DYNAMIC LOADING OF MULTIPLE INDEPENDENT CFP APPLICATIONS INTO A SINGLE MAIN VI

Author: Lonnie Haden, Systems Integrator, Certified LabVIEW Architect

Introduction:

The integration of major LabVIEW software modules can be a real challenge, especially when the modules are not interrelated and they have been developed by multiple programmers. This can be made even more problematic when faced with a short development time frame. In this case an industrial control systems consisting of several nearly independent subsystems needed to be developed, tested and deployed in a short amount of time. To further compound the problem system construction schedules dictated that certain subsystems would be brought online at different times. Also, the project timeline was such that multiple programmers were needed to meet the customer's deadlines. The approach taken was for each programmer to develop their own code as stand alone modules that could be debugged and tested independent of the main application GUI (Graphical User Interface).

The Systems:

This application required that software for five different industrial subsystems be written. The first subsystem consisted of heating elements that were used to control the surface temperature of a large casing. The second subsystem was a rough vacuum system used to evacuate the casing. The third subsystem consisted of a second vacuum system that would take over from the rough vacuum system when a certain vacuum level was reached, further evacuating the casing. The forth and fifth subsystems were cooling water systems for the vacuum pumps and other components in overall system.

The hardware that was chosen to control all of the systems in this application was National Instrument's Compact FieldPoint. There were three distinct Compact FieldPoint controllers (cFP-2120) with their associated backplanes. Each of these backplanes was populated with an assortment of Compact FieldPoint modules (various combinations of cFP-AI-111, cFP-AI-112, cFP-TC-120, cFP-RLY-421, cFP-DI-304 and cFP-DO-401). The GUI runs on a server located in a control room adjacent to the system.

The Solution:

The nature of this application required the system to continue to operate if network connections where lost or if the server went down. This in turn required that the software for each module be broken down into a GUI running on the server and a Real-Time application running embedded on the cFP-2120 controllers.

The software architecture for the GUIs consisted of Producer/Consumer Queued state machines which is a very flexible and expandable structure on which to build a robust application. The Real-Time applications consisted of a single looped Queued state machine. Messages between the GUIs and their corresponding Real-Time code are passed using a separate application referred to as the Phone Engine. Identical copies of the Phone Engine reside on both the server and each of the cFP-2120 controllers. The Phone Engine passes messages into the Queues of the appropriate software modules via TCPIP. Other

noteworthy tools used in this application are Queue tools developed in-house that allows messages to be passed between all Queues in the memory space. These tools make message passing between software modules very reliable.

Using this approach the various subsystem could be tested as they came on-line, once fully debugged and tested the software modules where then integrated into the Main GUI which consisted of tab controls, one for each of the subsystems. On each tab of the Main GUI a sub-panel was loaded with the corresponding dynamically loaded subsystem GUI. By switching tab-controls the operator is able to control all the various functions of the system. The main GUI handles closing all the software modules by activating the exit case of each individual software module.

Conclusions:

This software approach allows for the timely development and integration of several software modules into a single application. It also allows for the easy integration of software written concurrently by many different developers. This approach is flexible enough to also allow for easy expansion should system requirements change or if more functionality is required at a later time.

Abstract:

The integration of major LabVIEW software modules can be a major challenge, especially when the modules are not interrelated and they have been developed by multiple programmers. This can be made even more problematic when faced with a short development time frame. In this case an industrial control systems consisting of several nearly independent subsystems needed to be developed, tested and deployed in a short amount of time. Combining independently developed and tested software modules into a single Graphical User Interface using tab-controls and dynamic loading allows for an efficient and flexible approach to many software applications.