J. Synchrotron Rad. (1998). **5**, 587–589

A novel approach to the control of experimental environments: the ESCA microscopy data-acquisition system at ELETTRA

Roberto Pugliese,* Luca Gregoratti, Renata Krempska, Fulvio Billè, Juray Krempasky, Marino Marsi and Alessandro Abrami

Sincrotrone Trieste SCpA, SS 14 Km 163.5, I-34012 Trieste, Italy. E-mail: pugliese@elettra.trieste.it

(*Received 4 August 1997; accepted 20 October 1997*)

An efficient control system is today one of the key points for the successful operation of a beamline at third-generation synchrotron radiation sources. The high cost of these ultra-bright light sources and the limited beam time requires effective instrument handling in order to reduce any waste of measurement time. The basic requirements for such control software are reliability, user-friendliness, modularity, upgradability, as well as the capability of integrating a horde of different instruments, commercial tools and independent pre-existing systems in a possibly distributed environment. A novel approach has been adopted to implement the data-acquisition system of the ESCA microscopy beamline at ELETTRA. The system is based on YASB, a software bus, *i.e.* an underlying control model to coordinate information exchanges and networking software to implement that model. This 'middleware' allows the developer to model applications as a set of interacting agents, *i.e.* independent software machines. Agents can be implemented using different programming languages and be executed on heterogeneous operating environments, which promotes an effective collaboration between software engineers and experimental physicists.

Keywords: microscopy; data acquisition; software bus; object-oriented software; IDL.

1. Introduction

Software developers in advanced research facilities often face the problem of integrating different instruments, commercial tools and independent pre-existing systems in a possibly distributed environment. The dream of the experimental scientist is to handle everything through a graphical user interface on some workstations and, if possible, without having to learn new application enabling and development environments.

In this context, system integration is an 'art'. The components, even those whose 'openness' is claimed, present a very low level interfacing capability. Due to the lack of a standardized approach, practically every integration problem is a completely new story, which leads to little or no software re-use.

Moreover, experimental scientists ask for more and more advanced features. Distributed applications present design costs which are not present in unitary systems due to increased control complexity. Intelligent control systems are really welcome in this environment. Applications such as automatic beamline alignment, advanced diagnostics and automatic analysis of the experimental data are still open problems.

With this in mind we have designed an extension of the basic client/server model based on the concept of agents, *i.e.* independent software machines. Applications can be modelled as a set of interacting agents. Each agent is layered on top of an object-based message-passing communication substrate, which transparently manages the complexities of inter-program asynchronous interactions across networks of heterogeneous computers. We have named such communication substrate YASB (Yet Another Software Bus) (Billè & Pugliese, 1997).

2. Software development based on agents

Agents are independent software machines, *i.e.* active objects implemented in software, which are components of a computer system. Programming with agents thus involves constructing different specialized machines and then interconnecting them in order to achieve the required higher-level functionality. This is analogous to the way that hardware designers combine standard components to construct specialized circuit boards.

One of the benefits of the agent paradigm is that agents tend to be relatively self-contained and autonomous. This is because agents are specialized, leading to a tendency to incorporate in them most of the elements required to perform their functionality. As a result, an agent and its environment are not very tightly coupled. This is highly desirable since it enables the agent to be used in different contexts. In order to make the coupling between objects as weak as possible, communication between agents is based on a message-passing model. The essential feature of this model is that information between agents is exchanged by means of an intermediate artefact – a message. The purpose of the message is to reduce the coupling between the senders and the receivers. The only thing that a sender and a receiver must share is the format and general semantics of the message. They do not have to know anything about each others implementation. The sender packs the information that it wants to send into a message and then dispatches it to the destination. When the destination receives the message, it responds by performing an action appropriate to that message.

YASB has been designed to be easily portable across different platforms, integrable with commercial tools and in-home developed ones, expandable but with the mandatory constraint of remaining lightweight. YASB agents can be implemented using different programming languages [C, C++, IDL† (Research Systems, 1997), Labview, Tcl/Tk, Java] and can be executed on heterogeneous operating environments (Unix, LynxOS, MacOS, Win32).

3. ESCA microscopy data-acquisition system

The ESCA microscopy beamline at ELETTRA synchrotron light source is a facility for performing both high-spatial-resolution (<0.2 μm) scanning photoelectron and scanning transmission X-ray microscopy, as well as photoelectron spectroscopy from a microprobe (Casalis *et al.*, 1995, 1997).

The ESCA microscopy data-acquisition system (EDAS) controls the imaging and spectroscopy modes of operation of the beamline. It allows the user to integrate all experiment, beamline and machine operations in a single environment. The system also provides simple data analysis to guide further data acquisition.

† IDL, which stands for Interactive Data Language, was developed by Research Systems. It should not be confused with CORBA's IDL, which stands for Interface Definition Language.

Being a beamline open to external users, it is difficult to foresee the possible requirements needed by any different experiments. A highly flexible and easily maintainable system is thus mandatory.

EDAS is organized as shown in Fig. 1. The instrument, experiment and data manager agents provide a graphical user interface to the system operations and coordinate all the acquisition processes. The instrument manager is a tool which can be used to test and configure the instrumentation of the beamline and experimental chamber. The experiment manager is the core of the data-acquisition system and will be described in detail in the following section. The data manager deals with all the data-handling issues such as archive and retrieval of experimental data and data format conversions. All these agents have been developed using IDL, a complete computing environment for interactive analysis and data visualization. For this purpose, IDL, a powerful array-oriented language with numerous mathematical analysis and graphical display techniques, has been integrated with YASB; as a matter of fact an IDL programmer can access the software bus by using IDL procedures and functions.

At a lower level the instrumentation control agents, developed using the C++ language, provide a high-level view of the beamline and the experimental chamber instrumentation. These agents virtualize all the experimental chamber signal-conditioning electronics, power supplies, and motor systems needed to carry out the experiments (Arun AML stepper motors, Queensgate piezo motors, Scienta power supplies, VSW-Escaton electron analyser, HP and Keithley multimeters *etc.*).

All the named agents run on an HP9000/7xx host under the HP-UX operating system. Communications between the host and the previously described instrumentation is provided by IEEE488 and RS232 serial links. A beamline control system (BCS) (Abrami *et al.*, 1992) agent, which in turn provides access to all the beamline instrumentation (*e.g.* the monochromator, the undulator *etc.*), runs on a separate unit of the VME-based Beamline Control System under the LynxOS real-time operating system.

4. Experiment manager

The experiment manager (EMNG) provides the user with a friendly environment to control each aspect of the data-acquisition process.

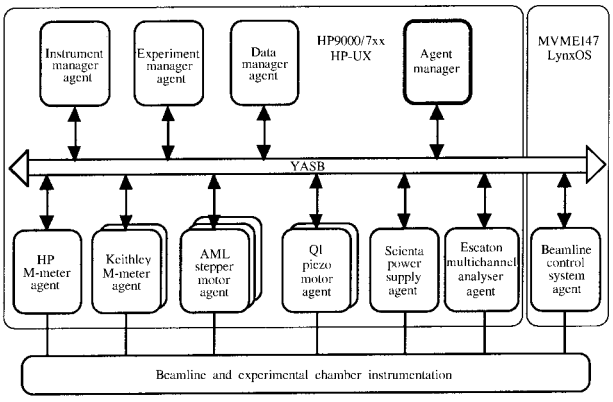

Figure 1
ESCA microscopy beamline data-acquisition system architecture.

Three different data types can be acquired by EMNG: spectra, images and a combination of the two that we call 'spims'. All these data types can be acquired using combinations of different signal sources (*i.e.* 16-channel electron analyser, HP and Keithley multimeters).

Spectra can be acquired using two operating modes: by changing the kinetic energy parameter of the electron analyser or by changing the energy of the photons with setting the photon monochromator. Data can be acquired by counting the number of photoelectrons or by measuring any meaningful physical quantity (*e.g.* the photo-induced current, the photon intensity *etc.*).

Images are collected by raster scanning the sample with respect to the X-ray microprobe using both coarse and fine motors with a resolution ranging from 1 to 0.01μm. As stated above, different data sources can be recorded for each pixel (*e.g.* photoelectron signals, photocurrents, X-ray intensity); data fusion techniques can thus be applied to increase the quality of the experimental data.

Spims are acquired by recording a spectrum for each pixel of the raster-scanned image.

The architecture of the EMNG is described in Fig. 2. Object-oriented design patterns such as model–viewer–controller (Gamma *et al.*, 1997) have been adopted in the development of this agent. When the user starts a data-acquisition action, a new data-acquisition object is created. This object will host all the information needed to identify the experiment, the environmental parameters and the scientific data itself. The data-acquisition object plays the role of model. Each data-acquisition object is associated with at least one interface object that plays the role of viewer and controller. An HDF (hierarchical data format) (National Centre for Supercomputing Applications, 1997) standard data format was adopted to represent and store persistently the data-acquisition objects. New experiments can be imple-

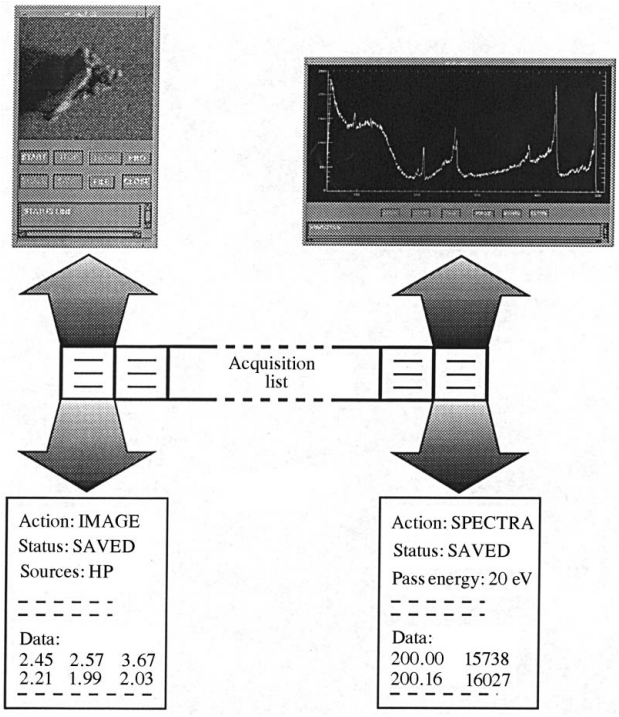

Figure 2
Experiment manager agent architecture.

mented simply by designing a new data-acquisition object and the relative interface objects.

The EMNG provides the user with a complete authoring environment which allows the definition of named data-acquisition actions and the setting of all the relevant parameters. The named actions can then be invoked directly or used to define an action sequence; the action sequence will be carried out automatically by the EMNG.

For example, starting from an acquired image, the user can program an action sequence by selecting a set of points and by associating each point with a specific set of actions. A mini-language based on data-acquisition and movement actions has been implemented; thus, the action sequence facility can be easily used by an IDL programmer to satisfy the requirements of specific experiments.

5. Conclusions

The data-acquisition system of the ESCA microscopy beamline at ELETTRA was implemented using a novel approach which is the result of an effective collaboration between software engineers and experimental physicists.

The system, based on a homemade software bus (YASB), presents a very modular and maintainable architecture. New instruments and measures models can be easily plugged in; new experiments can thus be designed to meet the requirements of the many beamline users.

The adoption of IDL as the implementation language for the user interface and the coordination agents provides the user with a uniform data-acquisition and processing environment; experimental data can thus be analysed during the acquisition process, which allows the evaluation of the quality of experimental data and provides a guide to further data acquisitions.

We believe that the use of middleware and the adoption of standards, especially if they are widely supported, such as HDF, is the key to stop 're-inventing the wheel', reach software maturity and make a technology out of an art.

Thanks are due to M. Kiskinova and G. Morrison for stimulating discussions and suggestions. The work was supported by Sincrotrone Trieste SCpA and EC grant ERBCH6ECT920013.

References

Abrami, A., Cagliardi, F., Galimberti, A. & Savoia, A. (1992). Sincrotrone Trieste Internal Report No. ST/S-TN-92/9. Sincrotrone Trieste, I-34012 Trieste, Italy.

Billè, F. & Pugliese, R. (1997). *Nucl. Instrum. Methods.* A**389**, 110–113.

Casalis, L., Gregoratti, L., Kiskinova, M., Margaritondo, G., Fernandez, F., Morrison, G. R. & Potts, A. W. (1997). *Surf. Interface Anal.* **25**, 374–379.

Casalis, L., Jark, W., Kiskinova, M., Lonza, D., Melpignano, P., Morris, D., Rosei, R., Savoia, A., Abrami, A., Fava, C., Furlan, P., Pugliese, R., Vivoda, D., Sandrin, G. & Wei, F.-Q. (1995). *Rev. Sci. Instrum.* **66**, 10.

Gamma, E., Helm, R., Johnson, R. & Vlissides, J. (1997). *Design Patterns: Elements of Re-usable Object-Oriented Software.* Reading, MA: Addison Wesley.

National Centre for Supercomputing Applications (**1997**). *HDF*, http://opus.ncsa.uiuc.edu/.

Research Systems (1997). *IDL*, http://www.rsinc.com/.