CROSS PLATFORM SCA COMPONENT USING C++ BUILDER AND KYLIX*

H. Nishimura, C. Timossi and J.L.McDonald, LBNL, Berkeley, CA94720, USA

Abstract

A cross-platform component for EPICS Simple Channel Access (SCA) has been developed. EPICS client programs with GUI become portable at their C++ source-code level both on Windows and Linux by using Borland C++ Builder 6 and Kylix 3 on these platforms respectively.

INTRODUCTION

The Advanced Light Source (ALS) control system is migrating from its original x86-based system[1] to an EPICS-based system[2]. To ease the migration of existing applications to EPICS for the application developers, a client API library called Simple Channel Access (SCA) was developed. Subsequently, SCA was packaged as a Java Native Interface and later, using the same interface, as an ActiveX component for the Windows platform (SCAcom [3][4]). This re-packaging makes the underlying SCA a better fit for both the Java environment and for the numerous development tools on Windows: Visual C++, Visual Basic, Borland Delphi/C++ Builder and LabView.

Recently we packaged SCAcom to make it available both to Delphi 6 for Windows and to Kylix 2 for Redhat Linux 7.3. These tools support development of native and portable applications across platforms using a framework called the Component Library for Cross-Platform (CLX) covering advanced GUI, database and network accesses. On Delphi, the ActiveX control SCAcom was imported into a custom CLX component called SCAclx[5]. For Kylix, SCAcom required some re-writing (keeping the same properties and methods) for compiling in gcc instead of importing SCAcom. With the release of Kylix 3, CLX is supported in the Borland C++ environments (C++ Builder 6.0 on Windows 2000 and Kylix 3.0 C++ Builder on Redhat Linux 8.0) extending the access to SCAclx to C++ applications.

SCACLX COMPONENT

CLX: Cross Platform Environment

Component Library for Cross-platform (CLX) is a framework and component library from Borland[6] shared by its development tools Delphi and C++ Builder on Windows and Kylix on Linux. A C++ program developed on Windows, as a CLX application in C++ Builder 6.0 can also be built to run natively on Linux (and visa versa). These portable programs can also make use of advanced GUI, database and network accesses routines. By creating a custom CLX component (SCAclx) for channel access, the same source code for channel access client programs can be built to native code for both Redhat Linux and Windows.

Porting SCAcom to SCAclx in C++

The ActiveX control, SCAcom, was built with Visual C++ 6.0. We took the C++ source code of SCAcom and modified it to create a CLX component called SCAclx by using Borland C++ Builder 6.0 on Windows. After confirming its function on Windows, the source code was moved to Linux and recompiled under Kylix 3.0 C++ Builder. At the platform specific level, SCAclx calls the same dynamic-link libraries (DLL) that implements the ActiveX control installed on each Windows machine in the control room. On Linux, SCAclx calls into a shared-object libscaclx.a which in turn depends on the channel access and simple channel access libraries sca.a, ca.a and Com.a. Fig.1 shows these layers. Compared to our previous implementation of SCAclx in Delphi, the implementation in C++ was much simpler. In addition to the portability of client programs, the source of SCAclx itself became portable.

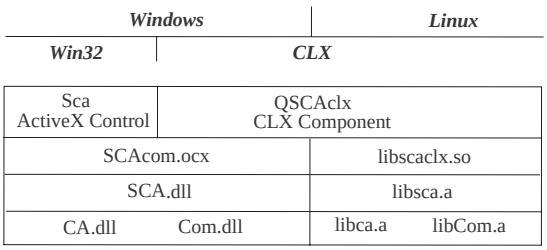

Figure 1: SCAclx and Libraries

Functions of SCAclx

The following members functions and properties have been implemented in SCAclx.

- published property bool GroupCall
- published property bool SetEnabled
- public int Status
- void addInt32Item(AnsiString pvname);
- void addDoubleItem(AnsiString pvname);
- double getDouble(AnsiString pvname);
- int getInt32(AnsiString pvname);
- void setDouble(AnsiString pvname,double value);
- void setInt32(AnsiString pvname,int value);

* This work was supported by the Director, Office of Energy Research, Office of Basic Energy Sciences, Material Sciences Division, U. S. Department of Energy, under Contract No. DE-AC03-76SF00098

The property GroupCall is used to turn on or off the group access mode that becomes important when many channels are added. SetEnabled is for the debugging to disable accidental value settings. A data member Status holds the status of the last EPICS call. The group name is taken from the component instance name property therefore it is not included here.

USING SCACLX

Once the newly developed SCAclx component is installed in both development environments, we can develop an EPICS client programs as CLX programs taking advantage of the other features with a provided by CLX components including GUI, database access and networking. (We prefer to develop in the Windows environment since Kylix has more features). The following example was written to demonstrate application portability. It reads and displays 8 vacuum values. We first developed it on Windows(see Fig.2) and rebuilt it on Linux (see Fig.3). Fig.4 shows this application running on Windows and Fig.5 shows it on Linux.

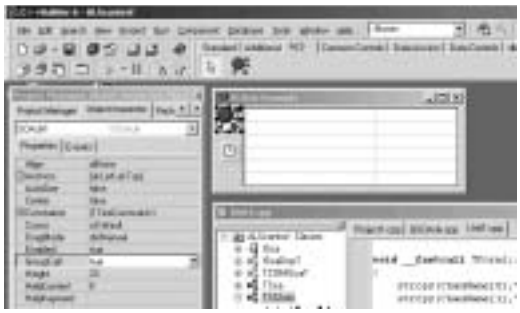

Figure 2: SCAclx on Windows at Design Time

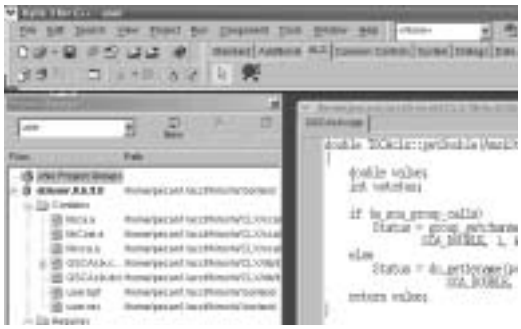

Figure 3: SCAclx on Linux at Design Time

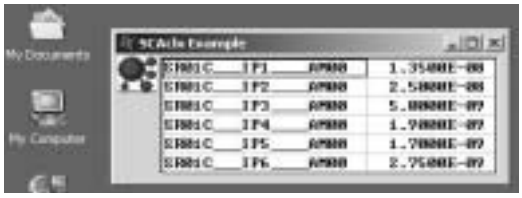

Figure 4: SCAclx on Windows at Run Time

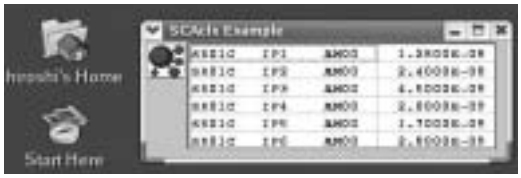

Figure 5: SCAclx on Linux at Run Time

CONCLUSION

The ActiveX component, SCAcom, has been ported to as a cross-platform CLX component called SCAclx. This component can now be used to build native Windows and Linux C++ programs. Compared with our previous effort of porting it for the Delphi Pascal, it much simpler in C++. Without using interpreter languages, we can support EPICS client program development in C++ on both Windows and Linux without compromising the robustness and performance

ACKNOWLEDGEMENTS

We express our thanks to A. Biocca and S. James for their support.

REFERENCES

[1] S. Magyary et al, ”The Advanced Light Source Control System”, NIM A 293 (1990) 36-43, North Holland. S. Magyary, ”Anatomy of a Control System; A System Designerś View”, IEEE PAC93, 93CH3279-7,1811,1993.

[2] L.R. Dalesio, et al., ”The Experimental Physics and Industrial Control System Architecture”, ICALEPCS ’93, Berlin, Germany, 1993.

[3] C. Timossi and H. Nishimura, ”Accelerator Control Software Construction Based On Software Components”, PAC 1997

[4] C. Timossi, www-controls.als.lbl.gov/epics_collaboration/sca/win32/JSca/SCA.JSca.html

[5] C. Timossi, J. McDonald and H. Nishimura, ”Experience with ActiveX Control for Simple Channel Access”, to be appeared in PCaPAC 2002

[6] http://www.borland.com

[7] J.L.McDonald, H. Nishimura and C. Timossi ”Cross Platform Development Using Delphi and Kylix”, to be appeared in PCaPAC 2002