NEW DEVELOPMENTS AT COSYLAB

Igor Kriznar, Grega Milcinski, Mark Plesko, Rok Sabjan, Matej Sekoranja, Igor Verstovsek, Dragan Vitas, Klemen Zagar, Cosylab, Sloveni.

Abstract

The word Cosylab stands for Control System Laboratory, which implies that new developments and technologies are at the centre of our efforts, as we have proven in previous conferences. In this article, we present the highlights of our new developments since the last PCaPAC conference and how they fit together as a whole. The main guideline is simplification for the user, be it the control system developer or the user of the control system. In order to achieve this, we have developed and improved such diverse components as a series of generic client applications, a native Java implementation of the EPICS channel access protocol, implemented load balancing and failover in ACS, replaced an expensive commercial tool with Visual DCT and developed a plug & play control system box with an embedded computer for integrating external devices. The presentation will conclude with successful examples of our technology transfer into other fields of computing and information technologies, such as Geographic Information Systems (GIS), which is becoming a new challenging field for Cosylab.

INTRODUCTION

Cosylab, which stands for Control System Laboratory, has been formally established as a company in 2001, but the key members of the company have been working together on several projects since 1996, starting with the synchrotron light source ANKA at the FZK (Forschungszentrum Karlsruhe), Germany. The control system has been outsourced through a commercial contract to the JSI (J. Stefan Institute) in Ljubljana, Slovenia. There, the KGB group (Kontrol Gruppe für Beschleuniger) has been founded in 1996 by one of the authors of this paper (M.P.) with the very naïve idea that a group of motivated, responsible and skilled students without previous experience in accelerator control systems can build such a system from a ten-pages long wish-list of features and ideas.

The project was successfully finished and so we had a veteran team with an average age of 22. The oldest members have already graduated and left and we would have lost all our investment if we had let this trend to continue. The institute cannot hire the whole team; therefore we have created a spin-off company. True to the research community that we grew in, the vision of the company is to make a living with our work instead of selling software licenses. And true to our philosophy of high motivation, all initial employees are co-owners. The only difference is that now we have to earn our living through commercial contracts and not through government sponsored projects.

We still keep the same model of working with students. Those, who were students on the ANKA project, are now experienced project leaders or technical experts. New good students are invited to work with Cosylab on interesting scientific and applicative projects under the leadership of those ex-students. By now, there are over 40 people collaborating with Cosylab. Because they are students, some work only part-time. Therefore they are effectively providing an equivalent of 17 engineers.

We are aware that the real asset of value in a software and engineering company are its people, therefore we recruit only people of top quality into our team. And not only of technical quality, but also of personal and social integrity.

ABEANS FOR JAPANESE ACCELERATORS

When Cosylab was founded 2001, Hitachi-Zosen (now NDS) was one of its first customers. Since then, we have successfully worked together on more than five control system software projects for various Japanese laboratories [1]

NDS and Cosylab have proven that control system software can be successfully developed on two hemispheres, despite the language barriers and cultural differences. Although unorthodox, many of the methods developed here can be used also for large collaborations such as the GAN. We therefore see our contribution in this field not so much on technical excellence, but as a sociologically successful experiment, which is perhaps even more difficult to achieve for us technical people. It would not have been possible without the honesty and forthcoming of the key people on both sides.

Of course off-site development would became somewhat more difficult with larger projects, but could, with imaginative approach and good specifications, still be done.

When software from both sides was put together, it fit as expected, without the need of on-site installation. We consider this an accomplishment worth mentioning, as this is rarely the case in control system development.

The years of our collaboration also prove that Cosylab can meet the demands from a Japanese company in a tough and competitive environment. We hope that our fruitful collaboration with NDS and possibly other Japanese companies will continue. It has been a challenge to start, but a pleasure to succeed.

ABEANS NEW RELEASE 3.1 AND CS APPLICATIONS

The generic nature of Abeans has been used to address issues related to EPICS control system[2]. This was used in the Control Desk application. Some features that were

added to Abeans for EPICS, can be used also for other control systems.

Abeans directory is already widely used in many applications, PropertyGroup can also be used for TINE and other similar control systems. We introduced several strategies that were used to improve performance of the application, but we must continue doing so.

At the moment we have problems during massive connection requests. Control Desk application can handle up to about 500 channels connection requests in one session. Most of our improvements were on the side of the plug. Now we need to find some improvements inside Abeans themselves and recently some minor improvements were already done. For example, the table application for APS Booster project can already handle up to about 2000 channels. Basaed on our evaluations (see section 5) the recent evolution of Java Virtual Machine will be useful on our path.

THE CONTROL SYSTEM "OFFICE" OR "DESKTOP"

The success of office suites is based on the three fundamentals: consistent look and feel across applications, a common set of data manipulations (navigation, clipboards, undo/redo, copy and paste, find and replace etc) and the ability of different applications to process the same data (a table can be a part of a document in a word processor and a spreadsheet). We have studied the same fundamentals in the context of control system software and shown that the "office paradigm" is relevant to it[3].

To support office-like functions, data beyond raw measured quantities is required: we call this data meta-data, and it encompasses – for example – machine-readable information about how different channels are logically organized into parallel hierarchies, how single values can be combined into group displays and how data items are interpreted differently according to the user interface context. We enumerate meta-data, describe how it is uniformly managed behind the scenes by Abeans, and how it enables generic applications such as table application, chart, archive or even IDE, to interoperate seamlessly as parts of an integrated suite.

In [3] we propose one possible architecture that realizes the Three fundamentals of an office suite: consistency of user experience, shared actions over data and uniform interpretation of data. The proposal is placed into the context of existing EPICS protocol for reliable and efficient data exchange and Abeans for meta-data representation and management. Steps in the direction towards integrating the two have already been made, along with the exploration of meta-data concepts in general.

STUDY AND USE OF NEW JAVA 1.5 FEATURES FOR THE CONTROL SYSTEM OFFICE

Even in J2EE created by professional programmers, the issue of complexity of use was a serious drawback for the users. Abeans Release 3 face similar issues, all stemming back to the fact, that they try to accommodate various communication layers below and arbitrary application models above.

It is therefore our proposal to leverage new Java technologies, such as generics and annotations[4]. The goal of the endeavour would be to create a set of data-source and data-sink components, living in an **existing** framework (such as Java JMX), tied together by **meta-data** stored in a **directory**. Office applications therefore become pieces of code that create **bindings** between sources (managed by next generation of engine that talks directly to communication layer such as CAJ) and sinks (visual components, for instance), and use annotations to specify details for these bindings.

Control System Office would then become much more loosely coupled collection of objects, and annotations would finally provide the mechanism for what we have desired to do before but lacked means of doing so: moving as much as possible from executive to declarative syntax. Of course, not everything is reducible to declarative form – algorithms (equivalent to J2EE business logic) are not; however, a large majority of applications consists of shuttling the data around and playing with its packaging, format and delivery options, which should be representable best by mechanisms specifically tuned for doing this: XML for persistent storage of components, directory for run-time storage of components, and annotations for both persistent and run-time storage of cross-cutting (aspect) oriented metadata.

VDCT - VISUAL EPICS

Visual DCT (Visual Database Configuration Tool) is fast becoming the most popular graphical database[5].

configuration tool for EPICS databases. EPICS is widely used control system based on a real-time database configured from text files. The configuration data comes from two types of files - one defines templates and the other instantiates channels from them. The current EPICS template substitution mechanism is very restricted in its capabilities and it only allows macros to be passed downwards into a template instance. This limitation makes an EPICS database totally flat, which can result in great difficulties when designing complex applications, not to mention maintaining them.

The next major release of EPICS will introduce hierearchy support into EPICS core. Instruments will be provided for templates to export fields that are of public interest and thus resembling mechanisms of object-oriented programming. However, this release may still be some years away, while developers need those features now.

The present version of Visual DCT therefore has full graphical support for hierarchical features, enabling developers of today to use the technology of tomorrow.

VDCT leads the field in feature richness and user friendliness and is the only graphical database configration tool, which not only supports hierarchies in the EPICS databases now, but also uses effective Database Flattening Tool to produce EPICS databases to be compatible with and executable on current core releases.

MICROIOC – A LOW-COST ALTERNATIVE TO VME

Traditionally, most control systems use VME boards to interface to controlled devices. VME boards are expensive, but offer very high performance and a large number of channels. However, there are some clear cases that manifest a clear need for a small, standalone input/output controller (IOC) in a control system [6].

It is very important to realize that a normal PC is not the best for the task. We have therefore built a "microIOC", which is meant for use in industrial/scientific environments and longer lifetimes through a careful design and use of industrial grade components and avoiding all moving parts:

- Embedded controller with fan-less CPU
- No hard disk – booting from a Compact Flash (CF) disk
- A higher quality power supply

To reduce network dependence of the system and reduce the overall network load during booting, the software (operating system, EPICS, runtime application) is locally stored on CF disks providing a cost effective and robust solution.

But an exceptional IOC is much more than just a piece of hardware and operating system. A rich and efficient software toolkit has to be provided to the end developer. Our goal is to eliminate all interfacing and installation effort, such as configuring the network parameters, I/O ports and sorting out naming conventions. Dedicated

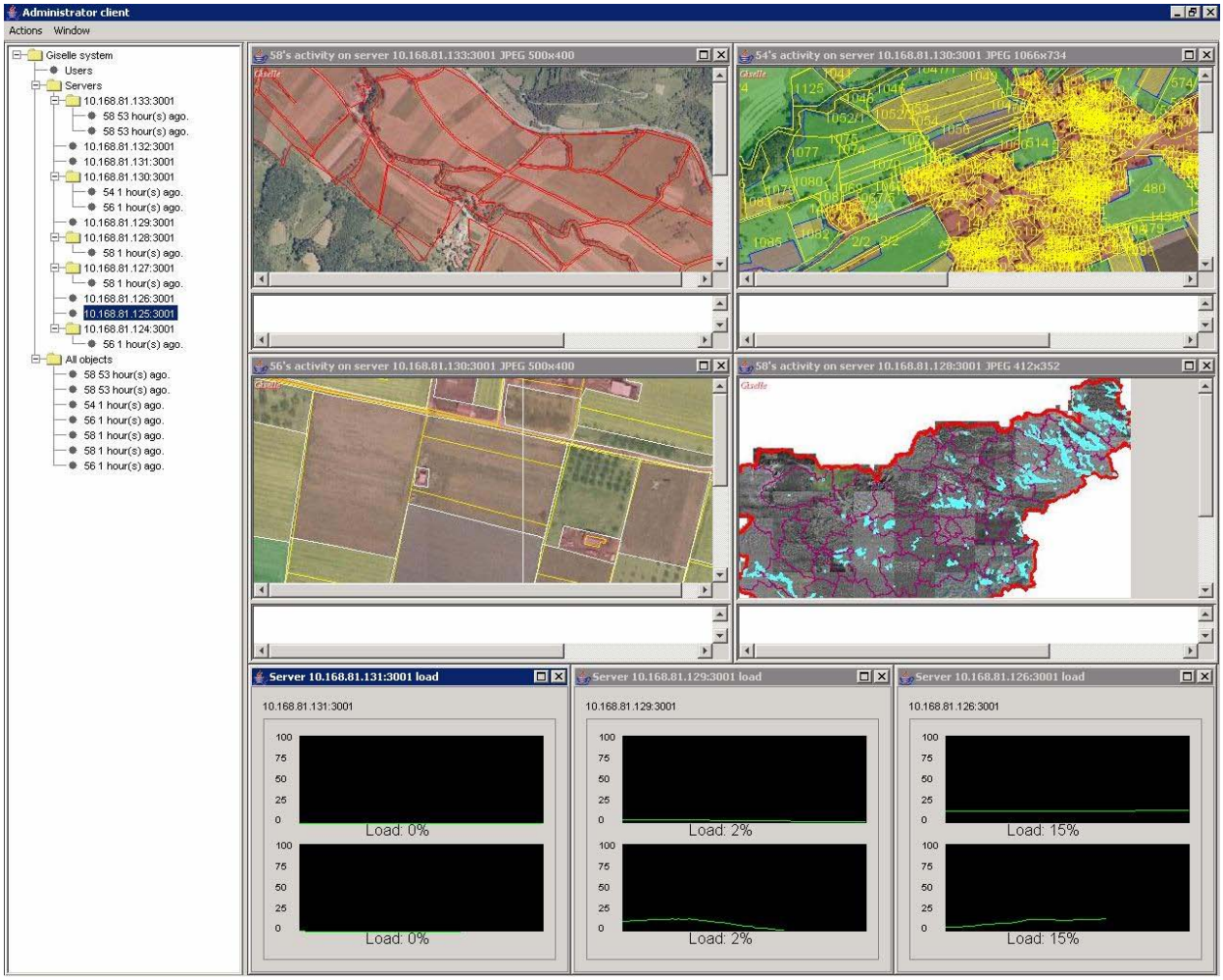

Figure 1: The administration panel of our application server, showing all parallel server machines (left tree) and their CPU load (bottom three windows), all connected users (left tree) and a view of four actual user screens (top right windows), giving total control over user actions.

wizards will make use of pre-prepared templates and device support routines to deploy a working control system node. Some wizards were already implemented (configuration of serial ports through a web page), and more will follow. Additionally, example GUI screens, alarm handler and archiver configuration files are also provided and made available using the web server installed on the IOC.

MAKE A REDUNDANT AND FAIL SAFE CONTROL SYSTEM

On an European Commission funded project with 8 Partners from Austria, Poland, Slovenia, Spain, and Sweden, we are developing a redundant and fail safe implementation of the ACS, a CORBA-based control system that we have jointly developed with ESO and others.

The new fail-safe API provides:

- Improved fault tolerance through replication
- Transparent replication mechanism
- Asynchronous alarm handler, archiver, etc.
- Hot-standby redundant servers, services and manager
- Recovery of servers, services and manager
- Advanced error handling and system diagnostics

One of the features that we get through this approach is a natural mechanism for CPU load balancing, which we use in a completely different environment, namely in an application server for the main Geographical database of the Slovenian Minstry for Agriculture, which manages all EU farmland subsidies. It is completely scalable, allowing several 1000 concurrent users connectring to the main government land parcel database and runs on cheap linux-powered PCs.

REFERENCES

[1] G.Pajor et al., A Long Successful Japanese-Slovenian Collaboration in CS Development, this conference
[2] A.Kosmrlj et al., New Features for New Applications With Abeans 3.1, this conference.
[3] G.Tkacik, Beating Commercial Products: Control System Office and Integrated Development Environment Are The Way To Go, this conference.
[4] G.Tkacik, Generic Types in Java: Abeans<T> specifically for you, Mr T?, this conference.
[5] G.Jansa et al., EPIC(S) developments of Visual DCT
[6] R.Sabjan et al., microIOC and EPICS: The Complete Control System Node, this conference.
[7] K.Zagar at al., ACS as a Dependable Distributed System, this conference.