DESIGN OF RECONFIGURABLE AND COLLABORATIVE CONTROL SYSTEM FOR PRODUCTIVE SYSTEMS

Robson Marinho da Silva, rmsilva@uesc.br
Universidade Estadual de Santa Cruz, Ilhéus, BA, Brazil

Fabricio Junqueira, fabri@usp.br
Diolino José dos Santos Filho, diolino.santos@poli.usp.br
Paulo Eigi Miyagi, pemiyagi@usp.br
Escola Politécnica da Universidade de São Paulo, São Paulo, SP, Brazil

Abstract. *The design of ACSs (Automation and Control Systems) is specific for each application, such as CNC machine, robot, machine controlled by PLC (Programmable Logic Controller), process controlled by SCADA (Supervisory Control and Data Acquisition) and equipment with FPGAs (Field Programmable Gate Arrays) for customization of their behavior. These technologies have different domains, i.e., the control solution can be implemented by hardware and software in different architectures and heterogeneous specifications. Although several engineering solutions facilitate the development for each application, the integrated design of ACS is still considered a complex task, because it is necessary to master different domains and, consequently, it involves a large number of specialists to understand the whole system. Moreover, in practice, most of ACSs are designed in a relatively short time due the competition of companies. ACSs must ensure the fulfillment of the current requirements in PSs (Productive Systems), such as flexibility, distributed architecture, and agility in response to changes imposed by the market or the inevitable occurrence of faults. In this context, one solution is a combination of techniques such as HCS (Holonic Control System) and AFTCS (Active Fault-Tolerant Control System), which allows the reuse of models, reconfigurability, autonomy, cooperation, and learning ability. Therefore, this paper proposes a procedure based on AHCS (Active Holonic Control System) concept for integrated design of the entire lifecycle of ACS: from requirement specifications to operation and maintenance, to ensure greater flexibility, efficiency and robustness of the PSs. The procedure combines bottom-up and top-down approaches using Petri net technique and its extensions - PFS (Production Flow Schema), through dynamic modeling of the system in E-PN (extended Petri Net) and the gradual refinement based on PFS. An example of application in FMS (Flexible Manufacturing System), considered a representative class of PS, is used to demonstrate the advantages of the proposal.*

Keywords*: system modeling, reconfiguration, holon, Petri net, component-based system, manufacturing system.*

1. INTRODUCTION

The crescent competitiveness and the need for efficiency triggered great changes on PSs (Productive Systems) requiring more flexibility under different demands, such as production volume, type of product and nature of resources involved. This evolution forced new solutions in mechatronic technology for product/service transformation or execution. However, to ensure that a PS meets its purpose, exploring the available technology, it is necessary to update their ACSs (Automation and Control Systems) considering requirements of integration, flexibility and agility. ACSs are composed of various sub-systems (that can be physically installed at different geographical locations), in which the productive tasks are divided according to the required functionality and processing capacity of the equipment. On the other hand, to assure that a PS does not suffer interruption due to faults, an AFTCS (Active Fault-Tolerant Control System) mechanism must also be considered (Silva *et al*., 2012a, 2012b).

ACSs' projects depend on specific application systems, such as CNC machine, robot, PLC (Programmable Logic Controller) controlled machine, SCADA (Supervisory Control and Data Acquisition) controlled process and equipment with FPGAs (Field Programmable Gate Arrays). These applications have different domains, i.e., the control solution can be implemented by hardware and software in different architectures and heterogeneous specifications (Artist, 2006). Although several engineering solutions facilitate the development for each case, it is still necessary to master different domains and, consequently, it involves a large number of specialists to understand the whole system. The sub-systems perform multiple and simultaneous processes with a relatively great number of variables, different type of maintenance teams, several equipment and automation levels, which makes the supervision and control of the systems global behavior complex. Moreover, most of these systems are designed in a relatively short time due to the competition of companies, but despite this it must ensure the fulfillment of the current requirements in PSs.

Thus, the development of ACS' for PS must attend the following: (i) system specification must be done in an easily understandable level of abstraction, which should describe the aim in each level of control, not only achieve it. That is, it is necessary to adopt systematic techniques that formalize the structure and behavior of these systems, to simplify its understanding and synthesis; (ii) combination of different techniques for application domain and specific approaches must be managed to overcome the constraints of each one, and allow the analysis, modeling and implementation of an integrated system; and, (iii) specification of system architecture and their components as a whole should not depend on specific languages of manufacturer of dedicated systems equipment.

Based on the productivity aspect of any man-made systems and their intrinsic feature of faults occurrence, the design of new ACS should also consider the reuse of models (Tommila *et al*. 2005; Woll, 2007) and AFTCS mechanisms (Zhang & Jiang, 2008). Then, the design of ACS should consider the use of "components" (a type of control unit or block that encapsulate hardware, software and data structures and algorithms) allowing definition of a common language and reuse of models. Each "component" should be autonomous with some level of knowledge, intelligence and ability to achieve the planned aim, but without a global view of the system which results from the interactions between them. AFTCS mechanisms involve fault detection, study of its effects, identification of their causes and finally, reconfiguration of the system that is done by relocating and choosing alternative paths between productive processes. In case of fault occurrence, the strategy is to recover the functionality of the system (also called regeneration) or maintain critical operations so that parts affected by the fault are disabled without affecting other parts of the system (also called degeneration) (Silva *et al*., 2011a).

In this context, the integration of MAS (Multi-Agent System) and HS (Holonic System) techniques with mechatronic technology, called holonic control system (HCS), is considered a trend for the intelligent automation of PSs (Schoop *et al.*, 2002; Sousa *et al.;* 2004; Colombo *et al.*,2001; Silva *et al.*, 2011a). The aim is to explore MAS and HS concepts, such as autonomy, reactivity, proactivity, cooperation, social capacity (i.e., consideration of the human interaction on processes), and learning resources; and to take advantage from the complementary features in the implementation of HSs by means of the MASs.

However, most ACSs do not adopt HCS and AFTCS mechanisms. In fact, the amount of material published about modeling of processes that consider the use of these techniques is relatively little (Schoop *et al*., 2002; Zhang & Jiang, 2008). Therefore, this paper introduces a procedure based on AHCS (Active Holonic Control System) for integrated design of the entire lifecycle of ACS: from requirement specifications to machine operation and maintenance, to assure flexibility, efficiency and robustness in the PSs. AHCS combines bottom-up and top-down approaches using PN (Petri Net) technique and its extensions -- PFS (Production Flow Schema), through dynamic modeling of the system in E-PN (extended Petri Net) and the gradual refinement approach associated to PFS. FMSs (Flexible Manufacturing Systems) are considered a representative class of PS and an example is used to demonstrate the advantages of this proposal.

2. PETRI NET (PN)

Considering also previous works (Silva *et al*., 2012a, 2012b, 2011a, 2011b) in the area of PSs (Productive Systems), we adopt in this paper the approach of PSs as a class of DES (Discrete Event System) (Reisig, 1985), i.e., Petri net (PN) and its extensions can be used for description of the system behavior (characterized by productive processes). If compared to other description techniques of DES, PN has at least an equivalent modeling power and it also has the characteristic and advantage of relatively easy system visualization (David & Alla, 1994, Hasegawa *et al*., 1999, Reisig, 1985). This tool allows a graphical and mathematical description of the system. PN provides: (i) the possibility for dynamic representation of the system and its structure in many levels of abstraction; (ii) a representation of the process with synchronism, concurrence, causality, conflict, sharing of resources and normal and abnormal situations in ACS (Automation and Control Systems) of PSs, and (iii) a mathematical support useful for performing formal tests on the dynamic properties of the system. This is especially useful in applications in which fault-tolerant control systems are essential (Riascos & Miyagi, 2010).

Some authors define a homogeneous PN model which includes a single formalism to describe the overall system. Other authors use different formalism for each part of the system. The former is formally more elegant but presents difficulties for practical cases, because the modelers are forced to adopt a single viewpoint for all parts of the system. The latter is derived from the heterogeneity of real systems and the different viewpoints of the designers for each part. As this work considers practical systems including abnormal situations the second approach is adopted, but in order to avoid the need for specialists in a great number of formalisms we considered only two PNs based systems.

To effectively model the dynamic behavior, a class of PNs based on the place/transition PN is adopted, called extended Petri net (E-PN), to which timed transitions (terms related to PNs are presented in Arial), inhibitor arcs and enabling arcs (David & Alla, 1994) were added.

To construct these models, a method that applies a derivation of channel/agent PN called PFS (Production Flow Schema) (Miyagi, 1996, Hasegawa *et al*., 1999) is used. The PFS is a technique developed to systematize and facilitate the modeling of PSs. Modeling of the system starts in a high level of abstraction, then successive refinements are applied and the model is more detailed at each level. The objective is to clearly represent the functionality of the structure of each part involved in the execution of activities and the flow of operations in the productive processes. The system's dynamic models are generated by means of E-PNs. Thus, the procedure combines the bottom-up and top-down approaches of the stepwise refinement associated to PFS.

Concerning the modeling of faults in discrete event systems, there are studies to represent the detection and diagnosis of these faults in DES. Riascos & Miyagi (2010), for example, show that it is possible to develop models in

PNs through the characterization of patterns and to detect faults based on sensors signals processing. Sampath *et al*. (1996) present a procedure for DES modeling based on models for fault diagnosis and Zhang & Jiang (2008) present a bibliographical review of existing approaches to fault detection and diagnosis, while fault-tolerant control systems in a general framework of AFTCSs are considered and classified according to different criteria, such as design methodologies and applications.

3. HOLONIC CONTROL SYSTEM (HCS)

Koestler (1969) presented the definition of holon and holarchy, a hierarchy of self-organized holons, which behave as autonomous wholes in supra-ordination to their parts, dependent parts in subordination to wholes/sub-wholes on higher levels, and in coordination/synchronization with their local environment. HMS-Consortium (Christensen, 2000) worked on the application of Koestler's concepts to propose a new generation of PSs (Productive Systems) and their controls, providing the definition of a more specific and accurate terminology and showing optimal adaptation of these concepts to many traditional productive activities. According to HMS-Consortium, a holon in PSs consists of production equipment capable of performing productive operations, and an associated intelligent component.

An agent is considered to be a software entity with enough intelligence capable of autonomous control actions in a given environment, and of cooperation relationships by participating in association agreements with other entities in order to attain its designed objectives. An agent should be able to act without the direct intervention of humans or other agents, and should have control over its own actions and internal states (Jennings & Wooldridge, 1998). An agent approach seems very suitable for control and supervision of mechatronic devices of an intelligent system (Colombo *et al*., 2001). A multi-agent-based software platform (i.e., a platform composed by two or more agents) can execute distributed intelligent supervisory control functions with communication, cooperation and synchronization capabilities, among others, that can cover the behavior specifications of the PS components and also the functional specification to be abided by the system.

A holon is a special case of agent: an autonomous and flexible entity which is capable to act in its environment (Wooldridge & Jennings, 1995). In lower level control, it can be seen as a structure of executable code adapted according to the project. Most techniques used in holonic systems, i.e., systems based on holons concepts, are present in multi-agent systems.

Indeed, holonic systems are considered a useful framework for designing intelligent control systems with distributed architecture, while multi-agent systems are considered a software developing technique that can be used to implement holonic systems (Giret & Botti, 2005). The integration of the "agent technology" and the holonic systems paradigms with mechatronic is presented as the basis for the intelligent automation of PSs (Schoop *et al*., 2002).

An agent-based representation of a PS on the physical device level allows the conception of an intelligent control component. In fact, each resource of a PS is mapped into an agent, i.e., a production/physical agent (Suessmann *et al*., 2002) that contains all the mechatronic parameters needed for the control, supervision and operation of resources. Moreover, the communications and the information processing capabilities with which the agent-based controls a system component, transforms it into a self-reconfiguring, intelligent element, i.e., a holon. The result is a distributed intelligent automation system associated with the lowest layer of a holonic system. This multi-agent based control system is called HCS (Holonic Control System) (Colombo *et al.*, 2001).

A multi agent system is a suitable approach to intelligent control, from the present characteristics of holonic applications such as modularity, decentralization and capacity for changes, to ill-structured and complex problems, for which the agents are best suited (Parunak *et al*., 1998). A HCS is an agent-based intelligent automation system based on multifunctional hardware-platforms for the flexible integration of hardware and software functionalities (holons). The main characteristic of this class of system is the fact that it is focused on the system's behavior instead of the process-centered approach of conventional automation (Colombo *et al*., 2001).

A survey of related publications (Schoop *et al*., 2002; Leitão & Colombo, 2006; Zhang & Jiang, 2008; Sousa *et al.*, 2004; Colombo *et al.,* 2001; Scheidt, 2002) shows that: (i) there is a small number of works that consider the integration of HCS and AFTCS requirements; (ii) there are few practical applications for these agent technologies, showing that there is still a long way to go to spread these holonic systems, (iii) in most of these systems, there is no negotiation mechanism between holons, and iv) there is no information about the use of a systematic method to structure and rationalize the models development, from specification until operation phase.

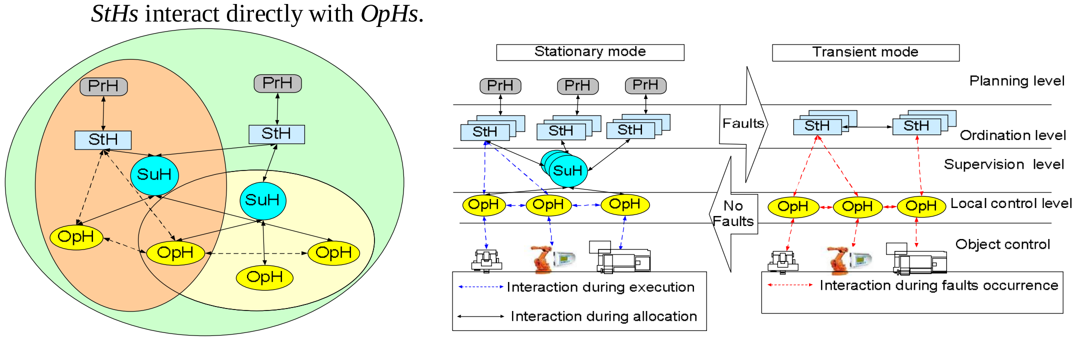
4. AHCS ARCHITECTURE

The proposed architecture for AHCS (Fig. 1) and its mechanisms are described here. In this control architecture, a holon can be a physical device (chiller, sprinkler, programmable controller, etc.) or a logical component (service, order, etc.). Each holon performs different functions, and its individual behavior contributes for other holons to compose the whole system behavior. The proposed architecture is divided into the following levels: planning, ordination, supervision and local control:

- the planning level holon (*PrH* - Product Holon) contains the necessary knowledge for the general operation of PSs and for choosing the general strategy that attains the planned objectives;
- the ordination level holon (*StH* – Strategies Holon) contains the knowledge to manage the execution of each productive strategy that results in a service;
- the supervision level holon (*SuH* – Supervisor Holon) contains all the knowledge to coordinate the holons of lower hierarchical levels, coordinating the tasks list of *OpHs* (agenda), registering the abilities of each component and providing services combined with other entities of the control system. Its main function is to prepare and to implement optimized plans for holons under its coordination taking into account that the system is operating without faults; and
- the local control level holon (*OpH* – Operational Holon) represents the PS physical resources that have specific control devices for its operation, and determines the behavior of these resources in accordance with its objectives and abilities.

Figure 1a illustrates how the holarchies concept is organized in AHCS. The holarchies are represented by ellipses, and a holon can belong simultaneously to different holarchies. The mechanisms adopted allow the control to be switched between two operational modes:

- the "stationary mode", at which the control system is coordinated in a hierarchical way, i.e., *StHs* coordinate the optimized sequence of activities executed by *SuHs*, which in turn supervise the activities executed by *OpHs* (Fig. 1b) during normal situations of the system;
- the "transient mode", at which *OpHs* interact directly with *StHs* (Fig. 1c) to assure more flexibility to the system and agile behavior. For the allocation of *OpH* services or commands, during the "stationary mode", *StHs* interact directly with *OpHs*.



(a) holarchies (b) normal operation (c) with fault occurrence

Figure 1. AHCS architecture.

The switching of control mode is regulated by fault tolerant mechanism called "autonomy factor", i.e., a discrete variable {*low*, *high*} associated to each *OpH*. Initially, *OpH* has an autonomy factor {*low*}, which allows its coordination by *SuH*, running the plans indicated by *SuH*, if *OpH* is available. This factor adapts the behavior of the holons in the presence of fault, thus the fault treatment is initiated. The inputs to this mechanism are: (i) the value of the variable autonomy factor (α), (ii) recovery time (τ) which is the estimate based in time needed for fault recovery, and (iii) the parameter (ρ) which indicates the type of fault.

The application of the fault-tolerance concept in AHCS is divided into four phases and they are present in each holon independently of hierarchical level.

The "estimation phase" involves:

- the detection of symptoms, which can identify the existence of faults by the supervision of processes; and
- the isolation of faults, which is based on a model containing characteristics (type, statistical data, etc.) that allow the identification of a fault.

When the detected symptoms do not allow any conclusion, the system should be programmed to identify the type of fault in very similar cases or to request external intervention.

The reconfiguration is decided in the "planning phase", which is based on predefined priorities such as reduction of performance, shorter recovery time, etc., and on historical data, from which it is possible to measure the statistical significance of each type of fault in terms of frequency rate, recovery time, and operational cost.

The "execution phase" involves sending commands for the execution of the selected action plan.

The last phase is the "learning phase", which involves the storage of relevant data for use in further cases.

Therefore, the AHCS acts in accordance with the following rules:

- if <symptoms> then <selects fault>;
- if <selected fault> then <selects action>;
- if <selected action> then <activates reconfiguration>; and
- if<executed reconfiguration> then <store relevant data>.

A way is described here to estimate the reconfiguration time (Fig. 2). If the resource affected by the fault become unavailable for a long period of time, the *OpH* estimates the recovery time (t_r), checks the planned orders during the recovery period and then cancels the current allocation of these orders notifying the *StH*. For this, OpH estimates two different parameters: (i) t_e, the time that the order needs to return to the *StH*, and (ii) t_c, the time to check if the fault was recovered, and to re-estimate and to re-apply the parameters of time if the fault was not recovered as expected. To calculate these two values it is determined the t_r spent in the previous treatment faults and it is considered on this time 50% for t_c and 90% for t_e. If the fault is not recovered, it is necessary to re-estimate the parameters of time and to cancel the strategies that are planned for this new time interval. It is obtained by t_c+t_e. During the time that the resource is unavailable, the *OpH* only receives new commands if they can be run outside the range of estimated time for recovery.

In AHCS the negotiation mechanism is based on: (i) contract net protocol (CNP) (Smith, 1980), used in a more broad way by Leitão (2009); and (ii) activity-based costing (ABC) (Cooper, 1988). Thus, AHCS adopts rules that allow negotiation between holons based on credits (rewards) and fee (penalties) depending if the order is completed in due time or not. When a *StH* is responsible for implementing a particular strategy, it receives the following information of *PrH*: the strategy chosen, a quantitative measure called “order production fund" (π), the scheduled time, an amount of penalty for delay (φ), and a reward value (ε) to be finalized successfully. The *StH* should manage the negotiation with the *OpHs* to achieve the goal without exceeding the service fund. Table 1 summarizes the evolution of this mechanism. for holonic composition process (HCP).

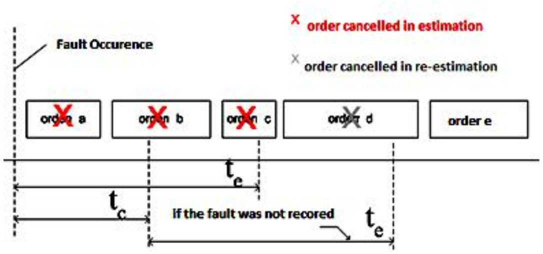


Figure 2: Estimation of reconfiguration time.

Table 1. CNP adapted from Smith (1980) and Leitão (2009).

HCP	*StHs*	*OpHs*
Operation allocation process.	Contracts the operation execution by ξ and the penalty by φ.	Contracts the operation execution by ξ and the penalty by φ.
Finish of an operation with success.	Pays the value ξ to the OpH ($\pi \leftarrow \pi - \xi$).	Increases the total credits by ξ ($\mu \leftarrow \mu + \xi$).
End of an operation with delay	Pays the value ξ and receives the value φ from the OpH ($\pi \leftarrow \pi - \xi + \varphi$).	Decreases the total credits by φ and increase by ξ ($\mu \leftarrow \varphi$).
Operation cancelled (delay, failure, etc.)	Receives the value φ from the OpH ($\pi \leftarrow \pi + \varphi$).	Decreases the total credits by φ ($\mu \leftarrow \mu - \varphi$).

5. PROCEDURE FOR AHCS DESIGN

Here the basic procedure for AHCS development is presented: analysis of requirements, modeling, analyzing/simulation, implementation and operation. The details of each phase are as follows.

Phase 1 – analysis of requirements – on this phase AHCS’ specifications are defined: aim of the system, control object, control devices, definition of tasks, strategies and control functions, and description of the interaction between the parts of the system, and the cases of reconfiguration.

Sub-phase 1.1 – identification of holons–on this sub-phase the holons are identified, i.e., *PrH*, *StH*, *SuH* and *OpH*. The identification of *PrH* (Product Holon) involves the definition of control functions of each product/service offered by the PS’ subsystems and how to perform production/service orders. Thus, *PrH* contains all knowledge necessary to operate the PS and to choose the better strategy to reach the objectives planned. The *StHs* (Strategies Holons) are the entities responsible for the management of control strategies that must be followed during execution phase. *SuHs* (Supervisor Holons) are responsible for coordinating *OpHs*. *SuH* contains all knowledge necessary to coordinate holons on lower hierarchic levels. The function of *SuH* involves the preparation of a program of tasks and coordination of decisions for the performance of these tasks. When a process requests a resource, in fact it is requesting functionality and *SuHs* check the available resources to control the allocation of the resource. *OpH* (Operational Holon) represents human operators and plant’s physical resources, which have any control device for its operation and establish these resources’ behavior according to the objectives and skills. *OpH* manages the behavior of these resources according to the objectives, characteristics and skills, such run time and type of operation.

To process an order, AHCS forms a holarchy which is a production process dynamically created based on the collaboration of holons. According to Fig. 1a, a holarchy is formed by the holons and other sub-holarchies, and these holarchies are represented at other holons. This type of organization is an important characteristic of HCS. The holarchies are represented by ellipses, and a holon can belong simultaneously to different holarchies. This recursive structure (holons made up of holons) allows the designer to analyze each holon in order to figure out the advantages of decomposing it into a new holarchy. This process is repeated until every holon is completely defined and there is no need for further decompositions. The challenge is to determine the best holarchy formed to fulll the order based on the available resources. To achieve this objective, AHCS calculates the timing constraints for the associated transitions in individual holons.

Sub-phase 1.2 – AFTCSs specifications – in this sub-phase the critical points of the system and the faults that may affect the normal performance of functions indispensable to the system are identified. After that identification it is necessary to analyze which critical processes will be subject to reconfiguration. AFTCS functions are divided into the four phases mentioned above. When the symptoms detected do not allow any conclusion, the system must be programmed to identify the kind of fault detected in similar cases or request external intervention.

Sub-phase 1.3 – definition of interaction patterns between holons – the interactive processes are considered in this sub-phase; for example: "request for products/services", "execution" of products/services, “fault treatment” and "reconfiguration" due to faults. The synchronization of E-PN models is made by enabling arcs and inhibitor arcs. These interactions are extracted from UML sequence diagram (Booch *et al.*, 1999).

Phase 2 – modeling considering reconfiguration –the interactions of negotiation between holons are represented using PFS models, and the submission of orders to operational holons *OpHs*; preparation and performance of these orders; and the treatment of faults upon their occurrence are explicitly described. The treatment for occurrence of faults must be represented by means of *SuHs* and *OpHs’* models. The control strategies of AFTCS are modeled on this phase, with the “diagnoser” and the “decider” to fulfill the requirements of the diagnosis and decision phases.

The steps to design the E-PN model of the “diagnoser” are: (i) construction of E-PN models for the components of the control object; (ii) construction of E-PN models of control strategies; (iii) definition of observable events – generally those related to control strategy commands; and non-observable events (Sampath *et al.*, 1996), generally related to faults; (iv) construction of E-PN models of sensors; (v) initiate the construction of the “diagnoser” from the initial state considered “normal” (without faults); (vi) relate, by means of transitions and enabling arcs, the performed strategies with possible observable and non-observable events which may happen from the initial state; and (vii) relate the states obtained with the states of the sensors. If the “diagnoser” does not indicate the correct state then the possible faults’ causes must be inferred to solve possible conflicts. Toward decision making phase of AHCS, some inference rules based on reasoning (Kuipers, 1994) may be adopted for the specification of a mechanism called "decider”. Besides, a system that considers the reconfiguration requires redundant resources to keep an adequate performance, and must also consider the transmission of control signals as part of the system to be controlled, because a fault on this communication network may also limit the coherence of command actions (Scheidt, 2002; Zhang & Jiang, 2008).

Phase 3 – analysis/ simulation – edition and structural analysis are developed with E-PN tools. The behavior and the quantitative analysis were carried out by means of simulation techniques and checking of E-PN properties. This type of analysis allows re-design and re-engineering of the control system during the design phase. This phase is subdivided in: qualitative and quantitative analysis. Qualitative analysis allows the verification of structural properties and behavioral models, sketching conclusions about the system operation, such as: (i) liveness, that is related to the complete absence of deadlocks in system operation; (ii) reachability, to study the dynamic properties of the system; (iii) reversibility, to recover from disruptive events of the system operation; and (iv) conservation and boundedness to verify the resource constraints of the system. The quantitative analysis requires the introduction of the time parameter associated with the transitions. Thus, it is possible to check if the firing is consistent with specifications of the models and evaluating the system´s performance -- see example of the diagnoser and decider in Silva *et al*. (2011a).

Phase 4 – implementation – for the practical use, the resulting models are interpreted as control program specifications to be performed by computers (supervisory control) and programmable controllers (local control level). This phase also comprises the codification, parameterization and development of wrapper interfaces. Hence, for low-level control applications, the code generation will follow one IEC 61131 (Christensen, 2003; IEC, 2001) graphical language (ex. Grafcet), and for high-level applications, the code generation follows a high-level language, such as Java or C.

Phase 5 – operation – in this phase, the real-time supervision of the automation control system is performed by synchronizing the operation of the AHCS with the E-PN models, in order to control and monitor the system. The signals from the sensors and the status of mechatronic devices are acquired and connected with E-PN models. The adaptation and re-configuration of the PS is supported using this procedure, i.e., the introduction or removal of components requires the addition or removal of a token in the corresponding E-PN models and, in some cases, the modification of associated holons models.

6. EXAMPLE

In the following explanation, some models of a FMS (Flexible Manufacturing System) (Fig. 3a) are used to illustrate the main features of the procedure.

Phase 1 - requirements analysis - The FMS consists of three workstations (*WS1*, *WS2*, *WS3*), a inspection and assembly station (IAS), a robot manipulator (*R*), a buffer of loading (*L*) and unloading (*U*), and a storage station pallet (*P*). A human operator (*H*) is responsible for supervision, inspection, maintenance, startup and shutdown. Workstations process one item at each a time, and each station has an input buffer (*Bin*) and an output buffer (*Bout*) considered with infinite storage capacity. *R* is responsible for loading and unloading of items (work pieces) and to move the pallets with items between stations there is a conveyor (*Tr*).

The objective is to perform work orders on items (*A0, A1, A2, A3, B0, B1, B2 e B3*) to compose the final product (*C*). Each *PrH* has an internal process flow, the required input types and output types, which are represented in PFS in phase 2. Work orders (*OpWS1, OpWS2, OpWS3, and OpIAS*) can be executed on items in *WS1*, *WS2, WS3* and *IAS*. After re-initialization (setup), *WS2* can carry out the operation of both *WS1* and *WS3*. This multifunctionality of *WS2* ensure some degree of redundancy in the system.

***Sub-phase 1.1 – identification of holons*:** In Tab. 2 are listed *OpHs* of the FMS, representing your resources. In this example, the input and output buffers are considered part of their workstations, and the set of buffers and station are treated as one *OpH*. There are seven intermediate *PrHs* (*PrH-[A0], PrH-[A1], PrH-[A2], PrH-[A3], PrH-[B0], PrH-[B1], PrH-[B2] and PrH-[B3]*) and there is the *PrH* of the final product (*PrH-[C]*).

StHs are responsible for the management the strategies control, see more details in Silva *et al.* (2012). The identification of *SuH* can be done by reviewing the description of levels of plant control. In this case, only one *SuH* (*SuH-FMS*) is considered for the control system.

Sub-phase 1.2 – AFTCSs specifications – Tab. 3 presents a survey of critical points, treatment fault and reconfiguration or degeneration. The strategies developed based on this survey are sent by *PrHs* to be implemented in both *StHs* and *OpHs*, depending on the desired level of reaction.

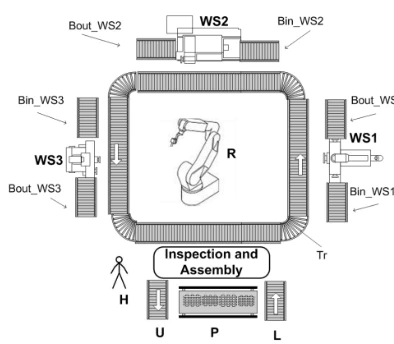
Table 2. OpHs identification.

OpH	Description	Functionality	Un.
R1	robot	transports/ loads/ unloads item (*A,B*)	1
WS1	Work station	executes *OpWS1* , and after setup *OpWS2* and *OpWS3*	1
WS2	Work station	executes *OpWS2*	1
WS3	Work station	executes *OpWS3*	1
IAS	Inspection and Assembly Station	executes *OpIAS*	1
Tr	transporter	transports between *WSs*	1
H	human operator	supervision, inspection, maintenance, initialization and finalization operations	1

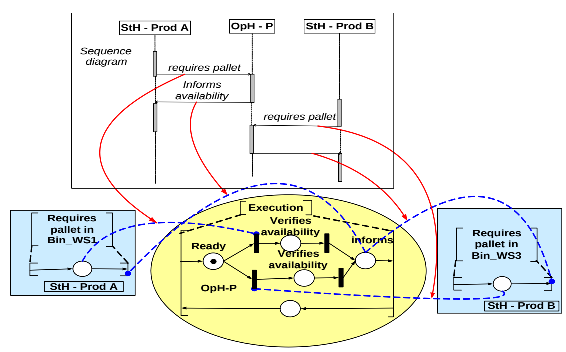
Table 3. List of AFTCS specifications.

critical points	fault treatment	reconfiguration and/ or degeneration
fault in load/ unload	acts in the control variables and resets the controller	switching to semi-automatic control mode and transporting via *OpH - H*
lack of energy	turns on power generator to maintain operations	if the energy generated is not enough to keep the whole plant, turn off energy into other subsystems and maintain the priority areas

Sub-phase 1.3 – definition of interaction patterns between holons –Fig. 3b shows UML diagram, PFS and E-PN interactions to [execution] activity, which occurs between *StHs* and *OpH* on the work order [require pallet]. Observe that *OpH* can receive execution work orders from different *StHs*, named (*StH- prodA*, *StH – prodB*).



(a) layout



(b) UML diagram and interaction between holons

Figure 3. Example of an AHCS for FMS.

Phase 2 – modeling considering reconfiguration – the modeling of *OpHs* involves detailing the behavior of physical devices. Figure 3b also shows the example where *OpH-P* receives requests from two *StHs* and informs the availability of pallets. Figure 4 illustrates the modeling procedure for *PrH-[A3]*: (i) PFS of the production plan, (ii) E-PN and its controller considering the influence of the transmission of control signals, (iii) E-PN of [load *[A3]* in WS2] activity, and (iv) an example for treatment fault and its reconfiguration.

The global production process in FMS and the required input types and output types between the *PrHs* are in Fig. 5, which has also an example internal process flow for *PrH-[A1]*. To model the workflow of a holon in E-PN, a place represents a state in the workflow while a transition represents an event or operation that brings the workflow from one state to another.

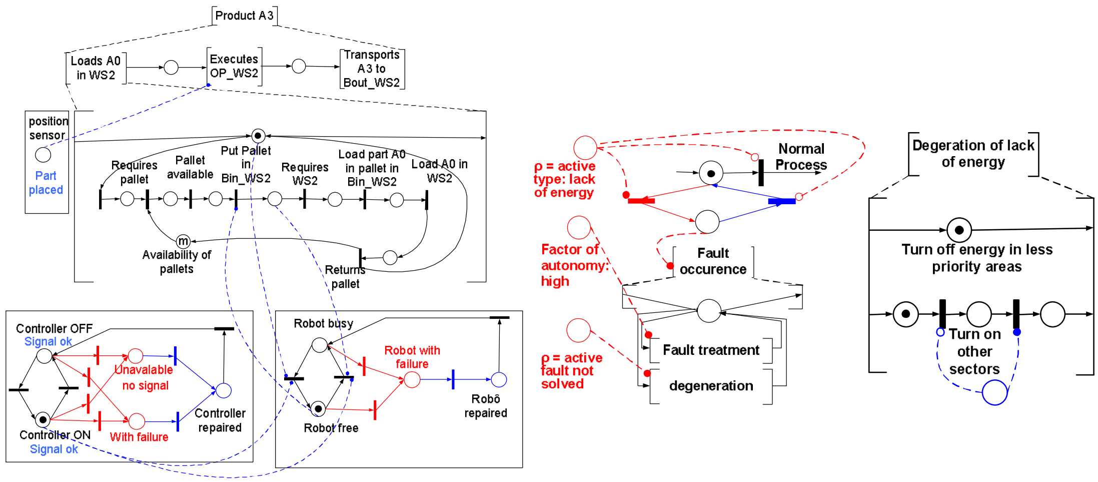
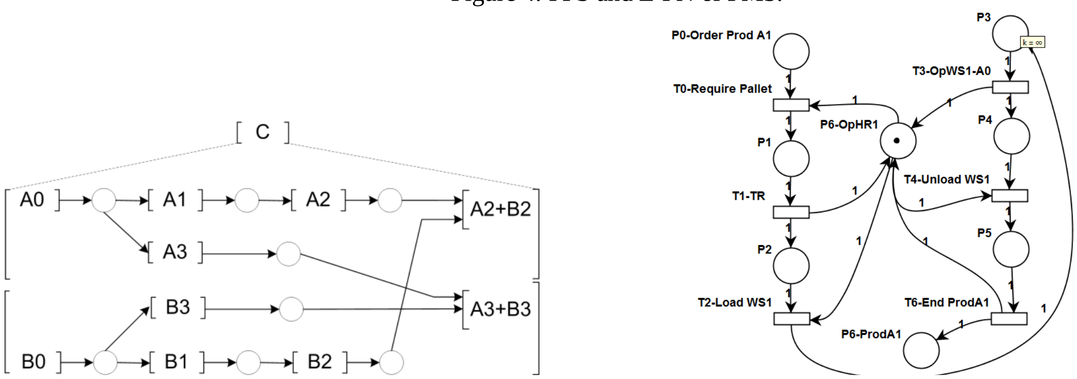


Figure 4. PFS and E-PN of FMS.



(a) global production process (b) production process for *PrH-[A1]*

Figure 5. PFS and E-PN of *PrHs*.

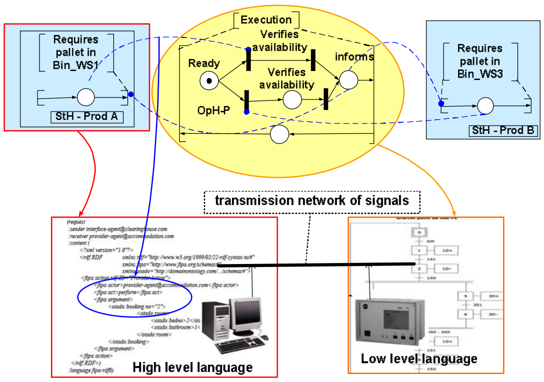
Phase 3 – analysis/ simulation – In this paper the editor and simulator PIPE (Bonet *et al.*, 2007) for E-PN class is used.. This software is a tool for Petri net simulation that has a relatively intuitive interface, is easy to use and supports PNs with timed transitions and inhibitor arcs. The qualitative analysis is based on structural analysis of the E-PN models. Quantitative analysis is performed through the simulation of PN with timed transitions. To this phase, scenarios also are identified using this software with models built for each case - for more details see Silva (2008). The models meet the restrictions and achieve the objectives outlined in the hypothesis.

Phase 4 – implementation – Fig. 6 presents, for low-level control applications, the code generation in Grafcet (derived of E-PN models), and the code generation of a high-level language in Java. For this task the designer may use JADE (Java Agent Development Framework) (JADE, 2011), a software framework fully implemented in Java language. It simplifies the implementation of multi-agent systems through a middleware that complies with FIPA specifications (FIPA, 2011) and through a set of graphical tools that supports the debugging and deployment phases. Fig. 6 shows the example of JADE fragment code for the interaction between *StH-ProdA* and *OpH-P*.

Phase 5 – operation– Silva *et al.* (2012a) present how this system should be globally implemented and also illustrates the operation of this control system. The real-time monitoring system for supervision and control is accomplished by synchronizing the operation of the models in the E-PN with sensor signals the state of devices, allowing the generation of reports and control charts in order to supervise and control the system.

7. CONCLUSIONS

A procedure to design an AHCS (Active Holonic Control System) that considers not only the normal operations but also the occurrence of faults in PSs (Productive Systems) was presented. The development procedure of AHCS is divided into five phases: phase 1 - analysis of requirements; phase 2 - modeling considering reconfiguration; phase 3 - analysis / simulation; phase 4 – implementation; and phase 5 - operation. The modeling process is based on an interpretation and an extension of Petri net, which are used to structure the development of components models and the workflow of the proposed procedure. Thus, the procedure combines bottom-up and top-down approaches, through dynamic modeling of the system in E-PN (extended Petri net) and the gradual refinement associated to PFS (Production Flow Schema).




```
public void action() {
ACLMessagemsg = myAgent.receive();
if (msg != null) {
ACLMessage reply = msg.createReply();
        String content = msg.getContent();
if (content.equalsIgnoreCase("Is it possible to produce product A?")) {
reply.setPerformative(ACLMessage.INFORM);
reply.setContent("I will verify if there is any palet available");
myAgent.send(reply);
System.out.println("The seller "+msg.getSender().getName() + "ordered a product A");
System.out.println("OpH-P:Production Started");
}else{
if(content.equalsIgnoreCase("Paletavailable.Starting Production")){
            String content2 = msg.getContent();
System.out.println("-->"+msg.getSender().getName()+":"+content2);
}else{
if(content.equalsIgnoreCase("Palet is busy")){
              String content2 = msg.getContent();
System.out.println("-->"+msg.getSender().getName()+":"+content2);
```

(a) (b)

Figure 6. Examples of (a) implementation and (b) JADE fragment code for *StH-ProdA* and *OpH-P* interaction.

Mechanisms for fault tolerance are proposed, focusing on reconfiguration, which ensures system flexibility, implementation of a hierarchical or heterarchical control structure, and quicker reaction to faults. AHCS explores the superposition of control system based on multi-agent system and HS (Holonic System) techniques, such as autonomy, reactivity, proactivity, cooperation, social interface (i.e., consideration of human interaction in processes) and learning resources, and takes advantage of the benefits of their implementation characteristics.

Therefore, AHCS innovate combining requirements of HCS and AFTCS (Active Fault-Tolerant Control System) to design PSs from the conceptual stage of the system until its operation, and propose control strategies that allow the reconfiguration of the system.

In this article, a FMS (Flexible Manufacturing System) is used as an example to demonstrate the advantages of AHCS. A larger project is being developed (Silva *et al.*, 2011a) involving researchers and students of UESC (Universidade Estadual de Santa Cruz) and EPUSP (Escola Politécnica da Universidade de São Paulo), which involves in addition to modeling, simulation and validation of E-PN models, and the development of an experimental case study of AHCS.

8. ACKNOWLEDGEMENTS

The authors would like to thank the partial financial support of the Brazilian governmental agencies:CNPq, CAPES, FAPESP, and the UESC, USP.

9. REFERENCES

Artist, 2006, “Roadmap on Real-Time Techniques in Control System Implementation – Control for Embedded Systems Cluster”, EU/IST FP6 Artist2 NoE. Available in: http://www.artist-embedded.org.

Bonet, P., Llad, C. M., Puijaner, R. and Knottenbelt, W. J., 2007, “Pipe v2.5: A Petri net tool for performance modeling”. Proc. of CLEI 23rd Latin American Conference on Informatics.

Booch, G., Rumbauch, J., Jacobson, I., 1999, “The Unified Modeling Language User Guide”. Addison Wesley Longman, Inc.

Christensen, J., 2000, “Basic Principles of HMS Architecture”. Proceedings of the HMS’00 Int. Symposium, Kitakyuschu, Japan.

Christensen, J., 2003, “HMS/FB Architecture and Its Implementation”. In: Agent-Based Manufacturing. Advances in the Holonic Approach. Springer Verlag, 53–88.

Colombo, A.W., Neubert, R., Schoop, R., 2001, “A solution to holonic control systems”. Proc. of ETFA the 8th IEEE Int. Conf on Emerging Technologies and Factory Automation, Sophia/Nice.

Cooper, R., 1988, “The rise of activity-based costing. part one: what is an activity-based cost system?”. Journal of Cost Management, vol. 2, n. 2, pp. 45-54.

David, R., Alla, H., 1994, “Petri nets for modeling of dynamic systems - a survey”. Automatica, vol.30, pp.175-201.
Giret, A., Botti, V., Valero, S., 2005, “MAS methodology for HMS”. Lecture Notes in Artificial Intelligence, 3593, pp.39-49.
Hasegawa, K., Miyagi, P. E., Santos Filho, D. J., Takahashi, K., Ma, L. Q., Sugisawa, M., 1999, “On resource arcs for Petri net modeling of complex shared resource systems”. J. Int. & Robotic Systems, vol.26, n.3/4, pp.423-437.
IEC, 2001, “International Electrotechnical Commission: Function Blocks, Part 1 - Software Tool Requirenments”. PAS 61499-2.
JADE, 2011, “Java Agent Development Framework”, http://jade.tilab.com/.
Jennings, N.R., Wooldridge, M. J., 1998, “Applications of intelligent agents”. *Agent Technology: Foundations, Applications, and Markets*, Springer, pp.3-28,
Koestler, A., 1969, “The Ghost in the Machine”. Arkana Books, London.
Kuipers, B., 1994, “Qualitative Reasoning: Modeling and Simulation with Incomplete Knowledge”. The MIT Press, Cambridge.
Leitão, P., Colombo, A.W., 2006, “Petri net based methodology for the development of collaborative production systems”, Proc. of ETFA the 11th IEEE International Conference on Emerging Technologies and Factory Automation, Prague, pp. 819-826.
Leitão, P., 2009, “Agent-based distributed manufacturing control: a state-of-the-art survey”. Engineering Applications of Artificial Intelligence, vol 22, n.7, pp. 979-991.
Miyagi, P. E. , 1996, “Controle Programável - Fundamentos do Controle de Sistemas a Eventos Discretos”. Editora E. Blucher, São Paulo.
Parunak, V.D., Baker, A., Clark, S., 1998, “The AARIA Agent Architecture: From Manufacturing Requirements to Agent-Based System Design”. Working Notes of the Agent-Based Manufacturing Workshop, Minneapolis.
Reisig, W., 1985, “Petri Nets an Introduction”. Springer Verlag, New York.
Riascos, L. A. M., Miyagi, P. E., 2010, “Fault Tolerance in Manufacturing Systems: Applying Petri nets”. 1. ed. Saarbrücken: VDM Verlag, 156 p.
Sampath, M., Sengupta, R., Lafortune, S., Sinnamhoideen, K. Teneketzis, D.C., 1996, “Failure diagnosis using discrete-event models”. IEEE Trans. on Control Systems Technology, vol. 4, n.2, pp.105-124.
Schoop, R., Colombo, A.W., Suessmann, B., Neubert, R., 2002, “Industrial experiences, trends and future requirements on agent-based intelligent automation”, Proc. of IECON the Annual Conf. of IEEE Industrial Electronics Society, vol.4, pp. 2978-2983.
Scheidt, D. H., 2002, “Intelligent agent-based control”, Johns Hopkins Technical Digest, vol.23, n.4, pp.383-395.
Silva, R. M., 2008, “Modelagem de sistemas de controle de edifícios inteligentes considerando a ocorrência de falhas”. Dissertação (Mestrado) – Universidade de São Paulo, 189p, São Paulo.
Silva, R. M. ; Miyagi, P. E. ; Santos Filho, D. J., 2011a, “Design of active fault-tolerant control systems”. In: Springer IFIP Advances in Information and Communication Technology, v. 349. p. 367-374.
Silva, R. M., Martins, M. P., Junqueira, F., Santos Filho, D. J., Miyagi, P. E., 2011b, “Projeto de Sistema Ativo de Controle Holônico para Sistemas Produtivos”. In: Anais of COBEF 2011, 6º Congresso Brasileiro De Engenharia De Fabricação.
Silva, R. M., Arakaki, J., Junqueira, F., Santos Filho, D. J. and Miyagi, P. E., 2012a, “Modeling of active holonic control systems for intelligent buildings”. Automation in Construction, vol. 25, pp. 20-33.
Silva, R. M., Arakaki, J., Junqueira, F., Santos Filho, D. J. and Miyagi, P. E., 2012b, “A procedure for modeling of holonic control systems for intelligent building (HCS-IB)”. Advanced Materials Research, vol. 383, pp. 2318-2326.
Smith, R., 1980, “The contract net protocol: high-level communication and control in a distributed problem solver”, IEEE Transactions on Computers, vol. 100, n.12, pp.1104-1113.
Sousa, P., Ramos, C., Neves, J., 2004, “The fabricare system”. Prod. Planning & Control, vol. 15, n. 2, pp.156-165.
Suessmann, B., Colombo, A.W., Neubert, R. , 2002, “An Agent-based Approach Towards the Design of Industrial Holonic Control Systems”. Proc. of the 5th IEEE- IFIP Int. Conf. on Information Technology for Balanced Automation Systems in Manufacturing and Services. Cancun, México.
Tommila, T. et al., 2005, “Next generation of industrial automation – Concepts and architecture of a component-based control system”. VTT Research Notes 2303.
Wooldridge, M., Jennings, N. R., 1995, “Intelligent Agents: Theory and Practice”. In Knowledge Engineering Review 10(2), 1995.
Woll, D., 2007, “Setting the Stage for the Next Generation of Automation Control System Software: IEC 61499”. ARC Advisory Group – Industry Trends.
Zhang, Y., Jiang, J., 2008, “Bibliographical review on reconfigurable fault-tolerant control systems”. Annual Reviews in Control, vol.32, pp.229-252.

RESPONSIBILITY NOTICE

The authors are the only responsible for the printed material included in this paper.