This page last changed on Jun 07, 2008 by dok.

1. Scope

1.1 Overview

The proliferation of user created content on the internet has lead to the desire for more conversational style interfaces in posting messages to online web pages. This differs from the typical message forum, in that it shares more similarity with IRC than it does a Usenet topic. This format, while not overly conducive to storing and categorizing information, can be much more accessible to those who would otherwise avoid communicating online. By providing individual channels for each user, the barrier of entry can be lowered further still. The purpose of this component, therefore, is to provide a mechanism by which a single thread of communication can be read and added to by multiple users. ~~The method for delivering this information should be in the form of web services, for easy integration onto an existing AJAX web application.~~

In this version of the component, we will remove the web services interfaces with the intention of accessing the component directly. In the future, a web services layer might be added.

1.1.1 Version

2.0

1.2 Logic Requirements

This component will only deal with the storage and retrieval of messages from a backing datastore. Security and registration will be performed by external components and network configuration, if it is desired. ~~With the exception of the first requirement, Web Service interfaces should be provided for all requirements.~~

1.2.1 Create a new Thread of communication.

~~To promote simplicity, the method used to associate a user (or other business entity) to a thread will be performed outside of this component. In order to do this, however,~~ Threads must be identifiable by a unique ID. It is preferred that this be an integral type. In addition to the integral generic identifier, threads should include an optional user defined unique key. This will be used to associate a thread with an external resource. A string should be used for this purpose to provide the greatest flexibility.

1.2.2 View Messages From a Thread

Once a thread has been identified, it should be possible to retrieve a listing of messages associated with it. This listing should support basic bounds for a paging system, but no filtering is required.

1.2.2.1 Provide Web Service Access to Messages

~~A web service call should be provided that will return an XML representation of messages.~~ Remove web service from design.

1.2.2.2. Provide Access to Messages using id

Provide access to retrieve the messages for a thread using the generic integral identifier.

1.2.2.3. Provide Access to Messages using user defined key

Provide access to retrieve the messages for a thread using the user defined key.

1.2.3 Post a New Message to a thread

It should be possible for a message to be posted to a thread of communication. A posting should include the following:

- Name
- Date
- Message
- In addition, it should be possible for the system to store any other information through the use of name/value pairs.

1.2.3.1 Provide Web Service Access for Posting

~~A method to provide the information necessary for a new posting should be provided as a web service.~~ Remove web service from design.

1.2.4 Post a response to an existing Message

The component must provide a method for providing a single response to an Message. It is not required that this response be capable of supporting responses. The data for a response should be:

- Name
- Date
- Message

1.2.4.1 Provide a Web Service Access for posting a response.

~~The method for posting a response to a message should be exposed as a web service.~~ Remove web service from design.

1.2.5. Remove a response

The component must provide a method for removing a response.

1.2.6. Remove a message

The component must provide a method for removing a message. Removing a message also removes its response (if it has one).

1.2.7. Update a message

The component must provide a method for updating the fields of a message.

1.2.8. Update a response

The component must provide a method for updating the fields of a response.

1.3 Required Algorithms

None.

1.4 Example of the Software Usage

A web site hosting personal pages for users could use this as a means to have a very simple message board hosted on each page. The same site could also offer simple boards for commenting on shared user content such as blog postings and videos.

1.5 Future Component Direction

In the future, the component could have moderation added to it. Getting much more complicated than it is, however, would most likely require a mechanism that is fundamentally different and more akin to standard message forums.

2. Interface Requirements

2.1.1 Graphical User Interface Requirements

None. ~~This component will simply define the web services used by a front end.~~

2.1.2 External Interfaces

None.

2.1.3 Environment Requirements

- Development language: Java 1.5
- Compile target: Java 1.5

2.1.4 Package Structure

com.topcoder.messaging

3. Software Requirements

3.1 Administration Requirements

3.1.1 What elements of the application need to be configurable?

The following should be configurable:

- Default number of messages to return per request
- Size of message accepted for postings
- All error messages

3.2 Technical Constraints

3.2.1 Are there particular frameworks or standards that are required?

~~It is required that Web Services be offered. Though~~ The use of framework is up to the designer.

3.2.2 TopCoder Software Component Dependencies:

**Please review the TopCoder Software component catalog for existing components that can be used in the design.

3.2.3 Third Party Component, Library, or Product Dependencies:

3.2.4 QA Environment:

- Solaris 7

- RedHat Linux 7.1
- Windows 2000
- Windows 2003

3.3 Design Constraints

The component design and development solutions must adhere to the guidelines as outlined in the TopCoder Software Component Guidelines. Modifications to these guidelines for this component should be detailed below.

3.4 Required Documentation

3.4.1 Design Documentation

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

3.4.2 Help / User Documentation

- Design documents must clearly define intended component usage in the 'Documentation' tab of Poseidon.