Proceedings of the
44th IEEE Conference on Decision and Control, and
the European Control Conference 2005
Seville, Spain, December 12-15, 2005

Distributed fault tolerant control of the two-tanks system benchmark

Marta Capiluppi, Andrea Paoli

Abstract— **In this work the application of a new Fault Tolerant Control strategy to the two-tanks system is presented. This system is viewed as a distributed system and it is divided, following a functional reasoning, into partial processes supervised at different levels. The results presented in the paper have been realized in the context of the European Project IFATIS (proposal number IST-2001-32122).**

I. Introduction

A (real-time) *distributed system* consists of a set of *nodes* interconnected by a *real-time communication network*. From a functional point of view, there is practically no difference whether a task is implemented using a centralized or decentralized architecture, however a decentralized architecture has to be preferred for the implementation of hard real-time systems. An exhaustive description of distributed systems can be found in [9].

When a fault affects a system, it causes a failure, i.e. the termination or degradation of the ability of an item to perform its required function. Then a failure mode is the effect of a failure on the system, i.e. it represents a loss of a functionality. In distributed control systems every component must provide a certain function in order to achieve a global functionality. This means that a single component failure may lead to a system failure if it propagates into the system structure. Fortunately due to the distributive aspect of these systems if there exists a one-to-one mapping between functions and nodes, the cause for a malfunction can be immediately diagnosed and the faulty node isolated. For this reason the design of a Fault Tolerant Control (FTC) architecture with distributive properties is really useful for system analysis, diagnosis and reconfiguration.

According to the description given in [2] and [10], a first functional analysis is performed to divide the distributed system in partial processes, i.e. a set of physical/logical controlled systems whose aim is to achieve the main functionalities for the accomplishment of the global one. The implementation of each partial process needs allocation of resources, i.e. plant components and controller modules. This allocation is dynamic, dependent on mode and reconfiguration decisions. The hierarchical architecture for fault tolerant control of such a system consists of three modular levels: the plant level just described; the control level in which *Fault Tolerant modules*, representing a possible hierarchical structure of modules, control partial processes to achieve their function using different working modes where necessary (for example nominal working mode or reconfigured/degraded working modes); the supervision level whose aim is to monitor performances of the system and to manage physical resources.

The functionality map linked to each partial process can be highlighted by a *Functionality Tree* (see [1]). This is a graphical tool by which the main functionality of the partial process, the root of the tree, is expressed in terms of sub-functionalities (i.e. intermediate nodes of the tree) causally related to the main one. The *Fault Tree*, showing the losses of functionalities as a consequence of faulty conditions (see [1], [3]), can be interpreted as a complementary version of the functionality tree. Losses of functionalities at each level of the tree are seen as caused by losses of functionalities sitting at lower levels until the main elementary cause, given by the physical fault, is reached. The fault and functionality tree provide a useful tool for the design of both diagnostic and reconfiguration algorithms. In fact it is possible to associate to each failure mode in the fault tree one or more residual signals detecting it. The graphical description given by the functionality tree can be used to identify a hierarchy also in the reconfiguration algorithm. It is possible to identify a set of reconfiguration actions to implement the reconfiguration of the specific functionality. A possible tool to identify reconfigurable functionalities is structural analysis (see [11]). Aim of the control level supervisors (local supervisor) is to choose the reconfigurable functionality which is the closest to the estimated failure mode. According to this description, each local reconfiguration supervisor can be thought as composed by a set of Fault Detection and Isolation (FDI) units providing an estimation of the failure mode observed in a specific process/sub-process, an Event Generator (EG) which processes this information to raise requests of working mode changes and a set of Working Mode Decision Logic (WMDL) units managing them. The theoretical key tool used in the design of the WMDL unit is the supervisor theory for Discrete Event Systems (DES) (see [7]).

More details on this design procedure for FTC architecture can be found in [5], [6] and references herein.

This paper aims to apply this modular fault tolerant architecture developed by the authors within the European project IFATIS to the benchmark of the project. This is an hydraulic system composed by two tanks interconnected through some pipes, valves and pumps. As it will be explained in Section 2, the system is very simple and also control objectives are

Corresponding Author: Eng. Marta Capiluppi; mcapiluppi@deis.unibo.it.

This research has been supported by EC-Project IFATIS (*Intelligent Fault Tolerant Control in Integrated Systems*) funded by the European Commission in the IST programme 2001 of the 5th EC framework programme (IST-2001-32122) and by MIUR.

The authors are with Center for Research on Complex Automated Systems (CASY) "Giuseppe Evangelisti", DEIS, Department of Electronic, Computer Science and Systems, University of Bologna, Via Risorgimento 2, 40136 Bologna, Italy

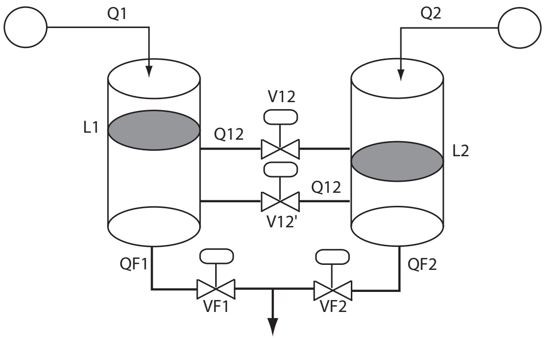


Fig. 1. Two-tanks representation.

classical; however the sharp distinction between functionalities/objectives and physical resources, as well as the presence of hardware redundancy to guarantee the existence of error containment regions, are classical key features of distributed system and make it a perfect benchmark to investigate the effectiveness of the proposed results. The focus of the paper will be not on the diagnosis and reconfiguration possibilities, but on the architecture which is possible to design for the system using the procedure presented in [5].

II. The two-tanks system

The two-tanks system (see [8]) is composed by two tanks supplied by two pumps with flow rates Q_1 and Q_2. The two tanks are connected through two redundant pipes with valves V_{12} and V'_{12}. The two pipes are placed over the lower limit level of liquid and at the same height. The output flows of the two tanks are mixed through valves V_{F1} and V_{F2}. A schematic representation of the system is shown in Fig. 1.

The system is equipped with two level sensors (L_1 and L_2) measuring liquid heights in the tanks and five flow-rates sensors measuring flows Q_1, Q_2, Q_{12}, Q_{F1} and Q_{F2}. The physical equations of the system are

$$\begin{aligned} S_1\dot{L}_1 &= -Q_{F1} - Q_{12} + Q_1 \\ S_2\dot{L}_2 &= -Q_{F2} + Q_{12} + Q_2 \end{aligned} \tag{1}$$

where

$$Q_{12} = sign(L_1 - L_2)R_{12}\sqrt{|L_1 - L_2|} \tag{2}$$

$$Q_{F1} = R_1\sqrt{L_1} \tag{3}$$

$$Q_{F2} = R_2\sqrt{L_2}\ . \tag{4}$$

In previous equations, R_{12} is the throttling[1] of valve V_{12}, R_1 is the throttling of valve V_{F1}, R_2 is the throttling of valve V_{F2} and S_1, S_2 are the sections of tanks 1 and 2 respectively. The controlled outputs are

$$\begin{aligned} y_1 &= R_1\sqrt{L_1} + R_2\sqrt{L_2} \\ y_2 &= \frac{R_1\sqrt{L_1}}{R_2\sqrt{L_2}}\ . \end{aligned} \tag{5}$$

These two outputs are regulated in order to reach two desired set points, denoted respectively as y_1^* and y_2^*, which can be considered to be proportional respectively to the desired total output flow rate (sum of output flows of tank 1 and tank 2) and to the desired ratio between flow rates from the two tanks (see equations (3) and (4)). Set points y_1^* and y_2^* can be rewritten as desired set points L_1^* and L_2^* for the measured levels L_1 and L_2. In fact from (5) it comes out that

$$\begin{aligned} L_1^* &= \left(\frac{y_1^* y_2^*}{R_1(1+y_2^*)}\right)^2 \\ L_2^* &= \left(\frac{y_1^*}{R_2(1+y_2^*)}\right)^2\ . \end{aligned} \tag{6}$$

When designing a supervisor strategy for a system it is necessary to decide the behavior of the system itself in every working condition. This is the reason why we will call working mode (WM) each behavioral condition in which the system works. It is possible to consider two nominal working conditions for the system, assuming that the throttling of valve V_{12} is constant: the first one in which the valve is closed (decoupled system), the other one in which the valve is opened (coupled system). In both situations flow-rates Q_1 and Q_2 are used as control variables. In the first situation (decoupled tanks) model (1) becomes:

$$\begin{aligned} S_1\dot{L}_1 &= -R_1\sqrt{L_1} + Q_1 \\ S_2\dot{L}_2 &= -R_2\sqrt{L_2} + Q_2\ . \end{aligned} \tag{7}$$

It is possible to achieve control objectives (5) or (6) using Q_1 and Q_2 in order to track specified references. This nominal condition can be considered as the first working mode of the system, named **WM0**. In case of open valve, the model of the system is represented by (1). The system is therefore coupled and must be controlled in a coupled way in order to achieve again

$$L_1 = L_1^* \qquad L_2 = L_2^*$$

where L_1^* and L_2^* are defined in (6). This can be considered as the second working mode of the system, named **WM0′**.

To cover some significant detection and reconfiguration possibilities for this system four realistic faults are considered:

- *Actuator fault*: pump 2 can stuck to a constant value $Q_2 = \overline{Q}_2$, namely it injects a constant flow in tank 2. This fault leads to a loss of controllability as it changes one of the structural constraints in (1).
- *Hardware fault*: valve V_{12} can stuck to a constant value $R_{12} = \overline{R}_{12}$ which can be 0 (namely the valve fails in stuck closed mode) or a constant finite positive value. As R_{12} is assumed constant even in nominal (un-faulty) operation mode, this failure corrupts only the coupled working mode for system (1)-(4) in the case the valve gets stuck closed.
- *Leakage fault*: we consider the possibility of having a hole in tank 2, i.e. there is an undesired and not measured outgoing flow from tank 2. The dynamic of

[1] V_{12} is an electromechanical valve and its throttling R_{12} is modelled as $k_1V + k_2$, where k_1, k_2 are suitably sized parameters and V is the applied electrical voltage.

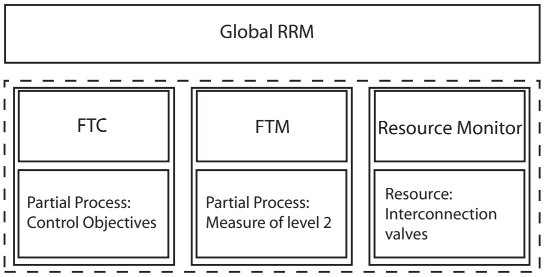

Global RRM
FTC
FTM
Resource Monitor
Partial Process: Control Objectives
Partial Process: Measure of level 2
Resource: Interconnection valves

Fig. 2. Two tanks basic decomposition.

L_2 is then corrupted by a term $\delta Q_{F2}(L_2) = h \cdot L_2$ where h is the section of the hole.

- *Level sensor fault*: the measure L_{2m} of level L_2 can be corrupted by a constant bias δL_2, i.e. $L_{2m} = L_2 + \delta L_2$.

Note that all faults are supposed to occur in tank 2, anyway analogous faults can be considered for tank 1.

III. FTC DESIGN

Due to the considered faults a first functional analysis of the system leads to a first decomposition in two partial processes. The first one aimed to achieve control objectives; the second one aimed to estimate in a reliable way the liquid levels which are the feedback values for control. Since faults are considered just in tank 2, these partial processes reflects into a Fault Tolerant Control (FTC) module and a Fault Tolerant Measurement (FTM) module managing fault tolerant measure of liquid level in tank 2. This situation is represented in Fig. 2. The interconnection valves are considered Hardware equipment supervised by a dedicated Resource Monitor achieving FDI (see [10]).

A. Functionality and Fault Tree Analysis

The functionality tree of the FTC module is represented in Fig. 3(a) where the main functionality is the tracking of both objectives y_1^* and y_2^*, i.e. to keep the sum and the mix of the flows to desired set points. This is obtained if both levels L_1 and L_2 track the respective set-points (6), i.e. if we have the possibility to maintain the levels at the desired values. Level 2 is maintained at the desired value only if the outgoing flow from tank 2 has the expected value Q_{F2}^* and the input flow of tank 2 has the desired (controlled) value Q_2^*. This last situation happens when the actuator (pump) works properly and the regulator (control law) is still valid. Only sub-nodes from level 2 are generated because the faults are supposed to happen only in tank 2, but the same structure can be generated from level 1.

From the analysis of the tree in Fig. 3(a) it follows that the main functionality is reconfigurable. In fact the loss of controllability properties leads to the impossibility of tracking simultaneously both objectives, but it is always possible to compute degraded references trajectories, consistent with the faulty conditions, and to enforce relaxed tracking objectives. Moreover the capability of enforcing a desired level $L_2^\star$ can be reconfigured in presence of a leakage inducing a loss of the functionality linked to the expected value Q_{F2}^*. In fact a leakage can be compensated by suitably switching to a robust controller for the input Q_2 thus preserving the desired reference $L_2^\star$.

With the same reasoning the functionality tree of the FTM module can be designed and analyzed to find out the reconfigurable functionalities. This tree is represented in Fig. 3(b).

The fault trees are designed starting from the functionality trees. For the FTC module the fault tree is represented in Fig. 4(a). This figure shows that when we cannot achieve both local objective L_1^* or L_2^*, then global objectives y_1^* and y_2^* cannot be achieved at the same time. Achieving local objective L_2^* is possible only if both desired control input flow $Q_2^\star$ and expected output flow rate $Q_{F2}^\star$ are achieved. The output flow rate can be different from the expected one only if a leakage fault has occurred. The desired control input flow is not obtained anymore if the actuator or the control law fails. Here the only considered fault on the actuator is the stuck of pump $\overline{Q}_2$. The fault trees will be used in the sequel to establish detectable losses of functionalities.

B. Residual Generation and failure modes

Now, starting from equations (1) it is possible to generate residual signals partially based on structural considerations (see [3]).

A first residual signal is

$$r_1 = |\hat{L}_2 - L_{2m}| \,. \tag{8}$$

where $\hat{L}_2$ is an estimate of L_2 given by the observer

$$S_2\dot{\hat{L}}_2 = -R_2\sqrt{\hat{L}_2} + Q_{12m} + Q_{2m} + G(\hat{L}_2 - L_{2m}) \,,$$

based on the reliable measures of system variables and where G is a suitable tuned negative gain.

Another possible observer is partially based on measures and partially on controlled input flow value Q_2^*:

$$S_2\dot{\hat{L}}_2' = -Q_{F2m} + Q_{12m} + Q_2^* + G\,'(\hat{L}_2' - L_{2m}) \,,$$

where $G\,'$ is a suitable tuned negative gain. Then the residual signal is

$$r_2 = |\hat{L}_2' - L_{2m}| \,. \tag{9}$$

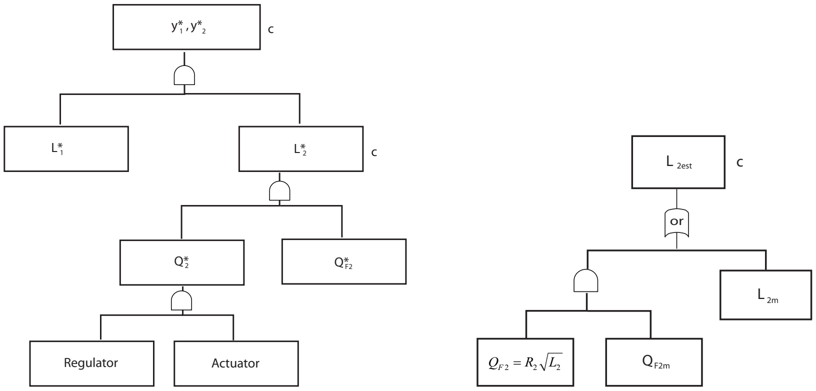
Anyway, due to the structural properties of the system, constraints linking throttling of valve V_{12} to other variables depend on the state of the valve itself. If the valve is open, we have the flow Q_{12} between the two tanks. Thus, since L_1 and Q_{12} are measurable, from eq. (2) it is also possible to estimate L_2 as

$$\hat{L}_2'' = L_{1m} - \left(\frac{|Q_{12m}|}{R_{12m}}\right)^2$$

and to generate a residual signal $r_3(t)$ as

$$r_3 = |L_{2m} - \hat{L}_2''| \,. \tag{10}$$

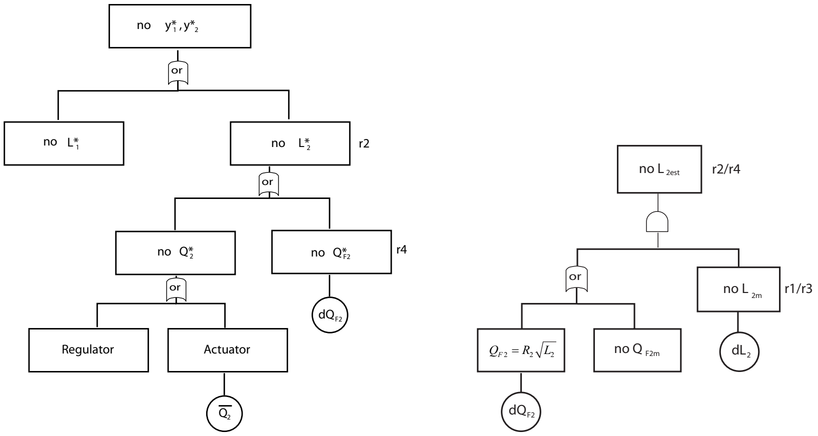
Finally consider equation (4). It states that in case of nominal conditions, the outgoing flow from tank 2 depends only on level L_2 and on throttling R_2 of the outgoing valve.



(a) Functionality tree for the FTC module.

(b) Functionality tree for the FTM module.

Fig. 3. Functionality tree analysis for the two tanks system: the reconfigurable functionalities are labelled with c.



(a) Fault tree for the FTC module.

(b) Fault tree for the FTM module.

Fig. 4. Fault tree analysis for the two tanks system.

In case of leakage in tank 2 this relation does not hold anymore, since there is another outgoing flow. This means that equation (4) becomes:

$$R_2\sqrt{L_2} = Q_{F2} + \delta Q_{F2}\ .$$

For this reason signal

$$r_4 = R_2\sqrt{L_{2m}} - Q_{F2m} \qquad (11)$$

is equal to zero in nominal conditions while it is different from zero in case of leakage in tank two or in case of a misreading of the level sensor.

From the above discussion, residual matrix in Table I comes out. It is possible to verify just by inspection of Table I that, with the set of residual signals generated, the system is fully detectable and isolable with respect to the considered set of faults.

In Fig. 4(a) and 4(b) the previously defined residual signals are associated to some failure modes, because they are used to reveal losses of functionalities. In particular residual r_2 is associated to the failure mode *loss of desired liquid level* L^*_2, in fact this signal is sensitive to both faults $\overline{Q}_2$ and δQ_{F2} (see Table I). Following the same reasoning, residual r_4 is associated to the *loss of expected output flow* in Fig. 4(a). In the FTM fault tree, residuals r_1/r_3 are associated to the *loss of measure*, while residuals r_2/r_4 are associated to the *loss of estimate* itself. From residual matrix in Tab. I it is possible to see how this new analysis allows to obtain a modular FDI engine, in fact the FDI unit of the supervisor is only present at the levels of the fault tree to which a residual signal is associated.

	$\overline{Q}_2$	$\overline{R}_{12}$	δQ_{F2}	δL_2
r_1	0	0	0	1
r_2	1	0	1	1
r_3	0	1	0	1
r_4	0	0	1	1

TABLE I

RESIDUAL MATRIX FOR THE TWO-TANKS SYSTEM

Note that diagnosis of fault $\overline{R}_{12}$ is not considered in this analysis because managed by the resource monitor.

C. Reconfiguration Actions

Here some possible reconfigurations for the reconfigurable functionalities described above are presented. For the sake of clarity these reconfigurations are directly related with physical faults. Like the nominal conditions, even these reconfigurations lead to different working modes for the system, because they lead to different behaviors of the system itself.

Consider first the fault on pump 2: pump 2 is stuck to a constant value such that $Q_2 = \overline{Q}_2$. If valve V_{12} is closed, the system is represented by (7) (the two tanks are decoupled). In this case we lose a control variable ($Q_2 = \overline{Q}_2 = \text{const.}$), and hence one degree of freedom. Since it is not possible to achieve both control objectives in (5) anymore, one of the two is chosen and the trajectory of L_1 is reconfigured in order to achieve this objective. Because the incoming flow in tank 2 is constant, the level in this tank will stabilize to a constant level $\overline{L_2}$ which can be measured. Hence

- if the sum functionality must be preserved then we compute the new trajectory L_1^* as
$$L_1^* = \left(\frac{y_1^* - R_2\sqrt{\overline{L_2}}}{R_1} \right)^2 .$$
We will call this working mode **WM1**.
- if the ratio functionality must be preserved then we compute the new trajectory L_1^* as
$$L_1^* = \left(\frac{R_2}{R_1} y_2^* \sqrt{\overline{L_2}} \right)^2 .$$
We will call this case **WM2**.

In both cases we do not need an estimation of $\overline{Q_2}$ because we use the measure of L_2. Note that in these two working modes the control law does not need to be changed.

If valve V_{12} is open, the system is represented by (1). We must again decide which of the two control objectives we want to satisfy, because we have only one control input. Suppose we want to track:

$$y_1^* = R_1\sqrt{L_1^*} + R_2\sqrt{L_2^*}.$$

We can control the error

$$e = R_1\sqrt{L_1} + R_2\sqrt{L_2} - y_1^* \; .$$

We will call this working mode **WM3**. Similarly, by forcing objective y_2^* instead of y_1^* and following the same procedure as above, we can define a working mode **WM4** in which the objective y_2^* is forced through a controller on the error

$$e = \frac{R_1\sqrt{L_1}}{R_2\sqrt{L_2}} - y_2^* \; .$$

Note that even the implementation of these control strategies does not require the estimation of $\overline{Q_2}$.

A leakage fault on tank 2 implies a parametric change in the model of the system and can be tolerated locally using a robust control law. In this sense it is possible to define a reconfigured working mode **WM5** in which the local control law on Q_2 is robust to this fault.

The sensor fault can be managed by the FTM which estimates the interested variable, hence no explicit reconfiguration is needed regarding fault δL_2, because we suppose to always have a reliable estimation $\hat{L}_2$ of the value of L_2. Anyway for consistency with the nomenclature used this situation is considered as a reconfiguration called **WM6**.

Consider now the fault on valve V_{12}. If the valve is stuck closed we can switch to valve V_{12}^I and use R_{12}^I, due to the parallel hardware redundance. This working mode is denoted with **WM7**.

D. Working Mode Decision Logic

To design the supervisor decision logic units sitting at different levels of the architecture, we make use of the Discrete Event Systems (DES) supervision theory. The supervisor must inhibit some moves in the whole system model in order to force the desired reconfiguration strategy. To this purpose, we use the working modes previously described to build specification automata representing the supervisors decision logic units. The lower level reconfigurable functionality is the loss of expected output flow-rate. The automaton representing the specifications for this supervisor must act in the following way: when the leakage fault occurs the system goes in its new working mode **WM5**. The more complex automaton representing specification for the reconfiguration of top level functionality has the following policy: after the occurrence of fault $\overline{Q}_2$, an activation event is sent from the event generator to the supervisor which decides the new working mode (**WM1** to **WM4**). In the same way it is possible to design the local supervisor for the FTM module.

The whole system is globally supervised to manage the hardware reconfiguration: when the interconnection valve V_{12} is stuck closed, the supervisor switches to the other redundant valve V_{12}' and the system moves in working mode **WM7**. The global supervisor manages also the control objective we want to force ($y_1^\star$ or $y_2^\star$) in case of fault $\overline{Q}_2$.

For sake of brevity the details and realization of the Decision Logic are omitted but can be found in [4].

E. Implementation

Concluding the discussion developed throughout the paper, in Fig. 5 the whole supervision architecture for the system is presented. At the lowest level the physical system is represented: it is divided in partial processes (PP) according to the functional point of view described in this work. The three FT modules depicted in Fig. 2 are here reported in their internal decomposition in terms of hierarchy of Fault Detection units (FD), Fault Isolation units (FI), Event Generator units (EG) and Working Mode Decision Logic units (WMDL). Residual signals (r_i) are generated by FD units and elaborated by FI units which give as output fault estimation signals. These signals, where needed, are used by EG units to rise requests (a_i) of changes of working modes. Finally WMDL units, elaborating fault estimation signals and requests of changes of working modes, decide the new working mode (**WMi**) to be imposed to the system. The physical resource (interconnection valve) is supervised

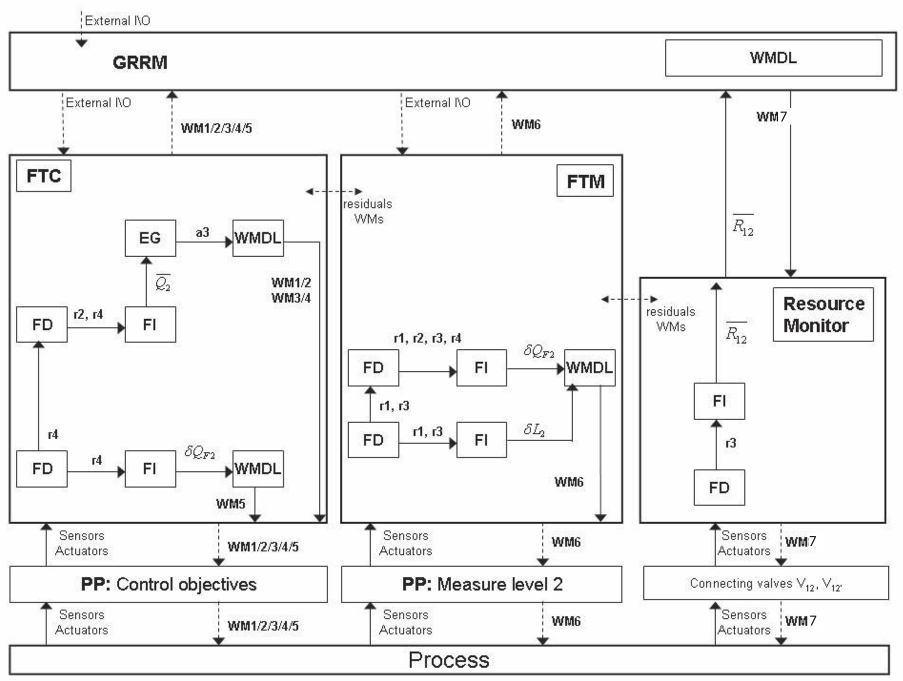


Fig. 5. Final decomposition for the distributed fault tolerant architecture of the two tanks system.

just by a resource monitor here represented by a FDI unit. The high level supervisor (GRRM) is also reported. Its task is to optimally manage some changes of working modes according to external inputs (e.g. changes of priority in specifications), fault estimation signals, requests by lower level WMDL units and physical resources availability. The reader can easily see in the figure the information flow going bottom up to refine the diagnosis inference and going top down to impose the new working mode to the system.

To enlighten the effectiveness of the proposed solution, some significative experiments have been performed on the real plant, located at the *Centre de Recherche en Automatique de Nancy* (CRAN). The interested reader can find the experimental results in [4].

IV. Conclusions

In this paper the application of a design procedure for a modular/hierarchical fault tolerant control architecture to a benchmark system has been illustrated. Given a possible fault scenario the fault tolerant control architecture for this system has been designed following a functional reasoning.

It is important to stress that the main effort in this work is to study a design procedure for fault tolerant control of distributed system which is algorithmic and therefore easily automatizable. For this reason future works will include the development of a design software that implement in an automatic way all the steps of the procedure.

A first attempt in this direction has been performed in the context of the IFATIS project by the software house TNI Valiosys which realized a design and verification tool for the two tanks system.

Acknowledgments

Authors wish to thank Prof. Dominique Sauter and his staff at *Centre de Recherche en Automatique de Nancy* (CRAN) for their precious help in implementing the architecture on the real plant. Authors wish also to thank Eng. Reine Gros from TNI Valiosys for software implementation.

References

[1] J. Andrews and T. Moss, *Reliability and Risk Assessment*. Professional Engineering Publishing, 2002.

[2] L. E. Bahir, R. Gros, M. Kinnaert, C. Parloir, and J.Yamé, “Final report on WP2,” IFATIS deliverable D2-5, January 2003, http://ifatis.uni-duisburg.de/.

[3] M. Blanke, M. Kinnaert, M. Staroswiecki, and J. Lunze, *Diagnosis and fault-tolerant control*. Springer-Verlag, 2003.

[4] C. Bonivento, M. Capiluppi, L. Marconi, and A. Paoli, “Designing multilevel fault tolerant control for distributed systems: a functional approach,” CASY-DEIS, University of Bologna, Tech. Rep., 2005, contact person Marta Capiluppi.

[5] ——, “An integrated design approach to multilevel fault tolerant control of distributed systems,” *In proceedings of XVI IFAC World Congress, Prague, Czech Republic*, 2005.

[6] M. Capiluppi and A. Paoli, “Hierarchical design of distributed fault tolerant control systems,” *In proceedings of XIII MED, Limassol, Cyprus*, 2005.

[7] C. Cassandras and S. Lafortune, *Introduction to discrete event systems*. Kluwer Academic Publisher, 1999.

[8] F. Hamelin, H. Jamouli, and D. Sauter, “The two tanks pilot plant,” IFATIS report IFAN014R01, February 2004, http://ifatis.uni-duisburg.de/.

[9] H. Koepetz, *Real-time systems: design principles for distributed embedded applications*, ser. Real-time systems. London: Kluwer academic publishers, 1997.

[10] U. Maier and M. Colnaric, “Some basic ideas for intelligent control systems design,” *Proceedings of the XV IFAC World Congress, Barcelona, Spain*, 2002.

[11] M. Staroswiecki, S. Attouche, and M. Assas, “A graphic approach for reconfigurability analysis,” *10th Int. Workshop on principle of diagnosis, Loch Awe*, 1999.