Multi-agent Platform and Toolbox for Fault Tolerant Networked Control Systems

Mário J. G. C. Mendes
Instituto Superior de Engenharia de Lisboa, Dept. of Mechanical Engineering, Lisboa, Portugal
Email: mmendes@dem.isel.ipl.pt

Bruno M. S. Santos and José Sá da Costa
TULisbon, Instituto Superior Técnico, Dept. of Mechanical Engineering, Lisboa, Portugal
Email: bruno.santos@bluecape.com.pt
Email: sadacosta@dem.ist.utl.pt

Abstract— **Industrial distributed networked control systems use different communication networks to exchange different critical levels of information. Real-time control, fault diagnosis (FDI) and Fault Tolerant Networked Control (FTNC) systems demand one of the more stringent data exchange in the communication networks of these networked control systems (NCS).**

When dealing with large-scale complex NCS, designing FTNC systems is a very difficult task due to the large number of sensors and actuators spatially distributed and network connected. To solve this issue, a FTNC platform and toolbox are presented in this paper using simple and verifiable principles coming mainly from a decentralized design based on causal modelling partitioning of the NCS and distributed computing using multi-agent systems paradigm, allowing the use of agents with well established FTC methodologies or new ones developed taking into account the NCS specificities.

The multi-agent platform and toolbox for FTNC systems have been built in Matlab/Simulink® environment, which is in our days the scientific benchmark for this kind of research. Although the tests have been performed with a simple case, the results are promising and this approach is expected to succeed with more complex processes.

Index Terms— **Networked control systems, Fault-tolerant systems design, Decentralized design, Multi-agent systems, Graph partitioning**

I. INTRODUCTION

Fault management and maintenance systems are today an important part of the automation process in every factory, they should include the knowledge about failure modes and their causes and also, they should give, as soon as possible, information about the presence of faults in the processes. This is the task of responding to abnormal events in a process, which involves the timely detection of an abnormal event, diagnosing its causal origins and then taking appropriate supervisory control decisions and actions to bring the process back to a normal, safe, operating state or, at least, with minimal process operation degradation. Resuming, these are the fault tolerant control (FTC) systems (Fig. 1).

It is this kind of supervision and control activity that, in the last decades, the scientific community in collaboration with industry, try to make it more and more automatic and human free. Fault-tolerance should be a key requirement for the control of modern processes [1]. Several different methods and techniques to deal with FTC systems can be found in the literature [2]–[4]. However, these methods and systems normally are globally designed and centralized, not being possible to use information and collaboration of different FTC stages to achieve a common goal. At the same time, the industrial processes usually are complex, physically distributed and heterogeneous, imposing a modular (physical or functional modules) and decentralised approach. Thus, there exist a need of new FTC systems in the actual industry and it will be a good option to make use of the Distributed Artificial Intelligence (DAI) techniques. Namely, the methodologies based on the multi-agent systems (MAS) can be a good option to create a distributed, modular and collaborative FTC system for the industrial processes [5]–[7].

Also, technological advances are delivering devices that

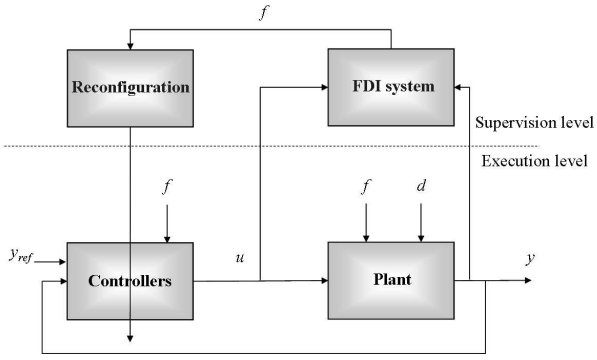

Figure 1. Typical centralised FTC system.

This work is partially supported by the "Programa do FSE-UE, PRODEP III, acção 5.3, no âmbito do III Quadro Comunitário de apoio" and by the project POCTI/EME/59522/2004, co-sponsored by FEDER, Programa Operacional Ciência, Tecnologia e Inovação 2010, FCT, Portugal.

have sensing and communication capabilities embedded in the physical world. The communication network is often adopted in many control and management systems to fulfil the information exchange and control signal transmission among system components. This type of systems, named networked control systems (NCSs) are systems whose sensors, actuators, estimator units, and control units are connected through communication networks having a pervasive mixed data flow with both time-critical data (periodic variables and events) and non-critical data (messages), as in the Fig. 2. The study of NCSs is an emerging interdisciplinary research area, combining among others control theory, communication theory and computer science. Significant research work has been done within the last decade on NCSs, making available systematic stability analysis and design tools from a different point of view, [8], [9].

This type of system has the advantage of greater flexibility, reliability and testability with respect to traditional control systems and enhances resource utilization. Also, it allows reducing wiring, as well as lower installation cost. It also permits greater agility in diagnosis and maintenance procedures. As any other system, NCSs are subject to faults or malfunctions or simply performance deterioration of equipments, network or processes, forcing to interrupt the normal operation and, if not detected in early stages, frequently leading to expensive repairs. To avoid performance deteriorations or components damage and humans risks or, at a larger scale, ecological disasters, faults have to be found as quickly as possible and actions that stop the propagation of their effects have to be taken. These actions should be carried out by appropriate control equipment through a fault diagnosis (FDI) system to detect (FD - Fault Detection), isolate (FI) and identify the kind of fault and its severity, and an appropriate tuning or re-design (FR) of the control algorithm that adapts to the faulty plant condition. If these actions are successful, the system operation will be satisfied and performance fully recovered, or slightly decreased in cases of serious faults. Examples of such NCSs can be seen in aircrafts, cars, building supervision, manufacturing plants and many other common systems.

In this paper, it is proposed an innovative FTC system, relatively to the traditional methodologies that have been used, in the last years [2]–[4], [10], [11], a Fault Tolerant Networked Control System based on Multi-agent Systems (FTNCS-MAS). It is proposed a new platform and toolbox for the FTC design in complex NCSs, having in mind their distributed nature, using the distributed computing power given by software agents organized in society, forming a multi-agent system spread through the NCS, and minimizing the number of critical communications needed among parts of the system to guarantee safe operation and performance even in faulting conditions, an approach like the Fig. 3. It is a hybrid framework involving simultaneously decentralized and centralized topology, independent of the methodologies used to tackle the FTC design. This platform resembles a shell to be fulfilled with FTC methodologies devised for centralized FTC problems or new ones directly developed taking into account the specificity of the NCS under study.

The paper is organised as follows. After the Introduction, Section II presents the multi-agent architecture used in this work. Section III describes the new toolbox proposed and the implementation of the new multi-agent platform for FTC systems. Section IV presents the benchmark used for test bed of the new platform and introduces the optimal partitioning, needed for implementing the proposed FTC multi-agent architecture in order to minimize critical data communications within the NCS. In Section V some platform tests and results are presented and discussed. And finally in the section VI some conclusion remarks are given.

II. MULTI-AGENT ARCHITECTURE

An agent is a computer system that is situated in some environment and is capable of autonomous action in that environment in order to meet its design objectives [12].

A multi-agent system (MAS) is, in a simple way, a problem of putting the agents together (agent organisation and society) (Fig. 3). Several different multi-agent architectures can be found in the literature [13]. The architecture proposed here for the FTNC multi-agent system is an unilateral relation MAS modified federated architecture coordinated by facilitators (Fig. 4), where

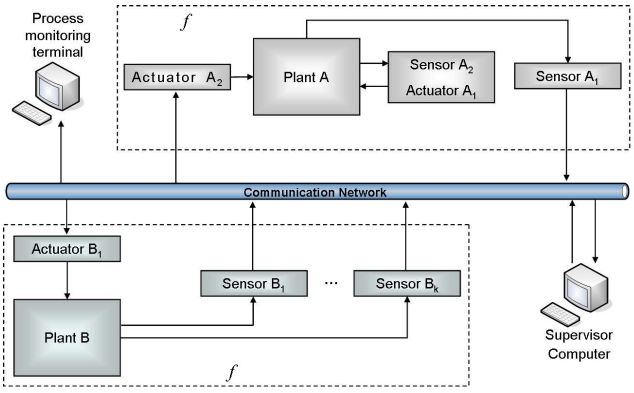

Figure 2. Networked control system.

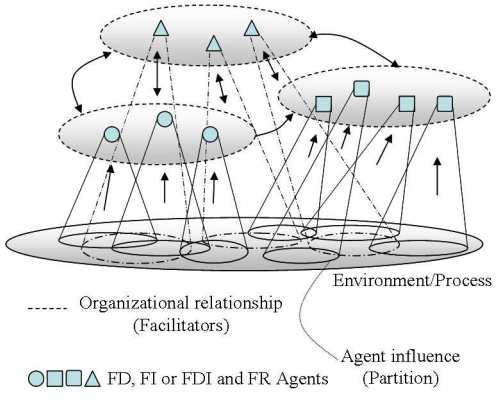

Figure 3. FTNC multi-agent system.

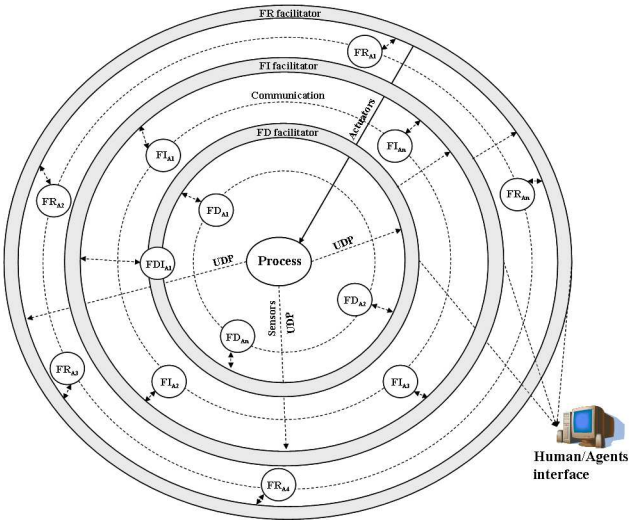

Figure 4. The proposed MAS modified federated architecture.

one agent depends on the other, but not vice versa [14]. The facilitators deal, respectively, with the unilateral communication between the different FTC task agents, manage de corresponding task decisions and informs supervision system agents. In this approach the agents are simpler, they have task separation but they are not fully autonomous because they have a facilitator dependency to communicate. A different approach relies on fully competition between the agents to achieve the goals, where there is no need of facilitators to do the negotiation and coordination between the agents. This approach will be addressed in future work [15].

In this architecture the FD_{An} are the fault detection agents, the FI_{An} are the fault Isolation and identification agents, the FDI_{An} are the fault detection, isolation and identification agents and FR_{An} are the fault reconfiguration agents which include the accommodation and reconfiguration tasks. The FD_{An} and FI_{An} agents only have the perception of the process without acting on it, while the FR_{An} agents have the perception and pass the reconfiguration possibility to the FR facilitator. All the agents have access to the main process data (perception) through the facilitator and using User Datagram Protocol (UDP) communications.

The FD, FI and FR facilitators are responsible for the different task agents communication and for the fault decision at the different level of the FTNC system. The FR facilitator has also the decision and actuation on the main process.

The MAS approach proposed here will bring to the FTNC system the following advantages: *Reliability*, MAS are more fault-tolerant and robust (redundant FTC agents); *Modularity and scalability*, instead of adding new capabilities to a system, agents can be added and deleted without breaking or interrupting the process; *Adaptivity*, agents have the ability to reconfigure themselves to accommodate new changes and faults; *Concurrency*, agents are capable of reasoning and performing tasks in parallel, which in turn provides more flexibility and speeds up computation and finally, *Dynamics*, agents will dynamically share their resources and decisions to solve problems.

III. Implementation of the FTNCS-MAS Platform

A new platform for a FTNC multi-agent systems must be in compliance (or at least with minimal requirements) with some standard rules for agent platforms [16]. In this case, the standard is FIPA (Foundation for Intelligent Physical Agents) [17], which define a standard model of an agent platform, as can be seen in the Fig. 5.

Several multi-agent platforms exist [16], [18], [19] to deal with different problems but none of them to deal with control systems tolerant to faults using Matlab/Simulink® environment, which is in our days the scientific bench to this kind of research. Thus, the implementation of the FTNCS-MAS platform proposed here is being developed in MATLAB®, using Simulink®, xPC Target and Distributed Computing Toolboxes. This last toolbox plus the MATLAB's Distributed Computing Engine (MDCE) ensure the services of a FIPA compliance MAS platform.

Simulink® is the environment for developing all of platform components, namely: agents, facilitators, process and supervisors (Fig. 6). Data communication between the agents is done with the xPC Target's UDP blocks. Although, it may be an unreliable way of data transfer, it is the fastest standard data transmission available. The MDCE and the respective Distributed Computing Toolbox, are an easy, fast and reliable way for remote deployment of the MAS components (agents and facilitators). The parallel job is used, thus giving a secure way for data transfer between agents, for situations that require starting, pausing, stopping and/or altering agents or facilitators. This is especially good for ensuring that a future

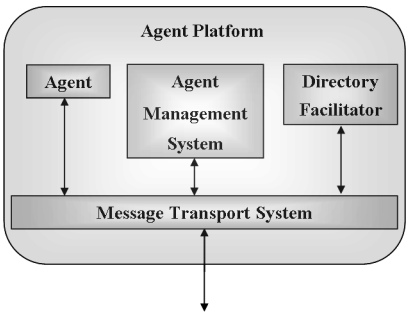

Figure 5. The standard FIPA agent platform

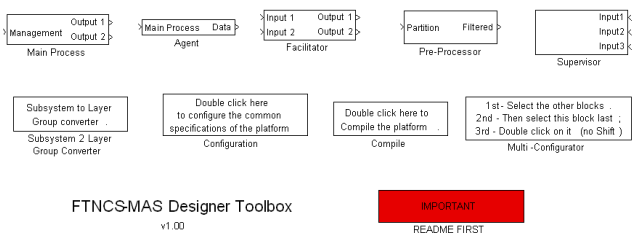

Figure 6. FTNCS-MAS designer toolbox.

implementation of an automatic platform reconfiguration can run without losing data in data transfers. An MDCE worker is assigned to each agent and to each facilitator.

The whole backbone of the platform has been developed. More specifically, a toolbox for MATLAB® has been developed, named FTNCS-MAS Designer Toolbox (Fig. 6). This allows an easy development of any FTNCS-MAS system, since it automatically sets up the communications between agents (UDP/IP) and all of the, ready to use, files for executing the platform. To the developer of a particular platform is left the task of defining specific configurations, like the computers IP addresses and additional files to use; the micro-level architecture of the agents, facilitators and supervisors have to be done manually, since there are infinite possible configurations for each. Also, for each agent and facilitator is possible to define the worker specific script to be executed. This script handles the execution of the model and can communicate with other workers. Scripts are already made and prepared for receiving commands to pause/continue, end or reconfigure each model.

Fig. 6 shows the Simulink library blocks available. At the moment, three different MAS architectures are possible to implement using this toolbox, the autonomous architecture, the federated architecture, and the so called modified-federated architecture [15]. To give a better view of what has been done, in MATLAB/Simulink®, Fig. 7 is a good example of a platform design using the MAS modified federated architecture, before compilation and deployment. All arrow lines indicate UDP connections. Notice that the process and the supervisor blocks are to be executed outside of the MDCE worker group, thus allowing better control over what is going on. More agents can be easily added, as simple as copy-paste-change. The same goes for the facilitators and supervisors.

The amount of data, to be sent via UDP, has to be pre-defined manually, since the dimensions of the outputs are dependent from the variables to be used in the FTC system. After the platform is designed, double clicking the available *Compile* block will activate the compiler function developed for this toolbox, that will: check if the design is done accordingly to the imposed limitations; afterwards will create the final models for each agent; for the process, it gives the send and receive blocks ready to be used. It also creates and copies the necessary files for the platform execution. Once compilation is complete, the platform is ready for deployment.

One additional detail about the UDP connections done by the compiler: to each send port Simulink block is associated an UDP port; when the data to be sent is less than or equal to 512 byte, data broadcast mode is used (IP address 255.255.255.255), but when it is more than 512 byte, dedicated UDP send blocks are automatically added. The toolbox is already prepared for a future possibility of an online full system reconfiguration, requiring only that the model reconfiguration algorithm is done and called from the already made scripts assigned to workers.

IV. BENCHMARK

A real benchmark, the AMIRA DTS200 Three Tank Process, is used to test the platform and FTNC system proposed. Fig. 8 presents a photo of the process and hardware used for testing the platform and FTNC systems proposed in this work.

The plant consists of three plexiglas cylinders T1, T3 and T2. These are connected serially with each other by cylindrical pipes. Located at T2 is the single so called nominal outflow valve. The outflowing liquid (usually distilled water) is collected in a reservoir, which supplies the pumps B1 and B2. Here the circle is closed. In case the liquid level of T1 or T2 exceeds this value the corresponding pump will be switched off automatically. Q_1 and Q_2 are the flow rates of the pumps B1 and B2 [20].

The model of the AMIRA DTS200 process is known and it is derived assuming nominal flow. Using the flow balance equation,

$$A\frac{dh_i}{dt} = \sum_{i,j} Q_{i,j} \tag{1}$$

or each tank T_i results,

$$A\frac{dh_1}{dt} = Q_1 - Q_{1,3} \tag{2}$$

$$A\frac{dh_2}{dt} = Q_2 - Q_{2,3} \tag{3}$$

$$A\frac{dh_3}{dt} = Q_{1,3} + Q_{2,3} - Q_{3,0} \tag{4}$$

Quantities $Q_{1,3}$, $Q_{2,3}$ and $Q_{3,0}$ can be determined using the generalized Torricelli-rule,

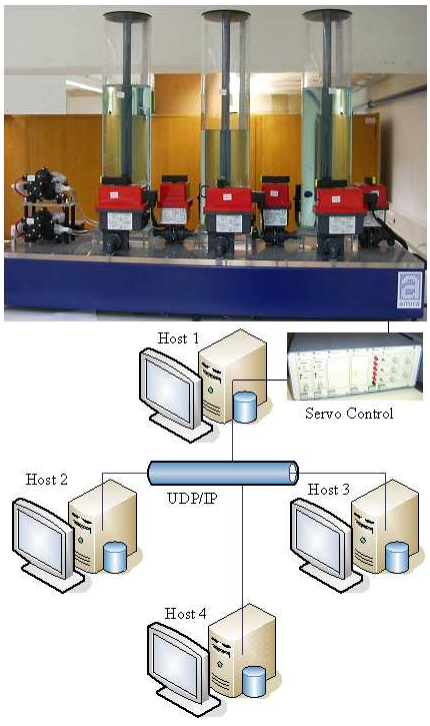

Figure 8. AMIRA DTS200 Three tank system and distributed hardware.

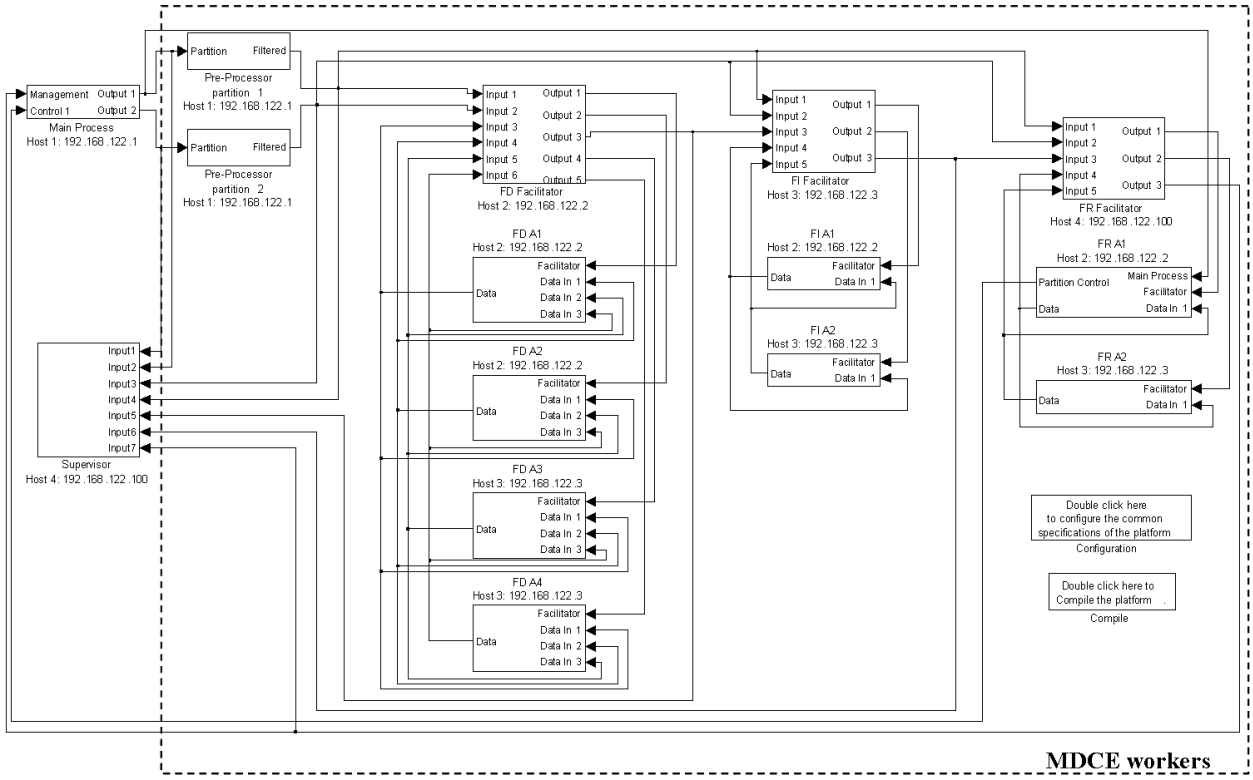

Figure 7. MAS modified federated architecture, in MATLAB/Simulink®, before compilation and deployment.

$$Q_{i,j} = \alpha_i S_n sgn(\Delta h_{i,j})(2g|\Delta h_{i,j}|)^{1/2} \quad (5)$$

where the liquid level difference $\Delta h_{i,j} := h_i - h_j$, g is the earth acceleration and the function sgn(•) denotes the sign of the argument.

The pump flow rates Q_1 and Q_2 denote the input signals, the liquid levels of T1 and T2 denote the output signals, which have to be decoupled, of the controlled plant. The necessary level measurements are carried out by piezo-resistive difference pressure sensors (PS1, PS2 and PS3). At each of it, a pressure line measures the pressure of the atmosphere [20].

A. Partition graph for the AMIRA DTS200 benchmark

The causal graph model of the three tank process and it's partition can be seen in the Fig. 9. The description of the graph and the partition methodologies are described at [21]. Here, it is presented only the new graph partition and the list of variables involved.

For the case presented here, the AMIRA DTS200 three tank process, it has been chosen the partition with 2 regions, because the two regions partition is similar to the natural partition of the process. On the other hand, the process is a simple process with a few variables in the interception border of the regions, not justifying a third partition. For more details it can be checked [21].

Fig. 9 also has represented all the faults affecting the process and which variables are affected by that faults. The faults implemented are: $F1$ - Leak in T1; $F2$ - Clog in branch with VL1; $F3$ - Clog in branch with VL2; $F4$ - Leak in T2; $F5$ - B1 pump fault; $F6$ - B2 pump fault; $F7$ - h_1 sensor fault; $F8$ - h_2 sensor fault and $F9$ - h_3 sensor fault. The chosen faults can be simulated by software or hardware implementation. All kinds of faults that can happen in a process like this, are implemented.

The process variables involved in the Graph partition are: Rh_1 - level reference for tank T1; Rh_2 - level reference for tank T2; h_1 - level in tank T1; h_2 - level in tank T2; h_3 - level in tank T3; Δp_1 - pressure difference on the pump B1; Δp_2 - pressure difference on the pump B2; Q_1 - flow through the pump B1; Q_2 - flow through the pump B2; ω_1 - "speed" of rotation of pump B1; ω_2 - "speed" of rotation of pump B2; VD_2 Output - Tank T3 discharge

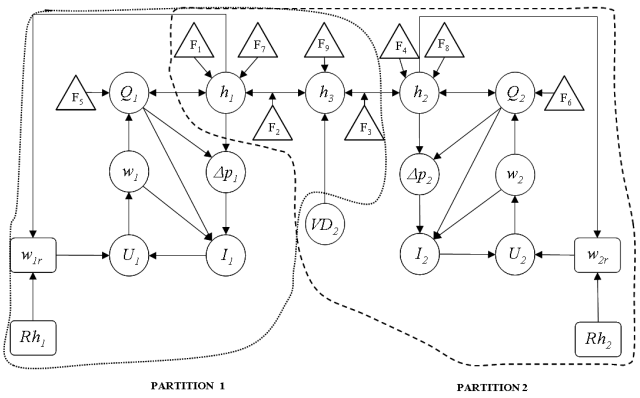

Figure 9. Graph partition of the AMIRA DTS200 Three Tank Benchmark

valve; U_1 - voltage on the motor in pump B1; I_1 - current to the motor in B1; U_2 - voltage on the motor in pump B2; I_2 - current to the motor in B2; ω_{1r} - reference for speed controller of the pump B1 and finally, ω_{2r} - reference for speed controller of the pump B2.

B. FTNCS-MAS Three tanks platform implementation

The current benchmark is the Three Tanks process presented before. Fig. 10 and Fig. 11 present the defined platform, using 4 computers (Hosts). Hosts H1, H2 and H3 have CPUs Intel Pentium IV 3.00 GHz, with 2 GB of RAM memory and the network board Realtek RTL8168/8111 PCI-E Gigabit Ethernet NIC, the host H4 has a CPU Intel Core 2 Duo 6600 2.40 GHz, 2 GB of RAM memory and also the same network board. Each computer has Windows XP®, Matlab/Simulink®, Java virtual machine and MDCE service installed. Also, only the Host 1 has the distributed computing toolbox and the Real Time Toolbox® from Humusoft®. Using this toolbox was possible to work, in real time, with the normal simulink® simulation environment.

For this benchmark, the FTNCS-MAS platform developed allows the agents to be distributed within four computers (as in Fig. 11). The backbone created has the basis for running 2 pre-processors, 4 FD agents plus their Facilitator, 2 FI agents plus their Facilitator and 2 FR agents plus their Facilitator. This sums it up to thirteen workers. Basically, the Host 2 has all the agents related with the partition 1 and the Host 3 has all the agents related with the partition 2 (see Fig. 9). For the moment, Host 1 has the main process control agent and the data pre-processor agents (A1 and A2). Host 2 has the FD A1, FD A2, FI A1, FR A1 agents and the FD facilitator. For the rest of the Hosts, please check the Fig. 11.

The communication network (e.g., Fig. 11) uses the Ethernet network (1 Gigabit speed), with a star topology and using a switch, which permits multiple simultaneous conversations between agents without traffic collisions, since the switches provide automatic network traffic isolation [22].

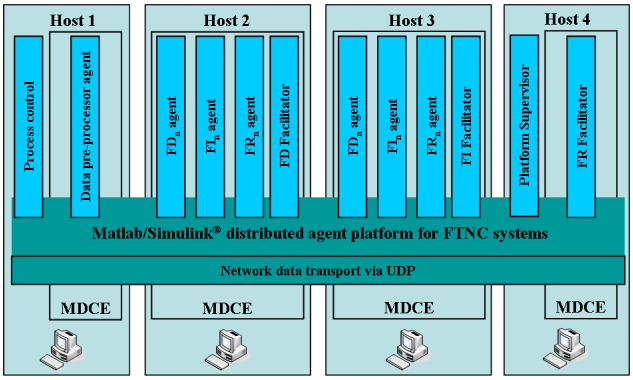

Figure 10. Three tanks FTNCS-MAS platform.

V. Platform tests and results

A. Network tests

The first network tests presented here have been done with a 100 Mbps network and using a hub for the star network. Some tests have been done to determine the lag for a full lap (process$\longrightarrow FD_{An}\longrightarrow$FD facilitator$\longrightarrow FI_{An}\longrightarrow$FI facilitator$\longrightarrow$process). One of the tests was: in real time and with host H1 controlling the process, a sine wave (with a 0.01 s sample time) was generated in the host H1 Simulink® control model and then, with the workers running agents in free infinite simulation, approximately 1.1 s lag was the result of the complete sequential loop. The time analysis shows that the main bottleneck was the UDP receive block at the process's end. Since it runs in real time, if the sample time of the UDP receive (in host H1 - process control model) is decreased to 0.001 s, the lag also comes down to one tenth of the previous, i.e. 0.11 s. In terms of Ethernet network performance, tests are already made and suggested along the years, for the 10 Mbps Ethernet network, for example tests for the ratio, number of Ethernet utilization (in MBits/s) by the number of hosts used, that tests can be easily (as the previous ones, for 100 Mbps) extended to 1 Gigabit Ethernet that is been using now, by multiplying by 10 for a 100 Mbps Fast Ethernet or multiplying by 100 for Gigabit Ethernet [22]. For the FTNC specific problem, other tests are needed.

So, Fig. 12 presents one example of the tests done with the AMIRA DTS200 Three Tank Process, but now with the 1 Gigabit speed, with the star topology and using the switch. This figure shows the level control in the tanks with the corresponding setpoints for two hosts, the figure presents not only the levels that are being controlled by the main process control agent (in the host H1), but also the levels that were transmitted by the main process control agent (in the host H1) to agents in the host H2 and come back to host H1. It can be seen that exist a lag between the data in host H1 and the same

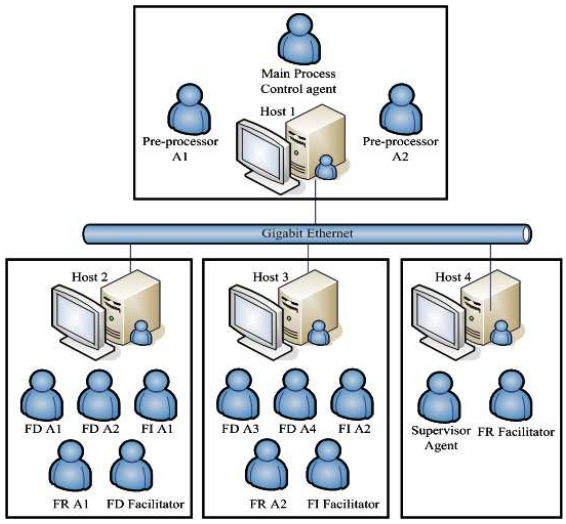

Figure 11. Agent distribution in the FTNCS-MAS platform.

data that came back from the agent in the host H2, for a sample time of 0.01 s, the lag obtained was 0.22 s (mean), but in the same way, if the sample time of the UDP received block is decreased, the lag time will also decrease, but this should be a trade-off between the needs for control and fault diagnosis problems and the possible host response due to the fact that some of the hosts could be already in over charge. For this specific problem and using the data and platform presented, the Ethernet (1 Gigabit) is far away of being saturated because the normal utilization percentage obtained with our tests has been 0.41 to 0.45% of network utilization, which means that the network saturation will never be a problem issue of this approach, with this process. The use of the Ethernet network on FTNC problems has the great advantage of being a standard TCP/UDP/IP network at a very low cost, and by simple using the existing cables and using full-duplex Ethernet switches instead of passive hubs, a control network can be build and it can guarantee that there will be no network collisions leading to deterministic. It seems perfectly realistic this approach for fault tolerant study problem based on MAS and some tests are needed to the possibility of remote control, knowing that the control definition of real-time and determinism requires that the Ethernet network must have available its bandwidth for time-critical data transfers in less than the maximum time period allowed for the control system [23].

B. Host charge

Several tests have been done to understand the limits of each host, in terms of memory, CPU, charge, page file us-age, etc, and in terms of Gigabit Ethernet network. Table I and Table II present the usage (obtained with the Windows task manager), when the platform is running, of the different hosts used in the platform depicted in Fig. 11, with the corresponding distribution shown.

Similar tables were obtained for only the windows systems running and also another test with 10 agents on the host 2, to see the corresponding increase on host charge and to verify the trade-off that should exist between the number of agents/host resources available/sample time used for control and network. From the comparison between the obtained tables it can be concluded that, for the platform and architecture chosen, the host's charge is equally divided among hosts and the hosts have yet some resources available making possible to deploy more agents for some of them. The problem is then, and depending of the micro-level of the FTC agents, if we increase the number of agents per host, the host do not has the capacity to respond in time. Further tests are needed to study the ideal maximum number of agents per host, using this platform and architecture, when dealing with problems in fault tolerant networked control.

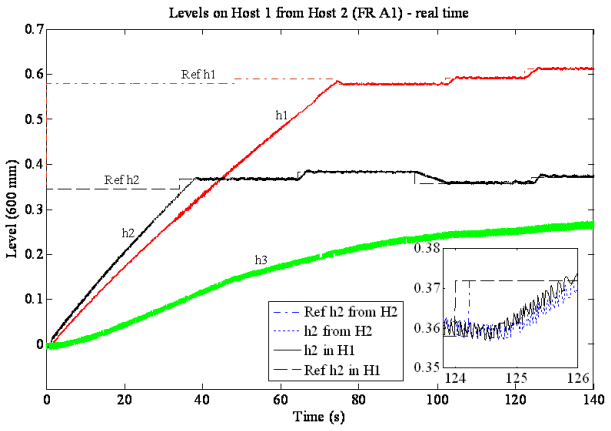

Figure 12. Level control on host H1 and the delay on the controlled signal from host H2.

VI. CONCLUSIONS

This paper proposed a new multi-agent platform and toolbox for fault tolerant control systems.

The multi-agent systems can be a valuable software engineering solution for the development of distributed FTNC computer systems in complex processes, where the complexity and the distribution of the processes asks and needs new approaches. In addition, the wide adoption of the Internet as an open environment, the increasing communication capabilities and the increasing popularity of distributed systems make the adoption of multi-agent technology a feasible and a very interesting solution.

TABLE I.
HOST USAGE.

Hosts		H 1	H 2	H 3	H 4
CPU usage (%)		100	100	100	100
Page File usage (GB)		1,17	1,19	1,19	0,95
Totals	Handles	12762	14348	14356	9897
	Threads	480	477	480	417
	Processes	33	32	33	29
Commit Charge (K)	Total	1236016	1252604	1256776	973768
	Limit	4026652	4026652	4026652	4010128
	Peak	1508316	1259764	1264396	1018436
Physical Memory (K)	Total	2088236	2088236	2088236	2071724
	Available	865308	1166160	1134848	1014972
	System Cache	333816	255800	265172	332972
Kernel Memory (K)	Total	52924	47108	47216	59236
	Paged	41608	35568	35756	42744
	Nonpaged	11316	11540	11460	16492

TABLE II.
CPU AND ETHERNET NETWORK USAGE.

Hosts	CPU usage *full working* [%]	Network usage *full working* [%]	CPU usage only monitorization *full working* [%]	Network usage only monitorization *full working* [%]	CPU usage only Agents viewer *full working* [%]	Network usage only Agents viewer *Control stopped in H1* [%]
1	100	0,44	100	0,42	100	0,02
2	100	0,43	100	0,41	100	0 - 0,01
3	100	0,41	100	0,39	100	0 - 0,01
4	100	0,45	50	0,31	8	0,02

More tests, with this new FTNCS-MAS platform, must be done to confirm these first promising results.

Future developments of the FTNCS-MAS Designer Toolbox are needed to adapt the requirements of a FTC system like this and also after the first feedback from the scientific community. New FD, FI and FR agents will be implemented for the three tank system and the FTNCS-MAS architecture, after micro-level agent architecture inserted, will be tested.

REFERENCES

[1] R. J. Patton, C. Kambhampati, A. Casavola, and G. Franzè, "Fault-tolerance as a key requirement for the control of modern systems," in *Proceedings of the 6th IFAC symposium on fault detection, supervision and safety of technical processes (SAFEPROCESS'2006)*, Beijing, china, August 2006, pp. 26–35.

[2] M. Blanke, M. Kinnaert, J. Lunze, and M. Staroswiecki, *Diagnosis and Fault-Tolerant Control*. Berlin: Springer, 2003.

[3] J. Korbicz, J. M. Kościelny, Z. Kowalczuk, and W. Cholewa, *Fault Diagnosis, Models, Artificial Intelligence, Applications*, 1st ed. Berlin, Germany: Springer, 2004.

[4] R. Isermann, *Fault-Diagnosis Systems, An Introduction from Fault Detection to Fault Tolerance*. Berlin, Germany: Springer, 2006.

[5] M. Albert, T. Längle, H. Wörn, M. Capobianco, and A. Brighenti, "Multi-agent systems for industrial diagnostics," in *Proc. of 5th IFAC SAFEPROCESS'2003*, Washington D.C., U.S.A, June 2003, pp. 483–488.

[6] B. Köppen-Seliger, T. Marcu, M. Capobianco, S. Gentil, M. Albert, and S. Latzel, "MAGIC: An integrated approach for diagnostic data management and operator support," in *Proceedings of the 5th IFAC symposium on fault detection, supervision and safety of technical processes (SAFEPROCESS'2003)*, Washington D.C., U.S.A, June 2003, pp. 471–476.

[7] M. J. G. C. Mendes, J. M. F. Calado, and J. M. G. Sá da Costa, "Fault diagnosis system based in agents," in *Proceedings of the 6th IFAC symposium on fault detection, supervision and safety of technical processes (SAFEPROCESS'2006)*, vol. 6-1, Beijing, china, August 2006, pp. 396–401.

[8] G. C. Walsh, Y. Hong, and L. G. Bushnell, "Stability analysis of networked control systems," *IEEE Transactions on Control Systems Technology*, vol. 10, no. 3, pp. 438–446, 2002.

[9] M. S. Branicky, V. Liberatore, and S. M. Phillips, "Networked control system co-simulation for co-design," in *Proceedings of the American Control Conference*, vol. 4, Denver, USA, 2003, pp. 3341–3346.

[10] M. Blanke, M. Staroswiecki, and N. E. Wu, "Concepts and methods in fault-tolerant control," in *Proceedings of the 2001 American Control Conference*, Arlington, USA, June 2001, pp. 2606–2620.

[11] M. J. G. C. Mendes, M. Kowal, J. Korbicz, and J. M. G. Sá da Costa, "Neuro-fuzzy structures in FDI systems," in *Proceedings of the 15th IFAC world congress on Automatic Control (b'02)*, vol. 15-1, Barcelona, Spain, July 2002, pp. 465–470.

[12] M. Wooldridge, *An Introduction to Multiagent Systems*. Chichester, England: John Wiley & Sons, LTD, February 2002.

[13] W. Shen, D. H. Norrie, and J. A. Barths, *Multi-Agent Systems for Concurrent Intelligent Design and Manufacturing*. London, England: Taylor and Francis, 2001.

[14] M. J. G. C. Mendes, B. M. S. Santos, and J. M. G. Sá da Costa, "Multi-agent platform for fault tolerant control systems," in *Proceedings of the 2007 IEEE International Conference on Systems, Man and Cybernetics (SMC'2007)*, Montreal, Canada, October, 7-10 2007, pp. 1321–1326.

[15] ——, "Multi-agent architectures for fault tolerant networked control systems," in *Proceedings of the IADIS International Conference on Intelligent Systems and Agents 2007*, Lisbon, Portugal, July, 3-8 2007, pp. 195–200.

[16] F. Bellifemine, G. Caire, A. Poggi, and G. Rimassa, "Jade - a white paper," *Exp in search of innovation*, vol. 3, no. 3, pp. 6–19, September 2003.

[17] FIPA, "The foundation for intelligent physical agents," *Available: http://www.fipa.org/*, 2007.

[18] P. Vrba, "Java-based agent platforms evaluation," in *Holonic and Multi-Agent Systems for Manufacturing*. Berlin Heidelberg: Springer Verlag, 2003, pp. 47–58.

[19] T. Garneau and S. Delisle, "A new general, flexible and java-based software development tool for multiagent systems," in *Proceedings of the International Conference on Information Systems and Engineering (ISE 2003)*, Montreal, Canada, July 2003, pp. 22–29.

[20] A. GmbH, *AMIRA DTS200 Laboratory Setup Three - Tank - System*, AMIRA GmbH, Duisburg, Germany, May 2002.

[21] C. D. Bocănială and J. Sá da Costa, "Causal models for distributed fault diagnosis of complex systems," in *Computational Intelligence in Fault Diagnosis*, ser. Advanced Information and Knowledge Processing, V. Palade, C. Bocănială, and L. Jain, Eds. Springer-Verlag, 2006.

[22] C. E. Spurgeon, *Ethernet: The Definitive Guide*. Beijing: O'Reilly & Associates, 2000.

[23] D. Caro, *Automation Network Selection*. USA: ISA - The Instrumentation, Systems and Automation Society, 2004.

Mário J. G. C. Mendes is currently finishing the PhD thesis in mechanical engineering at Technical University of Lisbon, Instituto Superior Técnico, Portugal. He received his MSc (2001) and Graduation (1998) degrees in mechanical engineering from the same University, in the fields of control systems and robotics and automation, respectively. He has an Adjunt Professor position at the mechanical engineering department, Instituto Superior de Engenharia de Lisboa, Polytechnic Institute of Lisbon, Portugal. His research interests include: multi-agent systems approaches in fault tolerant networked control, distributed systems, cybernetics, fault detection and isolation systems, and control of alternative energies/energetic efficiency.

Bruno M. S. Santos graduated in mechanical engineering from Technical University of Lisbon Instituto Superior Técnico, Portugal, in 2006. He received a 6 month investigation scholarship in networked fault tolerant control, under the orientation of José Sá da Costa, where he worked in the development of the FTNCS-MAS Designer Toolbox. Nowadays he works at blueCAPE Ltd, Portugal, as a R&D consultant. His research interests are focused on distributed control, modeling and simulation on multiple areas, namelly: visual servoing and robotics, metaheuristics based optimization and forest fire propagation.

José Sá da Costa graduated in mechanical engineering from Technical University of Lisbon Instituto Superior Técnico, Portugal, in 1974. He received the MSc and PhD degrees in theory and practice of automatic control in 1979 and 1982, respectively, from the University of Manchester Institute of Science and Technology. Nowadays is full professor in Systems & Control at Instituto Superior Técnico, Technical University of Lisbon, Portugal. His research interests are focused on fault-tolerant control, network control systems, active control of vibration and acoustic noise, flexible robots in surgery applications and control of ocean wave energy converters.