SEMI E94-1000
PROVISIONAL SPECIFICATION FOR CONTROL JOB MANAGEMENT

This provisional specification was technically approved by the Global Information and Control Committee and is the direct responsibility of the North American Information and Control Committee. Current edition approved by the North American Regional Standards Committee on July 14 and August 28, 2000. Initially available at www.semi.org September 2000, to be published October 2000. Originally published February 2000.

1 Purpose

1.1 This specification describes equipment provided services to the factory that supports a high level of factory automation. These services provide capabilities for the host to coordinate processing and disposition of materials on production equipment.

2 Scope

2.1 This specification may be applied to equipment that is compliant to SEMI E30 (GEM). However, it is also intended that this standard will be useful for future generation equipment interfaces that supercede SEMI E30, such as SEMI E53.

2.2 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 limitations prior to use.

3 Limitations

3.1 This standard presents a model of the ControlJob. The model delineates the services (messages) and behavior of the ControlJob. The model is protocol independent. Thus, an ancillary standard must be selected in order to provide a complete implementation.

3.2 This standard should not be applied to non-production equipment, such as, material transport systems or facilities (environmental) controllers.

3.3 *Provisional Status*

3.3.1 For this standard specification to be complete (removal of provisional status) the following areas need to be completed:

1) ControlJob to assure equivalent processing regardless of module used in multi-module equipment.
2) Complete specification of relationship to Carrier Management.
3) Standardized support for the parallel execution of ControlJobs.
4) Support for processing materials on equipment that have batch sizes that are not either multi-carrier or on which the carrier substrate location count is not an integer multiple of the batch size.
5) Possible linkages between ControlJob, material, and substrate tracking.
6) Complete work on the model for the control job queue. Additional functions and attributes for manipulating the job queue may be added.
7) A section on conformance.

3.3.2 The task force plans to finish these areas in 1999.

3.4 This specification applies to equipment for which the atomic unit of material is the same for all input and output carriers on the equipment. It may not apply to equipment which would perform operations such as slicing or assembly that would require or result in different input and output material objects.

4 Referenced Standards

4.1 *SEMI Standards*

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

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

SEMI E40 — Standard for Processing Management

SEMI E53 — Event Reporting

NOTE 1: As listed or revised, all documents cited shall be the latest publications of adopted standards.

5 Terminology

5.1 *control job* — defines a unit of work on equipment for one or more carriers. The work is described by a set of one or more process jobs to be applied to the material contained in the carriers.

5.2 *de-queue* — the act of removing an item from a queue. The de-queue implies nothing about the status of the item after removal.

5.3 *equipment* — the intelligent system that communicates with the host.

5.4 *host* — the intelligent system that communicates with the equipment.

5.5 *life cycle* — the processes and activities of something from its beginning (creation) to its ending.

5.6 *multi-module equipment* — equipment that has more than one distinct processing resource (e.g., chamber).

5.7 *production equipment* — equipment that measures or adds value to the product.

5.8 *protocol independent* — for software, this means that the message descriptions are independent of delivery mechanisms.

5.9 *set-up* — a description of the current process capability of an equipment.

5.10 *substrate* — basic unit of material on which work is performed to create a product. Examples include wafers, lead frames, CD's, die, flat panel displays, circuit boards, and disks.

5.11 *substrate port* — the carrier location from which substrates are accessed by the equipment.

5.12 *uni-carrier* — term for an equipment mode of operation in which all material is returned to the source carrier after processing.

5.13 *user start* — activities that are initiated on a system by another system or operator.

6 Conventions

6.1 *Object Models*

6.1.1 This standard uses object models to specify the control job interface.

6.1.2 *Object Services Standard*

6.1.2.1 This document conforms to the conventions established by SEMI E39.

6.1.3 *Formal Name of an Object*

6.1.3.1 The text capitalizes formal object name references, similar to the way capitalization is normally used when discussing entities. When describing something in the general (like cities) lower case is used, but when a specific entity is of interest (New York City), then first letters are capitalized.

6.2 *State Model Methodology*

6.2.1 This document uses the Harel state chart convention for describing dynamic operation of defined objects. The outline of this convention is described in an attachment of SEMI E30. The official definition of this convention is described in "State Charts: A Visual Formalism for Complex Systems" written by D. Harel in Science of Computer Programming 8, 1987. [1]

6.2.2 The Harel convention does not have the concept of state models of "creation" and "extinction" for expressing a temporary entity. The "job" described in this document is such an entity, and a copy of the same state model is used for an independent job newly created. In this document, a circle with a black circle inside is used for expressing extinction of an entity. A filled, black circle denotes the entry to the state model (the entity creation).

6.2.3 Transition tables are provided in conjunction with the state diagrams to explicitly describe the nature of each state transition. A transition table contains columns for Transition number, Previous State, Trigger, New State, Actions, and Comments. The "trigger" (column 3) for the transition occurs while in the "previous" state. The "actions" (column 5) includes a combination of:

1) Actions taken upon exit of the previous state.

2) Actions taken upon entry of the new state.

3) Actions taken which are most closely associated with the transition.

6.2.3.1 No differentiation is made between these cases.

6.2.4 The state models included in this standard are a requirement for Control Job Management compliance. A state model consists of a state model diagram, state definitions, and a state transition table. When using collection events, all state transitions in this standard, unless otherwise specified, shall correspond to collection events.

6.2.5 A state model represents the host's view of the equipment, and does not necessarily describe the internal equipment operation. When using collection events, all Control Job Management state model transitions shall be mapped sequentially into the appropriate internal equipment collection events that satisfy the requirements of those transitions. In certain implementations, the equipment may enter a state and have already satisfied all of the conditions required by the Control Job Management state models for transition to another state. In the case, the equipment makes the required transition without any additional actions in this situation.

1 Elsevier Science, P.O. Box 945, New York, NY 10159-0945, http://www.elsevier.nl/homepage/browse.htt

Num	*Previous State*	*Trigger*	*New State*	*Actions*	*Comments*

6.3 *Service Message Representation*

6.3.1 Services are functions or methods that may be provided by either the equipment or the host. A service message may be either a request message, which always requires a response, or a notification message, that does not require a response.

6.3.2 *Service Definition*

6.3.2.1 A service definition table defines the specific set of messages for a given service resource, as shown in the following table:

Message Service Name	*Type*	*Description*

6.3.2.2 Type can be either "N" = Notification or "R" = Request & Response.

6.3.2.3 Notification type messages are initiated by the service provider (e.g., the equipment) and the provider does not expect to get a response from the service user. Request messages are initiated by a service user (e.g., the host). Request messages ask for data or an activity from the provider. Request messages expect a specific response message (no presumption on the message content).

6.3.3 *Service Parameter Dictionary*

6.3.3.1 A service parameter dictionary table defines the description, format and its possible value for parameters used by services, as shown in the following table:

Parameter Name	*Description*	*Format: Possible Value*

6.3.3.2 A row is provided in the table for each parameter of a service.

6.3.4 *Service Message Definition*

6.3.4.1 A service message definition table defines the parameters used in a service, as shown in the following table:

Parameter	*Req/Ind*	*Res/Cnf*	*Comment*

6.3.4.2 The columns labeled REQ/IND and RSP/CNF link the parameters to the direction of the message. The message sent by the initiator is called the "Request". The receiver terms this message the "Indication" or the request. The receiver may then send a "Response" which the original sender terms the " Confirmation".

6.3.4.3 The following codes appear in the REQ/IND and RSP/CNF columns and are used in the definition of the parameters (e.g., how each parameter is used in each direction):

M	Mandatory Parameter — Must be given a valid value.
C	Conditional Parameter — May be defined in some circumstances and undefined in others. Whether a value is given may be completely optional or may depend on the value of the other parameter.
U	User-Defined Parameter.
-	The parameter is not used.
=	(For response only) Indicates that the value of this parameter in the response must match that in the primary (if defined).

7 Overview

7.1 This section provides an overview of the control job functionality. It does not contain the specifications which define that functionality.

7.1.1 Control jobs provide a supervisory level of control for process jobs on material processing equipment. They can be used to reduce the amount of host level interaction required for material processing. A factory host is provided with methods for instructing the equipment to provide only significant factory level events, such as, a carrier complete. The ControlJob also supplies methods for the disposition of material after processing.

7.2 *User Requirements*

7.2.1 To handle the complexity required for manufacturing, equipment must support the ability to coordinate its processing services with the factory's needs. The ControlJob provides the services that the factory needs to accomplish this coordination. The requirements that the ControlJob satisfies include: (1) a method by which the equipment coordinates related work, for instance, all process jobs associated with a carrier, and (2) a method by which the equipment can be informed of material destination after processing. The ControlJob is not a type of process job. It is not responsible for the coordination of the processing resource and the material to be processed.

7.2.2 *Initiate and Monitor Process Jobs*

7.2.2.1 ControlJobs are queued. ProcessJobs are not queued by equipment that supports control jobs, rather they are pooled waiting to be scheduled by their respective ControlJob. The ControlJob specifies the order for process jobs. The equipment follows that order as the equipment's resources become available (and when material is available).

7.3 *Supplier Requirements*

7.3.1 *Management of Process Materials*

7.3.1.1 Suppliers need to implement an operational model for managing material and processing in a manner consistent with factory expectations. For instance, the equipment must know when it is finished with a carrier so that it can either allow or signal the factory for the removal of the carrier. This standard provides mechanisms to meet this requirement. While the model implies some implementation it is only the external events that are required by this standard.

7.3.2 *Control Job Events*

7.3.2.1 Control jobs supply information to host systems as either responses to request messages or as events which are sent to the host. Typically the equipment can implement the event mechanisms either in GEM (SEMI E30) or the Event Reporting standard (SEMI E53).

7.3.2.2 All state transitions defined for state models in this document must be able to be reported by separate collection events as defined in section 6.2, State Model Methodology. The state model is the Control Job State Model (Figure 2). The data required for each state model transition event is defined per the following. This data is the minimum required per event. The host may assign other variable, as applicable, from Section 13, Variable Data, of this document, or from other equipment variable data.

7.3.2.3 The following data is required to be available for the Control Job State Model transition collection events:

CtrlJobID

7.4 *Operational Descriptions*

7.4.1 The ProcessJob as referenced in the specification of the control job model is the SEMI E40 process job. Within a ProcessJob the material processing order is managed by the equipment. For some equipment types, the user may be able to configure the material processing order. If available, this feature shall be fully documented by the supplier (see SEMI E40).

7.4.2 To support a simpler interface for single substrate processing, it is suggested to use the PRJobMultiCreate (see SEMI E40) service.

7.4.3 The use of control jobs restricts some SEMI E40 functionality. In particular, the equipment's queue management functionality for process jobs is superseded by the job order as defined in the control job.

7.4.4 The relationship between control jobs and process jobs varies by equipment type. The equipment supplier should document this relationship. In general aborting or stopping a process job does not stop or abort the control job. Equipment is responsible to disposition material correctly depending on how a process job ends. In the case of equipment types that always have a one to one relationship between a control job and a process job it may be convenient for a process job abort or stop to automatically abort the respective control job. In the same sense, if a control job specifies more than one process job, it may be convenient for an abort or stop of all process jobs to automatically abort or stop the respective control job.

8 ControlJob Object Model

8.1 This specification only standardizes the ControlJob object's interface. The other objects provide a context for the ControlJob interface. Since

only the interface is standardized, it is not a requirement for equipment to implement a control job object, it is only required that the equipment provide an external interface that provides the services and behavior defined for the ControlJob.

8.2 *Material to Job Linkage*

8.2.1 The equipment has relationships with many other components not illustrated in Figure 1. In particular, from its knowledge based on substrate and carrier tracking capabilities, the equipment shall connect the process job material list to the materials in the carriers that it has. It is the responsibility of the factory host to make sure that the description or identifiers of material contained in carriers can be mapped to the material identifiers in the process job definition.

8.3 *Control Jobs and Carriers*

8.3.1 A control job may specify work for several carriers. The supplier shall document the behavior of the equipment in the case where a carrier is specified for use in more than one control job.

8.4 *Attribute Definitions*

8.4.1 The attributes in Table 1 shall be accessible using the Object Services standard (SEMI E39). Object services is a set of messages which may be required of any service provider which is modeled by objects. An object model for a service provides a consistent naming convention for exchanging information between the service provider and user. Object services implementations shall be consistent with the service's object and state models. For instance, if an attribute can only be modified in a certain state, then a request to set that attribute when the model is in the wrong state shall be rejected (fail). ControlJob Attributes shall be modifiable if and only if the ControlJob is not in either the EXECUTING or COMPLETED states by using OSS to change them (see Section 13.1.3).

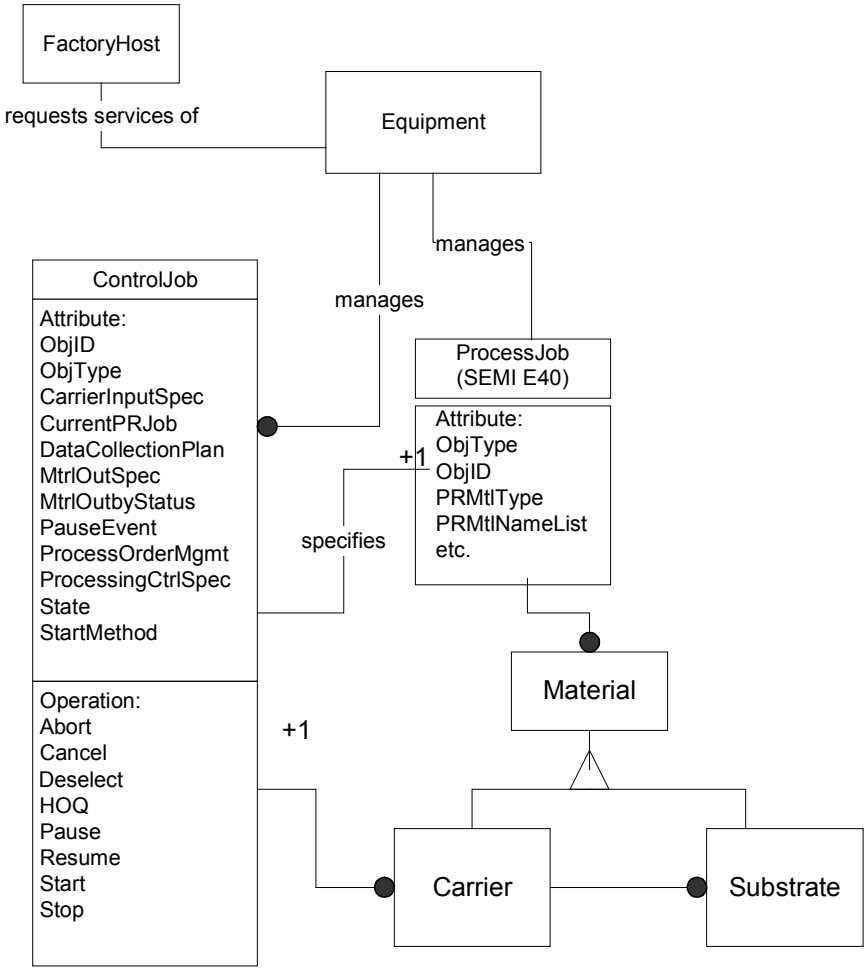

Figure 1
ControlJob Object Model

Table 1 ControlJob Attributes

Name	*Definition*	*Rqmt*	*Access*	*Form*
ObjID	Host defined identifier of the control job.	Y	RO	Text
ObjType	Object type	Y	RO	Text = 'ControlJob'
CurrentPRJob	Holds the identifiers for any currently running process jobs, even if job is paused.	Y	RO	(list of) PRJobID (see SEMI E40)
DataCollectionPlan	Identifier for a data collection plan to be used during execution of the control job.	N	RW	Text
CarrierInputSpec	A list of carrierID for material that will be used by the ControlJob. An empty list is allowed.	Y	RW	(list of) CarrierID
MtrlOutSpec	Maps material from source to destination after processing. For uni-carrier operation, the list shall be empty. The list shall also be empty, if CarrierInputSpec is an empty list.	Y	RW	List of Structure: SourceMap DestinationMap
MtrlOutByStatus	List structure which maps locations or Carriers where processed material will be placed based on material status.	N	RW	List of Structure: Destination MaterialStatus
PauseEvent	Identifier of a list of events on which the Control Job shall PAUSE.	N	RW	(list of) EventID
ProcessingCtrlSpec	A list of structures that defines the process jobs and rules for running each that will be run within this ControlJob.	Y	RW	(list of) Structure: PRJobID ControlRule OutputRule
ProcessOrderMgmt	Define the method for the order in which process jobs are initiated.	Y	RW	Enumeration: LIST ARRIVAL OPTIMIZE
StartMethod	A logical flag that determines if the ControlJob can start automatically. A user start may come through either the host connection or the operator console.	Y	RO	Boolean: TRUE – Auto FALSE – UserStart
State	The current state of the ControlJob.	Y	RO	Enumerated: per State Model

8.4.2 A number of the ControlJob attributes are composite data types. The constituent data is defined in Table 2.

Table 2 Attribute Data Definitions

Data Identifier	*Description*	*Form*
CarrierID	The identifier of a carrier that is the source or destination for substrates.	Text
ControlRule	Provides additional job control functionality. It is equipment type dependent. It may be used to modify processing based on processing results. Use of this attribute is not required for equipment, which does not support it. Suppliers shall document the use of this attribute when supported.	(list of) Structure: RuleName RuleValue
Destination	The identifier of a substrate location at which material can be placed. (Identifier should conform to standards for substrate tracking.)	Text
DestinationMap	Describes carrier positions into which finished material will be placed. If the list of carrier positions is empty, then follow sequential order of source.	Structure: CarrierID List of SubstrateLocation

Data Identifier	*Description*	*Form*
MaterialStatus	ControlJob processing assigns this value to finished material. The association of MaterialStatus to Destination enables ControlJob processing to put material at the desired destination.	Equipment dependent enumeration
OutputRule	Defines the MaterialStatus (such as Good, Reject, Aborted, Monitor, etc.) based on results of the process job.	Equipment dependent
PRJobID	A process job identifier as defined by SEMI E40. Host must supply same name in the ProcessingCtrlSpec as when it requested creation of process job. NOTE: SEMI E40 process jobs link material to a recipe.	See SEMI E40.
Rule Value	The value used by the equipment for execution of a control rule.	Equipment dependent
RuleName	Identifier of a control rule.	Text
SourceMap	Describes the locations from which material is taken for processing. If the list of location is empty, then assume the default of ascending order.	Structure: CarrierID List of SubstrateLocation
SubstrateLocation	A substrate position at a source and a destination. A carrier is an example of a multi-location destination. For a wafer carrier the SubstrateLocation is a slot number.	numeric

8.4.3 *ControlRule*

8.4.3.1 For equipment that supports this attribute (field), the host sets this in order to achieve better host processing control capabilities. For example, the host may have previously measured characteristics of the material to be processed. A standard recipe is used based on the product and process step, but based on the measured characteristics, the application of the recipe is biased by the specified rule and the value that is passed to the rule (RuleValue). However, use of ControlRule should not be considered to be limited to only this type of application.

8.4.4 *DataCollectionPlan*

8.4.4.1 The DataCollectionPlan is a name given by the host to associate data collection activities to a specific control job. In general, it provides a way for the equipment to then inform and coordinate with the host to receive data collection requests. A DataCollectionPlan is generic and will be applied to many control jobs. The variable itself, DataCollectionPlan, will hold no significance for the equipment. It is simply a label the equipment reports back when requested by the host. Normally, the host upon receiving the ControlJob START event would include DataCollectionPlan as a data variable to be reported. The host then knows that the time is appropriate to set up various trace reports and event reports on the equipment. Potentially, all jobs that specify the same product type and process capability could specify the same DataCollectionPlan.

8.4.5 *OutputRule*

8.4.5.1 This attribute can only be supported by equipment that has some means to determine the status of material that it has processed. For equipment with that ability, the rule will usually take the form of a list of name value pairs. The names will be material status and the values will be measurement thresholds that correspond to the status category (such as, Good, Reject, Rework, etc.). Substrate (material) status changes should be recorded in substrate histories created by the equipment.

8.4.5.2 Equipment which also supports the MtrlOutbyStatus shall use the status determined by the OutputRule to place substrates at the Destination associated with MaterialStatus.

8.4.6 *PauseEvent*

8.4.6.1 For equipment which can support it, this attribute contains a list of equipment events, specified by the host, at which the host expects the equipment to PAUSE the ControlJob. Equipment suppliers shall document any events that can be used for the pausing of control jobs. Pausing a control job causes it to stop initiating process jobs. The host might use this to stop processing after one or more process jobs has started in order to await results before processing the rest of the material in the control job.

8.4.7 *ProcessOrderMgmt*

8.4.7.1 This is an enumerated attribute that defines the order in which processing will occur. This standard defines three enumerations. For some equipment other enumerations may be possible. If they are the supplier shall document them.

8.4.7.2 LIST

8.4.7.2.1 When ProcessOrderMgmt is set to this value, process jobs shall be initiated in the order of the list in ProcessingCtrlSpec.

8.4.7.3 ARRIVAL

8.4.7.3.1 When ProcessOrderMgmt is set to this value, process jobs shall be initiated as the material for the job arrives. Any jobs that do not require material will be initiated first.

8.4.7.4 OPTIMIZE

8.4.7.4.1 When ProcessOrderMgmt is set to this value, process jobs shall be initiated in an order to be determined by internal equipment algorithms, that optimize the throughput of material in the equipment.

9 Control Job State Model – Behavior

9.1 The following state chart defines the behavior of the ControlJob.

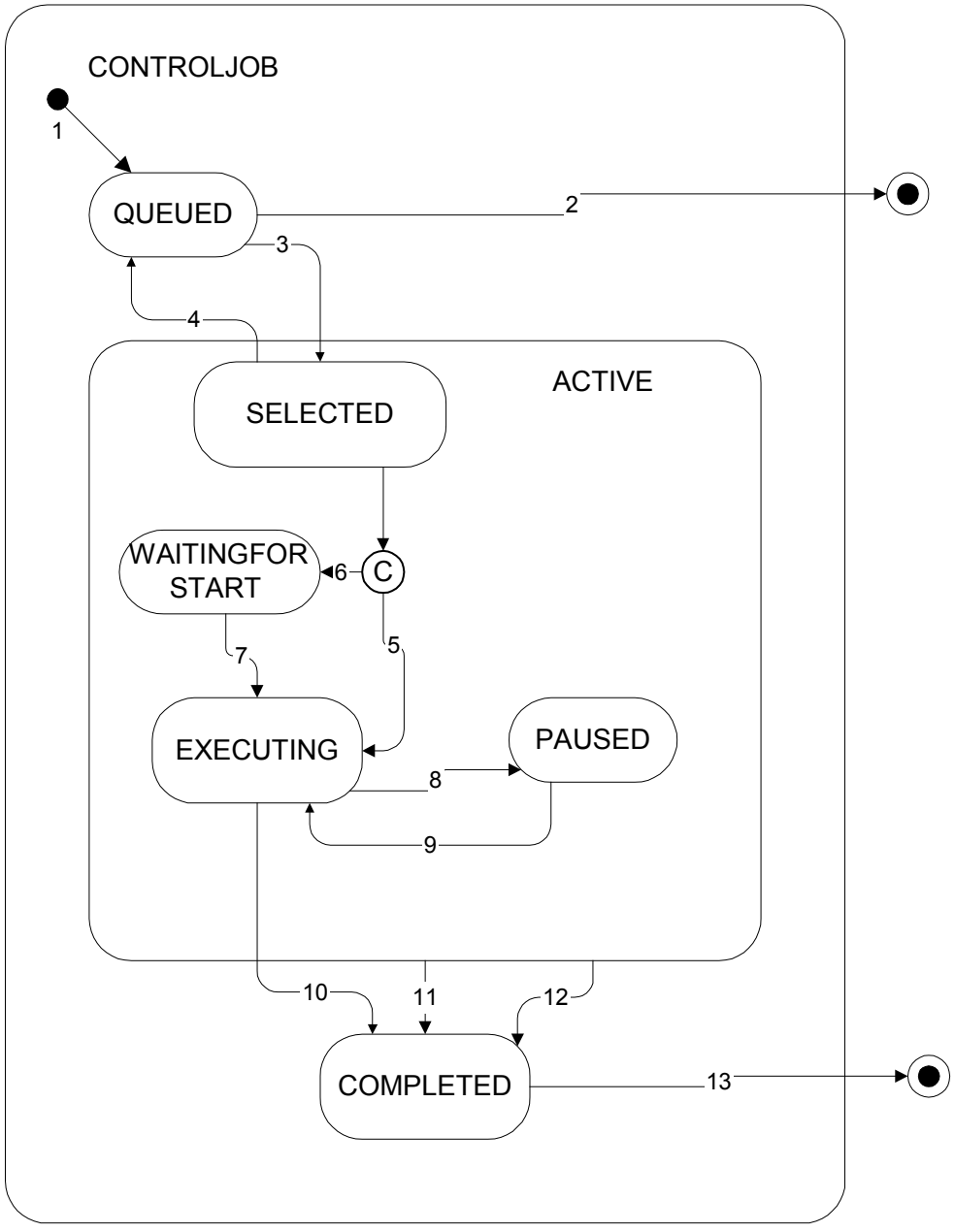

Figure 2
Control Job State Model

9.2 *State Definitions*

9.2.1 QUEUED — A ControlJob is queued after its creation or de-selection. A newly created ControlJob is placed at the tail of the queue.

9.2.2 SELECTED — In this state, the ControlJob does not initiate process jobs specified in it and therefore pre-defined (based on recipe variable parameters) process conditions can be modified. The processing resource is reserved (not available for any other jobs) by the ControlJob in the SELECTED state. If materials, required by the ControlJob, for processing have not arrived at the equipment, the ControlJob will stay in this state until materials arrive. If the ControlJob or the first process job in the ControlJob does not require material, this state is exited immediately. A SELECTED ControlJob can be de-selected if specified materials have not arrived.

9.2.3 WAITING FOR START — The ControlJob is waiting to receive a start command manually or remotely from the host. The ControlJob transitions to this state only if the StartMethod is set to FALSE (UserStart) and materials have arrived.

9.2.4 EXECUTING — In this state, each process job in the ProcessingCtrlSpec is initiated in order, based on the value of the ControlJob's ProcessOrderMgmt attribute as required resources become available and material for the job has been verified. Process jobs that have been initiated but that are WAITINGFORSTART or PAUSED shall block the availability of the resources that they require (see SEMI E40).

NOTE 2: Process jobs that have blocked available resources shall cause the ControlJob to stop initiating subsequent process jobs that use those resources.

9.2.5 PAUSED — When the ControlJob is paused it shall not commence the initiation of any more Process jobs. In this state, Process jobs that have not entered the "PROCESSING" state can be modified. Various attributes of the ControlJob can also be modified. This is equipment specific and shall be documented by the equipment supplier.

9.2.6 COMPLETED — A control job enters this state once all of its process jobs have been completed, stopped or aborted. In this state a control job can respond to requests for its attribute values.

Table 3 ControlJob State Transition Table

Num	*Previous State*	*Trigger*	*New State*	*Actions*	*Comments*
1	(No state)	Receive "Create" command from host or operator through operator console.	QUEUED	Create ControlJob and put it at the tail of a control job queue.	If job queue is full, "Create" request is rejected.
2	QUEUED	Receive "Cancel", "Abort", or "Stop" command from host or operator through operator console.	(No state)	De-queue and terminate the job. Send a "ControlJob Canceled" event to the host.	If other control jobs are waiting behind the canceled job in the queue, they are shifted forward to fill in the gap after the de-queuing of the canceled control job.
3	QUEUED	The processing resource has capacity to begin work on the next ControlJob.	SELECTED	Select and de-queue the job at the head of the queue. Send a "Selected" event to the host.	Materials are not necessarily at the equipment.
4	SELECTED	Receive "De-select" command from host or operator through operator console and materials for the control job have not arrived yet.	QUEUED	De-selected job moves to the head of the job queue and the job that was at the head becomes the SELECTED job.	The command shall be rejected if the resources for the job at the head of queue are not available. See the Queue Model.
5	SELECTED	Material for the first process job arrives or in the case where the first (or only) process job does not require material, this transition shall be taken as soon as the processing resource for that	EXECUTING	Send "Execution began" event to the host.	Process jobs associated with a carrier will not initiate until the identifier and substrate slot map for the carrier have been verified. Process jobs that don't use material can be initiated

Num	Previous State	Trigger	New State	Actions	Comments
		process job becomes available. "StartMethod" attribute in the ControlJob is set for Auto.			immediately.
6	SELECTED	Same as for transition 5 except that the "StartMethod" attribute in the control job is set for user start.	WAITING FORSTART	Send a "JobWaiting for Start" event to host and/or operator.	
7	WAITING FORSTART	User START command received.	EXECUTING	Same as for transition 5.	Same as for transition 5.
8	EXECUTING	Received "Pause" message from host or operator through operator console or a ControlJob. PauseEvent has occurred.	PAUSED	Send a "Paused" event to the host.	Process jobs which have not started can be modified in this state.
9	PAUSED	Receive "Resume" message from host or operator through operator console.	EXECUTING	Commence initiating process jobs. Send a "Resumed" event to the host.	
10	EXECUTING	All the ProcessJobs specified for the ControlJob have completed.	COMPLETED	Send a "Complete" event to the host.	It may include post processing completion.
11	ACTIVE	Receive "CJStop" message from host or operator through operator console or all the process jobs under the ControlJob have been stopped and material processing is stopped.	COMPLETED	Send a "Stopped" event to the host.	
12	ACTIVE	Receive "CJAbort" command from host or operator through operator console or all the process jobs under the ControlJob have been aborted and material processing is aborted.	COMPLETED	Send "Aborted" message to the host.	
13	COMPLETED	The ControlJob is deleted.	(No state)		Equipment should perform this function automatically for COMPLETED jobs after they have persisted for at least one day.

10 Control Job Queue Model

10.1 The Queuing mechanism for control jobs will generally operate under FIFO (First in- First Out) constraints. The commands used to monitor queue status and prevent deadlock conditions are specified here.

10.2 *Queue Integrity*

10.2.1 To maintain queue integrity, only one operation shall be performed at any given time (e.g., the "Create" request shall be rejected by the equipment if the CJHOQ service is being processed). The Queue is defined to be "locked" while it is performing an operation (and refusing any further operation until completion of the current operation).

10.3 *Head of Queue Service*

10.3.1 The Head of Queue service (CJHOQ) shall operate under the following rules:

1) All control jobs positioned between the specified control job and the head of the queue (including the job currently positioned at the head) will be moved back one position. The specified control job will then be moved into the head of queue position.

2) When the CJHOQ command is invoked, the queue will be "locked" to maintain integrity.

3) In the case where only one control job exists in the queue, the command will perform no action on the queue.

10.4 *DeadLocks*

10.4.1 The Head of Queue service (CJHOQ) requests a specific control job to be set as the next control job to be run. In order to prevent deadlock when the job at Head of Queue and the job in the SELECTED state are both awaiting material delivery, the CJHOQ command may be used to move a different job to the head of the queue position. The potential dead lock is then broken by issuing the DE-SELECT service request.

10.5 *Utilization of Queue for Control Job Priority Management*

10.5.1 Similar to Deadlock, certain cases may arise where a job at the head of the queue cannot be selected due to a lack of processing resources. In this case, DE-SELECT command shall be rejected and consequently a series of DE-SELECT and CJHOQ commands may be issued in an attempt to find a job which can transition to SELECTED state.

10.5.2 To force a "hot job" to be the next job run, it could be necessary to send the CJStop message to the SELECTED job. This case only happens when the job at the head of the queue (the hot job) does not yet have resources available.

NOTE 3: Most equipment where this is possible can support parallel execution of Control Jobs. Management of the queue(s) in this case is currently beyond the scope of this standard.

10.6 *Space in the Queue*

10.6.1 The QueueAvailableSpace variable data item is used to query the number of control job openings within the queue. The QueueAvailableSpace shall function according to the following rules:

1) This variable can only be guaranteed valid when no other operations are being performed simultaneously on the queue. For example, don't request this variable while a Create control job command is being processed by the equipment.

2) This variable shall be incremented whenever a control job in the queue is de-queued. That is, when a "Cancel", "Abort", or "Stop" command has been received and completed while the control job is queued. It should also be incremented when the SELECTED state is entered by a control job. However, if this transition occurs as a result of the "Deselect" command (at least one job in queue, and a control job in the SELECTED state), no change should be made to the variable value.

3) This variable should be decremented whenever a control job joins the queue. That is, when a "Create" command is received and accepted. It should also be decremented if a "Deselect" command is issued on a control job in the SELECTED state and no other jobs currently reside in the queue.

4) The equipment should reject the "Create" command when this variable is equal to zero.

10.7 *Getting a List of Queued Jobs*

10.7.1 The QueuedCJobs Variable Data Item is used to query the names of the control jobs currently residing in the queue. It lists items starting at the head of the queue.

10.7.2 The QueuedCJobs Variable Data Item shall function according to the following rules:

1) This variable can only be guaranteed valid when no other operations are being performed simultaneously on the queue.

2) This variable list will be modified whenever a successful "Create" command is received. Additionally, "Cancel", "Abort", and "Stop" commands issued on control jobs residing in the queue will modify this variable list. Any use of the "Deselect" command will also modify the variable list.

11 Properties for Carriers

11.1 Compliance to Control Job Management requires that the equipment track the status of individual carriers. In particular, the factory needs to know the status of carriers with respect to control jobs. Carriers may have various properties that are beyond the scope of this standard. However, there are specific properties that are needed for Control Jobs.

11.2 *Carrier Verification*

11.2.1 The equipment needs to know when a carrier has been verified as proper. Only process jobs associated with the carrier that has been verified shall be initiated by a control job. Depending on the equipment's capabilities, verification may include verification by an equipment read of the carrier's ID (identification) and the reading of the substrate (e.g., wafer) slot map. The ControlJob determines if the carrier is verified by checking the carrier's attributes that indicate the level to which a carrier has been verified. This section is reserved for further specification of this requirement.

11.3 *Carrier Completion for Control Jobs*

11.3.1 A carrier that has been loaded onto the equipment may go through three stages: first, it is in the "not processed" stage until it is accessed by a control job. When the carrier is at the substrate port and unloading of the substrates within the carrier begins, the carrier enters the "in process" stage. Once it enters this stage, it remains in this stage until all material has been returned to the carrier, no active control job exists that is using it and no control jobs in the queue exist that reference it. The equipment shall provide a property of the carrier that shows the current stage of all carriers. The equipment shall provide a "CarrierCompletion" event for each change in stage. The event when the carrier enters the "in process" stage informs the host that the carrier may not be removed from the equipment. The event when the carrier enters the "completed' stage informs the host that the carrier may be removed.

12 Requirements - Service Definitions

12.1 *Service Definitions*

Table 4 Service Definitions Table

Message Service Name	*Type*	*Description*
CJStart	R	To start a ControlJob.
CJPause	R	To request a ControlJob to pause.
CJResume	R	To request a PAUSED ControlJob to go to the EXECUTING state.
CJCancel	R	To request a ControlJob to be removed from the queue.
CJDeselect	R	To request a ControlJob to be deselected; it will no longer be the next job to run.
CJStop	R	To request a ControlJob to stop. Used to discontinue a job without risk to the material.
CJAbort	R	To request a ControlJob to abort. Used to discontinue a job on equipment that may be malfunctioning. Material is at risk when this command is issued.
CJHOQ	R	To request a particular ControlJob to be set as the next control job to be SELECT'ed.

12.2 *Parameter Definitions*

Table 5 Parameter Definitions Table

Parameter Name	*Description*	*Format: Possible Value*
ACKcode	To return indication of result of service call.	Enumeration: SUCCESS, FAILURE
Action	Directs activity of the control job service with regards to its process jobs when Cancel, Abort, and Stop requests are made. In many cases it is convenient to remove (delete) all the process jobs specified in the Control Job.	Enumeration: SAVEJOBS REMOVEJOBS
CtrlJobID	ObjID (object identifier) of a control job.	Text
ErrorCode	Contains the code for the specific error found.	Enumerated (ACKcode must equal FAILURE): *All services:* • Unknown object instance • Parameters improperly specified • Insufficient parameters specified *CJStart:* • Command not valid for current state *CJPause:* • Command not valid for current state *CJResume:* • Command not valid for current state *CJCancel:*

Parameter Name	*Description*	*Format: Possible Value*
		• Job cancelled • Command not valid for current state *CJDeselect:* • Command not valid for current state • Busy (when queue empty or resources for HOQ job would not be available) *CJStop:* • Job stopped *CJAbort:* • Job aborted *CJHOQ:* • Command not valid for current state
ErrorInfo	The parameter may be null or excluded on a SUCCESS.	(List of) ErrorCode ErrorText
ErrorText	Description of the error.	Text
Status	Information returned by service provider which indicates the result of the service call.	Structure: ACKcode ErrorInfo

12.3 *Message Details*

12.3.1 This section specifies parameter usage by the service messages.

12.3.2 *Creating ControlJobs* — ControlJobs shall be created by using the OSS (SEMI E39) Object Create message. The following table defines the use of the AttrSetting arguments to the Object Create service. Note: ObjType is a required argument of Object Create and therefore should not be reset by including it as an AttrSetting argument. In the table M indicates mandatory, O indicates optional, and R specifies restricted (shall be ignored if used).

12.3.3 The process jobs specified for a control job must exist prior to calling this message. If a process job does not exist, the Create service shall fail and the ObjStatus shall return a list of any PRJob identifiers that were not present. The Create request shall be rejected if the ControlJob queue is full. Newly created control jobs will be put at the end of the queue.

Table 6 SetAttr Arguments Table

Control Job Attribute Name	*Use as AttrSetting of Create Service*
ObjID	M
ObjType	R
CurrentPRJob	R
DataCollectionPlan	O
CarrierInputSpec	M
MtrlOutSpec	M
MtrlOutbyStatus	O
PauseEvent	O
ProcessingCtrlSpec	M
ProcessOrderMgmt	M
StartMethod	M
State	R

12.3.4 *CJStart* — Starts jobs that require a user start. The host sends this command only to a Control Job from which it has received a WAITINGFORSTART event.

Table 7 CJStart Service Parameter Definitions Table

Parameter	*Req/Ind*	*Rsp/Cnf*	*Comment*
CtrlJobID	M	M	Indicate which job
Status	-	M	Success or failure

12.3.5 *CJPause* — The ControlJob shall stop initiating process jobs. Process jobs in the EXECUTING state are not affected by this command.

Table 8 CJ Pause Service Parameter Definitions Table

Parameter	*Req/Ind*	*Rsp/Cnf*	*Comment*
CtrlJobID	M	M	Indicate which job
Status	-	M	Success or failure

12.3.6 *CJResume* — The ControlJob shall resume initiating process jobs.

Table 9 CJResume Service Parameter Definitions Table

Parameter	*Req/Ind*	*Rsp/Cnf*	*Comment*
CtrlJobID	M	M	Indicate which job
Status	-	M	Success or failure

12.3.7 *CJCancel* — Used to remove a ControlJob from the Queue. The command shall only succeed for jobs in the QUEUED state.

Table 10 CJCancel Service Parameter Definitions Table

Parameter	*Req/Ind*	*Rsp/Cnf*	*Comment*
CtrlJobID	M	M	Indicate which job
Action	M	-	
Status	-	M	Success or failure

12.3.8 *CJDeselect* — Shall only succeed for jobs in the SELECTED state. Deselected jobs must trade places with the job that is currently at the head of the queue. If the job at the head of the queue cannot transition to the SELECTED state, then the deselect request shall be rejected. See the section on Control Job Queue Model for information on breaking possible deadlocks.

Table 11 CJDeselect Service Parameter Definitions Table

Parameter	*Req/Ind*	*Rsp/Cnf*	*Comment*
CtrlJobID	M	M	Indicate which job
Status	-	M	Success or failure

12.3.9 *CJStop* — Stops the ControlJob from initiating any more process jobs. Equipment should issue a STOP command to all running process jobs. When the currently running process jobs have stopped, the ControlJob will send a complete event with a status code indicating the ControlJob has stopped. ControlJobStop shall only succeed on a job in the ACTIVE or QUEUED states. When CJStop is issued in the QUEUED state, its affect will be identical to that of CJCancel.

Table 12 CJStop Service Parameter Definitions Table

Parameter	*Req/Ind*	*Rsp/Cnf*	*Comment*
CtrlJobID	M	M	Indicate which job
Action	M	-	
Status	-	M	Success or failure

12.3.10 *CJAbort* — Stops the control job from initiating any more process jobs. The currently running process jobs are sent the Abort command by the equipment. When the equipment has detected the successful ABORT of currently running process jobs, the ControlJob shall send a complete event with a status code indicating the job was aborted. ControlJobAbort shall only succeed on a job in the ACTIVE or QUEUED states. When CJAbort is issued in the QUEUED state, its affect will be identical to that of CJCancel. The equipment due to a serious alarm situation (operator risk) may internally generate this command.

Table 13 CJAbort Service Parameter Definitions Table

Parameter	*Req/Ind*	*Rsp/Cnf*	*Comment*
CtrlJobID	M	M	Indicate which job
Action	M	-	
Status	-	M	Success or failure

12.3.11 *CJHOQ* — The other jobs in the queue are pushed back (rest of queue order remains unchanged).

Table 14 CJHOQ Service Parameter Definitions Table

Parameter	*Req/Ind*	*Rsp/Cnf*	*Comment*
CtrlJobID	M	M	Indicate which job
Status	-	M	Success or failure

13 Variable Data

13.1 The following table provides the definition of additional Variable Data that equipment shall support for Control Job Management.

Table 15 Variable Data Definitions Table

Variable Name	*Description*	*Type*	*Access*	*Comment*
CtrlJobID	Control job identifier, available to be used in control job related event reports.	Text	RO	
DataCollectionPlan	Used to report the ControlJob's DataCollectionPlan attribute. This information may be useful in the Collection Event that reports 'Execution began' for a ControlJob.	Text	RO	Support for this is optional.
QueuedCJobs	This is an ordered list of control jobs currently in the Queue. The first job in the list is the job at the head of the Queue.	(list of) Text	RO	Each list item is a control job identifier.
QueueAvailableSpace	Indicates number of jobs which the Queue can accept.	Numeric	RO	This value cannot be negative. When it is zero it indicates that the queue is full.
SetUpName	Host sets this to define the operational condition of the equipment.	Text	RW	If the equipment is manipulated locally, it should set this variable to "unknown", otherwise it returns the value set by the host (when requested).

14 Additional Requirements

14.1 *Serial Execution of Control Jobs*

14.1.1 Control jobs are initiated in sequential order by the equipment. The order is based on the queue. A ControlJob shall not issue a complete event (message) until all substrates have been placed in destination carriers. However, in many cases equipment must support multiple control jobs running at once; in order to support the factory requirement for equipment productivity. This will be particularly true for multi-module equipment. The next ControlJob in the queue shall start as soon as possible after processing has begun for the last ProcessJob in the previous ControlJob's processing control specification.

14.2 *Parallel Execution of Control Job*

14.2.1 Some equipment may be able to support parallel execution of control jobs. The supplier must fully document this behavior and any additional services needed to manage it.

14.3 *Modifying Control Jobs*

14.3.1 Control jobs shall be modifiable if and only if they are not in either the EXECUTING or COMPLETED states. Jobs shall be modified by using OSS to change their attributes. Modifications shall be rejected if the equipment is in the wrong state or requested value changes are out of range.

14.4 *Set-up, Pre- and Post-Conditioning*

14.4.1 Whenever equipment has completed some processing work, with or without material, the equip-ment can be considered to be "set-up" for a certain process capability. Information about the equipment's set-up is important to the factory in determining the best material routing. The SetupName variable defined in this standard is set by the host after host directed processing or changes to equipment constants. If the equipment is used for processing while off-line or not under host command, the value of the variable shall be set to "unknown". It shall also be set to "unknown" immediately after any changes to equipment constants.

14.5 *Event Relationships*

14.5.1 This section is reserved for specification of the relationship between process job events and control job events.

14.5.2 PRJob Paused

14.5.3 PRJob Aborted or Stopped

15 Compliance

15.1 Implementations compliant to this standard shall implement all the messages as specified in Section 12. All mandatory parameters must be supported. The supplier shall document support for any of the optional parameters. Any additional parameters and messages shall be fully documented by the supplier. Additional messages shall be used to support additional functionality and not as a replacement for any of messages specified herein.

15.2 Table 16 provides a checklist for Control Job Mangement (CJM) compliance.

Table 16 CJM Compliance Statement

Fundamental CJM Requirements	*CJM Section*	*Implemented*	*CJM Compliant*
Control Job Object	8	☐ Yes ☐ No	☐ Yes ☐ No
Control Job State Model	9	☐ Yes ☐ No	☐ Yes ☐ No
Control Job Queue Model	10	☐ Yes ☐ No	☐ Yes ☐ No
Carrier Properties	11	☐ Yes ☐ No	☐ Yes ☐ No
Service Message Implementation	12	☐ Yes ☐ No	☐ Yes ☐ No
Variable Data	13	☐ Yes ☐ No	☐ Yes ☐ No
Events	7.3.2	☐ Yes ☐ No	☐ Yes ☐ No
Additional Requirements	14	☐ Yes ☐ No	☐ Yes ☐ No
Additional CJM Capabilities	*CJM Section*	*Implemented*	*CJM Compliant*
none			

RELATED INFORMATION 1
APPLICATION NOTES

NOTE: The material contained in these Applications Notes is not an official part of SEMI E94 and is not intended to modify or supersede the official standard. Rather, these notes are auxiliary information describing possible methods for implementing the protocol described by the standard and are included as reference material. The standard should be referred to in all cases. SEMI makes no warranties or representations as to the suitability of the material set forth herein for any particular application. The determination of the suitability of the material is solely the responsibility of the user.

Service messages are presented with a "C" language style of application interface. For illustrative purposes values for variables may be included using the "=" in the argument list. If not used the optional arguments are not shown. Service names (e.g., PRJob) are prefixed to function name. The scenarios assume processing is for wafers. The services used in these scenarios are PRJob for Processing Management, ControlJob for Control Job Management, and CMS for Carrier Management Services.

R1-1 ControlJob for a Batch Processing Tool

R1-1.1 Tool processes the contents of a single carrier as a batch. This example demonstrates the simplicity of using control jobs for a simple situation. Wafer order is maintained, and material is returned to the source carrier.

#	*Comment*	*Dir*	*Message*	*CJS*	*PJS*
1	Create the process job.	H->E	PRJobCreateEnh (PRJobID=prj01_04, Mtrl = CS001, RecID=ILD3)	No state	No state
2		H<-E	PRJobCreateAck (PRJobID, PRJobStatus)		In POOL
3	Request a control job. Material out specification maintains wafer order from source carrier to destination carrier.	H->E	ControlJobCreate (CtrlJobID=cjf01_01, ProcessingCtrlSpec= (prj01_04, null,null), MtrlOutSpec=(CSA01,null,null), MaterialIn= CS001, StartMethod=AUTO)		
4	Request accepted.	H<-E	ControlJobCreateAck (CtrlJobID=cjf01_01, JobStatus)	QUEUED	
5	No CJ in selected state so the newly created job immediately transitions.	H<-E	Event (CJSELECTED, CtrlJobID=cjf01_01)	SELECTED	
6	A carrier for the selected control job arrives.	H<-E	Event (CARRIERIDREAD, CID=CS001)		
7	The equipment recognizes it and starts execution of the ControlJob.	H<-E	Event (ControlJobStart, cjf01_01)	EXECUTING	
8	ControlJob starts the ProcessJob; begins loading wafers to the processing boat.	H<-E	Event (PRJOBSETUP, prj01_01)		ACTIVE/ SETUP
9	Material processing starts after all material is in the processing boat.	H<-E	Event (PRJOBPROCESSING, prj01_04)		ACTIVE/ PROCESSING
10	Equipment begins to return material to the source carrier (= destination carrier).	H<-E	Event (PROCESSINGCOMPLETE, PRJob=prj01_04)		ACTIVE/ PROCESSING COMPLETE
11	Carrier is filled with wafers.	H<-E	Event (CarrierComplete=CS001)		
12	Host wants to get it.	H->E	Rcommand (CarrierOut=CS001)		

#	Comment	Dir	Message	CJS	PJS
13	Equipment indicates carrier can now be picked up.	H<-E	Event (ReadytoUnload, CarrierID=CS001, PortID)		
14		H<-E	Event (PRJOBCOMPLETE, PRJob=prj01_01,)		No state
15	Jobs may complete before the carrier is picked up.	H<-E	Event (ControlJobCompleted=cjf01_01, status=OK)	COMPLETE	

R1-2 ControlJob for a Single Wafer Processing Tool

R1-2.1 Will be added later.

R1-3 ControlJob for Single Wafer Processing with Recipe Variable Parameters

R1-3.1 Will be added later.

R1-4 Error Recovery of Batch Tool ControlJob, Carrier Slot Map Failure

R1-4.1 Will be added later.

R1-5 Carrier Swap During Processing

R1-5.1 Multiple carriers are loaded to a batch processing tool that has buffering, after carriers are emptied, they are removed and new empty carriers are loaded. In this example, it requires four carriers for a batch.

CJS = Control Job State, PJS = Process Job State

#	Comment	Dir	Message	CJS	PJS
1	Create the process job.	H->E	PRJobCreateEnh (PRJobID=prj01_01, Mtrl = CSA01,CSA02, CSA03, CSA04, RecID=ILD1)	No state	No state
2		H<-E	PRJobCreateAck (PRJobID, PRJobStatus)		In POOL
3	Request a control job. Material out specification maintains wafer order from source carrier to destination carrier.	H->E	ControlJobCreate (CtrlJobID=cjf01_01, ProcessingCtrlSpec= (prj01_01, null,null), ProcessOrderMgmt = ARRIVAL, MtrlOutSpec=(((CSA01,null),(CSB01, null)), ((CSA02, null), (CSB02,null)), ((CSA03, null), (CSB03,null)), ((CSA04, null), (CSB04,null))), MaterialIn= (CSA01,CSA02, CSA03, CSA04), StartMethod=AUTO)		
4	Request accepted.	H<-E	ControlJobCreateAck (CtrlJobID=cjf01_01, JobStatus)	QUEUED	
5	No CJ in selected state so the newly created job immediately transitions.	H<-E	Event (CJSELECTED, CtrlJobID=cjf01_01)	SELECTED	
6	A carrier for the selected control job arrives.	H<-E	Event (CARRIERIDREAD, CID=CSA01)		
7	The equipment recognizes it and starts execution of the ControlJob.	H<-E	Event (ControlJobStart, cjf01_01)	EXECUTING	
8	ControlJob starts the	H<-E	Event (PRJOBSETUP, prj01_01)		ACTIVE/SETUP

#	Comment	Dir	Message	CJS	PJS
	ProcessJob; begins loading wafers to the processing boat.				
9		H<-E	Event (CarrierEmpty,CSA01)		
10	The next carrier arrives.	H<-E	Event (CARRIERIDREAD, CSA02)		
11		H->E	Rcommand (CarrierOut=CSA01)		
	As carriers are emptied they are removed.		Steps 9 through 11 are repeated 3 more times		
12	Material processing starts after all material is in the processing boat.	H<-E	Event (PRJOBPROCESSING, prj01_01)		ACTIVE/ PROCESSING
13	Output carriers begin to arrive.	H<-E	Event (CARRIERIDREAD, CID=CSB01)		
	Until all output carriers arrive.		Step 13 is repeated 3 more times		
14	Equipment begins to load output carriers.	H<-E	Event (PROCESSINGCOMPLETE, PRJob=prj01_01)		ACTIVE/ PROCESSING COMPLETE
15	First output carrier is filled with wafers.	H<-E	Event (CarrierComplete=CSB04)		
16	Host wants to get it.	H->E	Rcommand (CarrierOut=CSB04)		
17	Gets rest of carriers.		Steps 15 and 16 are repeated 3 more times		
18		H<-E	Event (PRJOBCOMPLETE, PRJob=prj01_01,)		No state
19		H<-E	Event (ControlJobCompleted=cjf01_01, status=OK)	COMPLETE	

R1-6 Using Cleaning Wafers

R1-6.1 For this scenario we assume a single wafer processing tool, such as an RIE. The tool has three fixed load ports; two for product material, one for cleaning material. A cleaning wafer is run before the 1st and 13th wafer of each carrier of product wafers. Show the load and unload of the cleaning wafers and the running of the ControlJob for processing material. Note that control jobs do not provide functionality for dispositioning collateral material consumed during processing (the cleaning wafers). The equipment is responsible for providing mechanisms to determine when this collateral material has been consumed (should be replaced).

Step #	Comment	Dir	Message	CJS	PJS
1	Remove the previously spent wafers.	H<-E	Event (ReadytoUnload, PortID=Cleaning, CarrierID=CC01)	No state	No state
2	After host picks up the spent wafers, the equipment is ready to load.	H<-E	Event (ReadytoLoad, PortID=Cleaning)		
3	Delivered material is identified.	H<-E	Event (CarrierIDRead, CarrierID=CC02)		
4	Host directs carrier to be moved to the wafer access position.	H->E	CMSProceedwithCarrier (CarrierID=CC02)		
5	Create a job to run a cleaning wafer.	H->E	PRJobCreate (PRJobID=prj01_01, Mtrl = ACleanWafer, RecID=CleaningProcess)		
6		H<-E	PRJobCreateAck (PRJobID, PRJobStatus)		In POOL

Step #	*Comment*	*Dir*	*Message*	*CJS*	*PJS*
7	Create the jobs for the first 12 wafers in the carrier (PW1-12).	H->E	PRJobDuplicateCreate (PRJobSpecList = (prj01_02, PW1), (prj01_03, PW2), …(prj01_13, PW12), RecID = P64ME5, Start=AUTO, MtrlType=WAFER)		
8		H<-E	PRJobCreateAck (PRJobIDList, PRJobStatus)		In POOL
9	Another cleaning wafer	H->E	PRJobCreate (PRJobID=prj01_14, Mtrl = ACleanWafer, RecID=CleaningProcess)		
10		H<-E	PRJobCreateAck (PRJobID, PRJobStatus)		In POOL
11	The rest of the product wafers	H->E	PRJobDuplicateCreate (PRJobSpecList = (prj01_15, PW13), (prj01_16, PW14), …(prj01_27, PW25), RecID = P64ME5, Start=AUTO, MtrlType=WAFER)		
12		H<-E	PRJobCreateAck (PRJobIDList, PRJobStatus)		In POOL
13	Now the control job	H->E	CtrlJobCreate (CtrlJobID=cj01_01, ProcessingCtrlSpecList = (prj01_01, null, null), (prj01_02, null, null), …(prj01_27, null, null), CarrierInputSpec = CP01, Start = AUTO, ProcessOrderMgmt = LIST, MtrlOutSpec = (CP01, null, CP01,null)		
14		H<-E	CtrlJobCreateAck (ID = cj01_01, Status=OK)	QUEUED -> SELECTED	
15		H<-E	Event (ReadytoLoad, PortID = P1)		
16		H<-E	Event (CarrierIDRead, CarrierID=CP01)		
17		H->E	CMSProceedwithCarrier (CP01)		
18		H<-E	Event (CtrlJobStart = cj01_01)	EXECUTING	
	Host might not even have these sent.		Lots of PRJob Start and End Events		
19		H<-E	Event (CtrlJobComplete = cj01_01)	COMPLETED	
20		H<-E	Event (ReadytoUnload, CarrierID = CP01)		