Tables

1 Purpose

1.1 *Integrated Measurement*

1.1.1 The purpose of integrated measurement (metrology or inspection) is to facilitate intra-equipment process monitoring through rapid access to measurement data, reduce material handling between process and measurement equipment, and the opportunity to increase process monitoring with minimal or no decrease in throughput. The benefits of integrated measurement also allow Advanced Process Control systems to use the results with reduced feedback lag time.

1.2 *Specification for the Integrated Measurement Module Communications (IMMC)*

1.2.1 The purpose of the IMMC specification is to provide an object-based specification of an Integrated Measurement Module together with a standard interface between an integrated measurement module and its control and data ports where these are commonly implemented by different suppliers. The interface allows access to the properties and services of specific objects. This will facilitate the effort needed for the integration of the module into a larger system.

1.2.2 An additional purpose of this standard is to provide sufficient information through a combination of on-line services and interface documentation that an IMMC-compliant integrated measurement module may be integrated with multi-module equipment without requiring a software change in either the module or the equipment. This may require configuration changes made by the end-user through the equipment user interface where certain options are left to the module supplier.

2 Scope

2.1 *Scope of This Document*

2.1.1 The Integrated Measurement Module Communications specification covers concepts, behaviors, and services to be provided by a metrology or inspection module so that it may be integrated into production equipment intended for large substrates processing. However, nothing should preclude its application to smaller substrate manufacturing.

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 *Host/Integrated Equipment Interface*

3.1.1 This specification does not apply to the communications interface between the host and integrated equipment.

3.2 *Object-Based Implementation*

3.2.1 Compliance to this standard does not require object-oriented implementation. However, it does require the appearance of implentation of those objects defined in this document.

3.3 *In-Situ Processing*

3.3.1 This specification applies only to measurement components that are contained within a module that provides its own material handling and does not measure during a process step. Interactions with in-situ measurement components are expected to differ significantly from the current specification.

4 Referenced Standards and Documents

4.1 *SEMI Standards*

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

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

SEMI E30.5 — Specification for Metrology Specific Equipment Model (MSEM)

SEMI E32 — Material Movement Management (MMM)

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

SEMI E40 — Standard for Processing Management

SEMI E41 — Exception Management (EM) Standard

SEMI E42 — Recipe Management Standard: Concepts, Behavior, and Message Services

SEMI E53 — Event Reporting

SEMI E58 — Automated Reliability, Availability, and Maintainability Standard (ARAMS): Concepts, Behavior, and Services

SEMI E90 — Specification for Substrate Tracking

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

SEMI E116 — Provisional Specification for Equipment Performance Tracking

SEMI E120 — Provisional Specification for the Common Equipment Model (CEM)

SEMI M20 — Specification for Establishing a Wafer Coordinate System

4.2 *Other Sources*

Unified Modeling Language (UML) Specification, Version 1.4, OMG Specification 01-09-67, available from http://www.omg.org/technology/documents/modeling_spec_catalog.htm.

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

NOTE 1: Terms defined in §5 that are used in other definitions have been underlined.

5.1 *Abbreviations and Acronyms*

5.1.1 *IMM* — Integrated Measurement Module

5.1.2 *IMMC* — Specification for Integrated Measurement Module Communications: Concepts, Behavior, and Services

5.1.3 *IMMDTOSM* — IMM Data Table Object State Model

5.1.4 *SLOSM* — Substrate Location Object State Model (SEMI E90)

5.1.5 *STPO* — Substrate Transfer Path Object, an abstraction of the IMM capability to load and unload substrates into the IMM.

5.1.6 *STPOSM* — Substrate Transfer Path Object State Model

5.2 *Definitions*

5.2.1 *array*, n. — an ordered list of numeric values. A valid data type for table row entries.

5.2.2 *conversion recipe*, n. — a recipe or portion of a recipe that describes the steps required for taking measurement data and reporting a result. This does not need to be a physically separate recipe.

5.2.3 *error message*, n. — a notification to the user/client that an error has occurred. It may or may not be associated with an alarm. Note: depending upon the communication protocol, error messages may or may not be treated differently from normal collection events.

5.2.4 *inspection*, n. — an examination of an area of material to detect anomalies.

5.2.5 *inspection module,* n. — a measurement module that inspects substrates and reports information regarding anomalies. Inspection modules may determine the location of anomalies relative to a coordinate system and may also provide other types of data related to the anomaly.

5.2.6 *integrated measurement module,* n. — a measurement module intended to be integrated into manufacturing equipment, and with the capability of receiving substrates from the equipment, measuring those substrates, and returning the substrates and the measurement results to the equipment and other concerned clients.

5.2.7 *measurement module,* n. — an equipment module whose intended function is to measure or inspect the product and to report the results. Measurement of the product is the factory's means of gaining feedback on the manufacturing process.

5.2.8 *measurement recipe,* n. — a recipe or portion of a recipe intended for use during a measurement, that describes among other things the locations for measurement. This does not need to be a physically separate recipe.

5.2.9 *metrology module,* n. — a measurement module that collects and reports information on specific predetermined locations or features on a substrate with consistent data structure, or reports general information about the entire substrate.

5.2.10 *object-based,* adj. — a programming language, or database, is called object-based if it supports the concept of data abstraction, but partly or entirely lacks more advanced concepts such as class, inheritance, polymorphism, and so on. [Oestereich, Bernd, "Developing Software with UML," Addison-Wesley (1999)]

5.2.11 *substrate context information,* n. — information concerning the substrate that may be useful to for analysis, such as process flow step, substrate orientation, the identifier of the process equipment/chamber most likely to have affected results, the recipe run on that equipment/chamber, etc.

5.2.12 *substrate orientation,* n. — the angle of rotation from normal. For wafers, this is the angle of rotation from the primary fiducial.

6 Conventions

6.1 *Object Conventions*

6.1.1 This document conforms to the conventions for objects established by SEMI E39, including object diagrams, object terminology, and requirements for standardized objects. However, the notation used for object diagrams is the Universal Modeling Language (UML) notation (see ¶4.2 for additional detail).

6.1.2 *Formal Name of an Object* — The text capitalizes formal object name references, similar to the way capitalization is normally used when discussing entities. When describing something in the general (like cities) lower case is used, but when a specific entity is of interest (New York City), then first letters are capitalized. Where words are concatenated, they retain their capitalization to enhance readability.

6.1.3 *Object Attributes* — Attribute tables define those public attributes that can be read and set through the basic OSS services GetAttr and SetAttr. Simple attributes have a single data type as their value. Complex or compound attributes are made up of an ordered set of other elements, either a list or a structure. See SEMI Compilation of Terms for the definition of "form" for both the list of valid data type formats and for the different types of formats themselves.

6.1.4 *Attribute Definitions* — Attributes are formally defined in an attribute definition table with the following form:

Attribute Name	*Definition*	*Access*	*Reqd*	*Form*
The formal name of the attribute.	Defines the requirements of the attribute.	RO or RW	Reqd or Optional	Data type: see SEMI E39 ¶4.5. Complex attributes must declare the order of their components.

6.1.5 *Complex Attribute Data* — The individual data items of Complex attributes are defined in a separate Attribute Definition Table as individual attributes. However, these data items are not Attributes and the SEMI E39 GetAttr and SetAttr services are not valid for them.

6.1.6 *Lists of Data* — Lists of data shall be identified as 0+ when a null list is permitted. Otherwise, they shall be identified as 1+.

6.2 *State Model Conventions*

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

6.2.2 The Harel convention has no concepts of "creation" and "extinction" for expressing the state model of a temporary entity. The "job" described in this document is such an entity, and a copy of the same state model is used for an independent job newly created. In this document, a circle with a black circle inside is used for expressing extinction of an entity. A filled black circle denotes the entry to the state model (the entity creation). This is the convention used in the OMT notation as well.

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

1. Actions taken upon exit of the previous state.
2. Actions taken upon entry of the new state.
3. Actions taken which are most closely associated with the transition.

No differentiation is made between these cases.

#	*Previous State*	*Trigger*	*New State*	*Actions*	*Comments*

6.2.4 State models are referred to in the Event Variables tables as an acronym consisting of the first letters of the state model name, ending in "SM" for State Model. Specific transitions with the state model are represented with the state model acronym plus "-Tn", where n represents the number of the transition. State model transitions without numbers are not expected to be reported.

6.3 *Service Message Representation*

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

6.3.2 *Service Definition Table* — A service definition table defines the specific set of messages for a given service resource, as shown in the following table:

Message Service Name	*Type*	*Description*

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

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

6.3.3 *Service Parameter Dictionary* — A service parameter dictionary table defines the description, format and its possible value for parameters used by services, as shown in the table below. A row is provided in the table for each parameter of a service. See SEMI Compilation of Terms for the definition of "form" for both the list of valid data type formats and for the different types of these formats.

Parameter Name	*Description*	*Format: Possible Value*

6.3.4 *Service Message Definition* — A service message definition table defines the parameters used in a service, as shown in the following table:

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

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

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

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

7 General Requirements

7.1 *Protocol Requirements*

7.1.1 The communication protocol used must support use of all data types, including lists, structures, and arrays.

7.2 *Required Standards*

7.2.1 This section discusses specific SEMI standards that have been assumed for the proper operation of an integrated measurement module (IMM) as specified in this document.

7.2.2 *SEMI E40* — Fundamental compliance to SEMI E40 for Processing Management is required. A process job queue is not required unless the IMM is capable of holding multiple substrates.

7.2.3 *SEMI E90* — Fundamental compliance to SEMI E90 for Substrate Objects and Substrate Location Objects is required.

7.2.4 *SEMI E116* — Fundamental compliance to SEMI E116 (Equipment Performance Tracking) is required. The IMM is to be considered an EPT Module. Guidance for implementing EPT for the IMM is provided in Related Information 2.

7.3 *Required Basic Capabilities*

7.3.1 The requirements in this section are concerned with basic capabilities that may be satisfied in different ways. Because of this flexibility, the method supported must be clearly documented by the IMM supplier. The IMM Control Client should allow on-line configuration by a user to achieve seamless "plug and play" integration.

NOTE 2: New standards are evolving and may become the required method for satisfying these basic capabilities in the future.

7.3.1.1 These capabilities are specified between the IMM and its control client. If, as expected for integrated measurement modules, the client is located within or near the process tool, it is up to the client to have the ability to communicate the appropriate information (alarm reporting, recipe upload/download) to the host.

7.3.2 *Clock Management* — The ability to provide accurate time/date information for reporting purposes is required. In addition, standard methods for clock synchronization are mandatory. This requirement shall be satisfied by compliance to the Clock object specification in SEMI E98.

7.3.3 *Event Management* — The ability for the IMM to report events and associated data of interest to its client is required. Data provided with these events shall be configurable and shall be valid at the time of the event. Each event report shall contain the date and time that the event occurred (and not the time that the report was sent). This requirement may be satisfied through compliance to a standard event mechanism, such as the event reporting capability defined in SEMI E30 or SEMI E53.

7.3.4 *Alarm Management* — The ability to report any change in its alarm conditions is required. An alarm condition is either set or cleared. A set alarm represents danger to the factory, the module, the equipment, the substrates, or the process. This capability may be satisfied through compliance to a standard alarm mechanism such as is defined in SEMI E30 or SEMI E41.

7.3.5 *Exception Management* — An exception is an error. Exceptions may or may not be alarms, either because they do not represent danger or because they represent occurrences that do not have a set or clear state. Exceptions shall be reported and time-stamped at the time the exception occurs. This capability may be satisfied through compliance a standard for exception management such to SEMI E41 or through reporting exceptions as events.

7.3.6 *Recipe Management* — Compliance to this standard requires the support of recipe upload and download and directory functions as specified in one of the SEMI standards defining these methods, such as SEMI E30 for process programs or SEMI E42 recipes.

NOTE 3: New standards for recipe management may be used for these purposes in the future, as they become available.

7.4 *State Models*

7.4.1 *State Model Transitions* — All numbered transitions in all state models defined in this standard represent uniquely identifiable and reportable events unless otherwise noted.

7.4.2 State models in this specification represent behavior as it appears to the client and are used to provide events to the communications partner. Actual state models implemented may differ. For example, a certain state model transition in this document may combine multiple transitions as implemented. It is critical that event representation of state model transitions as specified in the state models defined in this document be followed for reporting.

7.5 *Objects*

7.5.1 *SEMI E39* — Fundamental compliance to SEMI E39 is required for all objects defined in, or required by, this standard. In addition, the Integrated Measurement Module is considered an owner of all objects that it uses, manages, or that are components. If the IMM supports all object types specified in this standard, then it is not necessary to support the GetType service in SEMI E39. If all object types implemented have only those attributes specified by name in this standard, then the service GetAttrName is not required. Filtering for GetAttr and SetAttr is not required. Otherwise, to provide for a self-configuring interface, owner objects shall conform to the following additional services defined in SEMI E39:

- GetType
- GetAttrName
- GetServiceNames
- GetServiceParameters

7.5.2 All objects defined in this standard shall satisfy the fundamental requirements of SEMI E39. It shall be possible for the client to inquire about the attributes of one or more objects through the service GetAttr. It shall be possible for the client to change the value of attributes designated as read/write (RW) using the service SetAttr.

7.5.3 For purposes of communicating using SEMI E39, all objects other than the IMM object itself shall be owned directly by the IMM. The IMM itself is considered to be owned by its Client. For purposes of normal operations during manufacturing, the factory host considers the Control Client as the IMM owner.

7.6 *Service Requirements*

7.6.1 This document does not redefine services defined in other SEMI standards. However, for emphasis, those that are beyond the requirements of fundamental compliance to other standards are included in the lists of services as applicable.

8 Overview

8.1 *Background*

8.1.1 Both modern manufacturing processes and modern manufacturing equipment are increasingly complex. A single installation of equipment may have hundreds of sensors and actuators. In order to manage this complexity, better methods of referencing the internal components of equipment are needed. Use of the object paradigm provides a means for the equipment to describe its internal composition to the factory in a natural way.

8.1.2 Definition of standardized objects allows the factory to be specific about its requirements and its need for information.

8.1.3 The measurement module is not concerned about the role of its client. Depending on the architectural layout, it might be communicating with the controller of the integrated equipment, the controller of the equipment's front-end carrier handling module or the factory host.

8.1.4 The IMM is one module of many that receive instructions about a specific substrate. The substrate may be given to the IMM at any point in a sequence of individual process steps. While it typically may be the first or the last in the sequence, or both, it may also appear at any point in the middle.

8.1.5 The IMM is given a substrate, a recipe, and a process job. It may or may not have an alignment station, and it may or may not be able to contain more than one substrate at a time. For example, it might be able to align one substrate while it is executing a process job on a previous substrate. Also, the IMM may or may not have an integrated data analysis subsystem. Modules with data analysis capability may require additional processing time to complete the analysis. Finally, the substrate is unloaded and the results of the measurement are sent on the data port.

9 Communication Ports and Clients

9.1 *Ports*

9.1.1 The IMM shall provide two or more communication ports, one of which to be used for control as well as information and is called the primary port. Other ports are called secondary ports and are mainly used for sending and receiving information to applications that may be either on- or off-tool, such as remote process control and/or fault detection, archiving substrate data tables, uploading and downloading recipes, etc.

9.1.2 Control in this case signifies the request for services that can change the hardware state of the integrated metrology module, including substrate exchange, process jobs, clock services, and maintenance-related activities. In general, the controller of the process tool, or one of its major components, will require the use of the primary port for control, although there may be special circumstances where this is not true.

9.1.3 The IMM shall support one or more additional communication ports to be used mainly for sending and/or receiving information, such as event reports, alarm-related messages, recipes, and data tables. It is recommended that at least one of the additional ports be physically separate to allow a connection to a different network.

9.1.4 The requirement to support a second or multiple ports is a requirement on the IMM side of the communication only. The decision to utilize these ports, and the manner in which they are utilized, is left to the application of the integrated metrology tool and the factory.

9.2 *Clients*

9.2.1 There are two types of clients, the Control Client and the Data Client.

9.2.2 All clients are able to configure the following independently of other clients:

- event reports,
- alarm reports, and

- table retention conditions (¶11.3.4).

9.2.2.1 In addition, each client may request these settings at any time and receive only their own settings in response.

9.2.3 The first message that the IMM receives from a potential client must be the ClientConnect Service request. This message is used to establish a client as either the Control Client or a Data Client. All other service requests from an unestablished client shall be rejected. At any point in time, there shall be no more than one Control Client.

9.2.4 *The Control Client*

9.2.4.1 The Control Client is a client who intends to request services that may change the physical state of the hardware of the IMM. Such services include those used for substrate exchange, process jobs, and any maintenance activities. In general, this will be the controller of the integrated metrology tool or one of its major components. It is up to the implementer whether clients connecting through a port other than the primary port can establish themselves as the Control Client.

9.2.4.2 The Control Client shall be able to use all IMM capabilities specified in this standard. Only the Control Client shall be permitted to use the IMM Clock Services and Process Job services, or to issue service requests that change the state of the hardware.

9.2.4.3 Once a Control Client is established through use of the ClientConnect service, it remains established as the Control Client until it disconnects by sending the ClientDisconnect Service request.

9.2.4.3.1 During the time that a Control Client is established, any requests to be a Control Client shall be rejected, and any requests for services that can change the state of the hardware from other clients shall be rejected.

9.2.4.3.2 Once the Control Client has disconnected, then another client may request to establish itself as the Control Client.

9.2.4.4 The Control Client shall be able to use all IMM capabilities specified in this standard. Only the Control Client shall be permitted to use the IMM Clock Services and Process Job services, or to issue service requests that change the state of the hardware.

9.2.5 *Data Clients*

9.2.5.1 The Data Client is a client who intends to request services for information only and not for any services that may change the physical state of the hardware of the IMM. Each Data Client may request information at any time.

9.2.5.2 The use of the Data Client is optional for the end-user.

9.3 *Bandwidth and Performance*

9.3.1 The IMM is responsible for managing its own bandwidth and performance. This standard allows for delays in message transmission in a multi-client environment in order to protect the integrity of the IMM measurement and data conversion abilities.

10 Integrated Measurement Module Object Model

10.1 *Description*

10.1.1 The Integrated Measurement Module (IMM) conforms to the architectural specification of the EquipmentModule object as defined in SEMI E120 (CEM).

10.1.2 Figure 1 shows an object model for the Integrated Measurement Module and its significant components.

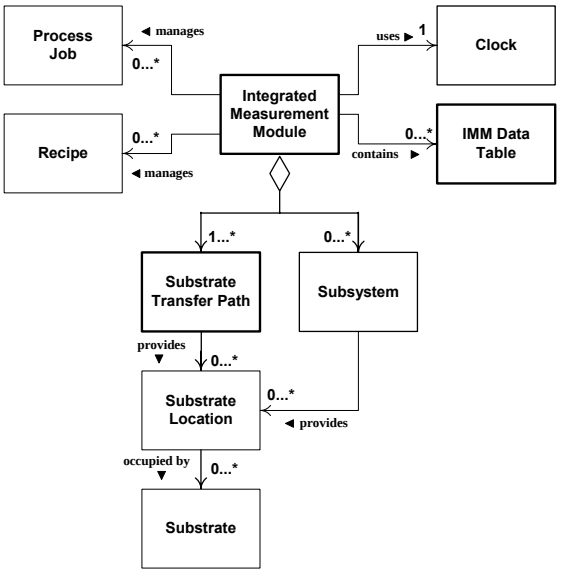

Figure 1
IMM Object Model

10.1.3 The IMM has exactly one internal clock. It must have one or more measurement subsystems capable of holding one or more substrates in a substrate location. It may have additional components, such as OCR subsystems. Pattern recognition and analysis subsystem typically do not hold any material. The Clock shall be provided as an OBEM-compliant clock object. Subsystems shown in Figure 1 are not required to be visible as objects externally.

11 The Integrated Measurement Module Object

11.1 *Relationship of IMM Object And Physical Module*

11.1.1 The IMM object represents the IMM module and all of its internal components to its client. It is responsible for ensuring proper behavior in all cases. In this document, the IMM object is considered synonymous with the IMM module. The IMM provides all services to the client other than those specifically directed to the STPO, including those specified by other SEMI standards.

11.2 *IMM Service State Model*

11.2.1 The IMM is either in a state where it can be used for normal measuring purposes or it is not. This is reflected in the Service State Model as shown in Figure 2.

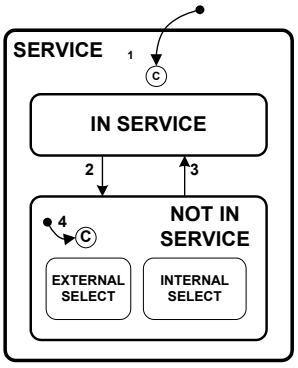

Figure 2
IMM Service State Model

11.2.2 The IMM Service state model shows its availability for process jobs. It has two states, IN SERVICE and NOT IN SERVICE. The IMM shall only accept or initiate process jobs while in the IN SERVICE state.

11.2.3 *State Definitions*

11.2.3.1 *IN SERVICE* — In this state, the IMM is capable of performing all its normal activities for measurement. It is able to accept and execute a process job or may be currently executing a process job. No faults or exceptions exist that would interfere with its capability of loading, measuring, and unloading a substrate.

11.2.3.2 *NOT IN SERVICE* — In this state, the IMM shall not accept new process jobs or initiate execution of existing process jobs. It may be able to perform special services required for maintenance and diagnostics, including calibration and associated loading/unloading activities. Either the external Control Client or the IMM may put it in an NOT IN SERVICE state. Once put into NOT IN SERVICE, only the one who put it NOT IN SERVICE may change the state back to IN SERVICE. This is reflected in the two substates, EXTERNAL SELECT and INTERNAL SELECT.

11.2.3.3 *EXTERNAL SELECT* — The Control Client has placed the IMM out of service.

11.2.3.4 *INTERNAL SELECT* — The IMM has placed itself out of service due to a fault condition.

11.2.4 *Service State Model Transitions* — Transitions are defined for the Service State Model in Table 1.

Table 1 Service State Model Transitions

#	*Previous State*	*Trigger*	*New State*	*Comment*
1	(no state)	IMM initialization.	IN SERVICE or NOT IN SERVICE	Initial state is determined first by existence of exceptions or alam conditions. If none, then initial state shall be the last known state.
2	IN SERVICE	The IMM detects one of the conditions for NOT IN SERVICE OR The IMM client has issued a ChangeService request.	NOT IN SERVICE	Transitions 2 and 4 are combined for reporting purposes.
3	NOT IN SERVICE	The IMM or communications partner has returned the IMM to IN SERVICE.	IN SERVICE	
4	System initialization or IN SERVICE	Entry into NOT IN SERVICE: 4-1 Following system initialization, 4-2 IMM initiated state change, and 4-3 Client requested state change.	EXTERNAL SELECT (trigger 4-3) or INTERNAL SELECT (triggers 4-1 or 4-2)	Transitions 2 and 4 are combined for reporting purposes.

11.2.5 The Control Client may request a state change through a Service request at any time.

11.2.6 A transition to OUT OF SERVICE shall not interrupt any process job currently executing. It is recommended that all transitions to OUT OF SERVICE be delayed until such process jobs have completed, either normally or abnormally. This avoids confusion or an unintended internally-generated abort of process jobs. Process jobs must be aborted separately and deliberately when necessary.

11.3 *Object Attributes*

11.3.1 Table 2 defines the attributes of the IMM Object as shown in Figure 3.

Table 2 IMM Object Attribute Definitions

Attribute Name	*Definition*	*Access*	*Reqd*	*Form*
ObjType	Defines object type.	RO	Y	Text = “EquipmentModule”
ObjID	Identifier of specific module.	RO	Y	Conforms to requirement of ObjID (SEMI E39).

Attribute Name	*Definition*	*Access*	*Reqd*	*Form*
EventList	List of event name/event number pairs.	RO	Y	List of text/integer pairs.
RetentionRules	Retention Rules supported by the IMM.	RO	Y	List of 0+ conditions.
Service	Current status of the Service State Model.	RO	Y	Enumerated: InService NotInService
SubstrateList	A list of zero or more ObjID for all substrates currently within the IMM, regardless of location.	RO	Y	List of 0+ text.
TableCapacity	Indicates maximum number of tables that may be stored.	RO	Y	Integer.
TableCount	Indicates number of tables currently stored.	RO	Y	Integer.
TableStorageAlert	Remaining table storage capacity. Either the number of additional tables that can be stored (integer) or the remaining percentage of total capacity (floating point) available. When remaining capacity reaches this value, the Table Storage Capacity Warning event message is sent.	RO	Y	Integer or floating point

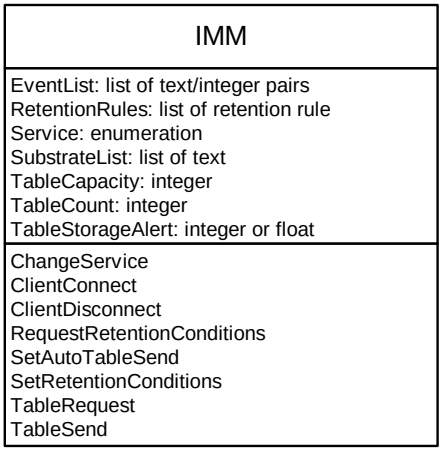

Figure 3
IMM Object

11.3.2 The attribute EventList identifies all of the errors that may be generated by the IMM object. Each such event shall have both a descriptive name and a numeric identifier, each of which shall be unique within the EventList list.

11.3.3 The attribute TableCapacity identifies the maximum number of tables that the IMM is capable of storing. The client may use this information for determining its own retention rules.

11.3.4 *Retention Conditions* — A question of concern to the IMM is the method of determining when a data table is no longer required. Each client is able to select individual conditions for retaining raw data and conversion tables. These conditions are the client-selected RetentionRules.

11.3.4.1 The IMM has a RetentionRules attribute that lists the specific retention rule elements supported by the IMM (see Table 3 for definitions of retention rule elements). An empty (null) list signifies that there are no retention rules that can be met to cause the table to be either deleted or retained.

11.3.4.2 The IMM maintains a list of rules for each client. This is a single level or two level list of rules from the RetentionRules list. This list is available on the RequestRetentionRules request by the client and is not publicly available as an attribute.

11.3.4.2.1 Once all retention rules have been satisfied for a given table for all clients, the IMM may delete the table at its own discretion based on internal requirements that are outside the scope of this standard. The IMM is responsible for managing both its storage space and those tables that do not need to be retained for any client.

11.3.4.2.2 As a single level list, when any item list is satisfied, the table may be deleted (a logical OR condition).

11.3.4.2.3 If one or more conditions need to be satisfied before a table may be deleted (a logical AND condition), the items should appear as a list of RetentionRules that occupy one element of the top level list. For instance, if the retention rule is that conditions A and B both need to be met, or condition C must be met, then the DefaultDataRetention rule is a two element list with one element being a 2 item list of A and B and the other is just condition C:

(A,B), C

11.3.4.2.4 An empty (null) list is equivalent to a "No Retention Rule" condition.

11.3.4.2.5 Actual representation of this list is dependent upon the communication protocol used.

11.3.4.3 Until or unless set by the client, the client's retention rule is a zero element list, signifying that no retention rule has been set.

11.3.4.4 Except when required by automatic rollover (¶11.3.5.2), no IMM Data Table Object shall be deleted until all client retention rules have been satisfied.

11.3.4.5 The definitions of individual standard retention rule elements are provided in Table 3. Additional rules may be defined in the future. All of the individual rules shall be integers, where a non-value indicates that this rule is enabled. Otherwise it is disabled. The default value for each element is zero.

Table 3 Retention Rule Element Definitions

Retention Rule	*Definition*
After Transfer	The table may be deleted after it has been sent to the client. Any non-zero value indicates that this rule is enabled. A value of zero disables it.
Client Delete	If ClientDelete is non-zero, tables are retained until deleted by the client. However, each time the table capacity drops below the value in ClientDelete, the TableStorageCapacityWarning event is generated and sent to the client, so the client can then delete tables no longer needed. If the client fails to delete sufficient tables, then the Automatic Rollover deletes the oldest tables as described below.
Maximum Tables Stored	This rule sets a value for the maximum number of tables that should accumulated before Automatic Rollover takes effect and the oldest table(s) are deleted when there is insufficient room to store a new table. Maximum value for this rule is Table Capacity – 1.
RetentionTime	The value in RetentionTime indicates the maximum length of time, in hours, that a table should be retained in storage. The table may be deleted after this period of time has expired. Any internal timers used are not visible to the client.

11.3.5 *Table Storage Management* — All tables shall be stored in non-volatile storage once they have been completed.

11.3.5.1 *Calibration Table* — The latest instance of Calibration Table shall always be retained in storage.

11.3.5.2 *Automatic Rollover* — In the event that there is not enough storage capacity remaining to store a new table, then the oldest tables of the same type as the new table (for either raw data or converted data) shall be deleted, one at a time, until there is room to store the new table, regardless of the presence or absence of retention rules.

NOTE 4: The size of Raw Data Table Objects and Converted Data Table Objects may be significantly different.

11.3.5.3 TableStorageAlert is a constant determined by the supplier that represents either the approximate number of tables (total) that may still be stored, or a percentage of total remaining storage capacity (TableCapacity). When the remaining capacity drops to the level indicated by TableStorageAlert, a Table Storage Capacity Warning event is sent (¶11.4.5).

11.4 *System Initialization* — Whenever the Integrated Measurement Module is powered on or reset, it goes through a system initialization before it is acknowledges communications with, or establishes, any potential client.

Initialization activities include both software initialization and hardware initialization. On the completion of initialization, the IMM is in a known condition, and the following requirements have been satisfied:

- All state models have been set to their entry state,
- Communications have been initialized,
- The physical status of the IMM is known,
- The absence or presence of substrates is known,
- Exceptions, including all alarm conditions, have been determined,
- No process jobs are defined,
- No clients have been established, and
- All internal equipment substrate locations have been checked for the presence of substrates, and the total count of substrates detected has been compared with the number expected.

11.4.1 *Substrate Count Discrepancy*

11.4.1.1 The determination of expected substrate count requires that sufficient information about actual count must be maintained in non-volatile memory during run-time so that is available to accurately determine expected count during system initialization. For example:

- The content of the IMM attribute SubstrateList is saved in non-volatile memory whenever it is updated, and expected count is determined by the number of valid entries in the list, or
- A list of occupied internal substrate locations is retained in non-volatile memory each time any one is updated, and the expected count is determined by the number of valid entries in the list, or
- A simple count of the number of current substrates is retained in non-volatile memory.

11.4.1.2 If both the expected count and the actual count are zero, and there are no other errors or alarms, then the IMM Service state is IN SERVICE, and the IMM EPT state is IDLE.

11.4.1.3 If both the expected count and the actual count are equal and non-zero, then the IMM Service state is OUT OF SERVICE, and the IMM EPT state is BLOCKED. In this case, the substrates are assigned temporary identifiers of "$unknown_i$", where i = 1,…, number of actual substrates detected. The substrates shall be unloaded in an orderly manner. When all substrates have been unloaded, and all alarms are cleared, the IMM Service state changes to IN SERVICE, and the IMM EPT state changes to IDLE

11.4.1.4 If the expected count is greater than the actual count, then a Missing Substrate alarm is set. If the expected count is less than the actual count, then an Unexpected Substrate alarm is set. In either case, the IMM remains OUT OF SERVICE and BLOCKED until the alarm condition is cleared. The IMM may provide a method for a maintenance technician to determine if the "lost substrate" is actually present or if the "unexpected substrate" is actually present (for example, the physical module may provide a view plate). and to indicate whether or not it is safe to unload. Such methods are beyond the scope of this standard. Note that sensor errors can result in an apparent loss or gain that is not real. Note also that it may be possible for a maintenance technician to interact with the IMM as a "control client".

11.5 *IMM Events and Alarms*

11.5.1 In addition to the special events and alarms described below, all state transitions shall be capable of triggering an event, including all state models required by other standards required by IMMC.

11.5.2 *Alarms* — Various alarms caused by hardware-specific malfunction may occur at any time, including those detected by the Substrate Transfer Path Object (STPO). In addition, alarms may be caused by a discrepancy between actual and expected substrate counts.

11.5.2.1 *Missing Substrate Alarm* — This alarm is set whenever a substrate is no longer detected. The substrate is either broken (and must be manually removed) or a sensor has failed so that it can no longer detect a substrate. This condition shall cause the IMM Service state to set to OUT OF SERVICE and the IMM EPT state to be set to BLOCKED so long as this condition exists. This condition may occur at any time, either during system

initialization or runtime. In either case, the alarm condition shall be cleared before processing continues. See ¶11.4.1.4 for further detail.

11.5.2.2 *Unexpected Substrate Alarm* — This alarm is set during system initialization when substrates have been detected, and the actual substrate count exceeds the number expected. See ¶11.4.1.4 for further detail.

11.5.3 *SubstrateListUpdate* — The SubstrateList attribute is a list of the ObjIDs for all substrates currently present within the IMM module. Whenever a Substrate Object is created or deleted, this attribute is updated and an event message is sent with the ObjID of the Substrate Object associated with the change.and whether it was added or removed.

11.5.4 *Table Storage Capacity Warning* — A notification sent to the client when table capacity decreases below TableStorageAlert and each time the storage further decreases.

11.5.5 *Table Storage Capacity Overflow* — Tables have been removed due to capacity overflow. Note: This is a trigger for Transition 6 in the IMM Data Table state model, ¶14.2.

11.5.6 *Event Variables* — Certain variables shall be provided that are associated with certain events and contain the values that were valid at the time the event occurred. For the IMM object, these variables are specified in Table 4. The names of variables as specified shall be implemented exactly as text strings.

Table 4 Event Variables for IMM Object

Variable Name	*Description*	*Type*	*Event*	*Comment*
DateTime	The date and time that the event occurred.	Conforms to the clock format in SEMI E98.	All	
Service	Current state of Service State Model.	Enumerated IN SERVICE OUT OF SERVICE	All	
SubstrateListChangeType	Indicates the type of change: addition or deletion.	Enumerated Addition Deletion	SubstrateListUpdate	
SubstrateID	The last SubstrateID added to or removed from the SubstrateList attribute.	Text.	SubstrateListUpdate	
TableCount	The number of tables currently stored.	Integer.	TableStorageCapacityWarning, TableStorageCapacityOverflow	

11.6 *IMM Object Services*

11.6.1 Table 5 defines the services specific to the Integrated Measurement Module. Message details are defined in §19.

Table 5 IMM Object Services

Message Service Name	*Description*	*Type*	*Reqd*
ChangeService	Set the IMM in service or out of service.	R	Y
ClientConnect	A request to become a Control Client or Data Client.	R	Y
ClientDisconnect	A request to terminate the Client role.	R	Y
RequestRetentionConditions	A request to send the rules and retention time for this client.	R	Y
SetAutoTableSend	A request to set AutoTableSend on or off for this client.	R	Y
SetRetentionConditions	The client sends a one or two level list of 0+ Retention Rules and an optional minimum retention time.	R	Y
TableRequest	The client requests a specific table, or table subset.	R	Y
TableSend	Used by the client to send a table or delete an existing table. Used by the IMM to request to send a table to the client by the AutoTableSend capability.	R	Y

12 Substrate Transfer Path Object

12.1 *Description*

12.1.1 The Substrate Transfer Path object (STPO) manages the spatial envelope used for loading (receiving) and unloading (sending) a substrate, along with all mechanisms used in effecting a physical transfer. It has one or more Equipment Substrate Location objects for holding substrates after they are received and before they are sent. Each Substrate Location can hold exactly one substrate when it is occupied. Substrate and Substrate Location objects are defined in SEMI E90.

12.1.2 If substrates can be loaded into, and/or unloaded from, more than one substrate location, then each of these locations shall be represented by a separate STPO object. This allows the Control Client to specify where it intends to load or unload.

12.1.3 The STPO object allows the IMM to interact with an external substrate handler to transfer substrates to and from a Substrate Port (a substrate location together with any necessary mechanisms), and also to exchange properties of the Substrate Object being transferred.

12.1.4 A Substrate Transfer has two required parts, which may occur in either order: the substrate transfer and the substrate information transfer. The transfer procedure consists of the messages and physical actions necessary to effect the loading or unloading of a substrate to or from the Substrate Transfer Location. The exchange is not complete until both parts have been successfully completed.

12.1.5 The minimum information required for processing is SubstrateID. Additional information is used to store as context data in the IMM Data Table Objects discussed in §13. The method provided for obtaining the additional information is the same as the method for providing SubstrateID.

12.1.6 The STPO shall provide two ordered lists of the sequence of text micro commands that are needed for loading and for unloading a substrate to and from the Substrate Transfer Location. The Control Client may obtain these lists on demand through the service XfrCmdListRequest. These micro commands are to be returned by the Control Client in the XfrCommand service, one at a time, in the same order given, to effect a load or unload hand-off. Micro commands are defined by the IMM supplier except for two that are required. Examples of micro commands are given in the Load and Unload scenarios in ¶12.11. The two required micro commands are “Confirm presence” and “Confirm Absence”, as specified in ¶12.11.2.3 and ¶12.11.3.3 respectively.

12.2 *Object State Model*

12.2.1 This section defines the Substrate Transfer State Model, which shows the behavior required for coordinating the loading of a substrate to, and unloading from, the substrate location accessed by an external substrate handling mechanism.

12.2.2 Figure 4 shows the diagram for the Substrate Transfer Path State Model.

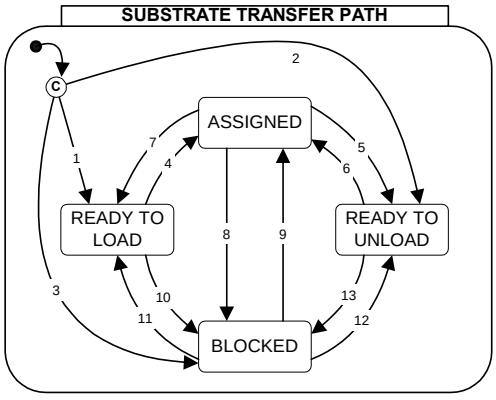

Figure 4
Substrate Transfer Path State Model

12.2.3 *Substrate Transfer States*

12.2.3.1 *ASSIGNED* — A substrate transfer operation, or normal operations (e.g., measurement) or conditions (e.g., calibration required or being performed) prevent substrate transfers to or from the module.

12.2.3.2 *BLOCKED* — Abnormal operations (e.g., abort) or conditions (e.g., faults) prevent substrate transfer to or from the module.

12.2.3.3 *READY TO LOAD* — A substrate is not present on the Substrate Location, and that location is available to be loaded with a substrate in this state. This is the only state in which substrate loads may be initiated.

12.2.3.4 *READY TO UNLOAD* — A substrate is present on the Substrate Location, and the substrate is available to be unloaded from that location in this state. This is the only state in which substrate unloads may be initiated.

12.2.3.5 *SUBSTRATE TRANSFER PATH* — The parent state of the Substrate Transfer Path State Model. This state has four substates, READY TO LOAD, READY TO UNLOAD, ASSIGNED and BLOCKED.

12.2.4 Table 6 defines the transitions of the STPO State Model.

Table 6 Substrate Transfer Path State Model Transitions

#	*Previous State*	*Trigger*	*New State*	*Comment*
1	(no state)	IMM initialization complete AND All conditions allow a substrate to be loaded.	READY TO LOAD	
2	(no state)	IMM initialization complete AND A substrate is present in the Substrate Location that may be unloaded.	READY TO UNLOAD	
3	(no state)	IMM initialization complete AND The substrate transfer location is either configured to be out of service, or a fault condition exists that does not allow substrate transfers.	BLOCKED	
4	READY TO LOAD	A substrate transfer is completed without error.	ASSIGNED	The STPO remains ASSIGNED until the Substrate is either no longer within the STPO and/or measurements have been completed and the substrate can be removed.
5	ASSIGNED	Conditions make the STPO available for a substrate unload operation.	READY TO UNLOAD	The substrate is at an unload location and is ready for removal.
6	READY TO UNLOAD	Normal conditions make the STPO unavailable for substrate unload.	ASSIGNED	Conditions that prevent a substrate unload operation include the termination of an unload operaton, another substrate transfer operation, or another measurement of the same substrate is now required. Note that no direct transition is provided betwen the READY TO LOAD and READY TO UNLOAD states.
7	ASSIGNED	Conditions make the IMM module available for a substrate load operation.	READY TO LOAD	For example, a substrate has moved to a location outside of the STPO, or a substrate removal has been successfully completed.
8	ASSIGNED	A fault condition occurred that prevents substrate transfers.	BLOCKED	
9	BLOCKED	All fault conditions are cleared.	ASSIGNED	
10	READY TO LOAD	A fault condition occurred that prevents substrate transfers.	BLOCKED	

#	Previous State	Trigger	New State	Comment
11	BLOCKED	1. All fault conditions are cleared AND 2. Conditions allow a substrate transfer into the STPO to occur.	READY TO LOAD	The substate load location is unoccupied.
12	BLOCKED	1. All fault conditions are cleared AND 2. Conditions allow a substrate removal from the STPO to occur.	READY TO UNLOAD	The substrate unload location is occupied.
13	READY TO UNLOAD	A fault condition occurred that prevents substrate transfers.	BLOCKED	

12.3 *Object Attributes*

12.3.1 The diagram for the STPO object is shown in Figure 5.

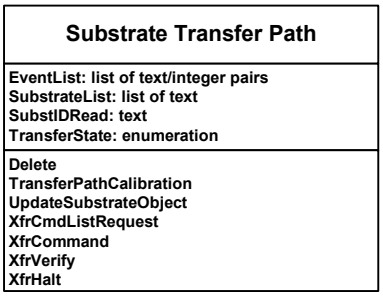

Figure 5
Substrate Transfer Path Object

12.3.2 Table 7 defines the attributes of the STPO object.

12.3.3 The attribute EventList identifies all of the errors that may be generated by the STPO. Each such event shall have both a descriptive name and a numeric identifier, each of which shall be unique within the EventList list.

Table 7 Substrate Transfer Path Object Attribute Definitions

Attribute Name	Definition	Access	Reqd	Form
ObjType	Defines object type.	RO	Y	Text = “SubstrateTransferPath”.
ObjID	Identifier of specific module.	RO	Y	Conforms to requirement of ObjID (SEMI E39).
EventList	List of event name/event number pairs.	RO	Y	List of text/integer pairs.
SubstrateList	A list of zero or more ObjID for substrates within the STPO.	RO	Y	List of 0+ text.
SubstIDRead	Substrate identifier last read by the STPO substrate ID reader.	RO	N	Required only if substrate ID reader supported. May be a null string if reader out of service.
TransferState	Current operational status.	RO	Y	Enumerated: ASSIGNED BLOCKED READY TO LOAD READY TO UNLOAD

12.4 *Object Services*

12.4.1 This section specifies the services provided by the STPO. The client for these services is restricted to the IMM's Control Client due to the physical requirements of the transfer.

12.4.2 Services starting with the prefix "Xfr" are based on similar services specified in SEMI E32 for micro moves that start with the prefix of "HO". The transfer services specified in this document include an object specifier that can be used to differentiate between multiple instances of a STPO.

12.4.3 Table 8 provides an alphabetical list of services required from the STPO. Message details are provided in §18. All services provided by a STPO shall be available to the Control Client and only to the Control Client.

12.4.4 *List of Substrate Transfer Path Services*

Table 8 List of Substrate Transfer Path Services

Operation	*Description*	*Type*	*Reqd*
Delete	The Control Client requests the STPO to delete a Substrate Object for a substrate that is not physically present within the STPO or IMM. This service is defined in SEMI E39.	R	Y
TransferPathCalibration	A request by the Control Client to provide any discrepancy in the actual position of an incoming substrate with the position required by the STPO for loading. Required only if the STPO is able to detect the (x,y,z) position of the center of an incoming substrate.	R	Y
UpdateSubstrateObject	The Control Client informs the STPO about a substrate that is about to be, or has just been, loaded.	R	Y
XfrCmdListRequest	The Control Client requests the ordered list of micro move commands required by the STPO to effect a hand-off.	R	Y
XfrCommand	Message sent by the Control Client to the STPO to request a certain action be performed. The response is sent when the action has been completed or if the action can not be performed at this time.	R	Y
XfrHalt	Message sent by the Control Client to stop all activities related to substrate transfer. This message ends the transfer procedure and requires external intervention to clear.	R	Y
XfrVerify	Message sent by the Control Client to the IMM when all necessary actions have been completed to effect a transfer of a substrate and to ask for confirmation.	R	Y

12.4.5 The service UpdateServiceObject must be supported by the Control Client. Receipt of this service request notifies the Control Client that a specific substate is about to, or just has been, unloaded, and it allows those substrate properties defined by IMMC to be passed to the Control Client with (possibly) updated values.

12.5 *Creation of Substrate Objects*

12.5.1 *Substrate Object Creation* — The UpdateSubstrateObject service, defined by SEMI E90, is sent by the Control Client to the STPO to accomplish three primary objectives:

- Identify the substrate through the ObjID attribute specification in the message,
- Define other attributes of the substrate that it considers important, and
- Prompt the STPO to create the Substrate Object.

12.5.1.1 An UpdateSubstrateObject request that is always assumed to be in connection with the loading of a specific substrate.

12.5.1.2 If the STPO receives this message with an ObjId (Substrate ID), and a Substrate Object with the same identifier already exists, the STPO shall assume that a physical substrate with the same identifier has already been loaded and still exists with the IMM. In this case, the STPO shall deny the request. If the substrate has been loaded, it must be unloaded.

12.5.1.3 Otherwise, the STPO creates the Substrate Object when it accepts the UpdateSubstrateObject request and populates its attributes as specified by the service.

12.5.1.4 The UpdateSubstrateObject object may only be sent and accepted by the STPO and not the IMM or other objects.

12.5.1.5 This message may be sent by the Control Client either immediately before the physical transfer is initiated or immediately following the successful completion of this transfer. When this message is received, a Substrate Object is created and its attributes populated with the attributes as specified by the service.

12.5.1.6 If a Subtrate ID reader is available, then the Substrate ID may be read before the UpdateSubstrateObject message is received.

12.6 *Substate Verification*

12.6.1 This section only applies to those implementations that provide a Substrate ID reader.

12.6.2 Whenever a Substrate ID is read, the ID as read is saved in the STPO attribute SubstIDRead, and a SubstrateIDRead event is triggered. No Substrate Object is created by this event. The STPO must wait until the UpdateSubstrateObject message is received before creating a Substrate Object with the expected ObjID.

12.6.3 *Verification Success* — If the ID as read is identical to the value sent in the UpdateSubstrateObject message as the substrate's ObjID, then the verification is successful.

12.6.3.1 If the ObjID received is a SEMI E90 compliant ID of "CarrierID" + "." + "SlotNumber", the STPO will consider the verification to be successful.

12.6.4 The value stored in the STPO SubstIDRead attribute is stored in the Substrate Object's SubstrIDRead attribute. The substrate is now ready to continue normally and is available for processing.

12.6.5 *Verification Failure* — If the ID as read is different from the ObjID received, and the ObjID received is not in the form of the SEMI E90 default Substate ID, the following actions are required:

12.6.5.1 The STPO shall immediately send an Unexpected ID error message to notify the control client that it has received a substrate with an unexpected identifier.

12.6.5.2 In addition, the STPO shall set an Unexpected ID alarm. This alarm shall remain set until either the unexpected substrate is unloaded or a SEMI E39 Delete message for the expected substrate object, followed by a new UpdateSubstrateObject message for the actual substrate, are received.

12.6.5.2.1 The STPO shall go immediately to the Ready to Unload state and wait for the either services for unloading or a Delete request, followed by a new UpdateSubstrateObject request.

12.6.5.2.2 When the Control Client receives the error message, it has to determine if the substrate that was transferred should be processed or unloaded. If it is to be processed, then it will need to ask the STPO to delete the Substrate Object with the original ID and send another UpdateSubstrateObject message with the ID of the substrate that was just transferred.

12.6.6 When the Substrate ID has been verified (or, in the case where there is no ID reader present, simply accepted), at some point it may leave the STPO's management and be passed to the IMM Object for processing.

12.6.7 *Substrate Object Deletion*

12.6.7.1 The STPO shall only accept requests to delete a Substrate Object when it has no physical substrate that corresponds to it. See ¶12.6.6. When the STPO has created a substrate object and has no physical substrate to which that object corresponds, and the STPO is in either the READY TO UNLOAD or the BLOCKED state, the Control Client may delete the erroneous Substrate Object with a SEMI E39 Delete message. If the STPO is then cleared of conditions that may have placed it in the BLOCKED state, it shall return to the READY TO UNLOAD

state to allow the physical substrate that it did load to be removed or to receive a new UpdateSubstrateObject message.

12.6.7.2 Any request by a client to delete a Substrate Object for a substrate physically located with the IMM in general shall be denied.

12.7 *Substrate Orientation*

12.7.1 When a substrate is delivered to or picked up from the IMM's STPO, the substrate has a certain orientation with respect to the position of the fiducial relative to the entry into the STPO, as illustrated in Figure 6.

12.7.2 The Control Client is able to specify both the incoming orientation of a substrate being loaded into the STPO and the desired outgoing orientation by passing values for the OrientationIn attribute and/or the OrientationOut attribute in the UpdateSubstrateObject service.

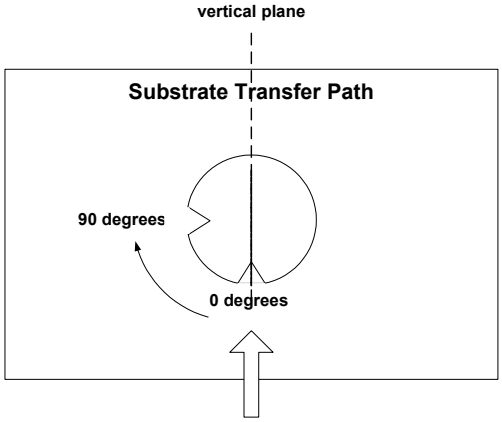

Figure 6
Illustration of Orientation

12.7.3 If the IMM is not capable of determining orientation, either within the STPO or any other of it's components, and if a value has been set in OrientationIn, the value in OrientationOut shall be set to the same value that was provided in OrientationIn.

12.7.4 Otherwise, the following rules shall apply:

- If neither OrientationIn nor OrientationOut are provided by the Control Client, then the IMM shall return the substrate in the same orientation as it was received.
- If OrientationIn is provided, and if the actual orientation is determined to be different, then the "OrientationError" error message shall be sent with the value in the event variable "ActualOrientation" set to the value as determined.
- If OrientationOut is provided, then the substrate shall be oriented in such a way that it is unloaded in the orientation specified.

12.8 *Transfer Path Calibration*

12.8.1 TransferPathCalibration for the STPO is a special service to negotiate adjustments in the arm of the robot of the Control Client. If the STPO has sensors that can detect the (x,y,z) location of the center of the substrate as it enters into the STPO, it reports any differences in (x,y,z) in the TransferPathCalibration service response. Through a series of TransferPathCalibration requests and response, the robot can make adjustments until it is able to place the substrate into the correct location. Figure 7 illustrates how the differences are calculated.

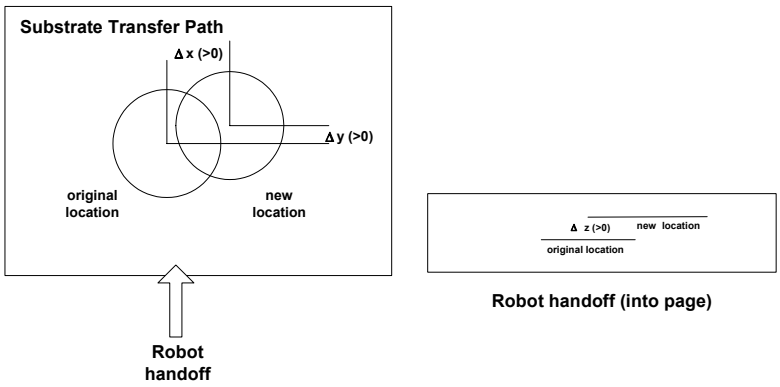

Figure 7
Illustration of (X,Y,Z) Calculations

12.9 *Substrate Transfer Path Events, Errors, and Alarms*

12.9.1 Hardware failures shall be reported as alarms.

12.9.2 Each transition of the STPO State Model (STPSM) defined in Table 6 represents a required reportable event.

12.9.3 For implementations with an available Substrate ID Reader, there are 2 additional events and one alarm:

- The SubstIDRead event occurs when the ID of an incoming substrate is read, successfully or not.
- An Unexpected ID error event and an Unexpected ID alarm occur when substrate verification fails.

12.9.4 For implementations able to determine Substrate Orientation, an OrientationError error message is triggered whenever the orientation of a substrate being loaded is different from the expected orientation provided in the OrientationIn attribute of the substrate.

12.9.5 A Missing Substrate alarm occurs whenever a substrate previously present within the STPO can no longer be detected. This can occur either because the substrate is broken or because of sensor failure In some cases, it may be possible for a maintenance technician to verify that the substrate is present and undamaged, and the failure is due to sensor failure to override this alarm. Otherwise, loading and unloading of substrates is prohibited until maintenance has been performed to remove the broken substrate. The IMM Service state is set to OUT OF SERVICE, and the IMM EPT state is set to BLOCKED.

12.10 *Substrate Transfer Path Event Variables*

12.10.1 Table 9 shows the variables that are required for STPO events. These variables are valid only at the time of the event with which they are associated.

Table 9 Event Variables for Substrate Transfer Path Object

Variable Name	*Description*	*Type*	*Event*	*Comment*
DateTime	The date and time that the event occurred.	Conforms to the clock format in SEMI E98.	All.	
SubstrXfrPathID	Object ID of STPO.	Text.	All transitions of Substrate Transfer Path Object State Model (STPOSM).	Not required if only one path exists.
SubstrateList	Current list of substrate IDs for substrates in the transfer path.	List of 0+ text.	All transitions of Substrate Transfer Path Object State Model (STPOSM).	Empty if no Substrates are in the transfer path.
SubstIDRead	The value of the substrate ID as read.	Text.	SubstIDRead, UnexpectedSubstrate.	Required if a Substrate ID Reader is available.

Variable Name	*Description*	*Type*	*Event*	*Comment*
OrientationIn	The value provided by the Control Client.	Floating point.	OrientationError.	Required if the Orientation capability is implemented.
Orientation	The actual orientation, as determined by the STPO, for an incoming substrate.	Floating point.	OrientationError.	Required if the Orientation capability is implemented.

[#1] The SubstIDRead variable is in addition to the SubstIDRead attribute in the STPO, Substrate, and DataTable Objects. In this case, it is based on SubstIDRead attribute of the STPO Object that generated the event.

12.11 *Substrate Transfer Path Scenarios*

12.11.1 This section provides examples of representative load and unload sequence to illustrate a typical sequence of messages and actions. Related requirements are indicated.

12.11.2 *Load Scenario With No Errors*

12.11.2.1 *Overview* — The Load Scenario in Table 10 illustrates a typical flow of messages related to loading a substrate. Other messages may occur between the Control Client and the IMM objects that are not related.

12.11.2.2 *Assumptions* — The following assumptions about the status of the IMM and the STPO at the start of this scenario:

- The IMM has completed its system initialization and is in the IN SERVICE state,
- The Control Client has been established and is in control of the substrate handler,
- The Substrate Location where the Substrate is to be loaded is in the UNOCCUPIED state,
- The STPO is in the READY TO LOAD state, and
- The ObjID of the STPO object is “STPO1”.

12.11.2.3 *Requirements* — The following requirements for loading are illustrated in Table 10.

- All services listed in Table 8 except for TransferPathCalibration,
- Support for micro command “Confirm Presence”, and
- Support for state model and events associated with state model transitions.

12.11.2.4 *Message Flow* — Table 10 shows an example of the typical flow of messages used to load a substrate into the IMM. Parameter values are examples only. Note that any particular STPO can only load one substrate at a time.

Table 10 Substrate Transfer Path Load Scenario

	Comments	*Control Client (CC)*	*Msg Dir*	*STPO*	*Comments*
1	The CC must obtain this list in order to load a substrate into the IMM. Typically it is one of the first thing following the establishment of the control client and status synchronization.	XfrCmdList-Request.req	→		
	Required field.	XfrType = “Load”			
			←	XfrCmdList-Request.req	
				List of transfer commands shall always include, at a minimum, the command “Confirm Presence”.	Required.

	Comments	*Control Client (CC)*	*Msg Dir*	*STPO*	*Comments*
				List of micro commands in the order in which they must be executed.	Example: Open door, raise pins, substrate confirm presence, lower pins, close door.
2a	This is a required message but may occur immediately after the transfer. This message must be sent before any processing can occur. Multiple messages might be sent for the same substrate.	UpdateSubstrateObject.req	→		Whenever the CC sends this request, the IMM can assume that it is in conjunction with a substrate transfer that either has occurred or is about to occur.
	These are required fields. Not all integrated metrology tools can identify orientation.	SubstID = “xyz” OrientationIn =			If substrate orientation can not be detected, then OrientationOut = OrientationIn.
2b			←	UpdateSubstrateObject.rsp	Check for existing Substrate Object for this ID. If it does not already exist, then it is created here with the attributes as sent. Otherwise, the attributes are updated with the new values.
3	Ready to place. The number of XfrCommand transactions that occur depends on the implementation.	XfrCommand.req	→		
	Indicated which fields are required. The values for XfrCmdName depend on the implementation.	ObjID = “STPO1”, XfrLink = 1, XfrCmdName = “Open door”			Note: this is an example only.
					Opens door.
3b			←	XfrCommand.rsp	Note: if command fails, transfer should be aborted.
				XfrStatus = successful completion, no errors.	
4			←	Event.nfy	STPOSM T5: READY TO LOAD -> Assigned
	Substrate Handler extends arm into IMM.				
5a		XfrCommand.req	→		
		ObjID = “STPO1”, XfrLink = 1, XfrCmdName = “Raise pins”			
					STPO raises pins.
5b			←	XfrCommand.rsp	
				XfrStatus = normal completion, no error.	

	Comments	*Control Client (CC)*	*Msg Dir*	*STPO*	*Comments*
	Substrate Handler lowers substrate onto pins.				
6a		XfrCommand.req	→		
	This XfrCmdName is required and shall be sent prior to XfrVerify.	ObjID = “STPO1”, XfrLink = 1, XfrCmdName = “Confirm presence”			
	If substrate presence is not confirmed, then the transfer shall either be retried or aborted.				STPO verifies that the substrate is detected.
6b			←	XfrCommand.rsp	
				XfrStatus = normal completion, no error.	
	Substrate Handler releases substrate, raises its arm, retracts arm from IMM.				
			←	Event.nfy	The Substrate Location where substrates are loaded changes from UNOCCUPIED to OCCUPIED (SLOSM T1).
7a		XfrCommand.req	→		
		ObjID = “STPO1”, XfrLink = 1, XfrCmdName = “Lower pins”			
					STPO lowers pins.
7b			←	XfrCommand.rsp	
				XfrStatus = normal completion, no error.	
8a		XfrCommand.req	→		
		ObjID = “STPO1”, XfrLink = 1, XfrCmdName = “Close door”			
					Door closed.
8b			←	XfrCommand.rsp	
				XfrStatus = normal completion, no error.	
9a	Confirm transfer is complete.	XfrVerify.req	→		
		XfrLink = 1			
9b			←	XfrVerify.rsp	Acknowledge transfer is complete.
				XfrStatus = normal completion, no errors.	

12.11.3 *Unload Scenario With No Errors*

12.11.3.1 *Overview* — The Unload Scenario in Table 11 illustrates a typical flow of messages related to unloading a substrate. Other messages may occur between the Control Client and the IMM objects that are not related.

12.11.3.2 *Assumptions* — The following assumptions about the status of the IMM and the STPO at the start of this scenario:

- The IMM has completed its system initialization and is in the IN SERVICE state,
- The Control Client has been established and is in control of the substrate handler,
- The STPO is in the READY TO UNLOAD state,
- The Substrate Location used for unloading is in the OCCUPIED state, and
- The ObjID of the STPO object is “STPO1”.

12.11.3.3 *Requirements* — The following requirements for loading are illustrated in Table 11.

- All services listed in Table 8 except for TransferPathCalibration,
- Support for micro command “Confirm absence”, and
- Support for state model and events associated with state model transitions.

12.11.3.4 *Message Flow* — Table 11 shows an example of the typical flow of messages used to unload a substrate from the IMM. Note that any particular STPO can only unload one substrate at a time.

Table 11 Substrate Transfer Path Unload Scenario

	Comments	*Control Client (CC)*	*Msg Dir*	*STPO*	*Comments*
1	The CC must obtain this list in order to unload a substrate from the IMM. Typically it is one of the first thing following the establishment of the control client and status synchronization.	XfrCmdList-Request.req	→		
	Required field.	XfrType = “Unload”			
			←	XfrCmdList-Request.req	
				List of micro commands shall always include, at a minimum, the command “Confirm Absence”.	Required.
				List of micro commands in the order in which they must be executed.	Example: Open door, raise pins, confirm absence, lower pins, close door.
2a			←	UpdateSubstrateObject.req	Updated information on substrate.
	Check for other required fields. The CC needs to accept attribute names and values defined in this standard.			SubstID = “xyz” OrientationOut = xxx	These are required fields.
2b		UpdateSubstrateObject.rsp	←		
3	Ready to pick up.	XfrCommand.req	→		
	Indicate which fields are required.	ObjID = “STPO1”, XfrLink = 1, XfrCmdName = “Open door”			Note: this is an example only.
4			←	Event.nfy	STPOSM T5: READY TO UNLOAD -> Assigned
					Opens door.

	Comments	*Control Client (CC)*	*Msg Dir*	*STPO*	*Comments*
3b			←	XfrCommand.rsp	Note: if command fails, transfer should be aborted.
				XfrStatus = successful completion, no errors.	
	Substrate Handler extends arm into IMM.				
5a		XfrCommand.req	→		
		ObjID = “STPO1”, XfrLink = 1, XfrCmdName = “Raise pins”			
					STPO raises pins.
5b			←	XfrCommand.req	
				XfrStatus = normal completion, no errors.	
	Substrate Handler grips substrate and moves up.				
7a		XfrCommand.req	→		
		ObjID = “STPO1”, XfrLink = 1, XfrCmdName = “Lower pins”			
					STPO lowers pins.
7b			←	XfrCommand.rsp	
				XfrStatus = normal completion, no errors.	
8a		XfrCommand.req	→		
	This XfrCmdName is required and shall be sent prior to XfrVerify.	ObjID = “STPO1”, XfrLink = 1, XfrCmdName = “Confirm absence”			
	If substrate absence is not confirmed, then the transfer shall either be retried or aborted.				STPO verifies that the substrate is no longer detected.
8b			←	XfrCommand.rsp	
				XfrStatus = normal completion, no errors.	
9			←	Event.nfy	The Substrate Location where substrates are loaded changes from OCCUPIED to UNOCCUPIED (SLOSM T2).
	Substrate Handler retracts arm from IMM.				
10a		XfrCommand.req	→		
		ObjID = “STPO1”, XfrLink = 1, XfrCmdName = “Close door”			
					Door closed.

	Comments	*Control Client (CC)*	*Msg Dir*	*STPO*	*Comments*
10b			←	XfrCommand.rsp	
				XfrStatus = normal completion, no errors.	
11a	Confirm transfer is complete.	XfrVerify.req	→		
		XfrLink = 1			
11b			←	XfrVerify.rsp	Acknowledge transfer is complete.
				XfrStatus = normal completion, no errors.	

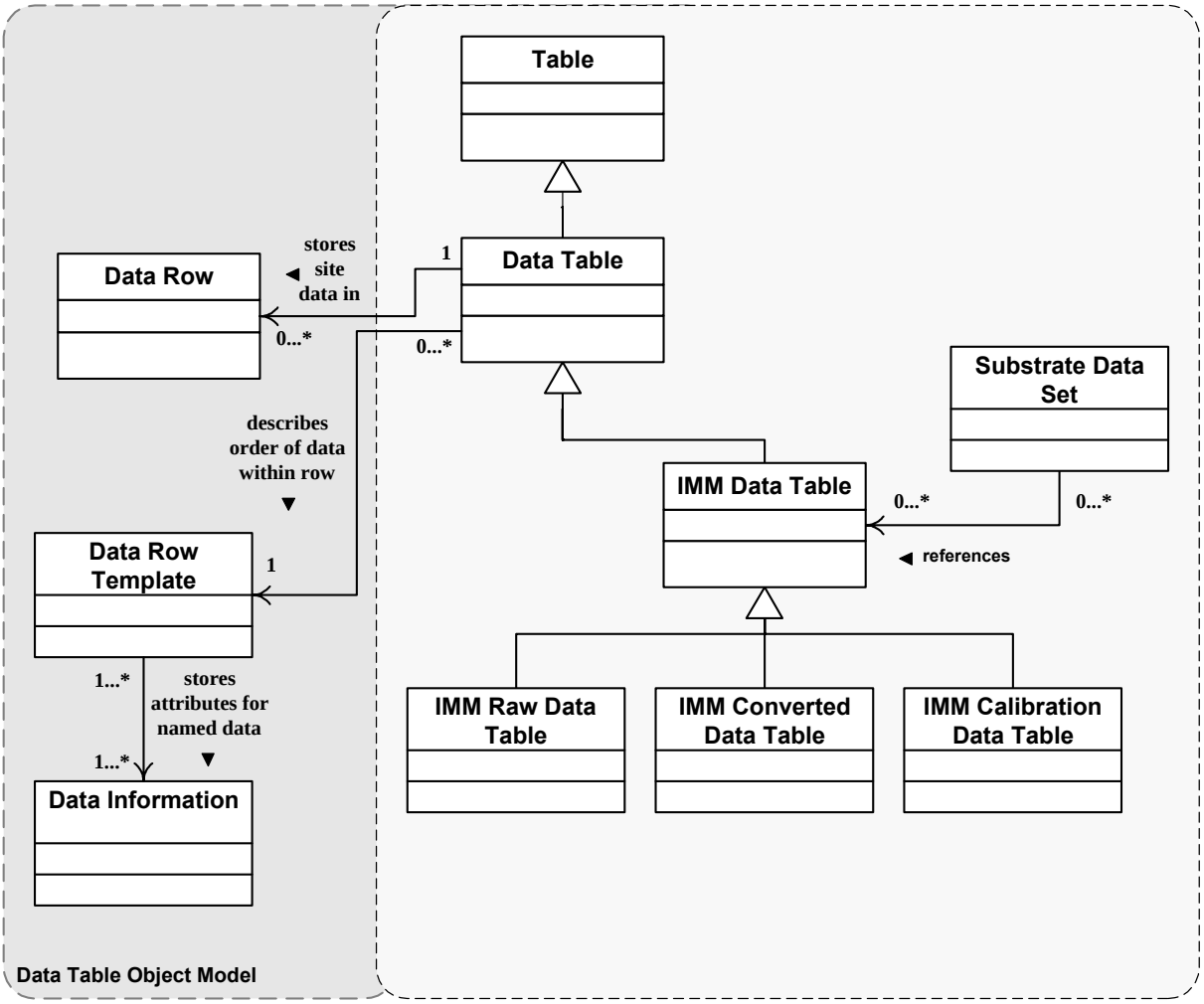

Figure 8
Full Data Object Model

13 IMM Data

13.1 *Overview*

13.1.1 IMM Data consists of substrate context information together with one of the following: (1) measurements taken on a substrate and stored as raw data, (2) meaningful results based on measurements, or (3) substrate

calibration results based on measuring a substrate with expected values. Internal storage of IMM Data is implementation-specific. Its apparent structure through the communication link is that of a type of table object. Table attributes contain context information about the substrate, primarily collected from the Substrate Object measured, while the data provided by the IMM is stored as ordered rows, where each row represents a single site where measurements were taken.

13.1.2 There are two primary aspects to IMM Data, the data itself that provides the content of a table, and the structure of the table as it appears using SEMI E39 object services.

13.1.3 Actual data table instances store the raw data collected during measurement, the results based upon that raw data, or the results of calibrating to a known good substrate. This is reflected in the subtypes of the Data Table object shown in Figures 9 and 10 below.

13.1.4 The second aspect is the structure of the Data Table Object, which is visible through use of templates. These templates allow a user to discover the types of data available from the IMM and the format of that data (signed or unsigned integer, floating point, text, etc.) without having to upload an actual table.

13.1.5 Data Table Objects and IMM Data Table Objects are abstract objects. All implementations of tables in IMM shall be of a subtype of IMM Data Table Objects.

13.2 *Tables*

13.2.1 Table objects are defined in SEMI E58, §10 and §13. A Data Table Object is a type of Table object that provides templates for individual data entries and for rows of entries. This allows a client that needs to use the data provided in a data table to determine the information available in a table before an actual table is transferred.

13.2.2 All objects shown in Figures 5, 6, and 8 are used for purposes of exchanging data over the communication link only. Data does not need to be stored internally in any particular manner.

13.2.3 *Object Ownership* — The IMM is considered to own both the tables that it stores and the set of Data Row Templates. A specific instance of IMM table is considered to own its Data Rows and the associated Data Row Template for that table. This allows the client to query a specific table for its template.

13.2.3.1 Ownership, as defined in SEMI E39, is not necessarily exclusive. An owner is expected to know about the types of objects that it owns, their attributes, and their services.

14 IMM Data Table Object

14.1 *Description*

14.1.1 The IMM Data Object is a type of Data Table Object and inherits its attributes and relationships with templates shown in Figures 8 and 11. The Data Table Object and its templates are discussed in §16. This section specifies the IMM Data Object and its subtypes. Figure 9 shows the model for a Data Object.

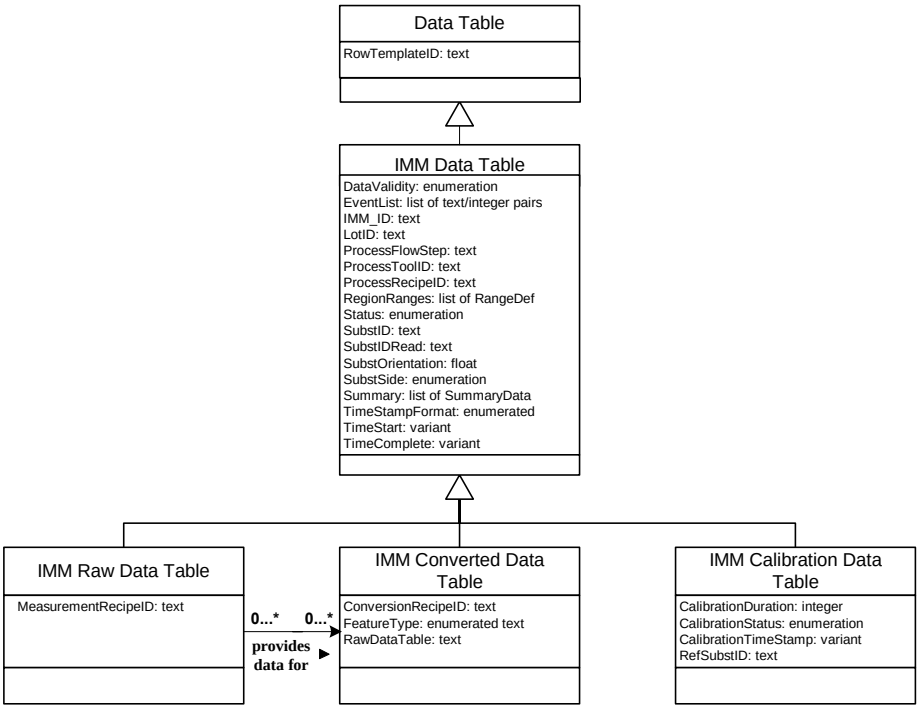

Figure 9
Data Table Object Hierarchy

14.2 *IMM Data Table Object State Model*

14.2.1 The IMM Data Table Object has a state model showing when a new instance of data object is created, when it is finished with the current set of data, and when it has been deleted after all current retention rules have been satisfied. The diagram of the state model is shown in Figure 10. Table data is available only for complete data rows (following transitions 2, 3, and 4).

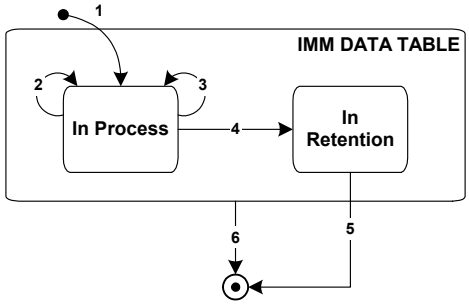

Figure 10
IMM Data Table Object State Model

14.2.2 *State Definitions*

14.2.2.1 The IMM Data Table Object has two states: IN PROCESS, and IN RETENTION.

14.2.2.1.1 IN PROCESS — An IMM Data Table Object is IN PROCESS as long as more data is to be stored in the data object.

14.2.2.2 IN RETENTION — This state is active when the IMM Data Table Object is complete and conditions for ending retention have not been satisfied.

14.2.3 Table 12 shows the transitions for the IMM Data Table Object state model.

Table 12 IMM Data Table Object Transitions

#	*Previous State*	*Trigger*	*New State*	*Actions*	*Comments*
1	(no state)	A measurement data table object is created.	IN PROCESS		The conversion recipe name must be known at creation of the object.
2	IN PROCESS	All data for one location has been stored. Note: This is a required transition for all rows, from the first to the last.	IN PROCESS	Site Measurement Data is available for conversion.	All data for the newly completed row must be available for an event report.
3	IN PROCESS	All data for a specified group of sites has been stored. Note: If this transition is supported, it is required for every such transition, including the last, prior to transition 4.	IN PROCESS	Measurement Data available for a site group per recipe specifications.	Optional transition. Implementation-specific. Region should be defined in the measurement recipe.
4	IN PROCESS	Data Table Object complete. Retention conditions exist.	IN RETENTION	Monitor conditions for retention until satisfied.	Data Table Object may be required internally or transferred to tool controller or remote host. All summary data for the table shall be available for a Table Complete Event report. If the automatic send feature is enabled, the table is sent to the designated client.
5	IN RETENTION	All client and internal retention conditions are satisfied and the table is removed from IMM storage.	(extinction)	Event notification required.	Normal exit.
6	IMM DATA TABLE	Abnormal conditions cause the Data Table object to be discarded, or to be removed from IMM storage due to Automatic Rollover.	(extinction)	Event notification required.	Table is not available due to abnormal circumstances, including forced deletion. Retention rules have not been satisfied.

14.3 *IMM Data Table Object Attributes*

14.3.1 Table 13 defines the attributes of the abstract IMM Data Table Object. Most of the information about the substrate that is maintained to be able to keep with the data is defined here. When a type of IMM_Data Table Object is first created, it has no row data, but it does have a RowDataTemplate that it will use.

14.3.2 All three subtypes of the IMM Data Table Object, or the IMM Data Table Object itself, may need additional implementation-specific attributes for summary data that applies to the substrate rather than to a specific location on the substrate. This could include totals, averages, mean, standard deviation, and similar calculations. Use of this type of data is encouraged and in some cases may be important for advanced process control.

14.3.3 GetAttr and SetAttr services may be directed to IMM Data Table objects. In this case, all subtypes of the IMM Data Table Object may be targets of the service request.

Table 13 IMM Data Table Object Attribute Definitions

Attribute Name	*Definition*	*Access*	*Reqd*	*Form*
DataValidity	Qualifies the “goodness”.	RO	Y	Enumerated: Valid and complete, incomplete, out-of-spec data.
EventList	List of event name/event number pairs.	RO	Y	List of text/integer pairs.
IMM_ID	ObjID of IMM object where measured.	RO	Y	Text.
LotID	Lot identifier.	RO	Y	Text, shall conform to SEMI E39 restrictions on ObjID.

Attribute Name	*Definition*	*Access*	*Reqd*	*Form*
ObjID	Object ID.	RO	Y	Text string assigned by the IMM. Shall conform in conformance with SEMI E39 restrictions on ObjID.
ObjType	Type of Object.	RO	Y	Text = “TableIMM_Data”
ProcessFlowStep	Identifies exposure-level step in product fabrication process.	RO	Y	User-supplied text.
ProcessRecipeID	Identifies recipe of previous process.	RO	Y	User-supplied text.
ProcessToolID	Name of last process tool or ObjID of last tool process module.	RO	Y	Text, shall conform to SEMI E39 restrictions on ObjID.
RegionRanges	Identifies first and last row for regions. Required if regions supported.	RO	N	Ordered nested list of 0+ structures of RegionDef. Empty list if no regions have been defined.
Status	Current state.	RO	Y	Enumerated: IN PROCESS IN RETENTION
SubstID	Substrate identifier.	RO	Y	Text, shall conform to SEMI E39 restrictions on ObjID.
SubstIDRead	Substrate identifier as determined from Substrate ID reader in IMM.	RO	N	Required only if the IMM has a Substrate ID Reader.
SubstOrientation	Specifies angle of rotation in radians.	RO	Y	Floating point.
SubstSide	Frontside, backside, or both.	RO	Y	Enumerated: Front Back Both Unknown
Summary	Provides summary data for the substrate.	RO	Y	List of 0+ SummaryData structures.
TimeComplete	Date/time when row data for the table was completed.	RO	Y	Depends on TimeStampFormat.
TimeStart	Date/time when row data entry for the table was started.	RO	Y	Depends of TimeStampFormat.
TimeStamp Format	Indicates format of timestamp.	RW	Y	Conforms to SEMI E98 Clock TimeStampFormat Enumerated

14.3.4 The attribute EventList identifies all of the errors that may be generated by the IMM Data Table object and its subtypes. Each such event shall have both a descriptive name and a numeric identifier, each of which shall be unique within the EventList list.

14.3.5 *Elements of Complex Attributes*

14.3.5.1 A complex form is either a list, an array, or a structure. A complex attribute is an attribute with a complex form for its value. An element of a complex attribute is a list element, an array element, or a structure element. For a complex attribute, its individual elements must be defined separately, because they themselves are not individually attributes and can not be accessed individually through SEMI E39 Object Services.

14.3.5.2 Elements of complex attributes of the IMM Data Table Object are defined in Table 14.

Table 14 Attribute Elements for IMM Data Table Object

Attribute Name	*Definition*	*Access*	*Reqd*	*Form*
RegionDef	Defines a region where a sequence of rows represent measurements within that region.	RO	Y	Structure consisting of RegionID FirstRowID FirstRowNum LastRowNum
RegionID	A text string that uniquely defines a region. May be recipe-defined.	RO	Y	Text. For tables containing measurements from both sides of a substrate, the text shall begin with either "Front" or "Back". Additional text may be added to represent regions within the front or back side.
FirstRowID	Row identifier for 1st row in the region.	RO	Y	Any simple form except floating point.
FirstRowNum	The sequence number of the first row within the region as it appears within a table. The first row in the table has a sequence number of 1, the second row has a sequence number of 2, etc.	RO	Y	Positive integer.
LastRowNum	The sequence number of the last row within the region as it appears within a table.	RO	Y	Positive integer.
SummaryData	Name and value of a specific summary element.	RO	Y	Structure consisting of SummaryDataName SummaryDataValue SummaryDataForm SummaryDataUnits.
SummaryDataName	Name of a specific element of summary data.	RO	Y	Text.
SummaryDataValue	Actual value of SummaryDataItem.	RO	Y	Any valid form of data.
SummaryDataForm	Identifies the data form of SummaryDataValue.	RO	Y	Enumerated per SEMI E39 Data Form definitions.
SummaryDataUnits	Identifies the units, if any, in which SummaryDataValue is expressed.	RO	Y	Enumerated text per SEMI E5, §9.

14.3.5.3 *RegionRanges* — A region is a group or set of site measurement and can be represented within a table by a sequential set of rows. To describe a region, it is only necessary to define these rows. Completion of measurements with a region corresponds to transition 3 in the IMM Data Table Object State Model.

14.3.5.3.1 *RegionID* — The identifier of a region is required for tables containing measurements made at both sides of a substrate. In addition, it is useful for other applications to be able to quickly identify the region of primary interest. In the former case, standard text strings are required. In the second case, identifiers are typically defined by the measurement recipe, but this is not a requirement.

14.3.5.3.2 *FirstRowID* — A SEMI E58-compliant table requires the first entry within a row to be unique among all first row entries within the table. In addition, it must have a simple form (numeric, text, or Boolean) and can not be floating point. This requirement allows the first entry to be used as a row identifier. Using the row identifier FirstRowID within RegionDef provides a double check for the client when counting rows.

14.3.5.3.3 *FirstRowNum and LastRowNum* — All rows appear in a certain sequence within a table. There is a first row, a second row, etc. In a table with n rows of data, the last row would be the nth row. In a generic SEMI E58 table, rows are not required to be ordered, but for an IMM Data Table Object, rows appear in exactly the same sequence as measurements were made. Providing the row numbers of the first and last rows allow an external application to count rows quickly. Verification of the counting accuracy occurs through the provision of the FirstRowID. If the LastRowNum of one region is i, then the FirstRowNum of the FirstRow of the following region will have a FirstRowNum value of i + 1.

14.3.5.4 *SummaryData* — The content of a row represents a set of measurements, and properties of those measurements, taken at a single site. SummaryData is information that summarizes the row content, typically statistically. This is often the information examined first by applications such as Advanced Process Control.

14.3.5.4.1 Examples of SummaryData include:

- <"MeanThickness", .002, "float", "mm">
- <"StdDevThickness", .0005, "float", "mm">

14.4 *IMM Data Table Object Event Variables*

14.4.1 Table 15 specifies variables in alphabetical order that shall be available for events associated with all subtypes of IMM Data Table Object. The Event column identifies any specific state model associated with the variable.

Table 15 Event Variables for IMM Data Table Objects

Variable Name	*Description*	*Type*	*Event*	*Comment*
$DataItem_i$	Column data for I = 1,n.	Dependent on specific column format.	IMMDTOSM:T2.	Actual data from row, all n columns, for transition 2 only.
DateTime	The date and time that the event occurred.	Conforms to the clock format in SEMI E98.	All.	
GroupNum	Identifies the group associated with the event.	Unsigned integer.	IMMDTOSM:T3.	For transition 3 only.
IMM_ID	ObjID of this IMM.	Text.	All transitions of IMMDTOSM.	
LotID	Value of LotID for the table associated with the event.	Text.	All.	
RecipeID	Recipe identifier for the recipe being used for measurement, conversion, or substrate calibration.	Text.	All transitions of IMMDTOSM.	
RowNum	Identifies the row associated with the event.	Unsigned integer.	IMMDTOSM:T2.	For transition 2 only.
SubstID	Value of SubstID for the table associated with the event.	Text.	All.	Identifies the associated substrate.
SubstIDRead	Value of SubstIDRead for the table associated with the event.	Text.	All.	Identifies the substrate ID value as read. SubstID may be in the form of a SEMI E90 default ID.
TableID	ObjID of the table associated with the event.	Text.	All transitions of IMMDTOSM.	Identifies the table with the event.
TableType	ObjType of the table associated with the event.	Text.	All transitions of IMMDTOSM.	Subtype of IMM Data Table object.

14.5 *Types of IMM Data Table Objects*

14.5.1 All of the data generated by both the physical measuring process and a software data conversion process is placed into an IMM Data Table subtype. As shown in Figures 8 and 9, the IMM Data Table object has three concrete subtypes, the IMM Calibration Data Table, IMM Raw Data Table, and IMM Converted Data Table objects. All IMM Data Table objects shall be one of these three subtypes.

14.5.1.1 A limited set of required information shall be provided in row data:

- site location where measurement was made (x,y coordinates, or x,y,z coordinates).
- substrate side measured, where this varies from one set of sites to another within one table.

14.5.2 All subtypes of the IMM Data Table inherit the attributes and state model of the IMM Data Table object. Additional attributes are also defined for each subtype in the following sections.

14.5.3 *IMM Raw Data Table Object*

14.5.3.1 The IMM Raw Data Table object holds the sensor-based measurements collected during the physical measurement process. Table 16 defines the attributes for the IMM Raw Data object.

Table 16 IMM Raw Data Table Object Attribute Definitions

Attribute Name	*Definition*	*Access*	*Reqd*	*Form*
ObjID	Object ID.	RO	Y	String assigned by IMM.
ObjType	Type of Object.	RO	Y	Text = “TableIMM_RawData”.
MeasurementRecipeID	Identifies the measurement recipe.	RO	Y	Text.

14.5.4 *IMM Converted Data Table Object*

14.5.4.1 The IMM Converted Data Table Object holds the data that is converted into information needed for analysis of the substrate properties for advanced process control and post-mortem problem solving. Its attributes are defined in Table 17.

Table 17 IMM Converted Data Table Object Attribute Definitions

Attribute Name	*Definition*	*Access*	*Reqd*	*Form*
ObjID	Object ID.	RO	Y	Text assigned by the IMM.
ObjType	Type of Object.	RO	Y	Text = “TableIMM_ConvertedData”.
ConversionRecipeID	Conversion Recipe identifier (may be the name of the IMM recipe).	RW	Y	Text.
FeatureType	Type of feature extracted.	RO	Y	Enumerated text: “CD linewidth” “Thickness” “Reflectivity” “GoodnessOfFit” ….
RawDataTable	ObjID of IMM Raw Data Table object.	RO	Y	Text.

14.5.4.2 *Event Variables for Conversion Data Table Objects* — Table 18 specifies event variables required for Conversion Data Table Objects in addition to those defined in Table 15.

Table 18 Event Variables for Conversion Data Table Object

Variable Name	*Description*	*Type*	*Event*	*Comment*
DateTime	The date and time that the event occurred.	Conforms to the clock format in SEMI E98.	All.	
FeatureType	Type of feature extracted – corresponds to the attribute of the same name.	Enumerated text	All events of the IMMDTSM.	

14.5.5 *IMM Calibration Data*

14.5.5.1 The IMM Calibration Data object is used to hold the results of a formal substrate calibration process performed on a known good wafer. It includes both expected and actual data. Its attributes are defined in Table 19.

Table 19 IMM Calibration Data Table Object Attribute Definitions

Attribute Name	*Definition*	*Access*	*Reqd*	*Form*
ObjID	Object ID.	RO	Y	Text string assigned by the IMM.
ObjType	Type of Object.	RO	Y	Text = “TableIMM_CalibData”.
CalibrationDuration	Expected or mean duration of calibration procedure, in seconds.	RW	Y	Integer.
CalibrationStatus	Indicates whether or not the last substrate calibration was good.	RO	Y	Enumerated: Successful Failed
CalibrationTimeStamp	Date/time when the last calibration was completed.	RO	Y	Depends on TimeStampFormat.
RefSubstID	Identifier of substrate used for calibration.	RO	Y	Text. Null string if no reference substrate used for calibration.

14.6 *IMM Data Table Object Services*

14.6.1 Services related to IMM Data Table Objects are provided by the IMM object and are listed in Table 5, ¶11.5.

14.6.2 Scenarios involving IMM Data Table Objects are illustrated in Related Information 1.

15 Data Table Objects

15.1 *Data Table Object Model*

15.1.1 As shown in Figure 11, a Data Table Object has an associated Data Row Template and a set of Data Rows. When a table is transferred, it transfers an ordered set of column headers (Data Information ObjIDs) and a set of one or more rows, where each row consists of one or more individual entries following the order specified by the column headers. The information in a Data Row Template corresponds to the set of column headers. The ith column header identifies the type of data in the ith position within a row. The column headers are available both as a list of headers and as one of the attributes in a DataEntryTemplate.

15.1.2 Individual measurement data themselves have attributes, such as units and data type. These are provided in the Data Information object. The Data Information provides this additional information about the specified type of data entry (that is, for a specific column or position within a row). Each Data Row Template uses as many Data Information objects as there are columns in the table. Any given Data Information may be used by multiple Data Row Templates, since the specific Data Row Template produced may vary by recipe.

15.1.3 A master Data Row Template provides all available Data Information objects, while a Data Row Template for a specific measurement table provides only those Data Information objects that are available for a specific table. For example, one table may provide CD linewidth, while a different table provides film thickness. Specific data depends on the recipe used.

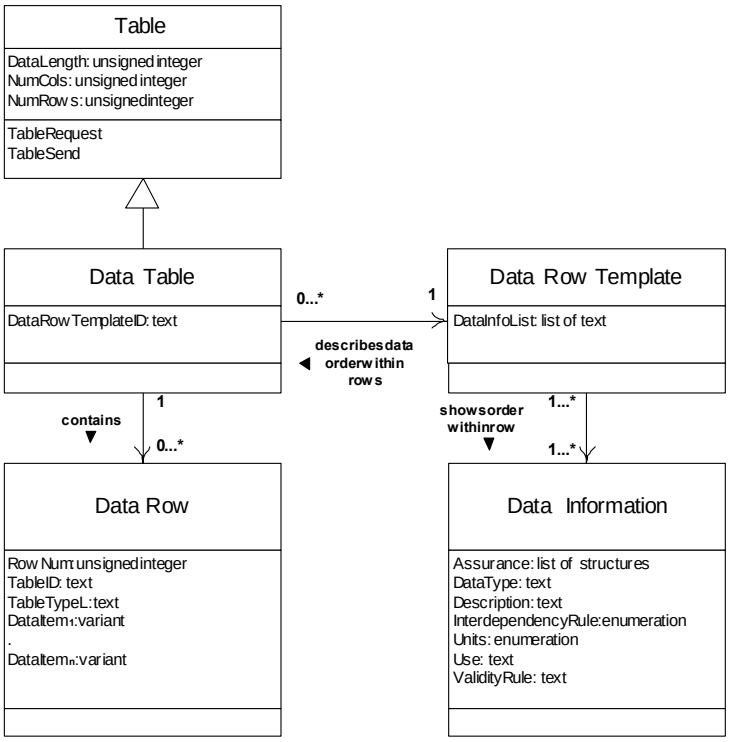

Figure 11
Data Table Object Model

15.1.4 *Data Table Object Attributes*

15.1.5 The attributes of the Data Table Object are defined in Table 20.

Table 20 Data Table Object Attribute Definitions

Attribute Name	*Definition*	*Access*	*Reqd*	*Form*
ObjID	Object ID.	RO	Y	Text. Complies with restriction of ObjID in SEMI E39.
ObjType	Type of Object.	RO	Y	Text = “TableData”.
RowTemplateID	ObjID of the associated Data Row Template.	RO	Y	Text.

15.1.6 The attributes of the Data Row object are defined in Table 21.

Table 21 Data Row Attribute Definitions

Attribute Name	*Definition*	*Access*	*Reqd*	*Form*
ObjID	Object ID.	RO	Y	Numeric text for the row number. Complies with restrictions of ObjID in SEMI E39.
ObjType	Type of Object.	RO	Y	Text = “DataRow”.
$DataItem_i$	The ith entry within the row.	RO	Y	Any data type except structures (lists of mixed data types).
RowNum	Identifies the row position within its table.	RO	Y	Unsigned integer.
RowOwner	Identifies the ObjType and ObjID of the table to which the row belongs.	RO	Y	Text. Conforms to the format of ObjSpec in SEMI E39.

15.2 *Data Row Template Object*

15.2.1 The Data Row Template provides the organization of data within any Data Row of a given Data Table Object.

15.2.2 Data Row Template Attributes are defined in Table 22.

Table 22 Data Row Template Attribute Definitions

Attribute Name	*Definition*	*Access*	*Reqd*	*Form*
ObjID	Object ID.	RO	Y	Text. Complies with restrictions of ObjID in SEMI E39.
ObjType	Type of Object.	RO	Y	Text = "DataRow_Template".
DataInfoIDList	Identifies the names of the individual data in the order in which the data will be transferred within a row when exported.	RO	Y	Ordered list of 1+ text.
NumEntries	Number of columns in table.	RO	Y	Unsigned integer.

15.2.2.1 *Data Information Object*

15.2.2.1.1 A Data Information object describes the properties available for an individual $DataItem_i$ in a Data Row for a table. The attributes of this object provide information about a kind of data item that can be reused for multiple tables, so that the information does not need to be duplicated.

15.2.2.1.2 All DataItems in a specific position within a row (that is, all that fall in the same column of a table) have the same set of properties. For example, a Data Information object for any cell in the column named "Xcoord" have the same description, the same data type, the same units, and so forth.

15.2.2.1.3 Assurance data is designated to be a container for information known by the IMM that describes the trustworthiness of the data itself. Since it is expected that the IMM Data will be used for decision making on the process tool, it is important that the client has access to not only the data but any other information to tell the client that the data is reliable. For Raw Data, examples of this might include actual voltages or laser power that is delivered to a measurement sensor (different from the setpoint), signal-to-noise ratios, or other parameters. For Result Data, examples might include goodness of fit for a model, confidence limit in a specific model parameter, or other information that is generated by the IMM.

15.2.2.1.4 The attributes of the Data Information object are defined in Table 23.

Table 23 Data Information Attribute Definitions

Attribute Name	*Definition*	*Access*	*Reqd*	*Form*
ObjID	Object ID.	RO	Y	Text. Corresponds to the name of the item of data (the column header). Complies with restrictions of ObjID in SEMI E39.
ObjType	Type of Object.	RO	Y	Text = "DataInformation".
Assurance	Information that describes the trustworthiness or believability in data generated.	RO	N	List of 0+ structures consisting of AssuranceName AssuranceDataType AssuranceValue AssuranceUnits

Attribute Name	*Definition*	*Access*	*Reqd*	*Form*
DataType	Identifies the format of the value of the Data Row attribute $DataItem_i$.	RO	Y	Enumerated: binary Boolean text integer unsigned integer floating point IEEE 754 array list structure
Description	Describes the type of entry.	RO	N	Text.
Interdependency Rule	Describes interdepencies with other entry types.	RO	N	Text.
Units	Units of measurement.	RO	Y	Enumerated.
Use	Descibes how the entry type is used.	RO	N	Text.
ValidityRule	Describes the conditions, such as boundaries, where the entry is valid.	RO	N	Text.

Table 24 Assurance Element Definitions

Element Name	*Description*	*Form*
AssuranceName	The name of the assurance data used.	Text.
AssuranceDataType	Identifies the form of AssuranceValue.	Enumerated: binary Boolean text integer unsigned integer floating point IEEE 754 array list structure
AssuranceValue	The actual measured or calculated value.	Per AssuranceDataType.
AssuranceUnits	Identifies the units (if any).	Enumerated text.

15.3 *Master Data Table Objects*

15.3.1 A Master Data Table Object is an empty table that provides a full set of Data Row Templates and Data Information for all of the possible row items (the superset) that the IMM is capable of producing. The existence of this table allows the client to obtain the description of data contained in Data Information beforehand. Actual row contents, and related Data Information, for an actual IMM Data Table Object must be a subset of those provided by the Master Data Table Object, and the order of data within a row may vary from table to table.

15.3.2 Exactly three Master Data Table Objects are required, one for each of the three types of tables. Their ObjIDs shall be defined as “RawDataMaster”, “ConvertedDataMaster”, and “CalibratedDataMaster”.

15.3.3 A scenario for obtaining Row Templates and Data Information from a Master Data Table Object is provided in Related Information.

15.3.4 Provision of Master Data Table Objects is an optional capability. In some implementations, due to the possible variation of Data Information attributes (for example, the Units attribute) and their dependence on recipes, it would be difficult to implement without having a specific recipe.

16 Requirements for Objects Defined in Other Standards

16.1 *Substrate Object*

16.1.1 SEMI E90 defines a generic set of substrate attributes needed for basic manufacturing and substrate tracking. The integration of measurement into a process tool requires additional information be available, both to the IMM itself when a substrate is received and to advanced process control systems that depend on the measurement results.

16.1.2 *Substrate Attributes*

16.1.2.1 Table 25 defines the additional substrate attributes required by this standard. By providing them with the substrate, they are available as context information for the IMM measurement tables defined below.

Table 25 Additional Substrate Attribute Definitions

Attribute Name	*Definition*	*Access*	*Reqd*	*Form*
ProcessFlowStep	Identifies exposure-level step in product fabrication process.	RW	Y	User-supplied text.
OrientationIn	Shows the orientation of the substrate as loaded.	RO	Y	Floating point, in radians.
OrientationOut	Shows the orientation of the substrate as unloaded.	RO	Y	Floating point, in radians.
ProcessToolID	Name of last process tool or ObjID of last tool process module.	RW	Y	User-supplied text. Shall conform to SEMI E39 restrictions on ObjID.
ProcessRecipeID	Name of recipe used in last process step.	RW	Y	User-supplied text.
SubstSide	Indicates the side of the substrate measured, as determined by the measurement recipe.	RO	Y	Enumerated: Front Back Both Unknown

16.2 *Process Job Management*

16.2.1 *Requirements*

16.2.1.1 The IMM shall be able to receive a Process Job Create request specifying a recipe, variable parameters, and list of one or more substrates, and it shall either accept or reject the request.

16.2.1.2 A Process Job Create request may be rejected if another Process Job still is active, or if there is an error in the request message.

16.2.1.3 A Process Job in the PROCESSING state shall transition to PROCESS COMPLETE when it has completed all measurement activities and stored the raw data. Upon this transition, the substrate may be removed from the IMM, and all IMM Raw Data objects generated are ready for use. The conversion of data may still be in process at this point.

17 Event and Exception Management

17.1.1 All data is defined either as the attributes of an object or as the attributes of an event. Data that must be available at the time of an associated event is defined for each type of object. The data provided with an event or alarm report shall be the new value that results from the event.

18 Message Service Definitions

18.1 *List of Services*

18.1.1 This section defines the content of service messages specified in this document and details the parameters for each message. Table 26 provides an alphabetical list of all services specified in this standard. Services marked with a Y in the column labeled “Control” shall be rejected if received from a Data Client.

Table 26 List of Services

Operation	*Description*	*Type*	*Reqd*	*Control*[#1]
ChangeService	The Control Client requests to change the Service state of the IMM.	R	Y	Y
ClientConnect	Request to be a client for either control or data services.	R	Y	—
ClientDisconnect	Request to discontinue client relationship.	R	Y	—
Delete	Request by Control Client to delete a Substrate Object (SEMI E39). (See ¶12.6.7 for detail.)	R	Y	Y
RequestRetentionConditions	Request for the current retention rules and retention time for the requesting client.	R	Y	—
SetRetentionConditions	Request to set or clear the client's current retention conditions.	R	Y	—
SetAutoTableSend	Set or clear automatic sending of completed tables for this client.	R	Y	—
TableRequest	Request for a table or table subset (SEMI E58).	R	Y	—
TableSend	Request to send a table or table subset (SEMI E58).	R	Y	—
TransferPathCalibration		R	Y	Y
UpdateSubstrateObject	Request by Control Client to create a Substrate Object and populate its attributes with the attributes provided. Also sent by the STPO for unloading the substrate.	R	Y	Y
XfrCmdListRequest	Request to send an ordered list of micro commands to be returned individually to the IMM STPO to affect a substrate hand-off.	R	Y	Y
XfrCommand	Message sent by the Control Client to an STPO to request a certain action (micro command) be performed. A response is sent when the action has been completed.	R	Y	Y
XfrVerify	Message sent by the Control Client to a STPO when all necessary actions have been completed to effect a transfer of a substrate and asking for confirmation.	R	Y	Y
XfrHalt	Message sent by either partner to stop all activities. This message ends the transfer procedure and requires manual intervention to clear.	R	Y	Y

[#1] Identifies services that may only be sent from the Control Client.

18.2 *Service Parameters*

18.2.1 *Table of Parameters and Parameter Elements* — Table 27 defines the parameters of the services defined in this document.

Table 27 Definitions for Parameters and Parameter Elements

Parameter	*Definition*	*Form*
AutoSendRawData	Enables (T) or disables (F) automatic sending for Raw Data Tables.	Boolean.
AutoSendConvertedData	Enables or disables automatic sending for Converted Data Tables.	Boolean.
AutoSendCalibrationData	Enables or disables automatic sending for Calibration Data Tables.	Boolean.
ClientID	An identifier used by the IMM to manage individual reports.	Text.
ClientInterfaceVersion	Identifies the version of the communication interface required by the client.	Text.

Parameter	*Definition*	*Form*
ClientConnectStatus	Identifies the success or errors.	Structure consisting of ConnectConnectAck List of 0+ pairs of ErrorCode ErrorText
ClientConnectAck	Indicates the success (T) or failure (F) of a connection or disconnection request.	Boolean.
ClientType	Identifies the type of client.	Enumerated: Control or Data.
Deltas	An ordered list of discrepancies for the differences in (x,y,z) between the expected position of the center of the incoming substrate and the actual position (expected value minus actual value).	Ordered list of 3 floating point values for (x,y,z) discrepancies.
ErrorCode	Contains the code for the specific error found.	Enumerated. Service-dependent.
ErrorText	Text in support of the error code.	Text.
OrientationIn	Specifies orientation of substrate to be loaded.	Floating point.
OrientationOut	Specifies orientation of substrate to be unloaded.	Floating point.
Properties	A list of attribute name/value pairs.	List of 0+ structures consisting of Attr.
RetentionRules	0–2 level list of retention rules.	List or nested list of name/value pairs of retention rules.
ServiceAck	Indicates the success or failure of a service request.	Enumerated: Service performed with no errors Can not perform now At least one parameter is invalid.
Status	Identifies the success or errors for a service request.	Structure consisting of ServiceAck List of 0+ pairs of ErrorCode ErrorText
XfrAck	Indicates success or failure.	Boolean.
XfrCmdList	An ordered list of XfrCmdName to be used for either a load or an unload operation.	List of 1+ text.
XfrCmdName	Indicates specific action requested.	Text.
XfrHaltAccept	True if halt accepted. May be denied only if the specified transfer procedure does not exist.	Boolean.
XfrLink	Transfer identifier.	Unsigned integer.
XfrStatus	Information about the success or failure of a Calibrate Transfer Path service request.	Structure of: XfrAck List of 0+ pairs of ErrorCode ErrorText

18.2.2 *RetentionRules Parameter Elements* — The elements within a list of RetentionRules are defined in Table 28.

Table 28 Definitions for RetentionRule Parameter Elements

Name	*Definition*	*Form*
AfterTransfer	The table may be deleted after transfer to the client.	Non-negative integer. A value of zero disables this rule, and any positive value enables it.

Name	*Definition*	*Form*
ClientDelete	The minimum number of tables the client wants to be able to be able to store prior to being notified by the event TableStorageCapacityWarning.	Non-negative integer. A value of zero disables this rule.
MaximumTablesStored	Maximum number of tables the client wants stored; beyond that they are to be deleted on a first-in, first out basis.	Non-negative nteger. A value of zero disables this rule.
RetentionTime	The minimum time that a table shall be retained in storage. The table may not be deleted before this period of time has passed unless Automatic Rollover is forced to delete it to make room for a new table.	Long integer for number of hours to retain tables. A value of zero disables this rule.

18.2.3 Valid ErrorCode enumerations are shown in Table 29. The actual format for ErrorCode is communication protocol-dependent.

Table 29 Valid ErrorCode Enumerations

Service	*Valid Logical Values*
ClientConnect	Client Already Connected Duplicate ClientID Invalid ClientType IncompatibleVersions
ClientDisconnect	Unrecognized ClientID (Client not currently connected)
XfrCommand	Rejected/Unrecognized Command Rejected/Parameter Error
XfrCommand XfrVerify	Failed (Completed Unsuccessfully) Failed (Unsafe) – External intervention required
XfrVerify	Sensor-Detected Obstacle Material Not Sent Material Not Received Material Lost Hardware Failure Transfer Cancelled
All other services	Action performed – no errors Action will be performed at earliest opportunity Action can not be performed now Action failed due to errors Unrecognized command At least one parameter has an error

18.3 *Message Detail*

18.3.1 *ClientConnect Service* — This service is used by a prospective client to request a connection as a Control Client or Data Client. If the request is as a Control Client, and there is already a Control Client connected, the IMM shall reject the request. The IMM shall also reject the request if a client with the same ID is already connected or if the required interface version can not be supported. Table 30 defines the parameters for ClientConnect.

Table 30 ClientConnect Service Parameters

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
ClientID	M	M(=)	Identifier for the requestor.
ClientInterfaceVersion	M	—	Identifies the version of the communication interface required by the client.
ClientType	M	—	Indicates type of connection (control or data).
ClientConnectStatus	—	M	Indicates the success or the specific errors.

18.3.2 *ClientDisconnect Service* — This service is used by a current client to gracefully terminate a connection. Table 31 defines the parameters for the ClientDisconnect service.

Table 31 ClientDisconnect Service Parameters

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
ClientID	M	M(=)	Identifier for the requestor.
ClientConnectStatus	—	M	Indicates the success or the specific errors.

18.3.3 *RequestRetentionConditions* — A client may request its current set of retention conditions for a Raw Data Table Object or Converted Data Table Object by sending the RequestRetentionRules request at any time. Table 32 defines the parameters for the RequestRetentionConditions service.

Table 32 RequestRetentionConditions Service Parameters

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
ClientID	M	M(=)	Identifier for the requestor.
TableType	M	—	Identifies the type of table affected.
(list of 0+) retention rules	—	M	Current list of rules, or empty list if no rules.
Status	—	M	Status of request.

18.3.4 *ChangeService* — The Control Client may request a change in the IMM Service state at any time. The actual state change from IN SERVICE to OUT OF SERVICE may be delayed until a currently active process job completes. Parameters for the Service service are defined in Table 33.

Table 33 Parameters for ChangeService

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
ServiceState	M	—	New Service state requested.
Status	—	M	Status of request.

18.3.5 *SetAutoTableSend* — A client may enable or disable the automatic sending of any or all of the IMM Data Table subtypes through the SetAutoTableSend service. Table 34 defines the parameters of the SetAutoTableSend service.

Table 34 SetAutoTableSend Service Parameters

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
ClientID	M	M(=)	Identifier for the requestor.
AutoSendRawData	M	—	Enables or disables automatic sending for Raw Data Tables.
AutoSendConvertedData	M	—	Enables or disables automatic sending for Converted Data Tables.
AutoSendCalibrationData	M	—	Enables or disables automatic sending for Calibration Data Tables.
Status	—	M	Status of request.

18.3.6 *SetRetentionConditions* — A client may set, change, or clear its current set of retention conditions at any time through the SetRetentionConditions service. Table 35 defines the parameters of the SetRetentionConditions service.

Table 35 SetRetentionConditions Service Parameters

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
ClientID	M	M(=)	Identifier for the requestor.
TableType	M	—	Identifies the type of table affected.

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
(list of 0+) RetentionRules	M	—	New list of rules, or empty list to clear all rules.
Status	—	M	Status of request.

18.3.7 *TransferPathCalibration Service* — This service is sent by the Control Client to request a STPO to perform an automatic calibration. Its parameters are defined in Table 36.

Table 36 TransferPathCalibration Service Parameters

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
ObjID	M	M(=)	Identifier of the STPO.
OrientationIn	C	—	Expected orientation of substrate loaded.
OrientationOut	C	—	Expected orientation of substrate unloaded.
Deltas		M	An ordered list of changes in the load plane in the x-, y-, and z-position of the center of the substrate in millimeters respectively.
Status		M	Status of request.

18.3.8 *XfrCmdListRequest* — This service provides the Control Client with an ordered list of necessary micro commands required during a substrate transfer that effect either a Load or Unload operation. Table 37 defines the parameters of this service. The individual micro commands are then passed to the STPO in the XfrCommand service below. These micro commands are proprietary, with the exception of two required micro commands, and need not be understood by the Control Client but must be used in the order in which they appeared in the list.

Table 37 XfrCmdListRequest Service Parameters

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
ObjID	M	M(=)	Identifies the SubstrateTransferPath object.
XfrType	M	M	Identifies the request is for either a load or unload operation.
XfrCmdList	—	M	List of 0+ micro commands.

18.3.9 *XfrCommand Service Parameters* — This service is used to effect an actual substrate transfer with the IMM object. Parameters for the XfrCommand service are defined in Table 38.

Table 38 XfrCommand Service Parameters

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
ObjID	M	—	Identifies the STPO.
XfrLink	M	M(=)	Identifies the specific transfer.
XfrCmdName	M	—	Specific micro command to be executed.
XfrStatus	—	M	Reports the success or failure of the micro command. If failure, supplies error detail.

18.3.10 *XfrHalt Service Parameters* — The XfrHalt service may be sent by either the Control Client or the IMM to effect an immediate halt. Parameters for the XfrHalt service are defined in Table 39.

Table 39 XfrHalt Service Parameters

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
XfrLink	M	M(=)	Identifies the specific transfer.
XfrHaltAccept	M	M	True if halt accepted. May be denied only if the specified transfer does not exist.

18.3.11 *XfrVerify Service Parameters* — This service is used to verify to the partner that the substrate transfer was completed successfully from the point of view of the sender. Parameters for the XfrVerify service are defined in Table 40.

Table 40 XfrVerify Service Parameters

Parameter	*Req/Ind*	*Rsp/Conf*	*Description*
XfrLink	M	M(=)	Identifies the specific transfer.
XfrStatus	M	M	Reflects the transfer status from the point of view of the sender of the request.

19 Services Provided by Client

19.1 *List of Client Services*

19.1.1 In addition to the services defined in §19, the client is also expected to provide specific services. These are listed in Table 41. The column labeled Defined identifies the standard where the service is defined. The column labeled Control indicates if the message is only sent to the Control Client.

Table 41 List of Client Services

Operation	*Description*	*Defined*	*Control**
UpdateSubstrateObject	Used by the STPO when unloading a substrate to provide information about the substrate.	E90	Y
TableSend	Used by the IMM for the AutoTableSend capability.	E58	—

20 IMMC Compliance

20.1 *Overview of Requirements*

20.1.1 An Integrated Measurement Module may be said to be fully compliant to IMMC if and only if all fundamental IMMC requirements, as shown in Table 42 Compliance below, are fully implemented as specified in the cited sections. All object attributes, state models, events, and event data shall be implemented.

20.1.2 Additional capabilities, specified in Table 42, are optional for the IMM supplier. If implemented, then they shall be fully implemented as specified in the cited sections.

20.1.3 Further capabilities, not covered in this standard, may be provided. However, they shall not contradict those capabilities that are specified here.

20.1.4 IMM Data Table Objects may be implemented by independent software modules for the purpose of allowing conversion of data in an IMM Raw Data Table, producing an IMM Converted Data Table. Such objects may be termed IMMC-compliant IMM Data Table Objects. Such an implementation that is not part of a fully IMMC-compliant Integrated Measurement Module shall not term itself as an IMMC-compliant IMM object.

20.1.5 Table 42 shows the sections that are required for IMMC Compliance.

Table 42 IMMC Compliance

Fundamental IMMC Requirements	*IMMC Section*	*Level of Compliance*
SEMI E40, SEMI E90, and SEMI E116	7.2	□ Full □ Partial □ Not Implemented
Basic Capabilities: Clock, Events, Alarms, Exceptions, Recipes	7.3	□ Full □ Partial □ Not Implemented
State Models	7.4	□ Full □ Partial □ Not Implemented
Objects and SEMI E39	7.5	□ Full □ Partial □ Not Implemented
Communication Ports and Clients	9–all	□ Full □ Partial □ Not Implemented
IMM Object	11–all	□ Full □ Partial □ Not Implemented

Fundamental IMMC Requirements	*IMMC Section*	*Level of Compliance*
Substrate Transfer Path Object	12.1–12.5, 12.7.3, 12.9–12.10, 12.11.2.3, 12.11.3.3	□ Full □ Partial □ Not Implemented
IMM Data Table Object	13, 14.1–14.4	□ Full □ Partial □ Not Implemented
IMM Raw Data Table Object	14.5.1–14.5.3	□ Full □ Partial □ Not Implemented
IMM Converted Data Table Object	14.5.1–2, 14.5.4	□ Full □ Partial □ Not Implemented
IMM Calibration Data Table Object	14.5.1–2, 14.5.5	□ Full □ Partial □ Not Implemented
Data Table, Data Row, Data Row Template, and Data Information Objects	15.1–15.2	□ Full □ Partial □ Not Implemented
Additional Requirements for Substrate Object	16.1–all	□ Full □ Partial □ Not Implemented
Additional Requirements for Process Job Management	16.2–all	□ Full □ Partial □ Not Implemented
Event Data	17–all	□ Full □ Partial □ Not Implemented
Message Service Definitions	18–all	□ Full □ Partial □ Not Implemented
Additional IMMC Capabilities	*IMMC Section*	*Level of Compliance*
Substrate Verification (Substrate ID Reader)	12.6	□ Full □ Partial □ Not Implemented
Substrate Orientation Detection	12.7–12.7.2, 12.7.4	□ Full □ Partial □ Not Implemented
Transfer Path Calibration (X,Y,Z Detection)	12.8	□ Full □ Partial □ Not Implemented
Master Data Table	15.3	□ Full □ Partial □ Not Implemented

21 Related Documents

21.1 *List of Documents*

21.1.1 This section is for useful documents related to this standard but not referenced within the specification.

21.1.1.1 Developing Software with UML, Addison-Wesley c1999 by Oestereich, Bernd

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

RELATED INFORMATION 1
USE CASES AND SCENARIOS

NOTICE: This related information is not an official part of SEMI E127 and was derived from work of the originating committee. This related information was approved for publication by full letter ballot procedures.

R1-1 Data Tables Objects

R1-1.1 *Obtaining A Directory Of Existing Table Objects*

R1-1.1.1 The scenario in Table R1-1 illustrates getting a directory of all IMM Data Table Objects using GetAttr.

Table R1-1 Getting a Directory of Tables

	Comments	*Client*	*Dir*	*IMM*	*Comments*
1		GetAttr.req	→		
		ObjSpec = object specifier for IMM ObjType = 'TableIMM_Data list of AttrName = ("Objtype", "ObjID")			
			←	GetAttr.rsp	
				list of ObjSetting = pairs of ObjType, ObjID for all existing tables.	

R1-1.2 *Getting Table Templates*

R1-1.2.1 The client is able to obtain the Data Row Template and Data Information either from an actual table object or for the Master Data Table Object of interest. In the example shown in Table R1-2, the client wants the information from the Master Converted Data Table. The same scenario, with different values for ObjType and ObjID, could be used to get the specific templates for an actual table object.

Table R1-2 Getting IMM Data Table Object Templates

	Comments	*Client*	*Dir*	*IMM*	*Comments*
1		GetAttr.req	→		
		ObjSpec = object specifier for IMM ObjType -= "TableIMM_ConvertedData" list of ObjID = "MasterConvertedData" list of AttrName = ("RowTemplateID")			
			←	GetAttr.rsp	
				list of ObjSettings = ("RowTemplateID", "MasterConvertedDataRowTemplate")	
		GetAttr.req	→		

	Comments	Client	Dir	IMM	Comments
		ObjSpec = object specifier for IMM ObjType = “DataRow Template” list of ObjID = (“MasterConvertedDataRow Template”) list of AttrName = (“DataInfoIDList”)			
			←	GetAttr.rsp	
				list of ObjSetting = (“DataInfoIDList”, list of ObjIDs) for the set of DataInformation Objects”_	
		GetAttr.req	→		
	All attributes of the Data Information object are requested for all of these objects.	ObjSpec = object specifier for IMM ObjType = “DataRow_Template” list of ObjID = same as contents of DataInfoIDList list of AttrName = (empty list)			
			←	GetAttr.rsp	
				list of ObjSetting = (((“ObjID”, “LineWidth”) (“Assurance”,assurance value), “DataType”, “F16”), : (“ObjID”, “Data Type”), (“Units”, “micron”)), ... (...)))	list of lists of name/value pairs for each supported attribute for each DataInformation ObjID.

R1-1.3 *Table Services*

R1-1.3.1 A client wants to download a Raw Data Table from a previous run for analysis. Table R1-3 illustrates this scenario.

Table R1-3 Sending a Table

	Comments	Client	Dir	IMM	Comments
1		TableSend.req	→		
		ObjSpec = object specifier for the IMM, TableType = “TableIMM_RawData”, TableID = “P19343445-Lot103102#5”, TableCmd = Entire table; list of TableAttr = name/value attribute pairs; list of ColHdr = list of names of column headers; list of values for 50 rows			
			←	TableSend.rsp	
				TableAck = Success:	