3

Development of Holonic Manufacturing Execution Systems

Fan-Tien Cheng, Chih-Feng Chang and Shang-Lun Wu

1. Introduction

Today, most semiconductor manufacturing companies utilize Manufacturing Execution Systems (MES) (MacDonald, 1993; Samanish, 1993; Nguyen, 1996; Scatt, 1996; MESA, 1997) to deliver information to optimize production activities from order booking through design, production, and marketing to realize the agile manufacturing enterprise. The MES market is composed of several vendors providing an integrated suite of application products (called an integrated MES), and 200, or so, vendors offering individual point solutions (Scott, 1996). An integrated MES may have many advantages, such as a single-logic database, rich functionality, well-integrated applications, and a single model of factories, products, and manufacturing processes. However, integrated MES's are sometimes regarded as monolithic, insufficiently configurable, and difficult to modify. Point solutions can offer best-in-class capabilities for a particular function (such as cell controller, work-in-process (WIP) tracking, statistical process control, scheduling, etc.); the end result is multiple databases, multiple models, and integration nightmares plus maintenance costs (McGehee, et al. 1994; Kadar et al., 1998).

In order to solve the problem of the dichotomy between the integrated MES and point solutions, the concept of the integratable MES has been proposed (Scott, 1996). With the integratable MES, each application can be both a self-sufficient point solution, and can be integrated into a larger suite of products. Therefore, the integratable MES offers an open, modularized, configurable, distributed, and collaborative environment such that rapid implementation, complexity reducing, agility, cost-effective integration, easiness of use, and ownership cost reducing may be achieved (McGehee et al., 1994; Kadar et al., 1998).

McGehee et al. (1994) presented the Texas Instruments Microelectronics Manufacturing Science and Technology (MMST) CIM System Framework, which was based on open-distributed system and object technologies. This re-

Source: Manufacturing the Future, Concepts - Technologies - Visions , ISBN 3-86611-198-3, pp. 908, ARS/plV, Germany, July 2006, Edited by: Kordic, V.; Lazinica, A. & Merdan, M.

engineering effort used the OMT methodology models (Rumbaugh et al., 1991) to express the MMST Framework. Following the MMST CIM System Framework, SEMATECH developed the CIM Framework Specification version 2.0 (SEMATECH, 1998), which is an abstract model for typical semiconductor manufacturing systems.

Several approaches to distributed manufacturing architectures were surveyed by Kadar et al. (1998), and their fundamental features were highlighted. Moreover, an object-oriented simulation framework for development and evaluation of multi-agent manufacturing architectures was introduced by Kadar et al. (1998). Further, Cheng, et al. (1999) applied the distributed object-oriented technologies to develop the MES Framework. This framework has the characteristics of openness, modularization, distribution, reconfigurability, interoperability, and easy maintenance.

Common automatic manufacturing systems have fragility and security problems that also need to be seriously taken into consideration, however these two issues are not considered in the MES frameworks mentioned above. This paper applies the concepts of holon and holarchy to redesign a Holonic Manufacturing Execution System (HMES) Holarchy that not only possesses the characteristics of the MES Framework (Cheng et al., 1999) but also has the properties of failure recovery and security certification.

The concepts of holon and holarchy are originated from mechanisms of social organizations and biological organisms (Valckenaers et al., 1994; Tonshoff et al., 1994; HMS; Van Leeuwen & Norrie, 1997). They have the characteristics of intelligence, autonomy, coordination, reconfigurability and extensibility. Based on these characteristics, the major weakness in the automatic manufacturing systems, fragility, is removed so that the failure recovery feature is attained. Security certification also can be considered.

A typical deployment diagram for HMES in the semiconductor packaging plant is displayed in Fig. 1. HMES includes Shop-Floor Holon, Scheduling Holon, WIP Holon, Data Warehouse, Material Handling, Equipment Holon, Equipment, AGV, AS/RS and so on. The HMES Holarchy will be developed by a systematic approach in this paper. For demonstration purpose, one of the functional holons - WIP Holon - will be designed and implemented. Most of the studies concerning holonic manufacturing systems (Markus et al., 1996; Ramos, 1996; Hino & Moriwaki, 1999) focus on factory architecture and/or how to assign a production task to each manufacturing holon. The purpose of this paper is to propose a systematic approach for developing a workable

Holonic Manufacturing Execution System (HMES) by applying up-to-date software and information technologies. The systematic approach is started with system analysis by collecting domain requirements and analyzing domain knowledge.

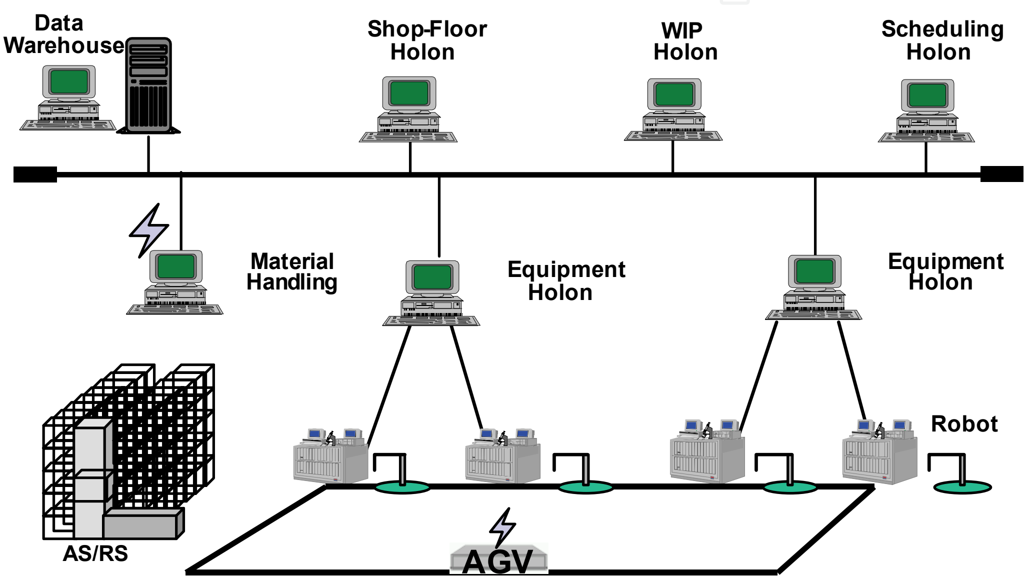


Figure 1. Deployment Diagram for Holonic Manufacturing Execution Systems

The HMES Holarchy is designed by the procedure of constructing an abstract object model based on domain knowledge, partitioning the application domain into components, identifying generic parts among components to form the Generic Holon, developing the Generic Holon, defining holarchy messages and the holarchy framework of HMES, and finally designing functional holons based on the Generic Holon. The technologies (Chen & Chen, 1994; Gamma et al., 1995; Mowbray, 1995; Orfali et al., 1996; Sparks et al., 1996) of distributed object-oriented approach, design pattern, framework, N-tier client/server architecture, and component software are applied to develop the entire HMES and its functional holons.

This paper is organized as follows: Section 2 introduces the characteristics of holon and holarchy. Section 3 describes the development procedure of HMES. This development procedure includes four stages: system analysis, holarchy design, application construction, and system integration and testing. Among those stages, holarchy design needs most elaboration and it is explained in detail in Section 4. Section 5 demonstrates WIP holon design. Section 6 describes application construction and system integration. Section 7 makes comparisons among Legacy MES, Framework MES, and Holonic MES. Finally, this paper ends with summary and conclusions.

2. Characteristics of Holon and Holarchy

Twenty-six years ago, the Hungarian author and philosopher Arthur oestler proposed the word holon to describe a basic unit of organization in biological and social systems. A holon, as Koestler devised the term, is an identifiable part of a system that has a unique identity, yet is made up of sub-ordinate parts and in turn is a part of a larger whole.

The strength of holonic organization, or holarchy, is that it enables the construction of very complex systems that are nonetheless efficient in the use of resources, highly resilient to disturbances (both internal and external), and adaptable to changes in the environment in which they exist. All these characteristics can be observed in biological and social systems.

The stability of holons and holarchies stems from holons being self-reliant units, which have a degree of independence and handle circumstances and problems on their particular level of existence without asking higher level holons for assistance. Holons can also receive instruction from and, to a certain extent, be controlled by higher-level holons. The self-reliant characteristic ensures that holons are stable and able to survive disturbances. The subordination to higher-level holons ensures the effective operation of the larger whole.

The task of the Holonic Manufacturing System (HMS) consortium is to translate the concepts that Koestler developed for social organizations and living organisms into a set of appropriate concepts for manufacturing industries. The goal of this work is to attain in manufacturing the benefits that holonic organization provides to living organisms and societies, e.g., stability in the face of disturbances, adaptability, and flexibility in the face of change, and efficient use of available resources.

As an initial step, the HMS consortium developed the following list of definitions (among others) to help understand and guide the translation of holonic concepts into a manufacturing setting (Van Leeuwen & Norrie, 1997; Ulieru, 1997):

a) **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.

b) **Autonomy:** The capability of an entity to create and control the execution of its own plans and/or strategies.
c) **Cooperation:** A process whereby a set of entities develops mutually acceptable plans and executes these plans.
d) **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.
e) **Holonic Manufacturing System (HMS):** A holarchy that integrates the entire range of manufacturing activities from order booking through design, production, and marketing to realize the agile manufacturing enterprise.
f) **Holonic Attributes:** The attributes of an entity that make it a holon. The minimum set is autonomy and cooperatives.

Based on the above definitions, it is clear that holonic manufacturing systems can be regarded as a unified way to approach the hierarchical control of any manufacturing unit from the production process to the whole enterprise level.
In this work, the concepts of holon and holarchy are adopted to develop the HMES Holarchy so that the functional holons of the HMES can possess the properties of intelligence, autonomy, cooperation, reconfigurability, and extensibility. In addition, the functional holons of the HMES Holarchy can have the capabilities of failure recovery and security certification.

3. Development Procedure of Holonic Manufacturing Execution Systems

As depicted in Fig. 2, the development procedure of HMES includes four stages: (a) system analysis, (b) holarchy design, (c) application construction and (d) system integration and testing. Note that the final step of holarchy design stage is functional holon design and implementation.
The first stage, system analysis, concentrates on collecting domain requirements and analyzing domain knowledge. The second stage, the most important stage, is holarchy design, which is further divided into seven steps as shown in Fig. 2.

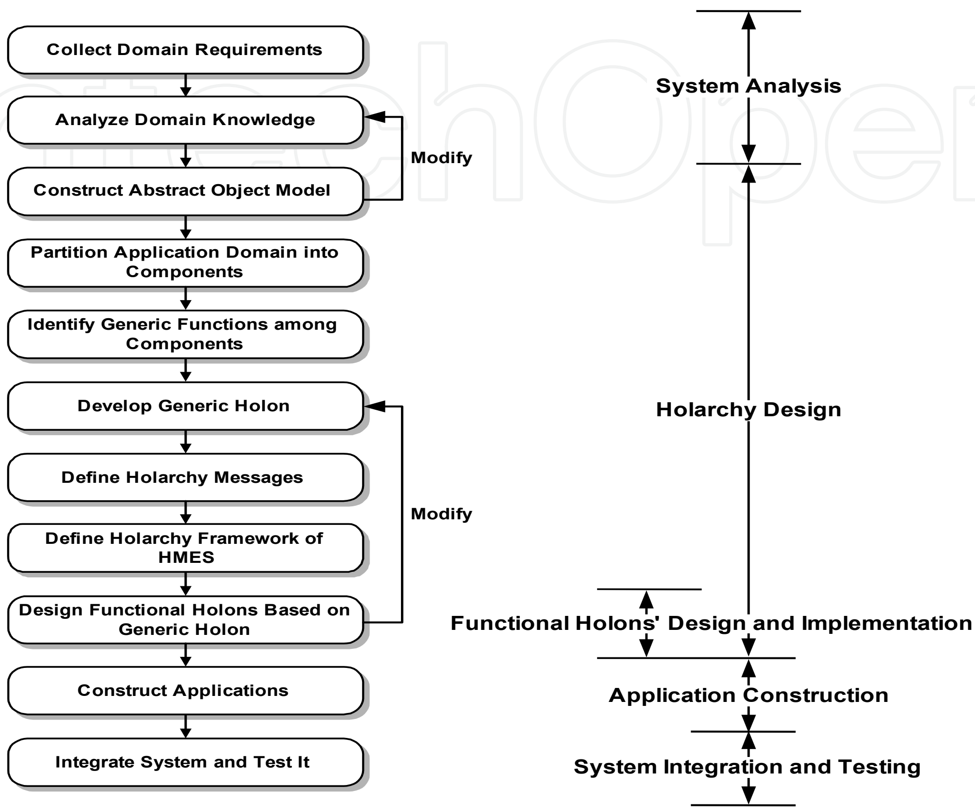


Figure 2. Development Procedure of Holonic Manufacturing Execution Systems

The system's object model is constructed according to the domain knowledge and requirements. The application domain is partitioned into components that will eventually become various functional holons. Within these components, their generic functions are further identified and extracted. Based on these generic functions, the so-called Generic Holon is developed. Holarchy messages among functional holons are defined and holarchy framework of HMES (also denoted HMES Holarchy) is developed. Finally, various functional holons can be designed by inheriting the Generic Holon and implementing the holarchy messages. The third stage of HMES development is application construction. Finally, the development procedure ends with system integration and testing.

4. Holarchy Design

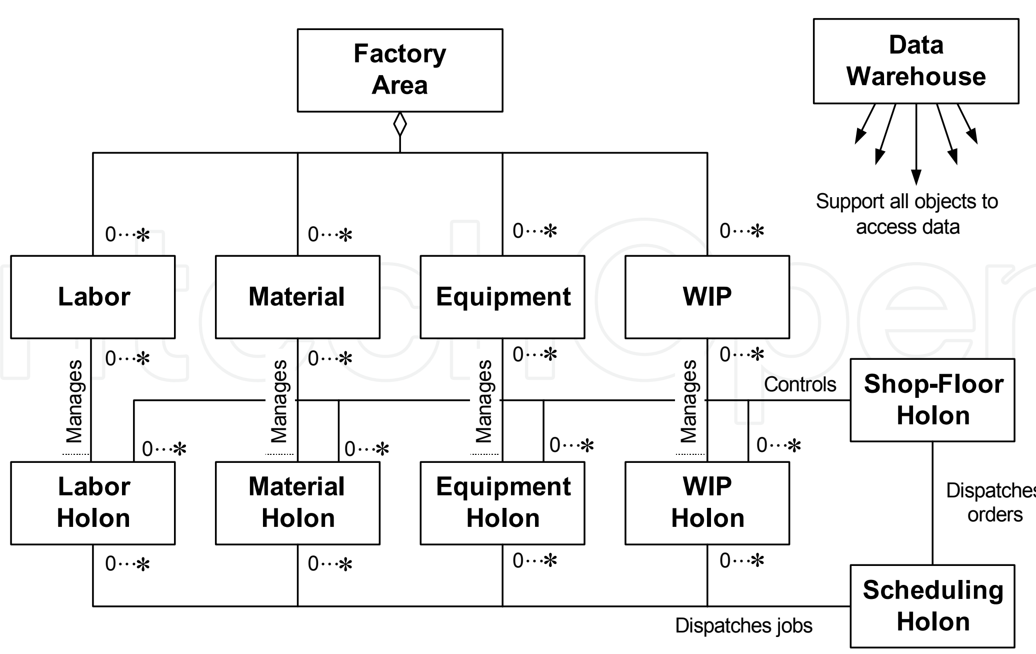
Seven steps are included in the holarchy design stage. They are explained below.

4.1 Constructing an Abstract Object Model

A typical deployment diagram for HMES is shown in Fig. 1. It is well known that MES is composed of several functional modules that handle specifics, e.g. material, equipment, labor, and planning (MacDonald, 1993). The abstract object model is constructed as in Fig. 3(a) (Cheng et al., 1999).
The four key elements of a factory are labor, material, equipment, and work-in-process (WIP). Each element is managed by its specific managing holon. All four of these managing holons are controlled by the Shop-Floor Holon. The Shop-Floor Holon also dispatches orders to the Scheduling Holon. The Scheduling Holon dispatches jobs to the Labor Holon, Material Holon, Equipment Holon, and WIP Holon.

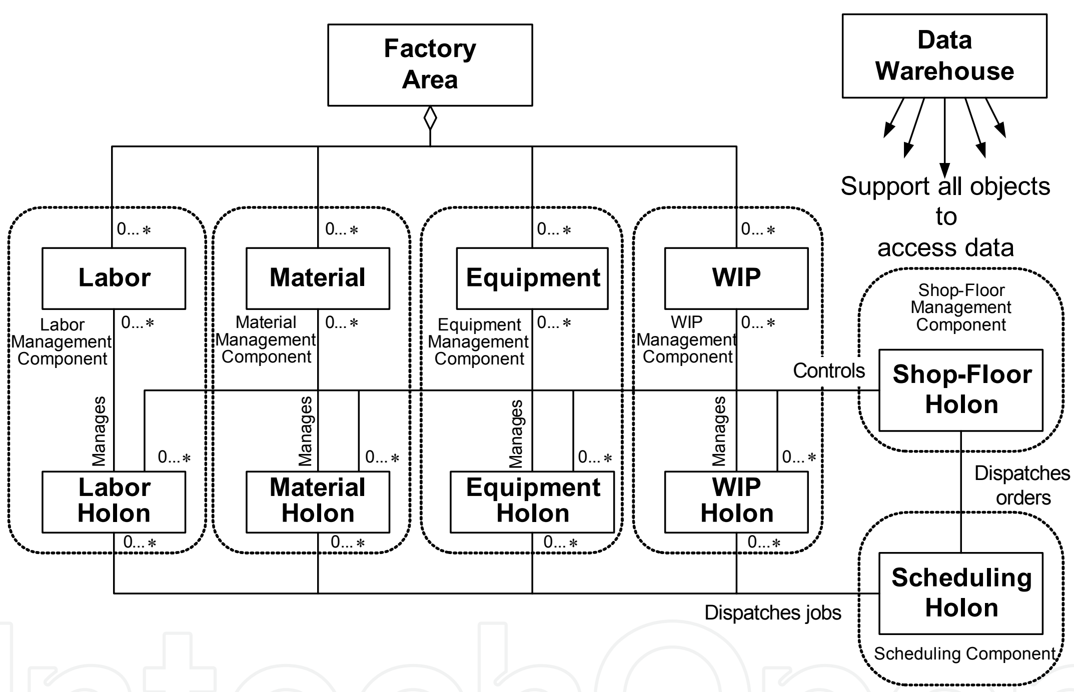
4.2 Partitioning Application Domain into Components

To design a distributed and integratable MES, its application domain is partitioned systematically as depicted in Fig. 3(b). In addition to the data warehouse, the system is divided into six components.



(a) Abstract Object Model

They are labor management, material management, equipment management, WIP management, scheduling, and shop-floor management components. The labor, material, equipment, and WIP management components handle labor, movements of materials, process equipment, and WIP tracking, respectively. The scheduling component takes care of scheduling and dispatching tasks of the system. The shop-floor management component is in charge of system-level services and management, i.e., order management, life-cycle services, collection services, and query services. Each management component has a specific functional holon, which serves as the manager of that specific management component



(b) Partitioning Application Domain into Components

Figure 3. Object Model of an HMES

As mentioned previously, each management component needs a specific functional holon to serve as the manager of that component.

4.3 Identifying Generic Functions among Components

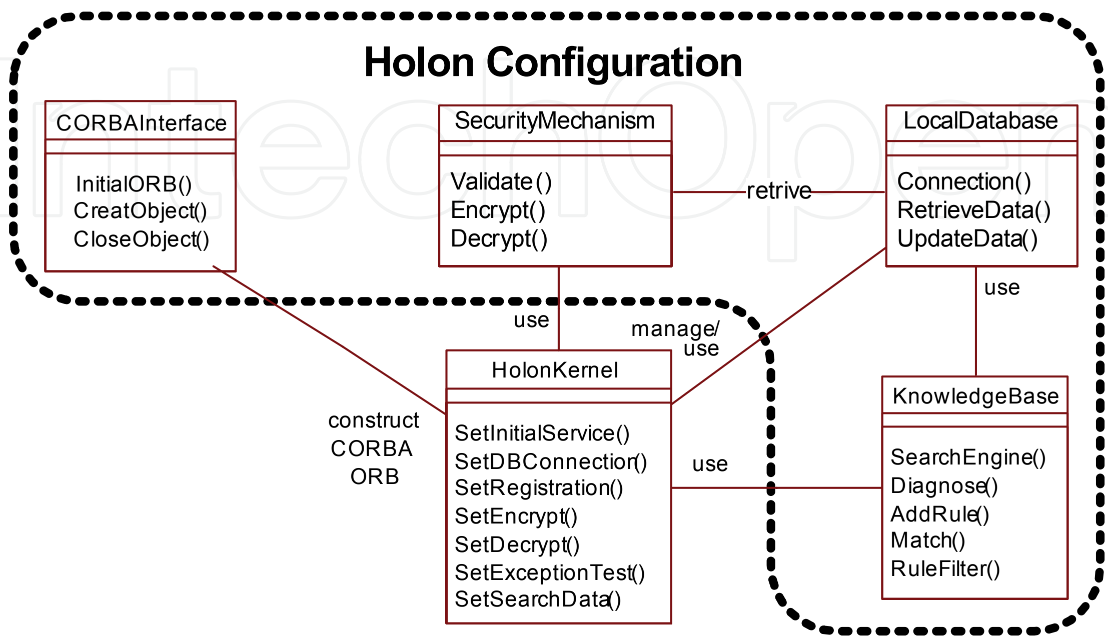
The purpose of this paper is to apply the concepts of holon and holarchy to design the HMES Holarchy and functional holons that not only possesses the properties of the MES Framework (Cheng et al., 1999) but also has the properties of failure recovery and security certification. Therefore, based on the principle of software reuse (Chen and Chen, 1994; Cheng et al., 1999), the Generic Holon which handles the generic functions of functional holons shall first be devised. After judicious consideration, the authors conclude that in addition to the communication infrastructure, the Generic Holon shall possess security mechanisms, search mechanisms, and intelligence mechanisms to deal with the generic functions that emphasize failure recovery and security certification.

4.4 Developing Generic Holon

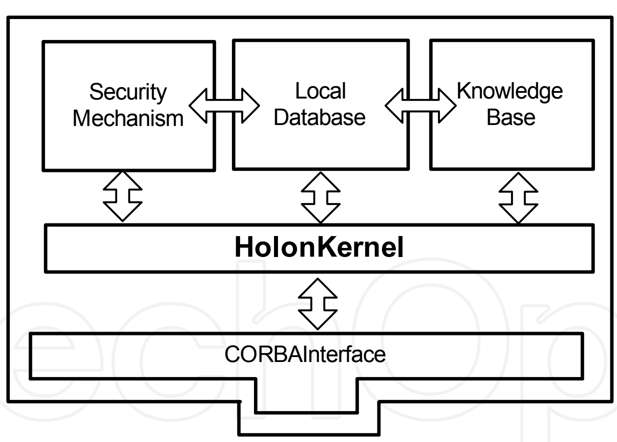
The requirements for developing the Generic Holon are:

a) It can construct the communication infrastructure for communication, collaboration, and extensibility purposes.
b) It provides the intelligence mechanism for exception diagnosis.
c) It provides the search mechanism for collaboration and reconfigurability.
d) It provides the security mechanism for security check and encryption / decryption.
e) It provides the ability to establish database services for information storage / retrieval.

According to these requirements and following the development procedure for object-oriented systems (Eriksson and Penker, 1998; Huang et al., 1999; Cheng et al., 2002), the Generic Holon's class diagram and internal architecture is obtained as shown in Fig. 4. For further illustration, please refer to (Lin, 2000; Chang, 2000) for the detailed designs of the Generic Holon.
Observing Fig. 4(a), the basic structure of the class diagram is HolonKernel manages/uses HolonConfiguration that consists of CORBAInterface, SecurityMechanism, LocalDatabase, and KnowledgeBase. By inheriting HolonKernel, a functional holon can possess all the characteristics of the Generic Holon. CORBAInterface is designed for constructing a communication infrastructure and achieves the collaboration platform. In order to establish secure communication, the SecurityMechanism is created for handling all the operations of security. KnowledgeBase constructs a search engine for searching desired services and a reasoning mechanism for exception diagnosis. The LocalDatabase sets the connection of database for SecurityMechanism and KnowledgeBase to access the database. On the other hand, the internal architecture of the Generic Holon is depicted in Fig. 4(b).



(a) Class Diagram



(b) Internal Architecture

Figure 4. Class Diagram and Internal Architecture of Generic Holon

Observing Fig. 4(b), the Generic Holon owns HolonKernel to communicate with other holons by CORBAInterface. Using LocalDatabase, the Generic Holon can maintain autonomous properties and necessary information. SecurityMechanism can retrieve the related information through LocalDatabase and then check user's authorization for security certification. The intelligence mechanism for exception diagnosis purposes of the Generic Holon is mainly considered in knowledgeBase that also needs the support of LocalDatabase. After completing the design of the Generic Holon, any functional holon can be designed by inheriting Generic Holon to obtain generic properties of holon and then adding the specific functions of that functional holon.

4.5 Defining Holarchy Messages

After partitioning the application domain into components, we need to define holarchy messages among all the functional holons so that interoperability and collaboration among all the functional holons are enabled. According to Fig. 1 and Fig. 3(b), the holarchy messages of HMES are defined as in Fig. 5.

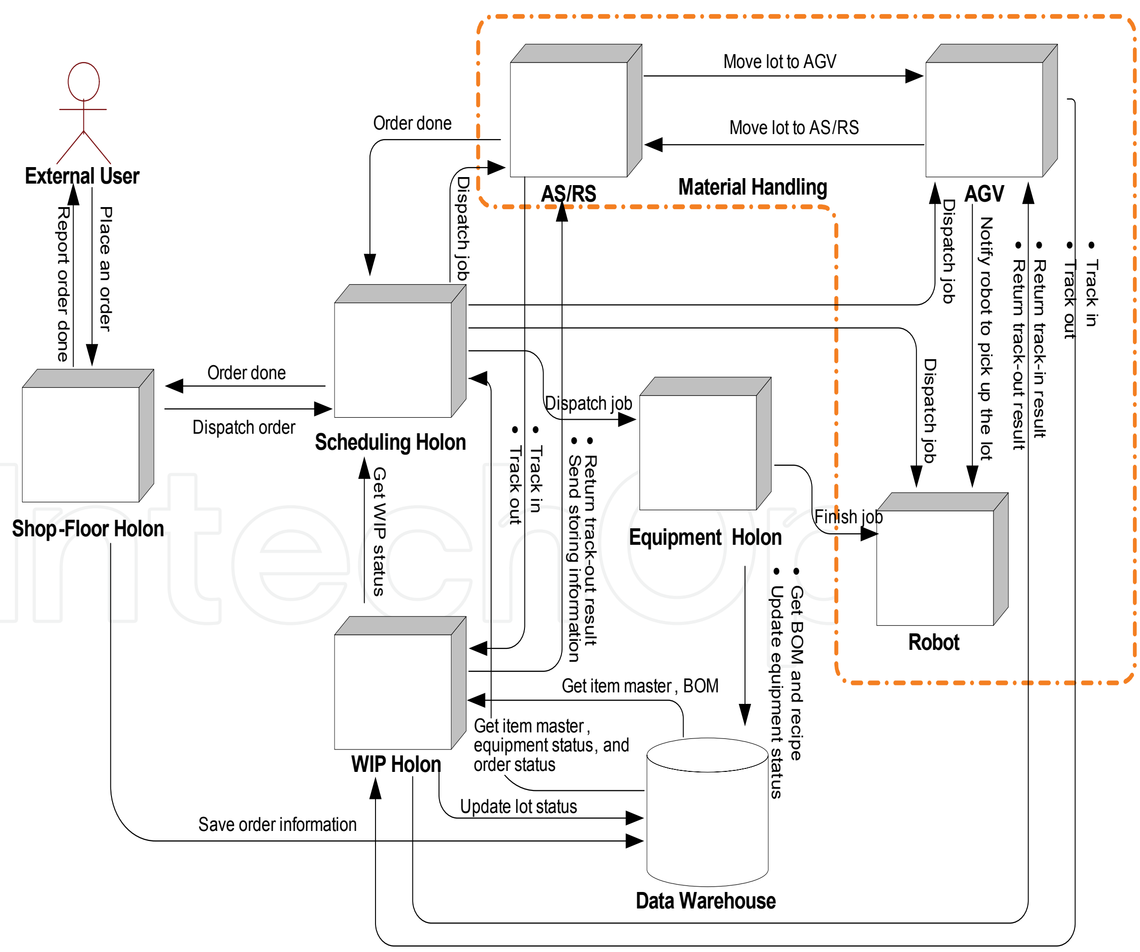


Figure 5. Defining Holarchy Messages

The Shop-Floor Holon receives a place an order message from an external user and the Shop-Floor Holon will reply report order done when the order is done. Based on the received order, the Shop-Floor Holon will send dispatch order to the Scheduling Holon and the Scheduling Holon will reply order done if the order is finished. The Shop-Floor Holon sends save order information to the Data Warehouse to save all the order information. Similarly, the interfacing holarchy messages of Scheduling Holon, WIP Holon, Equipment Holon, Data Warehouse, and Material Handling (which includes AS/RS, AGV, and robot) can be defined as shown in Fig. 5.

4.6 Defining Holarchy Framework of Holonic Manufacturing Execution Systems

After the development of the Generic Holon and holarchy messages, we are ready to define the holarchy framework of HMES (or HMES Holarchy in short).

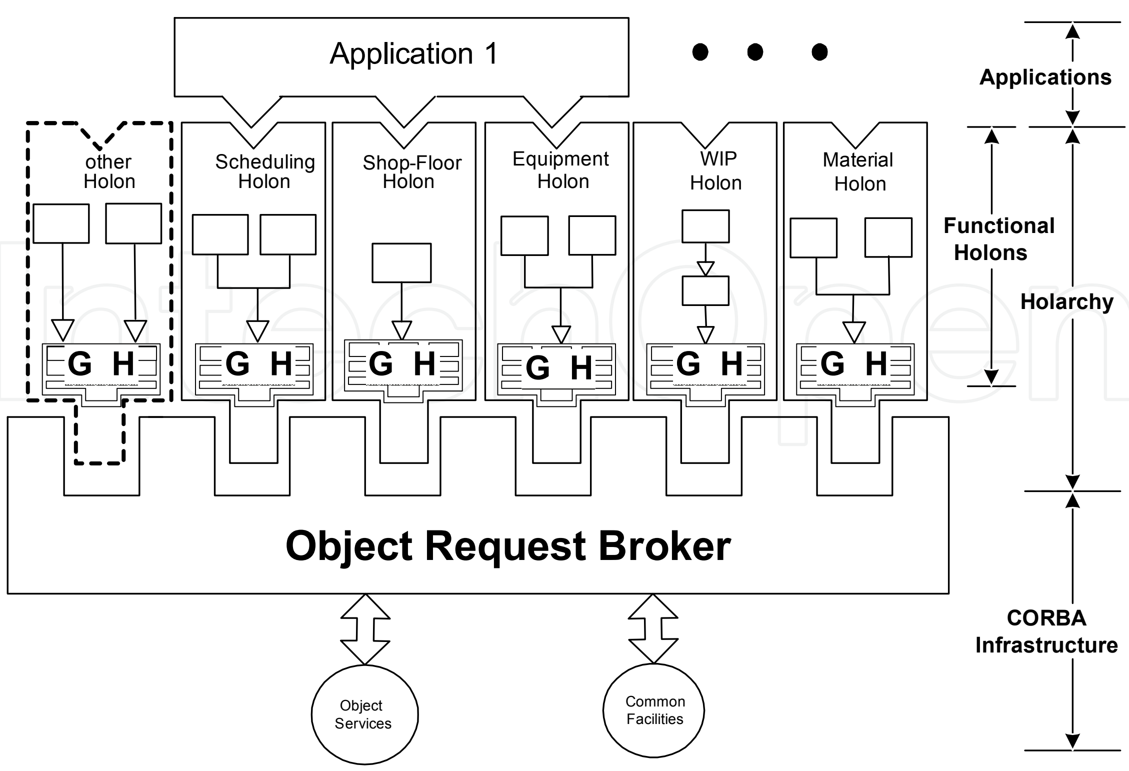


Figure 6. Holarchy Framework of Holonic Manufacturing Execution Systems

The HMES Holarchy is illustrated in Fig. 6 which utilizes CORBA infrastructure (Orfali et al., 1996; OMG, 1998) as the system's communication backbone. Every functional holon shall inherit the Generic Holon so as to possess the properties of a holon as well as the capabilities of failure recovery and security certification. Then, specific functions of each functional holon can be added individually to become a specific functional holon. The holarchy messages of each functional holon can be specified by CORBA IDL (Interface Definition Language) (Orfali et al., 1996; OMG, 1998). Therefore, each functional holon can be integrated into the HMES Holarchy in a plug-and-play fashion.

This HMES Holarchy is expandable. As illustrated on the left side of Fig. 6, other functional holon may also be integrated into the HMES Holarchy if this functional holon inherits the Generic Holon and defines the functional holon's CORBA IDL by the expanded holarchy messages. Finally, applications of the HMES can be easily constructed by invoking the related functional holons as depicted on top of Fig. 6.

4.7 Designing Functional Holons

The final step of holarchy design is to design various functional holons based on the Generic Holon. As mentioned in the previous sub-section, with the HMES Holarchy architecture, it becomes straightforward to design a functional holon by simply inheriting the Generic Holon, adding the functional holon's specific function, and defining its IDL based on the system's holarchy messages. In the following section, the WIP holon is selected as the example to elaborate the design procedure of a functional holon.

5. WIP Holon Design

The functional requirements for WIP holons are:

a) It manages the life cycle of WIP objects.
b) It performs track-in and track-out operations and updates the corresponding WIP information in real-time.
c) It provides WIP information to users and other holons.
d) Its interfaces are in compliance with the HMES Holarchy.
e) It possesses the capabilities of exception recovery and security certification.

Requirements (a) to (c) are the specific functions of WIP holons while Requirements (d) and (e) are the common requirements for the components of HMES Holarchy. It is natural to develop the WIP Holon by inheriting the Generic Holon first to take care of Requirements (d) and (e) and then considering the specific requirements (a) to (c). Based on the above design principle and following the development procedure for object-oriented systems (Eriksson and Penker, 1998; Huang et al., 1999), the class diagram of the WIP Holon is designed and shown in Fig. 7.

The upper portion of Fig. 7 is the Generic Holon that has been designed and illustrated in Fig. 4(a). WIPManager, which is the primary role of the entire WIP Holon, inherits the Generic Holon to accomplish Requirements (d) and (e).
WIPManager uses RecoveryManager to perform specific recovery operations. WIPManager also manages the life cycle of WIP objects and is in charge of track-in and track-out operations of all the WIP. A new WIP object is created when a new lot arrives. The WIP object contains its own specific attributes such as LotID, BOM, and ItemMaster, etc. A WIP object also performs its own Trackin() Trackout() operations and invokes NewVariables() methods of BOM and ItemMaster to obtain the associated production information. UserInterface provides the necessary operations for external users to interface with the WIP Holon.

Observing Fig. 7, the + sign before an operation means the operation is public, and the – sign stands for private. In the WIPManager, public operations stand for the IDL of the system; while in the UserInterface, public operations indicate the available functions for external users.

State diagrams show all possible states and transactions of a system. A change of state caused by an event is called a transition. Figure 8(a) illustrates the states and transitions of the WIP Holon. Please refer to Fig. 7 and Fig. 8 when reading the following explanation.

A user initiates the WIP Holon by invoking the Login() operation of UserInteface. If he passes the security certification, the WIP Holon will activate CORBA services by calling SetInitialService()of HolonKernel. Then, the system is ready to receive WIP object's creating commands.

In fact, the major functions of the WIP holon are how to trace and manage WIP. We define WIP to be temporal objects, as such they have life cycles. Figure 8(b) is the state diagram of WIP life cycle.

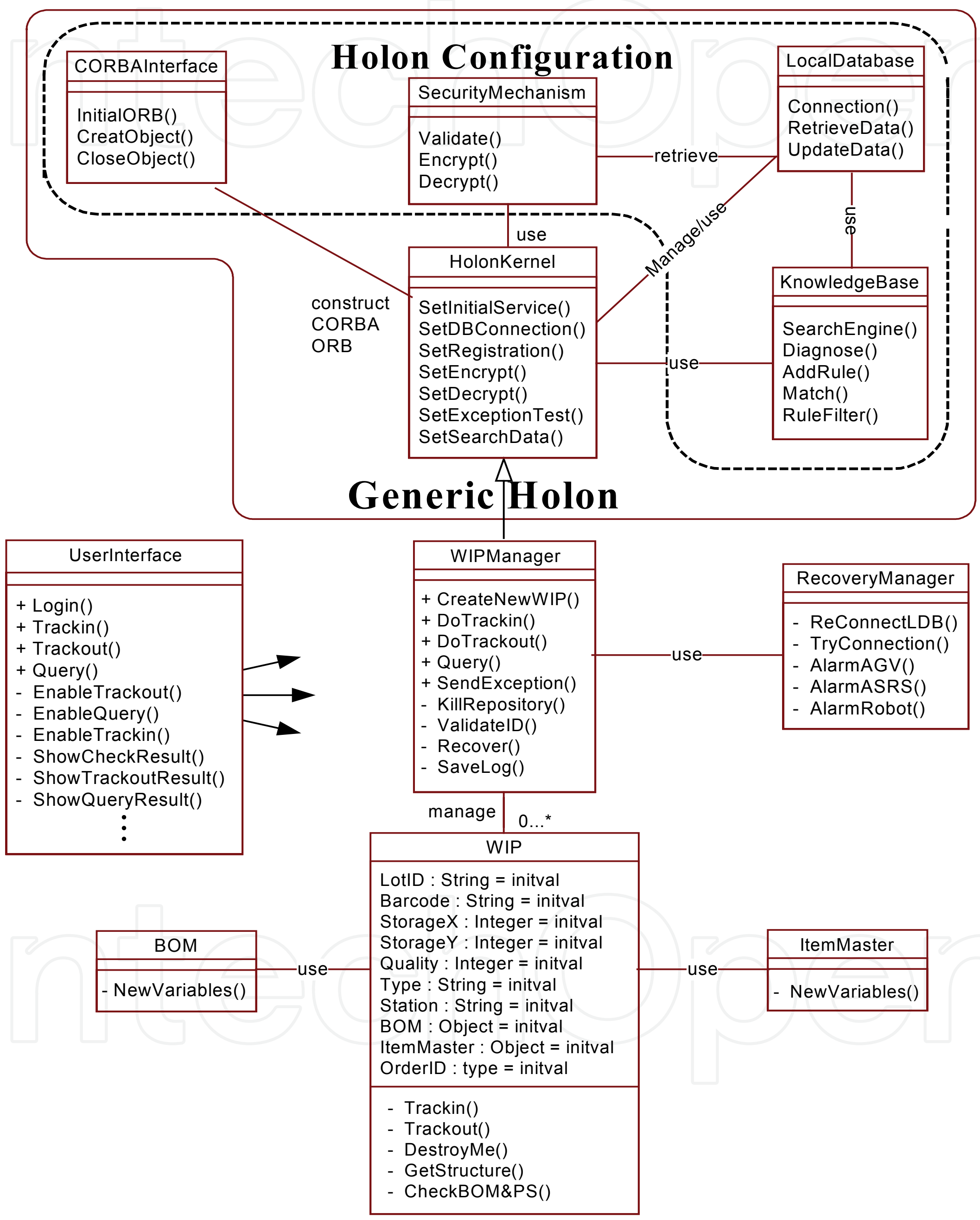


Figure 7. Class Diagram of WIP Holon

When WIPManager gets the message CreateNewWIP() from the Scheduling Holon, a new WIP object is generated based on the data transferred from the Scheduling Holon. WIP object uses NewVariables() operation in BOM to get the contents of BOM. WIP object uses the same approach to obtain ItemMaster information. Then, WIP object gets order status and saves it. Up to this point, initialization of WIP object is completed and it enters Wait for request state.

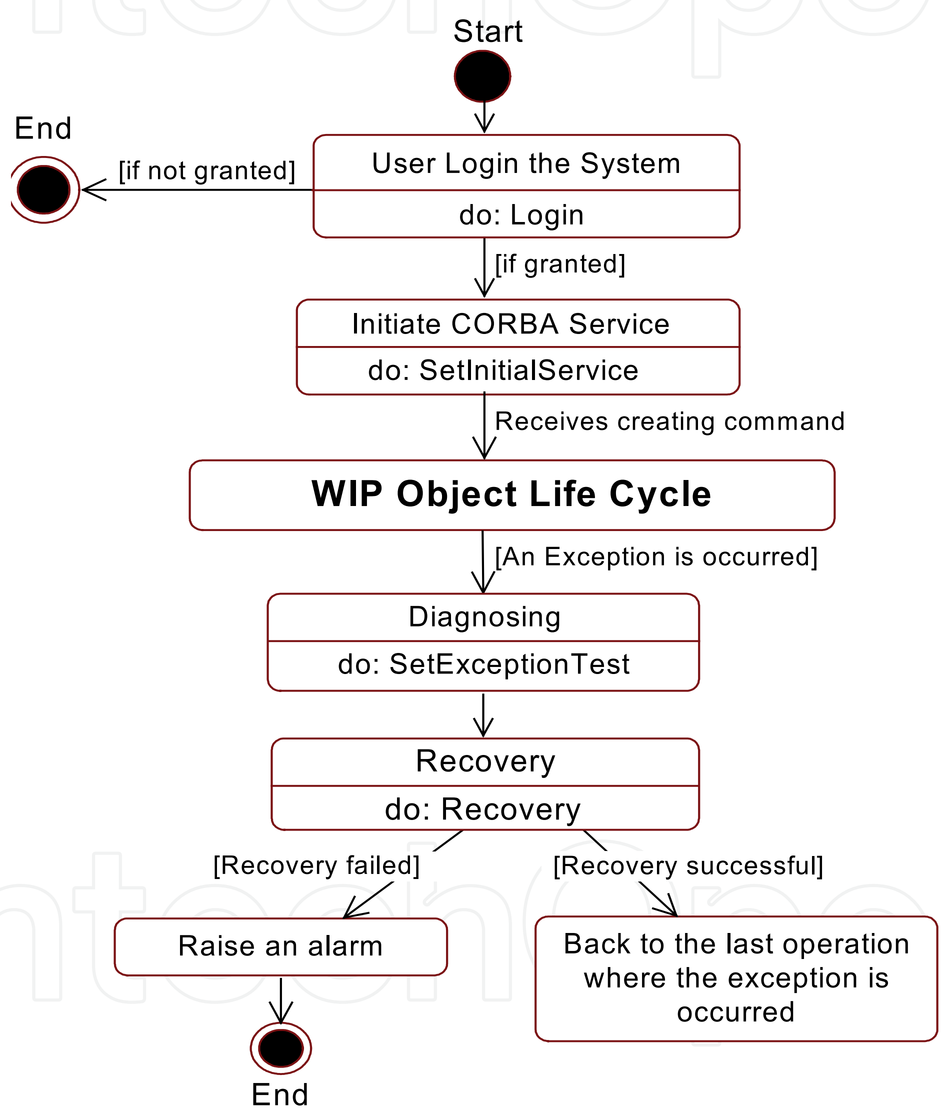
At Wait for request state, the WIP object can take commands, such as track-in, track-out, and query. The query request will bring the WIP object to the Provide WIP status state and the WIP status is then sent to the requester. Track-out and track-in commands will update the WIP status and store it to database. During track-in operation, the WIP object will check if this current process sequence is the last one or not. If it is not, just jumps back to Wait for request state. If it is the last process, this WIP object will be deleted and the memory will be released. It thus completes the life cycle of a WIP object.

Note that, the initial Generic Holon architecture shown in Fig. 4 only specifies the generic skeleton of the intelligence mechanism that consists of KnowledgeBase and LocalDatabase. After inheriting the Generic Holon to become a part of the WIP Holon, its KnowledgeBase and LocalDatabase shall be trained to contain the specific knowledge, information, and rules for WIP holon's exception-diagnosis usage only.

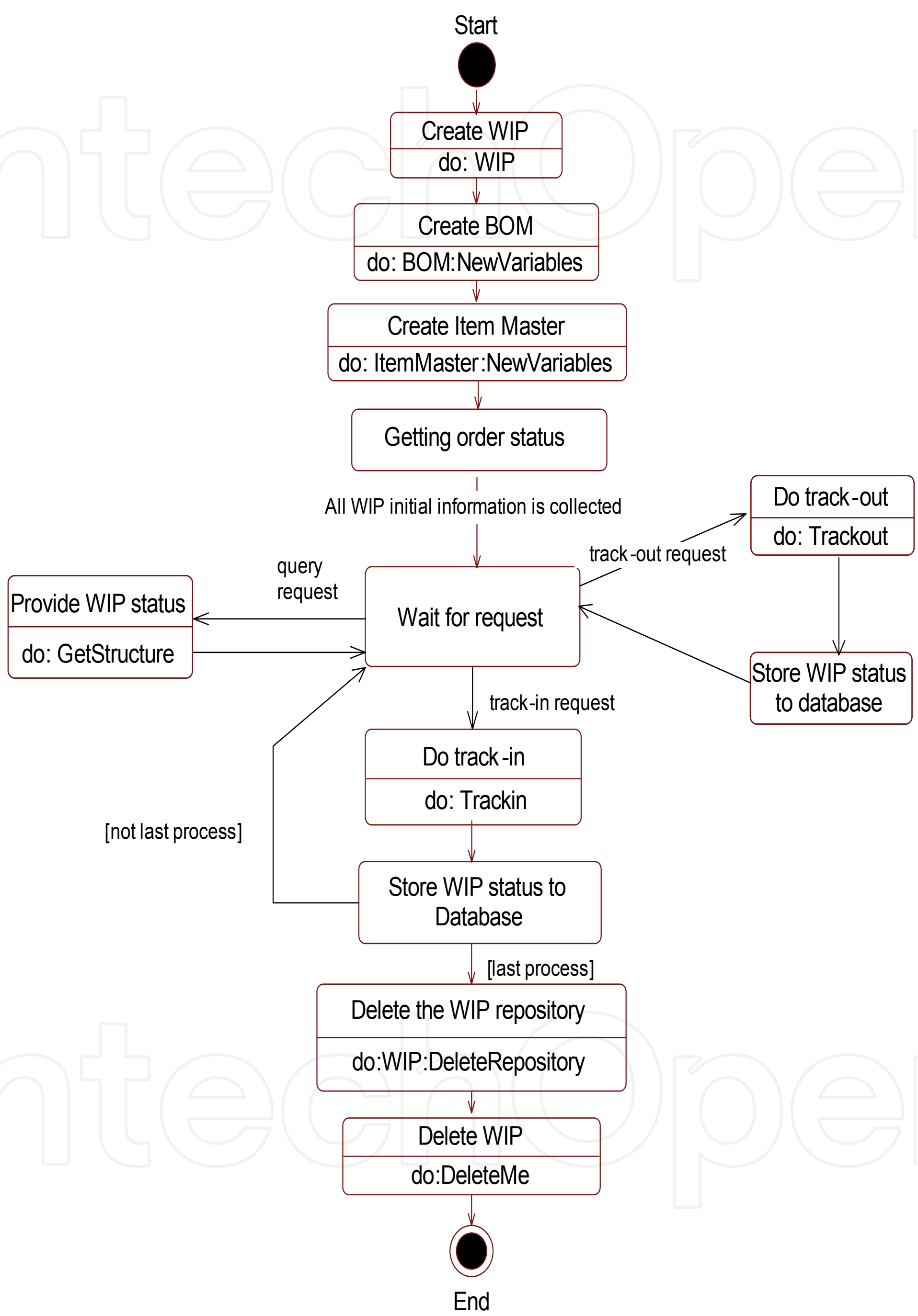
Now, observing Fig. 8(a), if an exception is occurred and detected during the WIP management process, the system will enter the Diagnosing state that invokes SetExceptionTest() of HolonKernel to diagnose the exception.

If the cause is identified by the intelligence mechanism of the Generic Holon, the system will enter the Recovery state that invokes the associated recovery operation implemented in RecoveryManager. If the recovery operation is successful, the system will jump back to the last operational state where the exception was occurred, otherwise the system will raise an alarm and then stop.

After demonstrating how to design functional holons, the holarchy design stage is completed. The following section will explain the application construction and system integration stages.



(a) Entire WIP Holon State Diagram



(b) WIP Object Life Cycle State Diagram

Figure 8. State Diagrams of WIP Holon

6. Application Construction and System Integration

As depicted in Fig. 2, the last two stages are application construction and system integration. Observing the top of Fig. 6, with the advantage of HMES Holarchy, it is obvious that applications can be constructed by invoking operations of associated holons. These holons will cooperate with one another by following the holarchy messages defined in Fig. 5. This meets the characteristics of holon and holarchy. In fact, the deployment diagram, holarchy messages, and a holarchy framework as shown in Figs. 1, 5, and 6, respectively, have been successfully implemented and running at the Factory Automation Laboratory of the Institute of Manufacturing Engineering, National Cheng Kung University, Tainan, Taiwan, Republic of China.

7. Comparisons among Legacy MES, Framework MES, and Holonic MES

The concepts and/or technologies of OOAD, component software, framework, holon, holarchy, security certification, and failure recovery have been taken into account for developing HMES. In this section, characteristic comparisons between Legacy MES, Framework MES, and Holonic MES are presented.

	Legacy MES	Framework MES	Holonic MES
Architecture	Centralization	Distributed OO	Holarchy
Open Interfaces	No	Yes	Yes
Modularization	Low	High	High
Interoperability	Low	High	High
Configurability	Low	High	High
Maintainability	Difficult	Easy	Easy
Security Certification	No	No	Yes
Failure Recovery	No	No	Yes

Table 1. Comparisons between Traditional MES, Framework MES, and Holonic MES

As indicated in Table 1, Legacy MES refers to the commercial products such as Promis, WorkStream, and Poseidon. Framework MES stands for Encore, SiView, and FACTORYWorks. Detailed comparisons are presented below.

7.1. Architecture

Concerning architecture, Legacy MES is a centralized system. All the computations and operations are executed in one large-scale mainframe computer. Framework MES belongs to distributed object-oriented systems that divide all the functions into individual various models. The computations and operations are also distributed into each model. In this way, Framework MES lowers the loading of each mainframe and increases the reliability of the system. Also, Framework MES avoids the malfunction of the entire system due to the breakdown of a single module. Holonic MES is designed with the concepts of holon and holarchy. It has the advantages of distributed object-oriented systems, and also the characteristics of intelligence, autonomy, coordination, and collaboration. Thus, Holonic MES's adaptability can meet the requirements and trends of future manufacturing systems.

7.2. Open Interfaces

When considering interfaces, Legacy MES is a closed system while Framework MES and Holonic MES are open systems. Systems with open interfaces have the advantage of being easy to cooperate and link with other related modules or systems.

7.3. Modularization

Modular design is very important to system software development. With component software, users can apply proper modules based on needs. This is beneficial both for design and maintenance. Both Framework MES and Holonic MES utilize modular design but Legacy MES does not.

7.4. Interoperability

A distributed object-oriented system usually has many functional modules that they need to interoperate with one another. Framework MES and Holonic MES are distributed object-oriented systems, therefore their interoperability with distributed modules is both essential and profuse.

7.5. Configurability

Configurability is important for a manufacturing system to deal with a dynamic, varying and rapidly changing environment. Framework MES and Holonic MES are easier to reconfigure than Legacy MES.

7.6. Maintainability

For Legacy MES, it is not easy to repair and maintain since it is a large-scale and centralized system. For Framework MES and Holonic MES, their maintenance is easier because they are distributed systems and each component of the systems can operate alone and be maintained separately.
7.7. Security Certification
The problem of security is becoming more and more serious. In Holonic MES, the ability of security certification is embedded in the design of the Generic Holon so that it is natural for all the functional holons to possess the capability of security certification.

7.8. Failure Recovery

Reliability is always the most important issue for automatic manufacturing systems. Once there is an exceptional condition that causes the entire production line to shutdown, the loss is beyond evaluation. As a result, a good set of MES needs a failure recovery mechanism so as to minimize the loss caused by occurrences of exceptional conditions. Among those three MES types, only Holonic MES incorporates the capability of failure recovery into the design.

8. Summary and Conclusions

Based on the characteristics of holon and holarchy and by applying distributed object-oriented techniques, this paper proposes a systematic approach for developing Holonic Manufacturing Execution Systems (HMES) with security-certification and failure-recovery considerations. The basic foundations required for developing HMES possessing characteristics of holon and holarchy are summarized. The HMES development procedure that consists of system analysis, holarchy design, application construction, and system integration and testing stages are proposed. Among these stages, holarchy design is the most important and consists of seven steps: (a) constructing an abstract object model, (b) partitioning the application domain into components, (c) identifying generic functions among the components, (d) developing the Generic Holon, (e) defining holarchy messages, (f) defining the holarchy framework, and (g) designing functional holons. WIP Holon, as an example of a functional holon, is developed for demonstration purposes. Comparisons between Legacy MES, Framework MES, and Holonic MES are made. It reveals that this systematic approach provides a new concept for developing next generation manufacturing execution systems.

Acknowledgments

The authors would like to thank the National Science Council of the Republic of China for financially supporting this research under contracts No. NSC-89-2212-E006-094, NSC-90-2212-E006-026, and NSC-91-2212-E006-062.

9. References

Chang, C.-F. (2000). Development of scheduling holons and WIP holons, Master Thesis of the Institute of Manufacturing Engineering, National Cheng Kung University

Chen, D. J. & Chen, D. T. K. (1994). An experimental study of using reusable software design frameworks to achieve software reuse, Journal of Object Oriented Programming, pp. 56-67

Cheng, F.-T., Shen, E., Deng, J.-Y. & Nguyen, K. (1999). Development of a system framework for the Computer-Integrated Manufacturing Execution System: a Distributed Object-Oriented Approach, International Journal of Computer Integrated Manufacturing, Vol. 12(5), pp. 384-402

Cheng, F.-T., Yang, H.-C. & Huang, E. (2002). Development of an educational supply chain information system using object web technology. Journal of the Chinese Institute of Engineers, Vol. 25(6), pp. 735-752

Eriksson, H.-E. & Penker, M. (1998). UML Toolkit. New York: John Willy & Sons, Inc.

Gamma, E., Helm, R., Johnson, R. & Vlissides, J. (1995). Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley, Greenwich, CT

Hino, R. & Moriwaki, T. (1999). Decentralized Scheduling in Holonic Manufacturing Systems, Proceedings of the Second International Workshop on Intelligent Manufacturing Systems, Leuven, Belgium, pp. 41-47

Holonic Manufacturing System. HMS Introduction and Overview, http://hms.ifw.uni-hannover.de/

Huang, E., Cheng, F.-T. & Yang, H.-C. (1999). Development of a collaborative and event-driven supply chain information system using mobile object technology, in Proceedings of the 1999 IEEE International Conference on Robotics and Automation, Detroit, Michigan, U.S.A., pp. 1776-1781

Kadar, B., Monostori, L.& Szelke, E. (1998). An object-oriented framework for developing distributed manufacturing architectures, Journal of Intelligent Manufacturing, Vol. 9, pp. 73-179

Lin J.-Y. (2000). The development of holonic information coordination systems with security mechanism and error-recovery capability, Master Thesis of the Institute of Manufacturing Engineering, National Cheng Kung University

MacDonald, A. (1993). MESs help drive competitive gains in discrete industries, Instrumentation & Control Systems, pp. 69-72

Markus, A., Vancza, T. Kis & Monostori, L. (1996). A Market Approach to Holonic Manufacturing, Annals of the CIRP, Vol. 45(1), pp. 433-436

McGehee, J., Hebley, J. & Mahaffey, J. (1994). The MMST computer-integrated manufacturing system framework, IEEE Transactions on Semiconductor Manufacturing, Vol. 7(2), pp. 107-115

MESA International (1997). MESA International - White Paper Number 6: MES Explained: A High Level Vision, MESA International, Pittsburgh, PA

Mowbray, Z. (1995). The Essential CORBA: Systems Integration Using Distributed Objects, ISBN0-471-10611-9, New York: John Wiley & Sons

Nguyen, K. (1996). Flexible computer integrated manufacturing systems, in Proceedings of the SEMICON Taiwan 96 IC Seminar, Taipei, Taiwan, R.O.C., pp. 241-247

OMG (1998) The Common Object Request Broker: Architecture and Specification, Revision 2.2. Object Management Group

Orfali, R., Harkey, D. & Edwards, J. (1996). The Essential Distributed Objects Survival Guide, New York: John Willy & Sons

Ramos, C. (1996). A Holonic Approach for Task Scheduling in Manufacturing Syatems, Proceedings of the IEEE International Conference on Robotics and Automation, Minneapolis, U.S.A., pp. 2511-2516

Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F. & Lorensen, W. (1991). Object-Oriented Modeling and Design, Englewood Cliffs, NJ: Prentice-Hall

Samanich, N. J. (1993). Understand your requirements before choosing an MES, Manufacturing Systems, pp. 34-39

Scott, D. (1996). Comparative advantage through manufacturing execution systems, in Proceedings of the SEMICON Taiwan 96 IC Seminar, Taipei, Taiwan, R.O.C., pp. 227-236

SEMATECH (1998). Computer Integrated Manufacturing (CIM) Framework Specification 2.0 SEMATECH, Austin, Texas

Sparks, S., Benner, K. & Faris, C. (1996). Managing object-oriented framework reuse, IEEE Computer, pp. 52-61

Tonshoff, H. K., Winkler, M., and Aurich, J. C. (1994). Product modeling for holonic manufacturing systems, in Rensselaer's 4th International Conference on Computer Integrated Manufacturing and Automation Technology

Ulieru, M. (1997). Soft computing issues in the intelligent control of holonic manufacturing systems, in Proceedings of the 1997 IEEE Annual Meeting of the North American Fuzzy Information Proceeding Society, NAFIPS'97, Syracuse NY, USA, pp. 323-328

Valckenaers, P., Bonneville, F., Brussel, H. V., Bongaerts, L. & Wyns, J. (1994). Results of the holonic control system benchmark at KULeuven, in Rensselaer's 4th International Conference on Computer Integrated Manufacturing and Automation Technology

Van Leeuwen; E. H. & Norrie, D. (1997). Holons and holarchies, Manufacturing Engineerings, pp. 86-88

Manufacturing the Future

Edited by Vedran Kordic, Aleksandar Lazinica and Munir Merdan

ISBN 3-86611-198-3

Hard cover, 908 pages

Publisher Pro Literatur Verlag, Germany / ARS, Austria

Published online 01, July, 2006

Published in print edition July, 2006