A Reference Architecture for Control of Mechanical Systems

Thomas R. Kramer, M. K. Senehi, John L. Michaloski, Steven R. Ray, William G. Rippey, Sarah E. Wallace, Richard Quintero, James S. Albus

National Institute of Standards and Technology
Gaithersburg, MD 20899, U.S.A.

Abstract

This paper presents a reference architecture for control of mechanical systems. The architecture, called the "joint architecture", is derived in part from existing Real-time Control System (RCS) and Manufacturing Systems Integration (MSI) architectures at the National Institute of Standards and Technology. The joint architecture is under development and not yet complete. It is a hierarchical control architecture and focuses on control of systems for manufacturing discrete parts by machining. A definition of "architecture" has been adopted which includes explicit levels of abstraction, here termed "tiers of architectural definition", and five elements of architectural definition: statement of scope and purpose, domain analyses, architectural specification, methodology for architectural development, and conformance criteria. This paper gives an overview of the joint architecture and describes its two most abstract tiers.

1 Introduction

As industrial equipment becomes ever more sophisticated, computers and communications more powerful, and robots more capable, the need for a method of unifying diverse machines into coherent systems becomes increasingly urgent. The unification of diverse systems can be accomplished using a machine control system architecture. Without the consistent overall approach provided by an architecture, integrating variegated equipment into a system that does useful work is a labor-intensive, error-prone undertaking. Despite the agreed benefits and the development of many architectural approaches, no broadly applicable architecture has gained widespread acceptance.

1.1 Reference architectures

A "reference architecture" is a generic architecture for a specific domain. Typically, a reference architecture specifies integration rules and standard interfaces among components. By adhering to the standard interfaces and integration rules required by the architecture, different vendors can construct components which are interoperable. Using the interoperable components and system integration rules and methods, components may be integrated to build a machine, groups of machines and people can be integrated to form a workstation, workstations may be integrated to form cells and so on, to any degree of complexity desired. The availability of a reference architecture which defines interoperable components can improve the flexibility, timeliness, reliability, safety and extensibility of control systems.

Once a reference architecture is available which can serve as a standard, tools for building control systems can be constructed and applied, and a body of knowledge about how to apply the architecture can be built. Public availability of the architecture, tools and the knowledge of how to apply them to real-world control problems will greatly reduce the time and cost required for building control systems.

1.2 Reference architectures at NIST

The Manufacturing Engineering Laboratory (MEL) at the National Institute of Standards and Technology (NIST) is conducting research on control of mechanical systems for use in such diverse fields as discrete parts manufacturing, coal mining, under-ice submarining, and space exploration.

As a result of differing requirements in each domain, the characteristics of control systems vary greatly. Nevertheless, more than seventeen years of experience within the Robot Systems Division (RSD) and the Factory Automation Systems Division (FASD) of MEL indicate that there are aspects of control which are common to all control systems in a broad range of domains. These aspects have been captured in a number of control system reference architectures that provide both specifications for the parts of the architecture and their behaviors and methodologies for constructing control systems according to the prescribed specifications. The Automated Manufacturing Research Facility (AMRF) control architecture was developed in MEL [12], [13], [15], [20]. A reference architecture developed by RSD is the Real-Time Control

System (RCS) architecture [1], [2], [3], [5], [6], [10], [16]. Specializations of RCS, such as the NASA/NBS Standard Reference Model for Telerobot Control System Architecture (NASREM) [4] have been defined. FASD has developed the Manufacturing Systems Integration (MSI) architecture [18], [19], [22].

The MSI and RCS architectures share many common features. For example, both consist of a set of controllers arranged in a command hierarchy. In both, each type of controller has its own specialized set of commands it can carry out. Both implement command execution by message passing between controllers, and so forth. But there are also some differences. Timing issues and sensory processing receive more attention in RCS, information integration, scheduling and resource definition issues more in MSI.

RSD and FASD engaged in a joint project to study the feasibility of formulating a single reference architecture, a "joint architecture", which includes features of RCS and MSI. The study [14] determined that a joint architecture is feasible and outlined it. The study recommended that the joint architecture focus on discrete parts manufacturing. That recommendation has been followed. The joint project was continued to develop the joint architecture more fully. The second phase report [17] is a snapshot of the joint architecture at its current, incomplete, stage of development. The joint project is continuing further, and it is planned to complete the joint architecture late in 1994, after which it will be implemented and refined. This paper describes the architecture as it is conceived in [17].

2 Definition of an architecture

This section presents the terms in which the joint architecture is defined. Two fundamental concepts in our definition of an architecture are tiers of architectural definition and elements of architectural definition.

The "architectural units" of an architecture are simply the concepts which are important to the architecture. Architectural units may be more or less concrete in nature. Architectural units of similar concreteness can be grouped together to form a cross section of the architecture. We shall refer to such a grouping as a tier of architectural definition, or simply tier. The concept of tier of architectural definition appears under different names in [7], [8], [9].

At each tier, the definition of an architecture consists of specifying a number of elements of architectural definition. These are:

(1) statement of scope and purpose
(2) domain analyses
(3) architectural specification
(4) methodology for architectural development
(5) conformance criteria

These elements of architectural definition vary in indispensability. For example, an architecture must have an architectural specification, but it is possible to use an architecture which omits conformance criteria. Existing reference architectures include different subsets of these elements and place emphasis on them in varying degrees. However, an architecture which is completely defined addresses all elements in a balanced fashion.

The remainder of this section expands on the notions of tiers and elements of architectural definition.

2.1 Tiers of architectural definition

An architecture consists of architectural units, each of which is more or less concrete in nature. Often, two architectural units are related by having the second be a specialization of the first — conversely, the first is a generalization of the second. Two architectural units connected in this way are said to have an abstraction relation. Abstraction relations may connect an entire chain of architectural units. For example: at an abstract level, one might define templates for information models, at a somewhat more concrete level, a set of information models conforming to the templates might be defined for a particular application, and at an even more concrete level, database software might be designed implementing the information models.

It is useful to be able to define an architecture at different levels of abstraction. To do this, we divide the architectural units of an architecture into groups. Each group is called a tier of architectural definition, or simply tier. Every architectural unit of an architecture is assigned to one tier or another. Whenever two architectural units are related by an abstraction relation, the more abstract one should be in a higher tier or the same tier as the more concrete one. Thus, the tiers of an architecture form cross-sections of the architecture, with higher tiers being, generally, more abstract than lower ones. Note that any two arbitrary architectural units need not be related by an abstraction relation.

It would be appealing to require that all architectural units in a tier be of similar concreteness; the feasibility study [14] defined tiers that way. There are several shortcomings to making this a requirement, however. First, while the abstraction relation provides a partial ordering, there is no absolute scale for measuring abstraction and no commonly agreed upon method for assigning an absolute measure of abstraction to an architectural unit. Second, any two chains of architectural units formed by abstraction relations may be different lengths, so tiers cannot be constructed by putting all the first links in the first tier, all the second links in the second tier, and so on. Third, it may be more convenient for defining an architecture to define some items concretely even at a high tier, while keeping others more abstract at lower tiers.

On figures showing architectures, the lower tiers appear lower on the chart. In the numbering system for tiers used here, however, the tier at the top is tier 1, the next lower tier is tier 2, and so on.

Different architectures may have different numbers of tiers of architectural definition. Tiers may be explicit or implicit. RCS may reasonably be divided into three tiers and MSI into two tiers, although neither of the two has explicit tiers in existing descriptions.

2.2 Elements of architectural definition

As stated, there are five elements of architectural definition. A description of each follows.

2.2.1 Statement of scope and purpose: The statement of scope of an architecture describes the range of areas to which the architecture is intended to be applied. It is useful to identify items which are explicitly out of scope, and to identify general characteristics of the domain which may extend or limit its applicability to other domains.

A statement of purpose identifies what the objectives of an architecture are within the given scope. The statement of purpose of an architecture should be a major determinant of the contents of the architecture. For example, if the objective is to achieve interoperability between components of an implementation, it would be expected that definitions of shared information and interfaces between components would be stressed. If the objective is to guarantee real-time performance of the resulting control system, execution models may be stressed.

2.2.2 Domain analyses: An analysis, in general, is an examination of the components of some complex and how they relate to one another. A critical step which must take place before an architecture can be formulated is to perform analyses of the target domain that reveal its essential characteristics. These analyses are “domain analyses”. The type of analyses done, the order in which the analyses are performed and the language in which the results are expressed are part of the methodology for domain analysis. The results of the domain analyses may be very different depending on the types of analysis performed and the analysis methodologies used.

Commonly used forms of domain analysis are functional analysis, information analysis, and dynamic analysis [11]. Functional and information analysis are particularly well entrenched and have been used in structured programming for many years.

A functional analysis of a domain is an analysis of all the activities within the scope of the architecture which a conforming control system should be able to perform.

An information analysis of a domain is an analysis of all the information within the scope of the architecture needed for a conforming control system to function properly.

A dynamic analysis of a domain is an analysis of the characteristics of the functions and information in the domain that vary over time during control system operation. It provides qualitative and quantitative information about the sequence, duration and frequency of change in the functions and information of the domain [11].

Many methods for performing analyses have been developed, but a discussion of them goes beyond the scope of this paper.

2.2.3 Architectural specification: An architectural specification is a prescription of what the pieces (software, languages, execution models, controller models, communication models, computer hardware, machinery, etc.) of an architecture are, how they are connected (logically and physically), and how they interact. In most architectures the architectural specification accounts for the bulk of the description of the architecture.

2.2.4 Methodology for architectural development: It is important for an architecture to have a set of procedures for refining and implementing the architecture. This set of procedures is called the methodology for architectural development for the architecture (which we usually shorten to “methodology”). The architectural specification at each tier of architectural definition is related to, and used in, generation of an architectural specification for the other related tiers as specified in the methodology for architectural development. If an architecture has more than one tier of architectural definition, a methodology will be needed to link each two adjacent tiers. If an architecture lacks a methodology for getting between any two tiers of architectural definition, control systems developers must devise their own methods for making the transition.

A methodology may specify top-down decomposition, bottom-up composition or some combination of both in constructing the complete architecture. For example, if the code or specifications for the lowest tier is available, as is often the case when dealing with vendor-supplied equipment, an implementation-independent template for the code may be developed. In this case, the methodology would describe how to use the template.

2.2.5 Conformance criteria: Conformance criteria are standards which specify how an architectural unit at one tier of an architecture conforms to the architectural specification of a higher tier, or how a process for building part of an architecture conforms to the development methodology given by the architecture for building that part.

Methods for determining conformance of a component of an architecture might include:

(1) reading source code,
(2) checking that documents which are supposed to be in computer-processable format are in fact computer-processable,
(3) observing an implementation in action,
(4) devising test cases and using them to test control systems,
(5) examining documentation of development activities.

3 The NIST joint architecture

This section gives an overview of the joint architecture and descriptions of the first two tiers of the architecture. The description of the tiers reflects the current contents of our formal model of the architecture, which is written in the EXPRESS modeling language [21].

3.1 Overview of the joint architecture

This section presents several broad aspects of the joint architecture, including some of our general strategies for building it. Additional strategies (for example: use fine granularity) not addressed in this paper are discussed in Section 8 of [14] and in early sections of [17].

3.1.1 Overall focus: In devising the joint architecture, we have decided to focus on control systems for shops which produce discrete machined metal parts. The architecture is to integrate shop planning, scheduling, and control functions in both nominal and error situations and to allow control of a shop with any combination of physical and emulated equipment. Certain aspects of the architecture are likely to apply to broader domains, but only the discrete parts shop is being given careful consideration while the architecture is being built. This focus area is nearly identical to the focus of MSI and overlaps heavily the territory of several existing applications of RCS. The focus area also reflects the continuing interests of MEL.

3.1.2 Major features of the joint architecture: Major features of the joint architecture are as follows.

Tiers of Architectural Definition — Tiers of architectural definition are explicitly defined, as already discussed.

Elements of Architectural definition — All five elements of architectural definition are explicitly included, as described earlier.

Command and Status Controller Interaction — Controllers interact via a command and status protocol.

Hierarchical Arrangement of Controllers — Controllers are arranged in a hierarchy. At any time, each controller must have one superior (except the controller at the top of the hierarchy, which has none) and may have zero to many subordinates.

Three Types of Control Unit — Three types of control unit are used: scheduled control units for upper layers of the control hierarchy, real-time control units for the lower layers, and transition control units between.

Hierarchical Task Decomposition — Predefined tasks are used as the basis for commands, and tasks are decomposed hierarchically to match the control hierarchy.

Non-Hierarchical Data Access — Data access is not hierarchical and may take place through different communications channels from those used for command and status messages.

Human Interfaces to Controllers — A human interface is available to each control unit.

Sensory Feedback for Closed-Loop Control — In real-time control units, sensory input may be used as feedback for closed-loop control.

Operating in Hard Real Time — Real-time control units can operate in hard real time — i.e., the control unit can always generate a response within fixed time interval.

Three-Stage Planning — Three stages of planning are used: process planning, production planning, and scheduling.

Definition of Messaging Protocols — Pre-defined messaging protocols are used for major systems functions: planning, carrying out work, error recovery, etc.

Error Recovery — Explicit provisions are made for error recovery, including special message protocols.

3.1.3 Tiers of architectural definition: The joint architecture has five tiers, as shown in Figure 1. Tier 1 is for defining a general hierarchical control architecture. Tier 2 is for defining a hierarchical control architecture suitable for the discrete parts shop domain. We have not firmly decided how to use the lower three tiers, but our current thinking is as follows. Tier 3 is for defining some specific application (such as a work cell with 3-axis machining, or a factory with several work cells). Tier 4 is for defining the logic of an implementation of the specific application, and tier 5 is for defining details of the implementation.

Since the joint architecture is to be suitable for control of a broad range of systems in a discrete parts shop, only the upper two tiers of the architecture will be heavily populated with elements of architectural definition when the architecture is complete. The lowest three tiers are intended to be defined differently for different applications and implementations, so the joint architecture itself will provide only the skeletons of those tiers. These skeletons remain to be built. This paper describes only the top two tiers of the joint architecture, and these tiers are not yet fully defined. Additional items of architectural specification and other elements are desirable at both tiers.

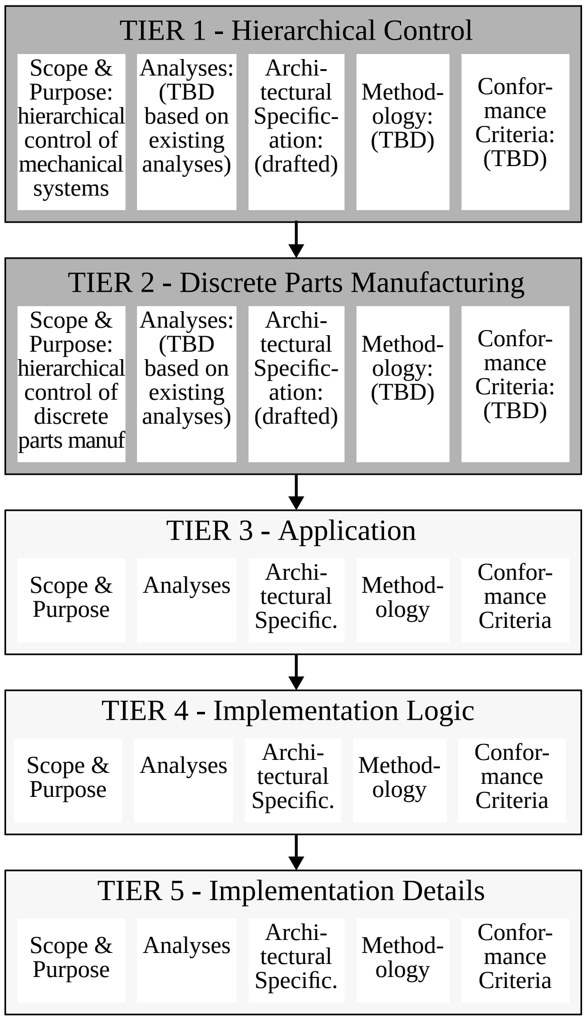
TIER 1 - Hierarchical Control
Scope & Purpose: hierarchical control of mechanical systems
Analyses: (TBD based on existing analyses)
Archi-tectural Specific-ation: (drafted)
Method-ology: (TBD)
Confor-mance Criteria: (TBD)
TIER 2 - Discrete Parts Manufacturing
Scope & Purpose: hierarchical control of discrete parts manuf
Analyses: (TBD based on existing analyses)
Archi-tectural Specific-ation: (drafted)
Method-ology: (TBD)
Confor-mance Criteria: (TBD)
TIER 3 - Application
Scope & Purpose
Analyses
Archi-tectural Specific.
Method-ology
Confor-mance Criteria
TIER 4 - Implementation Logic
Scope & Purpose
Analyses
Archi-tectural Specific.
Method-ology
Confor-mance Criteria
TIER 5 - Implementation Details
Scope & Purpose
Analyses
Archi-tectural Specific.
Method-ology
Confor-mance Criteria

Figure 1. Joint Architecture - Tiers of Architectural Definition

3.1.4 Elements of architectural definition: In the joint architecture's current, incomplete state, the specifications of elements of architectural definition are at various stages of completion.

We have completed the scope and purpose for the first two tiers of the architecture, since it necessary to decide what one is trying to do at the outset. Moreover, scope and purpose may be defined briefly without encountering difficult technical challenges.

The architectural specification of the first two tiers is partly developed.

The other three elements, domain analyses, methodology for architectural development, and conformance criteria, are less well developed. Extensive domain analyses have been performed in the past by the developers of RCS and MSI, however, so we have felt comfortable working on architectural specifications without having yet done any additional formal analyses.

3.1.5 Overall methodology for architectural development: The joint architecture employs a cyclic development approach. The idea of cyclic development is that one develops an architecture, assesses the finished product (the assessment would include implementing the architecture), and uses the results of the assessment as feedback to a cycle of refining the architecture. This may be done several times.

3.1.6 Modular construction: The architecture uses modular construction insofar as possible. Information, control, and communications are separated. Within communications, the logical definition of messages is separated from message encoding (the string-of-bits definition) and separated from the method by which bits are moved from one place to another.

3.2 First tier of architectural definition

This tier includes many generic control architecture concepts which could equally well serve as the foundation for radically different architectures. We are treating these generic concepts as part of the first tier, but the first tier could readily be split in two, and we have done that in the formal EXPRESS model of the architecture included in [17] by having separate schemas for generic control concepts and concepts specific to hierarchical control.

3.2.1 Scope and purpose: The scope at this tier is broad. It is assumed that there is a need for a control system, and that the system being controlled must interact with its environment and react to unpredicted changes in the environment. No further characteristics are assumed.

The purpose of this tier of the joint architecture is to give guidelines for the construction of a general control system in this very broad domain. This tier is specifically intended to be applicable to control systems for factories, robots, autonomous vehicles, construction machines, and mining machines.

3.2.2 Architectural specification: Since our formal model separates the description of generic control architecture from the description of hierarchical control architecture, we will do that here.

The generic architectural specification in the first tier of the joint architecture includes the definitions of "control architecture" and the components from which a generic control architecture is made. This includes elements and tiers of architectural definition as presented earlier. We will not name them again here, but they are explicitly included. Several subtypes not mentioned here are also included.

The basic active unit of a control architecture is called "interactive unit". An interactive unit may be a functional unit (which, in turn, may be either a control unit or a planner) or a data store manager. Interactive units interact by sending one another messages. The messages may be data messages or functional messages (commands or status messages). Related sets of messages between two specific interactive units for accomplishing some purpose form message protocols. A communication method and a set of interaction protocols between two interactive units forms an interaction setup. Interaction protocols and interaction setups may be either for data access or for functional activity.

The notion of "plan" is defined, but plans are not currently linked (as they should be) to control units at this generic level. The notions of "planning", "planner" and "planning model" are also defined at this generic level. A planner is a type of functional unit that produces plans.

The notion of "communication method" is introduced at this generic level, but has not yet been fully developed. The current model implicitly uses point-to-point communications. Full development of details of communications is expected to be a major task in completing the joint architecture. We anticipate that different communications mechanisms may be required for different purposes.

In the hierarchical control schema, the notions of command and status messages are introduced. These are subtypes of functional message. A command message is a message from one control unit to another which tells the receiving unit to do something. A status message is a message from one control unit to another in which the sender reports on the status of executing a command received earlier from the receiver of the status message, or the sender reports on its health.

A command and status protocol is a functional interaction protocol in which the messages are command messages and status messages. A unit consisting of a superior and all its subordinates is defined in terms of command and status protocols, and a hierarchy is defined as a set of these units (with appropriate restrictions, so that each subordinate has only one superior, and the arrangement is not cyclic). A hierarchical control architecture is simply a control architecture in which the control units are arranged in a hierarchy.

Whether the other functional units can have separate hierarchies remains to be determined in the joint architecture. Sensory processing may need to have a hierarchy different from the control hierarchy but has not yet been included in the formal model.

3.3 Second tier of architectural definition

3.3.1 Scope and purpose: The scope is narrowed to discrete parts manufacturing. The purpose (not part of the current EXPRESS model) is primarily to provide for real-time control of manufacturing equipment and integration of the control system in the environment. A secondary purpose is also to allow for integration of manufacturing functions with design, management, business, and maintenance functions at a later date.

3.3.2 Architectural specification: The joint architecture has three basic types of control unit: Scheduled Control Unit (SCU), Real-Time Control Unit (RTCU) and Transition Control Unit (TCU). Scheduled Control Units, patterned after the MSI generic controller, are to be used at high levels of control where real-time response is not required, or where there is the need to manage the allocation of resources among controllers which do not have the same immediate superior. Real-Time Control Units, patterned after the RCS model, are to be used when real-time control is required or when sensory input must be processed. Transition Control Units are to be used as superiors of RTCUs and subordinates of SCUs. The job of TCUs is to bridge between the two operational paradigms. The notion of a melded hierarchy is defined, in which the controllers at the top are SCU's, the controllers at the bottom may be RTCU's, and TCU's are in between.

The architectural specification provides for a three-phase planning model, with appropriate subtypes for plan, planner, and planning.

The first phase is process planning (which is done by a process planner and produces a process plan), in which it is determined how to make parts of a given design. Things required to make the part (raw stock, machine tools, cutting tools, etc.) may be specified in general terms in this phase.

The second phase is production planning (which is done by a production planner and produces a production plan). A production plan is the plan for producing a batch of parts. Production plans are prepared from corresponding process plans when orders for parts have been received and it is known how many parts of a given design are to be made in some time period (the next week, perhaps). To make a production plan, one or more alternatives from the process plan are selected and material handling steps are placed where needed.

The third phase is scheduling (which is done by a scheduler and produces a schedule). Schedules describe what parts will be made in specific work cells (or in specific work stations or on specific machines), at specific times, using specific cutting tools (or other specific resources). A schedule is required to guide the daily activities of a shop, and the SCU's in the control system for the shop work from schedules. Schedules are prepared from corresponding production plans.

4 Conclusion

This paper has presented the need for reference architectures, described how an architecture is defined, and presented the NIST joint architecture as currently conceived. The joint architecture is not yet complete. We plan to finish defining it, implement it, and refine it.

References

[1] Albus, James S.; Blidberg, D. Richard; *A Control System Architecture for Multiple Autonomous Vehicles*; Proceedings of the Fifth International Symposium on Unmanned, Untethered Submersible Technology; Merrimack, NH; June 1987

[2] Albus, James S.; *RCS: A Reference Model Architecture for Intelligent Control*; IEEE Journal on Computer Architectures for Intelligent Machines; May 1992; pp. 56 - 59

[3] Albus, James S.; Quintero, Richard; Lumia, Ronald; Herman, Martin; Kilmer, Roger D.; *A Reference Model Architecture for ARTICS*; Manufacturing Review; Vol. 4, No. 3; September 1991; pp. 182 - 193

[4] Albus, James S.; *A Theory of Intelligent Systems*; Control and Dynamic Systems; Vol. 45; 1991; pp. 197 - 248

[5] Albus, James S.; McCain, Harry G.; Lumia, Ronald; *NASA/NBS Standard Reference Model for Telerobot Control System Architecture (NASREM)*; NIST Technical Note 1235, 1989 Edition; National Institute of Standards and Technology; April 1989

[6] Barbera, Anthony J.; *An Architecture for a Robot Hierarchical Control System*; NBS Special Publication 500-23; National Bureau of Standards; December 1977

[7] Biemans, Frank P. M.; Vissers, Chris A.; *A Systems Theoretic View of Computer Integrated Manufacturing*; Proceedings of CIMCON ‘90, NIST Special Publication 785; National Institute of Standards and Technology; May 1990; pp. 390 - 410

[8] Bohms, Michel; Tolman, Frits; *RIA: Reference Model for Industrial Automation*; Proceedings of CIMCON ‘90, NIST Special Publication 785; National Institute of Standards and Technology; May 1990; pp. 114 - 132

[9] Dornier GmbH; *Baseline A&R Control Development Methodology Definition Report*; study managed by P. Putz; European Space Agency Contract Report; October 1991

[10] Herman, Martin; Albus, James S.; Hong, Tsai-Hong; *Intelligent Control for Multiple Autonomous Undersea Vehicles*; Neural Networks for Control; MIT Press; 1990

[11] Jayaraman, Sundaresan; *Design and Development of an Architecture for Computer-Integrated Manufacturing in the Apparel Industry Part I: Basic Concepts and Methodology Selection*; Textile Research Journal; Vol. 60, No. 5; May 1990; pp. 247 - 254

[12] Jones, Albert T.; McLean, Charles R.; *A Proposed Hierarchical Control Model for Automated Manufacturing Systems*; Journal of Manufacturing Systems; Vol. 5, No. 1; 1986; pp 15 -25

[13] Jones, Albert T.; McLean, Charles R.; *A Production Control Module for the AMRF*; Proceedings of the 1985 ASME Computers in Engineering Conference; August 1985

[14] Kramer, Thomas R.; Senehi, M. K.; *Feasibility Study: Reference Architecture for Machine Control Systems Integration*; NISTIR 5297; National Institute of Standards and Technology; November 1993

[15] McLean, C. R.; *Interface Concepts for Plug-Compatible Production Management Systems*; Proceedings of the IFIP WG5.7 Working Conference on Information Flow in Automated Manufacturing Systems; Gaithersburg, MD; August 1987. Reprinted in Computers in Industry; Vol. 9; pp. 307-318; 1987.

[16] Quintero, Richard; Barbera, Anthony J.; *A Real-Time Control System Methodology for Developing Intelligent Control Systems*; NISTIR 4936; National Institute of Standards and Technology; October 1992

[17] Senehi, M. K.; Kramer, Thomas R.; Michaloski, John; Ray, Steven R.; Rippey, William; Wallace, Sarah; *Reference Architecture for Machine Control Systems Integration*; NISTIR draft; National Institute of Standards and Technology; to appear

[18] Senehi, M. K.; Barkmeyer, Edward J.; Luce, Mark E.; Ray, Steven R.; Wallace, Evan K.; Wallace, Sarah; *Manufacturing Systems Integration Initial Architecture Document*; NISTIR 4682; National Institute of Standards and Technology; September 1991

[19] Senehi, M.K.; Wallace, Sarah; Luce, Mark E.; *An Architecture for Manufacturing Systems Integration*; Proceedings of ASME Manufacturing International Conference; Dallas, Texas; April 1992

[20] Simpson, J.; Hocken R.; Albus, J.; *The Automated Manufacturing Research Facility*; Journal of Manufacturing Systems; Vol. 1; Number 1, 1982

[21] Spiby, Philip; draft STEP Part 11 *EXPRESS Language Reference Manual*; April 1991

[22] Wallace, Sarah; Senehi, M. K.; Barkmeyer, Edward J.; Ray, Steven R.; Wallace, Evan K.; *Manufacturing Systems Integration Control Entity Interface Specification*; NISTIR 5272; National Institute of Standards and Technology; September 1993