18th ICPR paper: From System Requirements to Holonic Manufacturing System Analysis

Adriana Giret, Vicente Botti

► To cite this version:

Adriana Giret, Vicente Botti. 18th ICPR paper: From System Requirements to Holonic Manufacturing System Analysis. International Journal of Production Research, 2006, 44 (18-19), pp.3917-3928. 10.1080/00207540600696336 . hal-00512908

HAL Id: hal-00512908

<https://hal.science/hal-00512908>

Submitted on 1 Sep 2010

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

18th ICPR paper: From System Requirements to Holonic Manufacturing System Analysis

Journal:	<i>International Journal of Production Research</i>
Manuscript ID:	TPRS-2005-IJPR-0483
Manuscript Type:	Original Manuscript
Date Submitted by the Author:	29-Nov-2005
Complete List of Authors:	Giret, Adriana; Universidad Politecnica de Valencia, Sistemas Informaticos y Computacion Botti, Vicente; Universidad Politecnica de Valencia, Sistemas Informaticos y Computacion
Keywords:	MULTI-AGENT SYSTEMS, HOLONIC MANUFACTURING SYSTEMS
Keywords (user):

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

From System Requirements to Holonic Manufacturing System Analysis

Adriana Giret and Vicente Botti

Universidad Politecnica de Valencia, Camino de Vera s/n,

46022, Valencia, Spain

Phone: +34 963877000

{agiret, vbotti}@dsic.upv.es

Abstract

Research on Holonic Manufacturing Systems (HMS) is undergoing a huge amount of interest and developments. In spite of the great amount of effort devoted to architectures and algorithms for holonic control of manufacturing systems there are little work about design methodologies. In this work we present a multi agent approach for analysis of HMS, it is based on an abstract agent definition and the PROSA reference architecture for holonic systems. The analysis process to identify and specify holons is organized around five views of the system and integrates the advantages of proven multi agent methodologies to the manufacturing field. An industrial case example from the ceramic domain is presented to show the characteristics of the approach.

Keywords: holonic manufacturing systems, multiagent systems, requirements, analysis.

1 Introduction

To date, many of the developments in Holonic Manufacturing Systems (HMS) have been conducted in an almost "empirical way", without any software engineering methodology. There is a definite need to have methodologies for holonic systems (McFarlane and Bussmann 2003), that are based on software engineering principles in order to assist the system designer at each stage of development. This methodology should provide clear, unambiguous analysis and design guidelines. We believe that methodologies from the Multi Agent Technology are good candidates for modelling HMS due to the following: the similarities between the holonic and the agent approaches, the wide use of agents as the implementation tool for holonic systems, and the availability of complete Multi Agent System Methodologies. However, there are some extensions that must be included in a MAS methodology to be able to model the HMS requirements in a proper way: holon recursive structure, system abstraction levels, HMS specific guidelines, and a mixed top-down and bottom-up approach for the analysis and design steps. In this work we present a Multi Agent approach for HMS analysis. Section 1, presents the holon and agent modelling paradigms. In Section 2, we detail our approach for HMS analysis. Finally, in Section 3, we summarize the conclusions and future works.

2 Holonic Manufacturing Systems: a new modelling approach

Holonic Manufacturing is a paradigm developed in the framework of Intelligent Manufacturing Systems (IMS) programme. Holonic Manufacturing is based on the concept of "holonic systems", developed by Arthur Koestler (Koestler 1971). The word holon is a combination from the Greek holos = whole, with the suffix -on which, as in

proton or neutron, suggests a particle or part. Koestler propose the word holon to describe the hybrid nature of sub-wholes/parts in real-life systems; holons simultaneously are self-contained wholes to their subordinated parts, and dependent parts when seen from the inverse direction. Work in the HMS programme has translated these concepts to the manufacturing world, viewing the manufacturing system as consisting of autonomous modules (holons) with distributed control. The goal is to attain in manufacturing the benefits that holonic organization provides to living organisms and societies, i.e., stability in the face of disturbances, adaptability and flexibility in the face of change, and efficient use of available resources. The HMS concepts combines the best features of hierarchical and heterarchical organization (Dilts et al. 1991). It preserves the stability of hierarchy while providing the dynamic flexibility of heterarchy. The HMS consortium has defined the following holon characteristics and holonic concepts (HMS 1994):

- Holon - an autonomous and cooperative building block of a manufacturing system for transforming, transporting, storing and/or validating information and physical objects. The holon consists of an information processing part and often a physical processing part. A holon can be part of another holon.
- Autonomy - the capability of a holon to create and control the execution of its own plans and/or strategies (and to maintain its own functions).
- Cooperation - the process whereby a set of holons develops mutually acceptable plans and executes them.
- Self-organization - the ability of holons to collect and arrange themselves in order to achieve a production goal.

- Holarchy - a system of holons that can cooperate to achieve a goal or objective. The holarchy defines the basic rules for cooperation of the holons and thereby limits their autonomy.

In the last ten years, an increasing amount of research has been devoted to holonic manufacturing (HMS) over a broad range of both theoretical issues and industrial applications. We can divide these research efforts into three groups (McFarlane and Bussmann 2003): (i) Holonic Control Architectures, (ii) Holonic Control Algorithms and (iii) Methodologies for HMS. In spite of the large number of developments reported in the first two areas, there is very little work reported on Methodologies for HMS. There is a definite need to have methodologies for holonic systems that are based on software engineering principles in order to assist the system designer at each stage of development. This methodology should provide clear, unambiguous analysis and design guidelines. We are convinced that Multi-Agent System (MAS) technologies are good candidates for modelling HMS. In this work we illustrate it with a HMS analysis approach based on MAS technology.

The base concept in the MAS technology is the agent notion. An agent is an autonomous and flexible computational system that is able to act in an environment (Wooldridge and Jennings 1995). Flexible means, that the agent is:

- Reactive, it reacts to the environment it is in.
- Pro-active, it is able to try to fulfil its own plan or goals.
- Social, it is able to communicate with other agents by means of some language.

Some properties which are usually attributed to agents to a greater or lesser degree for solving particular problems are (Nwana 1996), (Franklin and Graesser 1996):

- Autonomy: agents can operate without the direct intervention of humans or other agents.
- Social ability: agents are able to interact with other agents (human or not) through an agent communication language.
- Rationality: an agent can reason about perceived data in order to compute an optimal solution.
- Reactivity: agents are able to perceive the environment's stimulus and these stimulus guide the agents' actions on their environment.
- Pro-activeness: agents are not only stimulus-reacting entities, but also have an enterprising character and can act guided by their own goals.
- Adaptability: it is related to the learning that an agent may do and with its capabilities to change its own behaviour based on this learning.
- Mobility: it is the capability of an agent to move through a network.
- Veracity: an agent can not deliberately provide false information.
- Benevolence: an agent is willing to help other agents if this is not against it's owns goals.

Holons and agents are very similar concepts (for a detailed comparison of these two notions see (Giret and Botti 2004a)). In (Giret and Botti 2004a), we pointed out that the recursive structure is the only holon property that is not presented as such in the agent definition. To cope with this limitation, in (Giret and Botti 2004d) we proposed the

Abstract Agent notion as a modelling artefact for autonomous entities with recursive structures. The Abstract Agent extends the traditional agent definition adding a structural perspective to the agent concept: "... an Abstract Agent can be an agent; or it can be a MAS made up of Abstract Agents ..."

In the intelligent manufacturing field, the need for some kind of hierarchical aggregation in real world systems has been recognized. These systems have to remain readable while they are expanded in a wide range of temporal and spatial scales. For example, a modern automobile factory, incorporates hundreds of thousands of individual mechanisms (each of which can be an agent) in hundreds of machines which are grouped in to dozens or more production lines. Engineers can design, build, and operate such complex systems by shifting from the mechanism, to the machine or to the production line (depending on the problem at hand) and by recognizing the agents of higher levels as aggregations of lower-level agents. Also, in e-commerce applications, an enterprise is a legal entity which is independent of the individual people whose are its employees and directors.

The Abstract Agent is an attempt to unify the concepts of holons and agents and to simplify and close the gap between holons and agents in the analysis and design steps. This will make it easier to translate the modelling products that are obtained from methodologies for HMS into coding elements for the implementation of the holonic system.

3 Analysis of Holonic Manufacturing Systems

In our approach, the HMS is specified by dividing it in more specific characteristics that form different views of the system. These views are defined in terms of MAS technology; therefore, we talk about agents, roles, goals, beliefs, organizations, etc. The way in which the views (models) are defined is inspired by the INGENIAS methodology (Pavon and Gomez 2003) (a complete MAS methodology that has good performance in the development of complex systems). The extensions we have made to the INGENIAS meta-models deal with the following: the addition of the Abstract Agent notion and the properties to model real-time behaviours (Julian and Botti 2004), the redefinition of some relations to conform to the new modelling entities and the dependencies between them. Here we summarize the models (for more information see (Giret and Botti 2004b, c)):

- The *agent model* is concerned with the functionality of each Abstract Agent: responsibilities and capabilities (what roles it may play, what goals it is committed to pursue and what task it is able to perform).
- The *organization model* describes how system components (Abstract Agents, roles, resources, and applications) are grouped together, which are executed in common, which goals they share, and what constraints exists in the interaction among agents and among Abstract Agents.
- The *interaction model* addresses the exchange of information or requests between Abstract Agents. Each interaction declaration includes involved actors (Abstract Agent, Role), goals pursued by interaction, and a description of the protocol that follows the interaction.

- The *environment model* defines the non-autonomous entities with which the Abstract Agents interacts. These elements are resources and applications.
- The *task/goal model* describes relationships among goals and tasks, goal structures, and task structures. It is also used to express which are the inputs and outputs of the tasks and what are their effects on the environment or on the Abstract Agent's mental state.

In Figure 1 we show the notation of our approach.

Following, we present the specification of the analysis phase of our approach. We use modelling diagrams from a Ceramic Tile Factory case study¹ to illustrate it. The tile factory is divided into departments each of which is in charge of a specific function (marketing, tile design, production planning, factory, raw material warehouse, finished tile warehouse, etc.). The tile production process is as follows: the clay is obtained, mixed, refined, dried, pressed or extruded, decorated/glazed and baked in ovens known as kilns. The HMS for the Tile Factory must: (i) integrate the different departments of the company, (ii) arrange factory resources for both on-demand (60%) and stock (40%) production orders, and (iii) automate resources and processes controls at different levels in the company.

The aim of the analysis phase is to provide high-level HMS specifications from the problem *Requirements*. The analysis adopts a top-down recursive approach. One

¹ The ceramic tile sector is recognized at the national and international levels as being a very dynamic sector in both the number of exports and the high investment capacity in advanced research technologies. The continuous market changes in this sector have motivated a great increase in product variety and ad hoc product properties. The case study requirements were defined in a joint research project between the GTI-IA group and the CIGIP group of the Polytechnic University of Valencia.

1
2
3
4 advantage of a recursive analysis is that its results, i.e. the *Analysis Models* (agent
5 model, organization model, interaction model, task/goal model, environment model),
6
7 provide a set of elementary elements and assembling rules.
8
9

10
11
12
13
14 In the analysis phase (Figure 3), the designer must specify the HMS in terms of the five
15 models described before and the *UML Use Case Diagrams*. This is a top-down,
16
17 recursive, incremental process. The main goal of the analysis phase is to identify the
18
19 constituent holons and to provide an initial holon specification.
20
21
22

23
24
25 The *Analysis Models* are produced from the *Requirements Set* and the *Domain*
26
27 *Definition*. Each iteration of the analysis phase identifies and specifies holarchies of
28
29 different levels of recursion (holons made up of holons). The first iteration identifies an
30
31 initial holarchy, which is made up of holons that cooperate to fulfil the global system
32
33 requirements. At the end of every iteration, the designer must analyze each holon in
34
35 order to figure out the advantages of decomposing it into a new holarchy. In this way,
36
37 each new iteration will has as many concurrent processes as constituent holons of the
38
39 previous iteration that was decided to decompose. This process is repeated until every
40
41 holon is completely defined and there is no need for further decompositions.
42
43
44
45

46
47 An iteration definition is as follows.
48
49
50
51

52
53 **3.1 Determine Use Cases**
54

55
56 The first step is to Determine Use Cases by building a Use Case Model from the system
57
58 Requirements. We have defined the HMS UC Guidelines to help the designer to
59
60 identify domains cooperation (Fletcher et al. 2000) and the system goals as use cases.

Use cases can be considered as simpler sub-problems that taken together define the entire system. The HMS UC Guidelines are the followings:

1. Is the system going to produce something? If yes, one goal of the system will be to produce it.
2. Is there some process to control? If yes, a system goal will be to control it.
3. Is there some external resource to work with? If yes, a system goal will be to work with it.
4. Can the system receive a work order? If yes, a system goal will be to process the work order and to control its execution process.
5. Do the system goals need to be controlled or managed? If yes, a system goal will be to manage them.
6. It is needed to communicate the system products to some external business process? If yes, a system goal will be to communicate the products and to manage the communication.
7. Define a Use Case for every goal derived from question 1.
8. Define a Use Case for each goal related with the execution control of a machine.
9. Define a Use Case for every goal related with the control of a process.
10. For each communication goal with external resources define a Use Case to manage the communication.
11. For each goal related with a work order define a Use Case to implement the domain cooperation to process it.
12. Define a Use Case for every goal derived from question 6.
13. Define a Use Case to manage every communication process with external business processes.

Figure 3a shows an initial Use Case Diagram of the Tile Factory which is a work product of the first iteration of the analysis phase.

3.2 Specify Use Cases Realization

The Use Cases identified in the previous step are analyzed in the step Specify Use Cases Realization. In this step every Use Case is represented as an Abstract Agent and the interaction and relationships among them are modelled by building *Interaction Models* (Figure 3b) and *Organization Models*.

Every Use Case is associated with a provider of its functionality. That is, an autonomous entity (Abstract Agent) which is in charge of providing the Use Case service without worrying whether it is an atomic entity or a composite one (this will be specified in subsequent analysis iterations). The different Abstract Agents identified in this step are organized with an *Organization Model*. The relationships and interactions among Use Cases are specified by modelling relations and interactions among the Abstract Agents responsible of the different Use Cases. These are specified completing *Interaction Models* and refining *Organization Models*. The interaction relations are detected when there are communication needs among the Abstract Agents. These interactions can be of coordination, planning, negotiation or cooperation types. While the organization relations can be of master-slave, service provider or service customer types.

The tasks that an Abstract Agent needs to implement to fulfil the Use Case it is in charge of are specified in a *Task/Goal Model*. These tasks are associated with the

system goals identified in previous step. Moreover, new goals may be specified answering questions such as, why is an interaction activated between two Abstract Agents? Is there some condition that needs to be satisfied in an interaction execution? Is there some management needs for an interaction?

Figure 4a illustrates the corresponding *Organization Model* obtained from the *Use Case Model* of Figure 3a.

3.3 Identify Holons

In the third step, Identify Holons, the designer works with the work products of the previous step, the system Requirements, and the PROSA Guidelines to identify any new Abstract Agent and to categorize the identified Abstract Agents. The PROSA Guidelines are defined based on PROSA types of holons (Van Brussel et al. 1998). The rules for identifying holons are:

1. There are four types of Abstract Agents: resource, product, work order and staff.
2. A product Abstract Agent models “things” that may be produced by the holarchy. If the holarchy represents the whole factory, the product Abstract Agent models items which are in the product catalogue of the company.
3. A work order Abstract Agent models the holarchy requested tasks. It is responsible to activate the resource Abstract Agents to start the production process. If the holarchy represents the whole factory, the work order Abstract Agent models the book order items.
4. A resource Abstract Agent models an element of the system which provides processing capabilities. If the holarchy represents the whole factory, the resource

Abstract Agent models the departments, shops, machines, conveyors, pipelines, components, tools, tool holders, personnel, etc.

5. An staff Abstract Agent models an entity responsible of the interaction management activities among different Abstract Agents.
6. Each production means (a factory, a department, a shop, machine, conveyor, pipeline, component, tool, tool holder, personnel, etc.) and the information processing that controls it, is modelled as an Abstract Agent. For every production means of the analyzed holarchy, there is an Abstract Agent which models it, and is in charge of its management and control.
7. A resource Abstract Agent provides production capabilities and services to the Abstract Agents with which he collaborates. He maintains the production resource assigning methods, the knowledge and processes to organized it, he uses and controls the production resource.
8. Each product definition or recipe is modelled as an Abstract Agent. The product Abstract Agent maintains updated product information, user requirements, product design, product process plans, material bills and quality assurance processes. He maintains the “product model” of a type of product, not the “production state” of a particular product instance.
9. A product Abstract Agent maintains the production knowledge and process to guarantee the correct manufacturing of the product, with the sufficient quality defined in the Requirement document. He acts like an information server to the work order Abstract Agents of the holarchy. The product Abstract Agent models product design related functionalities, process planning and quality assurance.

10. Each task in the manufacturing system (customer order, make-to-stock order, prototype-making order, order to maintain and repair resources, etc.), is represented as an Abstract Agent. A work order Abstract Agent is in charge of: managing a production process or a negotiation process; managing the production state of the requested products in the work order; and managing all the logistic information related with the task.
11. A work order Abstract Agent may be seen like a piece with some controlling behavior, which is in charge of managing the work order in the manufacturing system, that is, to negotiate with other entities and resources to get produced.
12. The goals identified in previous steps which are related with interaction management among different Abstract Agents, must be assigned to staff Abstract Agents.
13. A staff Abstract Agent has to assist with expert knowledge to the other Abstract Agents, and to implement the holarchy control.
14. A resource Abstract Agent is able to initiate processing tasks on products, that is, he is authorized to accept or reject an assigned task based on his goals.
15. A resource Abstract Agent controls and monitors the execution of his processes (pause, resume, abort), manages his sub-resources, and plans his tasks.
16. A product Abstract Agent implements the designing tasks of the product, the process planning, and the product quality verification.
17. A work order Abstract Agent executes production scheduling, deadlocks management, order execution monitoring, and triggers process events (start, pause, resume, abort or stop) within a resource.

18. A resource Abstract Agent maintains beliefs about his capabilities (list of processed products), his executing tasks, his sub-resources, and a log of his activities.
19. A product Abstract Agent maintains information about a process plan, a product specification and its quality requirements.
20. A work order Abstract Agent maintains information about the product state of production, the tasks execution, and tasks history data.
21. A staff Abstract Agent maintains management information about the interactions he controls and (possibly) expert knowledge (plans, heuristics, etc.) for assisting other Abstract Agents.
22. Types of work order Abstract Agents are: make to stock orders, first-of orders, maintenance orders, client orders.
23. Types of product Abstract Agents are: high quality products, a new version of a product, a product variation, mass production, spare parts, etc.
24. Types of resource Abstract Agents are: information resources, machines, tools, personnel, operations, etc.
25. In a $n + 1$ analysis iteration, the Abstract Agent interface which models it in the n iteration (previous iteration) can be modelled in two ways: (i) The management responsibility is assigned to a staff Abstract Agent, that is, every communication to/from outside must be controlled by the staff Abstract Agent; or (ii) The management responsibility of the holarchy interactions is distributed among the different Abstract Agents of the holarchy.
26. To trigger a new work order within a holarchy the following steps are needed: (i) the staff Abstract Agent, or the manager of the holarchy, figures out if the holarchy is able to process the order; if yes, (ii) he activates a work order Abstract Agent

responsible to process it and acknowledge it to the user; finally (iii) when the work order is done, the staff Abstract Agent communicates it to the user.

27. To execute the requested tasks of a work order the followings steps are needed: (i) the work order Abstract Agent requests a process plan to the product Abstract Agents; (ii) the work order Abstract Agent start a negotiation process with the resources Abstract Agents to assign tasks (of the process plan) to resources; finally (iii) the work order Abstract Agent start the tasks execution asking to the resources Abstract Agent to start the process.
28. When a new resource Abstract Agent is added to the holarchy, every Abstract Agent of the holarchy is notified. The product Abstract Agents must verify the utility of the new resource for his process plans.

Based on these rules the designer must refine both the *Organization Model* and the *Interaction Model* by adding new or modified relations and interactions among holons in the cooperation domains. Figure 5b shows the *Interaction Model* to define a new schedule for an order, or to modify a previously defined schedule due to factory failures or new orders. The *Agent Model* (Figure 4b) is built to specify holon capabilities and responsibilities in terms of tasks and goals which are described in detail in the *Task/Goal Model*.

3.4 Specify Environment Relations

The *Environment Model* is built in the fourth step, Specify Environment Relations, to represent non-autonomous domain entities. The *Environment Model* is built based on the system *Requirements* and the *Analysis Models* derived from the previous steps. A

list with external events that may affects to every Abstract Agent must be defined along with temporal constraints on the events. A list with applications with which the Abstract Agents may interact must be specified. Every identified application must be associated with the Abstract Agent that uses it specifying the operations, parameters and desired outputs.

4 Conclusions

In this work, we have presented the notation and modelling process of a MAS approach for HMS analysis. In our approach the HMS is specified by dividing it in more specific characteristics that form different views of the system. These views are defined based on INGENIAS (a proven MAS methodology for complex systems) (Pavon and Gomez 2003). We have extended the INGENIAS models to be able to model the HMS requirements properly. These extensions include the notion of Abstract Agent (Giret and Botti 2004d) and the properties to model real-time behaviours (Julian and Botti 2004). They also include the redefinition of some relations to conform to the new modelling entities and the dependencies among them. The aim of the analysis phase is to provide high-level HMS specifications from the problem Requirements, which are specified by the Client/User. The analysis adopts a top-down recursive approach. One advantage of a recursive analysis is that its results, i.e. the Analysis Models, provide a set of elementary elements and assembling rules. Our approach provides HMS-specific guidelines for identifying and specifying holons.

We are currently working on completing the design phase, the definition of the Holon Implementation stage and the Set-Up and Configuration Step. We are also working on

the evaluation of our approach with industrial case studies; for example: the Ceramic Tile Factory presented in this work, and an Assembly and Supplier Company of Automobile Parts. As future work we plan to develop CASE tools for our approach.

REFERENCES

Dilts, D., Boyd, N., and Whorms, H. (1991). The Evolution of Control Architectures for Automated Manufacturing Systems. *Journal of Manufacturing Systems*, 10(1):79–93.

Fletcher, M., Garcia-Herreros, E., Chritensen, J., Deen, S., and Mittmann, R. (2000). An Open Architecture for Holonic Cooperation and Autonomy. *Proceeding of HoloMAS'2000*.

Franklin, S. and Graesser, A. (1996). It is an agent, or just a Program?: A Taxonomy for Autonomous Agents. *Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages*. Springer-Verlag.

Giret, A. and Botti, V. (2004a). Holons and Agents. *Journal of Intelligent Manufacturing*, 15:645–659.

Giret, A. and Botti, V. (2004b). On the definition of meta-models for analysis of large-scale MAS. *Multiagent System Technologies*, pages 273–286. LNAI 3187.

Giret, A. and Botti, V. (2004c). Towards a Recursive Agent Oriented Methodology for Large-Scale MAS. *Agent-Oriented Software Engineering IV.*, LNCS 2935:25–35.

Giret, A. and Botti, V. (2004d). Towards an Abstract Recursive Agent. Integrated Computer-Aided Engineering, 11(2):165–177.

HMS, P. R. (1994). HMS Requirements. HMS Server, <http://hms.ifw.uni-hannover.de/>.

Julian, V. and Botti, V. (2004). Developing real-time multiagent systems. Integrated Computer-Aided Engineering, 11:135–149.

Koestler, A. (1971). The Ghost in the Machine. Arkana Books.

McFarlane, D. and Bussmann, S. (2003). Agent-Based Manufacturing. Advances in the Holonic Approach, chapter Holonic Manufacturing Control: Rationales, Developments and Open Issues, pages 301–326. Springer- Verlag.

Nwana, H. S. (1996). Software Agents: An Overview. Intelligent Systems Research. AA&T, BT Laboratories.

Pavon, J. and Gomez, J. (2003). Agent Oriented Software Engineering with INGENIAS. 3rd International Central and Eastern European Conference on Multi-Agent Systems (CEEMAS 2003) : V. Marik, J. Miller, M. Pechoucek:Multi-Agent Systems and Applications II, LNAI 2691, pages 394–403.

1
2
3
4 Van Brussel, H., Wyns, J., Valckenaers, P., Bongaerts, L., and Peeters, P. (1998).
5
6 Reference Architecture for Holonic Manufacturing Systems: PROSA. Computers In
7
8 Industry, 37:255–274.
9
10

11
12
13
14 Wooldridge, M. and Jennings, N. R. (1995). Intelligent Agents - Theories,
15
16 Architectures, and Languages. Lecture Notes in Artificial Intelligence, Springer-Verlag.
17
18 ISBN 3-540-58855-8, 890.
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

Figure Captions

Figure 1: Modelling notation.

Figure 2: Analysis work-flow for HMS specification.

Figure 3: Use Case Model and Interaction Model of the Ceramic Tile Factory.

Figure 4: Organization Model and Agent Model of the Ceramic Tile Factory.

For Peer Review Only

For Peer Review Only

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

```
ERROR: invalidaccess
OFFENDING COMMAND: --filter--

STACK:

/LZWDecode
-filestream-
[48 0 0 -129 0 129 ]
true
129
48
-savelevel-
```