SEMI E120-0704
SPECIFICATION FOR THE COMMON EQUIPMENT MODEL (CEM)

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 March 14, 2004 and April 22, 2004. Initially available at www.semi.org May 2004; to be published July 2004. Originally published March 2003, last published March 2004.

SUPERSEDED

1 Purpose

1.1 The purpose of this specification is to provide a generally applicable object model of semiconductor equipment structure. This object model is intended to be used in the following ways:

- As a guide to equipment suppliers on how to represent the external view of their equipment to the factory host.
- As a base model to be used and extended by other SEMI equipment communication standards.
- As a reference for the creation of technology-specific object model definitions, such as XML schema.

2 Scope

2.1 This specification provides a common set of constructs that can be used by the equipment manufacturer to model their equipment structure.

2.2 This specification defines the classes, attributes, relationships, and other detail appropriate for the model of an equipment's structure as viewed by a factory host application through the available communication interfaces.

2.3 The model contained in this specification documents features of the equipment structure considered necessary for more than one (current or future) equipment communication standards.

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

3 Limitations

3.1 This specification does not define "object diagrams" (that is, instances) of equipment. These are the responsibility of the equipment supplier to define in conformance with the model.

3.2 This specification does not define the behavior of objects/classes. Behavior definition of classes defined in this specification is left to other equipment communication standards.

3.3 This specification does not define any specific messaging between equipment and factory host. It is expected that communication related to the Common Equipment Model will be defined by other standards.

3.4 This specification addresses only the host view of the equipment. No other part of the factory is modeled by this document.

4 Referenced Standards

uuid: ISO/IEC 11578:1996 Information technology - Open Systems Interconnection — Remote Procedure Call (RPC), http://www.iso.ch/cate/d2229.htm.

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

5 Terminology

5.1 *Abbreviations and Acronyms*

5.1.1 *CEM* — Common Equipment Model

5.2 *Definitions*

5.2.1 UML terms used in this document conform to definitions in the Glossary of Terms of the OMG specification of UML (referenced above). Related Information, Section R1, contains selected definitions from that Glossary of Terms.

5.2.2 Definitions or descriptions of many of the terms used in this specification can be found in the SEMI Compilation of Terms, available on the SEMI web site, http://www.semi.org/.

5.2.3 When terms from the above two sources conflict, the definition from the OMG UML specification takes precedence in this specification.

6 Conventions

6.1 This section defines conventions followed in this document.

Documents Document Sharing Hub
Full document details: SEMI E120-0704

6.2 *Object Modeling*

6.2.1 *Unified Modeling Language (UML)* — This specification uses UML notation for all class diagrams and for object diagrams provided as examples. No other types of diagrams are used in this specification.

6.2.1.1 UML class diagrams have clearly defined meaning and are a part of this specification. Detail contained in these diagrams is not always repeated in the text.

6.2.2 *Behavior Not Defined* — This specification, defines classes, their relationships, and attributes. No behavior is specified either in the form of UML state, activity, sequence diagrams, or in the form of operations definitions.

6.2.3 *Name of a Class* — The text capitalizes class names.

6.2.4 *Abstract and Concrete Classes* — Each class is specified as Abstract or Concrete. Abstract classes are not directly implemented. In this specification "implemented" means represented to the factory through the communications interface. All classes defined as concrete may be directly implemented. This specification does not attempt to define equipment control system implementation. In UML class diagrams, abstract class names are shown in *italics*.

6.2.5 *Class Attribute Definition* — The attributes of a class are defined in table format as illustrated by Table 1 below. Note that the "+" sign in front of attributes in UML class diagrams indicates "public" attributes. All attributes defined in this specification are public.

Table 1 Attribute Table Format

Attribute Name	*Definition*	*Access*	*Reqd*	*Form*
		RO or RW	Y or N	See list below.

6.2.5.1 *Access* — Attributes may be settable (ReadWrite or RW) or not settable (ReadOnly or RO) through an interface to the equipment.

6.2.5.2 *Reqd* — Is this attribute required? Y – Yes, or N – No.

6.2.5.3 *Form* — Defines the data type of the attribute. Data types specified in the Common Equipment Model specification are described in the subsections that follow. Data types in this specification are high-level definitions and should be mapped to the data types of a specific technology as appropriate. Extensions to the Common Equipment Model are not limited to the list of data types defined in this specification. See the definition of "Form" in the SEMI Compilation of Terms for other Form values commonly used in SEMI documents.

6.2.5.3.1 *Enumeration* — A list of named values used as the range of a particular attribute type. In this document, these values are represented as text strings, but may be implemented differently (e.g. as integers that correspond to the named values).

6.2.5.3.2 *String* — A text string. Limitations on length are left to the implementation technology unless otherwise specified in this document.

6.2.5.3.3 *List of xxx* — An item that can hold multiple instances of a specified type (where xxx is the data type).

6.2.6 *Constraints*

6.2.6.1 Constraints in UML are shown with dotted lines, typically with a note attached to explain the constraint. Constraints shown in this way tend to clutter large UML diagrams. Therefore, in this specification, constraints are shown only on the diagrams focusing on individual classes.

6.2.7 *Association Navigability*

6.2.7.1 UML associations include the concept of "Navigability". When an association is navigable, the association may be traversed to reach the class instance at the opposite end of the association (target object).

6.2.7.2 By default, an association is navigable in both directions. If an arrowhead is shown on one end of an association, then navigability exists only in that direction.

SUPERSEDED

Documents
Document Sharing Hub
Full document details: SEMI E120-0704

6.2.7.3 From any object, it must be possible to obtain a reference to the target objects of any of its navigable associations.

6.2.7.4 The method of referencing the target objects is implementation dependent and is not specified in this document.

6.2.7.5 Any implementer or subordinate standards mapping the concepts of this specification to an implementation method shall specify the means of referencing the target object of an association from the source object.

6.3 *Model Extension*

6.3.1 Any concrete class in the CEM model can be extended. Extension is achieved through association. In "extension by association" the new capability is contained in a separate class that is related to the original by an association. In the case of CEM, an "*Extension*" class has been added to the model (see Section 8.5.3.4) along with the needed association.

6.3.2 The *Extension* class is associated with *Nameable*, so that any *Nameable* can have multiple *Extensions*. The *Extension* class has no attributes or services and does not itself add anything to the extended CEM class. To have a meaningful extension, a subclass of *Extension* must be created containing the new features.

6.3.3 So, in order to create an extension to a selected concrete CEM class, a new class would be designed with added features. For instance, it might contain a new attribute. This new class would be designed as a subclass of *Extension*. Remember that any *Nameable* can navigate the association to any of its *Extension*s. Therefore, the new subclass of *Extension* could be associated with any CEM class. See Figure 1 below for an illustration. The final step is to associate the new *Extension* subclass to the specific CEM class it is extending.

6.3.4 Most extensions will be designed to work on specific CEM classes. The CEM model does not enforce matching an extension with its intended CEM class. Therefore, the specification of the *Extension* class must describe its allowed usage.

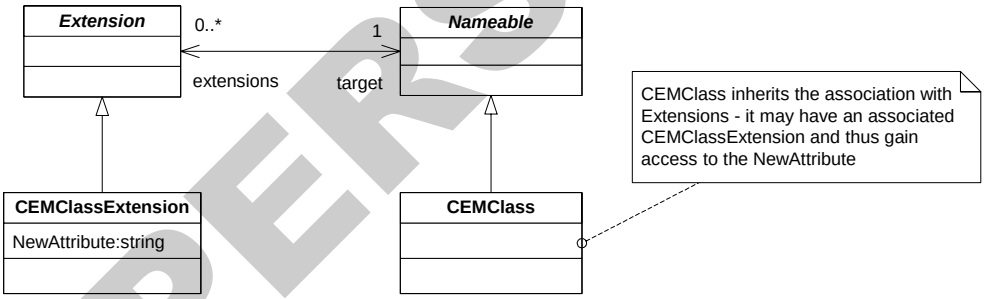

Figure 1
CEM Model Extension

6.3.5 The scope of CEM is limited to static information about the equipment. However, that limitation does not apply to extensions. So extensions might include static or dynamic attributes as well as links to state models or other complex constructs. An example of an extension that might be added by a supplier is a class containing "manufacturedDate" and "serviceTagNumber" attributes to be applied to an *Equipment* object.

7 Background

7.1 This specification documents common characteristics of equipment in a single specification that can be referenced and used by other standards. These other standards are expected to extend the Common Equipment Model to add characteristics unique to that standard. Thus, this specification attempts to be the core of a single, coordinated object model of the equipment with the full model being a composite of multiple standards.

SUPERSEDED

Documents Document Sharing Hub
Full document details: SEMI E120-0704

7.2 *Approach to Model Description*

7.2.1 *Class Model vs. Instance Model*

7.2.1.1 A model showing the specifics about a particular equipment is considered an "instance model". Each equipment instance model will have its own characteristics based on its function. No single model can directly describe all such instances. Instead, the class model contained in this specification defines the classes to be used to describe the equipment's structure and how they shall relate to one another. The instance model of the equipment is to be provided by the equipment supplier as specified by the CEM class model.

7.2.2 *Model Extensibility*

7.2.2.1 The Common Equipment Model is intended to be extensible (within defined limits). Section 6.3 describes acceptable ways to extend the model.

7.2.2.2 The instance model of the equipment shall thus be constructed by the equipment supplier based on

- the classes defined in this document,
- extensions to this model defined in other SEMI standards, and
- extensions to the standardized model made by the supplier to address unique equipment characteristics.

7.2.3 *External View — Not Equipment Design*

7.2.3.1 There is no expectation that equipment suppliers will use the model specified in this document as the basis for the design of their equipment. The CEM is intended to provide a structure to promote clear communication. It needs to provide an unambiguous view to the factory host that is representative of the equipment and meets the needs of the host.

7.2.4 *Communication Not Defined*

7.2.4.1 This specification leaves to other SEMI standards the definition of how the Common Equipment Model will be communicated (in whole or in part) between the equipment and the factory host. It is to be expected that there will be many uses for the model and the details of communication are best defined in context with planned use.

7.2.5 *Behavior Not Defined*

7.2.5.1 In order to keep the model simple, this document does not define behavior of classes in the CEM. Class behavior is related to function, and thus to functionality supplied by the equipment interface. This functionality and behavior is left to other SEMI standards to define.

7.2.6 *Applicability*

7.2.6.1 Though the CEM attempts to address all semiconductor related factory equipment, it is recognized that this may not be possible. The CEM covers only manufacturing equipment (i.e. equipment that perform their function on material—e.g. no independent environmental monitoring systems, etc.). The vast majority of semiconductor factory equipment should be able to comply with this specification.

7.3 *Uses For The CEM*

7.3.1 In order to understand the CEM, it is necessary to understand expected uses of the model. While it is not possible to anticipate every possible use, the following list presents several expected uses that may be defined by other standards.

7.3.2 *Equipment Self-Description Using Metadata* — An important consideration for the integration of semiconductor equipment into the factory is the ease with which the equipment's structure and capabilities can be understood and translated into functioning software. With traditional semiconductor communication technologies this is achieved by first reading the equipment supplier's communication manual and then configuring or coding factory software to match the documentation. Unfortunately, manuals are too often incomplete or out of date. If the equipment can provide its instance model (and the extended class model on which it is based) to the host, it is more likely to be complete and accurate. If it can construct that model in real-time, based on its current hardware configuration it will be more accurate. This self-descriptive data is also known as "metadata".

7.3.3 *Relation Of Data To The Equipment* — Data gathered from equipment gains greater meaning if it is clear as to what part of the equipment it pertains and what related data items provide context information. The instance model can be a reference in the definition of the data collection plans and reflected by actual data collected.

7.3.4 *Relation Of Parameters & Setting To The Equipment* — Proper and predictable control of equipment is possible only if the effects of all available parameters and settings is understood. These must be related to the equipment and to the measurable values that they affect. This information can be provided by the instance model.

7.3.5 *Understanding Equipment Capabilities* — The process capabilities of equipment depend on the modules present and the characteristics and status of those modules. In some situations, the hardware configuration of equipment can change on a regular basis.

SUPERSEDED

Documents
Document Sharing Hub
Full document details: SEMI E120-0704

8 Common Equipment Model Definition

8.1 This section documents the Common Equipment Model.

8.2 *Overview*

8.2.1 Documentation of the Common Equipment Model begins with system views of the model and moves to detailed descriptions of the classes with their attributes and relationships. No behavior is defined—that is, no state models and no methods are described for classes.

8.2.2 It is important to note that the two system views presented are of the same model. They are separated to avoid large, complex diagrams, thus enhancing understandability.

8.2.3 The model shows only classes and their relationships. Actual equipment will have corresponding instance models that define the detail of the equipment as viewed through an equipment interface. Thus, the Common Equipment Model in this document provides rules for constructing the instance model of an equipment.

8.2.4 The Common Equipment Model is represented in two system level views. The first is the inheritance hierarchy that shows which classes inherit characteristics (e.g. attributes and relationships) from which other classes. The second view represents the relationships among classes to show general equipment structure. These views are described in general terms to highlight important aspects of the model.

8.2.5 Everything in these two views is repeated in subsequent sections dealing with individual classes in the model. In the individual class views, class of interest is shown with all relationships to other classes. No other detail is shown in these views.

8.3 *Inheritance Hierarchy*

8.3.1 The inheritance hierarchy view, Figure 2, shows the superclass/subclass relationships of the Common Equipment Model.

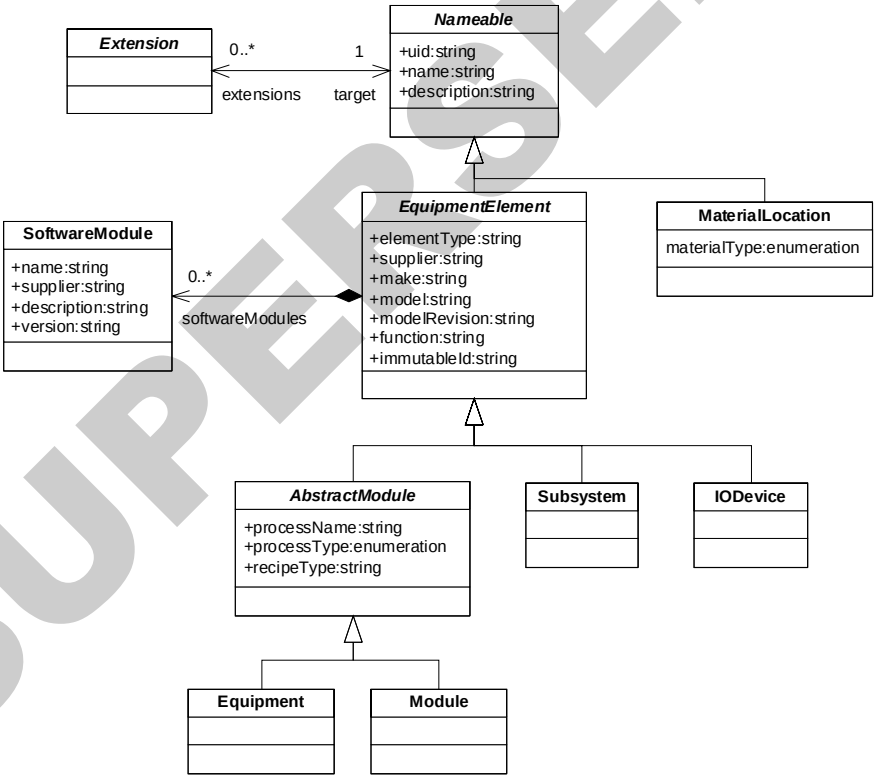

Figure 2
Inheritance Hierarchy

SUPERSEDED

Documents Document Sharing Hub
Full document details: SEMI E120-0704

8.3.2 Items to note about the inheritance hierarchy view:

- Nameable is at the top of the inheritance hierarchy and provides common accessibility to all classes in the Common Equipment Model.
- *Equipment* and *Modules* may be able to process material, but *Subsystems* cannot.
- Note that a subclass inherits the attributes of its superclass. Some subclasses have no additional attributes defined in this specification. It is anticipated that other standards will extend this model to add attributes, relationships, state models, etc. as equipment behavior is mapped onto this model.

8.4 *Relationship View*

8.4.1 The relationship view is shown in Figure 3. This view illustrates expected equipment. Items to note about the relationship view:

- Flexibility is built into the structure so that, for instance, *IODevices* may belong directly to almost any level of the equipment structure.
- *Subsystems*, *Modules*, and *Equipment* may have MaterialLocations (i.e. hold material).
- *Equipment* — *Equipment* aggregation allows an *Equipment* that acts on behalf of other *Equipment* to describe these *Equipment*.

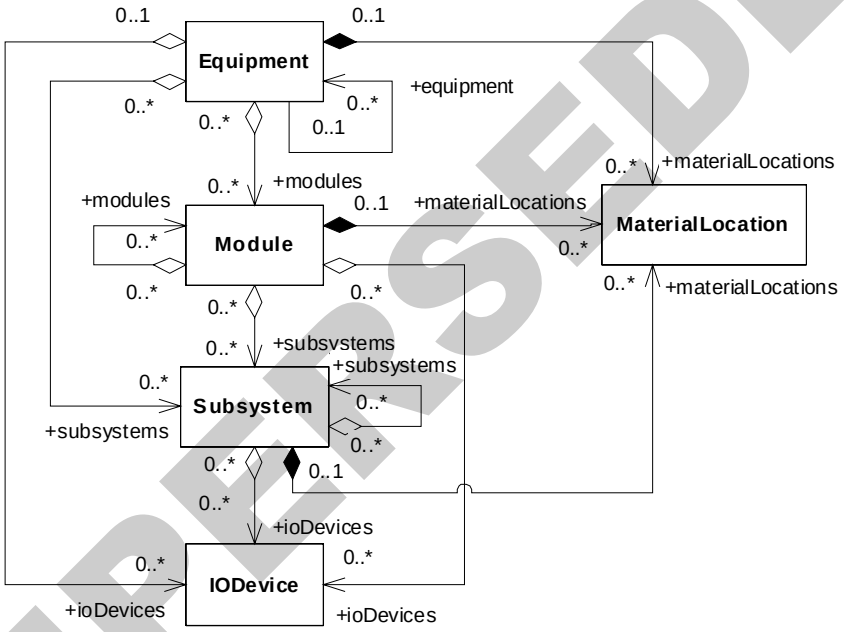

Figure 3
Relationship View

SUPERSEDED

Documents
Document Sharing Hub
Full document details: SEMI E120-0704

8.5 *Class Definitions*

8.5.1 Class definitions consist of a brief description of the highlighted class, a class diagram showing the class and all related classes (both from an inheritance and relationship standpoint), a table defining the attributes, and further text as necessary to fully define the class.

8.5.2 For clarity, the abstract classes are listed separately from concrete classes.

8.5.3 *Abstract Classes*

8.5.3.1 *Nameable* — The root of each class in the CEM class hierarchy. This class provides a means for uniquely identifying each component of the equipment structure. *Nameable* is a superclass of *EquipmentElement* and *MaterialLocation* (see Figure 4). It includes three attributes: *uid, name, and description* (see Table 2). *Nameable* is also associated with the class *Extension*. See Section 6.3 for more information on the use of *Extension*.

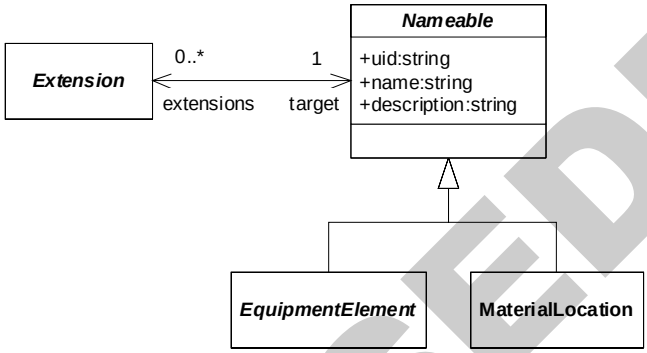

Figure 4
Nameable Class

Table 2 Attributes of *Nameable*

Attribute Name	*Description*	*Access*	*Reqd*	*Form*
uid	Identifier that uniquely and permanently identifies a *Nameable*. The *uid* contains a "uuid" string. See Section 4, ISO/IEC 11578:1996 for a reference to uuid definition. A properly formed uuid provides an identifier that is unique from any other uuid (in the factory and beyond).	RO	Y	String
name	Human-readable name for *Nameable*. The *name* is representative of the role or function that the Nameable plays in the equipment structure.	RO	Y	String – See rules for naming below.
description	Human-readable description of *Nameable*. Contents of this attribute shall provide enough information for a human to understand the purpose of this *Nameable*.	RO	Y	String

8.5.3.1.1 *uid* — The requirements for each *uid* value are:

- The *uid* value shall conform to the rules for constructing a uuid as defined in standard ISO/IEC 11578:1996.
- The *uid* shall be a fixed value with respect to the physical hardware and shall be unique among all *Nameables*. The *uid* shall remain with the *Nameable*, even if the *Nameable* is removed from one equipment and added to another. For example, if an *IODevice* fails and is removed from an equipment, its replacement will have a different *uid*. If the original *IODevice* is repaired and installed on a different equipment, it will retain its original *uid*. Note that a *MaterialLocation* is contained within an equipment component and will move with that component, retaining its own *uid*.

8.5.3.1.2 *name* — The naming requirements for each *Nameable* are:

- *Names* shall consist of alphanumeric characters, spaces, hyphens, and underscores. Names shall start with an alpha character, and shall not contain leading or trailing spaces.

SUPERSEDED

Documents Document Sharing Hub
Full document details: SEMI E120-0704