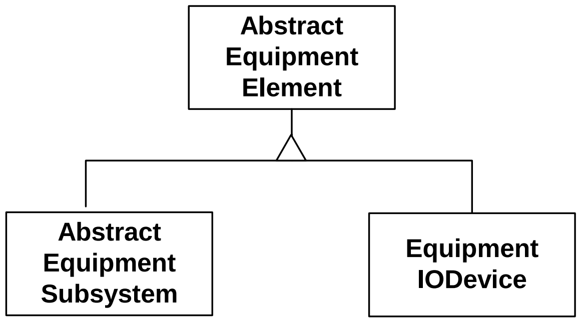


Figure 6
AbstractEquipmentElement Object Types

10.2.3 *AbstractEquipmentElement State Model*

10.2.3.1 The AbstractEquipmentElement is either IN SERVICE (available for work) or OUT OF SERVICE (unavailable for work). In some cases, the user may be able to set the AbstractEquipmentElement's operational state.

10.2.3.2 The state diagram for the AbstractEquipmentElement state model is shown in Figure 7.

10.2.3.3 The AbstractEquipmentElement shall be fault-free whenever it is in the IN SERVICE state.

10.2.3.4 An AbstractEquipmentElement in the OUT OF SERVICE state shall not be used by the equipment or the user for normal manufacturing purposes (while the Equipment or containing EquipmentModule is in the ARAMS superstate of MANUFACTURING).

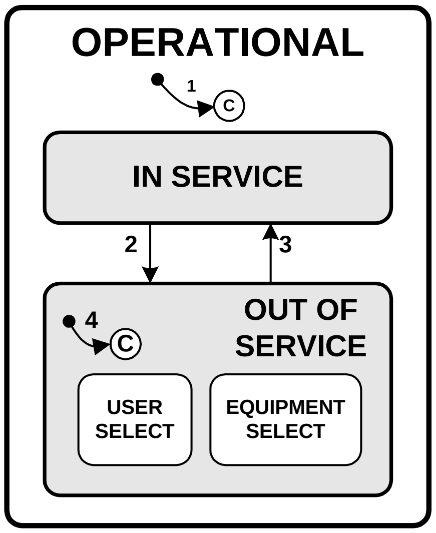


Figure 7
AbstractEquipmentElement State Model

10.2.3.5 The user may or may not be allowed to put a certain subtype of AbstractEquipmentElement out of service, depending upon the equipment design. This is optional for the equipment. In general, the user requires the ability to put major components into and out of service but does not require this for lower level components. In many cases, a component can not operate without some or all of its component elements.

10.2.3.6 Substates of OUT OF SERVICE are not required unless the user is able to put the AbstractEquipmentElement out of service. However, if a user places an AbstractEquipmentElement out of service, only a user shall be able to return it back in service. The substates are then required to retain the source of the out of service selection through the substates USER SELECT and EQUIPMENT SELECT.

10.2.3.7 *State Model Definitions* — Table 2 defines the states of the AbstractEquipmentElement object.

Table 2 AbstractEquipmentElement State Definitions

State Name	*Superstate*	*Definition*	*Comment*
IN SERVICE	OPERATIONAL	The AbstractEquipmentElement is error-free and may be used for work.	None
OUT OF SERVICE	OPERATIONAL	The AbstractEquipmentElement has one or more exceptions and/or has been made unavailable by the equipment or User.	None
USER SELECT	OUT OF SERVICE	The user requested the Abstract-Equipment-Element be put out of service.	Stays out of service until the user returns it to service. This substate may be omitted for low level elements.
EQUIPMENT SELECT	OUT OF SERVICE	The equipment determined that the AbstractEquipmentElement should be placed out of service.	Stays out of service until the equipment determines it is able to operate properly. This substate may be omitted for low level elements.

10.2.3.8 *State Transition Table* — Table 3 defines the state transitions of the AbstractEquipmentElement object.

Table 3 AbstractEquipmentElement State Transitions

#	*Previous State*	*Trigger*	*New State*	*Action(s)*	*Comment*
1	INITIALIZATION	Initialization is complete.	Depends upon previous state and current condition	None	If the user made the element unavailable, it shall remain unavailable. If the element has a fault, it shall be made unavailable by the equipment.
2	IN SERVICE	The user has placed the element OUT OF SERVICE by the user or the equipment.	OUT OF SERVICE	None	The equipment should put the element out of service whenever a fault condition exists at the element.
3	OUT OF SERVICE	The element has been returned to service by the entity that placed it out of service. All exceptions and any user restriction have cleared.	IN SERVICE	None	The original out of service condition has cleared. The element may be used for work.
4	IN SERVICE or INITIALIZING	Default entry into OUT OF SERVICE	USER SELECT or EQUIPMENT SELECT	None	

10.2.4 *AbstractEquipmentElement Object* — Figure 8 represents the AbstractEquipmentElement object.

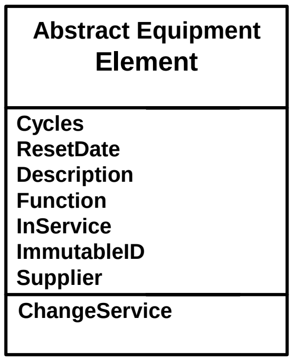


Figure 8
AbstractEquipmentElement Object Type

10.2.5 *AbstractEquipmentElement Attributes* — Table 4 defines the attributes of the AbstractEquipmentElement.

Table 4 AbstractEquipmentElement Attribute Definitions

Attribute Name	*Definition*	*Access*	*Reqd*	*Form*
ObjType	Object type	RO	Y	Text="EqpElement"
ObjID	Object identifier	RO	Y	Text
Cycles	Number of cycles (or hours) since installation or last reset.	RO	Y	Unsigned long integer
ResetDate	Date cycles set to zero.	RO	Y	Text
Description	User-definable text	RW	Y	Text
Function	Describes function of object (e.g., substrate-chuck, apply/develop track).	RO	Y	Text
Supplier	Name of manufacturer or provider	RO	Y	Text
ImmutableID	An unchangeable identifier such as serial number.	RO	Y	Text
InService	Available for work.	RO	Y	Boolean

10.2.6 *AbstractEquipmentElement Services* — The user may be allowed to change the operational state of specific AbstractEquipmentElement objects that are of significance for preventive maintenance. This is particularly desirable for those subsystems for which preventive maintenance may be performed while the equipment is still assigned to manufacturing operations.

10.2.6.1 *ChangeService*

10.2.6.1.1 The user may request that an element currently in the IN SERVICE state be placed in the OUT OF SERVICE\ USER SELECT state. This request shall not affect any current activity for which the element is being used. However, an element in the OUT OF SERVICE state shall not be used for any new activities.

10.2.6.1.2 The user may request that an element currently in the OUT OF SERVICE/USER SELECT state be placed back into the IN SERVICE state.

10.3 *EquipmentIODevice Object*

10.3.1 An EquipmentIODevice object is a specialization of AbstractEquipmentElement used to represent the sensors, actuators, and intelligent sensor/actuator devices that provide most of the process data sought by manufacturing engineering. Individual I/O points are represented by individual Observables contained within the EquipmentIODevice object.

10.3.2 Observables are not formalized as standardized objects, as indicated by the thin line in Figure 9. The EquipmentIODevice may represent a single I/O point, an aggregation of I/O points, or a virtual sensor providing composite data.

10.3.3 Table 5 defines the attributes of the EquipmentIODevice. Attributes are in alphabetical order, with the exception of the Observables$_i$ attribute, which is placed at the end with its structure members for clarity.

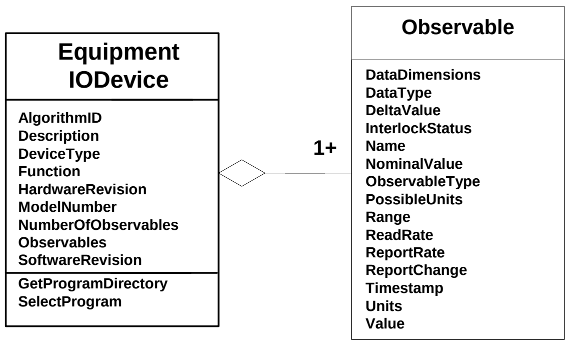


Figure 9
EquipmentIODevice and Observable Object Types

Table 5 EquipmentIODevice Attribute Definitions

Attribute Name	*Definition*	*Access*	*Reqd*	*Form*
ObjType	Object type	RO	Y	Text = “EqpIODevice”
ObjID	Object identifier	RO	Y	Text
AlgorithmID	Identifier of any algorithm used to produce the set of observable values.	RO or RW	Y	Text
HardwareRevision	Revision level for hardware.	RO	Y	Text. 1–5 characters.
ModelNumber	Manufacturer model number.	RO	Y	Text. 1–20 characters.
Numberof-Observables	The number of data items provided	RO	Y	Integer
SoftwareRevision	Revision level for s/w or firmware	RO	Y	Text. 1–5 characters.
Observables$_I$	Information about the ith item of observable data values.	RO	Y	Structure comprised of Name, Value, Units, and Range, and Timestamp as follows.
Name	The name of the data, sensor, or actuator.	RO	Y	Text
DataDimension	Used for arrays to indicate the number and length of “rows” of data.	RO	N. Required for arrays.	List of unsigned integers.
Datatype	The form of the data.	RO	Y	Enumerated.
DeltaValue	Amount of change from the target for reporting. Zero value indicates it is not used.	RW	N	As indicated by Datatype.
InterlockStatus	Status of current interlock.	RO	N. Required for Actuator types.	Enumerated: None, Off, On
NominalValue	Target used for reporting change of DeltaValue.	RW	N	Numeric text string. If empty, is not used.
ObservableType	Type of Observable.	RO	Y	Enumerated
Possible Units	List of possible units	RO	N	List of text
Range	Valid interval or set of possible discrete values.	RO	Y	Text
ReadRate	Interval between sequential reads, in seconds.	RO	Y	Floating point
ReportChange	A user-settable switch to enable or disable reporting.	RW	N	Boolean. Set TRUE when enabled.
ReportRate	Interval between sequential reports, in seconds.	RW	N	Floating point. Zero indicates no time-based reporting.
Timestamp	The date and time when the value was last read or set.	RO	Y	Numeric. Specific form to be resolved.
Units	Units of measurement	RO or RW	Y if value is scalar.	Text. Conforms to units specified in SEMI E5 (SECS-II) Units of Measurement Identifiers.
Value	Current value.	RO	Y	Single item or array of numeric, or text
DeviceType	Defines the type of device (e.g., “MFC”, “OES”, FTIR”, “CDG”, etc.)	RO	Y	Text 1–8 characters

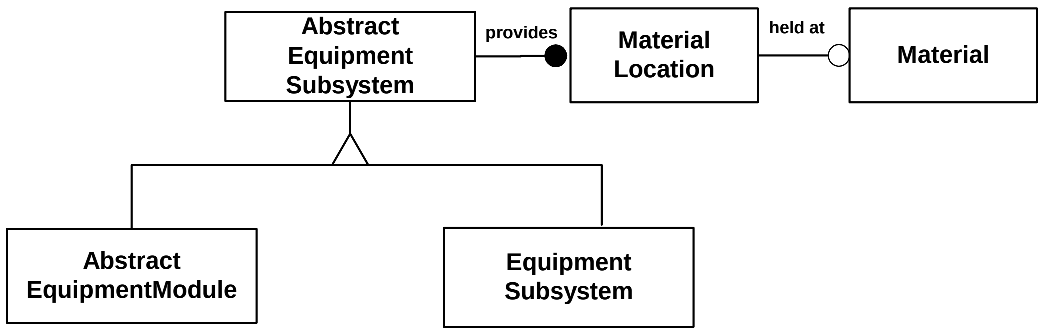


Figure 10
AbstractEquipmentSubsystem Subtypes

10.4 *AbstractEquipmentSubsystem Object*

10.4.1 The AbstractEquipmentSubsystem is a subtype of AbstractEquipmentElement and inherits all of its attributes and services. It may have associated MaterialLocations to hold Material. AbstractEquipmentSubsystem represents all subsystem and subassembly components of which the equipment is made up. The AbstractEquipmentSubsystem may or may not be capable of holding material and may or may not have recipe execution capability.

10.4.2 Figure 10 shows the relationships of the EquipmentSubsystem.

10.4.3 *Material Object*

10.4.3.1 An AbstractEquipmentSubsystem may or may not be able to hold material. Those that are able to hold material owns any material that it holds at any given point in time. They are expected to know the type of material that they hold, whether material is present, and the identifier of the material. In some cases, the AbstractEquipmentSubsystem may be able to hold multiple units of material or more than one type of material.

10.4.3.2 There are three major subtypes of material objects: consumables, durables, and substrates, where the major subtypes of durables are carriers and process durables.

10.4.3.3 Consumable materials, such as chemicals, are used up (as opposed to worn out) during the manufacturing process.

10.4.3.4 Durable material has a significant lifetime and is not part of the equipment (is removable). It includes material that some equipment may require to operate. The subtype of durable of common interest to almost all equipment is carrier. A carrier is a durable used to hold substrates. Process durables are of special interest to a limited subset of equipment. A process durable is an identifiable durable that is specified for a certain process step and recipe. For example, reticles are identified by product id and the mask level where used, and they are also identified by serial number. The factory is very interested in the management and tracking of process durables.

10.4.3.5 Product is typically a Substrate. In the semiconductor industry, Substrates include Wafer, Die, and Leadframe. Reticles are considered a Substrate by some equipment. In the semiconductor industry, reticles are considered as a process durable rather than a product. Specific reticles are specified in recipes for expose tools and modules. In other industries, substrates include flat panel (for flat panel displays) and discs (for hard disk drives).

10.4.3.6 Material objects are not formally defined in OBEM. Carrier objects are defined in SEMI E87 (CMS) and substrate objects are defined in SEMI E90 (STS).

10.4.4 *Material Locations*

10.4.4.1 A material location is a place that is capable of holding material.

10.4.4.2 A MaterialLocation is an abstraction used to facilitate the tracking of material without regard to the physical component associated with the tracking location. For example, a substrate chuck is a subsystem and can hold a single substrate on its surface. The entire subsystem consists of a stage that may or may not be able to move. It may use vacuum to hold the substrate in place, and it may have additional mechanisms and sensors. The MaterialLocation only represents the surface of the chuck. For this reason, the EquipmentSubsystem is said to *provide* a MaterialLocation rather than to be made up of MaterialLocations through aggregation.

10.4.4.3 This relationship is illustrated in Figure 11.

10.4.4.4 The MaterialLocation object is an abstract type. Figure 11 shows the subtypes of MaterialLocations of interest to OBEM.

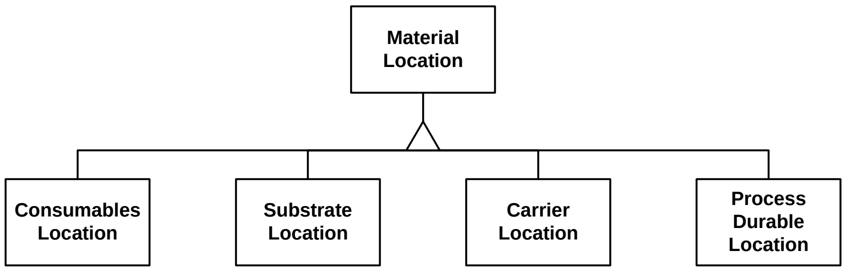


Figure 11
MaterialLocation Subtypes

10.4.4.5 *MaterialLocation Object* — Figure 12 shows the attributes of the MaterialLocation that are defined in Table 6. Every subtype of MaterialLocation shall have, at a minimum, attributes showing its current state and the ID of any material present.

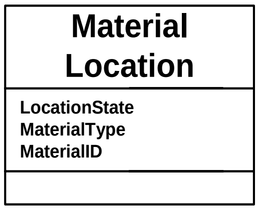


Figure 12
MaterialLocation Object Type

Table 6 MaterialLocation Attribute Definitions

Attribute Name	*Definition*	*Access*	*Reqd*	*Form*
ObjType	Object type	RO	Y	Text="MatlLoc"
ObjID	Object identifier	RO	Y	Text
LocationState	Current state.	RO	Y	Enumerated: Occupied Unoccupied
MaterialType	Type of material held.	RO	Y	Enumerated: One of, or a subtype of, the following Consumable Carrier Substrate ProcessDurable
MaterialID	Identifier of any material that occupies the location.	RO	Y	Text. "Unknown" if identifier is not known. Null (empty) string if location is unoccupied.

10.4.4.6 *CarrierLocation Object* — The CarrierLocation attribute LocationState may allow an additional enumeration Not Aligned, as shown in Table 7.

Table 7 CarrierLocation Attribute Definitions

Attribute Name	*Definition*	*Access*	*Reqd*	*Form*
ObjType	Object type	RO	Y	Text="CarrierLoc"
ObjID	Object identifier	RO	Y	Text
LocationState	Current state.	RO	Y	Enumerated: Occupied Unoccupied Not aligned

10.4.4.7 *SubstrateLocation Object* — The SubstrateLocation object is defined in SEMI E90 (STS). STS defines SubstID as an attribute of SubstrateLocation. Note this is identical to MaterialID in the general case for MaterialLocation.

10.4.5 *AbstractEquipmentSubsystem Requirements*

10.4.5.1 AbstractEquipmentSubsystems may or may not be able to hold material.

10.4.5.2 The AbstractEquipmentSubsystem owns all of the MaterialLocations that it provides or that are provided by one of its components. Similarly, it owns all of the Material at those MaterialLocations.

10.4.6 *AbstractEquipmentSubsystem Type* — Figure 13 shows the diagram for the AbstractEquipmentSubsystem object.

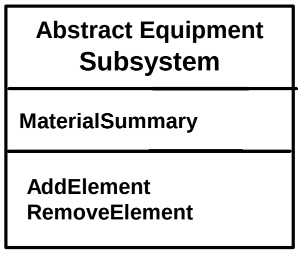


Figure 13
AbstractEquipmentSubsystem Object Type

10.4.7 *AbstractEquipmentSubsystem Attributes*

10.4.7.1 Table 8 defines the attributes of the AbstractEquipmentSubsystem.

Table 8 AbstractEquipmentSubsystem Attribute Definition

Attribute Name	*Definition*	*Access*	*Reqd*	*Form*
ObjType	Object type	RO	Y	Text="AbstractEqpSubsystem"
ObjID	Object identifier	RO	Y	Text
MaterialSummary	List of specifications of material capacity and current status for each type of material.	RO	N	List of structure composed of MaterialType, MaterialCapacity, and MaterialCount.

10.4.7.2 Table 9 defines individual elements of AbstractEquipmentSubsystem attributes.

Table 9 Definition of Elements of MaterialSummary Attribute

Element Name	*Definition*	*Access*	*Reqd*	*Form*
MaterialCapacity	Number of units of material that can be held at one time.	RO	Y	Unsigned integer.
MaterialCount	Current number of units of material present.	RO	Y	Unsigned integer.
MaterialType	Specifies the type of material held.	RO	Y	Text. Conforms to SEMI E5.

10.4.8 *AbstractEquipmentSubsystem Services*

10.4.8.1 *Add Element*

10.4.8.1.1 The user may be able to instruct the equipment to add a specific type of AbstractEquipmentElement. The supplier may restrict this service to specific AbstractEquipmentElement subtypes, both in destination (the AbstractEquipmentElement accepting the addition) and in the added element. This is an optional service.

10.4.8.1.2 A typical kind of AbstractEquipmentElement that the user may want to add is an "add-on sensor", a SensorActuatorDevice on a Sensor/Actuator Network. If the Equipment knows the properties, it is possible for it to read an added I/O device and to reference it within recipes.

10.4.8.2 *Remove Element* — The user may instruct the equipment to remove an AbstractEquipmentElement that was added earlier. This service is required wherever the Add Element service is supported.

10.4.9 *EquipmentSubsystem Object*

10.4.9.1 The EquipmentSubsystem is a concrete subtype of AbstractEquipmentSubsystem. It inherits all of the attributes, state models, and services of the AbstractEquipmentSubsystem and in addition defines the rules for aggregation. These rules are illustrated in Figure 14.

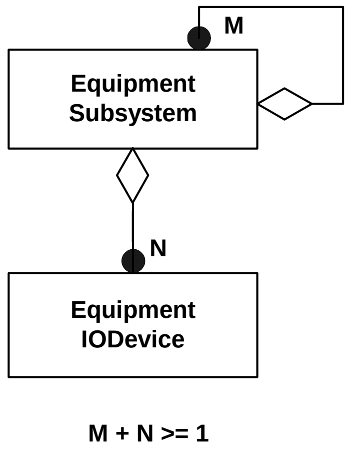


Figure 14
EquipmentSubsystem Aggregation

10.4.9.2 As shown in Figure 14, the EquipmentSubsystem may be made up of other, smaller EquipmentSubsystems and/or EquipmentIODevices. It is required to have at least one of these two components.

Table 10 EquipmentSubsystem Attribute Definition

Attribute Name	*Definition*	*Access*	*Reqd*	*Form*
ObjType	Object type	RO	Y	Text = "EqpSubsystem"
ObjID	Object identifier	RO	Y	Text.

10.5 *AbstractEquipmentModule Object*

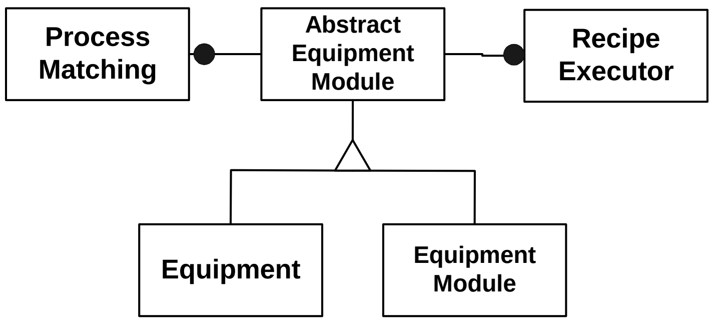


Figure 15
AbstractEquipmentModule Types

10.5.1 As shown in Figure 15, an *AbstractEquipment-Module* is a type of AbstractEquipmentSubsystem that represents a higher level of complexity and is of greater importance to the factory. It is mainly intended to represent process modules but may be used for other major intelligent subsystems capable of supporting the requirements for the AbstractEquipmentModule. It may be possible in some cases for the physical module to operate independently from the equipment.

10.5.2 The AbstractEquipmentModule has two subtypes, Equipment and EquipmentModule.

10.5.3 *ProcessMatching*

10.5.3.1 ProcessMatching provides one or more mechanisms for managing process differences between two or more identically configured subsystems of the same type to ensure that a generic recipe run on both subsystems will achieve the same process result. ProcessMatching may be either internal or external or both.

10.5.3.2 An example of an external method would be through provision of a ProcessMatching object that allows a user to manipulate offsets to process parameters. This can be done through setting parameter offsets for each subsystem so that, within specified constraints, all subsystems of the same type give the same process results. More sophisticated systems may use algorithms to determine process offsets based on mathematical models and module history.

10.5.3.3 As a simple example of process matching, three different individual hotplates may be matched for the temperature range 225–275° C by modifying their temperature offset by 1° C, 1.75° C, and –2.1° C respectively. This approach is illustrated in Figure 16.

10.5.3.4 The equipment supplier is responsible for determining the set of parameters to which offsets may be applied as well as the range of values that are valid.

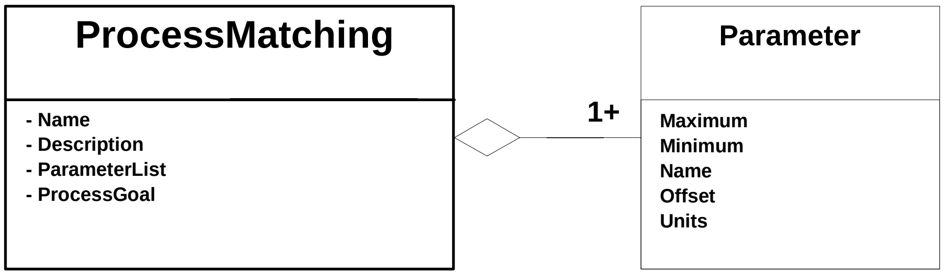


Figure 16
ProcessMatching and Parameter Objects

10.5.3.5 These parameters may be the same as those referenced within recipes for Recipe Variable Parameters.[6] In this case, Recipe Variable Parameters are given as setpoints that can be based on characteristics of the incoming substrate and are hardware independent, while Parameters used in Process-Matching are relative (applied to an existing setpoint) and are used to compensate for hardware differences. Support of both Recipe Variable Parameters and ProcessMatching parameters allows the user to compensate for hardware-specific differences separately from product-specific differences. The number of cycles since the last preventive maintenance performed on a process module affects its performance in a certain way, while the effect of too thin a film on the wafer will affect requirements in a different way.

10.5.3.6 The ProcessMatching object here is shown as an example. This is not a standardized object.

10.5.3.7 Internal methods of process matching may use other techniques, such as special types of recipes.

10.5.4 *AbstractEquipmentModule Requirements*

10.5.4.1 The AbstractEquipmentModule inherits the attributes, state models, and services of both the AbstractEquipmentElement and the AbstractEquipmentSubsystem. As a type of AbstractEquipmentSubsystem, the AbstractEquipmentModule is also able to hold one or more units of material.

10.5.4.2 The AbstractEquipmentModule represents major subsystems, such as process chambers. It supports basic operational commands: start, stop, pause, resume, and abort.

10.5.4.3 *ARAMS*

10.5.4.3.1 When implementing SEMI E58, the AbstractEquipmentModule shall provide compliance except as qualified in this section. The Abstract-EquipmentModule is the smallest component of equipment for which SEMI E58 (ARAMS) states should be maintained.

10.5.4.3.2 An AbstractEquipmentModule that can not be powered off separately from Equipment is not required to provide its own powerdown estimate or time of last powerdown.

10.5.4.3.3 Otherwise, an AbstractEquipmentModule implementing SEMI E58 shall comply with the requirements of Section 13, Object Services Compliance, in SEMI E58, including all of the attributes defined in Table 6, ARAMS Object Attribute Definitions in that document. These attributes are not repeated in OBEM. The user may change the ARAMS state. In some cases, a change in the ARAMS state of a module may cause an ARAMS state change for the equipment. The supplier shall document any relationships between the ARAMS state of the Equipment and the ARAMS states of its modules.

10.5.4.4 *AbstractEquipmentModule Service States* — For implementations of ARAMS, to be consistent with SEMI E10, the AbstractEquip-mentModule is IN SERVICE whenever it is in an uptime state. Otherwise, it is OUT OF SERVICE, as it can not be scheduled for manufacturing. To change the service state of an AbstractEquipmentModule, the user must change its SEMI E10 (RAM) state. Note that the service state of some modules will affect the service state of Equipment as well.

10.5.4.5 *Clock* — For modules able to operate independently of the equipment (e.g., modules with a dedicated CPU), AbstractEquipmentModules are required to have individual clocks. It is the responsibility of the Equipment to synchronize multiple internal clocks.

10.5.4.6 *Process Type*

10.5.4.6.1 An AbstractEquipmentModule has a Process Type that indicates its primary functionality as one of the following: Process, Measurement, Transport, or Storage. This is represented as a text string that may specialized further as needed.

10.5.4.6.2 The AbstractEquipmentModule has one or more ProcessCapabilities. ProcessType and Process-Capability are used for high-level and detailed process characterization. The user may add and remove ProcessCapability descriptions through operations Add Process Capability and Remove Process Capability.

10.5.4.7 *Process Matching*

10.5.4.7.1 All AbstractEquipmentModules supporting Recipe Execution shall provide one or more methods for process matching.

10.5.4.7.2 ProcessMatching allows the AbstractEquipmentModule to be tuned for a specific set of conditions in order to achieve the same results as other Abstract-EquipmentModules with the same process capability.

10.5.4.8 *Process Setup*

10.5.4.8.1 AbstractEquipmentModules capable of performing different processes may need to be reconfigured following one process before a process of a different type can be executed. Examples of setup requirements include: a new reticle for a stepper, a source change for an ion implanter, or a significant change in temperature for a furnace.

6 Variable Parameters are defined in SEMI E42.

10.5.4.8.2 Both process recipes and service recipes may be associated with specific setups.

10.5.4.8.3 Recipes of the class "/SETUP/" may be used to put the AbstractEquipmentModule into a specific state. The user uses the name of the setup for scheduling work. This is captured in the attribute ProcessSetup.

10.5.4.9 *Recipes*

10.5.4.9.1 An AbstractEquipmentModule may provide recipe execution services and the ability to accept, store, verify, select, and run Execution Recipes (SEMI E42). This ability is required for AbstractEquipment-Modules with a ProcessType of "Process" or "Measure-ment". Where provided, recipes and recipe execution shall comply with the fundamental requirements for the Execution Recipe and Recipe Executor as specified in SEMI E42 (RMS).

10.5.4.9.2 Some equipment performs its process in three stages, which can be characterized as preamble, main process, and postamble, where the preamble is preparation for the stable part of the process, and the postamble takes care of the transition from the stable part of the process until it is ready to unload the material. In a furnace, for example, the preamble could include both the period where it is ramping to attain the setpoint temperature and the gas flow setpoints. A fourth stage is sometimes defined to handle abnormal terminations.

10.5.4.9.3 From a recipe standpoint, these different stages can be represented as different sections of a single recipe or as separate recipes linked to a main recipe.

10.5.4.10 *Mechanical Dry Run*

10.5.4.10.1 The AbstractEquipmentModule shall support the capability to do a mechanical dry run. This allows the material handling subsystems and software functions to be exercised and tested without requiring full process hookups and without using process consumables. Typically this is done through a specially designated recipe that allows time settings and does not use settings for temperature, gases, plasmas, water, etc. In some cases, it may require a recipe of a special class, such as "/DRYRUN/". Environmental subsystems such as vacuum, nitrogen purge, particle detection subsystems, etc. must be allowed to function normally during a mechanical dry run.

10.5.4.10.2 Dry runs shall be prohibited during the ARAMS Manufacturing state.

10.5.4.10.3 Certain types of tools may need to modify the definition of the mechanical dry run to address special issues in a way that still satisfies the objectives. Equipment documentation shall specify the method used to satisfy this requirement.

10.5.5 *AbstractEquipmentModule State Model*

10.5.5.1 The AbstractEquipmentModule inherits the Operational State Model of the AbstractEquipment-Subsystem. In addition to the concurrent substates of SERVICE, the AbstractEquipmentModule adds the BEHAVIOR state. Figure 17 shows the Operational State Model with its two concurrent substates.

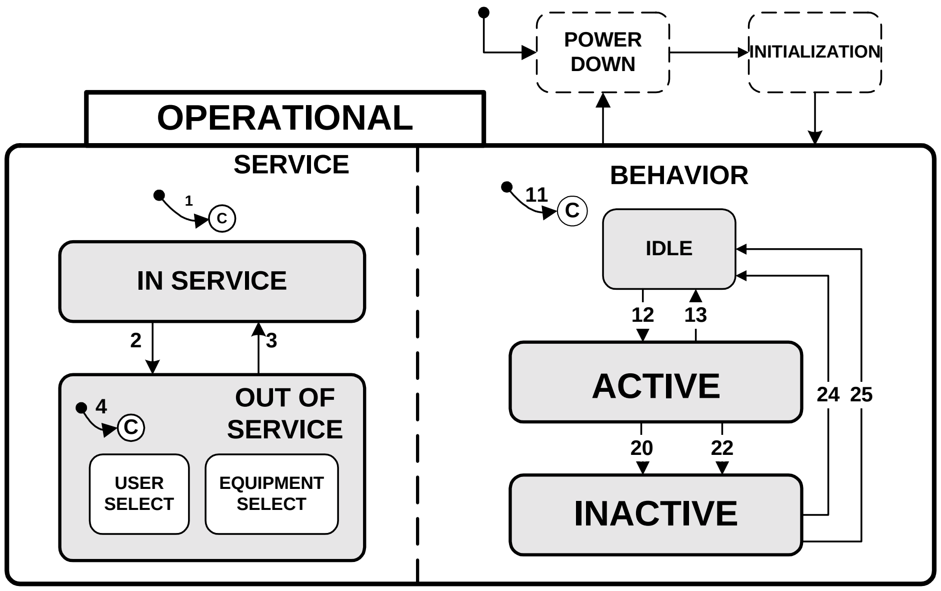


Figure 17
AbstractEquipmentModule Operational State Model

10.5.5.2 Additional details of the BEHAVIOR state are shown in Figure 18.

10.5.5.3 In addition to the states shown above, the AbstractEquipmentModule has an ARAMS state model, defined in SEMI E58. The diagram for the ARAMS state model for modules is shown in Figure 6 in that document.

10.5.5.4 *AbstractEquipmentModule Behavior State*

10.5.5.4.1 The details of the BEHAVIOR state are shown in Table 11. POWERDOWN and INITIALIZING states are included in the table to simplify comparisons with the ARAMS State Model and the Operational State Model defined in SEMI E58. State models are initialized during the INITIALIZATION state. Communications shall be initialized before the ARAMS state model is initialized in order to allow reporting of power down events.

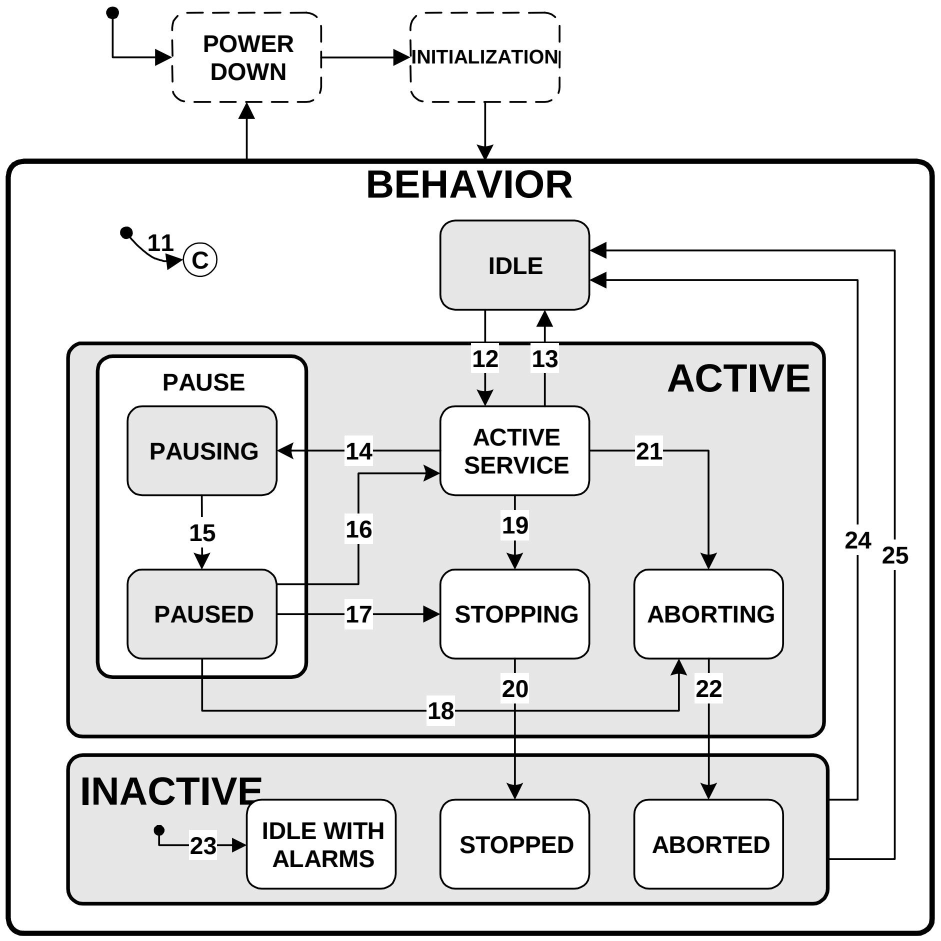


Figure 18
Behavior State

10.5.5.4.2 Table 11 defines the substates of the BEHAVIOR state.

Table 11 AbstractEquipmentModule Behavior State Definitions

State Name	*Superstate*	*Definition*
POWERDOWN	None	The AbstractEquipmentModule is powered off.
INITIALIZING	None	The AbstractEquipmentModule system, including hardware and software, is being initialized. It is not ready to accept instructions.
IDLE	IN SERVICE	The AbstractEquipmentModule is fault free and able to accept instructions to perform a manual or automated procedure that may change its physical state.

State Name	*Superstate*	*Definition*
ACTIVE	IN SERVICE	The AbstractEquipmentModule is performing activities that change its physical state.
ACTIVE SERVICE	ACTIVE	While assigned to manufacturing, the AbstractEquipmentModule may only perform its normal process.
PAUSE	ACTIVE	The AbstractEquipmentModule has received instructions to pause its activity.
PAUSING	PAUSE	The AbstractEquipmentModule is in the process of putting itself into a safe state.
PAUSED	PAUSE	The AbstractEquipmentModule is in a safe state.
STOPPING	ACTIVE	The AbstractEquipmentModule has begun the stop operation.
ABORTING	ACTIVE	The AbstractEquipmentModule has started the abort operation.
INACTIVE	IN SERVICE	The AbstractEquipmentModule has completed its stop or abort operation.
IDLE WITH ALARMS	INACTIVE	The AbstractEquipmentModule is idle but has a fault condition.
STOPPED	INACTIVE	The AbstractEquipmentModule has completed the stop operation.
ABORTED	INACTIVE	The AbstractEquipmentModule has completed the abort operation.

10.5.5.5 *AbstractEquipment Module State Transition Table* — Table 12 defines the state transitions for the AbstractEquipmentModule Operations state model.

Table 12 AbstractEquipmentModule Behavior State Transitions

#	*Previous State*	*Trigger*	*New State*	*Action(s)*	*Comment*
11	Any	State model is initialized.	depends on previous state and current condition	None	See Section 9.5.2.2.
12	IDLE	An activity changing the physical state of the module is started.	ACTIVE SERVICE	None	Substates of ACTIVE SERVICE are module dependent.
13	ACTIVE SERVICE	All activities changing the physical state are completed without fault.	IDLE	None	IDLE is fault-free.
14	ACTIVE SERVICE	The module received instructions to pause its activity.	PAUSING	None	The module is responsible for reaching a safe state.
15	PAUSING	All activity is paused.	PAUSED	None	
16	PAUSED	The module has received instructions to resume its activity.	ACTIVE SERVICE	None	
17	PAUSED	The module has received instructions to stop its activity.	STOPPING	None	The module is responsible for reaching a safe state.
18	PAUSED	The module has received instructions to abort its activity.	ABORTING	None	
19	ACTIVE SERVICE	The module has received instructions to stop its activity.	STOPPING	None	
20	STOPPING	The Stop operation has completed and all material started has been returned to its proper destination.	STOPPED	None	
21	ACTIVE SERVICE	The module has received instructions to abort its activity.	ABORTING	None	The module is responsible for reaching a safe state.
22	ABORTING	Completion of aborting activity.	ABORTED	None	
23	Any	Default entry into INACTIVE	IDLE WITH ALARMS	None	One or more exceptions detected during initialization.
24	INACTIVE	The module received instructions to return to IDLE.	IDLE	None	IDLE is fault-free.
25	INACTIVE	The module returns to IDLE automatically.	IDLE	None	IDLE is fault-free.

10.5.6 *AbstractEquipmentModule Attributes*

10.5.6.1 The AbstractEquipmentModule Object is shown in Figure 19.

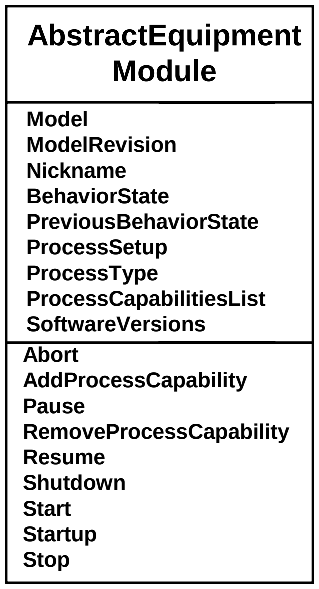


Figure 19
AbstractEquipmentModule Object Type

10.5.6.2 Attributes of the AbstractEquipmentModule are defined in Table 13.

Table 13 AbstractEquipmentModule Attribute Definition

Attribute Name	*Definition*	*Access*	*Reqd*	*Form*
ObjType	Object type	RO	Y	Text="AbstractEquipmentModule"
ObjID	Object identifier	RO	Y	Text
Nickname	User-assigned name	RW	Y	Text
Model	Model designator	RO	Y	Text
ModelRevision	Model Revision	RO	Y	Text
BehaviorState	Current substate of BEHAVIOR	RO	Y	Enumerated
ProcessType	Characterization of processes for user	RW	Y	Text
ProcessCapabilityList	One or more specific process capabilities	RO	Y	List of text
PreviousBehaviorState	The previous operations state	RO	Y	Enumerated
ProcessSetup	Name of current setup.	RW	Y	Text
SoftwareVersions	The set of software versions installed	RO	Y	Ordered list of text
Units	Type of material held	RO	Y	Text

NOTE 1: Required if the AbstractEquipmentModule provides recipe execution services.

10.5.7 *AbstractEquipmentModule Services* — The AbstractEquipmentModule has one or more automated activities, including its normal process (its intended function). The user shall be able to start, stop, abort, and resume the activities of the module.

NOTE 4: Changing the operational state of an individual AbstractEquipmentModule within a multi-process Equipment during automatic processing will affect both upstream and downstream operations. The Equipment is responsible for determining the impact on other AbstractEquipmentModules.

NOTE 5: Directly changing the operational state of the AbstractEquipmentModule is not the same as changing the state of a job or as changing the operational state of the Equipment. The equipment supplier shall document the relationships between the operational state of the Equipment, the AbstractEquipmentModule, and the different job types that it supports.

10.5.7.1 *Abort*

10.5.7.1.1 On receipt of an Abort request from the user, the module shall stop all of its current activity as soon as it is safe to do so. It should remove any material in process within the module whenever possible to do so safely.

10.5.7.1.2 A module may also self-abort when it detects a dangerous condition.

10.5.7.1.3 In the Behavior State Model, the module shall transition from ACTIVE SERVICE to ABORTING when an abort occurs. When all activity has ceased other than normal background activities for environmental control, the module transitions to ABORTED. The module shall not start a new activity until it is instructed to do so by the user or the user instructs it to return to IDLE.

10.5.7.2 *Add Process Capability* — ProcessCapabilityList is an attribute of the equipment module and consists of a list of text strings representing process capabilities that are meaningful to the user. Individual capabilities are added by the factory to recognize the type of processes the equipment module may be able to run.

10.5.7.3 *Pause*

10.5.7.3.1 On receipt of a Pause request from the user, the module shall suspend its activity at the next logical point in the process from which it will be able to resume activities at a later time and which will prevent unintended change to the product. It shall not accept new work while it is in this state.

10.5.7.3.2 In many cases, to avoid unintended change to the product, the AbstractEquipmentModule should wait until the current product has been removed and then pause, preventing new product from entering.

10.5.7.4 *Remove Process Capability* — This allows the factory to remove from the ProcessCapabilityList a capability that is no longer required or one that can not be performed by the equipment due to environment or actual physical changes in the equipment.

10.5.7.5 *Resume* — When a Resume request is received during a PAUSE state, the module shall resume its activity from the point where it was paused.

10.5.7.6 *Shutdown* — The Shutdown request is used to put an AbstractEquipmentModule in a state where it is safe to turn off its power and remove it from the equipment. The specific operations shall be documented by the supplier.

10.5.7.7 *Start*

10.5.7.7.1 Some activities may require specific Start instructions.

10.5.7.7.2 The user may issue a Start Activity request to a specific AbstractEquipmentModule.

10.5.7.8 *Startup* — The Startup request is used when an AbstractEquipmentModule has been shutdown. The specific activities that result shall be documented by the supplier.

10.5.7.9 *Stop*

10.5.7.9.1 On receipt of a Stop request from the user, the module shall stop its current activity in an orderly fashion as soon as it is safe to do so, with all product material returned as would be expected had the activity completed normally.

10.5.7.9.2 In the Behavior State Model, the module shall transition from ACTIVE SERVICE to STOPPING when a stop operation begins. When all activity has ceased other than normal background activities for environmental control, the module transitions to STOPPED. The module shall not start a new activity until it is instructed to do so by the user.

10.5.8 *EquipmentModule Object* — The EquipmentModule is a type of AbstractEquipmentModule and inherits all of its attributes, state models, relations, and services. As shown in Figure 20, the EquipmentModule may be an aggregate of EquipmentIODevice, EquipmentSubsystem, and other EquipmentModule objects.

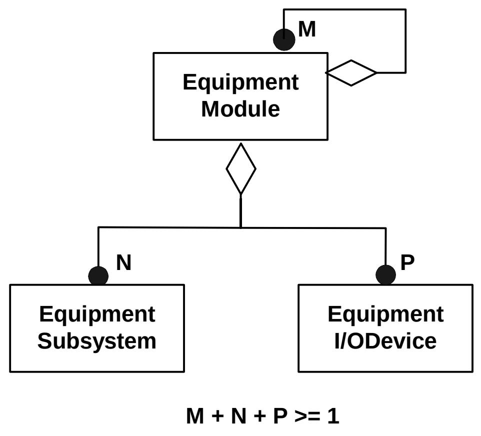


Figure 20
EquipmentModule Aggregation

10.6 *Equipment Object*

10.6.1 The Equipment is a type of AbstractEquipmentModule and supports all of the attributes, state models, and services defined for the AbstractEquipmentModule.

10.6.2 Equipment is an aggregation of EquipmentModules, EquipmentSubsystems, and EquipmentIODevices objects and owns all the objects of which it is made. Figure 21 shows the relationships of the Equipment.

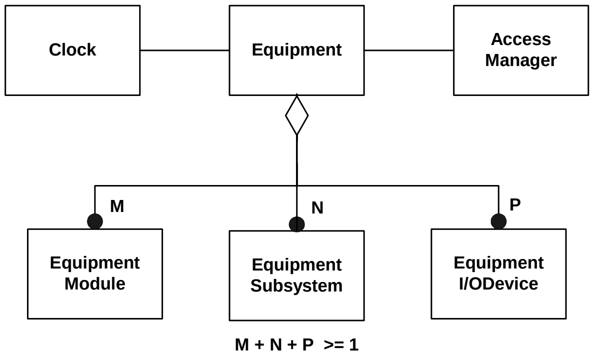


Figure 21
Equipment Object Relationships

10.6.3 *Equipment Requirements* — All equipment shall satisfy the following requirements:

- The Equipment provides the interface to the factory and shall not be made up of other Equipment.
- Equipment is the root object within the equipment hierarchy.
- Equipment may not be a component of any other AbstractEquipmentElement or any of its subtypes.
- Equipment may be made up of any standardized objects except for other Equipment objects.
- Equipment is required to have a Clock.
- Equipment is required to have at least one CarrierPort. Exceptions to this requirement may be made for testers and for exposure equipment in a linked litho cell that have separate communications with the factory systems.
- Equipment is responsible for (owns) all of its components

10.6.4 *Clock Object*

10.6.4.1 Clock represents current real-time date and time of day information. This information may be provided by a physical real-time clock or by other methods of calculating time. The Clock is used to timestamp data and events.

10.6.4.2 Requirements include:

- The user shall be able to read the current date and time at any time.
- Equipment shall be capable of using both factory network time services and user interface services to set current date and time.
- Clock resolution shall be 0.01 seconds or smaller.
- Date shall maintain a four-digit year.

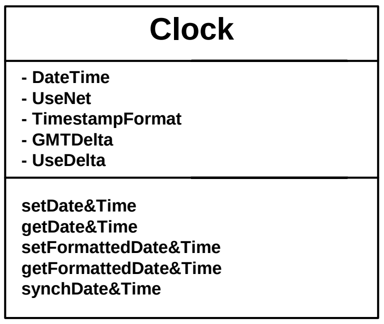


Figure 22
Clock Object

10.6.4.3 All calculations based on, or related to, date and time shall be fault-free in both the twentieth and twenty-first centuries.

10.6.4.4 Table 14 defines the attributes of Clock.

Table 14 Clock Attribute Definitions

Attribute Name	*Definition*	*Access*	*Reqd*	*Form*
ObjType	Object type	RO	Y	Text = "Clock"
ObjID	Object identifier	RO	Y	Text
DateTime	Current date and time	RO	Y	Protocol specific
UseNet	Enable/disable network time synchronization	RW	Y	Boolean
TimestampFormat	Date and time format used to timestamp events	RW	Y	Enumerated
GMTDelta	Difference, in minutes, between the current time and Greenwich Mean Time (GMT)	RO	Y	Integer
UseDelta	Enable/disable showing current GMTDelta on timestamps	RW	Y	Boolean

10.6.4.5 *Network Time*

10.6.4.5.1 It shall be possible to set current date and time by using time services provided by the network in the factory. Use of network time allows the different systems within the factory to remain closely synchronized with a high degree of accuracy, and it relieves the factory host from explicitly setting the equipment's clock. The attribute UseNet allows the user to enable and disable use of network time services.

10.6.4.5.2 When enabled, the equipment shall be responsible for ensuring that any change in time does not result in discontinuities in date/timestamp reporting. For example, if the time changes in the middle of a process, applying the time change to current process events can cause later events to appear earlier than previous events, or they can give the appearance of large pause in activities during processing. Both of these cases can cause time-based analysis of data to give wrong results and must be prevented.

10.6.4.6 *Timestamp Data*

10.6.4.6.1 Date/time information is used by the factory for a variety of purposes, including analysis of historical data. Typically a factory keeps local time. GMTDelta, the offset between local time and Greenwich Mean Time, is provided to allow time values to be compared outside of the local time zone. GMTDelta may also be added to the timestamp to clarify time data when local time changes within a process.

10.6.4.6.2 The attribute TimestampFormat allows the user to select the format used for timestamping events. Possible formats include: formatted time, a text string of the form YYYYMMDDhhmmsscc, long signed integer, and floating point.

10.6.5 *Equipment Attributes*

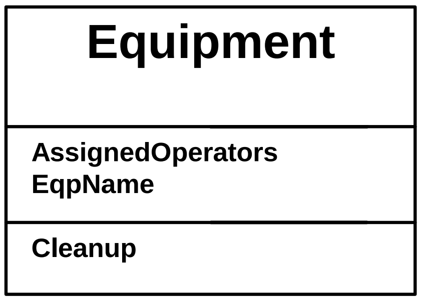


Figure 23
Equipment Object

10.6.5.1 Table 15 defines the attributes of the Equipment object, as shown in Figure 23.

Table 15 Equipment Attribute Definitions

Attribute Name	*Definition*	*Access*	*Reqd*	*Form*
ObjType	Object type	RO	Y	Text="Equipment"
ObjID	Object identifier	RO	Y	Text
AssignedOperators	Currently assigned operators.	RO	Y	List of text
Nickname	User-assigned name.	RW	Y	Text

10.6.6 *Equipment Services*

10.6.6.1 Basic operational commands may be directed to an EquipmentModule, the Equipment as a whole, or to an individual Process Job.

10.6.6.2 For multi-module or multi-process Equipment, the effect of an operational command varies depending upon the level of the target object and the configuration of the Equipment. Given the number of Process Jobs that are defined during the course of a work shift and their temporary nature; it is unlikely that an operator will differentiate easily between one Process Job and another. Whereas they will be able to more quickly identify an individual module or the Equipment when time is a critical factor.

10.6.6.3 *Operational Services*

10.6.6.3.1 *Abort* — The user may direct the Equipment to abort its operations at any time. The Equipment shall stop all activity as soon as possible while maintaining the safety of the product, the Equipment itself, and people. Product is not returned to the carriers.

10.6.6.3.2 *Cleanup* — When product is left in the Equipment following an abort operation, it may be necessary to request the Equipment to return all substrates to their carriers.

10.6.6.3.3 *Pause* — The user may request the Equipment to pause its operations. The Equipment is responsible for determining the appropriate points within its process where it is safe to temporarily stop. As with the Stop command above, the multi-module Equipment may direct individual EquipmentModules to Pause one at a time.

10.6.6.3.4 *Resume* — This command directs the Equipment to resume normal operations from the PAUSED state.

10.6.6.3.5 *Start* — The Start command directs the Equipment to begin an activity that is already setup and ready. Its use is not intended for automatic processing by Equipment providing Process Job capabilities.

10.6.6.3.6 *Stop* — The user may direct the Equipment to stop activity at any time. The Equipment shall be responsible for ensuring that all of its components stop their activities in an orderly fashion at the first opportunity. Processing shall be completed for all work in process (e.g., substrates that have already been removed from their input carrier), but no processing shall be started for remaining work.

For Equipment with multiple EquipmentModules performing individual process steps in a sequence of steps, a Stop command to the Equipment is equivalent to issuing a Stop command to the first EquipmentModule in the sequence, followed by subsequent Stop commands to each EquipmentModule once it has accepted the substrate last started.

10.6.6.4 *Time Services* — This section describes the services provided by the clock.

10.6.6.4.1 *Request Date and Time* — The user shall be able to read the equipment's current date and time as a numeric value.

10.6.6.4.2 *Set Date and Time*

10.6.6.4.2.1 If UseNet is disabled, the user shall be able to set the current date and time as a numeric value. Note that this method does not generally give good synchronization results.

10.6.6.4.2.2 Set Date and Time shall not cause a discontinuity in timestamps for data associated with an individual substrate in process.

10.6.6.4.3 *Request Formatted Date and Time* — The user shall be able to read the equipment's current date and time as a formatted text string.

10.6.6.4.4 *Set Formatted Date and Time* — If UseNet is disabled, the user shall be able to set the current date and time as a formatted text string. Note that this method does not generally give good synchronization results.

10.6.6.4.5 *Synchronize Date and Time* — The user may request the equipment to synchronize date and time from the network as soon as it can do so without disrupting the timestamp series for any work in process.

11 OBEM Equipment Functional Requirements

11.1 The functional areas of the equipment are described in the Functional View in Section 8.3. Section 11 defines the requirements for each of these functional areas. These are requirements at the level of the Equipment object, in addition to those specified in Section 10.

11.2 *Access Management*

11.2.1 Access Management is responsible for all communications with the user, including local operators, factory systems, and remote users. As illustrated in Figure 24, the equipment may be required to support multiple users, with at least one connection to the factory and at least one interface to the local operator. Where automation is used for material handling, more than one human interface will be required.

11.2.2 Objects shown in Figure 24 are not standardized objects. Further work is required to determine requirements for standardized objects to support access management.

11.2.3 *Arbitration* — This section is reserved.

11.2.4 *Authorization* — This section is reserved.

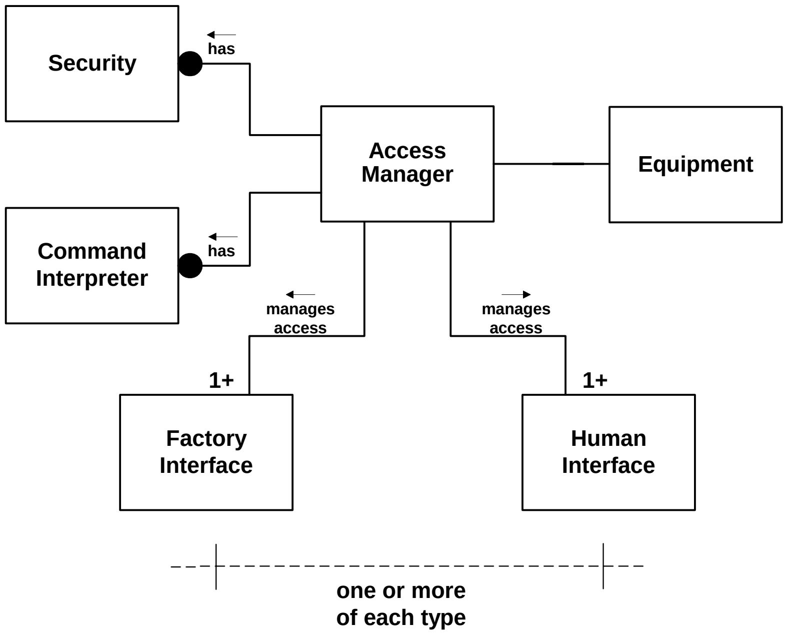


Figure 24
Access Management Model

11.2.5 *Multiple Operator Interfaces*

11.2.5.1 The equipment shall support a minimum of two user interfaces, one that is dedicated to factory system communications and one that is dedicated for local operations.

11.2.5.2 Equipment may support additional interfaces as well. Equipment handling large wafers (300 mm diameter or more) may be required to provide a second operator interface away from the traditional interface at the front. The factory may require separate connections for direct communications with AMHS and/or advanced process control. The equipment supplier’s customer service may need remote access for diagnostics.

11.2.6 *Remote Dial-in* — This section is reserved for requirements for remote dial-in using a modem.

11.2.7 *Access Attributes* — Table 16 defines the attributes of the Access Object.

Table 16 Access Session Configuration Data

Attribute Name	*Definition*	*Access*	*Reqd*	*Form*
Authorization	Level of authorization specifies user access.	RO	Y	Equipment-specific
FDControl	User may detect a probable fault and instruct that no new work be started.	RO	Y	Boolean
FormatCode	Indicates the code used for interpretation of text.	RW	Y	Enumerated
Language	Indicates the preferred language for text.	RW	Y	Enumerated
Recipe Access	The user may request recipe upload, download, and other recipe-access requests.	RW	Y	Enumerated
R2Rcontrol	User may issue instructions for run-to-run control.	RO	Y	Boolean
MaterialControl	User may issue instructions related to loading and unloading carriers.	RO	Y	Boolean
OperationsControl	User may issue operational instructions.	RO	Y	Boolean
UserID	Identification of the user	RO	Y	Text

11.3 *Date/Time Management* — Equipment shall have an OBEM-compliant Clock with sufficient resolution to differentiate the order in which events occur. (See Section 12.)

11.3.1 *Timestamp Data* — Date/time information is used by the factory for a variety of purposes, including analysis of historical data. Should time change significantly during processing, the assumed ordering of events can be affected so that later events appear earlier, large unexplained gaps can appear in a sequence of events, or data can seem to appear from the past. For these reasons, any change in current time shall not be applied to material in the middle of a process.

11.3.2 *Date/Time Values* — Date and Time information shall be expressed as a numeric value.

11.3.3 *Local Time Change* — Where local time changes automatically, the equipment is responsible for maintaining the correct difference between local time and Greenwich Mean Time (GMT) in the Clock attribute GMTDelta.

11.3.4 *Optional Timestamp Format* — The equipment shall provide an option for the user to include the value of GMTDelta as an additional field in timestamps. The user may enable and disable use of GMTDelta by setting and clearing the Boolean Clock attribute UseDelta.

11.4 *Equipment Control* — Control is divided into several areas: Fault Detection Control, Run to Run Control, Material-Control, and Operational Control. Control of each of these four areas shall be assigned to exactly one user at any point in time. However, that user may be the same for one or more of these areas.

11.4.1 *Advanced Process Control (APC)* — Two categories of APC are described herein: run-to-run control (between lots, carriers, or substrates), and fault-detection control. Support for APC may be required at the EquipmentModule level.

11.4.2 *Run-to-Run Control (R2R)* — The factory may dedicate a separate session for Run-to-Run Control. Run-to-run control consists of adjusting Variable Parameter settings of Process Recipes within a process job.

11.4.3 *Fault Detection Control (FDC)*

11.4.3.1 Fault detection is able to detect a probable fault before it can be detected by SPC, equipment failure, or product failure. The user may provide fault detection through analysis of process data in conjunction with data from external sensors.

11.4.3.2 A separate dedicated session may be used for fault detection control because of its need for large amounts of data. FDC may instruct the Equipment to put itself into a downtime state as soon as it completes work currently in process.

11.5 *Event Management* — Equipment shall provide standard mechanisms for notifying the user of events of interest to the user, together with any data the user has associated with specific events.

11.6 *Exception Management* — The equipment shall comply with SEMI E41 (EMS).

11.7 *Job Management* — This section is reserved for requirements related to specifications for managing jobs. Jobs are defined by SEMI E94 Control Job Management and by SEMI E40 Process Management.

11.8 *Material I/O Management* — Material I/O Management allows the user to request services related to management of the carrier ports and internal buffers required for managing the movement of product through the factory (material logistics). Equipment shall comply with the fundamental requirements of SEMI E87 (CMS).

11.9 *Material Management*

11.9.1 Material Management is responsible for all material present at the equipment from the time it is loaded from the factory until it is unloaded. This includes tracking of all material that is identifiable.

11.9.2 The location of all material that is mobile, such as product, shall be known at all times. The factory may inquire about what material is currently present and where (at what MaterialLocation) that material is placed.

11.9.3 Consumables that are not mobile; such as gold wire on a wire-bonder, may be identified by lot number. The equipment may be required to store current lot numbers for non-mobile material when that information is provided by the factory.

11.9.4 Equipment shall comply with the fundamental requirements for SEMI E90 (STS).

11.10 *Operational Control* — The user with Operation Control may issue operational commands related to starting, stopping, process job management, and recipe access, including upload, download, and recipe selection.

11.11 *Object Management*

11.11.1 This section is reserved for requirements concerning data and information.

11.11.2 Object Management is responsible for providing all data and information required by the factory through Object Services.

11.12 *Performance Management* — As a type of AbstractEquipmentModule, the equipment shall conform to the Operational State Model defined in Section 10.5.5 and compliance to SEMI E58 (ARAMS).

11.13 *Operations Management* — This section specifies the requirements related to operations, including processing.

11.14 *Recipe Execution* — As a type of EquipmentModule, Equipment providing temporary recipe storage and recipe execution capabilities shall comply to the fundamental requirements for the Execution Recipe and Recipe Executor as specified in SEMI E42.

11.15 *Recipe Management* — Equipment providing long-term storage of recipes shall comply with the fundamental requirements for the Managed Recipe and either the Recipe Namespace with its Recipe Namespace Manager, or the Distributed Recipe Namespace Segment, for all recipes stored, as specified in SEMI E42.

12 OBEM Service Definitions

12.1 This section defines the content of service messages specified in this document and provided by OBEM objects. The services are described in previous sections. This section details the parameters for each message.

12.2 Table 17 lists the services defined in this section. Services of Type "R" are asynchronous requests that require a response. Services of Type N are notifications and do not require a response.

12.3 OBEM compliant equipment is required to understand all service requests. However, if the service is not marked as required, then support for the service itself (the operations) is optional for the equipment, and a response of "request denied" may be returned.

12.4 Services are listed in alphabetical order.

Table 17 List of Services

Operation	*Description*	*Type*	*Reqd*
Abort	Directs an object to immediately stop all activity as soon as it is safe to do so.	R	Y
AddElement	Add a type of AbstractEquipmentElement.	R	N
AddProcessCapability	Add a process capability.	R	Y
ChangeService	Set an AbstractEquipmentElement in service or out of service.	R	N
Cleanup	Directs Equipment to return all substrates to their carriers.	R	N
GetDateTime	Returns the current date and time.	R	Y
GetProgramDirectory	Returns the list of identifiers of Sensor Programs stored by the specified Sensor/ActuatorDevice.	R	N
Pause	Directs an object to temporarily stop its activity as soon as it is at a point where it is safe to the process to do so.	R	Y
RemoveProcessCapability	Remove a process capability from the list.	R	Y
RemoveElement	Remove an element added by the user.	R	Y
Resume	Directs a paused object to resume its activity.	R	Y
SelectProgram	Provides an EquipmentIODevice Program identifier to be selected and executed.	R	N
SetDateTime	Provides current date and time for setting the equipment Clock.	R	Y
Shutdown	Prepare for powerdown.	R	N
Start	Directs an object to begin an activity that has been set up.	R	N
Startup	Prepare for standby for processing.	R	N
Stop	Directs an object to stop an activity in progress by completing work already started and not beginning new work.	R	Y

12.5 *Service Parameter Definitions* — Table 18 defines the parameters for OBEM services.

Table 18 Service Parameter Definitions

Parameter	*Definition*	*Form*
DateTime	Date and time setting.	To be resolved.
DeviceProgramID	Identifier for EquipmentIODevice program.	Integer
ErrorCode	Contains the code for the specific error found.	Enumerated
ErrorText	Text in support of the error code.	Text

Parameter	*Definition*	*Form*
ObjSpec	Object specifier of target object. An empty string implies Equipment is the target.	Formatted text. See SEMI E39 for restrictions.
ProcessCapability	User-defined description	Text
Property	An attribute name and value.	Structure of attribute name, attribute value
RequestAcknowledge	Acknowledgement of request.	Enumerated: Request performed Cannot perform now Invalid request Already in requested state Will perform when able and report results when complete
RequestStatus	Result of request	Structure consisting of RequestAcknowledge and Status
Service	New substate of SERVICE	Enumerated: IN SERVICE, OUT OF SERVICE
Status	Reports any errors found.	List of ErrorCode, ErrorText pairs

12.6 *Abort* — Table 19 defines the Abort service. The Abort request may be directed to the Equipment or to an EquiipmentModule.

Table 19 Abort Service Definition

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
ObjSpec	M	-	Object specifier for EquipmentModule.
RequestStatus	-	M	Result status

12.7 *AddProcessCapability* — Table 20 defines the service to add a ProcessCapability. This request may be directed to the Equipment or to an EquipmentModule.

Table 20 AddProcessCapability Service Definition

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
ObjSpec	M		Object specifier for owner of capability
ProcessCapability	M	-	User description of a process objective.
RequestStatus	-	M	Result status

12.8 *AddElement* — Table 21 defines the service to add an AbstractEquipmentElement. This request may be directed to the Equipment or to an EquipmentModule. Properties needed for the AbstractEquipmentElement must be included in the list of properties.

Table 21 AddElement Service Definition

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
ObjSpec	M	-	Object specifier for new owner
(list of) Property	M	-	Properties of new element
RequestStatus	-	M	Result status

12.9 *Cleanup* — This service directs the equipment to return all substrates that have been removed from their carriers to their appropriate destinations. Table 22 defines the Cleanup service.

Table 22 Cleanup Service Definition

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
RequestStatus	-	M	Result status

12.10 *GetDateTime* — Table 23 defines the GetDateTime service.

Table 23 GetDateTime Service Definition

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
DateTime	-	M	Date and time.
RequestStatus	-	M	Result status

12.11 *ChangeService* — Table 24 defines the ChangeService service. This service requests that the target object be placed either in service or out of service. The user shall only be allowed to place an object back into service if it was originally put out of service by a user.

Table 24 ChangeService Service Definition

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
ObjSpec	M	-	Object specifier for target
Service	M	-	New substate for SERVICE.
RequestStatus	-	M	Result status

12.12 *Pause* — Table 25 defines the Pause service.

Table 25 Pause Service Definition

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
ObjSpec	M	-	Object specifier for target object.
RequestStatus	-	M	Result status

12.13 *RemoveProcessCapability* — Table 26 defines the service to remove a process capability from the ProcessCapabilityList.

Table 26 RemoveProcessCapability Service Definition

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
ObjSpec	M	-	Object specifier for target object.
ProcessCapability	M	-	Capability to be removed
RequestStatus	-	M	Result status

12.14 *RemoveElement* — Table 27 defines the service to remove an element previously added by a user.

Table 27 RemoveElement Service Definition

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
ObjSpec	M	-	Object specifier for element to be removed
RequestStatus	-	M	Result status

12.15 *Resume* — Table 28 defines the service for starting an activity at a designated EquipmentModule or at the Equipment.

Table 28 ResumeService Definition

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
ObjSpec	M	-	Object specifier for target object.
RequestStatus	-	M	Result status

12.16 *SelectProgram* — Table 29 defines the service to select a program for an EquipmentIODevice.

Table 29 SelectProgram Service Definition

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
ObjSpec	M	-	Object specifier for target object.
DeviceProgramID	M	-	Text
RequestStatus	-	M	Result status

12.17 *SetDateTime* — Table 30 defines the service for setting the equipment's date and time.

Table 30 SetDateTime Service Definition

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
DateTime	M	-	Date and time
RequestStatus	-	M	Result status

12.18 *Shutdown* — Table 31 defines the service to shut down the equipment or one of its modules.

Table 31 Shutdown Service Definition

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
ObjSpec	M	-	Object specifier for target
RequestStatus	-	M	Result status

12.19 *Start* — Table 32 defines the service for starting an activity at a designated EquipmentModule or at the Equipment.

Table 32 Start Service Definition

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
ObjSpec	M	-	Object specifier for EquipmentModule or Equipment.
RequestStatus	-	M	Result status

12.20 *Startup* — Table 33 defines the service to start up the equipment or one of its modules.

Table 33 Shutdown Service Definition

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
ObjSpec	M	-	Object specifier for target
RequestStatus	-	M	Result status

12.21 *Stop* — Table 34 defines the service for stopping an activity at the equipment or one of its modules.

Table 34 Stop Service Definition

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
ObjSpec	M	-	Object specifier for EquipmentModule or Equipment.
RequestStatus	-	M	Result status

12.22 *SynchDate&Time* — The SynchDate&Time service instructs the equipment to synchronize date and time with the network at its next opportunity to do so at a time where a timestamp series for work in process will not be compromised by a change in date or time. This service is defined in Table 35.

Table 35 SynchDate&Time Service

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
RequestStatus	-	M	Result status

13 Services Provided by Factory Systems

13.1 This section describes the services provided by the factory system. These are messages sent by the equipment to the factory system.

13.2 *Date and Time Request* — The equipment may request the current date and time from the factory system. This message service is identical to the GetDateTime service defined in Section 12.6 and Table 23.

13.3 *Event Notification* — The factory system will receive standard event notifications from the equipment. Event notifications used will depend upon the protocol used.

13.4 *Alarm and Error Notifications* — The factory system will receive notifications of errors and alarms. Errors may be recoverable. Equipment shall comply to SEMI E41.

14 OBEM Compliance

NOTE 6: Section must be completed in order for the provisional status of OBEM to be removed.

15 Scenarios

NOTE 7: Section must be completed in order for the provisional status of OBEM to be removed.

15.1 This section contains scenarios showing typical interactions between factory and equipment, including interactions defined by other standards.

RELATED INFORMATION 1
MODELS

NOTE: This related information is not an official part of SEMI E98. This is Related Information for the provisional specification and is not intended to modify or supersede the official standard. Determination of the suitability of the material is solely the responsibility of the user.

This section provides examples of equipment models to clarify the standard.

R1-1 Linked Litho Model

R1-1.1 This section describes the application of OBEM objects to a linked litho as illustrated in Figure R1-1.

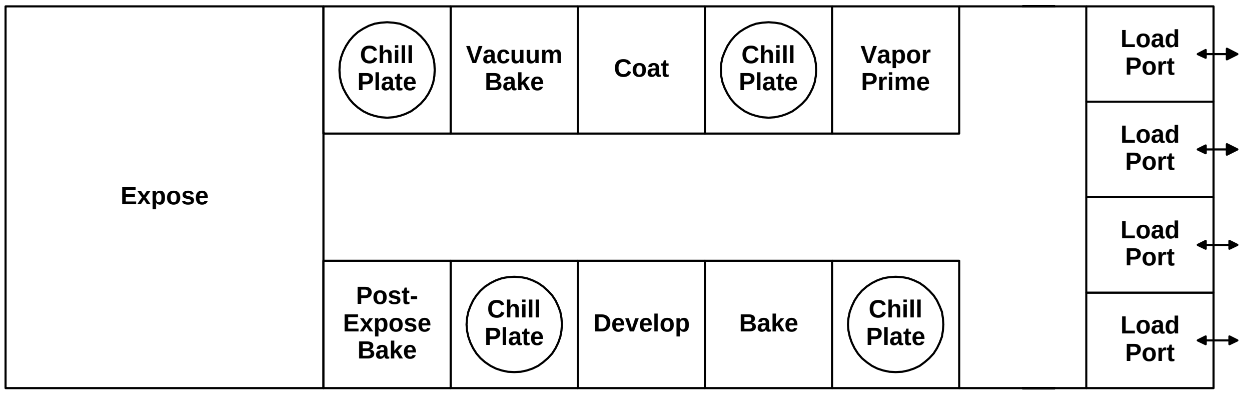


Figure R1-1
Illustration of Linked Litho

R1-1.2 A linked litho is an aggregate created by physically connecting an apply/develop track to a lithography equipment. Work enters through the track where it goes through a set of initial processing steps. Individual substrates are passed to the lithography equipment where they are exposed and returned to another part of the track, where they complete another set of processing steps. Both the track and the lithography equipment are capable of acting independently. However, in the linked-litho configuration, they must perform together as a single system and are treated as a single Equipment by the factory.

R1-1.3 *Apply/Develop Track Equipment*

R1-1.3.1 Before considering the object model for a linked litho, it is helpful to examine the model for an apply/develop track as a type of stand-alone equipment. This is illustrated in Figure R1-2.

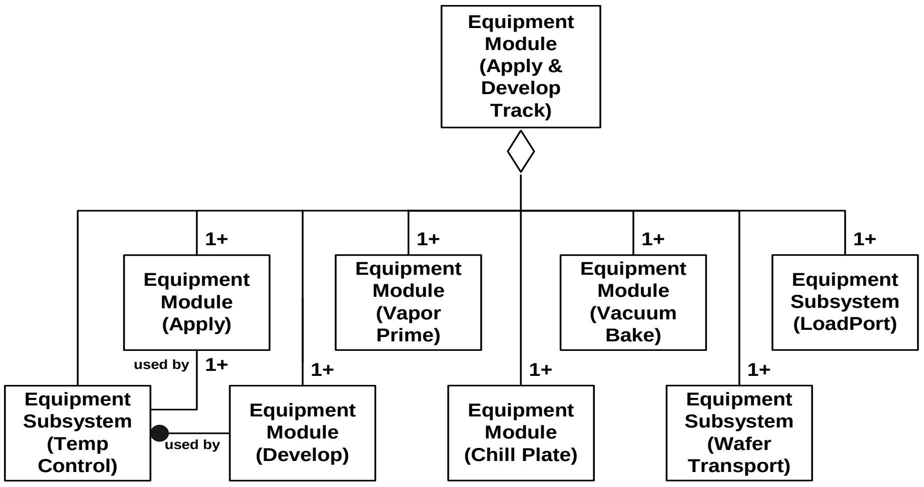

Equipment Module (Apply & Develop Track)
1+
Equipment Module (Apply)
used by
1+
Equipment Subsystem (Temp Control)
used by
1+
Equipment Module (Develop)
1+
Equipment Module (Vapor Prime)
1+
Equipment Module (Chill Plate)
1+
Equipment Module (Vacuum Bake)
1+
Equipment Subsystem (Wafer Transport)
1+
Equipment Subsystem (LoadPort)

Figure R1-2
Apply and Develop Track Equipment

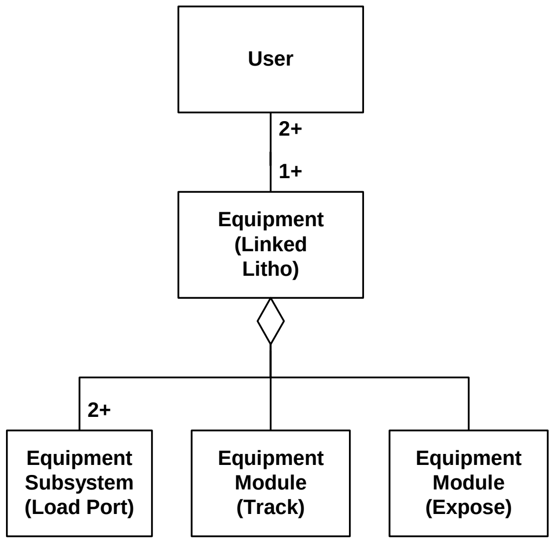


Figure R1-3
Object Model for Linked Litho Equipment

R1-1.4 *Combined Linked Litho*

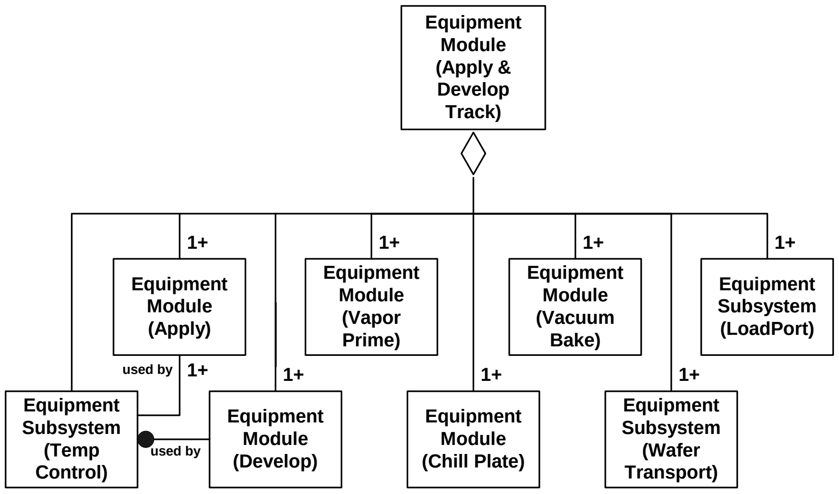

Equipment Module (Apply & Develop Track)
1+
Equipment Module (Apply)
1+
Equipment Module (Vapor Prime)
1+
Equipment Module (Vacuum Bake)
1+
Equipment Subsystem (LoadPort)
used by
1+
1+
1+
1+
Equipment Subsystem (Temp Control)
used by
Equipment Module (Develop)
Equipment Module (Chill Plate)
Equipment Subsystem (Wafer Transport)

Figure R1-4
Linked Litho Track Module

R1-1.4.1 The Linked Litho Equipment has two major EquipmentModules, the track module and the expose module in this model. The track, in turn, is composed of various types of Sub-modules, such as the Vapor Prime module, the Spin module, and so forth.

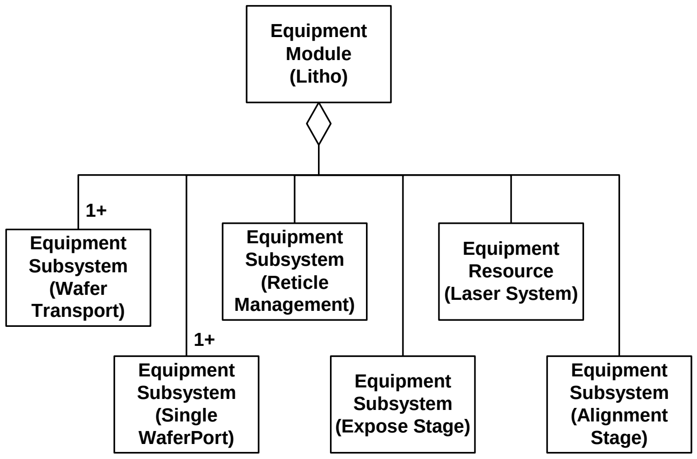

Equipment Module (Litho)
1+
Equipment Subsystem (Wafer Transport)
Equipment Subsystem (Reticle Management)
Equipment Resource (Laser System)
1+
Equipment Subsystem (Single WaferPort)
Equipment Subsystem (Expose Stage)
Equipment Subsystem (Alignment Stage)

Figure R1-5
Linked Litho Expose Module Object Model

R1-1.4.2 Figure R1-6 shows the makeup of another track module. This process may be extended as many times as is required by the particular installation.

R1-2 300 mm Equipment

R1-2.1 Because of the size and the weight of wafers with a diameter of 300 mm or greater, a 25-wafer cassette is too heavy for repetitive lifting by people. 300 mm factories must be able to integrate automated material handling systems and production equipment. To achieve this, equipment loadports must be standardized for access by different material handling technologies such as overhead tracks and rail-guided vehicles in addition to personal-guided vehicles (PGVs) used by operators.

R1-2.2 In addition to issues regarding access, users require that equipment be capable of continuous processing. To meet this requirement, certain types of equipment need to provide intermediate storage for carriers to prevent starvation, running out of work before new work is delivered.

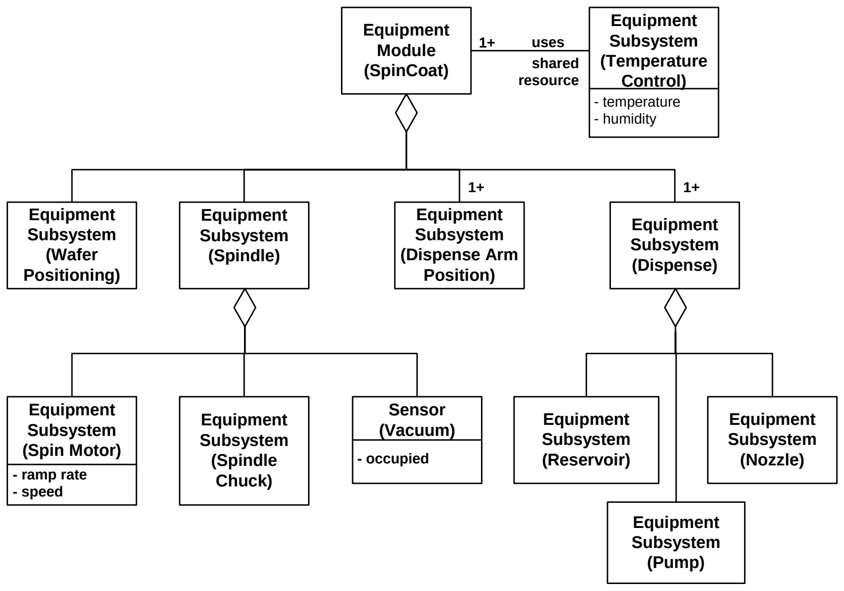

Equipment Module (SpinCoat)
1+
uses
shared resource
Equipment Subsystem (Temperature Control)
- temperature
- humidity
Equipment Subsystem (Wafer Positioning)
Equipment Subsystem (Spindle)
1+
Equipment Subsystem (Dispense Arm Position)
1+
Equipment Subsystem (Dispense)
Equipment Subsystem (Spin Motor)
- ramp rate
- speed
Equipment Subsystem (Spindle Chuck)
Sensor (Vacuum)
- occupied
Equipment Subsystem (Reservoir)
Equipment Subsystem (Nozzle)
Equipment Subsystem (Pump)

Figure R1-6
Spin/Coat Track Module Object Model

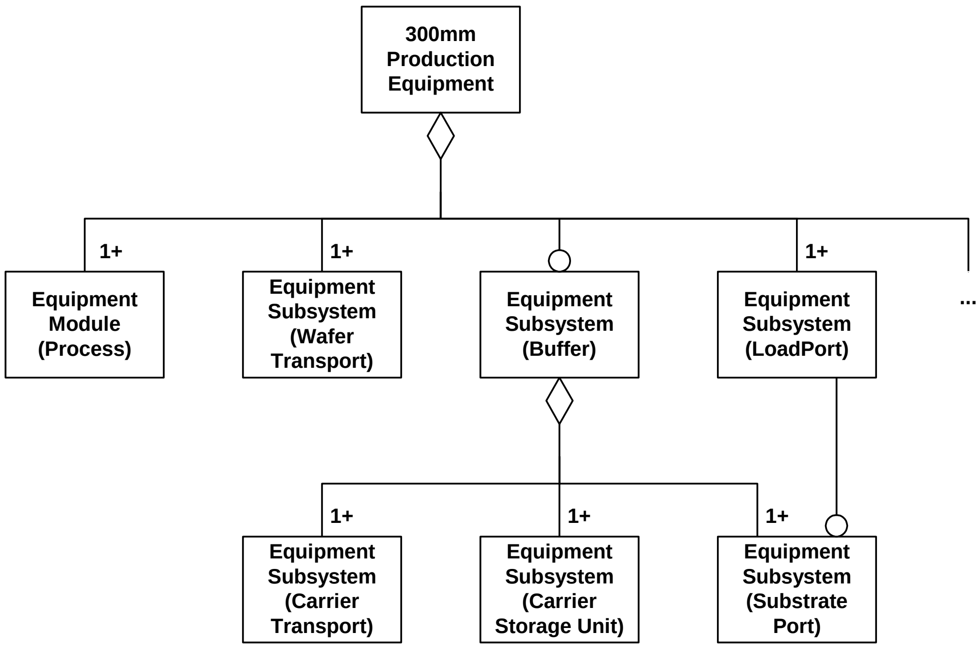

300mm Production Equipment
1+
Equipment Module (Process)
1+
Equipment Subsystem (Wafer Transport)
Equipment Subsystem (Buffer)
1+
Equipment Subsystem (LoadPort)
...
1+
Equipment Subsystem (Carrier Transport)
1+
Equipment Subsystem (Carrier Storage Unit)
1+
Equipment Subsystem (Substrate Port)

Figure R1-7
Example of Object Model for 300 mm Equipment

R1-2.3 The term “Equipment Front End Module” (EFEM) refers to the aggregation shown in Figure R1-8 consisting of all the components handling carriers, i.e., the loadport and buffer subsystems. An EFEM would not typically provide ARAMS capability and therefore would be considered as an EquipmentSubsystem and not considered as an OBEM EquipmentModule.

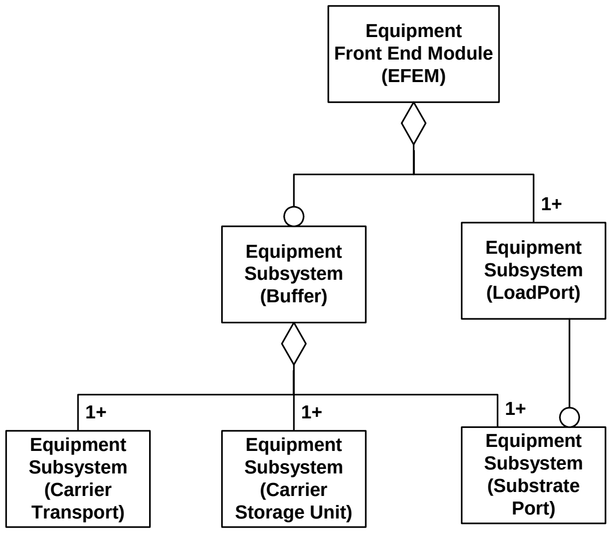

Equipment Front End Module (EFEM)
1+
Equipment Subsystem (Buffer)
Equipment Subsystem (LoadPort)
1+
1+
1+
Equipment Subsystem (Carrier Transport)
Equipment Subsystem (Carrier Storage Unit)
Equipment Subsystem (Substrate Port)

Figure R1-8
Equipment Front End Module (EFEM)

RELATED INFORMATION 2
RELATIONSHIP TO THE CIM FRAMEWORK

NOTE: This related information is not an official part of SEMI E98. This is Related Information for the provisional specification and is not intended to modify or supersede the official standard. Determination of the suitability of the material is solely the responsibility of the user.

This section discusses the relationship of OBEM and the OBEM objects to the objects defined in SEMI E81. It uses terms both from the CIM Framework and from OBEM. It is not part of the OBEM standard.

Additional documentation should be provided in this section as the CIM Framework develops.

R2-1 Equipment and Machine

R2-1.1 The interface to Equipment defined in OBEM is provided for the physical equipment on the manufacturing floor. From time to time, the Equipment will be powered off for maintenance and at these times is unable to communicate. In contrast, the Machine object, defined in the CIM Framework, is part of the manufacturing execution system and is always available, even when the Equipment is powered off or otherwise unable to communicate.

R2-1.2 The Machine provides a generic interface for factory logistics and is able to provide some services without relying on the Equipment itself. The Machine is responsible for communications with the Equipment and for representing the Equipment to the Factory and representing the Factory to the Equipment.

R2-2 EquipmentModule and MachineResource

R2-2.1 The EquipmentModule defined in OBEM corresponds to the MachineResource interface defined in the CIM Framework. The MachineResource is a type of Resource and therefore can respond to requests to start up and shut down. These services are provided for the EquipmentModule but not for its supertypes.

R2-3 Location Tracking

R2-3.1 The CIM Framework defines an interface for a MaterialTrackingLocation. The MaterialTracking-Location differs from the OBEM MaterialLocation. MaterialLocations represent places within the equipment itself where carriers and substrates can be placed, including wafer chucks, carrier slots, and robot end effectors. A MaterialTrackingLocation, in contrast, is at a higher level of granularity and provides the ability to locate material associated with a Machine. MaterialTrackingLocation does not track material locations down to the level of specific places within the equipment.

RELATED INFORMATION 3
ADDITIONAL DATA

NOTE: This related information is not an official part of SEMI E98. This is Related Information for the provisional specification and is not intended to modify or supersede the official standard. Determination of the suitability of the material is solely the responsibility of the user.

This section provides examples of equipment models to clarify the standard.

This section contains additional information that is not part of the standard.

R3-1 Representations of Date and Time

R3-1.1 This section discusses several date and time representations, and provides background information in support of the OBEM adoption of an IEEE floating point representation of date and time values.

R3-1.2 In this section, the term "date value" means an encoding of a date and time value, such as 12/25/1997 11:04:56.

R3-1.3 *ASCII Digit Representation*

R3-1.3.1 The standard date and time format in SECS communication is the ASCII representation "YYYYMMDDhhmmsscc". In this representation, date and time are represented as a fixed length sequence of ASCII digit characters, grouped into fields as indicated.

Advantages	*Disadvantages*
Very simple conversion to display format.	Date computations difficult. For example, adding two date values is not a trivial operation.
Simple to extract date and time from date value.	Many digit sequences are not valid date values.
	The format does not lend itself to expressions of time intervals. The representation of "ten minutes" might be "00000000001000", but the job of adding this interval to a date value is not simple.
	The resolution of the representation is fixed at seconds.

R3-1.4 *Long Signed Integer Representation*

R3-1.4.1 UNIX represents date and time as the number of ticks since Epoch. Typically the time interval of a tick is one second. Date values are expressed as the number of ticks since a certain instant, called the Epoch.

R3-1.4.2 There are 31,536,000 seconds in a 365 day year. If stored as a four byte signed integer, with ticks in seconds, this representation will overflow after approximately 68 years. Eight byte integers provide sufficient range for dates spanning human history many times. Unfortunately, eight byte integers are inconvenient on many computer platforms.

Advantages	*Disadvantages*
All bit sequences are valid date values.	Difficult to separate date and time from date value.
Intervals are simple to express.	Eight byte representation required for sufficient range, but inconvenient on many computer platforms.
Date value arithmetic is simple.	No simple way to convert date value to display format.
	The resolution of the representation is fixed at the tick value, typically seconds.

R3-1.5 *Floating Point Representation*

R3-1.5.1 This representation is used in many applications. It represents date values as floating point numbers, with units of days past 12/30/1899. It is very similar to the "Ticks since the Epoch" representation, except that the tick values are days.

R3-1.5.2 This representation has the advantages of the “Tick since Epoch” representation for date value arithmetic. Second, the separation of the date and time from a date value is simple. The date is the whole value, the time is the fractional value. Third, the minimum time resolution is very small, and is not fixed by the representation.

R3-1.5.3 The following calculation is used to determine roughly the minimum time resolution that can be expressed:

R3-1.5.4 The IEEE 8 byte floating point representation uses an excess 128 notation. This means 7 bits of exponent and one bit of sign. This leaves 7 bytes of significant. On November 18, 1997 the day number is 35752. This is hex 8BA8, and requires two bytes. This leaves 5 bytes of significant to represent the fractional day, or the time portion of the date value. These 5 bytes of significant contain 40 bits, and can represent values to 2^ 40, or one part in 1,099,511,627,776. There are 86,400 seconds in one day, so this representation is accurate to approximately 7.86E-08 seconds, or about 1/10 of a microsecond. In about a hundred years the day number will have doubled, and this resolution will have shrunk by a factor of 2. For practical purposes the resolution is no worse than one microsecond resolution.

Advantages	*Disadvantages*
All legal float values are valid date values.	Minimum resolution changes over time.
Simple to separate date and time from a date value.	No simple way to convert date value to display format (but simpler than the “Tick since the Epoch” representation).
Intervals are simple to express.	
Date value arithmetic is simple.	
The eight byte floating point representation used is supported on most computer platforms.	
The time resolution is not implied by the representation, but is valid to very small intervals.	

R3-1.6 *Examples*

11/18/1997	35752
11/18/97 1:03:13 PM	35752.543912037
One second	1.15740740740741E-05
One day	1
One microsecond	1.15740740740741E-11
1/1/1000	-328716
12/30/1899	0

RELATED INFORMATION 4
OBEM EQUIPMENTIODEVICE AND SENSOR/ACTUATOR NETWORK COMMON DEVICE MODEL

NOTE: This related information is not an official part of SEMI E98 and was derived from work developed in the Object-Based Model Task Force in North America, and the Sunsor/Actuator Bus subcommittee. This related information was approved for publication by full letter ballot on April 30, 2001.

R4-1 Relationship between OBEM EquipmentIODevice and Sensor/Actuator Network Common Device Model

R4-1.1 This section describes the relationship between the EquipmentIODevice defined in Section 10.3 and the Common Device Model defined in SEMI E54.1 Standard for Sensor/Actuator Network Common Device Model.

R4-1.2 SEMI E54.1 defines in detail the internal structure of a sensor/actuator device on an E54-compliant sensor/actuator network. That is, it defines the possible internal components of such a device and the relationships between these components. To ensure proper behavior, it was necessary to give sufficient guidance to implementers using an E54 network communication protocol. E54.1 defines sensor (input) objects, actuator (output) objects, controller objects, a sensor/actuator controller object (SAC), and a device manager object (DM) as individual components of a sensor/actuator device aggregation as components of an aggregate Common Device Model (CDM). The device aggregation must have at least one sensor or actuator or controller (combined) and exactly one SAC and one DM. In addition, the CDM defines additional classes to further classify components, such as the ActiveElement, Sensor, and Sensor-Analog Input classes, which should be considered. Some attributes of the EquipmentIODevice Observables only apply to elements of certain classes.

R4-1.3 E54.1 does not define the CDM, which is the aggregation of these elements, as an object in its own right,. Such an aggregation might have the sum of all of the attributes of its components.

R4-1.4 OBEM defines a high-level device, the EquipmentIODevice, that is of interest to factory system applications such as Advanced Process Control (APC) and Fault Detection Classification (FDC). These systems are seldom interested in all of the low-level details. Actual raw data as read directly by an analog sensor may not be desired but rather a calculated value that has been converted into some specified units, such as degrees Celsius. OBEM can support either.

R4-1.5 Some of the attributes of the EquipmentIODevice are maintained only by the equipment. Most of the attributes must come directly from the attributes read from the CDM components for E54.1 compliant devices. The following sections suggest a way to relate attributes of the EquipmentIODevice and attributes of the elements from the CDM in a consistent manner.

R4-1.6 Each individual sensor or actuator or controller element within the CDM corresponds to one Observable in the EquipmentIOElement model. Observables were not made into formal objects, as they could be, but rather are included as one set of data within the EquipmentIODevice. Each EquipmentIODevice may have multiple sets of Observables data.

R4-1.7 Table R4-1 lists all of the attributes of an EquipmentIODevice, including those it inherits from the AbstractEquipmentElement (marked with asterisks) and shows the object and attribute in the General Device Model hierarchy to which it corresponds, where a correspondence exists.

R4-1.7.1 The “em dash” character “—”is used where no correspondence exists.

R4-1.7.2 This table is intended for equipment implementers of E98 and not for the suppliers of E54-compliant devices.

Table R4-1 Relationships between EquipmentIOElement and Generic Device Model

EquipmentIODevice Attribute Name	*Generic Device Model Object Name*	*Corresponding Attribute Name*
ObjType	—	—
ObjID	—	—
AlgorithmID	—	—
Cycles* (units may be time)	SAC	Run Hours (SacA5)
Description*	—	—
DeviceType	Device Manager	Device Type (DmA1)
Function*	—	—
HardwareRevision	Device Manager	Hardware Revision Level (DmA6)
ImmutableID*	DeviceManager	Serial Number (optional) (DmA7)
InService*	—	—
ModelNumber	Device Manager	Manufacturer Model Number (DmA4)
NumberofObservables	—	—
SoftwareRevision	Device Manager	Software or Firmware Revision Level (DmA5)
Supplier*	Device Manager	Device Manufacturer Identifier (DmA3)
	Observable$_I$	
DataDimension	Sensor-Analog Input Class element	—
Datatype	Sensor-Analog Input Class or Actuator-Analog Output Class or Controller Class element	Datatype SaiA66or AaoA66 or Controller CA19 (translated for host as necessary to conform to SECS-II format types (10,11,20, etc.)
DeltaValue	Sensor of Sensor-Analog Input Class element	ReportDelta (SaiA70)
InterlockStatus	—	—
Name	Sensor or Actuator or Controller	Text strings “SenIn” or “ActIn” or “CntIn” where n represents the instance number of the corresponding device object (Table 2, E54.1)
NominalValue	—	—
ObservableType	—	—
Possible Units	—	—
Range	—	—
ReadRate	—	—
ReportChange	Sensor-Analog Input Class element	EnableReportROC SaiA71
ReportRate	Sensor-Analog Input Class element	ReportROC SaiA72
Timestamp	Device Manager	Date and Time (DmA21)
Units	Sensor-Analog Input or Actuator-Analog Output Class element	DataUnits (SaiA67 or Aao67)
Value	Sensor or Actuator or Controller	Value or Setting (nA16) or Setpoint (CA16)

SEMI E98.1-1102
PROVISIONAL SPECIFICATION FOR SECS-II PROTOCOL FOR THE OBJECT-BASED EQUIPMENT MODEL

This 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 August 29, 2002. Initially available at www.semi.org September 2002; to be published November 2002. Originally published March 2002.

1 Purpose

1.1 This document maps the services and data of SEMI E98 Object-Based Equipment Model (OBEM) to SECS-II streams and functions and data definitions.

2 Scope

2.1 This document applies to all implementations of OBEM that use the SECS-II message protocol (SEMI E5).

2.2 This specification is provisional. To remove the provisional status, the SECS-II format for the attributes of the remaining object defined in OBEM must be included in Section 6:

- EquipmentIODevice

2.3 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 health practices and determine the applicability or regulatory limitations prior to use.

3 Referenced Standards

3.1 *SEMI Standards*

SEMI E5 — SEMI Equipment Communications Standard 2 Message Content (SECS-II)

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

SEMI E39.1 — SECS-II Protocol for Object Services Standard (OSS)

SEMI E98 — Provisional Standard for the Object-Based Equipment Model (OBEM)

NOTE 1: Unless otherwise indicated, all documents cited shall be the latest published versions.

4 Services Mapping

4.1 Table 1 shows the specific SECS-II streams and functions that shall be used for SECS-II implementations of the services defined in OBEM.

Table 1 Services Mapping Table

Message Name	*Stream, Function*	*SECS-II Message Name*
Abort	S14,F19/20	Generic Service Request/Acknowledge
AddElement	S14,F19/20	Generic Service Request/Acknowledge
AddProcessCapability	S14,F19/20	Generic Service Request/Acknowledge
ChangeService	S14,F19/20	Generic Service Request/Acknowledge
Cleanup	S14,F19/20	Generic Service Request/Acknowledge
GetDateTime	S14,F19/20	Generic Service Request/Acknowledge
GetProgramDirectory	S14,F19/20	Generic Service Request/Acknowledge
Pause	S14,F19/20	Generic Service Request/Acknowledge
RemoveProcessCapability	S14,F19/20	Generic Service Request/Acknowledge
RemoveElement	S14,F19/20	Generic Service Request/Acknowledge
Resume	S14,F19/20	Generic Service Request/Acknowledge
SelectProgram	S14,F19/20	Generic Service Request/Acknowledge
SetDateTime	S14,F19/20	Generic Service Request/Acknowledge
Shutdown	S14,F19/20	Generic Service Request/Acknowledge
Start	S14,F19/20	Generic Service Request/Acknowledge
Startup	S14,F19/20	Generic Service Request/Acknowledge
Stop	S14,F19/20	Generic Service Request/Acknowledge

5 Service Parameter Mapping

5.1 Table 2 shows the mapping between service parameters defined by OBEM and the data items defined by SEMI E5. All OBEM messages map to Stream 14 Functions 19 and 20. The data item SVCNAME is the text string representation of the service name found in Table 1 and should be assumed as case-sensitive. The name of each parameter is the text string representation of the parameter name as it appears in Table 2. The SECS-II Data Item Reference, where different from SPVAL, provides additional information and restrictions for the value.

Table 2 Service Parameter Mapping Table

Parameter Name	*Parameter Range*	*SECS-II Data Item Reference*	*SECS-II Format*

DateTime	A valid date and time.	L,2
		1. <SPNAME>
		2. <SPVAL>
			SPNAME = "DateTime"
			SPVAL = (20,34,30)

DeviceProgramID	0–65,535	L,2
		1. <SPNAME>
		2. <SPVAL>
			SPNAME = "DeviceProgramID"
			SPVAL = (51,52)

ErrorCode	Enumerated as defined in SEMI E5 or by supplier.	ERRCODE	5()
ErrorText	1–80 characters	ERRTEXT	20
ObjSpec	0–80 characters	OBJSPEC	20

ProcessCapability	1–80 characters	L,2
		1. <SPNAME>
		2. <SPVAL>
			SPNAME = "ProcessCapability"
			SPVAL = (20)

Property	Each property is an attribute name/value pair for some object indicated in ObjSpec. Each attribute name maps directly to ServiceParameterName and each attribute value maps directly to ServiceParameterValue.	L,2 1. <SPNAME> 2. <SPVAL>	—
RequestStatus	Structure	L,2 1. <SVCACK> 2. Status	
RequestAcknowledge	Enumerated: 0 = Request performed 1 = Invalid request 2 = Can not perform now 4 = Will perform when able and report results when complete.	SVCACK	10

Service	Enumerated:
	0 = OUT OF SERVICE
	1 = IN SERVICE
		L,2
		1. <SPNAME>
		2. <SPVAL>
			SPNAME = "Service"
			SPVAL = (51)

ServiceParameterName	Text	SPNAME	20
ServiceParameterValue	Depends on parameter used.	SPVAL	10, 20, 34, 40, 51, 52

Parameter Name	*Parameter Range*	*SECS-II Data Item Reference*	*SECS-II Format*
Status	(list of) Structure	L,m 1. L,2 1. <ERRCODE$_1$> 2. <ERRTEXT$_1$> m. L,2 1. <ERRCODE$_m$> 2. <ERRTEXT$_m$>	

5.2 Table 3 shows the data items in SECS II messages that do not have a corresponding service parameter.

Table 3 Additional Data Item Requirement Table

Function	*SECS-II Data Item*
Used to satisfy SECS-II conventions for linking a multi-block inquiry with subsequent multi-block message.	DATAID
Name of OBEM service. Enumerated: "Abort" "Add Element" "AddProcessCapability" "ChangeService" "Cleanup" "GetDateTime" "GetProgramDirectory" "Pause" "RemoveProcessCapability" "RemoveElement" "Resume" "SelectProgram" "SetDateTime" "Shutdown" "Start" "Startup" "Stop"	SVCNAME
A unique number generated by the service requestor to identify a specific operation and connect it to a later completion confirmation.	OPID

NOTE 1: The text strings specified in Table 3 for SVCNAME shall be recognized by the equipment as vaild, whether the equipment is or is not case-sensitive.

6 Object Attribute Form

6.1 Section 6 defines the SECS-II form of the attributes of objects defined in OBEM, presented in alphabetical order. Note that attribute names in this section indicate the literal text string that shall be accepted in string comparisons for attribute names in Stream 14 messages.

6.1.1 By convention, all tables include ObjID as well as ObjType as the two attributes required by SEMI E39.

6.2 *AbstractEquipmentElement*

6.2.1 Table 5 specifies the attributes of the AbstractEquipmentElement.

Table 4 AbstractEquipmentElement Object Attribute Specification

Attribute Name	*Range/Value*	*SECS-II Format*
"Cycles"	0+.	54
"Description"		20
"Function"		20
"ImmutableID"		20

Attribute Name	*Range/Value*	*SECS-II Format*
"InService "	Enumerated: 0 = OUT OF SERVICE 1 = IN SERVICE	51
"ObjID"	Conforms to OBJID restrictions in SEMI E39.1, Section 6.	20
"ObjType"	"EqpElement"	20
"ResetDate"	Conforms to DateTime attribute of Clock	20,30,34
"Supplier"		20

6.3 *AbstractEquipmentModule*

6.3.1 Table 6 specifies the elements of the AbstractEquipmentModule object.

Table 5 AbstractEquipmentModule Object Attribute Specification

Attribute Name	*Range/Value*	*SECS-II Format*
"BehaviorState"	Enumerated: 0 = IDLE 1 = ACTIVE SERVICE 2 = PAUSING 3 = PAUSED 4 = STOPPING 5 = STOPPED 6 = ABORTING 7 = ABORTED 8 = IDLE WITH ALARMS	
"Model"		20
"ModelRevision"		20
"Nickname"		20
"ObjID"	Conforms to OBJID restrictions in SEMI E39.1, Section 6.	20
"ObjType"	0 or more	20
"ProcessType"		20
"ProcessCapabilityList"		0
"ProcessSetup"		20
"PreviousBehaviorState"	Enumerated: 0 = IDLE 1 = ACTIVE SERVICE 2 = PAUSING 3 = PAUSED 4 = STOPPING 5 = STOPPED 6 = ABORTING 7 = ABORTED 8 = IDLE WITH ALARMS	51
"SoftwareVersions"		0
"Units"	Conforms to SEMI E5, Section 9.	20

6.4 *AbstractEquipmentSubsystem*

6.4.1 Table 7 specifies the attributes of the AbstractEquipmentSubsystem.

Table 6 AbstractEquipmentSubsystem Object Attribute Specification

Attribute Name	*Range/Value*	*SECS-II Format*
“ObjType”	“AbstractEqpSubsystem”	20
“ObjID”	Conforms to OBJID restrictions in SEMI E39.1, Section 6.	20
“MaterialSummary”	Ordered list. See Table 8 for detail.	0

6.4.2 Table 8 specifies the elements of the MaterialSummary attribute.

Table 7 MaterialSummary Attribute Specification

Attribute Name	*Range/Value*	*SECS-II Format*
“MaterialCapacity”	0 or more	5()
“MaterialCount”	0 ≤ MaterialCount ≤ Material-Capacity	5()
“MaterialType”	Conforms to SEMI E5, Section 9.	20

6.5 *CarrierLocation Object*

6.5.1 Note that the CarrierLocation inherits the attributes of the MaterialLocation.

6.5.2 Table 9 specifies the additional attributes of the CarrierLocation object. See Table 13.

Table 8 CarrierLocation Object Attribute Specification

Attribute Name	*Range/Value*	*SECS-II Format*
“ObjType”	“CarrierLoc”	20
“ObjID”	Conforms to OBJID restrictions in SEMI E39.1, Section 6.	20
LocationState	Enumerated: 0 = Unoccupied 1 = Occupied 2 = Not Aligned	51

6.6 *Clock*

6.6.1 Table 10 specifies the attributes of the Clock object.

Table 9 Clock Object Attribute Specification

Attribute Name	*Range/Value*	*SECS-II Format*
“ObjType”	“Clock”	20
“ObjID”	Conforms to OBJID restrictions in SEMI E39.1, Section 6.	20
“DateTime”	As specified by TimestampFormat.	20,30,34

“UseNet”	FALSE = DISABLED
	TRUE = ENABLED
		11

“TimestampFormat”	Enumerated: (see SEMI E54.1, Section 7.3.1.23) 0 = Text “yyyymmddhhmmsscc” 1 = Univeral Time Coordinated 2 = Standard Time and Date	51
“GMTDelta”	-32,768 to +32767 (minutes) *actual value in use at locale	32

“UseDelta”	Boolean:
	TRUE = use GMDelta
	FALSE = do not use GMDelta
		11

6.7 *Equipment*

6.7.1 Table 11 specifies the attributes of the Equipment object.

Table 10 Equipment Object Attribute Specification

Attribute Name	*Range/Value*	*SECS-II Format*
"ObjType"	"Equipment"	20
"ObjID"	Conforms to OBJID restrictions in SEMI E39.1, Section 6.	20
"AssignedOperators"	List.	0
"Nickname"	Any.	20

6.8 *EquipmentSubsystem*

6.8.1 Table 12 specifies the attributes of the AbstractEquipmentSubsystem object.

Table 11 EquipmentSubsystem Object Attribute Specification

Attribute Name	*Range/Value*	*SECS-II Format*
"ObjType"	"EqpSubsystem"	20
"ObjID"	Conforms to OBJID restrictions in SEMI E39.1, Section 6.	20

6.9 *Material Location*

6.9.1 Table 13 specifies the attributes of the AbstractEquipmentSubsystem object.

Table 12 MaterialLocation Object Attribute Specification

Attribute Name	*Range/Value*	*SECS-II Format*
"ObjType"	"MatlLoc"	20
"ObjID"	Conforms to OBJID restrictions in SEMI E39.1, Section 6.	20
LocationState	Enumerated: 0 = Unoccupied 1 = Occupied	51
MaterialType	Conforms to SEMI E5, Section 9.	20
MaterialID	Conforms to OBJID restrictions in SEMI E39.1, Section 6.	20

SEMI E102-0600
PROVISIONAL SPECIFICATION FOR CIM FRAMEWORK MATERIAL TRANSPORT AND STORAGE COMPONENT

This provisional specification was technically approved by the Global Information and Control Committee and is the direct responsibility of the Japanese Information and Control Committee. Current edition approved by the Japanese Regional Standards Committee on March 14, 2000 and by the North American Regional Standards Committee on March 2, 2000. Initially available at www.semi.org March 2000; to be published June 2000.

1 Purpose

1.1 Wafer fabrication factories will require the baseline capabilities of stocker storage and interbay transport. In addition to these baseline capabilities, intrabay transport will be added as a result of ergonomic and safety requirements brought about by the increased size and weight of 300 mm wafer carriers. These stocker, interbay transport, and intrabay transport systems will be required to be fully integrated with each other and the factory Manufacturing Execution System (MES) in order to realize the full vision of cost effective automated material transport to and from production equipment.

1.2 A baseline requirement of Automated Material Handling System (AMHS) equipment is efficient integration with the factory MES. Therefore, the purpose of this specification is to enable cost effective integration of interoperable AMHS systems, as illustrated in Figure 1. Manufacturers require a SEMI standard which specifies the visible behavior of the AMHS Integration system and its interface to the factory MES. The purpose of this standard is to specify these interfaces and the interactions between MES and AMHS systems as a part of the CIM Framework. While the term AMHS is commonly used to refer to a wide range of equipment and systems that support automated material handling, the CIM Framework specifies a software component called “Material Transport and Storage Component (MTSC)” which represents the standard interface of the MES to the complete suite of AMHS capabilities.

2 Scope

2.1 This specification provides the common interfaces required by Manufacturing Execution Systems for runtime interactions between the factory (as represented by other CIM Framework components) and the Material Tracking and Storage Component. Some interfaces supporting configuration and tracking of the material transport and storage equipment are the responsibility of the Equipment Tracking and Maintenance component of the CIM Framework. These interfaces are complementary to the interfaces of Transport Machines and Storage Machines presented in this specification.

2.2 The responsibilities of the Material Transport and Storage Component include interfaces that

- Support scheduling of material transport and processing by predicting time for material delivery to specific locations. NOTE: The interfaces for delivery time prediction are deferred as one of the deficiencies noted in this provisional specification.
- Execute and monitor transport jobs to move material to specific locations.
    - Validate that the job can be done: the material is available, the destination is reachable and has available storage or loadport capacity, and there are sufficient material movement resources (cars, storage space, etc.) to implement the job.
    - Request operations from AMHS equipment controllers to enact internal material movement and storage actions. Implementations of the Material Transport and Storage Component will use lower level standards such as IBSEM and StockerSEM for communication with equipment that performs the physical movement and storage actions.
    - Collect and record data on transport job execution and history.
    - Capture, record, interpret, and respond to equipment events and fault detection.
    - Generate MES-level job status events and material location change events.
- Record and report on material locations and material transport histories for material in the system.
- Interact with machine and port interfaces for material hand-off handshake protocols.

2.3 Automated Material Handling Systems (AMHS’s) are an important part of any semiconductor factory and have, typically, been implemented and integrated with a

Manufacturing Execution System (MES) as a separate logical software entity. An AMHS is made up of the AMHS Framework, the system controllers for AMHS equipment (the software) and the transport and storage machines (the hardware). In keeping with this tradition, the CIM Framework views the AMHS as a "black box" where the interface into that system is visible, but the inner workings and control of that system is the responsibility of the AMHS supplier and not in the domain of the MES. The CIM Framework specifies the Material Transport and Storage Component (MTSC) as the MES level interface to AMHS capabilities.

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

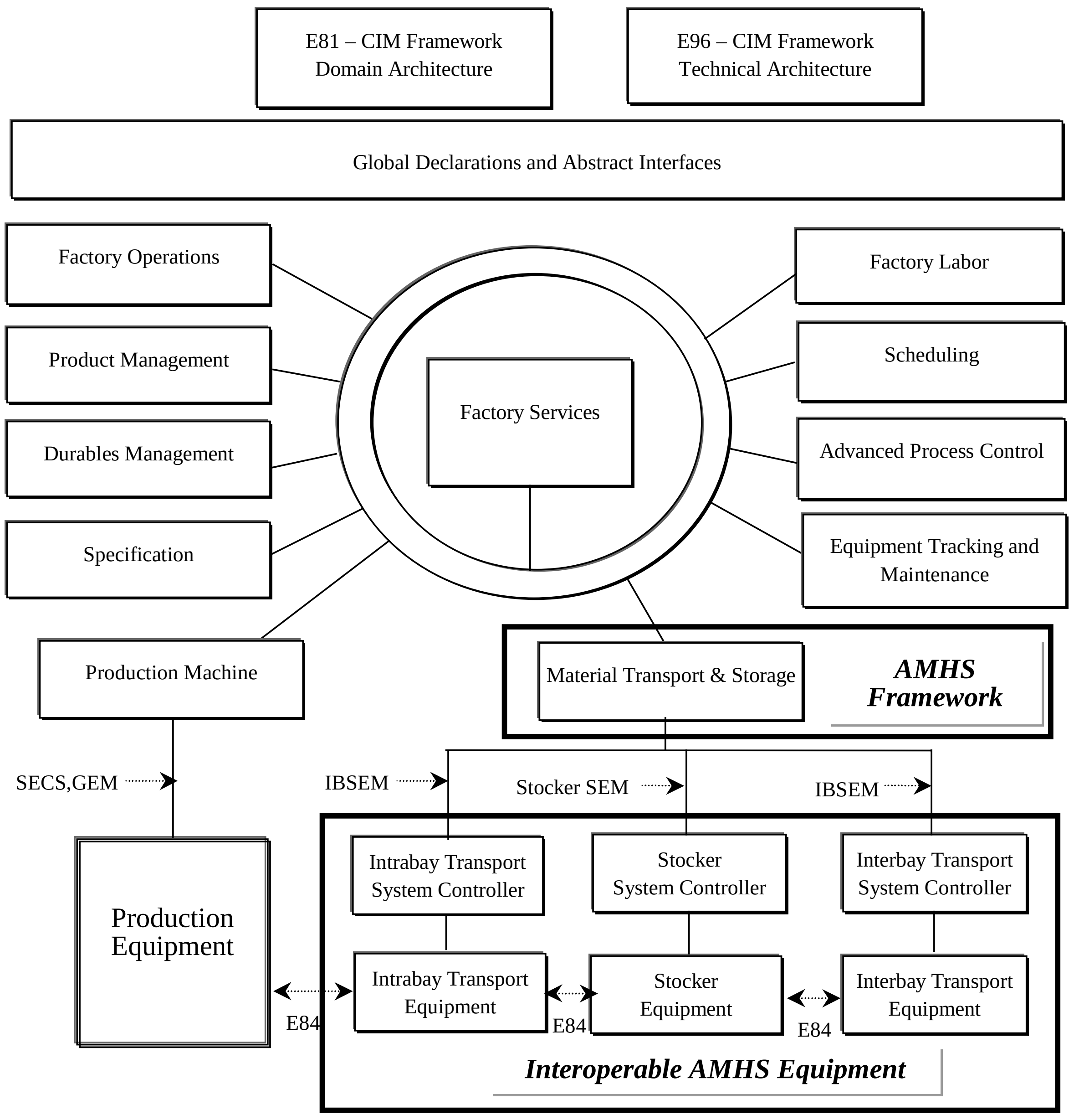

E81 – CIM Framework Domain Architecture
E96 – CIM Framework Technical Architecture
Global Declarations and Abstract Interfaces
Factory Operations
Product Management
Durables Management
Specification
Factory Services
Factory Labor
Scheduling
Advanced Process Control
Equipment Tracking and Maintenance
Production Machine
Material Transport & Storage
AMHS Framework
SECS,GEM
IBSEM
Stocker SEM
IBSEM
Production Equipment
Intrabay Transport System Controller
Stocker System Controller
Interbay Transport System Controller
Intrabay Transport Equipment
Stocker Equipment
Interbay Transport Equipment
E84
E84
E84
Interoperable AMHS Equipment

Figure 1
AMHS Environment

3 Limitations

3.1 The Material Transport and Storage Component provides AMHS Framework level interfaces to interoperable AMHS Equipment. This level of material transport is factory level movement, i.e., the transport of product and durables between machines, areas or material tracking locations. The Material Tracking and Storage Component does not provide interfaces that make material movement within machines visible.

3.2 *Provisional Status* — This specification is designated as provisional due to known areas that need to be completed. The following items summarize the deficiencies of the provisional specification to be addressed before a subsequent ballot to upgrade it to full standard status.

3.2.1 *Dependence on Global Declarations and Abstract Interfaces* — Components of this document reference global declarations and specialize abstract interfaces for material, resource and job concepts. The provisional specification for these declarations and interfaces was balloted and recommended by the SEMI Information and Control Committee in July, 1999. When this specification is published a revision will be undertaken to clearly define the relationship of this specification to those common elements of the CIM Framework.

3.2.2 *Dependence on Durables Management Component* — The Material Transport and Storage Component depends upon interfaces specified as part of the Durables Management Component of the CIM Framework. This document includes only the required interfaces from Durables Management. Those interfaces, found in Section 6.2.3, will be removed from this document when the Durables Management Component of the CIM Framework is adopted in a future ballot.

3.2.3 *Dependence on Machine Abstract Interface Group* — The Material Transport and Storage Component depends upon interfaces specified as part of the Machine Abstract Interface Group (Machine AIG) of the CIM Framework. This document includes only the required interfaces from the Machine AIG. Those interfaces, found in Section 6.2.4, will be removed from this document when the Machine AIG of the CIM Framework is adopted in a future ballot.

3.2.4 *Dependence on Factory Operations Component* — The Material Transport and Storage Component depends upon interfaces specified as part of the Factory Operations Component of the CIM Framework. This document includes only the required interfaces from the Factory Operations Component. Those interfaces, found in Section 6.2.5, will be removed from this document when the Factory Operations Component of the CIM Framework is adopted in a future ballot.

3.2.5 *Deterministic Prediction of Transport Time* — The responsibility for providing estimates of predicted transport time of future TransportJobs in a deterministic way is beyond the scope of the Provisional Specification. This capability may be offered in future upgrades of this specification based on added experience with such predictions. It may be found that heuristic methods for transport time prediction are the most practical way to provide such estimates.

4 Referenced Standards

4.1 *SEMI Documents*

SEMI E81 — Provisional Specification for CIM Framework Domain Architecture

4.2 *Other Documents*

ISO/IEC International Standard 14750 (also ITU-T Recommendation X.920): Information Technology – Open Distributed Processing – Interface Definition Language[1]

UML Notation Guide, Version 1.1, document number ad/97-08-05, Object Management Group[2]

5 Terminology

5.1 *Abbreviations and Acronyms*

5.1.1 *AMHS* — Automated Material Handling System

5.1.2 *MTSC* — Material Transport and Storage Component

6 Requirements

6.1 The following sections specify the interfaces that comprise the Material Transport and Storage Component of the CIM Framework. Additionally, the subsets of interfaces from Durables Management, Machine Abstract Interface Group, and Factory Operations required to support Material Transport and Storage are included here pending future ballots that will address the full scope of these other specifications.

6.1.1 Figure 2 illustrates the context of the Material Transport and Storage Component within the CIM Framework. The Factory Operation component requests the transport of material by the Material Transport Manager which provides services for creation

1 ISO Central Secretariat, 1, rue de Varembé, Case postale 56, CH-1211 Genève 20, Switzerland

2 UML Notation Guide v1.1 is available to the general public at http://www.omg.org/cgi-bin/doclist.pl, +1-508-820 4300, Object Management Group, Inc. , Framingham Corporate Center, 492 Old Connecticut Path, Framingham, MA 01701.

of Transport Jobs. The Material Transport and Storage Component uses services provided by the Machine AIG to access interfaces of Machines and Ports that serve as the source or destination of Transport Jobs. The MES Factory defines areas within a factory which represent groups of machines. Finally, Durables Management provides the services needed to manage the material containers that are moved.

6.1.2 Figure 3 shows the Material Transport and Storage Component information model, including the inherited interfaces and the associations between interfaces. These interfaces are fully defined in Sections 6.3 and 6.4. The interfaces associated with the Material Transport and Storage are as follows:

1. TransportJobSupervisor – abstract interface,
2. MaterialTransportManager,
3. MaterialTransportController, and
4. TransportJob.

6.2 *Required Subsets of Future CIM Framework Specifications*

6.2.1 This specification depends on interfaces from Durables Management Component, the Machine Abstract Interface Group and the Factory Operations Component that will eventually be contained in future CIM Framework component specifications. They are included with this document to support a complete specification for AMHS support within the CIM Framework. When these items are balloted as separate documents, they will be removed from this specification and referenced in their full form.

6.2.2 Figure 4 illustrates the dependencies of the Material Transport and Storage Component.

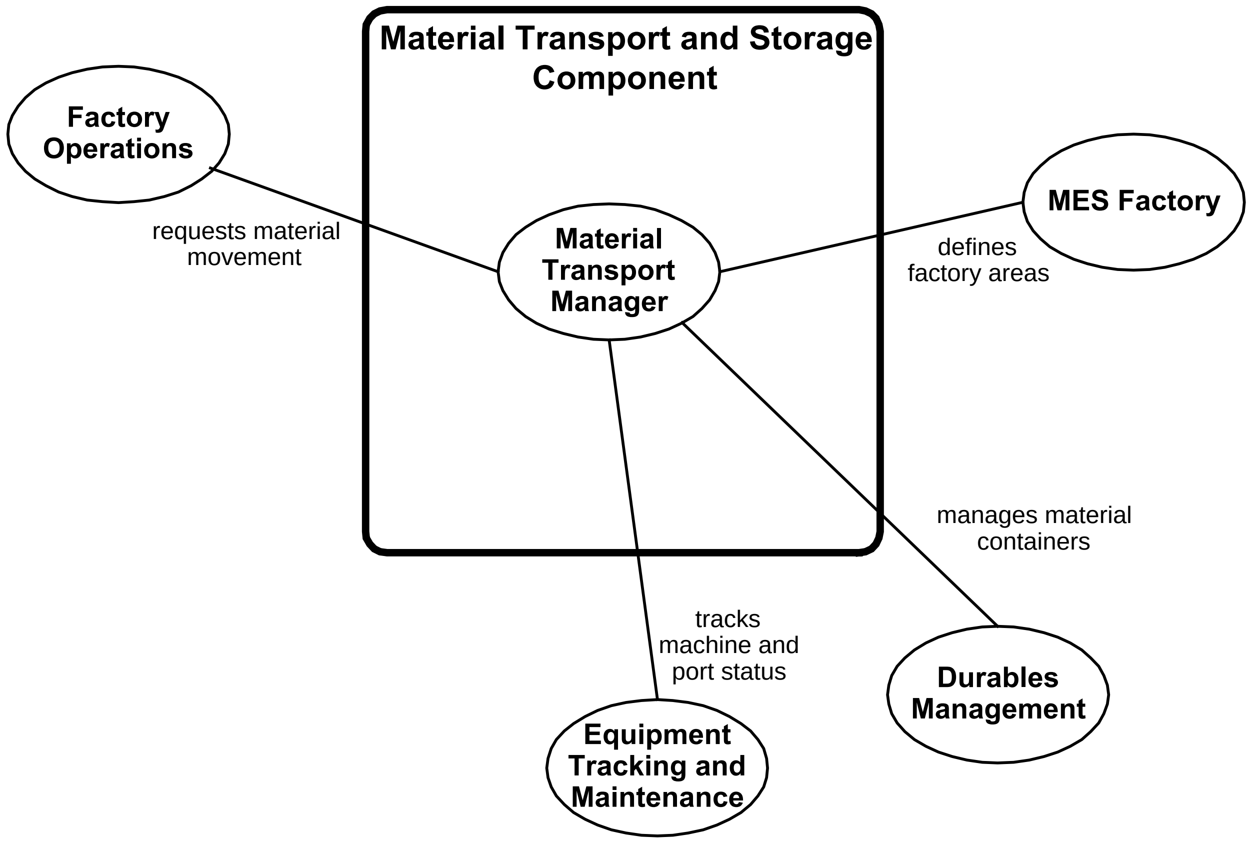
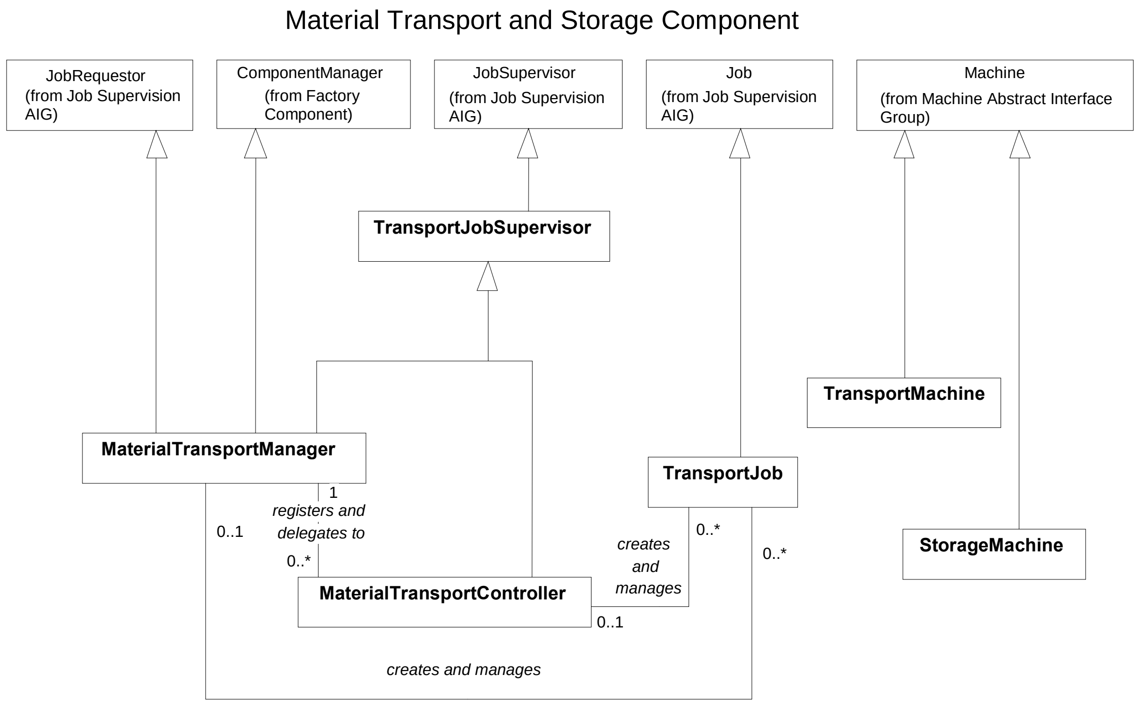

Material Transport and Storage Component
Factory Operations
requests material movement
Material Transport Manager
MES Factory
defines factory areas
manages material containers
tracks machine and port status
Durables Management
Equipment Tracking and Maintenance

Figure 2
Material Transport and Storage Component Context



Material Transport and Storage Component
JobRequestor (from Job Supervision AIG)
ComponentManager (from Factory Component)
JobSupervisor (from Job Supervision AIG)
Job (from Job Supervision AIG)
Machine (from Machine Abstract Interface Group)
TransportJobSupervisor
TransportMachine
MaterialTransportManager
TransportJob
StorageMachine
MaterialTransportController
1
registers and delegates to
0..1
0..*
creates and manages
0..*
0..*
0..1
creates and manages

Figure 3
Material Transport and Storage Information Model

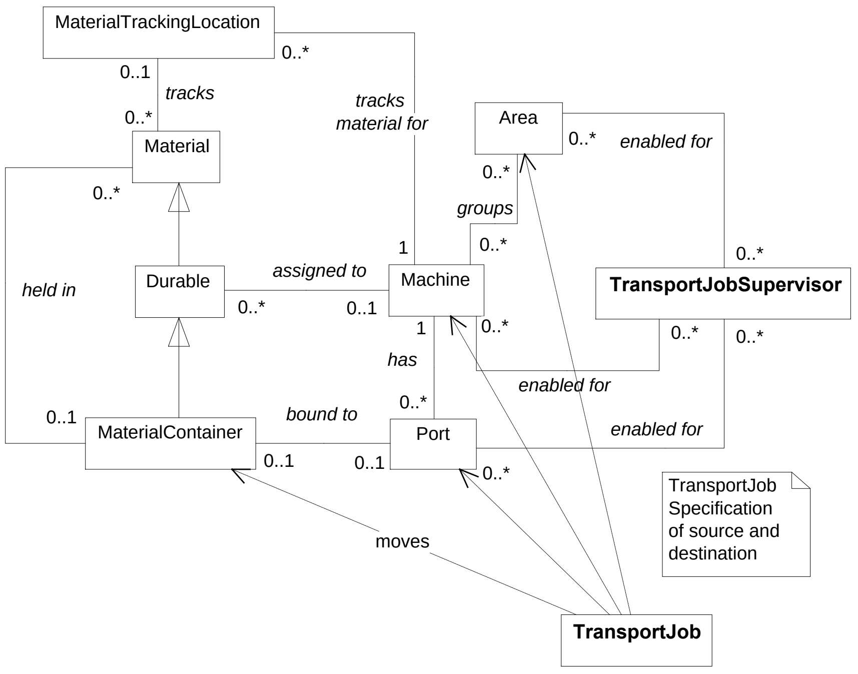


Figure 4
Material Transport and Storage Dependencies

6.2.3 *Durables Management Component (subset)* — This section includes only those interfaces and specific operations from the Durables Management Component that are referenced within this component specification. These interfaces are provided here for the referential integrity of this specification and are not included as part of this component. Further, this is not intended to be a comprehensive treatment of the required interfaces, but only the subset needed here for reference prior to the full specification of Durables Management being adopted through a subsequent ballot. When that occurs, this specification can be updated by a follow-on ballot to reference the interfaces directly rather than replicate them here.

6.2.3.1 *Durable Interface (subset)*

Module:	DurablesManagement
Interface:	Durable
Inherited Interface:	Material

```
interface Durable : AbstractIF::Material {
```

Description: The Durable interface represents any Material in the Factory used to facilitate manufacturing but not normally consumed in the process. It is capable of relocation within the Factory and requires dynamic tracking. This includes containers used to transport the Material, fixtures (attachments for holding material in a fixed position), and tools (such as reticles, load boards, workholders) used by equipment or personnel in the manufacturing process. Grouping of Durables (such as all reticles that are usable for a given process) can be achieved by creating MaterialGroups for the specific categorizations required.

Exceptions:

Published Events:

/* Notifies subscribers of a change in the MaterialTrackingLocation that is tracking the Durable. */

DurableLocationChangedEvent

Provided Services:

/* Set and get the location of the durable. If the MaterialTrackingLocation is set to NULL, the durable is not in any MaterialTrackingLocation and may be under manual control within the factory. */

```
void setMaterialTrackingLocation (
      in MachineAIG::MaterialTrackingLocation
            aMaterialTrackingLocation)
      raises (Global::FrameworkErrorSignal,
            MachineAIG::MaterialTrackingLocation::MaterialTrackingLocationFullSignal);

MachineAIG::MaterialTrackingLocation getMaterialTrackingLocation ( )
      raises (Global::FrameworkErrorSignal);
```

/* Set and get the Unit that determines which MaterialTrackingLocation this Durable will be tracked through for a given Machine. The Unit is a string value that maps to the Unit attribute of the MaterialTrackingLocation. */

```
void setUnit(
      in string aUnit)
      raises (Global::FrameworkErrorSignal,
            MachineAIG:: MaterialTrackingLocation::InvalidUnitSignal);

string getUnit( )
      raises (Global::FrameworkErrorSignal);

}; // Durable
```

Dynamic Model: None

6.2.3.2 *MaterialContainer Interface (subset)*

Module:	DurablesManagement
Interface:	MaterialContainer
Inherited Interface:	Durable

```
interface MaterialContainer : Durable {
```

Description:	A MaterialContainer interface represents any receptacle for holding Material for transport, processing or storage. Examples of MaterialContainers are shipping boxes, tubes, standard mechanical interface (SMIF) pods, etc.
Exceptions:	None.

Published Events:

/* Notifies subscribers of a state change in the MaterialContainer. */

```
MaterialContainerStateChangeEvent
```

Provided Services:

/* The following operations are provided to trigger MaterialContainer state transitions. */

```
void makeManualControl ( )
     raises (Global::FrameworkErrorSignal,
          Global::InvalidStateTransitionSignal);

void makeStored ( )
     raises (Global::FrameworkErrorSignal,
          Global::InvalidStateTransitionSignal);

void makeInTransit ( )
     raises (Global::FrameworkErrorSignal,
          Global::InvalidStateTransitionSignal);

void makeProcessing ( )
     raises (Global::FrameworkErrorSignal,
          Global::InvalidStateTransitionSignal);

}; // MaterialContainer

typedef sequence <MaterialContainer> MaterialContainerSequence;
```

Dynamic Model:

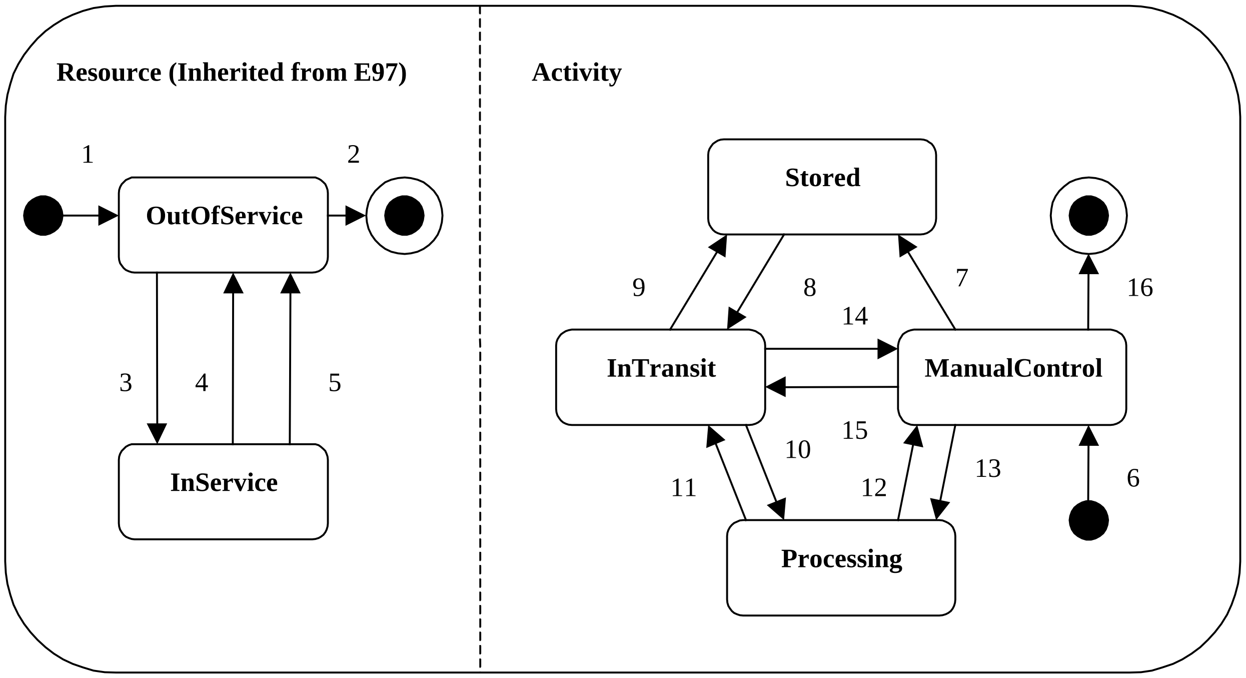


Figure 5
MaterialContainer Dynamic Model

Table 1 MaterialContainer State Definitions

State	*State Definition*	*Example*
InService	Inherited from Resource State Model	
OutOfService	Inherited from Resource State Model	
InTransit	Container is en route as the subject of a Transport Job.	Container is on an OHT transport vehicle.
Stored	Container is stored in a Storage Machine and not subject to a Transport Job.	Container is on a stocker shelf.
ManualControl	Container is outside of the control of the Material Transport and Storage Component.	Container is on a PGV.
Processing	Container is at process equipment.	Container is on a lithography tool.

Table 2 MaterialContainer State Transitions

#	*Current State*	*Trigger*	*Next State*	*Action*	*Comment*
1–5	NOTE: Transitions 1–5 are defined in the dynamic model inherited from Resource.				
6	N/A	makeManual Control()	ManualControl	MaterialContainer instance is created.	Initial state
7	ManualControl	makeStored()	Stored	DurableLocation ChangedEvent Published by the instance of MaterialContainer	Container is placed on input port of a Storage Machine without a TransportJob assigned to it.
8	Stored	makeInTransit()	InTransit	DurableLocation ChangedEvent Published by the instance of MaterialContainer	Container has started to move. TransportJob state transitions to Executing.
9	InTransit	makeStored()	Stored	DurableLocation ChangedEvent Published	Container stored. TransportJob state