CONTROLS THROUGH PICTURES - GRAPHICAL TOOLS FOR BUILDING CONTROL SYSTEM SOFTWARE

S. Hunt, PSI, Villigen, Switzerland

Abstract

Advances in controls hardware performance and reliability have not been matched in the domain of controls software development. Commercial tools are available to improve the software development process, but these are often targeted at the software professional, not the equipment specialist who is often now responsible for low-level equipment interface software. Within the Epics collaboration, graphical tools have been available for some time to aid in the production of systems, but this was restricted to the functional (dataflow) behavior. In order to build the dynamic (state machine) aspects of a system it was still necessary to write code. In Epics this is done in a specialized language State Notation Language) which is C like and provides the necessary constructs to build state machines for Epics systems. In order to improve the development and maintenance of systems for the Swiss Light Source, 'Visual State Notation Language' (VSNL), a tool to graphically build Epics state machines, has been developed. This tool allows equipment specialists to very quickly build the code necessary for low-level control of the accelerator. In place of using a intermediate representation of the states, and saving separately the graphical content, VSNL consists of a set of objects containing both graphical and state information. This approach, makes the program easy to maintain, but also easy to adapt to other output formats and target environments.

INTRODUCTION

Advances in hardware technology (processor speed continuing to double every 18 months), and improving reliability are now taken forgranted when planing and implementing accelerator control systems. Implementing controls software however has remained a time consuming and error prone task. However a number of tools are becoming available to improve this situation.

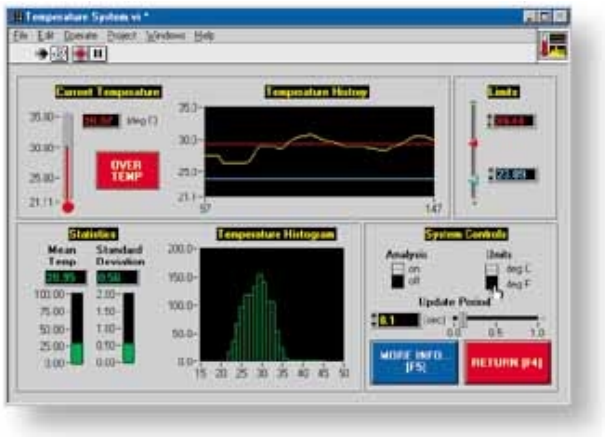
COMMERCIAL TOOLS

Case Tools

Many Computer Aided Software Engineering (CASE) tools are now available to improve the software development process. Often based on a software design methodology such as SASD or more commonly now Object Oriented Analysis and Design they aid the author in the analysis, design and documentation of software. Tools such as Rational Rose[1] and Software Through Pictures[2] take the process further than design - being able to generate templates or code in the chosen language such as C++ or Java. The more advanced tools now allow back annotation or reverse engineering - the ability for changes in the code to be incorporated in the visual design or to import existing code into the design. The negative aspects of these tools is they are aimed only at the software professional, have a relatively steep learning curve, and are best implemented on a project wide basis.

LabVIEW

"Why not use LabVIEW ?", is perhaps the question most asked of controls specialists at the start of a project. LabVIEW[3] is a toolkit which runs on PC's and Unix workstations. It allows control and data acquisition functions to be defined, modified and operated all within a graphical environment. Support is provided for a vast number of PC, VXI and GPIB devices. It is very easy for a user to get started and quickly get data from an instrument or configure a simple control loop.



Despite increased use in the accelerator community [4][5] it has not been widely accepted for control of accelerators and large experimental control systems due to limitations (perceived or real) in performance, scalability, and maintainability of large LabVIEW designs. Although LabVIEW designs can be well structured[6], most are not, and it can be very difficult for anyone but the original author to understand a moderately large design.

TOOLS FROM THE ACCELERATOR CONTROLS COMMUNITY

The Epics Revolution

Epics[7], perhaps to become the ubiquitous accelerator control system toolkit, gives the possibility to non programmers to build accelerator controls. Systems are built using text files which configure the operation of standard functions. Most standard control functions needed are supported (Read from input, Write to an output, PID control, …) as well as data manipulation (convert to engineering units, setting alarm limits, send data only on change). However in almost all real systems some programming is required - for writing support for new hardware, adding new functions, or building state machines. This programming is carried within a well defined structure, and not everyone involved on a project need have this level of knowledge.

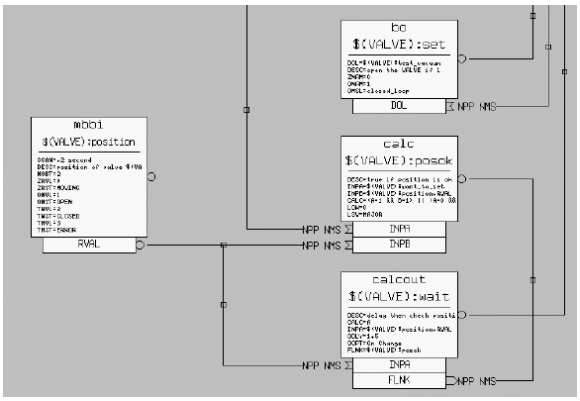
Epics configuration tools

Epics also has a number of tools to aid the developer in configuring her system, these fall into two categories : Forms Based, where the user is prompted to fill in tables of values; Graphical, where the developer builds a visual representation of the control process. Database Configuration Tool (DCT)[8] is a forms based tool, which prompts the user to fill in the necessary values. Basic type checking can be performed, and the user is able to pick from a list of possible values. GDCT [9] provides the same functionality as DCT, but displays each function block graphically, including the links between blocks. This allows the developer to more easily see the structure of the design. Capfast is a commercial schematic editor, which has been adapted to configure Epics control systems. It also displays the function blocks graphically, but has the advantage of allowing hierarchy.

TOOLS WRITTEN FOR THE SLS PROJECT

Visual database configuration tool

Visual DCT is a new Epics configuration tool written in Java. Its development was necessary because of a desire for more features (particularly hierarchical designs) than available with DCT, and the unavailability of Capfast on Linux (our development platform). We would also potentially have had licensing complications distributing Capfast to outside collaborators, institutes and companies who are building components for us. A specification was written at SLS and the implementation was carried out as part of a collaboration, by the Joseph Stefan Institute.



Visually Visual DCT resembles GDCT, users can create, move and link records. Fields can be modified, and any fields not using the default values are displayed. Data flow direction is indicated by an arrow. Records can be grouped together in a logical block, which allows a hierarchy of functions to be built, making the design easier to understand.

Visual State notation language

Visual State Notation Language (Visual SNL) is a graphical tool used to generate Epics State Notation Language Code. SNL gives the ability for the system designer to add state machines in an Epics Input Output Controller (which is normally a VME system running VxWorks). The language has a C like syntax and has constructs for building state machines:

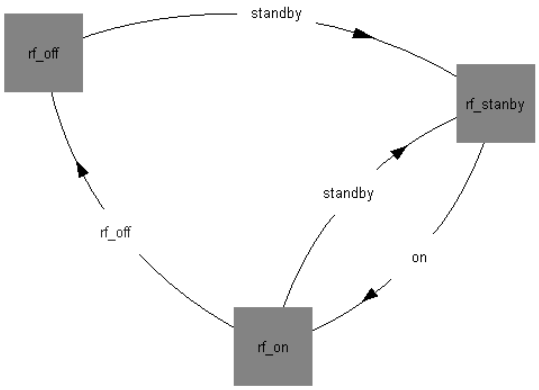
```
state rf_off {
        when (heater==ON && water==OK){
                hv_power_supply=ON;
        } state rf_standby
}

state rf_standby{
….
}
```

The SNL source code is pre-processed to create C code, which is in turn compiled to create a VxWorks object file to run in the IOC. Visual SNL enables non programmers to build these state machines, enables programmers to avoid having to learn yet another language, and provides better documentation.

Visual SNL is written in Java (JDK 1.1),without swing extensions, and so should run in any Java 1.1 enabled browser, or it can be run as an application. Running in a browser has the restriction that it is not possible, without redefining the security configuration, to load or save designs. The user creates on screen a state transition

diagram, and can add, delete, move, and edit states as well as defining transitions between states. New features are being implemented, including the ability to read an existing SNL file.



CONCLUSION

Graphical tools are becoming available to aid the implementers of accelerator controls software. While commercial tools can be used as is, or adapted to meet our needs, there is a place for tools to be built closely tailored to the needs of the accelerator controls community, with a very simple user interface and without a steep learning curve.

REFERENCES

[1] Rational Rose - trademark of Rational Software Corperation
[2] Software Through Pictures - trademark of Aionix Inc.
[3] LavVIEW - trademark of National Instruments Inc.
[4] E.Carlier, et. al., “Outsourcing of the New WaveForm Acquisition, Surveillance and Diagnostic System for the LEP Injection Kisckers”, ICALEPCS 97, Beijing, China.
[5] Xu Jing-wei, et.al., “The Virtual Instrument Control System”, ICALEPCS 97, Beijing, China.
[6] K. Rybaltchenko, “Object-Oriented Technology in LabVIEW Programming”, ICALEPCS 97, Beijing, China.
[7] L. Dalesio, et. al., “The Experimental Physics and Industrial Control System Architecture: Past Present, and Future”, ICALEPCS, Oct, 1993.
[8] J. Anderson, “Database Configuration Tool (DCT) - Tcl/Tk Version for EPICS 3.13”,
[9] J. Kowalkowski, “GDCT User’s manual”.