A Security Framework for Collaborative Distributed System Control at the Device-Level

Xu, Y.; Korba, Larry; Wang, L.; Hao, Q.; Shen, W.; Lang, S.

National Research Council Canada
Conseil national de recherches Canada

Institute for Information Technology
Institut de technologie de l'information

NRC · CNRC

A Security Framework for Collaborative Distributed System Control at the Device-Level *

Xu, Y., Korba, L., Wang, L., Hao, Q., Shen, W., Lang, S.
August 2003

* published in the Proceedings of the 1st IEEE Conference on Industrial Informatics. Banff, Alberta, Canada. August 2003. NRC 46501.

Canada

A Security Framework for Collaborative Distributed System Control at the Device-Level

Yuefei Xu, Larry Korba
Institute for Information Technology
National Research Council Canada, Ottawa, ON, Canada
[Yuefei.Xu | Larry.Korba] @nrc.gc.ca

Lihui Wang, Qi Hao, Weiming Shen, Sherman Lang
Integrated Manufacturing Technology Institute
National Research Council Canada, London, ON, Canada
[Lihui.Wang | Qi.Hao | Weiming.Shen | Sherman.lang] @nrc.gc.ca

Abstract

In today's globalized business world, outsourcing, joint ventures, and cross-border collaborations have led to work environments that are geographically distributed across organizational and national boundaries. There are critical research needs to develop highly secured collaborative work environments and security solutions for deployment, configuration, monitoring, and device control of interoperating services. This paper presents a well-shaped security framework for distributed system control with a focus on device-level system control, monitoring and services re-configuration in open and dynamic environments. The characteristics of portability, reconfigurability, interoperability, and interchangeability of these new environments are considered as key factors to produce new security risks and challenges. By adopting Public Key cryptography, software agent and XML binding technologies, the major security problems of authenticity, integrity, confidentiality, and safe execution are addressed in this framework. The core modules for secure task delivery and execution are presented in detail.

Keywords: *Security Framework, Computer-Supported Cooperative Work; Software Portability, Interoperability, Reconfigurability, Interchangeability, Collaborative Distributed System Control; IEC61499*

1. Introduction

With the growing globalization and decentralization of businesses, the boundary between what is "inside" and "outside" of an organization is blurring. Businesses and interaction are now happening across traditional physical boundaries. The decentralization of organizations has become a major impact to the traditional business model. Services and resources are distributed everywhere and sourced anywhere through global supply chains. For example, in the area of e-manufacturing, product design, processing planning and manufacturing have shifted rapidly from within one factory to global networks. To cope with this trend, a collaborative environment with interactive design, scheduling, monitoring, and control capabilities is essential for any factory to increase its competitiveness and profitability.

However, how to keep all activities under control in an open and dynamic environment is still a very challenging question. The challenge is much larger when one considers the distributed low-level tasks, services, machines, devices and processes involved in such a system. Even for modern factories with PLCs (Programmable Logical Controllers), their status and processes are all kept in closed environments and are separated from outside networks. As well their status and operations are hard to predicate and control from a remote site. This situation has created a barrier to forming new collaborations with the moving global supply chains and other activities. Recently, there are strong needs in industry to add portability, operability, configurability, and other features to current industrial control systems, so that: 1) control tasks or components can be designed and exchanged between different vendors; 2) different devices can be operated, monitored and communicated with outside and each other; 3) different devices can be re-configured remotely to respond unanticipated events.

In the past few years, a number of research projects have been formed to address these problems. Significant projects include *NIIIP* in the USA [1]. The goal of this project is to develop open industry software protocols that can make software interoperation possible between manufacturers and their suppliers. The latest *Cimplicity* from GE Fanuc Automation (USA) allows users to view their factory's operational processes through an XML-based *WebView* screen, including all alarms on every *Cimplicity* system on the network [2]. To bring legacy machine tools on-line, e-Manufacturing Network Inc. (Canada) introduced its *ION Universal Interface* and *CORTEX Gateway*. In 1999, Hitachi Seiki (Japan) introduced *FlexLink* to its turning and machining centers, making possible to do in-process gauging, machine monitoring, and cycle-time analysis. Since 1998, Mazak (Japan) has operated its high-tech *Cyber Factory* concept at its headquarters in Oguchi, Japan. The fully networkable *Mazatrol Fusion controls* allow Mazak machines to communicate over wireless networks for applications including real-time machine tool monitoring and diagnostics. *MetaMorph II* [3] introduced a hybrid agent-based mediator-centric architecture to integrate partners, suppliers, and customers in a dynamic manufacturing environment. At the same time, the Internet and World Wide Web have been widely used as a medium for exchanging information and are expanding to industrial control areas.

Despite all these accomplishments, the available systems are either for off-line simulation or for monitoring only. Most systems require a specific application to be installed instead of a standard interface, like a web browser. The requirement of specific application has limited the systems' portability. Advanced system design, scheduling, and execution functionality remains isolated from the collaborative processes. To be more competitive, users are now demanding integrated solutions for these requirements.

More seriously, with system control and information processing continuing to move towards open, reconfigurable and interactive environments, existing traditional security methods, like user name, password approach are not sufficient. Portability, configurability, interoperability, and interchangeability produce new security risks and challenges. The security mechanisms that are fully compatible with the demands of open and dynamic situations are critical for the safety and the functionality of collaborative systems.

This paper explores the security mechanisms for collaborative distributed systems control in open and dynamic environments. In section 2, we analyze the distributed system control model and respective security requirements. In section 3, a security framework for collaborative distributed control is proposed to address how distributed devices can be accessed securely by mobile tasks, which travel within untrustworthy networks. Section 4 presents the core modules of Security Control Gateway for meeting the security challenges of authenticity, integrity, confidentiality, and safe execution. Section 5 presentes the security mechanisms of two schemes. Before the conclusions in Section 7, we outline our implementation considerations in Section 6.

2. Distributed System Control Models

As we mentioned in Section 1, there have been many efforts towards increasing the portability, interoperability, configurability, interchangeability and other collaborative features of new distributed systems. Most significantly, the International Electro-technical Commission (IEC) Function Block (FB) specification (IEC 61499) is an effort to standardize these efforts. This specification provides an architectural and modeling approach for distributed Industrial Process Measurement and Control Systems (IPMCS) [4]. It also offers a series of reference models to cover the whole control system life cycle, including system planning, design, implementation, validation, operation and maintenance.

According to IEC61499 for disturbed IPMCS, a distributed system control is defined as a collection of devices interconnected and communicating with each other by means of one or more communication networks, as shown in Figure 1.

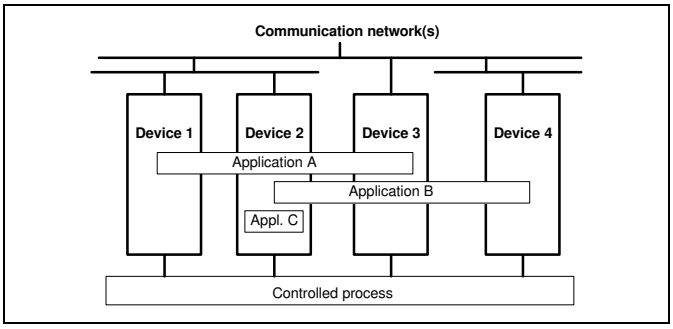

Figure 1 – Distributed System Model

A function performed by the control system is modeled as an application which may reside in a single device, such as application C in Figure 1, or may be distributed among several devices, such as applications A and B in Figure 1. For instance, an application may consist of one or more control loop in which input sampling is performed in one device, control processing is performed in another, and output conversion in a third.

According to the guidelines given in IEC 61499-4, the mainly implemented features specified in its compliance profile are illustrated in Figure 2. These features are:

- Portability: exchanges of software (control tasks) between software tools and suppliers are supported;
- Configurability: devices from multiple vendors can be configured by different software tools from multiple suppliers;
- Interoperability: devices from different vendors can operate with each other;
- Interchangeability: devices and resources from one vendor can exchange with the devices and resources from another vendor.

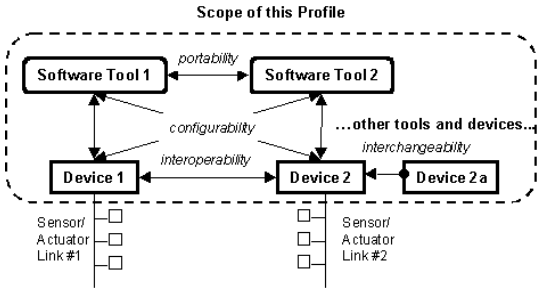

Figure 2. Implemented features of IEC61499

In an open distributed system, distributed devices may communicate and interact with each other. Each device is controlled dynamically by mobile tasks, expressed by Function Block codes, which may travel across open and untrustworthy networks. In these processes, any change to the devices, software modules or tasks may result in potentially hazardous conditions. Such changes include new function addition or modification, data/software transfer, remote diagnosis and maintenance amongst others. The open environment and dynamic processes bring new challenges to the system. Significantly, security risks may come from the network, data storage, operating platform, and application modules. Compared with the traditional control systems, running in closed trust environments, the new scenario has the following security challenges:

a). Authenticity: Active device nodes should access only trusted, reliable code from vendors whatever the transmission modes or forms, and the preservation manner;
b). Integrity: Active code must be ensured that it is not changed while traveling across open networks;
c). Confidentiality: Valuable code and data must be protected from malicious attackers or competitors;
d). Execution Safety: Each active device has independent local task execution protection policies and the respective support mechanisms.

3. Security Framework for Collaborative Distributed System

We have designed a security framework collaborative distributed control to address the security challenges discussed in section 2. The framework is illustrated in Figure 3.

There are three logical domains in this framework: client domain, task repository domain, and low-level control device domain. Each entity in different domains is geographically distributed and connected through networks (LAN or WAN). For an open collaborative distributed system, the above domains and its distributed entities may run in open and unsecured network, like the Internet.

On the client side, a user may hold one or more credential, which certifies that s/he has some granted rights to request some services under limited conditions. For example, before an operator wants to query the status of a device, the operator must provide a proof (credential) signed by the device's administrator or others who have the delegation authority. Depending on a user's priorities and responsibilities, s/he may request different services, like application design, device monitoring, device operation, or system re-configuration. The outputs and requests are all signed and encrypted by respective Security Agents (SA).

On the repository side, there are task repositories which store task components (IEC61499 Function Block components). These task components contain different application functions (i.e. PID Control) and come from different suppliers. These repositories may be distributed across several physical locations, but also connected by networks. Each repository maintains a series of policies (not shown in Figure 3 due to space limitations), which provide detailed security requirements. Only requests compatible with these policies can be served.

On the device side, there are a series of Security Control Gateways, with several local devices beneath them. By default, each Security Control Gateway and its associated devices are considered to be within one trust boundary. This means there is no security risk between entities in one trust boundary.

Considering this framework, suppose that an operator wishes to reconfigure some application on a running device. The typical remote task re-configuration process is described briefly as follows:

- First, the operator chooses control tasks by checking and retrieving task components from different repositories.
- The reconfiguration command is sent to the corresponding Security Control Gateway with appropriate credentials.

- Request and Credentials are collected and checked against the security policy in the Security Control Gateway.
- If the request and credentials are not compatible with the security policy, the request will be rejected. It is possible that the operator turns to seek more credentials to support his/her request, or coordinate with the device administrator on some adjustments of the gateway policy.
- If the policy is satisfied, Request and Task Code are delivered to Gateway securely.
- The Gateway executes the corresponding request. Some additional actions, such as logging and altering are also executed according to its security policy requirements.

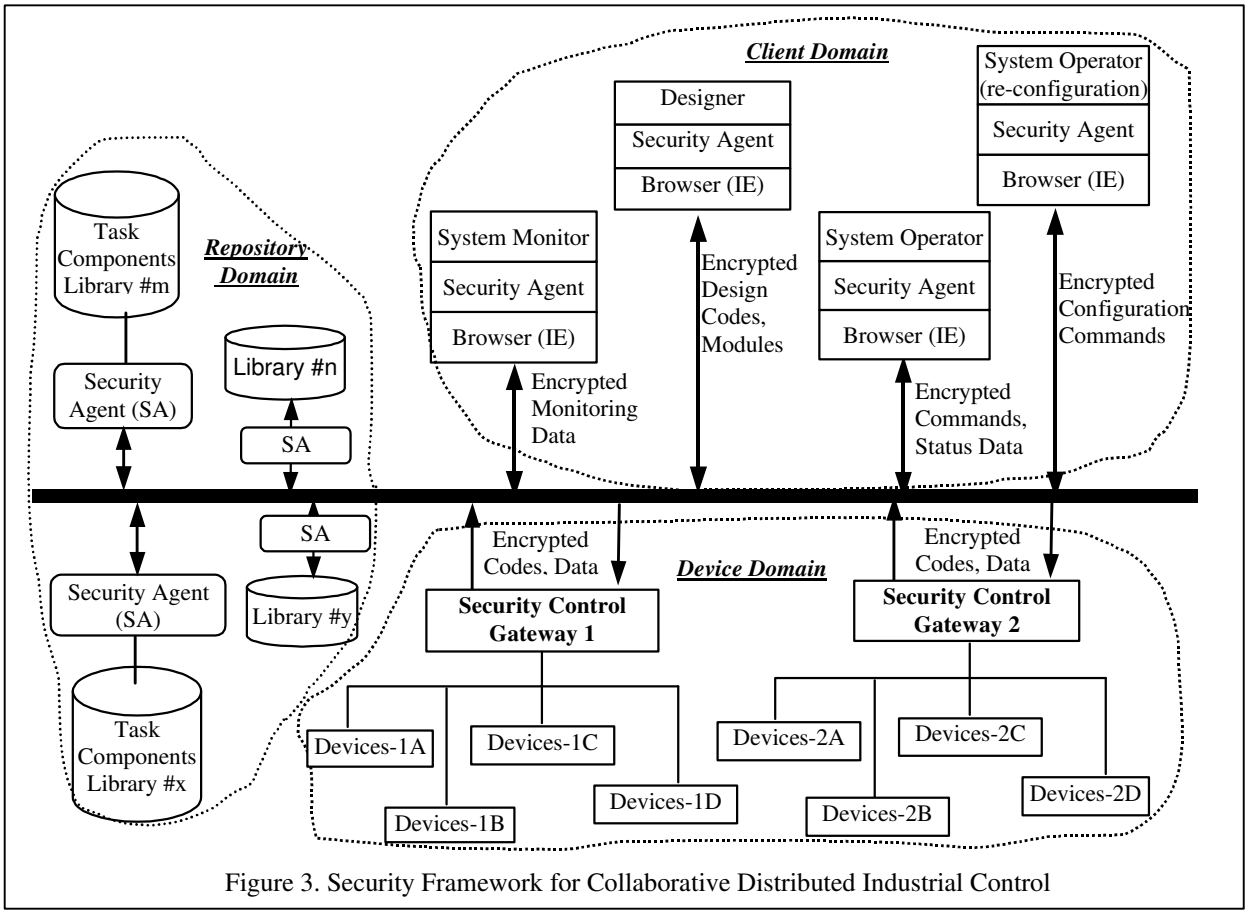

Figure 3. Security Framework for Collaborative Distributed Industrial Control

4. Security Control Gateway

In the above framework, the Security Control Gateway plays a very important role in mediating outside request and internal control actions. It guarantees the secure task delivery and execution of low-level control processes. The architecture of this gateway is show in Figure 4. The main components are:

- Security Agent: For incoming dataflow, it checks the authenticity and integrity of the data. Then it decrypts dataflow and form XML data, which may contain requests and task codes. For outgoing dataflow, it signs and encrypts outgoing XML data and delivery results. Details on the mechanisms are discussed in the next section.
- XML-Binder Agent: This agent is responsible for unmarshaling/marshaling XML data to/from Java Objects. For incoming XML data, it unmarshals XML data to runtime objects and generates OS-supported schedulable tasks (i.e. threads). The generated OS tasks are sent to the Admission Agent. For outgoing information, e.g. when the Execution Agent has feedback corresponding to the request, the respective feedback objects are marshaled to XML data and sent to the Security Agent to sign and be sent out. In these processes, because only those tasks compatible with predefined XML schema can be

unmarshaled or marshaled, this module contributes the execution safety partially.

- Admission Agent: When a new OS Task (i.e. thread) is created, the Admission Agent will check whether it is available to be executed according to the admission policy. For example, an admission policy specifies how much of the processor (CPU) bandwidth is reserved for real-time, If there is insufficient processor bandwidth available, the new OS Task will be refused to register in the Task Queue.

  This Admission Agent keeps the control process running safely by only allowing the entry of those OS Tasks that are compatible with its policies. This module is an important part of the Gateway. It guarantees the predictability of the distributed low-level control.

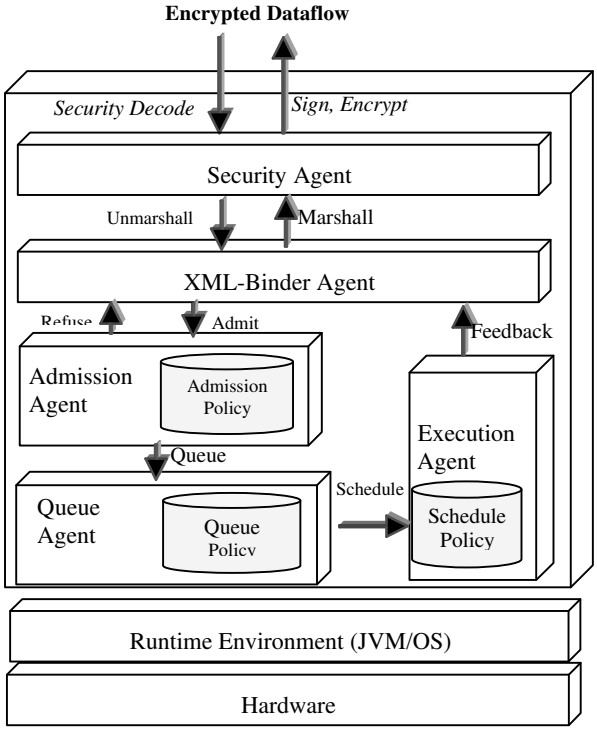

Figure 4. Architecture of the Security Control Gateway

- Queue Agent: When an OS Task passes the admission test, an OS Task ID will be assigned to it. The task will then be put into the Task Queue. The first task in the queue is always waiting for execution in the next scheduling period.

  The Queue Agent maintains the queuing policy for ordering queried tasks. Specified rules can be set, for example, the rule of "earliest deadline first" is used as the most appropriated ordering rule for real-time device control.

  Using different rules can constitute different task queues that are suitable for different types of devices and task control requirements. For example, the task queuing policy may be set according to the following factors:
  - Priority of each task
  - Task entry time
  - Fairness-guarantee that each waiting task will have a 'fair' opportunity to run
  - Desired completion time of each task

- Schedule Agent: the Schedule Agent is responsible for controlling the execution of a task. It gets the first ordered task from the task queue and uses the API of device to begin the real execution of that task on specified devices.

5. Security Mechanisms for Security Agent

In the distributed control network, distributed devices are controlled dynamically by mobile tasks, which may travel within untrustworthy networks. The Security Agent addresses three main security requirements: Authenticity, Integrity, and Confidentiality.

In the collaborative framework, each of the individuals including clients, repositories, and security control gateways has a unified key pair: a public key and a private key. A designated authority provides both of these. The public key is used as the identification of the key holder. The private key is used to form a signature on the credential and request and other cryptographic functions. This key is a key secret to the individual.

Before a task is delivery to other parties, it must be signed by the sender to identify the task and also secure its integrity or contents as well. Two schemes are provided with the latter one provide the protection of content confidentiality.

Scheme 1. Authenticity and Integrity

Figure 5 illustrates the process of the Scheme 1, including signature and verification process. It provides authenticity and integrity. The hash value of the original message is formed by hashing algorithm, which means a one-way transformation of a string of characters into a usually shorter fixed-length value. Then only the hash value of the message is signed in this scheme, which avoids the time-consuming process to sign large message. The original message is sent with the signature. The receiver verifies the signature by decrypting the hash with the sender's public key and matching it with the hash generated against the received message.

The basic protocol is described as:

- The sender creates the hash value (a) of the original message

- The hash value is encrypted with sender's private key
- The message and the encrypted hash value are sent to the receiver
- The receiver decrypts the encrypted hash value with the sender's public key
- The sender creates the hash value (b) of the original message
- Receiver verifies by comparing the hash value (a) it received and the hash values it created (b).

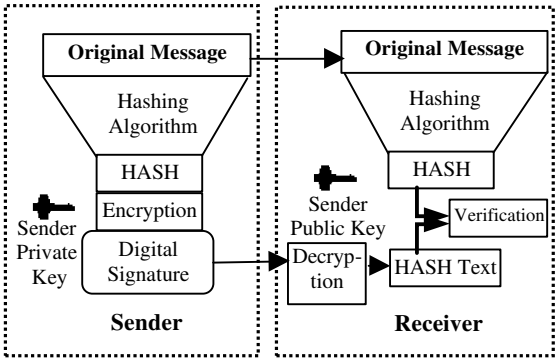

Figure 5. Scheme 1-Authenticity and Integrity

Scheme 2. Authenticity, Integrity and Confidentiality

Scheme 1 does not guarantee confidentiality since the message is sent as plaintext. To further guarantee confidentiality, instead of sending the plaintext message, the message is encrypted with the receiver's public key. The process is illustrated in Figure 6.

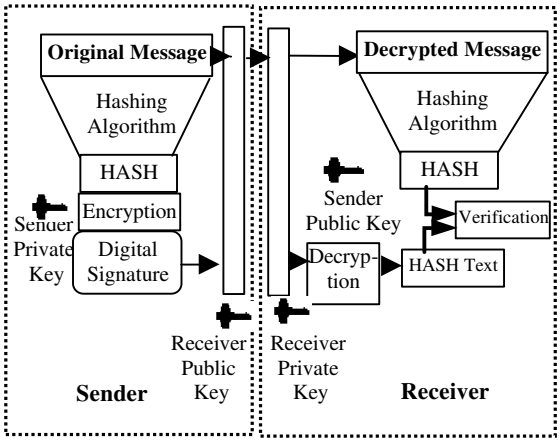

Figure 6. Scheme 2-Authenticity, Integrity and Confidentiality

6. Implementation Consideration of the Security Control Gateway

For distributed device-level control, most applications and requests contain rigid time constraints. Therefore, all the specified behaviors must be predictable, which means the execution and associated transaction process must be guaranteed to complete without violating the given time constraints. The security control gateway must also respond "fast enough" as defined by the characteristics of the request. Furthermore, the security control gateway should have the abilities to support time-critical execution of some request tasks. To support these abilities, appropriate scheduling mechanisms are critical for the implementation of the security control gateway.

The common real-time scheduling approaches for time-critical applications can be divided into two types: cyclic execution scheduling and preemptive scheduling. The preemptive scheduling is the ideal mechanism for the implementation of the security control gateway.

For prototyping, we consider Java Virtual Machine (JVM) to be the platform of the implementation. The main reasons are:

(1) Java has desirable features of portability, safety, and wide availability.
(2) Java incorporates thread and related concurrency constructs. Especially, Java intrinsically provides methods to support "preemptive" operations, where a thread can be preempted by the Java runtime to another thread.
(3) Java threads can potentially execute on more than one separate processor, as could be the case under Solaris or Windows NT running on a multiprocessor platform. Thus, the Java thread interface allows the system to take advantage of all available resources.
(4) The Java thread model provides means to synchronize tasks that may run at different speeds. This has special meaning for time-sensitive tasks.

As a virtual Operating System platform, combining with the advantages of its multi-threading model, JVM became our first choice for implementing the security control gateway. Furthermore, for product development, we are considering Real-time Java Operation Systems as the system kernel to implement the Security Control Gateway. More details will be reported in latter publications.

7. Conclusion and future work

Facing open and dynamic environments, an effective security framework is critical for the control of distributed collaborative systems and devices. The architecture and mechanisms we propose here are designed to solve the security challenges involved in control system design, configuration, operation, monitoring, and maintenance through open networks. By combining cryptographic functions, agent technologies, and XML data binding technologies, we address four major security problems of authenticity, integrity, confidentiality, and execution safety.

Security of distributed system control is a multifaceted issue touching multiple disciplines, domains, departments and even cultures. For different industrial application domains, security problems may have to be considered differently. For example, for the control of PLC devices in safety-related Medical Systems, special requirements like IEC 601-1-4 should be considered.

At the same time, there are other important issues to be investigated to extend the research on the proposed framework. For example, considering the limited resources and capabilities for low-level device control, lightweight security mechanisms form an important aspect to be addressed in our research. The trust mechanisms between different entities (clients, repositories, smart devices) are also under investigation and will be reported in later publications. As well, a detailed security analysis of the effectiveness is under development in our research group.

Reference

[1] Vision of the National Industrial Information Infrastructure Protocols (NIIIP),1999. http://www.niiip.org/vision.html.

[2] Waurzyniak, P. Electronic Intelligence in Manufacturing, SME Manufacturing Engineering, Vol.127, No.3, pp.44-67. 2001.

[3] Shen, W., Xue, D., and Norrie, D. H. An Agent-Based Manufacturing Enterprise Infrastructure for Distributed Integrated Intelligent Manufacturing Systems. Proceedings of PAAM'98, London, UK. 1998

[4] IEC TC65/WG6. Function Blocks for Industrial-Process Measurement and Control Systems. IEC-TC65/WG6 Committee.2000

[5] Adelson, B. Developing Strategic Alliances: A Framework for Collaborative Negotiation in Design, Research in Engineering Design, Vol.11, pp.133-144. 1999

[6] Bussmann, S. "An agent-oriented architecture for holonic manufacturing control", In: Proceedings of the 1st Int. Workshop on Intelligent Manufacturing Systems, EPFL, Lausanne, Switzerland, 1998.

[7] Caldwell, N. M. and Rodgers, P. A. WebCADET: Facilitating Distributed Design Support, IEE Colloquium on Webbased Knowledge Servers, London, U.K., pp.9/1-9/4. 1998.

[8] Morad, N. and Zalzala, A. Genetic Algorithms in Integrated Process Planning and Scheduling, Journal of Intelligent Manufacturing, No.10, pp.169-179. 1999.

[9] NCMS. Factory-Floor Internet: Promising New Technology or Looming Security Disaster Manufacturing In Depth, National Center for Manufacturing Sciences, November. 2001.

[10] Shen, W., Brooks, C., Li, Y., Lang, S. and Wang, L. XML-based Message Services for Internet Based Intelligent Shop Floors, Proceedings of SPIE Conference on Internet-Based Enterprise Integration and Management, pp.135-144. 2001

[11] Shen, W., Lang, S., Korba, L., Wang, L., and Wong, B. Reference Architecture for Internet Based Intelligent Shop Floors. Proceedings of SPIE, Vol.4208, pp.63-71. 2000

[12] Smith, C. S. and Wright, P. K. (1996). CyberCut: A World Wide Web Based Design-to-Fabrication Tool, Journal of Manufacturing Systems, Vol.15, No.6, pp.432-442.

[13] Sun, J., Zhang, Y. F. and Nee, A. Y. C. (2001). A Distributed Multi-agent Environment for Product Design and Manufacturing Planning, International Journal of Production Research, Vol.39, No.4, pp.625-645.

[14] Wang, L. and Norrie, D. H. (2001). Process Planning and Control in a Holonic Manufacturing Environment, Journal of Applied Systems Studies, Vol.2, No.1, pp.106-126.

[15] Wang, L., Wong, B., Shen, W. and Lang, S. Java 3D-Enabled Cyber Workspace, Communications of the ACM, Vol.45, No.11, pp.45-49. 2002

[16] Y. Xu, D. H. Norrie, Real-time Task Control Model for Holonic Systems, Agents 2001 Workshop on Holons: Autonomous and Cooperative Agents for Industry, Montreal, Canada, May 29, 2001. 7.

[17] Y. Xu, R. Brennan, et al, A Genetic Algorithm-based Approach to Holon Virtual Clustering, in Proceedings of World Multiconference on Systemics, Cybernetics and Informatics (SCI'2000), Vol. III, pp. 380-385, Orlando, Florida, USA, July 23-26, 2000.