SFC++: A TOOL FOR DEVELOPING DISTRIBUTED REAL TIME CONTROL SOFTWARE

X. C. Pardo, R. Ferreiro & J. Vidal

Dep. Electrónica e Sistemas. Universidade da Coruña
E.S. Mariña Civil, R/ Paseo de Ronda 51, 15011. A Coruña. SPAIN
Tl: 34+81 167000 Fax: 34+81 167101
e-mail: {pardo, ferreiro, vidal}@des.fi.udc.es

Abstract

This paper describes a visual tool for developing real time software for the control of distributed manufacturing systems. The aim of this project, currently in progress, is to get a visual programming environment which integrates both the advantages of object oriented modelling for the design and simulation of systems and the power of modern distributed control systems (i.e. computers with real time operating systems interconnected by means of industrial real time networks). To bridge the gap between the object oriented system model and the implementation level, at which we have multiple parallel tasks running over a network, Sequential Function Charts are used as a standard formalism (IEC, 1988; UTE, 1992) for the description of system dynamics and control software programming.

Introduction

Designing real time control software has become increasingly a complex task that requires the assistance of tools for the modelling, simulation, validation, automatic code generation and ‘on-line’ debugging. More and more, Software Engineering methodologies and techniques have been applied to improve the real time control software development process. In some cases these methodologies has been combined with other formalisms to better adapt them to specify complex dynamics like those of manufacturing systems. Currently it seems clear that Object Oriented Methodologies -OOM- give a lot of advantages in the modelling of systems (Taylor, 1995), and so, in modelling manufacturing systems, mainly since unified method UML (Rational, 1997a) has appeared.

OOM allow us to properly model all of the abstraction levels in defining a manufacturing system: requirements, design and implementation, as well as its different temporal aspects: static and dynamic, and its different views: functional, informational, physical and organisational. Currently the interest of using Object Oriented modelling in manufacturing systems is in defining reference models from which particular models are instantiated and adapted (Huguet and Grabot, 1996; Chalmeta et al, 1997). These models can be seen as part of a more general enterprise ontology as that defined by CIMOSA (AMICE, 1993). CIMOSA encourages the definition of standard reusable models as a way of reducing costs and interchanging enterprise knowledge between tools and users (Heulluy and Vernadat, 1997). So that, object oriented modelling of manufacturing systems is an activity that helps in getting CIM working.

All OOM make use of some formalism for describing dynamic behaviour of objects and their interactions. OMT (Rumbaugh et al, 1991) used StateCharts (Harel, 1987) that have been added to UML, too. In order to get better modelling of DEDS and hybrid systems, OMM have been extended with other formalisms: SFC (Soriano et al, 1996), Petri Nets (Azzopardi et al, 1996), SyncCharts (Andre, 1996), Object-StateCharts (Zaytoon et al, 1996), etc. These extensions allow taking advantage of control engineers knowledge in automation systems and existing tools for the simulation, verification, validation and automatic code generation of those formalisms.

With regard to SFC, its expression power had been compared with that of StateCharts (Boissier et al, 1994) and a lot of work have been made in defining formally its semantics. Results guarantee its deterministic interpretation (Lhoste et al, 1997) and reinforce its synchronous and reactive nature (Marcé and Le Parc, 1993). Besides that, it was defined by a well-known standard increasingly used around the world (Baracos, 1995). It can represent sequence, concurrency and synchronisation of single and hierarchical tasks and it is possible to fulfil steps with different visual or textual languages (IEC 61131-3). So SFC can be used as a visual tool for PLC programming as defined by standard, as well as a general description tool for problems that can be formulated as sequences of operations (David, and Alla, 1992; David, 1995). Even some work has been made using SFC as a control language for real time expert systems, associating rule sets with each step (Arzen, 1991; Arzen, 1993). So that, SFC can be used as a graphical modelling formalism in designing, validating and simulating dynamics of manufacturing systems, as a graphical language for programming their control software and as a management tool in supervisory level sequence control, monitoring and diagnosis, etc.

Nowadays, design methods for getting real time control software have to take into account new communications and hardware devices. Real time industrial networks and field buses as ProfiBus (DIN 19245) and the increasingly number of multiprocessor industrial computers and PLCs requires new control software architectures. Distribution of control software and using of concurrent programming techniques becomes mandatory, but all those issues are far away from control engineers. Modern visual tools must keep away those cumbersome details from them adding automatic generation functions of all software structures needed to get the application working as defined. Automatic generation techniques for distributed software must be developed that guarantee carrying out with semantic defined by system designers. A lot of work has been made in distributed control software design (Dummermuth, 1985; Boudebous and Derniame, 1993; Dangelmaier et al, 1995) and in automatic PLC code generation from graphical languages, mainly Petri Nets (Satoh et al, 1992; Jones et al, 1996; Jones and Karimzadgan, 1997). But there exists little work in automatic generation of distributed control software defined by means of graphical languages, namely SFC (Kouthon et al, 1996).

In short, there exists visual tools for supporting object oriented design and modelling of systems and even automatic generation, to some extent, of code from these specifications (PS&S, 1995; Rational, 1997b). This tools lacks of certain needed functionality when designing control software for manufacturing systems, as support for reference manufacturing models, improved dynamic behaviour specification, design validation or system simulation. Generated code is not well suited for running on manufacturing environments where heterogeneous systems from different manufactures are interconnected by means of industrial networks. On the other hand, widely used control programming tools are mainly oriented to single PLC programming, using in some cases information about control system network configuration as a shortcut accessing those programs but with neither distribution support nor system modelling integration. Great support for automatic distribution, concurrent execution, task synchronisation and on-line debugging is needed without bringing control engineers down to the details.

Objectives and description of SFC++ project

Main objective of SFC++ project is to implement an easy-to-use visual programming tool for assisting control engineers in designing control software for distributed manufacturing systems. SFC++ will assist in system modelling, validation, simulation and automatic generation of code needed to get the system working as defined. SFC++ will incorporate also support for system on-line debugging, supervision and fault monitoring and diagnosis.

SFC++ is intended for control systems with hardware architecture such as shown in figure 1. Single and multiprocessor computers or PLCs with real time operating systems, local networks (real time or not) interconnecting them and field buses for linking to process and other equipment. Such architecture is appropriate for implementing hierarchical structures for integral automation (Chacón et al, 1996).

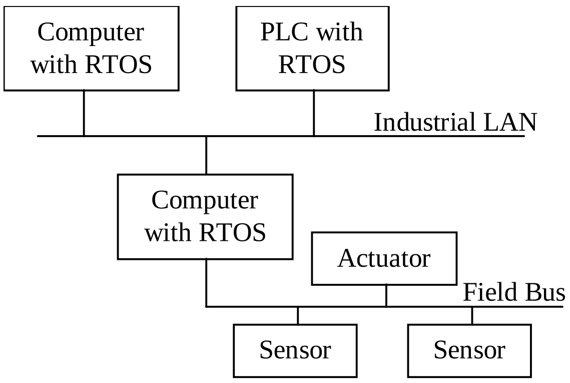

Figure 1 – Control System Architecture.

SFC++ logical architecture is showed in figure 2. As other developing tools is divided in two main components: development subsystem and run-time subsystem. The development subsystem is used for system modelling, validation and simulation before no code is generated. Once the system is running, the model developed is used as graphical interface for on-line debugging. Adding some extra features to modelling (e.g. changing graphical aspect of objects as function of their running state) can allow using that model as part of operator's supervision interface.

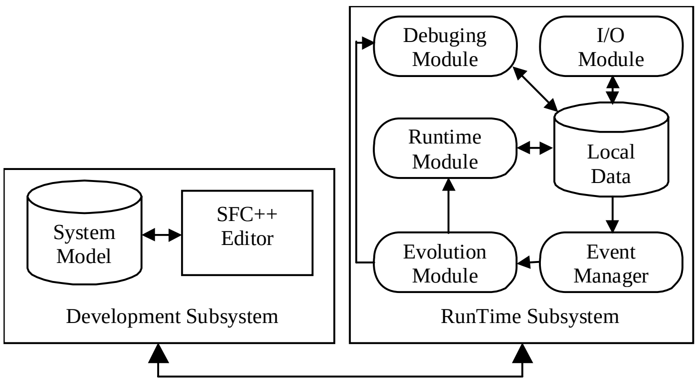

Figure 2 - SFC++ Logical Architecture.

Details of the functional architecture of SFC++ development subsystem are given in figure 3. The main component of this subsystem is a graphical editor that integrates object oriented manufacturing reference models with SFC used as formalism for defining object dynamics as well as sequential operations involving several objects. With regard to SFC as graphical programming language, the aim is to get an IEC 61131-3 compliant editor extended with other useful features (e.g. support for C/C++ code in steps). Such an editor will allow using SFC for traditional PLC programming, distributed real time programming as well as applying it on adaptive control (Ferreiro et al, 1997b), fault detection and diagnosis on hybrid systems (Ferreiro et al, 1998), supervisory control (Ferreiro et al, 1997a) or intelligent control (Ferreiro and Novo, 1995).

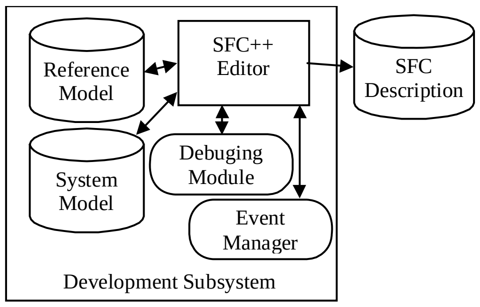

Figure 3 - Logical Architecture of SFC++ Development Subsystem.

As it is shown in figure 3, the SFC++ editor won't be limited to one manufacturing reference model but it would work with different ones provided they were stored in a supported format. Some techniques for automatic verification of SFC models will be included, and for a more complete analysis suitable SFC descriptions would be saved to be used by other programs. Validation would be carried out by simulation of dynamics defined by SFC model or some its parts, using time and event information defined in transitions and I/O information defined in steps. Finally generation, compilation and automatic distribution of modules needed to run the system is done using a C/C++ compiler and dynamic loading capabilities of RTOS used. It must be noted that architecture described in figure 3 could be extended to support multi-user access to system model information and development subsystem capabilities.

Multiple runtime subsystems can be executed simultaneously, each of one is compound of several modules with specific function:

- I/O module, directly coupled with process and responsible for acquiring input and setting output values.
- Evolution module, plays evolution rules of SFC local algorithm.
- Event manager, responsible for detecting local events and synchronising global evolution of control system by means of sending/receiving messages. This module is directly related to evolution module.
- Runtime module, runs defined code in SFC steps.
- Debugging module, gather and exchange SFC situation information and current local data values with development subsystem. It can force data values in runtime as defined by user.

Coupling of both subsystems is done by means of debugging module and event manager ran on development subsystem side. Dynamic code loading capabilities are needed on runtime subsystem side for supporting online changes and automatic distribution of code generated by development subsystem.

Current status of SFC++ project

Implementation of a SFC++ editor is currently in progress. First objective is to get an initial prototype that allows editing complex SFC structures with support of partial grafcets, macrosteps, steps definition with C/C++ code, etc. Second step will be implementation of automatic code generation for single computer environment from such descriptions. This code is intended to be platform independent (POSIX compliant), so some libraries will be developed to allow portability to specific RTOS (at the beginning VxWorks and RMOS). The rest of characteristics will be added as project evolves.

References

Andre, C. (1996). Representation and Analysis of Reactive Behaviours: A Synchronous Approach. In: Proceedings of Computational Engineering in Systems Applications IMACS Multiconference. Symposium on Discrete Events and Manufacturing Systems. Lille. France. pp: 19-29.

AMICE (1993). CIMOSA: Open System Architecture for CIM. Springer-Verlag. Berlin.

Arzen, K.E. (1991). Sequential Function Charts for knowledge-based, real time applications. In: Proceedings of third IFAC Workshop on AI in Real Time Control. California.

Arzen, K.E. (1993). Grafcet for intelligent real time systems. In: Proceedings of IFAC World Congress. Sydney.

Azzopardi, D., Holding, D.J. and Genovese, G. (1996). Object Oriented Petri Net synthesis for modelling a semiconductor testing plant. In: Proceedings of Computational Engineering in Systems Applications IMACS Multiconference. Symposium on Discrete Events and Manufacturing Systems. Lille. France. pp: 374-379.

Baracos, P (1995). Automation engineering in North America. In: Proceedings of 2nd International Conference on Industrial Automation. Nancy. France. pp: 3-8.

Boissier, R., Dima, B., Razafindramary, D. and Soriano, T. (1994). Hybrid systems modelling using Statecharts and Grafcet. In: Proceedings of IFAC WRTP 94. Annual Review in automatic programming. vol 18. pp: 73-79.

Boudebous, D. and Derniame, J.C. (1993). Method for a design of Distributed Control Systems. In: Proceedings of COMPEURO'93. pp: 274-281.

Chacón, E., Moreno, W. and Hennet, J.C. (1996). Towards an implementation of hierarchical hybrid control systems for the integrated operation of industrial complexes. In: Proceedings of Computational Engineering in Systems Applications IMACS Multiconference. Symposium on Discrete Events and Manufacturing Systems. Lille. France. pp: 396-401.

Chalmeta, R., Williams, T.J., Lario, F. and Ros, L. (1997). Developing an Object Oriented reference model for manufacturing. In: Proceedings of IFAC WorkShop on Manufacturing Systems: Modelling, Management and Control. Vienna. Austria. pp: 379-384.

Dangelmaier, W., Felser, W., Henkel, S. and Holtkamp, R. (1995). OOPUS – a distributed system for modelling, simulation and production planning and control. In: ESPRIT Working Group 9245 (Ed.): Improving Manufacturing Performance in a Distributed Enterprise: Advanced Systems and Tools. Edinburgh (UK). pp: 79-88.

David, R. and Alla, H. (1992). Petri Nets and Grafcet: tools for modelling discrete event systems. Prentice Hall Editions, London.

David, R. (1995). Grafcet: A powerful tool for specification of logic controllers. IEEE Transactions on Control Systems Technologies, vol 3, pp: 253-268.

Dummermuth, E.H. (1985). Distributed Real Time Control. In: Proceedings of IFAC Distributed Computer Control Systems. California. USA. pp: 63-67.

Ferreiro, R. and Novo, F. (1995). Bases para el diseño de software orientado al CIM. 5as Jornadas Nacionais de Projecto, Planeamento e Produçao asistidas por computador. Guimaraes. Portugal.

Ferreiro, R., Vidal, J. and Pardo, X.C. (1997a). Adaptive SFC based Supervision Algorithm on Flexible Production Systems. In: Proceedings of IFAC workshop on Control of Industrial Systems. Belfort. France.

Ferreiro, R., Vidal, J. and Pardo, X.C. (1997b). Adaptive SFC based Algorithm on Flexible Production Systems. In: Proceedings of IFAC workshop on Manufacturing Systems: Modelling, Management and Control. Vienna. Austria. pp: 445-450.

Ferreiro, R., Pardo, X.C. and Vidal, J. (1998). Fault detection and isolation on hybrid systems by parity equations. In: Proceedings of Computational Engineering in Systems Applications IMACS Multiconference. Symposium on Discrete Events and Manufacturing Systems. Hammamet. Tunisia.

Harel, D (1987). StateCharts: A visual formalism for complex systems. Science of Computer Programming, vol 8, pp: 231-274.

Heulluy, B. and Vernadat, F.B. (1997). The CIMOSA enterprise modelling ontology. In: Proceedings of IFAC workshop on Manufacturing Systems: Modelling, Management and Control. Vienna. Austria. pp: 301-306.

Hughet, P. and Grabot, B. (1996). Specification of Production Activity Control Systems: Reference Models and reuse of past experiences. In: Proceedings of Computational Engineering in Systems Applications IMACS Multiconference. Symposium on Discrete Events and Manufacturing Systems. Lille. France. pp: 244-249.

IEC (1988). Preparation of function charts for control systems. International Electrotecnic Commission. Publication 60848.

IEC 61131-3. Programmable Controllers – Part 3: Their programming languages. International Electrotecnic Commission. Publication 61131-3.

Jones, A.H., Uzam, M., Khan, A.H., Karimzadgan, D. And Kenway, S. (1996). A general methodology for converting Petri Nets into Ladder Logic: the TPLL methodology. In: Journal of Intelligent Manufacturing, vol 5, pp: 103-120.

Jones, A.H. and Karimzadgan, D. (1997). Transforming automation Petri Nets for control into IEC 1131-3 Instruction Lists using the TPL methodology. In: Proceedings of IFAC workshop on Manufacturing Systems: Modelling, Management and Control. Vienna. Austria. pp: 63-68.

Khouton, T., Decotignie, J.D. and Koppenhoefer, S. (1996). On distribution of Gracet software. In: Proceedings of Computational Engineering in Systems Applications IMACS Multiconference. Symposium on Discrete Events and Manufacturing Systems. Lille. France. pp: 498-506.

Lhoste, P., Faure, J.M., Lessage, J.J. and Zaytoon, J. (1997). Comportement temporel du Grafcet. European Journal of Automation. vol 31.

Marcé, L., and Le Parc, P. (1993). Defining the semantics of languages for programmable controllers with synchronous processes. In: Control Engineering Practice. vol 1. pp: 79-84.
PROFIBUS standard DIN 19245 part I, II & III, translated from german. PROFIBUS Nutzeroorganisation.
PS&S (1995). System Architect: User Guide & Reference Manual. Popkin Software & Systems Incorporated, USA.
Rational Software (1997a). UML 1.0 Specifications. http://www.rational.com/uml. Rational Software Corporation.
Rational Software (1997b). Rational Rose Software Family. Rational Software Corporation.
Rumbaugh, J, Blaha, M, Premerlani, W, Eddy, F, and Lorensen, W (1991). Object-Oriented modelling and design. Prentice-Hall, Englewood Cliffs, New Jersey.
Satoh, T., Oshima, H., Nose, K. And Kumagai, S. (1992). Automatic generation system of Ladder List program by Petri Net. In: IEEE International workshop on Emerging Technologies on Factory Automation – Technology for the Intelligent Factory. pp: 128-133.
Soriano, T., Boissier, R., Razafindramary, D. and Raddadi, M. (1996). Object Oriented analysis of hybrid aspects of a machine tool. In: Proceedings of Computational Engineering in Systems Applications IMACS Multiconference. Symposium on Discrete Events and Manufacturing Systems. Lille. France. pp: 390-395.
Taylor, D. (1995). Bussines Engineering with Object Technology. Ed. John Wiley.
UTE (1992). Function charts Grafcet: Extensions of basic principles. Union Technique de l'électricité. Document UTE C03-191.
Zaytoon, J., Richard, E., Moughamir, S. and Angelloz, L. (1996). A formalism for complex Control Systems: Application to a machine for training and re-education of lower limbs. In: Proceedings of Computational Engineering in Systems Applications IMACS Multiconference. Symposium on Discrete Events and Manufacturing Systems. Lille. France. pp: 385-389.