Requirements Specification

1. Scope

1.1 Overview

The Heartbeat component sends a message to keep a connection or session active. Most network communication is configured to timeout after a specified period of inactivity. The Heartbeat component is intended to bypass this configuration. The message sent is dependent upon the protocol being used.

1.2 Logic Requirements

1.2.1 KeepAlive Request

- Send a KeepAlive request to the destination server.
- The KeepAlive request may involve more than just keeping the socket connection open.
- Custom applications may use a custom protocol that manages state. The Heartbeat component should accept these objects and forward them onto the destination server at the specified interval.

1.3 Example of the Software Usage

An example of the Heartbeat component is in the TopCoder Software Rules Engine to maintain a continuous connection between the application and the server. Without doing so, the client application would need to re-establish a connection and potentially re-authenticate the user after each timeout. The Heartbeat is configured to occur at specified intervals that are less than the timeout period configured on the server side.

2. Interface Requirements

2.1.1 Graphical User Interface Requirements

None.

2.1.2 External Interfaces

None Specified.

2.1.3 Environment Requirements

- Development language: Java1.4
- Compile target: Java1.2, Java1.3, Java1.4

2.1.4 Package

Com.topcoder.util.heartbeat

3. Software Requirements

3.1 Administration Requirements

3.1.1 What elements of the application need to be configurable?

None.

3.2 Technical Constraints

3.2.1 Are there particular frameworks or standards that are required?

None.

3.2.2 Third Party Component, Library, or Product Dependencies:

None.

3.2.3 QA Environment:

- Solaris 7
- RedHat Linux 7.1
- Windows 2000

3.3 Design Constraints

3.3.1 Development Standards:

- It is required that all code be clearly commented.
- All code must adhere to javadoc standards, and, at minimum, include the @author, @param, @return, @throws and @version tags.
    - When populating the author tag, use your TopCoder member handle. In addition, please do not put in your email addresses.
    - Copyright tag: Copyright © 2002, TopCoder, Inc. All rights reserved
- For standardization purposes, code must use a 4-space (not tab) indentation.
- Do not use the ConfigurationManager inside of EJB's. The usage of the java.io.* package to read/write configuration files can potentially conflict with a restrictive security scheme inside the EJB container.
- All code must adhere to the Code Conventions outlined in the following: http://java.sun.com/docs/codeconv/html/CodeConvTOC.doc.html

3.3.2 Database Standards:

- Table and column names should be as follows: table_name, not tableName.
- Every table must have a Primary Key.
- Always use Long (or database equivalent) for the Primary Key.
  For maximum portability, database objects (table names, indexes, etc) should be kept to 18 characters or less. Informix and DB2 have an 18-character limit.

3.4 Required Documentation

3.4.1 Design Documentation

- Use-Case Diagram
- Class Diagram
- Sequence Diagram
- Component Specification
- Test Cases

3.4.2 Help / User Documentation

- Javadocs must provide sufficient information regarding component design and usage.