

T. H. E. SOLUTION LLC

Product Development Consulting

Java Framework for Control Applications
Initial Prototype
Version 0.1

5/2/01

Copyright © 2001 by T. H. E. Solution LLC

1 INTRODUCTION

This document provides the definition and design for T. H. E. Solutions' Java based Control Application Framework. This version of the framework is a conceptual "port" of the C++ Framework for Windows to Java 2 SDK v1.3. The base requirements and basic design philosophy are covered in "C++ Framework and Toolkit for Control Applications, Base Embedded Device Control Applications". The initial prototype provides a limited sub-set of the generic control application (see "C++ Framework and Toolkit for Control Applications, Generic Application Example"). The primary supported features are:

1. Communication links using TCP/IP and the high reliability point-to-point serial protocol but without modem support.
2. A limited command set: connect, disconnect, system status and file upload.

2 PRIMARY DESIGN DECISIONS

From the beginning the primary design decision was to duplicate the capabilities of the C++ Framework, especially the idea of an application being composed of a set of asynchronous components that reside in the framework. Therefore to the maximum extent possible, the over all design remains the same. The principal changes are:

1. The main window and control panel portions of the C++ framework were combined into the MainWindow Java class.
2. The overall interaction model (see figure 1) remains the same but some key implementation specifics have changed. These are:
 - a. Reflection and a single mandatory method, MenuEntry, are used to provide interface polymorphism in MainWindow.
 - b. The general-purpose mailbox used in the C++ framework was replaced with a response linked list to support the asynchronous communication of component completion etc.
3. The communication model and implementation is generally the same as used in the C++ framework. Changes of note are:
 - a. The high reliability point-to-point serial protocol was implemented as a Java native interface at the link level.
 - b. Ports are opened as part of the connection process not when the application is opened.

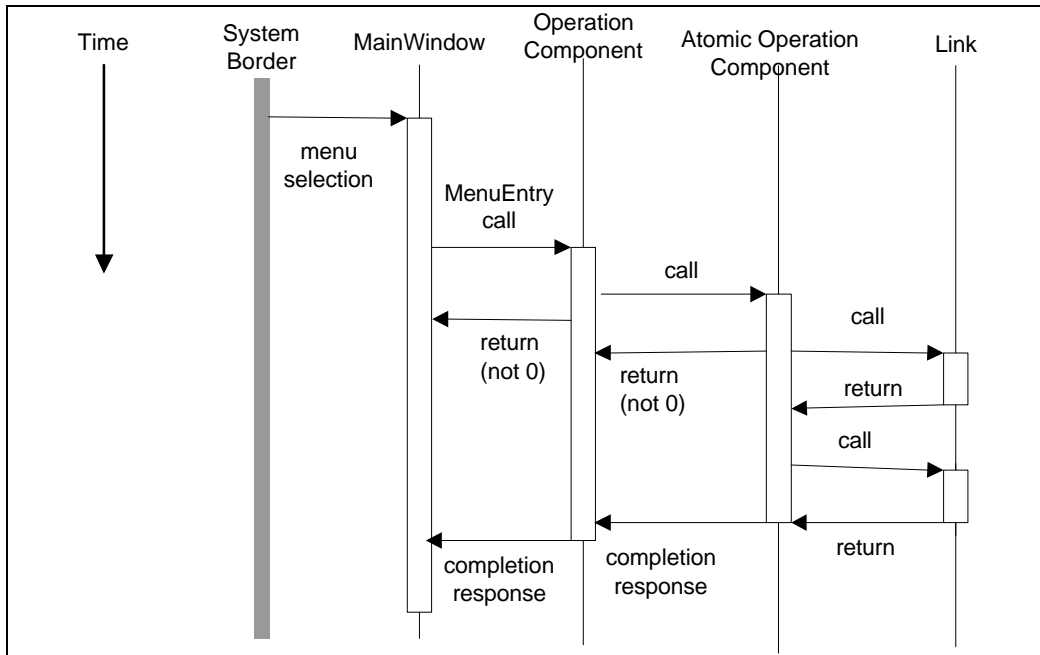


Figure 1, Interaction Model

The only other significant design decision was how to implement the GUI. It was fairly obvious that the Swing GUI toolkit would support quicker development. Using Swing has some negative impact, primarily in terms of performance and platform limits, but for the initial prototype this was not considered to be important.