A Hierarchical Structure for Control Of Discrete Events Systems and Monitoring Of Process Failures

Eric Zamai, Audine Chaillet-Subias, Michel Combacau and Agnan de Bonneval
Laboratoire d'Analyse et d'Architecture des Systemes
Centre National de la Recherche Scientifique
7, Avenue du Colonel Roche,
31077 Toulouse
FRANCE
zamai@laas.fr chaillet@laas. fr combacau@laas. fr agnan@laas. fr

Abstract: In a first part this work presents a hierarchical and modular architecture for real-time control and monitoring of Discrete Events Systems like Flexible Manufacturing Systems (F.M.S.). The first characteristic of this architecture is based on a splitting up procedure to build a hierarchical structure allowing to manage the complexity of the process. The second characteristic concerns the organisation of this structure to drive the process. On the one hand, the control system, the monitoring system and the model exhibiting the process capabilities are distributed in each node of this hierarchical structure. So each of these nodes is a "control and monitoring module". On the other hand, a further process model more complete than the one included in each control and monitoring module is put near the hierarchical structure. The last contribution of this work introduces an inter level communication mechanism suitable for real-time control and monitoring requirements. Adding a supervisor distributed in each control and monitoring module allows to manage the complex data flow taking part in the inter levels communication. In a second part, an example is considered to illustrate different aspects of the approach previously described.

Keywords: discrete events systems, control, monitoring functions, process model, communication mechanism, operating modes.

Eric Zamai was born in France in 1971. He received the DEA degree in Electrical Engineering from the University of Toulouse, France, in 1994. Currently he pursues the Ph. D degree in Electrical Engineering at the Laboratoire d'Analyse et d'Architecture des Systemes (LAAS).

Audine Chaillet-Subias was born in France in 1968. She received the Ph. D degree in Electrical Engineering from the University of Toulouse, France, in 1995. Since 1995 she has been Assistant Professor at the Institut National des Sciences Appliquées de Toulouse. Her research interests include Monitoring of DES, Information Systems and Diagnosis.

Michel Combacau was born in France in 1959. He received the Ph. D degree in Electrical Engineering from the University of Toulouse, France, in 1991. He is currently Assistant Professor at the University of Toulouse and carries out his research at the Laboratoire d'Analyse et d'Architecture des Systemes (LAAS). His research interests include Monitoring of Discrete Events Systems.

Agnan de Bonneval was born in France in 1963. He received the Ph. D degree in Electrical Engineering from the University of Toulouse, France, in 1993. He is currently Assistant Professor at the Institut Universitaire de Technologie des Pays de l'Adour, Mont de Marsan, France.

1. Introduction

This paper presents an approach to real-time control and monitoring of Complex Discrete Events Systems. This approach has been developed at the Laboratoire d'Analyse et d'Architecture des Systemes [5] and is structured on three aspects: the complexity of the physical process to be controlled, the complexity of the controller and monitor, and so, the complexity of the communication between the physical process and the control and monitoring system. To solve the first problem we have split up the whole process into simpler sub-problems (a hierarchical structure). For the second problem, a generic control and monitoring module with process models is proposed for each node of the hierarchical structure. Then, a communication mechanism is established; it allows levels to communicate by taking control and monitoring requirements into account. Before concluding, an illustration of this approach is given.

2. A Hierarchical and Modular Structure

In F.M.S., systems to control and monitor are becoming more and more complex. To cope with this complexity, a structured approach should be considered . With this aim, we have elaborated a hierarchical and modular structure of decision [11], [10]. This structure is based on different abstraction levels for the real-time levels of control (coordination and local control

levels). Its elaboration is performed in a bottom up fashion, from the process components to the highest level. At this level, all the process capabilities are expressed. Each level provides to the upper level, services that can be performed by the lower levels. At each of these levels, several independent modules can be specified, grouping services performed by the lower levels. So, two modules of the same level are quite autonomous. Using such modules allows to implement the control and monitoring system in a generic form (see Figure 1). Indeed, each module is made up of six functions that are managed by a supervisor (see Figure 2). The main characteristics of these components can be described as follows [5], [6]:

- The control function contains a model of the activities sequence to be executed on the manufactured product including time aspects of the part routing (control net). This sequence satisfies the constraints imposed in a process model (named "reference model").
- The detection function must detect all the unexpected process evolutions. This function is integrated into the control function to avoid an exhaustive list of unusual situations.
- The diagnosis function must find the cause of the detected failure. This function is built around a rule-based system.
- The decision function must modify the control model, trigger the emergency procedures or a specific recovery sequence, or propagate the failure treatment to the upper level if necessary.
- The operator interface allows the human operator to dialogue with the automatic system if needed by the monitoring system [8].
- The emergency function must apply some procedures, having priority in front of the control sequence, which may be triggered to protect the human operators and/or the process components.

Then, using the services implemented by the lower level, each module is able to execute requests received from the upper level (control requests, diagnosis refinements, recovery requests, emergency requests). Moreover, with the monitoring functions the module is able to perform its own particular treatment for unexpected failures. These treatments consist in linking the monitoring functions as best as possible. These links are adapted according to the failure and to the context in which the failure occurred. But, these monitoring functions need a lot of various information about the process [9], [7]. In our approach, this information is distributed in two models described in the following Section.

3. Process Models

3.1 The Reference Model

This process model (named "reference model", see Figure 2) exhibits the process capabilities. Moreover, this model expresses how a required activity must be executed according to the process behaviors allowed by the designer. These behaviors are modeled through the activity concept including the constraints (shared resource, sequencing, collision avoidance...) to be observed when an activity is triggered whatever the control sequence. These constraints are modeled using Petri Nets with Objects [12]. The state of this model evolves at each sensor value occurrence from the process and upon each request from the operating part. So, at any time, this model tries to represent as best as possible the accurate situation of the process . Of course, the abstraction level of this model depends on the considered node in the hierarchical structure.

3.2 The Information System

An information system joins this control and monitoring structure. This information system includes a process representation as accurate as possible [3]. This process model is richer and more detailed than the one included in the reference model of the control system. This representation delineates the main entities of the process and the links between these entities. It includes time notion to represent dates, durations, etc. This representation also supports abstraction mechanisms (aggregation and generalization) to manage complex representations. Moreover, this process model is based on derived data i.e. data which are computed using the information already existing in the model. Finally, a history mechanism, based on versioning, is added to the model to keep a track of the data evolution. The aim of this information system is to feed the different monitoring functions with the information they need. The information system also supports several procedures allowing the exploitation of

the process model it includes (update, retrieval, research,...) [3]. These applications are defined in a generic way and are used according to the current control. The structure of the process model is directly linked to the process

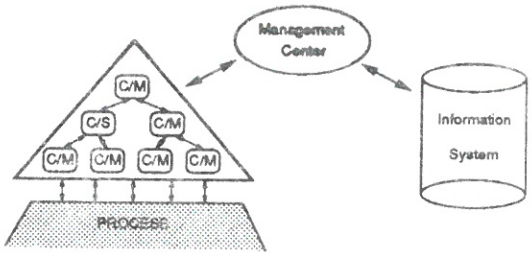

Figure 1. Proposed Control and Monitoring Architecture

composition. Its elaboration results from an inventory of the process elements. To produce the conceptual schema of the information system the data model used is an Extended Entity Relationship Model [4]. The choice of such kind of a model results, among other things, from the wish to have a model which stresses on adaptability rather than on rigidity.

A "management center" (see Figure 1) allows the exploitation of the information included in the information system [2]. This center plays the role of a conductor and manages the operation of the overall structure. It ensures all the interactions between the control and monitoring structure, and the information system. It has an external and global vision of the structure which allows it to select the procedures to trigger and the treatments to execute (update, retrieval, research, path-determination,...).

3.3 Use of the Process Models

During normal operation, the "reference model" and the control model which include the operating sequence imposed by the manufacturing process are strongly synchronized and evolve simultaneously (note that this control sequence is a particular linking of the activities expressed in the "reference model"). Before sending a request to the process, the process model is consulted by the control function. According to the process state, the request is authorized (normal operation) or not (detection of an erroneous control). If the expected report occurs before the due date, the control and the process model evolve simultaneously and the process model included in the information system, is updated via the management center. Else, a process failure is detected. Moreover, if the expected report corresponds to a possible evolution in the "reference model", this model evolves to represent the accurate state of the process. Then, a symptom of the failure is sent to the diagnosis function. This function must find the origin of the failure according to the transmitted symptom. Briefly, when a diagnosis process is triggered the reasoning based on rules is first invoked in order to elaborate a diagnosis hypothesis. If no hypothesis is generated the failure processing is propagated to the upper level, where a similar treatment is performed. Else, the deep data included in the information system are then used to precise this result. This step corresponds to a diagnosis refinement step [3]. The diagnosis refinement step first supposes that the decision function asks the management center to execute a "research operation". The complete description of this operation can be found in [2]. Briefly this research consists in selecting candidate modules. A candidate module is an entity of the information system including data which may concern the failure origin. The research is based on the locality or adjacency principle. The algorithm is modeled after an in- depth graph analysis using a recursive technique. When a candidate is generated the management center sends back a report to the decision function. This report includes several parameters namely the identification number of the control and monitoring module (target module) corresponding to the candidate. Then the decision function must trigger a diagnosis process in the target module. This supposes that the management center determines the path to follow in the hierarchical structure to reach the target (path-determination operation). Indeed, each module of the hierarchy has a limited vision of the structure [3]. Then, in the target module a diagnosis process is triggered and the result is passed to the local decision function in order to correct the consequences of the failure.

4. Inter Levels Communication Principle

The inter levels communication in such a hierarchical and modular control and monitoring structure is more complex than a simple remote/call procedure. The information exchanged between levels of the hierarchy consists not only of control requests and execution report but also of emergency requests, diagnosis requests, messages pointing out that a controlled system returns to its optimal running,

unexpected sensors values when the process evolves in a manual mode, messages pointing out that the process does not evolve as scheduled during an emergency shutdown, etc.

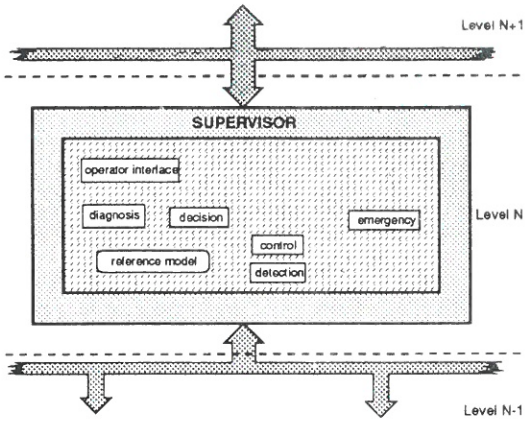

Figure 2. Module of the Hierarchical Structure

In fact, this set of information can be classified in two sub-categories: the **requests flow** and the **constraints flow** (messages arriving from the lower levels of the hierarchy).

To manage this data flow, a **supervisor** is integrated into each module of the hierarchy (see Figure 2). This supervisor has to direct each of these messages to the suitable control or monitoring function (already described in the previous parts). Each of these functions is able to treat its own category of message.

Now, let us analyze the principle used to direct these two kinds of messages.

4.1 Requests Flow

In this category, a control request (received during normal operation) must be treated by the control function, a diagnosis request (received during a high level failure processing) must be treated by the diagnosis function, and at any time an emergency request by the emergency function. In fact, the module which sends one of these kinds of requests writes into a "target" attribute, the name of the target function that corresponds to the class of request. Then, the supervisor of the module receiving this request has just to direct the message according to the value of this "target" attribute.

4.2 Constraints Flow

For the messages arriving from the lower levels of the hierarchy the principle is more complex. Indeed, these messages are uncertain and it is impossible to foresee all their occurrences. For example, an error message indicating that the request cannot be executed must be directed to the detection function whereas a message pointing out that the local control system returns to its optimal operation (a recovery sequence ends) must be treated by the decision function. These two messages must be differentiated to direct them to the suitable function. The solution of this problem depends on the context in which the incoming message is received. For instance, after a failure, the following messages (erroneous for normal operation) must be considered as the consequence of the failure already detected. For this reason, we propose to define the context in which a message is received considering the executed activity and the three states of the process: the process can be in a running, or in a shutdown or finally in an emergency situation. In the first case, all the incoming messages must be considered as execution reports and directed to the detection function. In the second case, no message has to arrive from the process. However, if it is the case, they are directed to the decision function able to take this new unexpected constraint into account. Finally, the process can be in an emergency situation in which the incoming messages must be directed to the emergency function. This one allows for instance to avoid the serial alarms triggering. Indeed, these alarms are considered as normal in such a situation.

operating modes / functions	Ready	Optimal Production	Emergency Action	Forced Work	ShutDown without Emergency	Recovery
detection		√		√		√
decision	√				√	
emergency			√			

Figure 3. Table of the Target Functions for An Activity

Moreover, as our approach is based on a hierarchical structure of control and monitoring, we have extended the process state concept to the operating modes. In fact, each of the three process states is refined into several operating modes as described by the GEMMA [1] for the control system. Of course, we have adapted the definition of these modes so as to take the monitoring requirements into account. In our

approach, we need the following operating modes:

- *Ready (to start):* the associated activity ready to be executed,
- *ShutDown (without emergency):* th activity is suspended for repair, slow down running, etc. The system waits for a return in the "Optimal Production" mode of this activity,
- *Optimal (production):* the activity is executed as scheduled and the constraints imposed by the module of the upper level are respected,
- *Forced (work):* the activity must be continued whereas a received message has pointed out a system failure. In this case, the execution of the activity is not optimal,
- *Recovery:* the system is returning in a usual situation for the activity during which the failure has been detected,
- *Emergency (action):* the monitoring system must launch an emergency sequence after the occurrence of a particular failure.

Then, a monitoring function is associated with each operating mode. This can be schematized using a decision table (see Figure 3) in which the operating modes are represented at the top of this table and the associated monitoring function at the row entries of this table. When a message arrives from the lower level of the hierarchy, it is directed to the suitable function corresponding to the operating mode of the activity concerned by this message. This is made by the supervisor (see Figure 2) using the decision table of Figure 3. After having directed the transmitted message, the monitoring system must apply the optimal monitoring loop offered by the current operating mode ("detection, diagnosis, recovery" for an unexpected report in optimal operation mode, "detection, emergency shutdown, diagnosis, recovery" in emergency mode, etc.). In this way, the monitoring processing is always adapted to the actual context of the failure and so, the efficiency of the monitoring system is increased.

5. Example

Now, let us consider an example to illustrate our approach. The system to drive is a transportation system including the possibility of sorting the carried parts according to their types. It is made up of a conveyor system and a handling robot (see Figure 4).

The operation of the system must be the following :

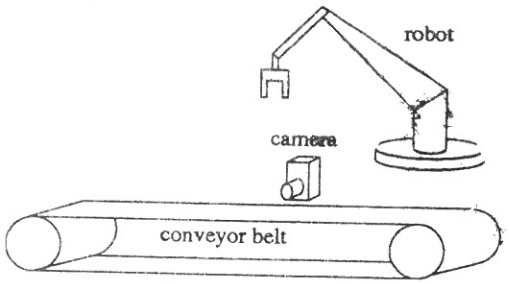

Figure 4. A Transportation System

- the conveyor belt runs in order to bring a part in front of a camera,
- when a part comes in front of the camera, the conveyor belt must stop and the type of the part is identified,
- in this example, the robot must pick up parts of type 1 and put them in a local buffer (of unlimited capacity),
- the conveyor belt is started again in order to bring a new part or to evacuate a part of type 2 (not picked up by the robot).

5.1 The Hierarchical and Modular Structure

According to § 2. we have elaborated the hierarchical and modular structure corresponding to this example (see Figure 5).

This structure is made up of three levels:

- the local level with a node for the Computer Numerical Control (CNC) of the robot, another for the camera, and finally a node for the conveyor belt,
- the upper level integrates two nodes, one for the robot and one for the conveyor system,
- the coordination level that represents among other things all the control capabilities of the considered process.

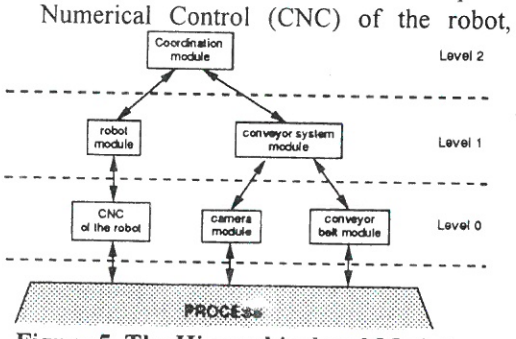

Figure 5. The Hierarchical and Modular Structure of the Transportation System

These control capabilities are, for instance, to bring a part in a specific place or to pick up a part in the buffer.

5.2 The Nodes of This Structure

For each of these nodes, we have built a control and monitoring module which integrates a supervisor and its control and monitoring functions (control, detection, diagnosis, decision, operator interface and emergency functions).

For the moment, the supervisor is provided with the mechanism allowing to analyze and direct each request, and with a model of a decision table (see Figure 3) based on the operating mode concept (see § 4.2) to direct them to the suitable control or monitoring function.

5.3 The Process Models

5.3.1 The Reference Models

Each module of the hierarchy must be provided with its own reference model. All the models we need are designed according to § 3 and represented in Figures 6, 7 and 8. For example, the usual capabilities of a robot are to "Grasp" or to "Put" a part from the conveyor belt to the buffer. These two capabilities or activities must be executed successively; after having grasped the part, the robot must put it. This sequencing constraint is represented in the reference model using Petri Nets formalism (see Figure 7).

5.3.2 The Information System

Finally, we have built the database of the information system. This database results from the process elements inventory. For the sake of simplicity we have limited the information modeled in the database. We present now the main entities of this process model:

- *a robot* entity which has some properties: a model name [Model], an identification number [Nber], a condition [Cond] which is "available" or "working". The robot is qualified to process specific parts and so a relationship "rp" specifies the part which is worked by the robot. The composition of the robot is described by means of an aggregation mechanism, namely the robot is composed of the entities *grip, handler*, and *base,*
- *a conveyor system* entity which has a condition of use [Cond], and an identification number [Nber]. It works on parts and the relationship "csp" indicates the part it processes. The conveyor system is constituted by a *conveyor belt* and a *camera* related by the relationship "supports/ supported by",
- an entity *part* which must be processed by the transportation system. Each part is described by means of a state [State] ("ready", "working" "worked"), an identification number [Nber] and the type [Type] ("1" "2").

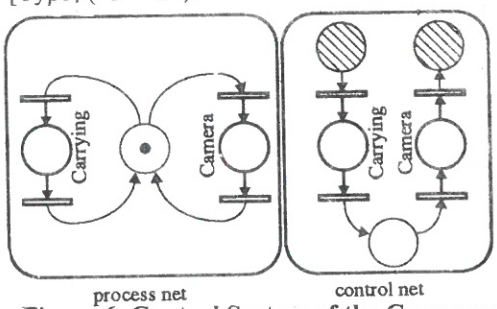

Figure 6. Control System of the Conveyor System

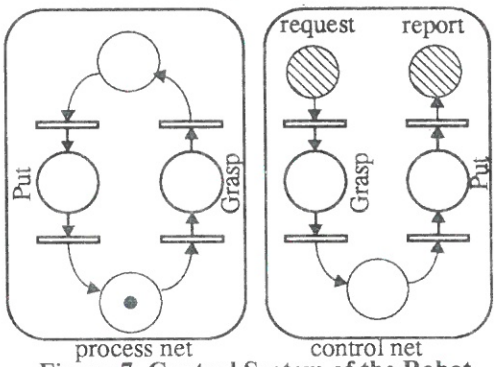

Figure 7. Control System of the Robot

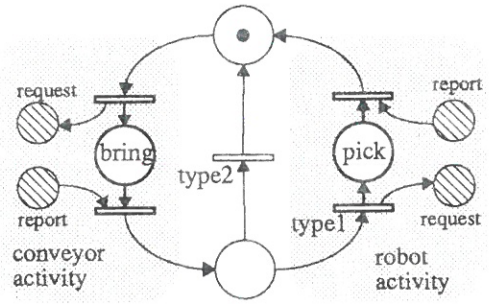

Figure 8. Control System of the Coordination Module

an entity *activity* with a name [Name], an identification number [Nber], a state [State] ("execution", "non-execution"), a starting date [Sdate] and a duration [Duration]. The relationships "ar" and "acs" relate each activity to the transportation system

elements. We have described the fact that an activity of the conveyor system is an activity by a generalization mechanism (relation *is- a*). Moreover, we have considered that an activity can be composed of components of other activities by the use of the relation "component/assemblies". For instance the activity *Bring* is composed of a conveyor belt activity *(Carrying)* and a camera activity *(Identification).*

In Figure 9 we can find a part of the conceptual scheme of the transportation system using an Extended Entity Relationship Model. The presentation of this data model is beyond the scope of this paper, nevertheless we briefly present here the formalism used: entities are represented by rectangular boxes. An undirected arc drawn from one entity to a circular box denotes an entity property. The name of the

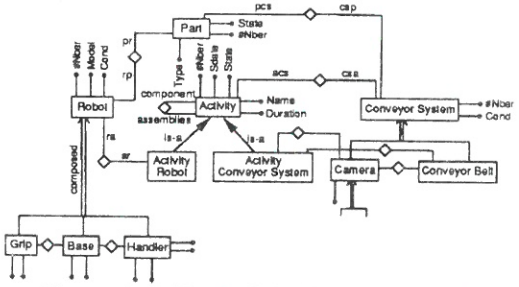

Figure 9. A Part of the Transportation System Conceptual Scheme

property is displayed next to the arc. The relations between entities have several representations with regard to their type, i.e. relationship or abstraction mechanism. A single line with diamond between entities denotes a relationship. Connection names are attached to these lines. Directed arrows between entities model an abstraction relation. The type of abstraction mechanism, namely aggregation or generalization, is indicated by a label on the arrow.

5.4 A Simulation

Now, the architecture to drive this process is established. So, let us examine a specific running of this system. Let us suppose that a malfunction of the camera occurs. After a part of type 2 had been evacuated using a *Bring* sequence, another part was detected and identified as type 1 by the camera, whereas there was actually no part. This failure is not immediately detected because the execution report "there is a type 1 part in front of the camera" corresponds to an expected report for the control function. So, the control function of the coordination module sends a *Pick* request to the robot (see Figure 8). This request is constituted of some attributes, and particularly of the target function attribute and the target control and monitoring module attribute. In this case, the target function is the name of the function which must receive this request i.e. the control function and the target module is the robot module. Then, the supervisor of the robot module receives this request. So, it reads the target function attribute of the request and directs it to the name of this target function: the control function.

In the robot module, the receipt of the *Pick* request induces the control sequence described in the control net (see Figure 7). The execution of this control sequence amounts to sending a *Grasp* request with its attributes to the CNC of the robot. The CNC executes the requested service and sends an execution report to the supervisor of the robot module. This message is considered as an execution report in this module because the mode associated with the *Grasp* activity is the Optimal Production mode. Then according to the decision table (Figure 3) this message is directed to the detection function. This execution report renders the fact that the activity *Grasp* has not been achieved as scheduled (the sensors of the gripper are not engaged during the *Grasp* activity). So, a failure is detected during this activity. The diagnosis function is then triggered in the robot module. Whatever the analysis of the diagnosis function at this level, the causes of the failure cannot be found (the robot is not directly responsible for the failure). So, the local decision function of the module can only decide to reset the robot (the mode of the *Grasp* activity evolves in the "recovery" mode), and sends back a report to propagate the failure processing.

At the upper level (coordination), this message is At the upper level (coordination), this message is directed by the local supervisor to the detection functic.. (the *Pick* activity was in an Optimal Production mode) and then a diagnosis process is triggered. This diagnosis checks that all the conditions are satisfied, then the *Pick* activity starts. In this order, the diagnosis process will interact with the information system to retrieve the information on date of the failure, using the history mechanism (see § 3.2). At the end of this analysis, the *Pick* activity is declared faultless. The diagnosis function must then analyze the execution conditions of the *Bring* activity to find the cause of the detected failure. The result of

the diagnosis function points out this activity to be responsible for the failure. This diagnosis result is then passed to the local decision function. According to the abstraction level of the coordination module, this result lacks precision. So the decision function decides to refine this diagnosis result by means of the deep data stored in the information system. The mode of the *Pick* activity evolves in a "Shutdown without Emergency" mode waiting for more details about the failure.

The research operation in the information system manipulates different versions to retrieve the relevant information existing before the failure occurred. This research first consists in examining the *Bring* entity. Next it consists in searching the activities which compose the *Bring* entity using the aggregation relation. The activity *Carrying* is analyzed and then using the relationship "acs" the conveyor belt is studied. We suppose that this part of the process is faultless, no candidate is generated; under these conditions the research must be continued in another part of the process. So, the *Camera* entity, to which the conveyor belt is related by means of the relationship "supports/supported by", is analyzed (we have not detailed the *Camera* entity in the Figure i.e. its composition, its properties,...). The research points out that the *Identification* activity offered by the *Camera* entity is suspect, and the *Camera* entity is declared a candidate. The management center sends back this result to the decision function of the coordination level. It also determines the path in the control and monitoring hierarchy from the coordination module to the Camera module, and indicates it to the detection function. This function, according to the path previously determined , sends a request for diagnosis refinement to the conveyor system node. This request includes the target function attribute i.e. diagnostic, and the list of the modules which compose the path to follow (a conveyor system and a camera), and the activity (*Identification*) which must be examined in the target control and monitoring module (the camera module).

In the conveyor system node, the supervisor receiving this request extracts the control and monitoring module id (*Camera*). This id does not match, the conveyor system supervisor ignores the request and transmits it to its subordinate (the Camera node).

In the *Camera* control and monitoring module, the local supervisor receives the request for diagnosis refinement. As the id of this local supervisor belongs to the target module, it directs the request to the diagnosis function of the module. Then the diagnosis process analyzes the *Identification* activity. Using its diagnosis rules and/or a human operator [2] the diagnosis process concludes that the camera is faulty. This conclusion is passed to the local decision function which decides to launch a recovery sequence to correct the failure consequences. This last aspect is currently under study.

6. Conclusion and Future Work

These works present an approach to Control and Monitoring of Discrete Events Systems. The need for process models has been pointed out and the problem of inter levels communication has been stressed. At present, the mechanisms allowing the exploitation of the two process models (reference model and information system) for the monitoring functions use are defined [5], [2]. The supply of the communication mechanism between levels (based on the operating modes concept), not only adapted to the control requirements but also to the monitoring ones, gives us a new insight into the monitoring of the process failures. Our future research will focus on the modeling problem of the operating modes. To achieve this goal, we have to define the main properties of the model according to the constraints imposed by the operating modes, the changes between these modes, history... Then, we have to look for the model, among the available ones, corresponding to our requirements. Moreover, we will focus on the interaction between modes that are not independent. It is the case, for instance, when two activities share a resource. Indeed, if the resource fails during one of the activities, this may imply a change of mode for the other activity.

REFERENCES

1. **LE GEMMA, Guide d'Etude des Modes de Marches et d'Arrets**. Adepa, 17 rue Perier, 92120 Montrouge, 1981.
2. CHAILLET, A., **Approche multi modeles pour la commande et la surveillance en temps réel des systemes a evenements discrets**, Ph.D Thesis, Université Paul Sabatier, December 1995.
3. CHAILLET, A. and COURVOISIER, M., **An Information System for Control and**

Monitoring Purposes in F.M.S., IECON'94 Twentieth Annual Conference of the IEEE Industrial Electronics Society, Bologna, Italy, September 1994.

4. CHEN, P., **The Entity-Relationship Model Toward a Unified View of Data**, ACM TRANSACTIONS ON DATABASE SYSTEM, Vol. 1, 1976, pp. 9-36.
5. COMBACAU, M., **Commande et surveillance des systemes a événements discrets complexes: application aux ateliers flexibles**, Ph.D Thesis, Université Paul Sabatier, December 1991.
6. COURVOISIER, M., COMBACAU, M. and DE BONNEVAL, A., **Control and Monitoring of Large Discrete Event Systems: A Generic Approach**, ISIE'93 IEEE International Symposium on Industrial Electronics, Budapest, Hungary, June 1993.
7. CRUETTE, D., **Application a la conception et la validation hiérarchisée de la commande de cellules flexibles de production dans l'industrie manufacturiere**, Ph.D Thesis, Université des Sciences et Techniques de Lille, Flandres Artois, December 1990.
8. DE BONNEVAL, A., **Mécanismes de reprise dans les systemes de commande a événements discrets**, Ph.D Thesis, Université Paul Sabatier, September 1993.
9. HOLLOWAY, L.E. and KROGH, B.H., **Fault Detection and Diagnosis in Manufacturing Systems: A Behavioral Model Approach**, Second International Conference on Computer Integrated Manufacturing, May 1990.
10. HUVENOIT, B., **De la conception a l'implémentation de la commande modulaire et hiérarchisée des systemes flexibles de production manufacturiere**, Ph.D Thesis, Université de Lille 1, October 1994.
11. JONES, A., **A Multi-layer/Multi-level Control Architecture for Computer Integrated Manufacturing Systems**, IEEE Int. Symp. on Intelligent Control, Albany, NY, September 1989.
12. SIBERTIN-BLANC, C., **Le modele de données objet comme formalisme de modélisation de base de données**, MDB, AFCET, Vol.9, June 1988.