Modular Integration of Tools for Real-Time Process Supervision

J. Meléndez, J. Ll. de la Rosa, J. Colomer, C. Pous, and J. Vehí.
Unitat d'Enginyeria de Sistemes i Automàtica
Universitat de Girona
Av. Luis Santaló s/n
E17071-Girona (Catalonia)

Abstract **The interest of this work is to apply co-operative ideas by means of expert system's shells that use the object oriented approach to support them, being applied to real processes. The expert system of this work that contains typical fuzzy reasoning and other reasoning mechanisms, also supports qualitative reasoning, abstractive modules for abstracting significative information and current concepts of modularization. The fact is that this system needs interfacing to real processes, instruments and simulation resources for having the enough infrastructure for being applicable to real problems in the field of process control. To take into account openness and maintainability features leads to modular conception of the supervisory framework mainly based on current products that incorporate data exchange and remote calls procedures.**

I. INTRODUCTION

Current process supervision could have the difficulty of introducing new ideas within commercial products for building up appropriate supervisory frameworks with up-to-date technology from research. The many available products have a closed structure that users, and even developers, cannot access. That is, there are not enough open products in the sense to take advantage from other existing tools (simulation or analysis tools) is not possible. Thus, to dispose a simulator with a model of the whole plant or subplants is useful to get information from process levels where sensors could not. This plant model could sometimes used in simulation to get expert rules. To have the simulator and the expert system running together could be useful for example having an adaptable plant model that let choose different rules and expert actions for expert systems. It is conceivable to have a CASD (Computer Assisted Systems Design) available for designing controller to run under the whole supervisory framework.

Research supported by project TAP93-0596-C04-03 CICYT program of the Spanish government.

Another possibility for ES supervisors is to use significative models of controlled plant by using high level qualitative information given by 'abstractors'.

Different supervisory frameworks are in each of these different cases needed accounting the real process and pursued control goals. Thus, a flexible and open structure is needed. Open in sense that communication among several tools (for instance: analysis, simulation, modelling, direct control or abstraction tools) must be supported, and must be flexible to allow the environment to be changed by sharing resources and letting easy set-ups.

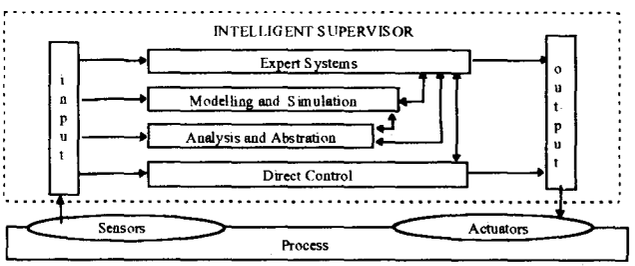

Fig. 1

Therefore, the main goal is to develop an open framework so that experimentation of several supervisory structures could be straightforward by means of providing the supervisory ESs with significant information and designing controllers, based on interconnectivity among developing tools and powerful interfaces with plants.

Once new ideas are in concretion and prototypes are in mind, possible prototypes immediately crashes with closed commercial systems for acquisition, monitoring and process control, due to both business privacy and obsolete technology. This is the case of a new ES that is developed in this university, CEES: to integrate it within real control loops, monitoring and supervisory frameworks is not feasible at once.

II. REQUIREMENTS FOR FLEXIBLE INTELLIGENT SUPERVISORY SYSTEMS

The requirements for integrating tools in a flexible way "Fig. 1" to have an intelligent supervisory environment available with direct interface with plants, are the following:

- *Functionality.* The system must have an interface to acquire data from processes (plants) and controllers, and pass this data to every module of the supervisory system that requires this information. Some direct control actions could also be needed therefore output interfaces are needed with access to plants and to controllers' parameters.

- *Openness.* It must be an open system in the way that any supervisory helpful tool could have a quick and straightforward integration with supervisory environments. The aim is to take advantage of powerful already existing applications and future ones. Thus, dialogue between applications would be senseful to define.

- *Loosely coupled systems.* All these tools could be separately used without the supervisory framework. So, to change the framework would not be essential for doing simulations or analysis of process signals during the design time in the supervisory framework development stages. At the same time to have these tools as supervisory support tools is useful, giving information that could not be supplied by at any moment expert supervisors request it.

- *Control hierarchy.* To define some hierarchical abstraction on plant information regardless a hierarchy in the control framework itself is often interesting. Thus, it is for instance possible to experiment yet in direct control as well in supervisory control based on qualitative models and other type of abstracted knowledge.

III. PROPOSED SUPERVISORY FRAMEWORK

To reach the earliest goals a modular supervisory framework is proposed. Modules have communication capabilities to interact each other. Following the agents' approach [7], each module is a specialist with a special assigned task and can deal with received information from other modules, from the process or from other external sources (for example an ES can receive expert knowledge from plant engineers). For instance, a module could be an ES, a simulator, an abstractor, a controller or an analyser module. The modular architecture does not constrain to work within a only predefined hierarchy and, therefore, to experiment with co-operative solutions among ESs [2] and compare results using other knowledge hierarchies. Also the possibility to settle a hierarchy based on information abstraction is possible using abstractive modules [3].

To take advantage of powerful commercial tools for analysis and simulation, the proposed structure is based on the communication between applications supported by Windows† called DDE (Dynamic Data Exchange). This link between windows applications is based on a master-slave treatment of the applications allowing information and services demand between applications [6]. To reach the global framework a structure as "Fig. 2" shows how integration among commercial products and a shell oriented to develop and design ESs (CEES) is done. The following points describe the relevant aspects of the items presented in "Fig. 2":

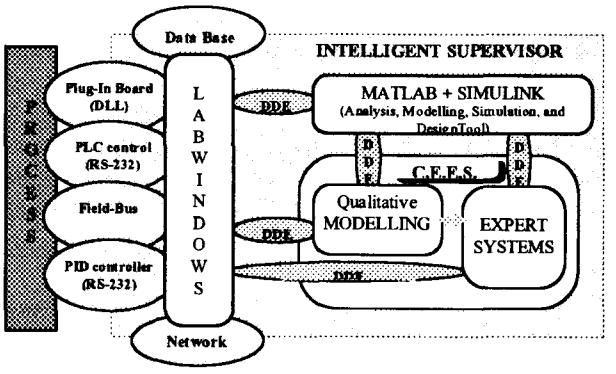

Fig.2 The Proposed Framework

- *Input/output interface*: There are a lot of commercial products presenting good solutions for aspect. Most of them are able to access processes from plants: plug-in data acquisition boards, serial ports (with drivers for PLC lectures), network access capabilities, data base access, and so forth. Moreover, they can share this acquisitions with other Windows† applications using the DDE protocol. Using the same method these products can give and outputs signals to the process. It is also possible to take advantage of other capabilities of this kind of products like filtering, direct control algorithms, analysis functions, and so on (for instance, one product could be LabWindows† or LabView†).

- *Modelling and Simulation:* Are other desirable capabilities for the global supervision framework. Why do not make profit of MATLAB† capabilities using the DDE communication? MATLAB+SIMULINK† and their toolboxes are good tools to be incorporated in supervisory systems

because can provide with simulated information ESs and can act as a analysing tool, as well.

- *Expert Systems and Qualitative Modelling*: To get an intelligent supervisory framework is indispensable to dispose of a shell for de ES development. In this case CEES (C++ Embedded Expert System) is a shell developed at the Universitat de Girona (UdG) within object oriented languages that permits to define qualitative models and co-operative ESs as independent objects with agent capabilities [2].

The fact of modular structure suppose some advantages like:

- Redefinition of the main problem into simpler subproblems with particular goals. This is helpful for engineers in order to get a methodology to deal with large supervisory and control problems.

- The alone validation of each module: Partial objectives are reached obtaining partial results in short time. This allows changes in the design of the supervisory framework before projects have finished.

- Abstracted information: The definition of each expert systems modules at design time could be done independently of the kind of information they had to use. For instance, ESs can deal with qualitative information and a specialised module, called abstractor, assume the role of converting signal (data) from supervised and controlled plant process into qualitative information. This could be conceived as a generalisation of current numeric-symbolic interfaces for Ess.

- Module co-operation: Modules definition is not in the sense of input-output modules but they allow communication in the sense of co-operation. Modules can ask for information to one or different modules and valorise this information depending on the prestige of each answering module, and on necessity of asked information.

IV. ABSTRACTORS

Current trends in intelligent control, as seen in bibliography, are to defining supervisory integration at different levels [1] with respect to scales of abstractive processes of supervisory knowledge on plant. On the other hand, to build up a flexible supervisory structure, that could be able to work both in hierarchical supervisory framework and in co-operative one [2], having knowledge structured by specialised abstractors and ESs that deal with supervisory task from different viewpoints and knowledge.

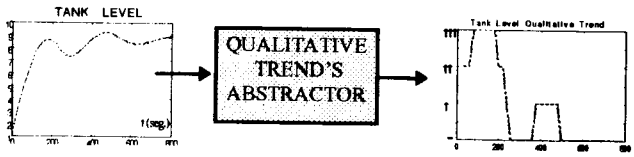

Fig. 3 An Example of an Abstractor

Abstraction on information is done at different stages by specialised modules (abstractors) from processes data "Fig. 3". The overall system could have numeric, statistic, and qualitative abstractors that provide with information with various sorts of meaning. An ES can also act as an abstractor, intelligent abstractor, dealing with provided information following expert knowledge from plant engineers together with elaborated process information.

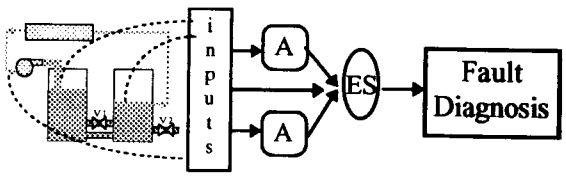

Fig. 4 A: Abstractor (qualitative output), ES: Expert System

The proposed supervisory framework, the abstractive knowledge is important if modular structures is intended to be declared, because these would be the connective nexus among processes and ESs.

V. CO-OPERATION

Co-operation among expert systems is conceived in CEES [2] as the possibility to define modules equipped with 'Inference Engines' local knowledge bases, working for achieving local goals so that require information from other both intelligent or conventional modules. This information is necessary reviewed in terms of certainty so that consistency within a co-operative framework could be maintained. This consistency is supported by criticisms functions based on conventional *prestige* ("confidence on sources") parameters and *necessity* ("confidence on information itself") parameters that are coded as intelligent rules within each knowledge base of the cluster of ESs.

Basic features of high level communication of co-operative ESs are partly efficiently solved by means of object oriented languages, therefore this framework should be developed mainly on software that compatibilise with this philosophy, as CEES applies.

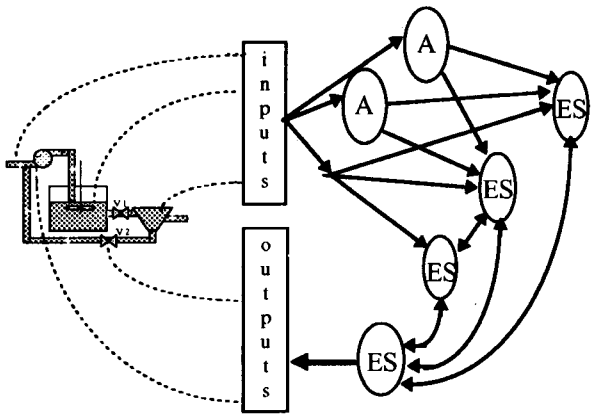

Fig. 5
A: Abstractor (qualitative output). ES: Expert System.

VI. STATE OF THE PROJECT

The framework prototype here proposed is now under development. Nowadays the Shell CEES [2] is already developed, and it lets the modular implementation of co-operative ESs based on various sorts of knowledge and the creation of qualitative simulators for studying co-operative structures ("Fig. 5"), working on MS-DOS operating system. Some abstractor modules are already created so that this information is available for typical ES diagnosers ("Fig. 4"). Next step is to reshape this shell for working on Windows so that the DDE interfacing utility could be applied on already developed communication software between C++ applications and MATLAB. Communication via DDE of C++ software with process and instrumentation interfacing LABWINDOWS is under development.

VII. CONCLUSIONS

Dealing with supervisory frameworks contains both technological and research development. Current research results have some difficulties to be as fast as possible transferred to technology. However, ideas of open systems, large scale systems and intelligent control are largely consensed as necessary to be applied. These ideas converge on modular systems and could be solved by means of co-operative frameworks. These frameworks are under development both on high level protocols level and distributed solver systems. For the specific field of process control abstractive knowledge (mainly based on signal treatment and statistics) is a promising work line so that ESs could work with further significative knowledge. All these ideas need of economic environments that could be Windows based software.

A DDE based structure extremely leads to take advantage of calculus, reasoning and interfacing resources because of introducing CASD tools (MATLAB) in supervisory processes. Therefore, parallel analysis, design and test with more meaningful information is possible.

However, it is not intended to assert that the here proposed structure was solution for all problems, and, according to technical specifications, there already exist outstanding commercial "Real-Time Expert Systems Shells" as G2, Cogys, RTAC [5]. This is another proposal for integrating new research ideas with current open systems in order to gain maintainability, openness, functionality, so that calculus, reasoning and interfacing resources could be further exploited.

Of course, there are constraints and difficulties come across, like communication via DDE gives you facilities for passing any type of information, both numeric or qualitative. This implies receiving modules must be configured for receiving many types of information. Object oriented software has utilities with objects themselves for dealing with different incoming information by means of polymorphism of object methods, but not all current applications are object oriented.

VIII. REFERENCES

[1] Harris, C.J., "Problems and Progress in Intelligent Systems Control", Intelligent Systems Engineering pp.1-6,Conference Publication no395 IEE,1994.

[2] de la Rosa, J.Ll. "Heuristics for Cooperation of Expert Systems. Application to Process Control", doctoral thesis 1994, Chapt 4-6, ISBN 84-605-0275-9.

[3] Colomer J. et alt., Technical Note 95/003-EI, Departament d'Enginyeria Industrial, Universitat de Girona.

[4] Nacsa, J. Kovacs, G. L., "Communication Problems of expert Systems in Manufacturing Environment", Artificial Intelligence in Real-Time Control, (AIRTC'94), IFAC , Valencia. Spain 1994.

[5] Laffey T.J., et alt. , real time knowledge based systems, AI Magazine, 9,1,27-45, (1988)

[6] Marquès, M., "Estudio y ejemplos de DDE con MATLAB y creación de MEX-Files", Final Graduation Project., E.U.P. Vilanova i la Geltrú,- UPC, Feb-1995

[7] R. Smith and R. Davis, "Framework for Cooperation in Distributed Problem Solving", *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-11, nº 1, January 1981.

† MATLAB and SIMULINK are a Trade Mark of MathWorks. Windows is a Trade Mark of Microsoft Corporation. LabView and LabWindows are Trade Mark of National Instruments Inc.
Authors on leave at

Fax: +34 72 41 83 99 and e-mail: peplluis@ei.udg.es