An Open Architecture for Holonic Cooperation and Autonomy

M. Fletcher [2] E. Garcia-Herreros [3] J.H. Christensen [1] S.M. Deen [2] R. Mittmann [3]

[1] Rockwell Automation, 24800 Tungsten Road, Euclid, OH 44121, USA. JHChristensen@ra.rockwell.com

[2] Computer Science Dept. University of Keele, Staffs. ST5 5BG, UK. {deen, martyn}@cs.keele.ac.uk

[3] Softing, Richard Reitzner Allee 6, Haar, Munich 85540, Germany. {gh, Rainer.Mittmann}@softing.com

Abstract

The paper examines some issues relating to an open architecture for holon cooperation and autonomy. We identify the requirements of a holonic system architecture and discuss the merits of our approach in comparison with classic agent-based models. A suitable architecture to satisfy these requirements is also presented, together with a discussion of the holonic kernel needed to support distributed holonic control. The material presented here is based on results from the international programme on Holonic Manufacturing Systems (HMS).

1 Introduction

Holonic Manufacturing Systems have been proposed by the HMS Consortium [1] as a way to meet the ever-increasing needs for robustness to disturbances, adaptability and flexibility to rapid change, and efficient use of available resources on the factory floor of modern manufacturing enterprises. These systems in turn can lead to the realization of the "agile manufacturing" vision, where "reprogrammable, reconfigurable, continuously changeable production systems, integrated into a new information intensive manufacturing system, make the lot size of an order irrelevant." [21]

To facilitate the widespread adoption and deployment of HMS technology, a comprehensive architecture is required. Such an architecture must provide a framework for the unambiguous specification of the structure and relationship among functional units (*holons*) in the system. In particular, the architecture must enable the accomplishement of the characteristics of:

- **autonomy**: Each holon must be able to create, control and monitor the execution of its own plans and/or strategies, and to take suitable corrective actions against its own malfunctions.
- **cooperation**: Holons must be able to negotiate and execute mutually acceptable plans and take mutual actions against malfunctions.
- **openness**: The system must be able to accommodate the incorporation of new holons, the removal of existing holons, or modification of the functional capabilities of existing holons, with minimal human intervention, where holons or their functions may be supplied by a variety of diverse sources.

In this paper, we propose an open architecture for holonic systems to meet these requirements, based on a number of evolving public standards.

2 State of the Art

A Holonic Manufacturing System (HMS) utilizes engineering paradigms including the Function Block Architecture [2] and multi-agent systems to guarantee predictability and termination during task execution. Multi-agent systems originate from research in Distributed Artificial Intelligence (DAI) [3] and use mentalistic approaches to problem solving by imitating human interaction. These approaches are often based on speech acts or beliefs, desires and intentions. Such models are inherently unpredictable and may be unstable in real-world manufacturing where criteria such as fault-tolerance and reconfiguration are paramount.

The Speech Act theory of Searle [4] encouraged multi-agent researchers to develop inter-agent cooperation protocols, treating communication as a type of action to be incorporated into planning and reasoning processes. Primitives inspired from speech act theory include propose, refuse, respond and inform; these have been used as a basis for many multi-agent prototypes.

Belief, desire and intention (BDI) was introduced as the foundation for single-agent architectures by Bratman et al [5] and was developed further by Rao and Georgeff [6]. Since its conception, the BDI scheme has become a solid foundation for research into multi-agent architectures. The BDI model defines an agent's internal processing through a set of mental categories with a control framework for the rational selection of action plans to satisfy goals. At present there are several multi-agent architectures based on various aspects of speech act and BDI principles. These include the

Cosy architecture of Haddadi et al [7], INTERRAP from Mueller [8], the GRATE model from Jennings [9] and the MECCA architecture from Steiner et al [10]. These architectures use artificial intelligence to control agents' behaviour and model agent coordination using human social metaphors. However none addresses the problems of distributed control and cooperation within manufacturing.

During the 1990s, DARPA brought about the introduction of a LISP-based environment that integrates a knowledge interchange format (KIF) [11] and a knowledge querying and manipulation language (KQML) [12]. These formats are becoming the de facto standards for exchanging knowledge between agents. Thus FIPA [13] (foundation for intelligent physical agents) has defined an agent architecture and message exchange formats based on the aforementioned BDI and KQML ideas respectively. The HMS project does not intend to define new standards where existing ones are sufficient. Hence holons pass messages complying with the FIPA standard, however it is unclear how the agent model can fulfill the architectural requirements of the HMS project [14].

Recently there have been a number of papers that address the closely related areas of agent-based and holonic manufacturing. In [15], Parunak proposes a manufacturing schedule and control testbed (MASCOT) as a virtual factory in which agent ideas can be experimented with. The PROSA architecture of Brussel et al [16] consists of three types of basic holons: product, resource and order. These holons are structured using the object-oriented concepts of specialization and aggregation. Staff holons assist basic holons by providing expert knowledge. The research group headed by Norrie [17] is producing a considerable volume of relevant material relating to the MetaMorph multi-agent architecture, concept graphs and holon/human interaction.

As a consequence of work undertaken as part of the systems engineering work-package of the HMS project, several papers have been produced. Deen [18, 19] has produced cooperation models whereby autonomous agents maintain fault tolerance and task execution preferences without getting into deadlocks. The temperature model of Fletcher [20] provides algorithms to redistribute tasks throughout the system, thus ensuring greater robustness and efficiency. In this paper, we shall present an open architecture for holon cooperation and autonomy, inspired by these multi-agent/holon frameworks, and drawing from work done by the authors in defining the function block architecture.

3 System Architecture

3.1 Holons

A manufacturing holon is an autonomous and cooperative building block of a manufacturing system for transforming, transporting, storing and/or validating information, and/or physical objects. Therefore a manufacturing holon usually comprises knowledge and software components with an (optional) hardware component. Functionally, a holon may be considered to comprise an intelligent control system (head) and a processing system (base).

The elements of the holon's Intelligent Control System (ICS) are shown in Figure 3.1.

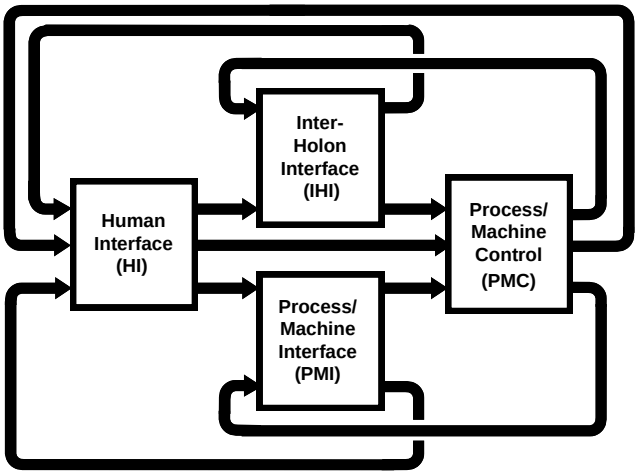

Figure 3.1 - Structure of the Holon Intelligent Control System (head).

- *The process/machine control PMC* responsible for execution of the control plan for the process being controlled. The control block may include besides traditional control algorithms rule-based reasoning, fuzzy logic, and neural nets
- *The process/machine interface PMI* provides the logical and physical interface to the processing system via a suitable communication network (e.g. Fieldbus, Profibus or TCP/IP based, wireless). Real-time communication is supported through a Real-time layer. The PMI may itself contain intelligent elements such as e.g. self-diagnosis, process model-oriented diagnosis, etc.
- *The human interface HI* comprises the interfaces to humans such as operators, supervisors, maintenance personnel, and process engineers. It may include front ends, diagnostic and explaining components
- *The inter-holon interface IHI* handles the inter-holon communication. It also comprises the elements to permit the holon to negotiate and cooperate with other holons. The inter-holon interface also provides facilities to support the cooperation domain interfaces through a Cooperation Communication System or Layer.

Focusing on the internal organization, a holon consists of the intelligent control system (ICS), consisting of the control and regulation components, and the processing system. The processing system consists of all processing

components necessary to realize a manufacturing activity as transforming, transporting, storing and/or validating information and/or physical objects (e.g. database management system). The ICS is responsible for the interacting behaviour of the internal components, as well as the set of procedural rules and decision-making functions that govern the interaction of the components. In this way the ICS enables the holon to offer manufacturing skills as an autonomous subsystem in coordination with the environment and acquaintance holons. The processing system is responsible for the manufacturing functionality according to rules and operating strategies imposed by the ICS.

3.2 Cooperation Domains

A Cooperation domain is considered as a logical space in which holons communicate and operate, that provides the context where holons may locate, contact and interact with each other. It is possible that a cooperation domain does not exist by itself, and that all cooperation domains may be dynamically generated by the operations of holons' constituent parts. The following premises are valid for a cooperation domain: A holonic system contains at least one cooperation domain. A holon is a member of one or more cooperation domains. A cooperation domain has one or more member holons.

A cooperation domain comprises the following elements

- Data structures into which holons may write and read knowledge which controls cooperation, e.g. querying the value of a variable that indicates the status of a joint task. .
- Facilities to pass transient messages between holons and the cooperation domain. If the cooperation domain and the holons involved within that cooperation domain are located on the same device then message passing can be achieved by shared memory, else the messages must be encoded as packets and passed through a communication network.
- Decision making mechanisms to support holons in their activities, such as task planning, negotiation, information exchange and so forth
- Techniques and rules to decompose and allocate tasks among compound holons, as well as facilities to schedule and control tasks within a holon
- Facilities to monitor the status of a distributed task, and schedule/control all actions within this task

Within a cooperation domain, constituent holons coordinate, through their respective views of the holarchy, to generate and execute task plans. A holon is mapped onto one or more holonic resources within the system, where at least one such resource must provide cooperation domain management services. Within a cooperation domain, information-processing elements of holons interact with each other to accomplish a specified task.

A holon may be composed of a set of other holons in a recursive containment hierarchy/heterarchy (a holarchy) to form a *compound* or *parent holon*. In this case, lower level holons included within the holon cooperate with each other through their respective cooperation domains to generate task plans and to carry out these tasks. In the case where a holon does not include lower level holons (namely an *atomic* or *child holon*), the internal cooperation domain represents the holon's private autonomous functions and information. The architectural structure of the cooperation domain described here is founded upon properties and behaviour of the holon components contributing to the cooperation domain.

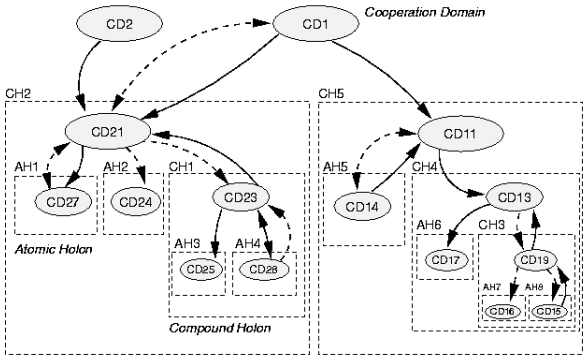

Figure 3.2 - Holarchy of holons and its cooperation domains

Inside a cooperation domain, we envisage two types of cooperation among autonomous holons. *In simple cooperation*, an autonomous system is committed to answer queries from another holon, even if the response is non-cooperative, e.g. access denied. All holons posses this ability. A *complex cooperation* achieves a joint goal, for instance agreeing a mutual plan of executing tasks for solving a distributed problem. We present a framework for such complex inter-holon cooperation in a later section, which is based on basic cooperation domains, thus providing scalability, flexibility and so simplifies the holonic architecture.

The structure created by this task holarchy is dynamic while the relationships among holons form a more static configuration. Holons respond to task requests from their cooperation domains so that interaction is carried out or new tasks are generated according to these responses. If a task cannot be executed due to a lack of equipment or skills then the task may be altered or a new component could be introduced, under the control of the corresponding minder, into a holon to satisfy the domain's requirements. As a result, the internal structure of the holarchy represented by this cooperation domain is altered through generation of a new compound holon.

3.3 Open Interfaces

Figure 3.3-1 illustrates the characteristics that must be achieved at theinterfaces among engineering tools (T) and holons (H) in order to achieve the architectural goal of openness, namely: *portability* of software elements at the T/T interface; *configurability* of holons at the T/H interface; and *interoperability* of holons at the H/H interface.

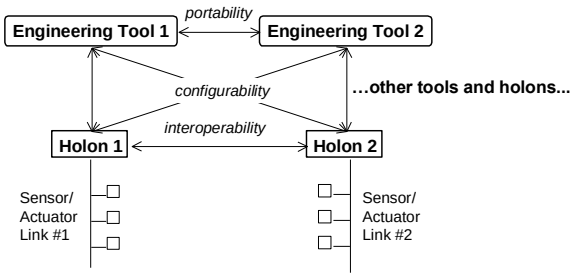

Figure 3.3-1 - Open Interface Characteristics

The IEC 61499 standard[1] achieves these interface characteristics by the following means:

- Software *portability* is achieved through standardized semantics and XML [22] syntax for exchange of software library elements among software tools.
- Holon *configurability* is achieved through standardized semantics and syntax for configuration commands and responses, which may be exchanged between engineering tools and holons, among holons, or within a holon.
- Holon *interoperability* is achieved through standardized syntax and semantics for exchange of information among holons in real time.

The fundamental building block of functionality in the IEC 61499 architecture is the *function block*. As illustrated in Figure 3.3-2, IEC 61499 defines a graphical representation and textual syntax for representation of the following aspects of function block *types* (classes):

- Event and data interfaces;
- An event-driven state machine for the control of the execution of the function block's algorithms and issuance of resulting output events.

In addition, IEC 61499 provides graphical representations and textual syntax for specifying the sequence of service primitives and their association with events and data when the function block represents an interface to *services* such as communications, machine or process input/output, and human interface elements. It is this capability which enables all elements shown in Figure 3.1 (HI, PMC, PMI and IHI) to be implemented as interconnected *instances* of IEC 61499 function block *types*. IEC 61499 also provides graphic representations and textual syntax for the construction of such *function block networks* for the configuration of holonic control devices.

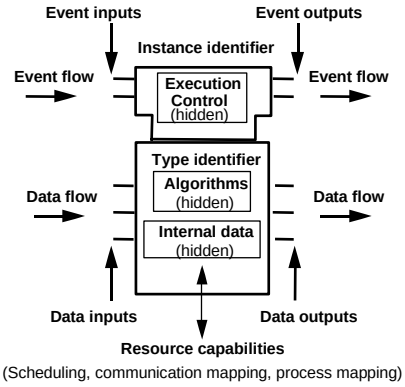

Figure 3.3-2 - Open Interface Characteristics

Finally, IEC 61499 defines the interface to device management services, such as the creation and interconnection of function block instances, as simply another type of service interface function block. It is this characteristic of the IEC 61499 architecture which permits holonic controllers to be dynamically reconfigured in response to task negotiations, as discussed below.

4 Holonic Kernel

The *holonic kernel* (HK) is a layered framework of IEC 61499 function blocks. A holonic kernel resides on each holonic resource and facilitates holon management through the provision of suitable services. We assume that a holon is dynamically created as a heterarchy of function blocks upon one or more holonic resource(s). This formation must satisfy the requirements demanded by that holon's autonomy, cooperation and openness roles. Contrary to traditional manufacturing paradigms, holons are managed in a distributed fashion through interaction with their respective holonic kernels. The holonic kernel assists the holons by offering services including:

- Selecting appropriate function blocks to provide the autonomous skills needed to perform a given task.
- Managing data/event flows between function blocks.
- Supporting interaction and negotiation protocols etc with other holons through cooperation domains.
- Access to data/knowledge bases via suitable interfaces.
- Assisting in task decomposition, information filtering, creating and validating schedules, and handling interrupts.

These are essential services for achieving the necessary flexibility of a HMS as discussed earlier. Structurally, the

holonic kernel is a connected group of four service interface function blocks:

- *Function Block Manager* (FBM) to support function block administration (i.e. creating, configuring and ultimately deleting function blocks) upon a resource.
- *Coordination Manager* (CM) to facilitate interaction control (e.g. task decomposition, result aggregation, planning and conflict resolution) both within the holon and amongst other holons.
- *Cooperation Domain Interface* (CDI) gives interfaces from holons to one or more cooperation domains for transferring knowledge relating to tasks, ontologies etc.
- *Data/Knowledge Base Interface* (DKBI) to manage information in the holon's local repository.

The layered structure of a holonic kernel and the relationships between these kernels are shown in Figure 4.1.

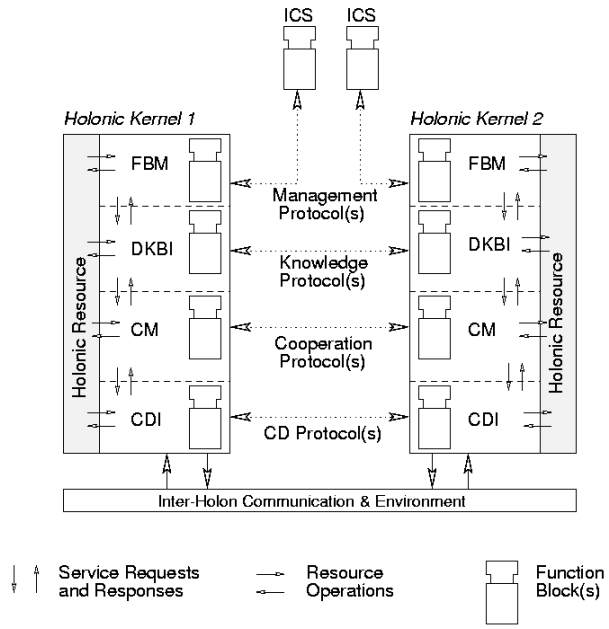

Figure 4.1 - Structure of the Holonic Kernel

The CDI supports holon interaction by:

- Arranging a holon's contribution to every cooperation domain it is simultaneously participating within.
- Exchanging knowledge as a consequence of executing a given cooperation strategy.
- Representing the coordinated task using a suitable schedule of atomic actions.

There is one CDI for every cooperation domain the holon is participating within. Therefore the holonic control system generates a heterarchy of CDI function blocks that reflects a task decomposition structure and the holon's relationships.

Each CDI can be implemented with varying degrees of sophistication. This complexity reflects the design of the cooperation domain, which can be either:

- A logical inter-connection structure that exists only while holons exchange messages.
- Have data retention associated with coordinated actions and performs processing (in conjunction with holons).

We assume a robust communication system enables holons to pass messages through the cooperation domain. As shown in Figure 4.2, a cooperation domain can be constructed using services provided holonic kernels. Each holon joins a cooperation domain by interacting with the CDI and so creating appropriate SUBSCRIBE and PUBLISH function blocks to acquire/present data from/to each cooperation domain. The CM function block can then use this cooperation domain and associated function blocks to exchange relevant information with other holons. We assume that a holon has one CM function block per cooperation strategy that is being executed concurrently. This is because the holon can perform:

- A negotiation strategy during task agreement.
- A coordinated scheduling strategy during planning.
- A conflict resolution strategy during task execution.
- A two-phase commit to ensure data consistency.

Each strategy exchanges knowledge and activity requests via a single cooperation domain associated with the task.

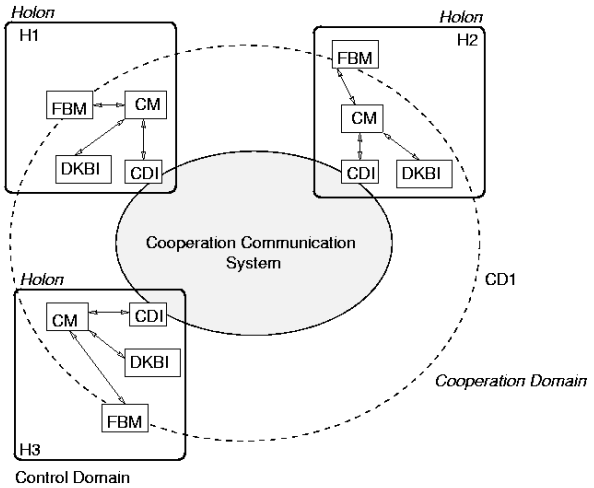

Figure 4.2 - Creating a Cooperation Domain

Concentrating on the CDI, appropriate service primitives are offered either to the holon or to the cooperation domain. Such services can be invoked by passing events and data to the CDI function block.

A UML specification of the CDI is given in Figure 4.3.

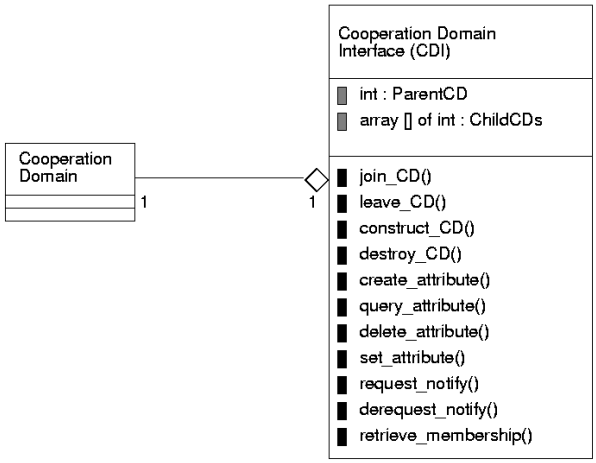

Figure 4.3 - UML Specification of CDI

Like other service interface function blocks, CDI maps events and data (see Figure 4.4) to service primitives (see Figure 4.5).

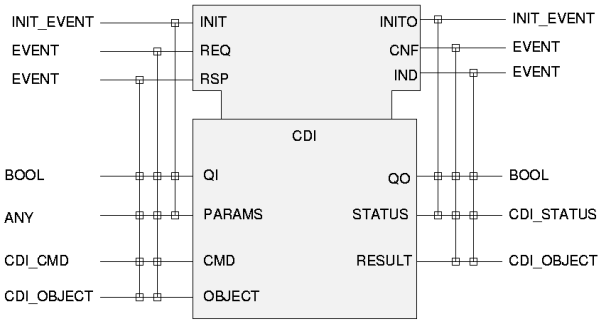

Figure 4.4 - Events and Data Interface to CDI.

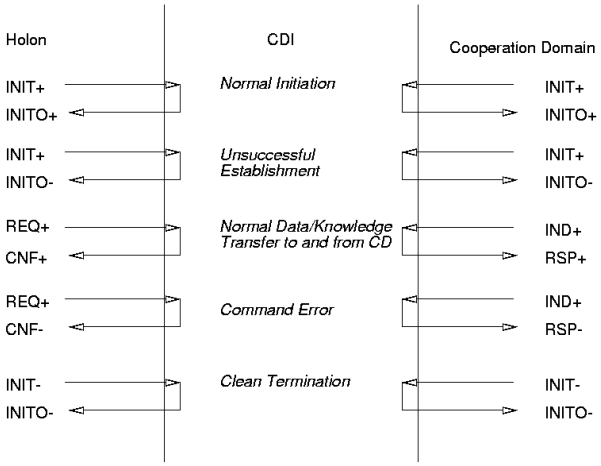

Figure 4.5 - Service Sequences for CDI.

Developers can then implement the functionality of CDI to satisfy these interfaces.

5 Conclusion and Future Work

The paper has presented an architectural model for building holonic manufacturing systems using holons and cooperation domains. We have illustrated how this framework satisfies some of the requirements for agile, customized and fault-tolerant environments that will typify manufacturing in the 21st Century.

Future work within the HMS programme is planned on:

- Automated application generation.
- Task/application reasoning and negotiation.

Acknowledgements

European authors wish to acknowledge that this material is based on work supported by the European Union project (EU IMS Project No: 26508) on Holonic Manufacturing.

References

[1] "The IMS (Intelligent Manufacturing Systems) project on holonics is an international programme with participation of major industries, academic institutes and vendors. The programme is being supported by USA, Canada, Europe, Australia and Japan." Overview at http://hms.ifw.uni-hannover.de/public/overview.html

[2] "Draft - Publicly Available Specification - IEC 61499: Function Blocks for Industrial-Process Measurement and Control Systems, Part 1 - Architecture, Part 2 - Engineering Task Support," International Electro-technical Commission, Geneva, April 2000.

[3] "Open Information Systems Semantics for DAI", C. Hewitt, in AI Journal, January 1991.

[4] "Speech Acts", J.R. Searle, published by Cambridge University Press, 1969.

[5] "Plans and Resource Bounded Practical Reasoning", M.E. Bratman et al, in Computational Intelligence (4:4), November 1988.

[6] "BDI Agents – From Theory to Practice", A.S. Rao and M.P. Georgeff, in Proceedings of the 1st International Conference on MAS, May 1995.

[7] "Communication and Cooperation in Agent Systems – A Pragmatic Theory", A. Haddadi, published by Springer, October 1996.

[8] "The Design of Intelligent Agents", J.P. Mueller, published by Springer, May 1996.

[9] "Controlling Cooperative Problem Solving in Industrial Multi-Agent Systems using Joint Intentions", N.R. Jennings, in Artificial Intelligence (74:2), July 1995.

[10] "Understanding Cooperation – An Agent's Perspective", D.D. Steiner and A. Lux, in Proceedings of the 1st International Conference on MAS, May 1995.

[11] "Knowledge Interchange Format Reference Manual", M.R. Genesereth and R.E. Fikes, Computer Science Department, Stanford University, June 1992.

[12] "KQML – A Language for Protocol and Information Exchange", T. Finin and R. Fritzson, in Proceedings of 13th International Workshop on DAI, September 1993.

[13] "Specification of Agent Architecture", FIPA is a non-profit organization with 35 members to define agent standards, 1998.

[14] "Holonic Manufacturing Systems: Initial Architecture and Standards Directions", J.H. Christensen, in Proceedings of the 1st European Conference on Holonic Manufacturing Systems, July 1994.

[15] "MASCOT: A Virtual Factory for Research and Development in Manufacturing Scheduling and Control", H. Parunak, in Proceedings of Workshop on Intelligent Scheduling in Manufacturing, January 1993.

[16] "Reference Architecture for Holonic Manufacturing Systems: PROSA", H.V. Brussel et al, in Computers In Industry (37:3), March 1998.

[17] "Agent-Based Systems for Intelligent Manufacturing: State of the Art Survey", W. Shen and D.H. Norrie, in Knowledge and Information Systems (1:2), May 1999.

[18] "A Fault-Tolerant Cooperative Distributed System", S.M. Deen, in Proceedings of the 9th IEEE Workshop on Database and Expert System Applications, August 1998.

[19] "A Theoretical Foundation for Cooperating Knowledge Based Systems", S.M. Deen and C.A. Johnson, in Proceedings of 11th International Symposium on Methodologies for Intelligent Systems, June 1999.

[20] "Task Rescheduling in Agent-Based Manufacturing", M. Fletcher, in Proceedings of the 10th IEEE Workshop on Database and Expert System Applications, August 1999.

[21] "21st Century Manufacturing Enterprise Strategy: An Industry-Led View," Iacocca Institute, 1991

[22] "eXtended Markup Language (XML) Specification," W3C Consortium, 1998.