

SEMI E38-1296 (Withdrawn 0307)

CLUSTER TOOL MODULE COMMUNICATIONS (CTMC)

This standard was technically approved by the global Information & Control Committee. This edition was approved for publication by the global Audits and Reviews Subcommittee on November 21, 2006. It was available at www.semi.org in February 2007. Originally published in 1995; previously published December 1996.

NOTICE: This document was balloted and approved for withdrawal in 2007.

1 Purpose

1.1 Cluster tools fulfill a need for modularity and flexibility in semiconductor manufacturing equipment. This standard addresses the communications with and among modules within a cluster tool for automated control.

1.2 The module communications services defined here will enable standard interoperability of modules from independent suppliers. Together with other cluster tool standards, this is intended to result in the emergence of standards-compliant interoperable sub-systems. They should allow applications software to be developed which can assume the existence of these services and allow software products to be developed to offer them.

1.3 The adoption of the standards described will greatly reduce the effort required to integrate cluster tool components from independent suppliers. Compliance requires support of a minimal, but specific, set of standard services.

2 Scope

2.1 Cluster tool module communications address only communications with and among modules within a cluster tool and not the communications between the cluster tool and the factory. It is the modules and their interrelations which are modeled and not the cluster tool seen as a single equipment.

2.2 This standard does not specify cluster controller functionality (e.g., scheduling, human machine interface, cluster tool recipe editing, and management) but will enable development of standards in this area. The cluster controller is included only to the extent that it represents the entity or entities responsible for supervisory control of the modules in a cluster tool.

2.3 This standard identifies the communications services necessary to achieve automated control of independent transport, process, and cassette modules in a cluster tool. It defines the essential module architecture and the concepts and models on which the communications services are based.

2.4 The scope includes primary control services for material processing in process modules, material movement within the cluster tool, and material input/output with the factory. Support services also exist to enable recipe handling, resolution of exception conditions, event reporting, and data access.

2.5 A reliable communications environment is required for distributed control and is specified in supplementary standards.

2.6 This standard specifies the application of more general service standards as required within a cluster tool. This includes the limitations imposed by the cluster tool architecture and the fundamental functionality needed for compliance. The details of the general services, protocols, and communication environment elements are defined in the corresponding standards documents referenced.

2.7 This version of the standard is provisional because a number of the general services on which it relies are not yet standardized. The sections referring to these services have been omitted until they have become standards. These sections will be balloted separately with the corresponding general services and incorporated into this provisional standard.

2.8 The communication between the cluster tool as a single equipment and the factory is beyond the scope of cluster tool module communications standardized here and should follow the applicable SEMI Standards (GEM, MMMS).

2.9 These standards place no restriction on where, within the cluster tool control architecture, this specific set of definitions is implemented. However, to have complete compliance to the standards in the area of the communications interface definitions, the entire layered structure as outlined in this document must be implemented.

NOTICE: This standard does not purport to address safety issues, if any, associated with its use. It is the responsibility of the users of this standard to establish appropriate safety and health practices and determine the applicability of regulatory or other limitations prior to use.

3 Referenced Standards and Documents

3.1 SEMI Standards

SEMI E21 — Cluster Tool Module Interface: Mechanical Interface and Wafer Transport Standard

SEMI E22 — Cluster Tool Module Interface: Transport Module End Effector Exclusion Volume Standard

SEMI E30 — Generic Model for Communications and Control of SEMI Equipment (GEM)

SEMI E32 — Material Movement Management Standard

SEMI E39 — Object Services Standard: Concepts, Behavior, and Services

4 Terminology

4.1 The following terms are those that are appropriate for describing the overall structure of cluster tools. These terms, in addition to the more specific terms located in the individual service, protocol, and communications environment standards, complete the glossary of terms required.

4.1.1 *agent* — an intelligent system within a factory that provides one or more service resources and uses the services of other agents. A generalization of host, equipment, cell, cluster, cluster module, station controller, work station. Agents are associated with a physical system or a collection of physical systems, including computer platforms.

4.1.2 *alarm* — an alarm is related to any abnormal situation on the equipment that may endanger people, equipment, or material being processed. [SEMI E30]

4.1.3 *atomic transfer* — the basic unit of movement. The transfer of a single material between two partners where only one change in ownership occurs.

4.1.4 *attached module* — any component of a cluster tool which mechanically attaches to the transport module at an interface flange. Specifically, the cassette, process, and docking modules are all attached modules. The attached module can be isolated from the transport module by an isolation valve controlled by the transport module. In addition, the attached module may also possess an additional isolation valve. The point of separation between the isolation valve on the transport module and the attached module is called the interface plane.

4.1.5 *am transfer job* — a material transfer job for an attached module specifying all criteria for receiving the material from, or sending the material to, a transport module.

4.1.6 *carrier (material)* — a device for the holding of material for various processing steps in semiconductor manufacturing. [SEMI E1]

4.1.7 *cassette module* — an attached module which provides the means of exchanging material between the intertool environment and the intratool environment of a particular cluster. There is no requirement that the material interface to the intertool environment with the use of cassettes.

4.1.8 *cm input/output transfer job* — a material transfer job for a cassette module, specifying all criteria for receiving the material from, or sending the material to, the factory.

4.1.9 *cluster controller* — the entity or entities responsible for supervisory control within a cluster tool, achieved through communicating with the various modules. The form of the cluster controller is not dictated. It may be a single platform, distributed, or incorporated with one or more modules.

4.1.10 *cluster tool* — an integrated, environmentally isolated, manufacturing system consisting of process, transport, and cassette modules mechanically linked together. There is no requirement that the modules come from the same supplier. [SEMI E21]

- 4.1.11 *decision authority* — an entity requiring to be notified of significant exception condition changes and which decides how to proceed to resolve abnormal situations related to recoverable error conditions. The decision authority may be represented by a supervisory controller interacting with an operator who may ultimately choose the recovery action.
- 4.1.12 *docking module* — a component for allowing the exchange, within a cluster tool, of material between multiple transport modules. The docking module presents an attached module interface to both transport modules and is modeled as a multi-port process module for intra-cluster tool communications.
- 4.1.13 *end effector* — a physical location attached to a transfer resource capable of holding material during an end-to-end material transfer.
- 4.1.14 *error condition* — an exception condition which is not an alarm and which may support recovery actions requested by a decision authority.
- 4.1.15 *exception condition* — a managed condition for reporting on and providing recovery from an abnormal situation in the equipment.
- 4.1.16 *factory* — entities outside the control domain of the cluster tool and from which material is received for processing and returned upon process completion. Also considered to be the intertool environment.
- 4.1.17 *factory transport resource* — an operator or a piece of equipment specialized in the transport of material from one equipment to another.
- 4.1.18 *form* — type of data representing information contained in an object attribute or service message parameter. The data types are detailed in § 4.2.
- 4.1.19 *fundamental requirements* — the requirement for information and behavior that must be satisfied for compliance to a standard. Fundamental requirements apply to specific areas of application, objects, or services.
- 4.1.20 *handoff micro move* — an operation requested by an attached module and performed by a transport module to achieve all or part of a material handoff between them.
- 4.1.21 *interface flange* — the boundary plane separating an attached module and a single transport module.
- 4.1.22 *intertool environment* — defined relative to a particular cluster tool to be everywhere outside of the intratool environment.
- 4.1.23 *intertool material transfer job* — a transfer job in the cluster controller to receive material from, or send the material to, the factory. In automated transfer, it may be an element of a factory transfer job.
- 4.1.24 *intertool port resource* — a module resource associated with a particular cassette module which operates on behalf of the said module during material transfer between the cassette module and the intertool environment.
- 4.1.25 *intratool environment* — the entire environmentally isolated volume contained within a cluster tool.
- 4.1.26 *intratool port resource* — a module resource associated with a particular interface plane of a particular attached module which operates on behalf of the said module during material transfer between the transport module and the attached module.
- 4.1.27 *intratool material transfer job* — a transfer job in a cluster controller for transfer of material from one attached module to another through the linking transport module.
- 4.1.28 *isolation valve* — a mechanical component used at an interface plane to permit environmental isolation between the transport module and an attached module.
- 4.1.29 *material handoff* — the process by which material moves from the sending transfer partner to the receiving transfer partner.
- 4.1.30 *material location* — a physical position capable of holding a single material.
- 4.1.31 *material slot* — a material location in a carrier capable of holding a single material with an assigned identifier.
- 4.1.32 *module* — an independently operable unit that is part of a tool or system. [SEMI E21]

4.1.33 *module resource* — an entity which provides specific capabilities required of the module to be accessed or controlled externally. This includes the physical capability and the associated control and access management. It may contain material in one or more material locations.

4.1.34 *namespace* — a domain within which object identifiers are unique.

4.1.35 *port* — a point on the cluster at which a change of ownership of material occurs. A port is not itself a location but must have an associated location, such as an interface plane.

4.1.36 *process job* — a material processing job for a process module, specifying and tracking the processing to be applied to the material while in the module.

4.1.37 *process module* — an attached module which provides manufacturing value to material in a cluster tool.

4.1.38 *processing resource* — a module resource within a module which is independently capable of providing manufacturing value to material.

4.1.39 *protocol (communications)* — the message encoding used when transferring a message across some communication facility.

4.1.40 *recipe* — the pre-planned and reusable portion of the set of instructions, settings, and parameters under control of a processing agent that determines the processing environment seen by the material. Recipes may be subject to change between runs or processing cycles.

4.1.41 *recipe executor* — a component of a module that stores and executes recipes.

4.1.42 *recipe namespace* — a logical management domain with the responsibility for the storage and management of recipes. It ensures the uniqueness of recipe identifiers and provides services pertaining to recipes stored within that domain.

4.1.43 *recovery action* — an operation associated with an error condition with the aim of resolving the abnormal situation detected. It may supply information to the exception agent or request the exception agent to perform some activity.

4.1.44 *service* — the set of messages and definition of the behavior of a service provider that enables remote access to a particular functionality.

4.1.45 *service-provider* — the software control entity that is the provider of any of the related services.

4.1.46 *service-user* — the software control entity that is the user of any of the related services.

4.1.47 *transport module* — a module containing one or more transfer resources and a number of interface planes. It is capable of end-to-end transfer between any of the interface planes.

4.1.48 *tm transfer job* — a material transfer job for a transport module specifying all criteria for receiving the material from the source-attached module and sending the material to the destination module.

4.1.49 *transfer resource* — a module resource within a transport module independently capable of transferring material from one attached module to another.

4.2 Data Type

4.2.1 *form* — type of data: positive integer, unsigned integer, integer, enumerated, boolean, text, formatted text, structure, list, ordered list.

4.2.2 *positive integer* — may take the value of any positive whole number. Messaging protocol may impose a limit on the range of possible values.

4.2.3 *unsigned integer* — may take the value of any positive integer or zero. Messaging protocol may impose a limit on the range of possible values.

4.2.4 *integer* — may take on the value of any negative or unsigned integer. Messaging protocol may impose a limit on the range of possible values.

4.2.5 *enumerated* — may take on one of a limited set of possible values. These values may be given logical names, but they may be represented by any single-item data type.

- 4.2.6 *boolean* — may assume one of two possible values, equating to TRUE or FALSE.
- 4.2.7 *text* — a text string. Messaging protocol may impose restrictions, such as length or ASCII representation.
- 4.2.8 *formatted text* — a text string with an imposed format. This could be by position, by use of special characters, or both.
- 4.2.9 *structure* — a complex structure consisting of a specific set of items, of possibly mixed data types, in a specified arrangement.
- 4.2.10 *list* — a set of one or more items that are all of the same form (one of the above forms).
- 4.2.11 *ordered list* — a list for which the order in which items appear is significant.

5 Conventions

5.1 Harel State Model

5.1.1 This document uses the Harel State Chart notation to describe the dynamic behavior of the objects defined. An overview of this notation is presented in an Appendix of SEMI E30. The formal definition of this notation is presented in *Science of Computer Programming 8*, “Statecharts: A Visual Formalism for Complex Systems,” by D. Harel, 1987.

5.1.2 Transition tables are provided in conjunction with the state diagrams to describe explicitly the nature of each state transition. A transition contains columns for Transition #, Current State, Trigger, New State, Action(s). The “trigger” (column 3) for the transition occurs while in the “current” state. The “actions” (column 5) include a combination of (1) actions taken upon exit of the current state, (2) actions taken upon entry of the new state, and (3) actions taken which are most closely associated with the transition. No differentiation is made.

5.2 *OMT Object Information Model* — The object models are presented using the Object Modeling Technique developed by Rumbaugh, James, *et al.*, in “Object-Oriented Modeling and Design,” Prentice Hall, Englewood Cliffs, NJ, ©1991. Overviews of this notation are provided in an appendix to SEMI E39.

5.3 *Object Attribute Representation* — The object information models for standardized objects will be supported by an attribute definition table with the following column headings:

<i>Attribute Name</i>	<i>Definition</i>	<i>Access</i>	<i>Rqmt</i>	<i>Form</i>
The formal text name of the attribute.	Description of the information contained.	RO or RW	Y or N	(see below)

The Access column uses RO (Read Only) or RW (Read and Write) to indicate the access that service-users have to the attribute.

A ‘Y’ or ‘N’ in the requirement (Rqmt) column indicates if this attribute must be supported in order to meet fundamental compliance for the service.

The Form column is used to indicate the format of the attribute. (See § 4.1 for definitions.)

6 Overview

The Cluster Tool Module Communications standard specifies the communications services necessary to achieve automated control within a cluster tool. It defines the essential cluster tool architecture and the concepts and models on which the communications services are based.

Cluster tools provide a means of grouping a number of independent process steps into a single equipment. The primary function of a cluster tool is material processing. Material enters the cluster tool, undergoes a number of sequential process steps which adds value to the material, and then exits.

From a mechanical viewpoint, Cluster Tool Module Interface standards-compliant equipment has a physical structure where process modules and cassette modules are attached in a standardized way to some form of transport

module, a robotic material handler operating in an isolated environment. Cassette modules provide the input and output of material for the cluster tool, usually in cassettes.

This communications standard enables the distributed control of these cluster tools, and of other cluster tools and multi-resource equipment which can be decomposed into process, transport, and cassette modules, whether environmentally isolated or not. A basic assumption is made that the primary human-machine interfaces are not located at the individual modules.

Communication and control services are specified among service-users and service-providers within a cluster tool. The service-providers are the controlled processing, transfer, intratool port, and intertool port resources in the appropriate modules. The service-users are the application entities responsible for supervisory control and data acquisition within the cluster tool.

The services are fully defined in terms of the service-providers, and as such, do not dictate the architecture of the service-user(s). An example of cluster tool control architecture is shown in Figure 1, where the service-user is a single cluster controller platform providing internal scheduling, human-machine interface, and standardized communications interface to the factory.

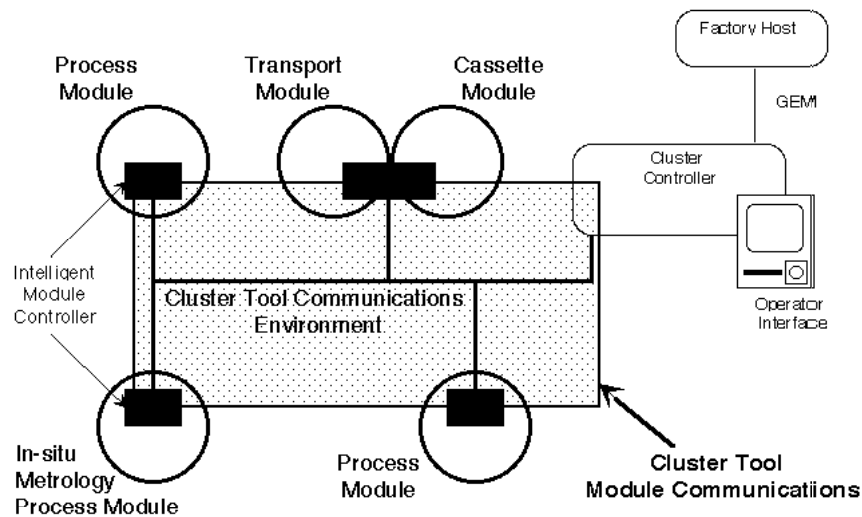


Figure 1
Example Cluster Control Architecture

The communication services include primary control services for material processing in process modules (processing management) and material movement within the cluster tool and material input/output with the factory (material movement management). The cluster controller, which is responsible for supervisory control within the cluster tool, is the service-user of these primary control services. The form of the cluster controller is not dictated. It may be a single platform, distributed, or incorporated with one or more modules.

Support services are also specified to enable resolution of exception conditions, recipe handling, event reporting and data access. The decision authority, which is responsible for making error recovery decisions, is the service-user for exception management. Appropriate agents are the designated service-users for the recipe management and event reporting support services. All service-users access data using the Object Services.

The communications services are defined independently of protocol in order to allow future standardization of alternative protocols without invalidating the services. Figure 2 shows the full communications stack needed for compliance, where the services are mapped to the communications environment, which is defined in supplementary standards.

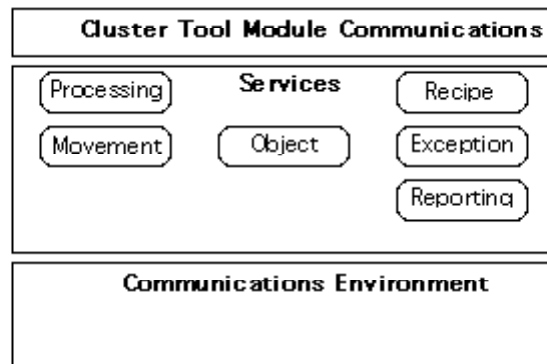


Figure 2
Cluster Tool Module Communications Standards

This standard describes the cluster tool model on which the communications are based. It then defines the application of each of the services required within a cluster tool. This includes limitations imposed by the cluster architecture as well as clear a statement of the fundamental requirements in order to minimize the implementation effort. Connection establishment and maintenance are defined together with the requirements of the communications environment. Communications environment is described in a subsection in order to allow additional options to be standardized in the future. The application notes include factory integration issues and an example scenario of standard communications for the lifecycle of a single material in a cluster tool.

6.1 Compliance — Cluster tool module communications compliance between entities in a cluster tool requires that all applicable exposed interfaces conform to the standards specified in this document.

The standardized interfaces are:

- Process module to Service-user (cluster controller)
- Transport module to Service-user (cluster controller)
- Cassette module to Service-user (cluster controller)
- Transport module to Cassette module
- Transport module to Process module

The exposed interfaces are those which result from a particular partition of the control entities in a cluster tool which are from independent suppliers. Three examples follow:

1. All modules independent, single cluster controller platform: Interfaces are required between the cluster controller and each module, and between the transport module and each process and each cassette module.
2. Material handler, process modules independent, single cluster controller platform: Interfaces are required between the cluster controller and each process module, between the cluster controller and each logical module in the material handler (transport and cassette modules), and between the material handler and each process module.
3. One independent process module integrating to a cluster tool control system: Interfaces are required between the cluster tool control system and the process module supporting the process module to cluster controller communications, and the process module to transport module communications.

For fundamental compliance, each exposed interface shall support the fundamental requirements of each of the service groups required by the entities communicating. The required service groups for each interface are detailed in ¶ 8.6, and the required services for each service group are detailed in ¶¶ 8.1–8.6.

7 Concepts

The Cluster Tool architecture and requirements are modeled in order to extract only the entities necessary for robust automated control and monitoring communications within a cluster tool.

Figure 3 illustrates the major module domains and the primary relationships between them. Module resources and material domains are represented differently to others because they embody the physical entities acting and being acted upon respectively.

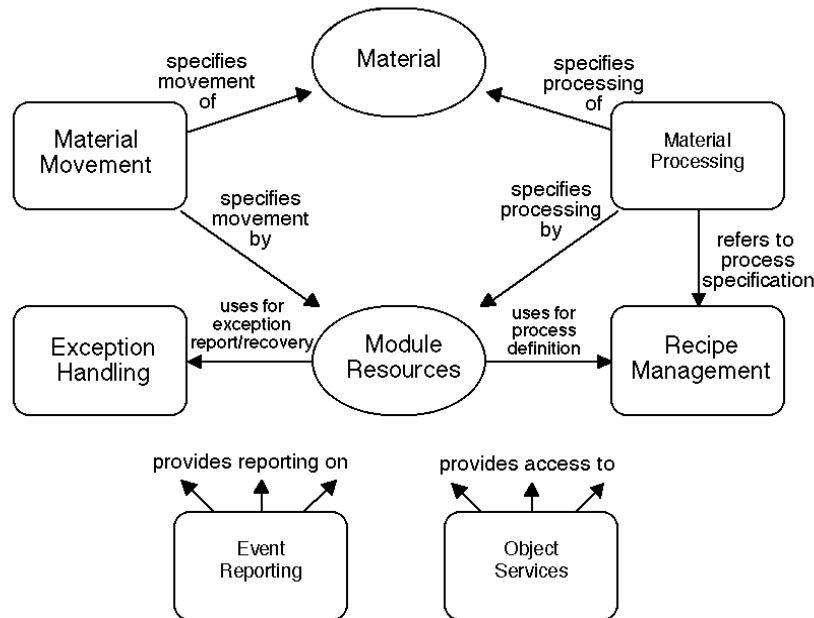


Figure 3
Cluster Tool Module Illustration

The domains illustrated are as follows:

Material — The material, or carrier of material, which is being received and processed in the cluster tool.

Material Processing — Specifies the required application of processing to material as defined by the assigned recipe, adding value to the material.

Material Movement — Includes exchange of material with the factory and material movement within the cluster, to get the material to the correct processing resource at the right time.

Module Resources — Perform and coordinate the processing and material movement activities.

Recipe Management — Provides the services to transfer recipes to be loaded as required for material processing, and manages recipe change control.

Exception Handling — Provides for interactive handling of exception conditions, including reporting and error recovery.

Event Reporting — Enables flexible event-based reporting of variable data as required by service-users.

Object Services — Provide services to obtain and modify standard object attribute data.

Each partition is presented in a subsection below with detailed information models.

The concepts are presented in this section using two types of formal models: OMT object information models describing entities and relationships, and Harel state models describing the behavioral characteristics of the cluster tool. The graphical representations of these models are described in Conventions, § 5.

The events and actions which result from objects dynamically interacting form the basis for the communication requirements embodied in the standards. It is this mapping that specifies the communication services messages to be used.

The major requirement of the cluster tool control system is that it provide robust supervisory control with minimal overhead. Within a cluster tool there is a significant amount of parallel processing. In addition to concurrent processing in multiple independent process modules, each material processing requires the use of recipe management (to get the correct recipe with the correct material) and the use of material movement (to get the correct material to the correct processing resource at the right time).

The cluster controller maintains the overall mission of the system. Submissions are defined for, and responsibility given to, particular available module resources for execution. Once this responsibility is given, that module resource is fully in control, including direct communication with resources of other modules as required to attain the objective.

7.1 Object Information Models — The cluster tool objects relevant to distributed control in a number of models, each emphasizing a particular aspect of the system as partitioned (see Figure 4).

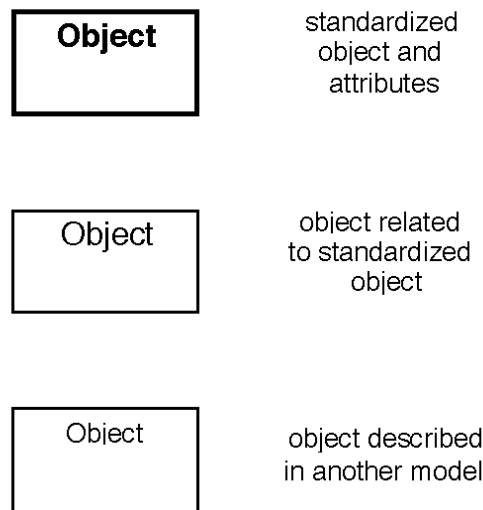


Figure 4
Object Designation

In addition to OMT conventions (see ¶ 5.2), each object in the information models has one of three designations: service objects with standardized attributes (thick border); other objects described in the sub-section (medium border); and objects described in other subsections but included to show significant relationships (thin border).

The models are process- and configuration-independent and do not dictate the internal architecture of the controlled modules beyond the physical requirements of cluster tools.

The first two models show major entities of the processing equipment and material being processed. Subsequent models are oriented towards one of the major functional domains.

7.1.1 Cluster Tool Module Model — The fundamental component of a cluster tool is the module. Figure 5 shows that there are three module types within a cluster tool:

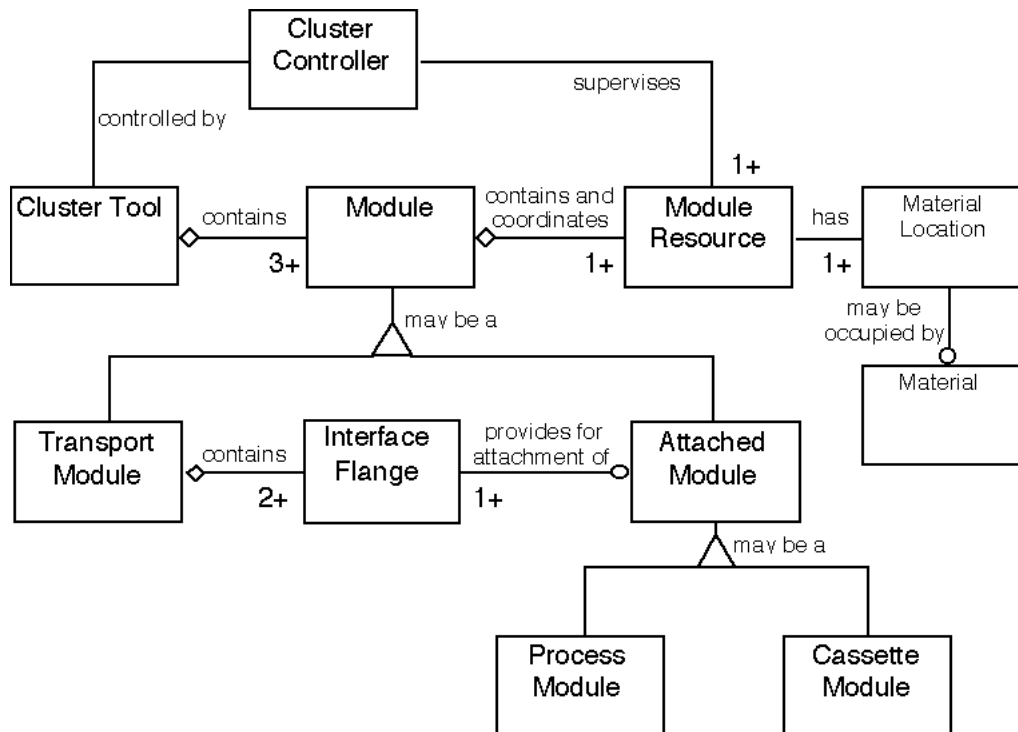


Figure 5
Cluster Tool Module Information Model

- process module (PM) — provides manufacturing value to material
- transport module (TM) — transfers material within cluster
- cassette module (CM) — exchanges material with factory

A cluster tool must contain one or more of each of these module types, as each performs a critical function in the cluster tool mission. Any module can contain processing resources.

Supervisory control within the cluster tool is the responsibility of the cluster controller. The cluster controller achieves this by supervising the various module resources using the standard communications services. It is the primary service-user in the cluster tool. The form of the cluster controller is not specified in this standard. It may be a single platform, distributed, or incorporated with one or more modules.

A module contains and coordinates a number of module resources dedicated to fulfilling a particular control function. Each module resource contains at least one material location, which may or may not contain material. Note that the location for material processing is defined separately from the location where the material is received or sent. This may be virtual if they are physically the same position, but they cannot be assumed to be so as this would exclude certain architectures.

The purpose of each type of module is described in detail below.

Cassette Module — The component of a cluster used to interface the cluster to the rest of the factory. It provides material input and output for the cluster. It makes only one material at a time available to the cluster transport (the intratool environment), regardless of how the material enters the cassette module from outside the cluster (the intertool environment). It is a specialization of attached module (AM).

Process Module — A process module is any component within a cluster which provides one or more steps of material processing for the cluster system. Material enters the process module through an interface flange, undergoes some transformation, and exits through the same or another interface flange. It is also a specialization of attached module (AM).

Transport Module — The transport module is responsible for transfer of material from one cluster-attached module to another. It consists of a robotic handler capable of exchanging material with each of the attached modules at the interface flanges. It can be a variety of physical topologies (radial or linear, for example). In SEMI standard cluster tools, the transport module has an interface isolation valve at each point of attachment to the attached modules. The transport module is responsible for managing constraints on the simultaneous opening of isolation valves as it relates to contamination of the transport module's environment by that of an attached module, as well as cross contamination between attached modules.

The attributes of modules and related objects in Figure 5 are not standardized as these objects are equipment and process-dependent.

7.1.2 Material Model — Material is received from the factory, in a carrier or singly, at an intertool port resource. They are transported within the cluster and processed by processing resources. Once processed, they are returned to the factory at the same, or another, intertool port resource.

The processing which is to be performed on material is determined from the process specification for that material.

The material model in Figure 6 establishes the relationships between the material and material locations. Two material types are of interest in cluster tool module communications, single material and carriers. Single material may be grouped in carriers. This is modeled by defining the material slot object, which is a type of material location. A carrier is made up of an ordered set of material slots. A single material occupies a material slot when in the carrier. In addition, a single material may be assigned a particular material slot which it does not yet occupy, indicating that it is to be received into that material slot when it arrives.

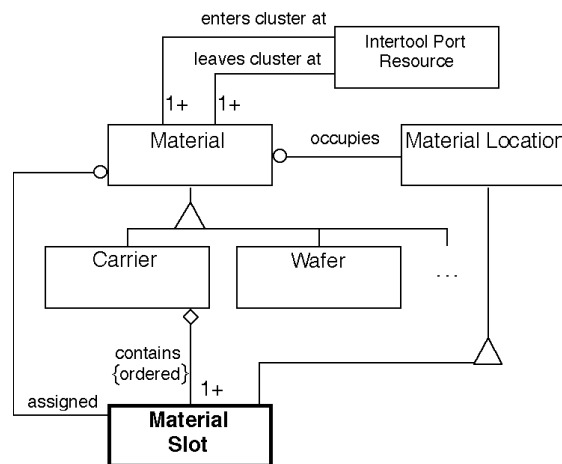


Figure 6
Material Information Model

Carriers may be cassettes used to transport single material in the factory and to input single material to the cluster tool, or they may be material boats in batch process modules.

The attributes of the material slot object are standardized to enable mapping of single material in carriers. Other objects in the diagram are not standardized. However, the identifier of the material, MaterialID, is an attribute of the material slot object.

Other materials (consumable gases and fluids, etc.) are present in the cluster tool, but outside the scope of these standards.

7.1.3 Material Processing Model — The material processing model in Figure 7 shows the use of the process job to direct the processing resource to apply the appropriate process, as specified by the recipe, to one or more materials in a process module. A processing resource in a process module has processing capabilities which are specified by corresponding recipes.

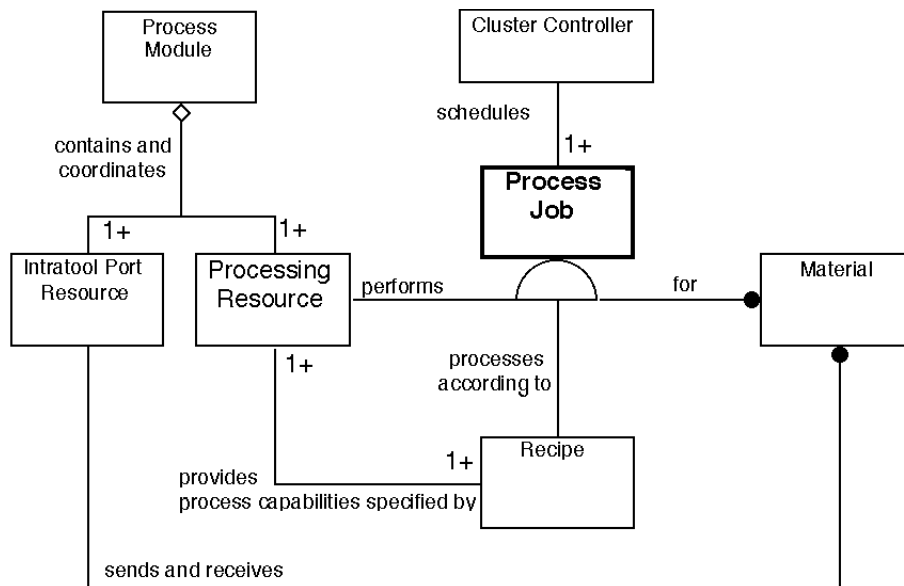


Figure 7
Material Processing Information Model

A process job can control multiple material only if identical processing is applied to all. Processing should begin and end simultaneously, synchronized with the arrival of the whole group of material.

The process job lifecycle extends beyond the active processing of the material. It exists from just before material arrival at the intratool port resource, through setup and processing, and until just after the material departure at the same, or another, intratool port resource. This allows for material-related pre-conditioning and post-conditioning of the processing resource before the material is received and after it is sent. As material input and output is controlled by the intratool port resource, it is the responsibility of the process module to ensure coordination of the intratool port and processing resources.

This is the model used to establish the Processing Management definition. There is no standardization beyond creation and control (starting, canceling, stopping, aborting, and pausing) of the high-level process job since the low-level control of process modules is application-dependent.

There is a component of scheduling embodied in this model. The material, recipes, and processing resources are scheduled by the cluster controller according to the process specification to a particular process module through the use of the process job. In order to perform the required processing, material is transferred to the processing resource, as directed by the cluster controller.

7.1.4 Material Movement Models — Material movement in a cluster tool includes all services required within the cluster tool to move material from the factory to a sequence of process modules in order to be processed and then back to the factory. Material is moved individually within a cluster tool and may be received from the factory in cassettes or individually.

The material movement model is divided into three parts as follows:

- intratool material movement defines the transfer of material between attached modules, with the transfer agent being a transport module.
- intertool material movement defines the exchange of carriers and single material with the factory at the cluster tool cassette module.
- carrier mapping defines presence and identifiers of single material contained in carriers.

These three models are used to establish the Material Movement Management definition within a cluster tool.

7.1.4.1 *Intratool Material Transfer* — The intratool material movement model is shown in Figure 8. An intratool material transfer is the movement of material from one attached module to another attached module through the linking transport module.

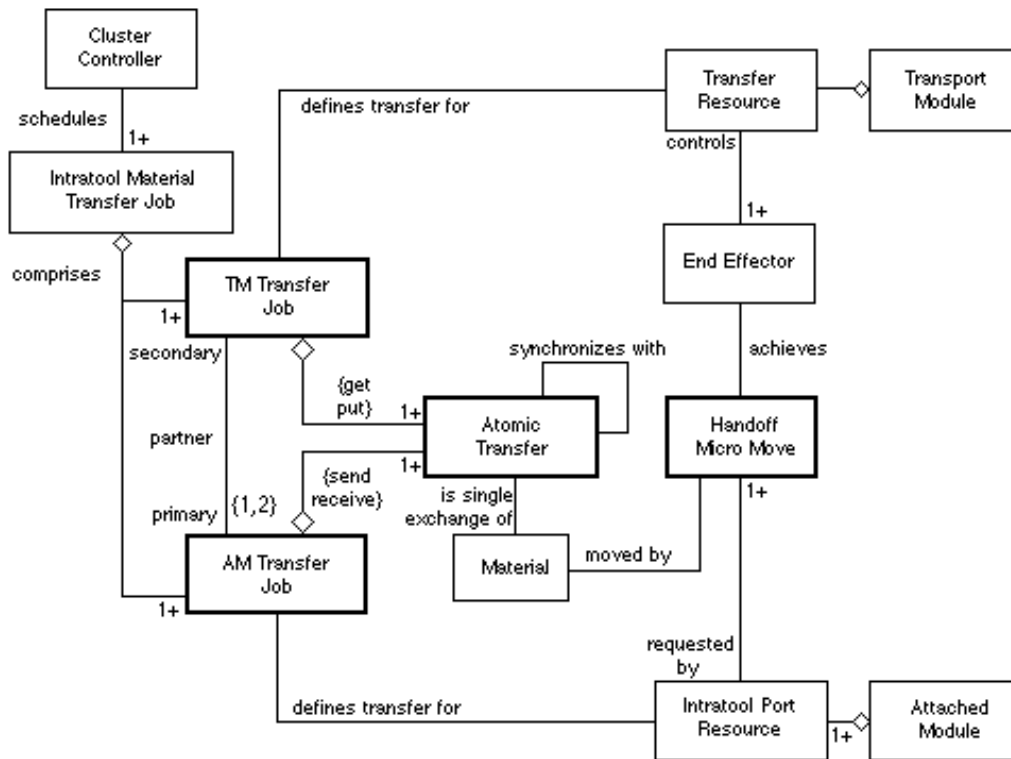


Figure 8
Intratool Material Movement Information Model

The cluster controller schedules an intratool material transfer job when intratool material transfer is required. This is achieved by TM and AM transfer jobs assigned to the appropriate module resource in each of the participating modules:

- to the transfer resource in the transport module to get the material from the source attached module and put it into the destination attached module.
- to the intratool port resource in the source attached module to send the material to the transport module.
- to the intratool port resource in the destination attached module to receive the material from the transport module.

Transfer jobs are made up of atomic transfers. An atomic transfer in one transfer partner synchronizes with the corresponding atomic transfer in the other partner to achieve the material transfer through handoff micro moves. In a cluster tool, the attached module is the primary partner, requesting the handoff micro moves, and the transport module is the secondary partner achieving those micro moves by controlling the end effector on which the material is transported.

The transport module is responsible for managing constraints on the simultaneous opening of isolation valves as it relates to contamination of the transport module's environment by that of an attached module, as well as cross contamination between modules. The attached module ensures that its environment will not contaminate the transport module before synchronizing for handoff. The rules for re-establishing isolation at verification are determined by the transport module.

7.1.4.2 *Intertool Material Transfer* — The cassette module performs material exchange with the factory. The material generally consists of cassettes with single material or empty cassettes, but may be individual single material.

The intertool material movement model, shown in Figure 9, is similar to the intratool material movement model. An intertool material transfer is a transfer in the factory which includes the cluster tool as one of the transfer partners.

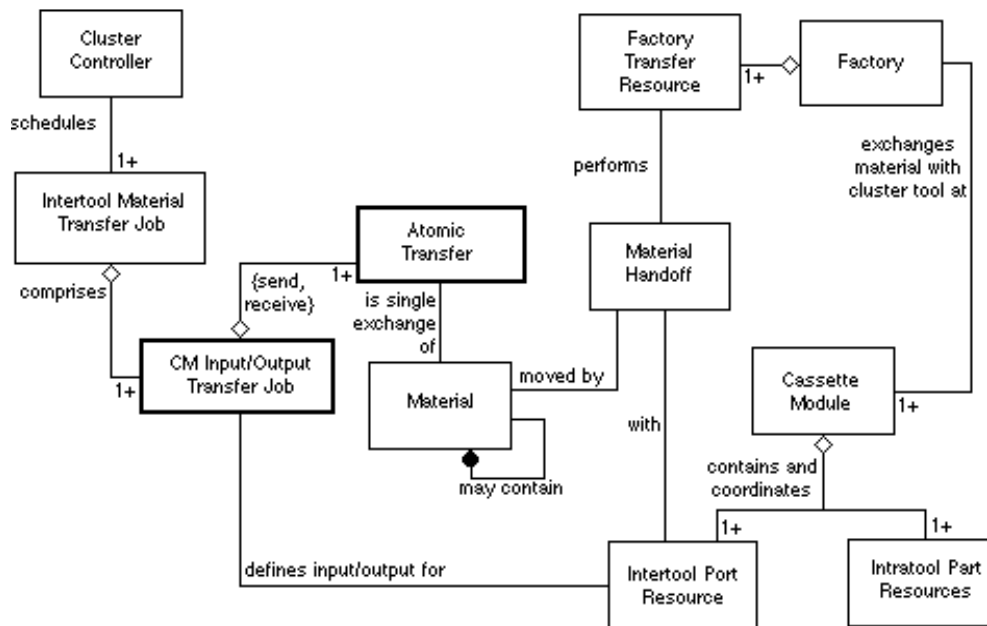


Figure 9
Intertool Material Movement Information Model

In the cluster tool, the cluster controller schedules an intertool material transfer job when material exchange with the factory is required. This job may be an element of a factory material transfer job in which the cluster tool is a partner.

The intertool material transfer job is achieved by a CM input/output transfer job assigned to the intertool port resource in the participating cassette module. It defines the material exchange with the factory for the cassette module. The CM input/output transfer job comprises an atomic transfer, which defines the roles in the material handoff.

In the cassette module, material transfer with the transport module is controlled by the intratool port resource. It is the responsibility of the cassette module to coordinate its intertool and intratool port resources.

The factory transfer resource may be an operator, SMIF, AGV, or other mechanism. Its definition, associated transfer jobs, and the handoff, are beyond the scope of this standard. Factory material movement may be achieved automatically using applicable SEMI standards.

7.1.4.3 Carrier Mapping — The final element of material movement in a cluster tool is the mapping of single material in a carrier. Once a cassette is received by the cluster tool, it is necessary to communicate the presence and identifiers of the material it contains. Batch process modules also require these carrier mapping services in order to define loading of the process carrier.

In carrier mapping, the carrier may be a cassette used to hold single material loaded at the cassette module from the factory, or a process carrier used to hold multiple material for batch processing. For the purposes of carrier mapping, these are equivalent, and the relationships are shown in the material model, Figure 6.

A carrier has an ordered set of material slots, a specialization of material location, each of which can hold a single material. The material slot of a carrier is only of interest when the carrier is in the cluster tool, so it is viewed as a location of the module which contains the carrier. In addition to occupancy, a single material may be assigned to a material slot, indicating, for example, the particular slot to which a single material is to be moved when available. Information on material presence and assignment of the material identifier may be detected in the module or determined by the cluster controller or a supervisory controller.

7.1.5 Exception Model — In an automated control system where the operator interface is remote from the module controller, it is necessary to have interactive exception handling for error recovery. In addition to exception

reporting, a module requires input from a decision authority to resolve recoverable abnormal situations. These include exception conditions which extend beyond the module domain and those for which the module has insufficient information, such as in hardware failure.

The decision authority in a cluster tool is the cluster controller. It may interact with an operator to determine the appropriate recovery action to perform.

Figure 10 shows the exception model. Module resources detect appropriate exception conditions, which may be set or cleared. An abnormal situation is indicated by the corresponding exception condition being in the set state. A significant change in exception condition information generates an exception report to notify the decision authority. Both detection of the abnormal situation and its resolution generate exception reports. Reporting of each exception condition may be enabled and disabled independently in order to mask nuisance exceptions.

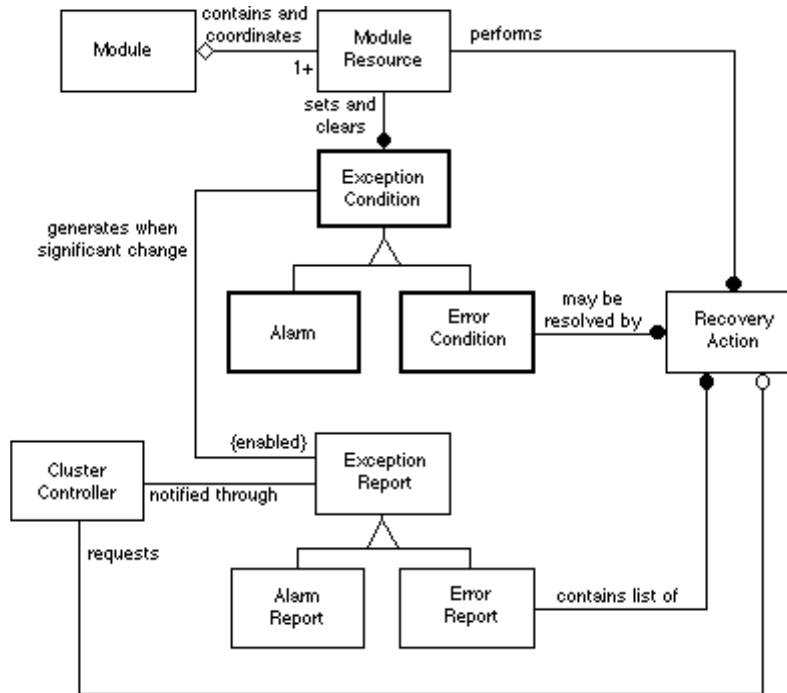


Figure 10
Exception Information Model

Error exception conditions may have associated recovery actions which can be performed by the module resource to attempt to recover from the abnormal situation. Alarm exception conditions cannot, by definition, be resolved using recovery actions.

A list of possible recovery actions is included in a posted error report. The decision authority can request one of the offered recovery actions to resolve the error condition. The selected recovery action is performed by the appropriate module resource and may result in clearing the error condition.

This model is used to establish the Exception Management definition.

7.1.6 Recipe Management Model — Recipe management in a module varies with the requirements of the application. Simple modules do not provide local recipe editing and don't require management of multiple logical domains (namespaces) and local recipe version control.

Figure 11 shows the full recipe management model for a sophisticated module.

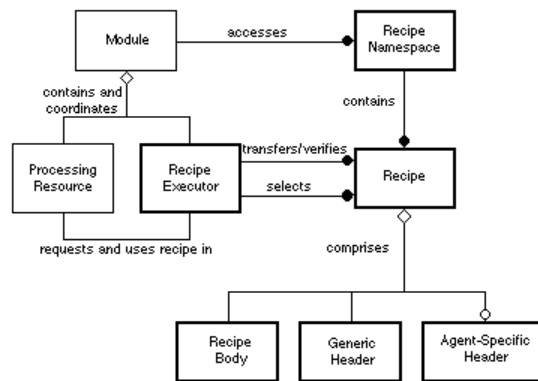


Figure 11
Recipe Management Information Model

Process recipes specify the actual parameters of the processing to be achieved. The module processing resource identifies, through the process job, the required recipe(s) to be loaded by the recipe executor in order to process a particular material.

The recipe executor accesses and selects recipes for execution. It uploads and downloads recipes from the cluster controller and is capable of verifying that a recipe is syntactically correct. The recipe executor also provides for a service-user to select a recipe to be loaded into the recipe execution area. The recipe is made up of a recipe body and may have headers if the module supports recipe namespace management.

The module accesses recipe namespaces where requested by service-users. A module which does not provide local editing and linking of recipes does not need recipe namespace management. Recipe namespace management requires management of recipe generic and agent (module) specific headers. For example, a process module could manage a recipe namespace containing its process and service recipes.

This model is used to establish the Recipe Management services requirements for cluster tool modules.

7.1.7 Event Reporting Model

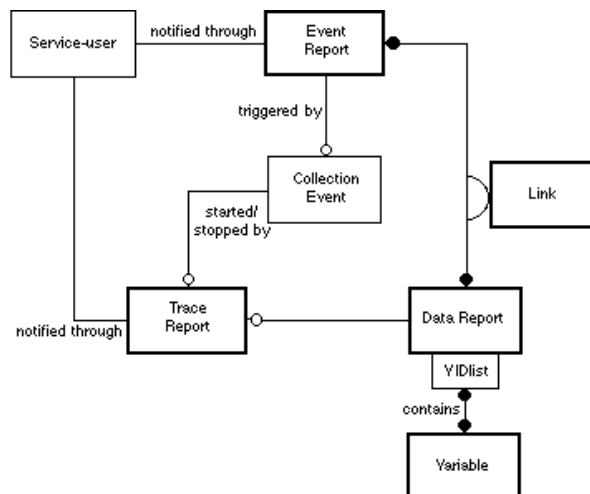


Figure 12
Event Reporting Information Model

Event reporting provides a dynamic and flexible means by which a service-user can receive notification of events and data relating to module resources.

The services-user can define and enable two types of reporting, event reporting and trace reporting. Both may convey data, the values of identified variables at the data collection time. The variable identifiers required to be reported are specified in the data report definition.

In event reporting, data reports are linked to event report definitions associated with each collection event type. On occurrence of the collection event, an event report message is generated according to the event report definition and the linked data report definitions. Reporting may be enabled and disabled for each collection event type. Data reports and links may be predefined or specified dynamically by the service user.

Trace reporting is time-based data reporting. Trace reporting is specified by the service-user dynamically as required. Collection events may initiate and terminate the tracing in order to link reporting to significant conditions, such as processing. The trace data is collected at a frequency specified by the service-user. Trace reports are generated according to the trace report definition and the associated data report definition.

Through associating the event report and trace report definition with the service-user, the model provides for the association of multiple service-users to a single service-provider. This allows for event reporting between modules where needed as well as the ability for direct connection of a data acquisition entity independent of the cluster controller.

7.2 Cluster Tool Module Message Flow — The appropriate objects in the information models in the previous section are grouped in the cluster tool module types. The communications with the essential objects are shown in the message flow diagrams in this section.

The diagrams show message flow between entities within an agent (in this case a module) and those outside the agent. Entities are rectangular boxes, and the agent is a rounded box. As emphasis is on the module and its services, only one agent is shown on each diagram. Outside entities in the cluster tool module communications domain appear within an agent on another diagram. The messaging is shown by arrows in each direction, with a brief label indicating the services requested or provided.

The communications are shown for the primary control of the process module (Figure 13), cassette module (Figure 14), and transport module (Figure 15). Note that the transport module transfer resource communicates with multiple process and cassette module intratool port resources.

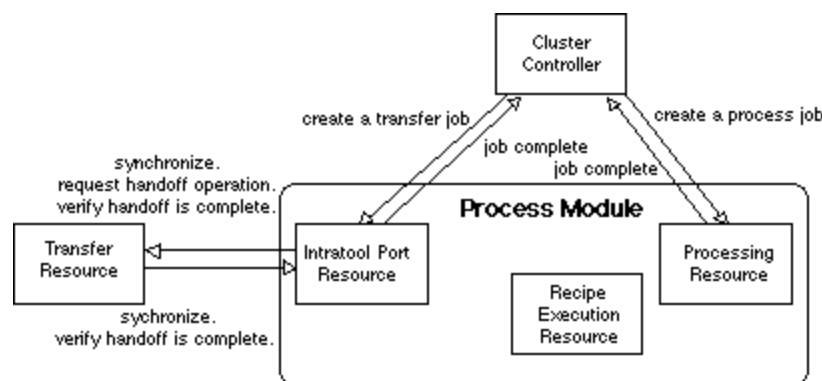


Figure 13
Process Module Primary Control Message Flow