SEMI E133-0212
SPECIFICATION FOR AUTOMATED PROCESS CONTROL SYSTEMS INTERFACE

This Standard was technically approved by the global Information & Control Technical Committee. This edition was approved for publication by the global Audits and Reviews Subcommittee on December 24, 2011. Available at www.semiviews.org and www.semi.org in February 2012; originally published March 2004; previously published November 2010.

SUPERSEDED

1 Purpose

1.1 In order to more efficiently facilitate the integration of process control systems (PCS) such as run-to-run (R2R) control, fault detection (FD), fault classification (FC), fault prediction (FP), statistical process control (SPC), etc. into current and future fabs, there is a need to define interfaces for PCS that enable them to interact effectively and share data (1) among themselves and (2) with the other interdependent factory systems (including equipment data collection). This Standard is intended for use by equipment suppliers, semiconductor manufacturers, equipment subsystem suppliers and control system suppliers who will utilize the PCS standards in future developed products.

2 Scope

2.1 This Standard focuses on defining PCS capabilities, functional groups (FCs), and functional group (FC) interfaces. The Standard is structured so that new FCs can be added to the Standard through future balloting. The components in this document include:

- Definitions of key terms and concepts pertinent to this domain,
- Description of the specification conventions used,
- Identification of the major FGs within the scope of PCS,
- Description of the capabilities expected in each FG (required and optional), and
- Definition of interfaces for these FGs, including:
  - Data that is available across these interfaces. This includes the interface data descriptions (inputs and outputs). Future enhancements to this Specification will include detailed data structures in the form of inheritance and aggregation models that are common to all PCS FGs, as well as specific models for each,
  - Services supported and Interface behavior assumed, and
  - Description of an approach for compliance verification testing.

2.2 The specific format and protocol implementation of these interface specifications, which are technology specific, will be delineated in supporting 'dot' specifications to this Standard. As an example a possible format is eXtensible markup language (XML) (which would include data type definitions and metamodel schemas).

2.3 In addition to the information embodied in this Standard, the Standard describes a validation process to identify and address real implementation issues as early as possible.

NOTICE: SEMI Standards and Safety Guidelines do not purport to address all safety issues associated with their use. It is the responsibility of the users of the Documents to establish appropriate safety and health practices, and determine the applicability of regulatory or other limitations prior to use.

3 Limitations

3.1 The proposed PCS standards will **not** focus on:

- Low-level communication mechanisms (e.g., hardware connection schemes, transport layer software, etc.) between PCS FGs,
- Methodology for implementing actual equipment or factory level control strategies,
- Specifying internal structure or implementation methods of PCS FGs,

Documents Document Sharing Hub
Full document details: SEMI E133-0212

- Capabilities of complementary applications (recipe management, data collection, engineering analysis, data storage, scheduling, change control, …),
- The management of information and collaboration between FGs. This is left to implementation,
- Communications within equipment or within factory systems,
- Determination of the proper control algorithm, model, or runtime selectable behavior. It is optional for a PCS function to contain 'context matching' (CM),
- The creation and management of PCS jobs, or
- Defining test procedures required for validating the Standard.

4 Referenced Standards and Documents

4.1 *SEMI Standards and Safety Guidelines*

SEMI E81 — Provisional Specification for CIM Framework Domain Architecture

SEMI E121 — Guide for Style and Usage of XML for Semiconductor Manufacturing Applications

SEMI E125 — Specification for Equipment Self Description (ESD)

SEMI E134 — Specification for Data Collection Management

4.2 *IEC Standards*[1]

IEC 61158-5 — Digital Data Communications for Measurement and Control – Fieldbus for Use in Industrial Control Systems – Part 5: Application Layer Service Definition

4.3 *ISO Standards*[2]

ISO 8601 — Data Elements and Interchange Formats – Information Interchange – Representation of Dates and Times

4.4 *Other Documents*

SEMATECH — Equipment Engineering Capabilities (EEC) Guidelines (Phase 2.0), Version 2.5 (July 2002)[3]

Rumbaugh, James, Jacobson, Ivar, and Booch, Grady "The Unified Modeling Language Reference Manual." Addison Wesley Professional, 1999.

Booch, Grady, Rumbaugh, James, and Jacobson, Ivar "The Unified Modeling Language User Guide." Addison Wesley Professional, 1999.

Rumbaugh, James, *et al.* "Object-Oriented Modeling and Design." Prentice Hall, Englewood Cliffs, NJ (1991).

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

5 Terminology

5.1 *Abbreviations and Acronyms*

5.1.1 *AE* — analysis engine

5.1.2 *APC* — advanced process control

5.1.3 *CM* — context matching

5.1.4 *FC* — fault classification

5.1.5 *FD* — fault detection

[1] International Electrotechnical Commission, 3 rue de Varembé, Case Postale 131, CH-1211 Geneva 20, Switzerland; Telephone: 41.22.919.02.11, Fax: 41.22.919.03.00, http://www.iec.ch

[2] International Organization for Standardization, ISO Central Secretariat, 1 rue de Varembé, Case postale 56, CH-1211 Geneva 20, Switzerland; Telephone: 41.22.749.01.11, Fax: 41.22.733.34.30, http://www.iso.ch

[3] SEMATECH, 257 Fuller Road, Suite 2200, Albany, NY 12203, USA; Telephone: 518.649.1000, http://www.sematech.org

SUPERSEDED

Documents
Document Sharing Hub
Full document details: SEMI E133-0212

5.1.6 *FDC* — fault detection and classification

5.1.7 *FG* — functional group

5.1.8 *FP* — fault prediction (or prognosis)

5.1.9 *PCS* — process control system

5.1.10 *PCS job* — process control system job

5.1.11 *R2R* — run-to-run

5.1.12 *SPC* — statistical process control

5.1.13 *W2W* — wafer-to-wafer

5.1.14 *XML* — eXtensible markup language

5.2 *Definitions*

5.2.1 *advanced process control (APC)*[4] — the manufacturing discipline for applying control strategies and/or employing analysis and computation mechanisms to recommend optimized machine settings and detect faults and determine their cause.

5.2.2 *analysis engine (AE)* — a process that utilizes data and possibly operational instructions to produce a response. In the context of this Standard, this term is used to encompass the characteristics common to all PCS functional groups.

5.2.3 *context* — a series of attributes that uniquely identifies a manufacturing entity (e.g., wafer, lot, module, tool, reticle) and its status in the manufacturing operation.

5.2.4 *context matching (CM)* — the process of comparing and matching the values of a set of attributes that represent the state of a system (e.g., process, product and equipment) to a set of stored or computed values. This is usually done so that a unique action can be specified by the CM system.

5.2.5 *fault classification (FC)* — the technique of determining the cause of a fault once it has been detected.

5.2.6 *fault detection (FD)* — the technique of monitoring and analyzing variations in tool and/or process data to detect anomalies. FD includes both univariate and multivariate statistical analysis techniques.

5.2.7 *fault detection and classification (FDC)* — combination of FD and FC.

5.2.8 *fault prediction (or prognosis) (FP)* — the technique of monitoring and analyzing variations in process data to predict anomalies.

5.2.9 *functional group (FG)* — a collection of closely related software capabilities that one would expect to be provided as an integrated product.

5.2.10 *process control system (PCS)*[3] — a system capable of performing process control, which includes one or more of R2R control, FD, FC, FP, SPC, or any future process control functionality defined in this Standard for a PCS functional group.

5.2.11 *process control system job (PCS job)* — a unit of work that could be tracked and have data associated with it. In the context of this Standard a PCS job is associated with a unit of work that a PCS AE performs; a unique identifier such as an ID number is often utilized to track the PCS job.

5.2.12 *run-to-run (R2R) control* — the technique of modifying recipe parameters or the selection of control parameters between runs to improve processing performance. A 'run' can be a batch, lot, or an individual wafer.

5.2.13 *statistical process control (SPC)* — the technique of using statistical methods to analyze process or product metrics to take appropriate actions to achieve and maintain a state of statistical control and continuously improve the process capability.

[4] APC and PCS terms are used interchangeably in some companies. Both can be seen as the umbrella of components for process control.

SUPERSEDED
Documents Document Sharing Hub
Full document details: SEMI E133-0212

5.2.14 *wafer-to-wafer (W2W) control* — a form of R2R control in which a 'run' is the processing of a single wafer. This is differentiated from 'wafer level control' in which the recipe may be determined for multiple individual wafers before processing begins, but not necessarily modified from wafer to wafer; for example, between wafers.

6 Background

6.1 With the development of PCS applications at the factory level and inside the equipment, it is now apparent that the lack of interface standards for these applications has inhibited widespread adoption of process control at the factory level. Moreover, adding process control and integrated metrology capabilities inside the tool requires the ability for the factory level applications to collaborate with tool level applications.

7 Conventions

7.1 This section defines conventions followed in this Document.

7.2 *Object Modeling*

7.2.1 *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.

7.2.2 UML class diagrams have clearly defined meanings and are a part of this Specification. Detail contained in these diagrams is not always repeated in the text.

7.2.3 *Class Diagrams* — UML Class Charts.

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

7.2.5 *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.

7.2.6 *Class Attribute Definition* — The attributes of a class are defined in table format as illustrated in the table 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, RW, or NA	Y or N	See list below.

7.2.6.1 *Access* — Attributes may be settable (ReadWrite or RW) or not settable (ReadOnly or RO) through the interface services. If the attribute refers to a component of a message (e.g., that is communicated to/from a PCS analysis engine [AE]), the concept of access does not apply and 'Not Applicable' (NA) is used in the Access field.

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

7.2.6.3 *Form* — Defines the data type of the attribute. Data types specified in the PCS 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 PCS interface specifications are not limited to the list of data types defined in this Specification. See the definition of 'form' in the *SEMI International Standards Compilation of Terms* for other form values commonly used in SEMI Documents.

7.2.6.4 *Constraints*

7.2.6.4.1 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.

SUPERSEDED

Documents
Document Sharing Hub
Full document details: SEMI E133-0212

7.2.7 *UML Associations*

7.2.7.1 The mechanism used for representing associations in UML between classes is implementation dependent. This Document is abstract in nature, and does not specify or imply any such mechanism. Any adjunct standard that provides an implementation of this Specification must include a description of the mechanism used for representing UML associations used in this Document.

7.2.8 *Class Association Table* — This table provides an example of the tables used to list and describe associations between classes defined in this Specification.

Table 2 Association Table Format

Association Role Name	*Role Definition*	*Comments*

7.2.8.1 *Association Role Name* — The name of the association role being defined.

7.2.8.2 *Role Definition* — Describes the function or purpose of the association.

7.2.8.3 *Comments* — Any additional comments or notes regarding the association.

7.3 *Data Types*

7.3.1 The following data types may be used in this Specification to describe the form of an attribute or other similar high level definitions. Note that extensions to the PCS interface specifications are not limited to the list of data types defined in this Specification.

- *Any* — The format may be any of the others listed in this section. Format is determined by the implementation.
- *Binary* — A sequence of bytes that may have any value. Binary values are sometimes called 'unformatted' because their structure is not apparent.
- *Boolean* — Takes the value of 'true' or 'false.'
- *Checksum* — A checksum value calculated on a specific stream of data using a specific method of calculation.
- *Enumeration* — A format that allows a specified list of possible values. 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).
- *Status* — The Status type is a structure that contains information about whether or not an error that has occurred in a transmission and information on the type of error. The form of the structure is given in § 9.
- *Integer* — A numeric value, Integers are always whole numbers. The form and number of bytes is left to the implementation definition.
- *List of xxx* — An item that can hold multiple instances of a specified type (where xxx is the data type).
- *Real* — A numeric value that may represent any whole or fractional number. The form and number of bytes is left to the implementation definition.
- *Text* — A text string. Limitations on length are left to the implementation technology unless otherwise specified in this document. Strings are recommended to be implemented as UTF-8.
- *Time* — Representation of the date and time of the occurrence of interest. The structure and form of items of this type is left to the implementation.

7.4 *Service Message Representation*

7.4.1 *Service Resource Definition*

7.4.1.1 The service resource definition table defines the specific set of messages for a given service group as shown in the following table:

SUPERSEDED

Documents Document Sharing Hub
Full document details: SEMI E133-0212

Service	*Reqd*	*Type*	*Description*
Service name	Y/N	N or R	The intent of the service.

7.4.1.2 Reqd can be Yes (Y) or No (N). Type can be either Notification (N) or Request (R). A notification service consists of messages initiated by the service provider. No response is expected. A request service consists of a request message initiated by the service user which generally asks for data or for an operation to be performed, and a related response message returned by the service provider.

7.4.2 *Service Parameter Definition*

7.4.2.1 The service parameter definition table defines parameter detail for each service in which parameters are explicitly specified, as shown in the following table:

Parameter	*Request*	*Response*	*Form*	*Description*
Service parameter name	(see below)	(see below)	Data type	The intent of the parameter.

7.4.2.2 The Request and Response entries can take on the following values:

M	Mandatory Parameter – Must be given a valid value.
C	Conditional Parameter – May be defined in some circumstances and undefined in others. Whether a value is given may be completely optional or may depend on the value of the other parameter.
-	The parameter is not used.
=	The entries M and C in the response can be modified with (=) to indicate that the value in the response must match the request.

7.5 *Behavior Representation*

7.5.1 *Sequence Behavior* — Where appropriate, this document uses the Harel State Chart notation to describe the dynamic behavior of the AEs. An overview of this notation is presented in an Appendix of SEMI E30. The formal definition of this notation is presented in Science of Computer Programming 8, *Statecharts: A Visual Formalism for Complex Systems*, by D. Harel, 1987.

7.5.2 *State Behavior* — Where appropriate, transition tables are provided in conjunction with the state diagrams to explicitly describe the nature of each state transition. Each transition table contains columns for transition #, 'Previous State,' 'Trigger,' 'New State,' 'Actions,' and 'Comments.' The trigger for the action occurs while in the "Previous State." The "Actions" include a combination of: (1) actions taken upon exit of the 'Previous State,' (2) Actions taken upon entry to the 'New State,' and (3) actions taken which are most closely associated with the transition. No differentiation is made.

8 Overview

8.1 Using the simple definition of a FG given in the terminology section, the initial set of PCS FGs has been identified. To identify the proper scope (and therefore the boundaries) for these groups, the following criteria are applied to test whether or not a given set of capabilities constitutes a valid PCS FG. Specifically, a FG should satisfy one or more of the following requirements:

- Support a coherent set of tasks
- Share common set of data and must use it under similar constraints (performance, retention, etc.)
- Internal objects are tightly coupled

SUPERSEDED

Documents
Document Sharing Hub
Full document details: SEMI E133-0212

- Legacy implementation/integration constraints exist (e.g., the capabilities are inseparable because an external system must interact with them as an integrated whole)
- Strong associations exist in the market (i.e., one would expect these capabilities to be provided as an integrated product)

8.2 Based on these criteria, and the stated implementation priorities of the semiconductor manufacturers and suppliers involved on the task force, the following initial PCS FGs have been identified:

- R2R control
- FD
- FC
- FP
- SPC

8.3 R2R control includes W2W control. In this case a 'run' is the processing of a single wafer.

8.4 Note that it is understood that other PCS FGs can be added as needed. Future versions of the document will describe procedures for adding a new group.

8.5 The aforementioned groups can be implemented at the tool or factory level or both. As a requirement, FG interfaces should be defined in a consistent way. Figure 1 represents the proposed FGs and an example of their interactions within the factory and tool environments.

8.5.1 PCSs may consist of a variety of capabilities or functions at both the factory and equipment level as shown in Figure 1. Moreover, this may include multiple instances of the same function type from different suppliers. In the example presented in Figure 1, several R2R controllers are running at the factory level with differences that are only visible as internal capabilities; these factory-level controllers could be providing R2R control at a lot-to-lot level of granularity, while the equipment level R2R controller could be providing W2W control, though any R2R control capability—including W2W control—can be provided inside or outside of the equipment as shown in Figure 1. This interface standard will facilitate easier collaboration between the tool level functions and factory level systems independent of the supplier that developed the internal function's capabilities.

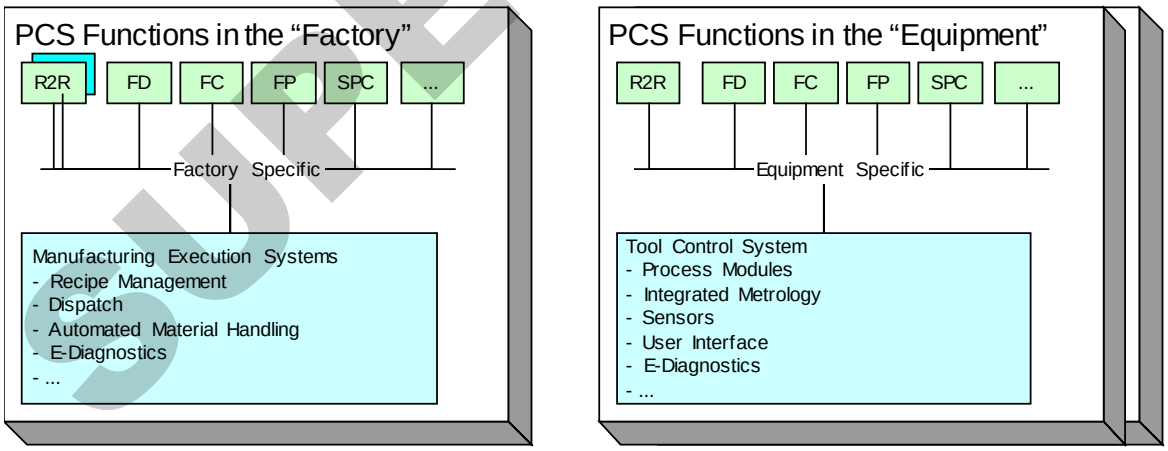

Figure 1
PCS Functions

SUPERSEDED

Documents Document Sharing Hub
Full document details: SEMI E133-0212

9 PCS FG Capabilities and Interface Data Structures

9.1 *FG Partitioning*

9.1.1 PCS are partitioned into FGs using the criteria defined in § 8. This Standard defines the interface data structures, services and behavior associated with these FGs. In this section the capabilities of the PCS FGs are defined in more detail. First of all, the basic capabilities common to all FGs are presented. Then the additional basic capabilities unique to each FG are presented. A high level description of the specifications of data structures, services and behavior is provided in this section.

9.1.2 It is important to note that the specific implementations of these various FGs may differ greatly in the level of capability delivered, performance, quality of service, and internal architecture. Moreover, in a given factory, there may be multiple instances of the same FG running concurrently, from one or more suppliers. Some FGs may be completely independent of other PCS FGs (in other words, they can operate in relative isolation, depending only on information from systems external to PCS); others may depend heavily on other FGs, collaborating closely to provide a higher level of capability than they could on their own.

9.1.3 The goal of the PCS interface standards is to enable all of these implementation scenarios (and more) without dictating internal product architectures.

9.2 *PCS AE*

9.2.1 As defined in § 8, all FGs can be thought of as specializations of a PCS AE and therefore inherit the structure and behavior of a PCS AE. This section defines the common capabilities of a PCS AE.

9.2.2 *General Capability Description*

9.2.2.1 All PCS AEs shall have inputs and outputs as shown in Figure 2. This means that the PCS AEs shall support the consumption of the specified inputs and production of the specified outputs as necessary to achieve the required capabilities of that analysis engine as specified in this Standard. These inputs and outputs are specified as aggregate data structures. The structures are aggregated into 'collection types' according to the category of data. A definition of these collection types is provided in Table 4. The PCS AE shall have the general behavior of utilizing specified inputs, performing tasks based to some degree on these inputs, and generating outputs. Any additional structure and behavior of various PCS AEs are specified in the reminder of this section.

9.2.2.2 Note that, in addition to limitations of specification defined in § 4, the following are beyond the scope of definition of the PCS AE:

- The level of synchronization of inputs with performance of tasks and/or generation of outputs, and
- The specification of any internal states of the PCS AE.

9.2.2.3 All PCS AEs belonging to this FG shall have the capabilities listed in Table 5.

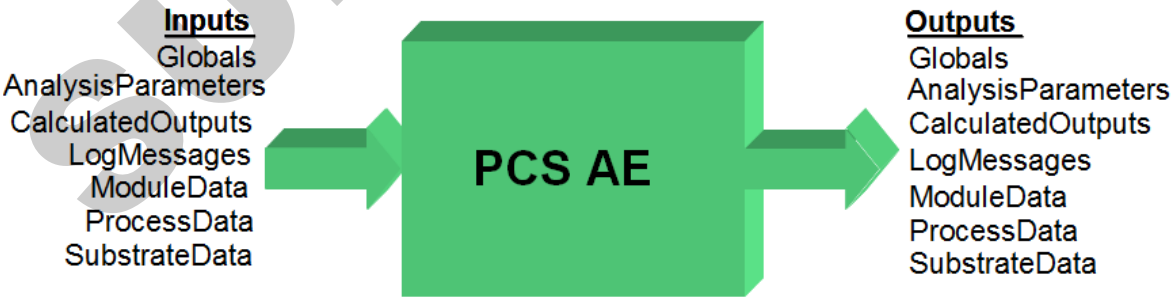

Figure 2
PCS AE Common Input/Output Structure

9.2.2.4 Note that based on the simple data model shown in Figure 2, Inputs are available as an Output from an AE. Data provided as input to an AE should be accessible as output from the AE for historical purposes.

SUPERSEDED
Documents
Document Sharing Hub
Full document details: SEMI E133-0212

Table 3 PCS AE Inputs and Outputs Collection Type Definitions

Collection Type	*Definition*	*Comments*
Globals	Information that applies globally across an entire job.	A lot id, for example, could be specified once as a global so that it would not have to be specified with many other data sets.
AnalysisParameters	Information that identifies numeric data used by the AE to perform various calculations.	Equation coefficients, feedforward data, etc.
CalculatedOutputs	Data representing the results of analysis from an AE.	A calculation derived from various inputs to the AE indicating wafer state, process state or model state information, recommended machine settings, etc.
LogMessages	Data describing the activity of an AE as it executes its tasks.	Data describing the activities of the AE. Note that this does not include log information on the communication protocol with the AE. An example that is acceptable is "Algorithm #3 with model parameters 'A', 'B', and 'C'."
ModuleData	Information representing the current or last known physical state of a process module that will be used by an AE to process material.	Machine, or module E10 state, RF hours, number of wafers processed, etc.
ProcessData	Information representing the process itself, such as the name and type of the process and the current or last known process state based on measurements from sensors of the processing environment.	Name of the process (CMP), in situ process measurements, trace data, and summary statistics etc.
SubstrateData	Information about the substrate being associated with the AE. This could include data representing a single wafer, multiple wafers, or one or more lots, as well as attributes of the material.	Profile or topology data, and other characteristics measured physically or electrically.

Table 4 PCS AE FG Capabilities

#	*Description*	*Reqd*	*Comments*
3.1	**Provide a PCS AE capability** PCS AE shall be capable of performing one or more of R2R control, FD, FC, FP, SPC, or any future functionality defined in this Standard as a PCS FG.	Y	
3.2	**Provide the required PCS AE methods** The PCS AE shall provide the required interface services as defined in this Standard.	Y	These include PCSAnalyze and PCS UpdateModel.
3.3	**Maintain models of selected PCS environment** The PCS AE shall maintain explicit or implicit models of the selected process control environment. These models shall be utilized to provide the analysis or control capability of the Analysis Engine.	Y	The specific form and behavior of these models is beyond the scope of this Standard.

SUPERSEDED

Documents Document Sharing Hub
Full document details: SEMI E133-0212

#	Description	Reqd	Comments
3.4	**Provide a mechanism to uniquely instruct the AE to perform an action** The PCS AE shall support an input mechanism whereby the analysis engine can be instructed to take a specific unique action.	Y	The mechanism could be a single key variable; e.g., model name or a complex set of context data utilized internally to perform CM, etc. The action could include application of a specific model and generation of a PCS result, update of a model, or communication of model parameters.
3.5	**Interface with multiple external systems** The PCS AE shall be capable of accepting inputs from, or generating outputs to multiple external systems.	Y	The definition of these systems will be a function of the application environment; these could include process tool or metrology systems, or user interfaces.
3.6	**Edit configuration of models** The PCS AE shall provide the capability to create/add, edit configuration, and delete models.	Y	This capability will be supported through the Analysis Engine interface.
3.7	**Interface with other AEs** The PCS AE shall be capable of either accepting outputs from other AEes as inputs, or producing outputs that are utilized by other AEs.	N	May be a requirement of AEs belonging to a specific FG.
3.8	**Advanced model management** The PCS AE may provide advanced model management capabilities including but not limited to: secure access to models, and automatic updating of models based on process control environment changes.	N	

9.2.3 *Common Interface Data Structures*

9.2.3.1 *Common Data Model Overview* — The common interface data structures from which all PCS FGs may inherit are shown in Figure 3. This figure shows the basic types of information ('classes' in object terminology) that can be supported by the interface. This model applies to the structure of inputs as well as outputs of a PCS AE as shown in Figure 2. This model applies to a complete message associated with a PCS job. That is it applies to the complete set of information communicated to or from a PCS AE for a PCS job. Note that a complete message into or out of a PCS AE may be broken into multiple application-to-application communications as defined in ¶ 9.2.3.5.

SUPERSEDED

Documents
Document Sharing Hub
Full document details: SEMI E133-0212

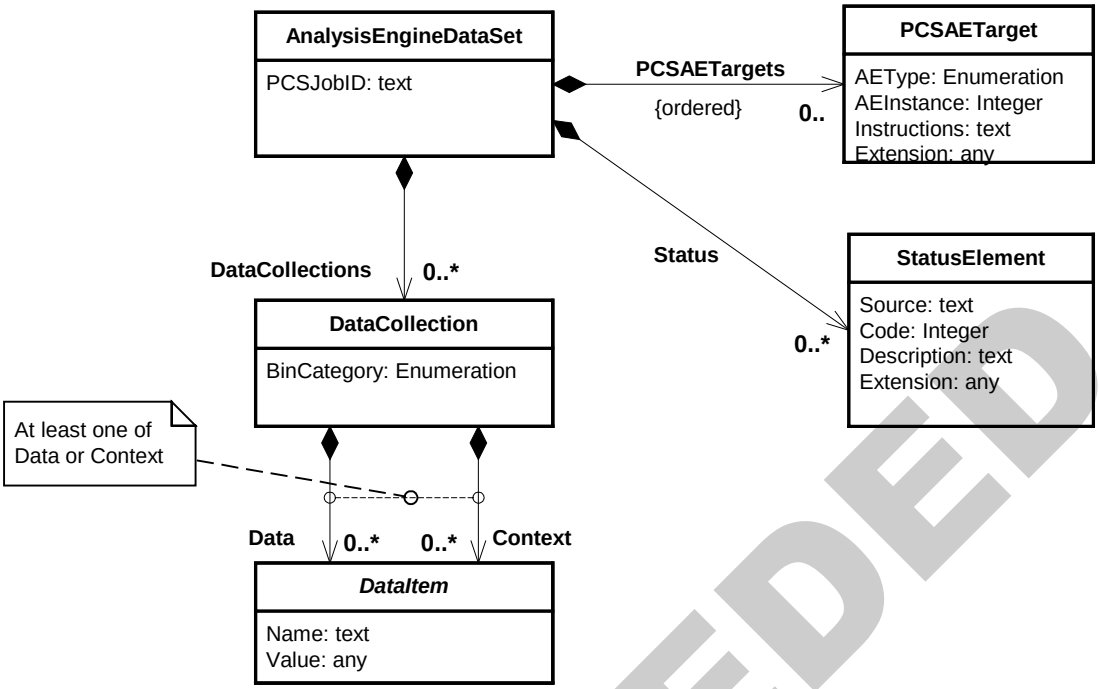

Figure 3
AE Data Structure Model for a Complete Message Associated with a PCS Job

9.2.3.2 *Class Definitions* — Class definitions consist of a brief description of each class, a table defining any attributes belonging to that class, and further text as necessary to fully define the class.

9.2.3.2.1 *AnalysisEngineDataSet* — A composite data structure that represents the data associated with a single input or output message of an AE.

Table 5 Attributes of AnalysisEngineDataSet

Attribute Name	*Description*	*Access*	*Reqd*	*Form*
PCSJobID	A unique identifier of a PCS job.	NA	C	ObjID – Reference SEMI E39.
PCSAETargets	An attribute that indicates a list of AE types targeted by the AnalysisEngineDataSet.	NA	Y	An ordered list of structured data of type PCSAETarget; see below.
Status	An attribute formatted as an extensible data structure that relates properties of the success or failure of a PCS AE, identified in PCSAETargets, to execute a PCS job.	NA	C	An unordered list of structured data of type StatusElement; see below.

9.2.3.2.1.1 *PCSJobID* — The specification that PCSJobID is a required attribute of AnalysisEngineDataSet along with the specification that it is a unique identifier of a PCS job means that a unique PCSJobID shall be associated with each PCS job. The management of the uniqueness of the PCSJobID can be performed by an AE or by an application that wishes to utilize the capabilities of an AE. The application communicating with the AE is assumed to generate and manage PCSJobIDs for all PCS jobs, however the AE is required to provide this functionality in the event that the application does not wish to provide it. The existence of the PCSJobID attribute is conditional upon the following:

SUPERSEDED

Documents Document Sharing Hub
Full document details: SEMI E133-0212

- If the application communicating with the AE is managing a PCSJobID for a PCS job, then that PCSJobID shall be provided as the value of the PCSJobID attribute for all interface data structures of type AnalysisEngineDataSet associated with that PCS job.
- If the AE is providing the functionality of generating a PCSJobID for a PCS job, then the PCSJobID shall be provided in the first communication from the AE (to any consuming application) for this job and all subsequent communications associated with this job.
- The PCSJobID is not required for transactions that do not involve the AE executing a PCS job such as status checks, or pre-processing queries.

9.2.3.2.1.2 *PCSAETargets* — This attribute indicates a list of AE types targeted by the AnalysisEngineDataSet; it is structured as an ordered list of type PCSAETarget (see below). At most one PCSAETargets value shall be associated with Each PCSJobID. Thus any two messages received by a PCS AE with the same PCSJobID will have at most one value for PCSAETargets, which could be provided in either or both messages. Each AE that receives this parameter in an input AnalysisEngineDataSet will echo the parameter and its value for all output AnalysisEngineDataSet messages with that PCSJobID.

9.2.3.2.1.2.1 *PCSAETarget* — This attribute class represents an AE targeted by the current PCS job. The PCSAETarget attribute is used to differentiate PCS AE targets when a single message travels between multiple AEs (often referred to as a 'daisy chain'). In this case the AEType and AEInstance attributes of the PCSAETarget (see below) can be utilized to identify a particular consumer for the AnalysisEngineDataSet data. Note that in many cases, such as point-to-point communications between an AE client and a single AE, the PCSAETarget attribute may not be necessary.

9.2.3.2.1.2.1.1 This attribute has the following form:

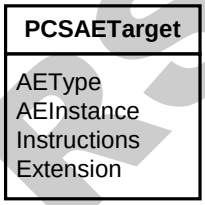

Figure 4
PCSAETarget Data Structure

Table 6 PCSAETarget Attribute Definition

Attribute Name	*Description*	*Access*	*Reqd*	*Form*
AEType	An enumeration that represents the AE type to which the instruction is targeted.	NA	N	Enumeration, with possible values being: fault detection (FDD), fault classification (FCC), fault prediction (FPP), run-to-run control (R2R), statistical process control (SPC), process control system (PCS).
AEInstance	A number indicating the specific instance of an AE within a type that is targeted.	NA	N	Integer.
Instructions	An ordered list of Instructions, where each instruction is of type text.	NA	Y	Ordered list of elements of type text.
Extension	This attribute allows the supplier to include additional information regarding the targeted AE instance.	NA	N	Any.

SUPERSEDED

Documents
Document Sharing Hub
Full document details: SEMI E133-0212

9.2.3.2.1.2.1.2 The specification of how the AEs will consume the information is beyond the scope of this Standard.

9.2.3.2.1.2.1.3 The AEType enumeration of 'PCS' is used to indicate a generic PCS type, or is an unspecified type.

9.2.3.2.1.2.1.4 An AEInstance value is not specified indicates that the instance of AEType is unspecified.

9.2.3.2.1.2.1.5 When multiple PCSAETarget instance values are specified, the target PCS application to which these values refer should execute the PCS job associated with the PCSJobID in sequence. The mechanism for ensuring this behavior is beyond the scope of this Standard.

9.2.3.2.1.2.1.6 Instructions is an ordered list of elements, each of which is an instruction and is of type text. Note that no two elements in the ordered list can have the same value. An *instruction* is an attribute that relates a specific task that is being requested to be carried out by a targeted AEInstance.

9.2.3.2.2 *Status* — This conditional attribute class relates properties of the success or failure of a PCS AE, identified in PCSAETargets, to execute a PCS job. The attribute has the following form:

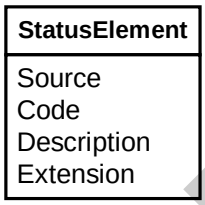

Figure 5
StatusElement Data Structure

Table 7 StatusElement Attribute Definition

Attribute Name	*Description*	*Access*	*Reqd*	*Form*
Source	Represents the document from where the attribute code value is defined. For example 'Source' could identify a SEMI Standards Document. For status codes defined by SEMI Standards, the content of Source is the urn representation of the Standard number. For status codes defined by a supplier, it is the urn representation of the supplier's name. (e.g., for SEMI E132 errors, the content of 'Source' is: urn:semi-org:E132; for supplier company errors, for instance, where company name is 'Acme,' the content is: urn:Acme-com) additional information may be included by the supplier by adding such information after the ':' colon symbol and separating the fields with a '.' period symbol. Please refer to SEMI E121 for additional information.	NA	N	Text.
Code	A value that directly maps to a particular specification status code list. The source of this list is beyond the specification of this document. A status code of zero indicates that no error has occurred, while any non-zero status indicates an error has occurred. Specific error codes defined for this Standard are provided in Table 8.	NA	Y	Integer.
Description	A human readable description of the corresponding code attribute being reported. It maps directly to the definition of the Code attribute.	NA	N	An unordered list of structured data; see below.
Extension	This attribute allows the supplier to include additional information regarding the status code encountered. Additional information may be supplied in this attribute to provide more detail about any error (e.g., Common Component, Code 5001: Insufficient Arguments Provided:Argument 'SessionID' missing).	NA	N	Any.

SUPERSEDED

Documents Document Sharing Hub
Full document details: SEMI E133-0212

Table 8 SEMI E133 Code Attribute Value Definitions

StatusElement Parameter			
Source	*Code*	*Description*	*Extension*
SEMI E133	0	No error has occurred	<manufacturer specific>
SEMI E133	≠ 0	An error has occurred	<manufacturer specific>
SEMI E133	−1	Message syntax error (e.g., XML validation error, see SEMI E133.1)	<manufacturer specific>
SEMI E133	−2	Unknown or unexpected PCSJobID	<manufacturer specific>
SEMI E133	−3	Unknown or unexpected service	<manufacturer specific>
SEMI E133	−4	Unknown or unexpected PCSAETarget	<manufacturer specific>
SEMI E133	−5	Service not supported	<manufacturer specific>
SEMI E133	−6	Service unavailable at this time	<manufacturer specific>
SEMI E133	−7	Unknown or unexpected data item	<manufacturer specific>
SEMI E133	−8 to −31	<Not specified: reserved for future generic AE use>	<manufacturer specific>
SEMI E133	−32 to −99	<Reserved for specific AE type use; reserved unless otherwise indicated in AE FG specification>	<manufacturer specific>
#1	<−99	Manufacturer specified	<manufacturer specific>
#1	>0	Manufacturer specified	<manufacturer specific>

#1 Manufacturer Specific.

9.2.3.2.2.1.1 The existence of the StatusElement attribute is conditional on the properties of the message containing a data structure of type AnalysisEngineDataSet. The StatusElement attribute shall exist if and only if the following conditions apply:

- The message is an output from an AE.
- The message is a response to a request for service(s) from that AE (e.g., a reply message).

9.2.3.2.3 *DataCollection* — A set of data and associated context for that data that conforms to the PCS AE common input/output structure as defined in ¶ 9.2.2.1.

Table 9 Attributes of DataCollection

Attribute Name	*Description*	*Access*	*Reqd*	*Form*
BinCategory	An enumerated identifier of the type of data collection among the possible types defined in the collection type enumeration, listed in Table 4.	NA	Y	CollectionType – As defined.

9.2.3.2.3.1 *CollectionType* — An enumeration of BinCategory as defined in Table 4.

9.2.3.2.4 *DataItem* — An individual component of a data collection that has meaning in the PCS environment. Each data item has a role of either ‘context’ or ‘data’ as determined by the client of the PCS AE.

Table 10 Attributes of DataItem

Attribute Name	*Description*	*Access*	*Reqd*	*Form*
Name	Human-readable name for DataItem.	NA	Y	StringValue – See rules for naming below.
Value	An assigned or calculated quantity.	NA	Y	AbstractValue – As defined in SEMI E134.

SUPERSEDED

Documents
Document Sharing Hub
Full document details: SEMI E133-0212

9.2.3.2.4.1 *Name* — This attribute is used to provide a level of understanding of the meaning and purpose of the DataItem. 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.

9.2.3.2.4.2 *Value* — AbstractValue (as defined in SEMI E134) is an abstract base class that is used to represent standard types including: Integer, Real, Boolean, String, Binary, Structure, Array, and Enumerations. In this way, parameter values collected via SEMI E134 may be passed directly into the PCS compliant AE via the value attribute of a DataItem. Refer to SEMI E134 for further definition of the AbstractValue type. AbstractValue base class allows for arbitrarily complex data items to be passed into the AE.

9.2.3.2.4.3 A DataItem that belongs to a DataCollection whose CollectionType is 'Globals' may not serve a role of data (i.e., it can only serve the role of context).

9.2.3.2.4.4 It is not required that the PCS AE utilize a data item in the same role (Data or Context) as is specified by the PCS AE client, however the PCS AE is required to maintain the role representation in I/O data structures produced or consumed across a PCS AE interface.

9.2.3.3 *Ordering of Elements Within an AnalysisEngineDataSet* — In addition to compliance with a data structure model, this Standard also defines rules governing ordering of classes and attributes within each AnalysisEngineDataSet instance, corresponding to the data content of an AE I/O message. Figure 4 provides a general illustration of this ordering. The following are rules governing that ordering.

9.2.3.3.1 The first attribute of any AnalysisEngineDataSet shall be the PCSJobID, if the PCSJobID value has been determined as defined in ¶ 9.2.3.2.1.1.

9.2.3.3.2 The first attribute of any DataCollection shall be BinCategory.

9.2.3.3.3 The first DataCollection shall be of BinCategory 'Globals' if the AnalysisEngineDataSet contains a DataCollection of that kind. At most one DataCollection shall be of BinCategory 'Globals' in any AnalysisEngineDataSet.

9.2.3.3.4 In each DataCollection all DataItems whose role is 'Context' shall occur before the first DataItem whose role is 'Data' occurs.

9.2.3.4 Additional Requirements of an AnalysisEngineDataSet

9.2.3.4.1 If multiple AnalysisEngineDataSet instances are 'related,' as defined in ¶ 9.2.3.2.1 and both contain the same DataCollection with the same value of BinCategory, and further contain the same DataItem (with the same 'Name') within that DataCollection, the last DataItem shall be assumed to be correct. The interpretation of ordering of information as communicated by the underlying protocol is beyond the scope of this Document.

9.2.3.4.2 Within an AnalysisEngineDataSet instance or a group of instances that are 'related,' as defined in ¶ 9.2.2.1, a DataItem may exist within multiple DataCollections that has a role of context. If this condition exists, the following rules shall apply.

9.2.3.4.2.1 Any Context DataItems defined in a DataCollection with BinCategory = 'Globals' should be used as Context DataItems for all other BinCategorys in any related message except as defined in ¶ 9.2.3.4.2.2 below.

9.2.3.4.2.2 If a DataCollection with a BinCategory other than 'Globals' defines a Context DataItem that is also defined in the Global BinCategory of the current message or a related message, the local BinCategory DataItem value should be used as context (i.e., the local Context DataItem overrides the Global value inside this DataCollection).

9.2.3.5 *Using Multiple Application-to-Application to Communications to Communicate a PCS Message* — A PCS message may be broken into a number of application-to-application communications or partitions, however it is not required that a PCS AE support this partitioning capability. Each of these communication partitions must conform to the data model of Figure 3, with the following modifications:

9.2.3.5.1 PCSAETarget minimum cardinality is zero.

SUPERSEDED

Documents Document Sharing Hub
Full document details: SEMI E133-0212

- PCSJobID == “[0-9a-fA-F]{8}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{12}”
- PCSAETargets
  - APCSAETarget: PCSAETarget
    - AEType == “R2R”
    - AEInstance == 1
    - Instructions
      - Instruction == “PCSAnalyze”
    - Extension == “execute serially”
- Status
  - AStatusElement: StatusElement
    - Source == “urn:semi-org:E132”
    - Code == 0
- ADataCollection: DataCollection
  - BinCategory == “Globals”
    - Role == “Context”
      - ADataItem: DataItem
        - AName: (type == text)
        - AValue: (type == any)
      - ADataItem
        - AName: (type == text)
        - AValue: (type == any)
  - BinCategory == “ModuleData”
    - Role == “Context”
      - ADataItem: DataItem
        - AName: (type == text)
        - AValue: (type == any)
      - …
    - Role == “Data”
      - ADataItem: DataItem
        - AName: (type == text)
        - AValue: (type == any)
      - …
  - BinCategory == “ProcessData”
    - Role == “Context”
      - ADataItem: DataItem
        - AName: (type == text)
        - AValue: (type == any)
      - …
    - Role == “Data”
      - ADataItem: DataItem
        - AName: (type == text)
        - AValue: (type == any)
      - …
  - …

Figure 6
Illustration of Ordering of Information in an AnalysisEngineDataSet Instance
(Partial example; ‘…’ is used to indicate possible continuation in the listed format;
PCSJobID in this example is aligned with specifications in SEMI E120.1)

SUPERSEDED

9.2.4 *Common Interface Services*

9.2.4.1 Except as specified in ¶ 0 PCS AEs shall provide services by accepting and producing messages that are compliant with the common interface data structure as defined in § 9.2.3. PCS AEs shall support both Request and Notification services as specified herein. For Request services, both the Request and Response messages associated with the service shall comply with the common interface data structure specified in § 9.2.3, which includes a specification of required and optional elements. The success or failure of performing this service is conveyed in the Response message utilizing the ‘Code’ attribute (see ¶ 9.2.3.2.2). The ‘Code’ attribute is required in all response messages associated with request services. For Notification services, the message associated with the notification, whether produced or consumed by an AE, shall comply with the common interface data structure. Each service

Documents
Document Sharing Hub
Full document details: SEMI E133-0212

parameter, when provided in a service message, shall be provided as a DataItem in a data bin of the appropriate BinCategory (see Figure 3 and associated text), unless otherwise specified.

EXCEPTIONS: The following services are not required to comply with aspects of the common interface data structure as specifically noted.

- PCSPing service: A Ping service response is not required to contain a PCSJobID parameter.

9.2.4.2 All PCS AEs shall support the following common interface services:

9.2.4.2.1 A set of services shall exist to support access to data from the engine.

9.2.4.2.2 A set of services shall exist to support logging and exception handling.

9.2.4.2.3 Specific services of an AE are invoked utilizing the Instructions element of the PCSAETarget attribute, as defined in § 9.2.3. Specifically the service name is the value of an Instruction list element. For Request (Synchronous) services, the service name is the value of an Instruction list element for both the Request message and the corresponding Response message. For Notification services the service name is the value of an Instruction list element for the notification message. Utilizing this mechanism a set of services shall exist as defined in Table 11.

Table 11 General AE Object Service Resource Definition[#1]

Service	*Reqd*	*Type*	*Description*
PCSAnalyze	Y	R	Used to instruct an AE to compute and process a logical data sample. The results of the analysis are provided in the Response message. This service can be used for both synchronous and asynchronous PCS jobs.[#2]
PCSStartJob	N	R	Used to instruct an AE to begin an asynchronous PCS job.[#2]
PCSAddMoreData	N	N	Used to provide more raw data to an AE executing an asynchronous PCS job.[#2] The AE consumes this notification message.
PCSEndJob	N	R	Used to instruct an AE to complete an asynchronous PCS job.[#2]
PCSJobCompleted	N	N	Used by the AE to indicate that an asynchronous PCS job has been completed.[#2] The AE produces this notification message.
PCSAbortJob	N	N	Used to abort (i.e., terminate) an asynchronous PCS job.[#2] The AE consumes this notification message.
PCSJobAborted	N	N	Used by the AE to indicate that an asynchronous PCS job has been aborted.[#2] The AE produces this notification message.
PCSUpdateModel	Y	R	Used to instruct the AE to update the analysis model with new information. This service can be used for both synchronous and asynchronous PCS jobs.[#2]
PCSEvent	N	N	Used to provide an indication of detection of an event. An event could include the receipt of specific data. The specification of what constitutes the event is beyond the scope of this Standard.
PCSProduceLogData	N	R	Used to instruct the AE to produce log messages. Information passed with a log data request could relate the date range and context associated with the request, while information returned could include the requested log data.
PCSProduceExceptionData	N	R	Used to instruct the AE to provide information on the occurrence of exceptions raised by the AE. The exception information would be returned by the AE to the requestor, and could include a date range, similar to log data.
PCSExceptionProduced	N	N	Used by the AE to indicate that an exception has occurred. The exception information provided with the notification could include a date range, similar to log data.
PCSPing	Y	R	Used to ascertain the existence and communication health of the AE.
PCSCapabilitiesDiscovery	N	R	Use to query a PCS AE as to its support of capabilities, including those defined in this Standard, and whether or not each capability is currently enabled or disabled. The value of the Capabilities DataItem (as defined in ¶ 9.2.4.2.4) in the service response indicates which capabilities are supported, and, among these, which are enabled and which are disabled.

SUPERSEDED

Documents Document Sharing Hub
Full document details: SEMI E133-0212

Service	*Reqd*	*Type*	*Description*
PCSEnable	N	R	Used to put the PCS AE in a state where it is providing a subset of the capabilities it is specified as being capable of providing. The value of the Capabilities DataItem (as defined in ¶ 9.2.4.2.4) indicates which capabilities are requested to be enabled or disabled. The value of the Capabilities DataItem in the corresponding response indicates which capabilities were actually enabled.

#1 Except as specified in ¶ 0 PCS AEs shall provide services by accepting and producing messages that are compliant with the common interface data structure as defined in § 9.2.3.

#2 See § 9.2.5.

9.2.4.2.4 *Service Parameters* — Parameters for these services are described in Table 12.

Table 12 PCS Service Parameter Definition

BinCategory	*DataItem Name*	*Description*	*Role*[#1]	*Form*
Globals	DataSource	Indicates the source or sources of data associated with this PCS job. For example, it could indicate an EDA tool and data collection plan.	D	The form of the DataSource parameter is beyond the scope of this Standard. For example it could be a text string that identifies an EDA data source or context data to point to a particular interface on a particular tool.
Analysis-Parameters	Capabilities	A complex data item containing: (1) a data element of Boolean form; a value of TRUE in a request service is an indication that ALL supported capabilities are to be enabled; a value of TRUE in a response service message is an indication that all capabilities are currently enabled; a value of FALSE in a request service is an indication that ALL supported capabilities are to be disabled; a value of FALSE in a response service message is an indication that not all capabilities are currently enabled. (2) A list of elements each of which provides information on each AE FG Capability, as listed in Tables 13 and 14, that is supported. Specifically each element in the list provides the following information; (2a) The number of the capability, as defined in this Standard, that is supported (e.g., R2R.A.1) (2b) The different ways in which the capability is supported. (2c) In a request message, an indication as to whether the capability is to be enabled or disabled; in a response message, an indication as to whether the capability is currently enabled or disabled.	D	See description; additional specification of form is beyond the scope of this Standard. It is only required that this parameter be used to indicate which capabilities are supported. However, as desired indication of lack of support (e.g., via a value of FALSE) can be provided for 2b to indicate that a capability is not supported.

#1 Where, in the Role Column, D indicates 'Data' and C indicates 'Context.'

9.2.4.2.4.1 *PCSStartJob* — Table 13 describes the parameters specified for this service.

SUPERSEDED

Documents
Document Sharing Hub
Full document details: SEMI E133-0212

Table 13 PCSStartJob Service Parameter Definition

Parameter	*Request*	*Response*	*Form*	*Description*
DataSource	C[#1]	-	See § 9.2.4.2.4.	See § 9.2.4.2.4.

#1 Parameter provided in the Request message if the service requestor wishes to indicate the data source or sources to be utilized in this (asynchronous) PCS job.

9.2.4.2.5 *PCSCapabilitiesDiscovery* — Table 14 describes the parameters specified for this service.

Table 14 PCSSCapabilitiesDiscovery Service Parameter Definition

Parameter	*Request*	*Response*	*Form*	*Description*
Capabilities	N	R	See § 9.2.4.2.4.	See § 9.2.4.2.4.

9.2.4.2.6 *PCSEnable* — Table 15 describes the parameters specified for this service.

Table 15 PCSEnable Service Parameter Definition

Parameter	*Request*	*Response*	*Form*	*Description*
Capabilities	R	R	See § 9.2.4.2.4.	See § 9.2.4.2.4.

9.2.5 *Common Behavior*

9.2.5.1 All PCS AEs shall be able to accept inputs and provide outputs through a combination of services which include those defined in Table 11, General AE Object Service Resource Definition.

9.2.5.2 *PCSJob Management Behavior*

9.2.5.2.1 A PCS AE provides PCS capabilities by executing PCS jobs. This section defines the behavior of the AE associated with the management of a PCS job in that AE. A PCS AE may manage multiple PCS jobs simultaneously. The maximum number of concurrent jobs that shall be maintained by an AE is beyond the scope of this Standard, however this section defines the behavior of each job that PCS AE is managing.

9.2.5.2.2 The following PCS job behavioral classes are specified.

- Synchronous PCS job: The AE responds to PCS job request message for a new job with a single response message produced that contains the response to the request that is the job output. After the response is issued by the AE, the job ends.
- Asynchronous PCS job: The AE responds to PCS job requests message for a new job by opening a session with the requestor. During that session, for messages associated with that particular PCS job, the AE may respond synchronously to request messages (by producing appropriate response messages), consume notification messages without providing a response, and/or asynchronously produce notification messages. The asynchronous PCS job ends at an AE if any of the following conditions are met:
  - A request to end the job is received by the AE via a PCSEndJob service, the AE determines that it can and should end the job, and the AE sends a positive response to the request;
  - The AE determines that the job is complete and it should end the job, and provides a notification via the PCSJobCompleted service that the job has ended;
  - The AE receives a notification to abort the job via the PCSAbortJob service, and the AE can process the request;
  - The AE determines that it should abort the job and sends a notification to abort the job via the PCSJobAborted service.

SUPERSEDED

Documents Document Sharing Hub
Full document details: SEMI E133-0212

9.2.5.2.3 An AE may support one or both of these behavioral classes; additional specification as to this support may be provided in the individual AE type specification. The behavioral class to which the AE complies in executing a PCS job depends on the specific type of service request received (see § 9.2.4) and the capabilities of the AE.

9.2.5.2.4 Additional specification of AE behavior can be found in the behavioral specifications for that specific analysis engine type. An AE may provide or specify additional behavior as long as it does not conflict with the behavior defined for that class in this Standard.

9.2.6 *AE General Capability Description*

9.2.6.1 An AE module should accept the following inputs and outputs using the common interface structure described in Figure 2. Not all inputs and outputs are required depending on the functionality needed by the AE.

9.2.7 *Inputs* — The following data types and data values are used in AE input messages.

9.2.7.1 No data types or data values are specified for the general AE.

9.2.8 *Outputs* — The following data types and data values are used in AE output messages.

9.2.8.1 No data types or data values are specified for the general AE.

9.3 *R2R Control FG*

9.3.1 This section defines the common capabilities and interface data structures of the R2R control FG. Note that this includes W2W control capabilities.

9.3.2 Table 16 below describes the specialization for the R2R control FG of each inherited general capability described in Table 5.

9.3.3 Table 17 describes the additional capabilities unique to the R2R control FG.

9.3.4 *R2R General Capability Description*

9.3.4.1 A R2R module should accept the following inputs and outputs using the common interface structure described in Figure 2. Not all inputs and outputs are required depending on the functionality needed by the R2R model. For example, R2R function may only need feedback data in the form of Substrate Data.

9.3.5 *Inputs* — The following data types and data values are used in R2R control input messages:

9.3.5.1 The R2R controller AE inherits the input data structure and data structure requirements from the PCS AE.

9.3.5.2 The following data types and data type values shall be supported by a R2R control AE when included in an input data structure, where supported means that the AE will be able to utilize the information provided by the data type and perform any indicated action associated with that data:

9.3.5.2.1 *PCSAETarget Class*

9.3.5.2.1.1 Attribute AEType with a value of ‘R2R.’

9.3.5.2.1.2 A list element of attribute Instructions with a value of ‘PCSAnalyze.’

9.3.5.2.1.3 A list element of attribute Instructions with a value of ‘PCSUpdateModel.’

9.3.6 *Outputs* — The following data types and data type values are used in R2R control output messages:

9.3.6.1 The R2R controller AE inherits the output data structure and data structure requirements from the PCS AE.

9.3.6.2 The following data types and data type values shall be supported by a R2R control AE when included in an output data structure, where supported means that the AE will be able to provide these data types and values when behavioral conditions warrant, as indicated below.

9.3.6.2.1 *DataCollection Class* — The following DataItem Names or Name/Value pairs shall be supported for DataItems in the specified BinCategory.

9.3.6.2.2 *ControlAdvice* — This attribute contains the control advice, structured as an unordered list of name-value pairs. The method in which the control advice is utilized is beyond the scope of this Document.

SUPERSEDED

Documents
Document Sharing Hub
Full document details: SEMI E133-0212