Proceedings of the 17th World Congress
The International Federation of Automatic Control
Seoul, Korea, July 6-11, 2008

Design of Distributed Fault Tolerant Control Systems*

Jose Sa da Costa* Mário J. G. C. Mendes**

**IST, Technical University of Lisbon, Portugal*
(e-mail: sadacosta@dem.ist.utl.pt).
***ISEL, Polytechnic Institute of Lisbon, Portugal*
(e-mail: mmendes@dem.isel.ipl.pt)

Abstract: When dealing with large-scale complex networked control systems, designing FDI/FTC systems is a very difficult task due to the large number of sensors and actuators spatially distributed and networked connected. Despite the research effort on developing FTC systems for NCS most of these developments still being designed globally leading to centralized FTC solutions inadequate to NCS or, assume the communication network and the process itself as two different entities loosing the potentiality of the integrated design. The FDI/FTC design method presented in this paper is able to use simple and verifiable principles coming mainly from a decentralized design, based on causal modelling partitioning of the NCS and distributed computing using multi-agents systems, allowing the use of well established FDI/FTC methodologies, or new ones, developed taking into account the NCS specificities. The design methodology is made easy using a FTCNS-MAS toolbox introduced in this paper.

1. INTRODUCTION

Nowadays, communication networks are adopted in many control and management systems to fulfil the information exchange and control signal transmission among system components of large-scale complex systems. These type of systems, named networked control systems (NCS), are systems whose sensors, actuators 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). Usually, NCS involve a hierarchy of local and global embedded control/diagnostic structures that makes the overall system large and complex.

Like any other system NCS are subject to faults or malfunctions or simply performance deterioration of equipments, network and 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 detected as quickly as possible and actions that stop the propagation of their effects have to be taken, i.e. it must have fault tolerance (Blank *et al.*, 2003) capabilities. These actions should be carried out by appropriate control equipment through a fault diagnosis system to detect, isolate and identify the kind of fault and its severity, and an appropriate tuning or reconfiguration of the controller, or the overall system itself to adapt to the faulty NCS condition. If these actions are successful, the system operation will be satisfied and performance fully recovered, or slightly decreased in cases of serious faults. Thus, NCS require highly sophisticated supervisory control systems with distributed fault tolerance to ensure that high performance can be achieved and maintained under adverse conditions.

The straitfoward application to NCS of the classical centralized design of Fault Tolerant Control (FTC) system results in a high dimensional problem, limits the expandability and reconfiguration capabilities of the overall system and requires high data exchange. Multi-agent technology provides a natural way to overcome such problems and to design and implement distributed intelligent FTC systems in large, open and complex environments as first proposed in Bocănială and Sa da Costa, (2004) and later followed by others (Patton *et al.* 2007).

Only recently, researchers start to address the fault detection and isolation problem and the fault tolerance control design in NCS (Ding and Zhang, 2006; Fang and Zhong, 2006; Sauter *et al.*, 2006; Vatanski *et al*., 2006). Most of these developments follow a global and centralized design, i.e. consider the system as a whole and the resulting FTC system needs to monitor all system variables to take appropriate actions. However, for large, physically distributed and heterogeneous NCS, where the communication network among the parts of the system forces a distributed vision of the overall system this approach is completely inadequate. Also, the network itself is subject to malfunctions and faults making the design of FTC systems even more difficult and making most of the times the previous methodologies inadequate to address this design. Thus, there is a pressing need to develop new approaches that intrinsically takes advantages of the characteristics of large, physically distributed and heterogeneous NCS.

Recently, (Sa da Costa and Mendes, 2006) propose a general framework for the FDI/FTC design of complex and

* This work was partially supported by the "Programa do FSE, PRODEP III, action 5.3, under EU framework III and by project POCTI/EME/59522/2004, co-sponsored by FEDER and "Programa Operacional Ciência, Tecnologia e Inovaçao 2010", FCT, Portugal.

distributed NCS, having in mind their distributed nature, using the distributed computing power given by software agents organized in society, forming a multi-agent system (MAS) spread through the NCS. The approach minimizes the number of critical communications needed among parts of the NCS to guarantee safe operation and performance even in faulting conditions. The proposed approach is an autonomous hybrid framework involving mainly decentralized topology but also having a few centralized capabilities at higher level of execution. This framework is independent of the methodologies used to tackle the different stages of FTC system and will allow the co-design of the overall controller of the NCS, namely process control, network control and fault tolerance control. In the case of distributed fault tolerant control system design this new framework resembles a shell to be filled with fault detection (FD), fault isolation (FI) and reconfiguration (FR) algorithms using both classical FTC methodologies or new ones, directly developed taking into account the specificities of the NCS under study.

The above mentioned approach gave good FDI results for the simple three tanks benchmark system (Sa da Costa and Mendes, 2006) but revealed the need to develop a computational tool to help the design of the autonomous multi-agents FTC system in a systematic way to make easy to deal with large and complex NCS. In this paper we present both the distributed fault tolerant control system design methodology and the actual version of a Matlab® Toolbox to help the design of the autonomous multi-agents FTC systems for distributed NCS.

In this paper, Section 2 is devoted to summarize the main steps of distributed fault tolerant control system design methodology adopted, emphasizing the assumed distributed and decentralized multi-agents architecture and the centralized supervision needed at management level. Section 3 introduces the toolbox to design Fault Tolerant Networked Control Systems based on Multi-Agent Systems. Section 4 shows some of the features of this toolbox by presenting performance tests regarding the hosts charges and network capability for the three tanks benchmark. Finally, Section 5 gives some conclusions and points future developments.

2. DESIGN OF DISTRIBUTED FTC

NCS are most of the time associated with high dimensionality, spatial distribution, complex and sometimes time-varying topology, nonlinear and hybrid models, and several sources of uncertainty. The proposed framework to design distributed FTC systems for large, distributed and complex NCS relies on the inherent properties of these types of systems (Sa da Costa and Mendes, 2006), i.e. it allows us to make the partitioning of the NCS in smaller subsystems to be deal with independently, reducing dimensionality and complexity, making more manageable the distributed design. However, this partition should minimize the number of critical communications needed among the subsystems of the NCS to guarantee a certain degree of autonomy, safe operation and performance even in faulting conditions, in the process itself or in the communication network.

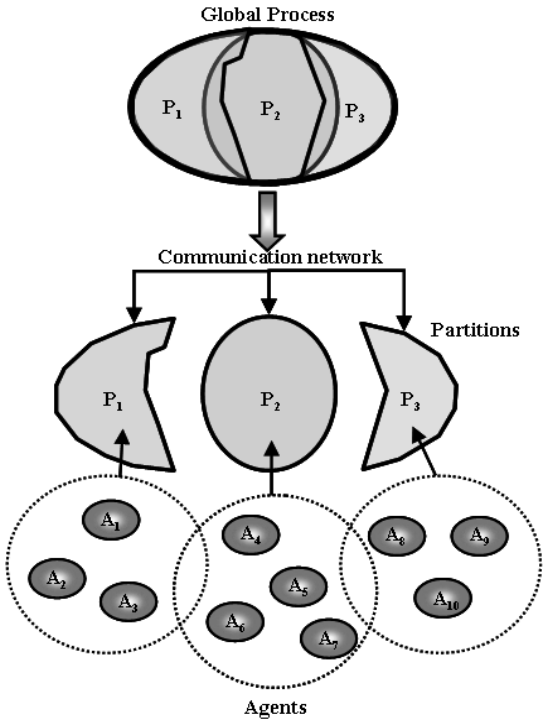

Fig. 1. Design approach to complex NCS

The adopted design approach relies on two main steps: first, to define an appropriate set of subsystems of the NCS; secondly, to assign FTC multi-agent systems to each subsystem to perform local FTC as shown in Fig. 1 (P-subsystems, A-agent). The global FTC is obtained by defining a proper communication scheme among the distributed FTC subsystems.

This approach allows a large number of degrees of freedom during the design stage, mainly dependent of the NCS considered, the detail we want to reproduce it, and the degree of redundancy we are interested to perform the FTC. The number of subsystems depends of the detail the designer consider appropriated to reproduce the NCS. Besides, since subsystems can overlap, allowing in this way a certain degree of redundancy, it is possible to consider a hierarchical set of subsystems, being the FTC performed in a top down manner, whenever necessary, without the need to perform deep analysis every time. This feature makes the approach appropriated to time varying topological changes in the NCS due to faults or plug and play of subsystems, due to operational or managerial reasons.

Redundancy can also be obtained by using different methods concurrently (performed by agents) for FD, FI, FDI and FR in each partition, or set of subsystems, or even for the all NCS, when performing FTC.

2.1 System partitioning

Bocănială and Sa da Costa (2004, 2005, 2006), proposed the use of a distributed FDI framework based on optimal partition of the directed graph (digraph) of the overall system that minimizes communication among different subsystems. This approach can also be easily extended to the FTC case.

The causal model of the NCS may be encoded as a digraph where the vertices represent the available sensors and actuators readings, and edges represent the causal links between these measurements. The complexity of the system is reflected in the complexity of the associated digraph. The partitioning procedure (*i*) considers the causal model of the system (digraph) as a map, (*ii*) partitions this map into edge disjoint subsystems separated by borders formed by vertices, and (*iii*) assigns a dedicated FTC to each subsystem. For step (*ii*), notice that each subsystem may be treated recursively in the same manner as the initial map, therefore inducing a local hierarchy of the FTC subsystems. The local expertise of the agents performing FD, FI, FDI and FR, as well as the interaction among them is used to robustly detect and isolate the faults in the subsystem and to perform control reconfiguration. The use of this distributed scheme allows maintaining the focus only on those subsystems of the map that are affected by faults. In this way, FTC of a complex NCS becomes a tractable problem.

In order to comply with the natural requirement for a FTC computational time, as small as possible, the previous partitioning is required to satisfy the following conditions: (*i*) the agents should be able to independently asses the state of the system in the assigned area, and (*ii*) the interaction between different agents should be kept as small as possible. The complexity of the interaction between two agents is given by the number of vertices located on the borders between the corresponding regions. The first condition is fulfilled by using the d-*separation* criterion to split the map in separate subsystems (Bocănială and Sa da Costa 2004). The criterion offers a parallel between the causal independency and the vertex separation in digraphs. However, the d-*separation* criterion only applies to acyclic digraphs which are not usually the case in controlled system. To solve this drawback, a new methodology that allows cyclic causal models to be transformed into acyclic models has been developed without actually losing the structural and behavioural information given by the digraph loops, (Bocănială and Sa da Costa, 2006). The second condition is fulfilled by using the multilevel hyper-graph partitioning (Karypis, 2002). The analyzed causal model is transformed into a hyper-graph so that the following equivalence holds: the causal model has a minimal number of vertices on the subsystem's borders if and only if the equivalent hyper-graph has a minimal number of hyper-edges cut by the subsystem's borders. For more details and simple examples see (Bocănială and Sa da Costa, 2006).

The effectiveness of a partition is given by the complexity reduction it offers by performing local FTC in this subsystem instead of in a larger one and the degree of autonomy, i.e. the number and degree of importance of the data required from other subsystems to perform the FTC task.

2.2 Multi-agent systems in distributed FTC

The FTC framework proposed is based in agents, or more exactly, in a society of agents usually called multi-agent systems (MAS). Several different multi-agent architectures can be found in the literature (Shen *et al.*, 2001). Several architectures have been implemented and tested. So far, the one that gives better results is the modified federated MAS architecture with facilitators (Mendes *et al*., 2007b), where one agent depends on the other, but not vice versa. The facilitators deal, respectively, with the unilateral communication between the different FTC task agents, manages 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 among the agents to achieve the goals and there is no need of negotiation and coordination among the agents. This approach will be studied in future work.

The use of MAS to implement FTC systems will bring the following advantages: *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; *Dynamics*, agents can dynamically collaborate to share their resources and solve the FTC problems, and finally, *Reliability*, MAS are more fault-tolerant and robust than traditional FTC systems since redundant FTC agents can be used in a straight forward manner.

3. DESIGN TOOLBOX

To help on the design of distributed fault tolerant control systems a new toolbox was developed, named *Fault Tolerant Networked Control Systems based on Multi-Agent Systems* (FTNCS-MAS). On this version of the toolbox, only the design of the multi-agent architectures and the platform setup and start-up are included (Sa da Costa *et al*., 2007). 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.

3.1 FTNCS-MAS Toolbox

The FTNCS-MAS toolbox for Matlab/Simulink® environment is a great help on the design and development of a new computational platform for FTC using MAS, decreasing a lot the time needed to construct a FTNCS-MAS system that must be in compliance (or at least with minimal requirements) with some standard rules for agent platforms (Bellifemine *et al.*, 2003). In this case, the standard is FIPA (Foundation for Intelligent Physical Agents) (FIPA, 2007), which define a standard model of an agent platform.

The FTNCS-MAS toolbox presented here has been developed in MATLAB®, using Simulink®, xPC Target® and Distributed Computing Toolbox®. This last toolbox plus the MATLAB's Distributed Computing Engine® (MDCE) service ensure the services of a FIPA compliance MAS platform.

The communication network (e.g., Fig. 2) uses the Ethernet network (1 Gigabit or 100 Mbps speed), with a star topology and using a switch, which permits multiple simultaneous

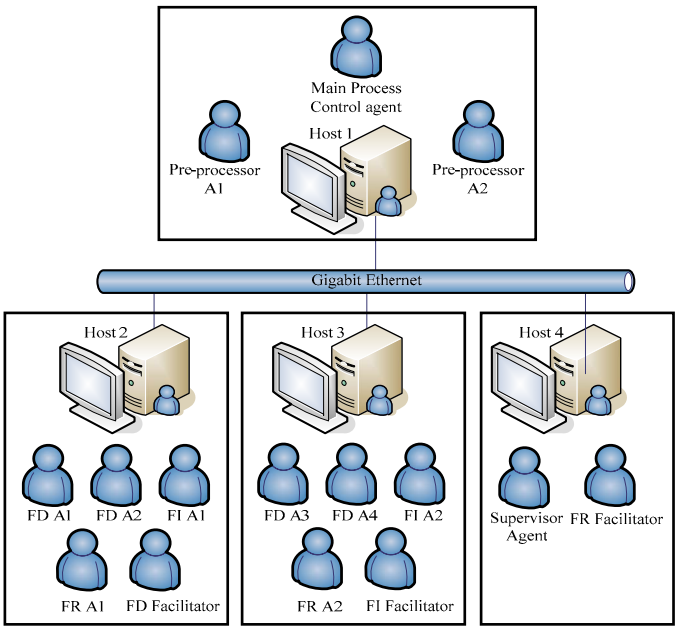

Fig. 2. Example of agent distribution in the FTC Platform

conversations between agents without traffic collisions, since the switches provide automatic network traffic isolation (Spurgeon, 2000). Data communication between the agents is done with the xPC Target's UDP blocks (UDP/IP protocol). Although, it may be an unreliable way of data transfer, it is the fastest standard data transmission protocol available on this environment. The agent deployment by host and the platform communication for turn on and start-up is done using the TCP/IP protocol. In order for an agent (and host) be able to send an Ethernet packet to another agent (and host), it must know the MAC address of the destination host, so the Address

Resolution Protocol (ARP) is also used at the start-up of the platform. Essentially, the communications between hosts make use of the ARP, TCP/IP and UDP/IP protocols.

Simulink® is the environment for developing all platform components, namely: agents, facilitators, process and supervisors. The MDCE service 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 changing agents or facilitators. This is especially good for ensuring that a future 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 amount of data, to be sent via UDP/IP, has to be predefined manually, since the dimensions of the outputs are dependent from the variables to be used in the FTC system.

This allows an easy development of any FTNCS-MAS, 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 (easily reconfigurable) 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. A library of scripts are already made and prepared for receiving commands to pause/continue, end or reconfigure each model.

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.

When the architecture and platform are designed, it is only needed to compile them. This compilation can be done with a compiler function developed on this toolbox, which 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.

The UDP/IP 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 an online full system reconfiguration, requiring only that the model reconfiguration algorithm is done and called from the already made scripts assigned to workers.

The FTNCS modified federated multi-agent architecture (Mendes *et al*, 2007b) has been implemented using this new toolbox and platform (Mendes *et al*, 2007a) to design Fault Tolerant Networked Control Systems based on Multi-Agent Systems. The next section presents the results of the performance tests executed using the agent distribution shown in the Fig. 2, when using the FTNCS modified federated multi-agent architecture.

4. PLATFORM TESTS

The AMIRA DTS200 Three Tank Process is being used to help in the development and the testing of the FTNCS-MAS Designer Toolbox and the generated platform. In the following several performance tests demonstrate the possibilities of the toolbox, showing that the chosen platform and architecture are the correct choice for this simple FTNC systems based in multi-agents systems.

4.1 Host charge

Several tests have been done to understand the limits of each host, in terms of memory, CPU, charge, page file usage, etc, and in terms of Gigabit Ethernet network.

Table 1 presents the usage (obtained with the Windows task

Table 1. Hosts usage

Hosts		H1	H2	H3	H4
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

manager), when the platform is running with the different hosts used in the platform depicted in Fig. 2, with the corresponding distribution shown.

Host 1 (H1) has the main process controller agent, and two pre-processors agents, host 2 (H2) has five agents, two fault detection agents (FD A1 and FD A2), one fault isolation agent (FI A1), one fault reconfiguration agent (FR A1) and the fault detection facilitator (FD Facilitator). Similar distribution has the host 3 and finally, the host 4 has the supervisor agent and the fault reconfiguration facilitator agent (FR facilitator). The 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.

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 agent, 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, and also, different micro-level agent architectures must be tested to see the influence on the distributed platform performance.

4.2 Network tests

The network test presented here was done with a 100 Mbps network and using a hub for the star network, some test to determine the lag for a full lap (process→FD_{An}→FD facilitator→FI_{An}→FI facilitator→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 (Spurgeon, 2000).

For the FTC specific problem, other tests are needed. Fig. 3 presents one example of the tests done with the AMIRA DTS200 Three Tank Process. This figure shows is the control of the levels in the tanks with the corresponding setpoints from 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 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 FTC 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 defini-

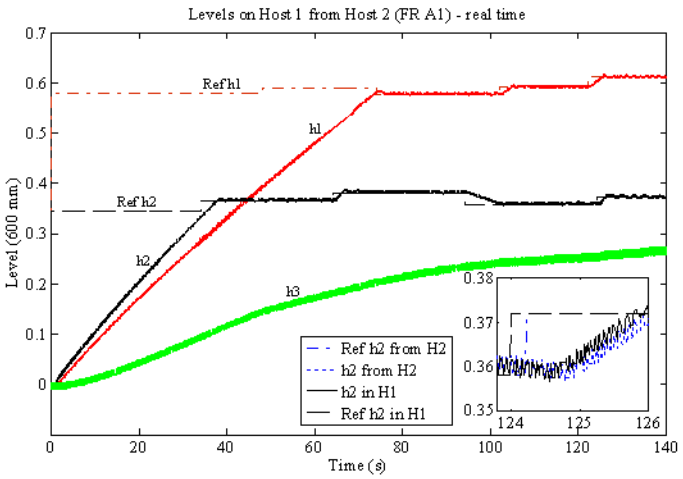

Fig. 3. Level control on host H1 and the delay on the controlled signal from host H2

-tion 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 (Caro, 2004).

5. CONCLUSIONS

In this paper, we addressed the problem of design of distributed fault tolerant control systems for large and complex NCS. The proposed approach relies on splitting the NCS in simpler but coherent set of subsystems, in terms of control and fault tolerance, and the use of the distributed capacity of hybrid multi-agent systems to easily deal with the associated fault tolerant tasks. The partition algorithm minimizes the information needed to be transmitted through the communication network to both, make as much as possible the subsystem autonomous in terms of control and fault tolerance, and to minimize the effects of faults (delays, drop of packets) in the communication network. The use of hybrid multi-agent systems allows the use of well known FDI/FTC methodologies for each subsystem, and new ones to deal with networking problems inherent to NCS. To facilitate the design of distributed fault tolerant control systems based on this approach a toolbox was introduced and an example of performance tests given for a simple case study.

Up to now, in this development, the communication network has been left out of the partition design of the NCS. Future work will focus on real-time reconfigurable partition of the NCS due to changes and faults in the communication network, using multi-criterion optimization algorithms taken into account control performance, data exchange and fault tolerance.

REFERENCES

Blanke, M., M. Kinnaert, J., Lunze and M. Starowswiecki (2003). Diagnosis and Fault-Tolerant Control. Springer.

Bocănială, C.D. and J. Sa da Costa (2004). Novel Framework for Using Causal Models in Distributed Fault Diagnosis" In: *Proc. Workshop on Advanced Control and Diagnosis, Karlsruhe, Germany,* pp. 142-147.

Bocănială, C.D. and J. Sa da Costa (2005). Novel methodology for partitioning complex systems for fault diagnosis purposes. In: *Proc. of the 16th IFAC World Congress*, Prague, Czech Republic.

Bocănială, C.D. and J. Sa da Costa (2006). Causal Models for Distributed Fault Diagnosis of Complex Systems. *Computational Intelligence in Fault Diagnosis*, Eds. V. Palade, C.D. Bocănială and L.C. Jain. Series: Advanced Information and Knowledge Processing. Springer-Verlag.

Caro, D. (2004). *Automation network selection*. The Instrumentation, Systems and Automation Society. USA.

Ding, S., P. Zhang (2006). Observer-based monitoring of distributed networked control systems. In: *Proc. IFAC Symposium SAFEPROCESS'06*, pp. 337-342, Beijing, P.R. China.

Fang, H., H. Ye, M.Zhong (2006). Fault diagnosis of networked control systems. In: *Proc. IFAC Symposium SAFEPROCESS'06*, pp. 1-12, Beijing, P.R. China.

FIPA (2007). The foundation for intelligent physical agents. *Available: http://www.fipa.org/.*

Karypis, G. (2002). *Multilevel Hypergraph Partitioning*, Technical Report 02-25, Department of Computer Science and Engineering, University of Minnesota, USA.

Mendes, M.J.G.C., B.M.S. Santos and J. Sá da Costa (2007a). Multi-agent platform for fault tolerant control systems. In: *Proc. 2007 IEEE International Conference on Systems, Man and Cybernetics*, October 7-10, Montreal, Canada, Accepted.

Mendes, M.J.G.C., B.M.S. Santos and J. Sá da Costa (2007b). Multi-agent architectures for fault tolerant networked control systems. In: *Proc. IADIS'2007 - International Conference Intelligent Systems and Agents,* pp. 195-200. IADIS Press, Lisbon, Portugal.

Patton, R. J., Kambhampati C., Casalova A., Zhnag P., Ding S. and Sauter D. (2007), A generic strategy for fault-tolerance in control systems distributed over a network, *European Journal of Control*, vol. 12, pp 1-17.

Sa da Costa, J. and M.J.G.C. Mendes (2006). FDI/FTC for complex networed control systems based on multi-agents. In: *Proc. 2nd Intern. Workshop on Networked Control Systems: Tolerant to Faults,* Rende, Italy.

Sa da Costa, J., B.M.S. Santos and M.J.G.C. Mendes (2007). Multi-agent toolbox for fault tolerant networked control systems design. In: *Proc. NeCST'07 – 3rd International Workshop on Networked Control Systems: Tolerant to Faults, Session: Autonomous Fault Tolerant Control of NCS*, University of Nancy, Nancy, France.

Sauter, D., T. Boukhobza and F. Hamelin (2006). Decentralized and autonomous design for FDI/FTC of networked control systems. In: *Proc. IFAC Symposium SAFEPROCESS'06*, pp. 163-168, Beijing, P.R. China.

Shen,W., D. H. Norrie and J.P. A. Barths (2001). *Multi-Agent Systems for Concurrent Intelligent Designand Manufacturing*. Taylor and Francis. London, England.

Spurgeon, C.E. (2000). *Ethernet: the definitive guide*. O'Reilly & Associates. Beijing.

Vatanski, N., J.P. Georges, C. Aubrun and S.L. Jämsä-Jounela (2006). Control reconfiguration in networked control system. In: *Proc. IFAC Symposium SAFEPROCESS'06*, pp. 349-354, Beijing, P.R. China.