Project #3 – Message-Passing Communication

due Thursday, April 10

version 1.1

Purpose:

This project requires you to design and implement a socket-based[1] message-passing communication channel. Messages are modeled after HTTP messages containing a text header and optionally a body that may consist of binary or text data. The header contains a sequence of lines, the first of which is a command, and remaining header lines contain message attributes[2], e.g., address of sender and receiver, and, if the message contains a body, a content-length attribute that specifies the number of bytes in the body. The message header is terminated with a blank line.

Your communication channel is required to provide, on the receiving end, a dispatcher that posts the message to one of a collection of registered communicators. Each communicator has an input queue and a child thread that processes messages from its queue.

Requirements:

Your Message-Passing Communication project:

1. **shall** use standard C++[3] and the standard library, compile and link from the command line, using Visual Studio 2013, as provided in the ECS clusters and operate in the environment provided there[4].

2. **shall** use services of the C++ std::iostream library for all input and output to and from the user's console and C++ operator new and delete for all dynamic memory management.

3. **shall** use Sockets[5] to implement a Message-Passing Communication facility that transports messages, as described in the Purpose section, to an endpoint specified by an ip address and port number or network name.

4. **shall** provide abstract classes that define message-handling and dispatching and concrete classes derived from those to demonstrate the ability to pass messages. You **shall** also provide a class for constructing and interpreting messages.

5. **Shall** support transporting binary and text files to a specified endpoint.

6. **shall** provide sender and receiver classes that encapsulate the communication channel facilities into easy to use packages.

7. **Shall** demonstrate the construction of a peer-to-peer communication channel that has a sender and receiver on each end and the sending of a request and receipt of a reply on one end and the construction of the reply on the other end.

8. **shall** provide test executive and display packages, that, combined with the communication facility, demonstrates you meet all the requirements of this specification. It is important that your demonstration is accurate, complete, and clear. Your score for meeting requirements will be based on this display. You will get no credit for requirements met but not accurately demonstrated.

9. **shall** provide one compile.bat and one run.bat file that build and then execute your demonstrations Please package your project code, projects, solution, and batch files in a single zip archive. Please do not submit archives that are not zips.

We will use the results you have provided here to build a remote server monitoring facility in project #4.

---

[1] Socket-based means that you may not use Windows Communication Foundation facilities. This will allow your code to run in environments other than Windows.
[2] Attributes are name value pairs separated by a colon, ":".
[3] This means, for example that you may not use the .Net managed extensions to C++.
[4] VC++ 2012 is available in all the ECS clusters.
[5] You may use the Sockets package provided here:
http://www.lcs3.syr.edu/faculty/fawcett/handouts/Coretechnologies/SocketsAndRemoting/code/Sockets/