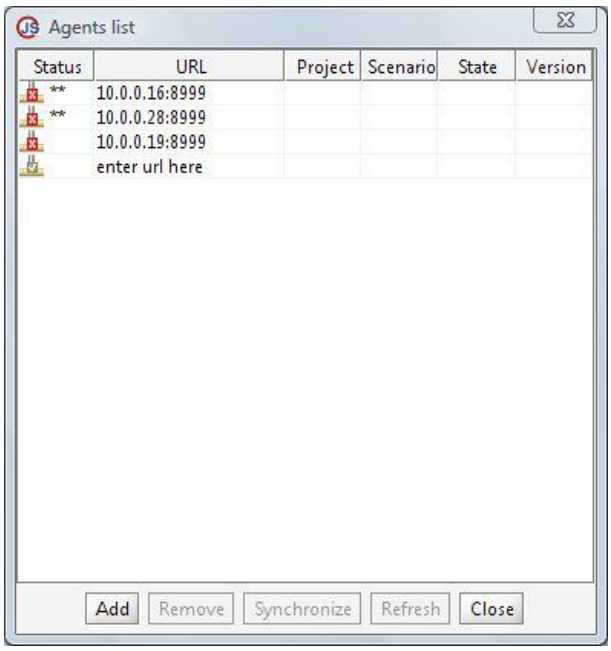

Figure 8: Select Agent Window

7.4.4 Assignation Notes

1. The top level assignments overrides assignments made to the offspring, as is illustrated in the following scenario.

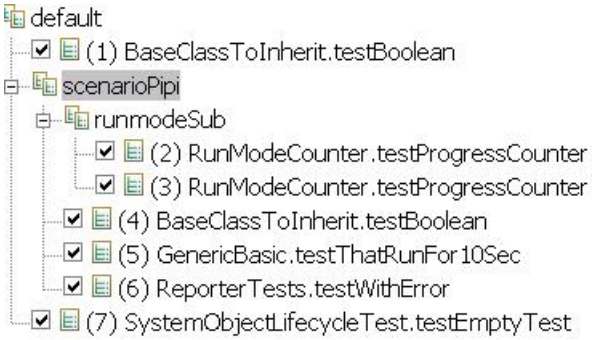

Figure 9: Assigning Notes to a Scenario

2. Assignments are made to the "**scenarioPipi**" and override assignments made to the "**runmodeSub**" scenario.
3. The scenario "**runmodeSub**" is executed on the agents that are associated with the "**scenarioPipi**" scenario.
4. The root scenario cannot be assigned to run on a remote agent.

7.4.5 Running a Distributed Scenario

1. In order to run the scenario, press the "**Play**" button.
2. The system identifies that some of the tests/scenarios are configured to run are on a remote agent and shows the following message.

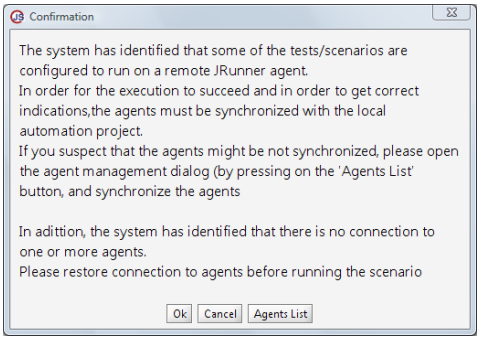

Figure 10: Confirmation Message

3. The user can open the "**Agents Management**" dialog by pressing on "**Agents List**" button.
4. If the system identifies that the scenario was configured to run on an agent that the JRunner cannot connect to, a message is sent to the user.

Once the scenario starts to run, the user can view the progress in two places:

5. Scenario tree

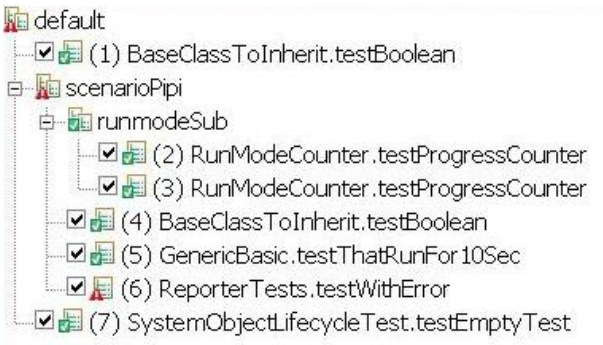

Figure 11: Scenario Tree

6. **Reporter Tab** - A reporter table is created for each of the agents during the run, the icon on the tab indicates the execution state and the table shows the log events received from the agents.

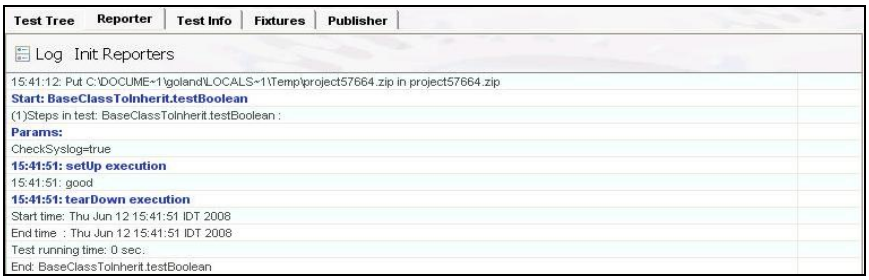

Figure 12: Reporter Tab

7.4.6 Analyzing Results

Once the test execution has completed, the user can now press on the "**Log**" button in the agents "**Reporter**" tab and then analyze the execution results.

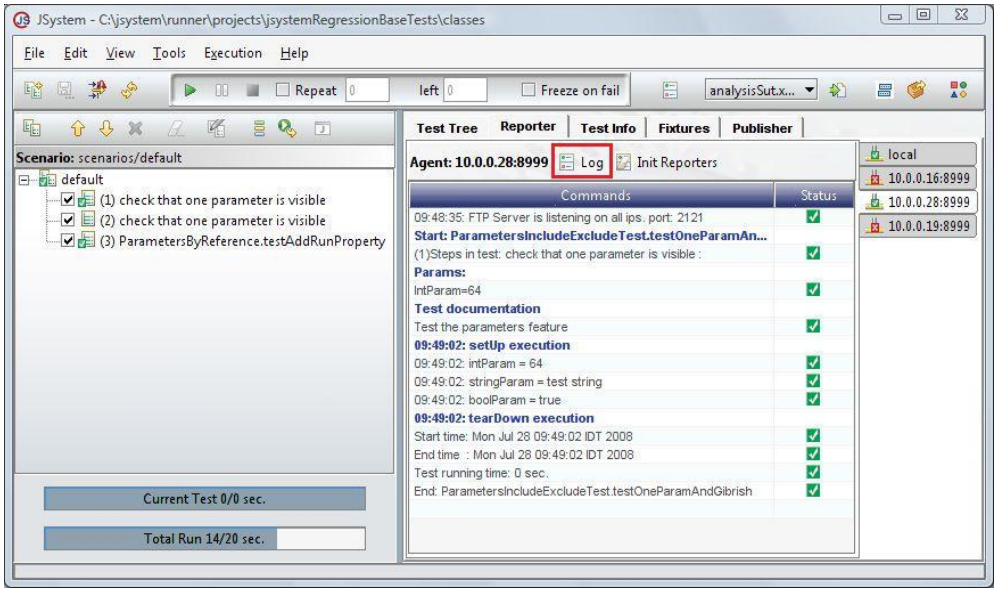

Figure 13: Reporter Tab Execution Results

7.4.7 Disabling a Distributed Execution

In order to disable the distributed execution, select Tools>Define JSystem Properties, click on the "**Agent**" tab, and set the "**ignore.distributed.execution**" to true.

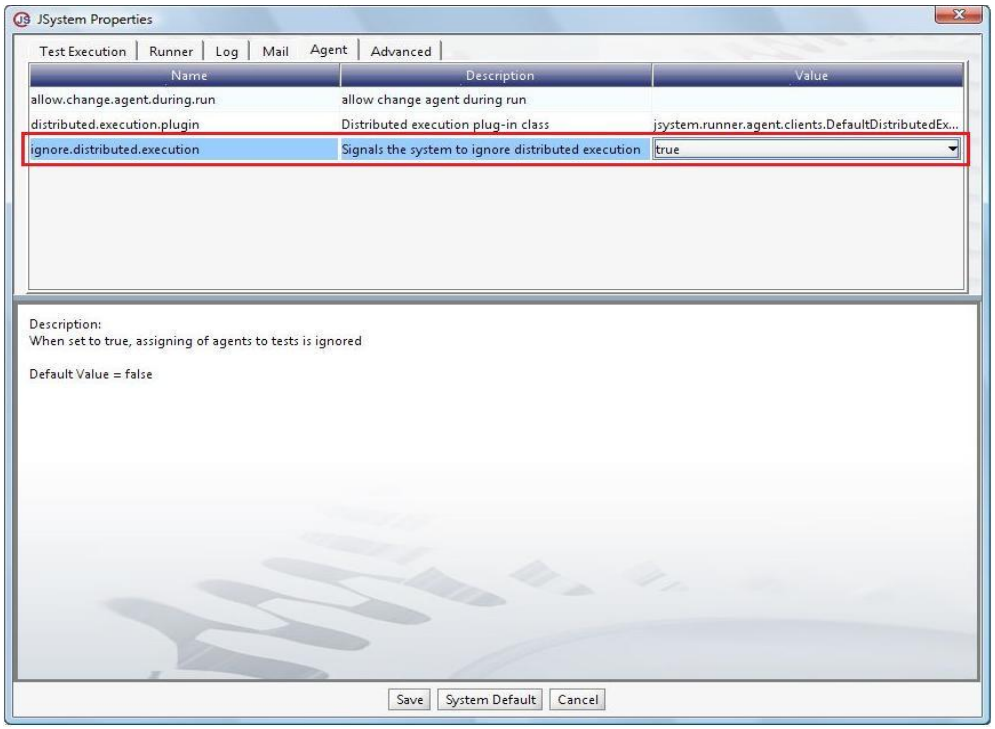

Figure 14: Disabling an Execution

7.4.7.1 Confirmation Dialog

The "**ignore.distributed.execution**" property panel enables the user to easy plan and develop the scenario. While working on the scenario, the user can set the property value to true, and all tests will run locally.

When the property value is set to true, and the scenario is run, the following message is appears.

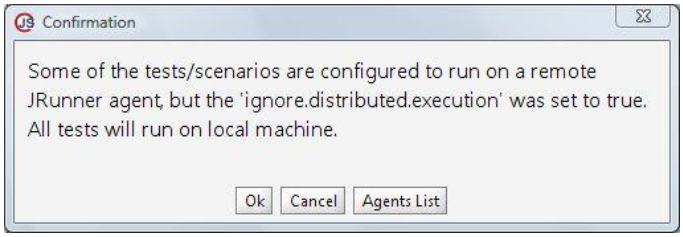

Figure 15: Confirmation Dialog

Once the scenario is ready, the user can set the property to false and the scenario will be executed in a distributed manner.