**User Guide**

**jWebSocket**

**jWebSocket Watchdog Client**

**1.0**

**Control de versiones**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| *1/4/2012* | *1.0* | *Creating the Documment* | *Lester Alfonso Zaila Viejo.* |

**1. Characteristics of the solution**

jWebsocket Watchdog Client is a desktop application that allows you to monitor the jWebsocket server to perform an action when it fails. It is a control mechanism that takes certain actions to be taken such as: notifying system administrators via e-mail or SMS. Letting them know the server status at the time it fails, contributing to efficiency and productivity of software firms.

This solution is different from other remote management tools because it uses the WebSocket protocol to make the test to the jWebSocket server.

**2. Main Features**

**Functionalities of jWebSocket Watchdog Client.**

1. Manage Test.
2. Manage Task.
3. Task run manually.
4. Automatic Task Run.
5. Notify

jWebSocket Watchdog Client can be used mainly in the nodes where required to be in control of the applications that are running on the jWebSocket server. It can be useful if you have multiple servers running jWebSocket thus avoid wasting time on that system administrators have to manually read the log of all servers.

**3. Problems to be solved**

The jWebSocket framework does not allow applications to be monitored and managed remotely separated, consequently resulting in insufficient control of the operation of the servers. Maintain proper operation of the servers that use the framework jWebSocket is the top priority of network administrators. A malfunction would bring instability consequences services for customers and businesses unavailability and economic losses, conveying a lack of credibility to the companies using this framework. For many servers have the network administrator requires much effort and time to do manually review applications as they do not receive a notice from the current state. The network administrator's job becomes cumbersome and expensive in this situation because they do not know the current status of running applications.

 With jWebSocket Watchdog Client will constant awareness of the status of your server, since in the same instant that fails, you will be notified via e-mail or SMS.

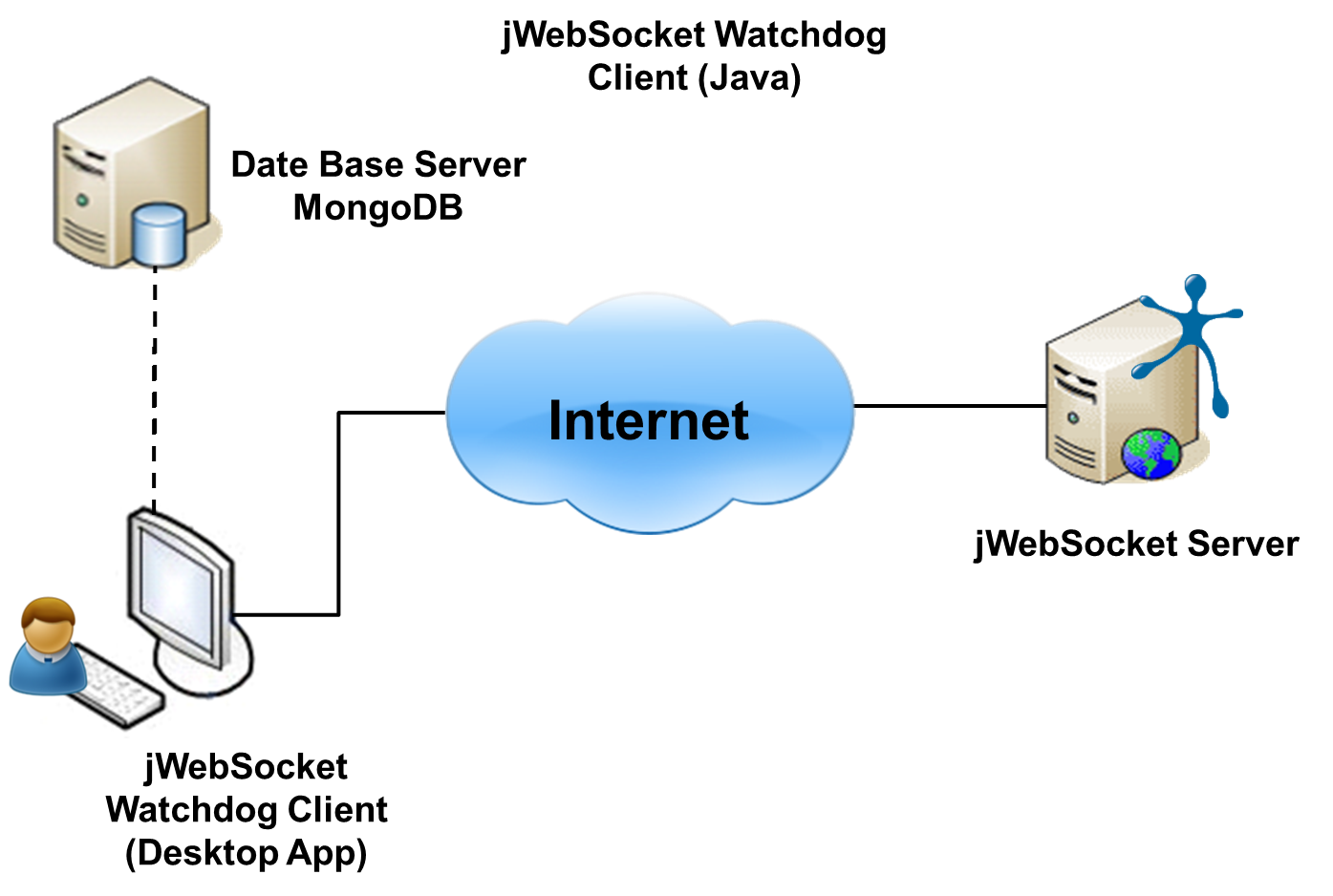
**4. Glossary**

**jWebSocket:** is a new technology aimed at developing WebSockets based applications that provide high levels of speed, scalability, security and real-time work, a key element for the web today.

**Websocket:** WebSocket protocol defines the procedures to update the HTTP connection through a fully bidirectional connection by using TCP WebSocket. The client sends an HTTP GET request to establish communication with the server WebSocket. Subsequently, the communication remains active until it is closed, allowing exchange messages between the client and server.

**SMS: (***Short Message System) Sistema de Mensajes Cortos*

**5. Solution Model**



To use jWebSocket Watchdog Client is required to have previously installed OpenJDK 1.7 or higher. jWebSocket Watchdog Client connects to the database server to load or insert MongoDB tasks to execute the test to the jWebSocket server. The tasks can be run manually or automatically.

**7. Requirements for Use**

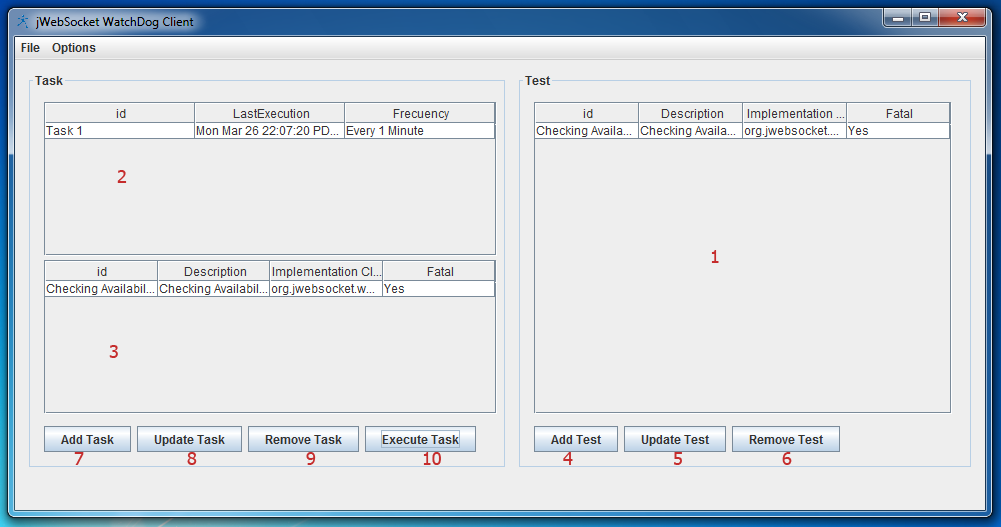
The application has the advantage of working on Linux, Windows and Mac OS, but to work you need:

* OpenJDK 1.7 or higher installed.

**8. Roles of the solution**

**System User:** Interact with the application; you can perform all actions provided by the system.

**9. System Operation**

**Main view of the application:** allows users to interact with the application.

1 -------- Area where the test are described.

2 -------- Area where the tasks and their test are described.

3 -------- Area where the test contained in the selected task with their descriptions.

4 -------- Add a test.

5 -------- Update a test.

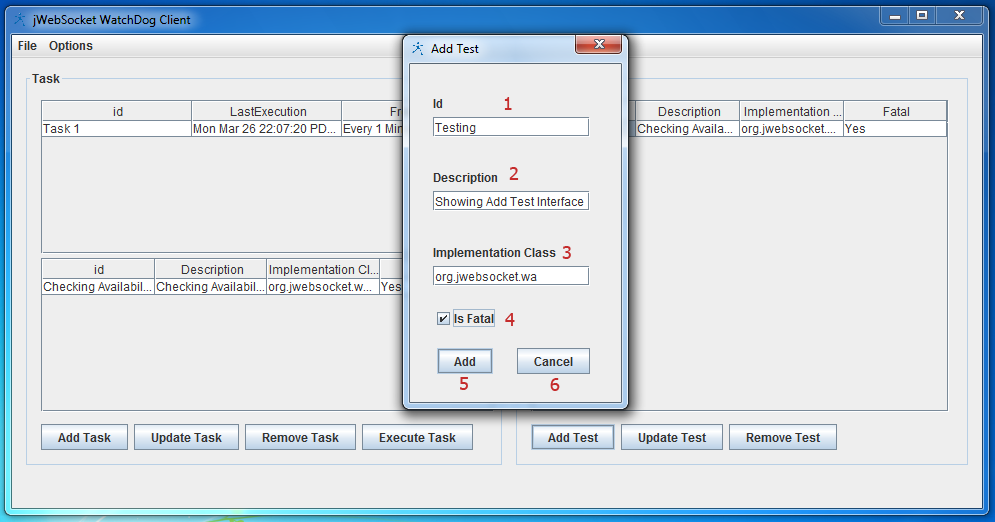
6 -------- Remove a test.

7 -------- Add a task.

8 -------- Update a task.

9 -------- Remove Task.

10 ------ Executes a previously selected test.

**Add test View: **

Users can add a task. They will have to introduce the id, the description of that test, which is that it is the test, especially if it is fatal or not the test to fail, to be used as the denominator when notifying the system administrator.

1 -------- Enter test ID

2 -------- Enter description of the test

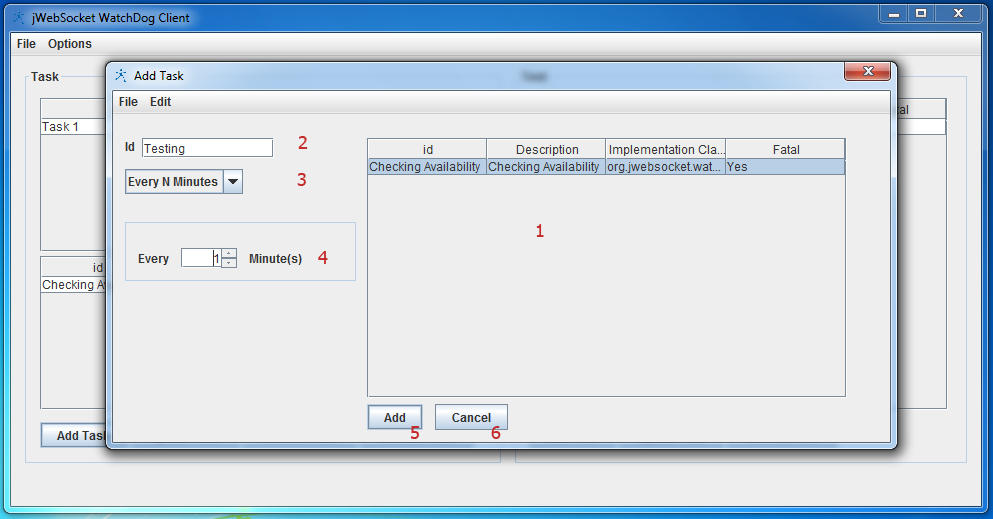
3 -------- Enter where the test is located.

4 -------- Indicate whether fatal or not the test to fail.

5 -------- Once all data is entered, we add the test.

6 -------- Cancel if you want to do it later.

**Add task view:**



1 -------- Task Area

2 --------Introduce Task ID.

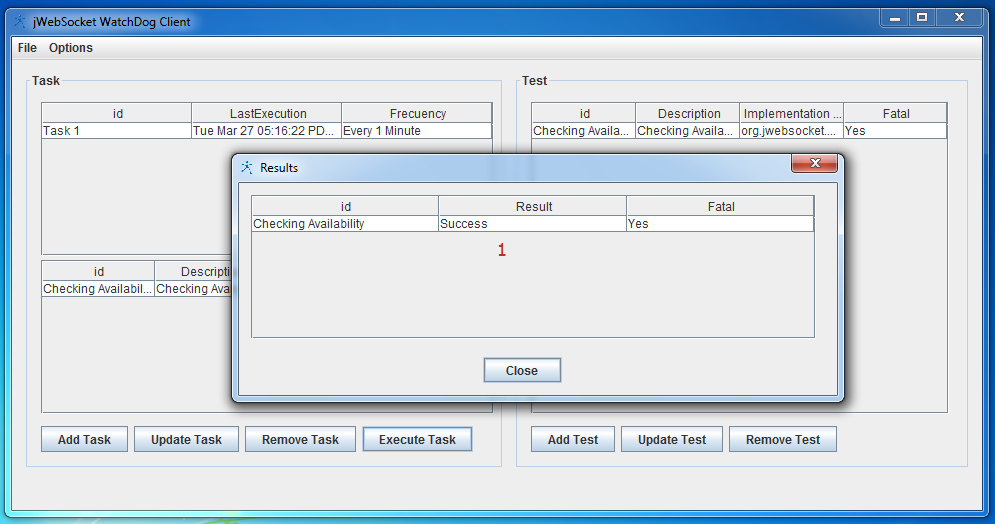
3 -------- Enter the frequency to execute the task.

4 --------Enter each time given the frequency that will run the task.

5 --------  Add the task once all fields are entered.

6 -------- Cancel if you want to do it later.

**Running Task View:**

****

1 -------- Results area showing id of the test, the result of it and if it is fatal or not, the test to fail for it to be notified to the system administrator.

**10. Rules of the solution**

**R1:** If the user deletes a task, it can not be recovered.

**R2:** If the user deletes a test, it can not be recovered.