SEMI T12-0710
SPECIFICATION FOR TRACING JIGS AND IMPLEMENTS

This standard was technically approved by the global Traceability Committee and is the direct responsibility of the Japanese Traceability Committee. Current edition approved by the Japanese Regional Standards Committee on November 24, 2004. Initially available at www.semi.org January 2005. Originally published March 2004; previously published July 2004.

NOTICE: The designation of SEMI T12 was updated during the 0710 publishing cycle to reflect the reapproval of SEMI T12.1.

1 Purpose

1.1 Semiconductor production equipment may use Jigs and/or Implements to produce semiconductor devices. Some case such important conditions as quality of products and tool performance may depend on Jigs or Implements. This difference often comes from such historical events as processing cycles, processing products, stocking environment. To make sure correlation between the conditions and the events tracing Jigs and Implements is required.

1.2 Purpose of this document is to establish concept to trace Jigs and/or Implements used on equipment for processing material and make sure required information, communications and services to realize the concept.

2 Scope

2.1 Scope of this specification includes management of required information on equipment. Because a specific piece of jig may be used on other equipment, usually same type and/or same supplier, information transfer between equipment and factory host computer is also in the scope of this specification.

2.2 Such physical issues mark position and mark readers are out of scope, for this specification treats just tracing information, communication of it and logical management. The physical specification may be standardized separately.

2.3 As long as management on this specification and above requirement is expected on an equipment, the equipment is in the scope regardless of the type of process or type of equipment.

2.4 But if Jigs and/or Implements for certain type of equipment have been already traced or managed complying with existing standard or along with some specification conventionally, they are out of scope of this specification. But this specification can be applied even to such jigs and tools if user and/or suppliers want.

2.5 Because this specification describes just base functionalities to realize the traceability, applications of the functionalities including checking or management of counterfeits are out of scope.

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

3 Limitations

3.1 This specification defines and makes sure concept and information transfer basics. Specification about physical ID marking on jigs and tools including readers/writers, and such application of this tracking as security and operations are dependent to standard users. They are responsible for their application and related regal issues.

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

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

INACTIVE

Documents Document Sharing Hub
Full document details: SEMI T12-0710

5 Terminology

5.1 *Definitions*

5.1.1 *attachment* — general term of Jig and Implement.

5.1.2 *implement* — a kind of Attachment used to assemble/maintain/improve measurement or production equipment.

5.1.3 *jig* — a kind of Attachment used to support measurement or production for material on one or more equipment.

6 Convention

6.1 This section defines the conventions followed by this document.

6.2 *Object Conventions* — This document conforms to the conventions for objects established by SEMI E39, including object diagrams, object terminology, and requirements for standardized objects. Accordingly, notation is based on Unified Modeling Language (UML).

6.2.1 *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.

6.2.2 *Components of Complex Attributes* — The names of object attributes defined in tables are left-justified. The individual elements of complex attributes are right-justified in order of appearance below the complex attribute.

6.3 *State Model Conventions*

6.3.1 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."[1]

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

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

1. Actions taken upon exit of the previous state.
2. Actions taken upon entry of the new state.
3. Actions taken which are most closely associated with the transition.
4. No differentiation is made between these cases.

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

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

6.4.1 *Service Definition*

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

1 Harel, D. "State charts: A Visual Formalism for Complex Systems." *Science of Computer Programming 8* (1987).

INACTIVE
Documents
Document Sharing Hub
Full document details: SEMI T12-0710

Message Service Name	*Type*	*Description*

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

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

6.4.2 *Service Parameter Dictionary*

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

Parameter Name	*Description*	*Format: Possible Value*

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

6.4.3 *Service Message Definition*

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

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

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

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

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

7 General Requirements

7.1 *Identification Means* — Equipment has to have any means to identify physical Jigs and/or Implements.

7.2 *Communication Means* — Each equipment and managing system has to have any electric communication means to transact.

8 Overview

8.1 *Attachments in Semiconductor Manufacturing* — A semiconductor production line consists of many amount pieces of equipment with many types. Some types of equipment require such attachments as Jigs and/or Implements

INACTIVE

Documents Document Sharing Hub
Full document details: SEMI T12-0710

to manufacture semiconductor devices. Because some of them act very important roles, quality, yield or productivity of process may depend on them and their management.

8.2 *Management of Attachments* — Management is required on specific attachments to have better product quality, yield and productivity. This document specifies definitions, concepts, behavior and services to manage attachments on equipments and also in factory. However not all the attachments require such management and specification of this document should be applied to restricted ones especially to the ones used for important processes, sensitive and incompatible combinations between attachments and equipment.

8.3 *Effective Concepts for a Variety of Attachments* — There are many kinds of equipment types and also a variety of attachments. To be effective to almost all of them, specification of this document is generic and idealized without any physical limitations or practical details. It is also important to be independent to rapidly changing technologies and production environment.

8.4 *Logical and Informational Specification* — To keep track of conceptual document and keep shared area as wide as possible, this document specifies just logical and informational issues, except related information part. Physical specifications are assumed to be defined as separated documents; they may be specified in other standards, propriety specifications or mixed together.

8.5 *Tracking Oriented Management* — Because there may be very specific managements or operation for strategy in factories and they may not be shared by many fabs, strategic management for specific factory or products may not be better to be standardized. Most parts of the management specified in this document are related to traceability. In other words "Tracking of Jigs and Implements" is the standardized area of "Management of Attachments" in this document.

8.5.1 *Tracking Jigs and Implements* — In a semiconductor factory, it is necessary for any management of Jigs and/or Implements to be tracked.

8.5.1.1 There are two types of information to track Jigs and/or Implements. One of them is transferring information: e.g., where it is and who use it. It is required to discriminate between logical information and physical attachment. The other is historical information of the Attachment: e.g., how much it was used, what condition the processes were and when it was repaired.

8.5.1.2 Such an attachment as Jig or Implement is transferred from stock yard to equipment or the other way and may be between two pieces of equipment. At the time it is picked up at source location for destination, the fact should be reported to Tracking System. When it arrives at the destination, this fact is required to be reported to the system. Verification has to be done at the destination before it is dealt: e.g., process, maintain or stored. If a problem happens during transportation, an exception is raised.

8.5.1.3 When it is used on equipment; e.g., for process or maintenance, its history information is accumulated and the information is reported to the system at every milestone. Examples of the history information are time duration, cycle time, such process parameters as temperature and pressure, material characteristics and known damage occurred during the process.

8.5.1.4 The tracking system summarizes reported information. Clients of the system may inquire the information. One of the clients may be scheduler/dispatcher and it may change process machine for specific material to get better product quality.

8.5.1.5 A client may ask target equipment to handle specific event which shows reaching limited cycle time for a jig on the equipment. Expected handling for the event; e.g., is to stop processing. The client may ask to pause before starting a lot if current cycle time of the jig is very close to expiration cycle time.

8.5.2 *Application of Tracking System* — Because this document specify basic capability tracking Jigs and Implements, specific applications for semiconductor device manufactures or equipment suppliers and such advanced applications as securitized tracking are out of scope of this document. Using this basic specification and additional capability supplied by equipment users, equipment suppliers or system suppliers, expected advanced capabilities can be realized.

INACTIVE

Documents
Document Sharing Hub
Full document details: SEMI T12-0710

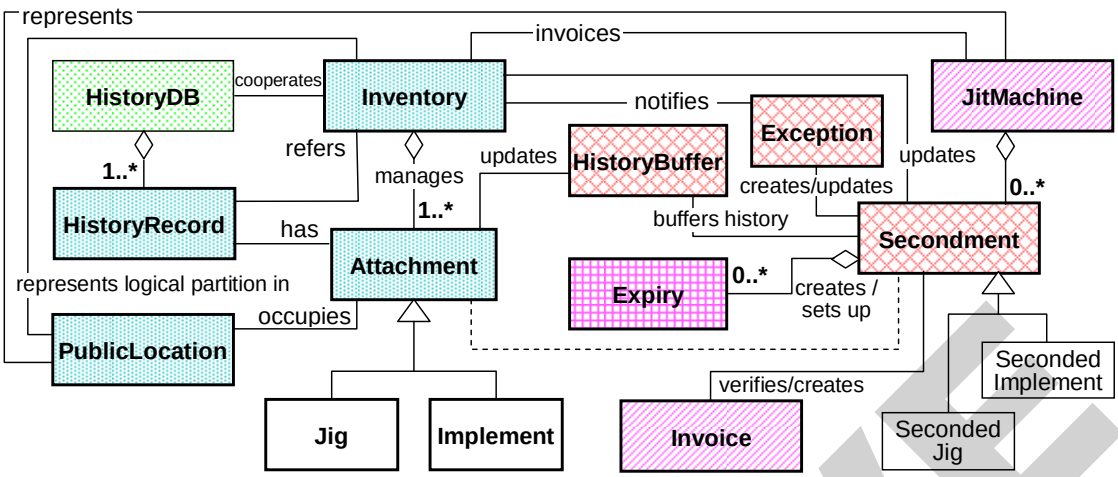

Figure 1
Modeling Jigs-Implements Tracking

9 Concepts

9.1 *Domain Analysis and Object Based Modeling* — To model and design complicated system or to have shared specification of a part of system which is used widely in deferent applications, some sophisticated methodologies are required. Analyzing problem domain by object oriented paradigm is one of the best practices to be taken. Because this technology is frequently used in many SEMI Standards, there is no confusion if this document follows the way.

9.1.1 *UML Representation* — Above diagram is modeling of tracking capability for Jigs and/or Implements in UML diagram. It is not required to make system with object oriented programming language for compliant to this specification. Even if UML is typical object oriented representation, object oriented implementation may be expected but may not be mandatory.

9.1.2 *Total Modeling* — To make sure what the tracking system is, the model depicted above describes not just tracking information acquisition but also includes fundamental management of the whole tracking system.

9.1.2.1 *Acquisition of Minimum Tracking Information* — Such capabilities as awareness of Attachments, accumulation of use history until release timing and its report are required to acquire use-history of Jigs and/or Implements on equipment. Cross-hatched classes (HistoryBuffer and Secondment) in light red and their collaboration correspond to the acquisition capabilities. These capabilities are usually implemented on equipment.

9.1.2.2 *Action on Expiration* — One of the biggest purposes of tracking such attachments as Jigs and/or Implements is to prevent misprocessing with invalid attachments. An Attachment may be invalid when its use gets beyond expected duration or cycles. This is such a capability as an expected action when an attachment has been expired by facing invalid situation. Light-red grided class (Expiry) and its collaboration correspond to the special action capabilities. This capability is usually implemented on equipment.

9.1.2.3 *Identification and Verification of Attachments* — Such information system as equipment controllers may not understand what the Attachment is even if attachment ID is known. Capability to understand that and verify between information and physical attachment helps not only above fundamental capability but also ensured tracking. Pink-hatched classes (JitMachine and Invoice) and their collaboration correspond to this capability. This capability is also implemented on equipment usually.

9.1.2.4 *Tracking Information Management* — Such information system as Inventory is necessary to manage and apply acquired Attachment tracking information effectively. Blue thick-shadowed classes (Attachment, HistoryRecord, Inventory and PublicLocation) and their collaboration correspond to this capability. This capability is usually installed on master equipment, cell controller or host computer. Or it may be distributed on these platforms.

9.1.2.5 *Further Application Interface* — Final purposes of tracking such attachments as Jigs and Implements are controlling life time of attachments, increasing effectiveness of attachments, preventing fault process and achieve

INACTIVE

Documents Document Sharing Hub
Full document details: SEMI T12-0710

good quality of products. In this regard this interface capability helps applications. Green light shadowed class (HistoryDB) corresponds this capability. This capability may be installed on the host or application server.

9.1.3 *Collaboration in Class Modeling* — An example of collaboration is as follows.

9.1.3.1 Application seeks certain type of jigs through HistoryDB. It finds good ones to produce expensive products. It asks Inventory where they are.

9.1.3.2 It picks up the ones and delivers them to target equipments. It updates inventory and inventory sends invoices to JitMachines. Invoice object is created on each JitMachine. At the time each jig arrives at each piece of target equipment, the Invoice verifies. If verification succeeds, this fact is notified to inventory, location is updated, Secondment is created and JitMachine identifies what the sent jigs are. At the same time HistoryBuffer is also created by the Secondment.

9.1.3.3 Equipment runs. HistoryBuffer accumulates events, process conditions and cycle. At some moment, it reports the buffer content to inventory. The inventory manages HistoryRecord.

9.2 *Attachment* — Attachment represents a Jig or an Implement. This concept is used to handle common nature of the both items.

9.2.1 *Attachment object* — Attachment object represents such supporting resource as Jig or Implement logically. Attachment object is an instance of attachment class defined later. Attachment object identifies corresponding physical attachment and its information. This object is abbreviated as "Attachment" in this document and often 'object' is omitted.

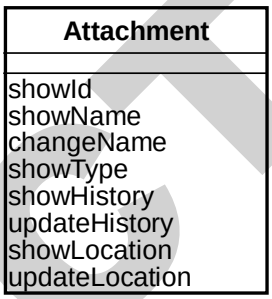

Figure 2
Attachment Class

9.2.2 *Attachment Class* — Attachment class defines its inherent data, behavior and services. This class generalizes Jig and Implement registered in an Inventory. This class is illustrated as above for reference.

9.2.2.1 *Responsibility of Attachment* — This class is responsible for holding and organizing such information as history for a Jig or Implement belonging to an Inventory.

9.2.2.1.1 To complete the responsibility it accepts creation/removal of a HistoryRecord, adding events or increment time/duration requests and inquiry. They are defined as services of Attachment in this specification. Service names may be different from capabilities above.

9.2.2.1.2 Inherent data is used to complete the responsibility. The data is also referred to as attributes and they have to be defined in some way. This document doesn't define attributes directly with attribute definition table shown in some SEMI standard documents but defines them indirectly as service parameters.

9.2.2.1.3 Behavior is also defined to complete the responsibility. Behavior is often represented by collaboration with outside of this class and state model. Collaboration reflects a part of associations of this class with others. Collaboration works by asking services defined on accompanying object which is the other end of association on class diagram. State model is representation of the other part of behavior of this class. However it has to be defined if this class has explicit state, state model is not defined here because this class is stateless.

9.3 *Exception* — Exception is unusual state on any Jig and/or Implement or on the tracking capability including hazardous situation: potential or already destroy materials, attachments, equipment, explosive or insure/kill human.

INACTIVE

Documents
Document Sharing Hub
Full document details: SEMI T12-0710

Exception happens on equipment or may be storage capability of corresponding Inventory. This fact may be added to history.

9.3.1 *Exception Object* — Exception object represents unusual situation happened usually on equipment. It is created at the time it is raised. It is destructed after normal condition is recovered from the situation. Exception object is an instance of Exception class defined later. This object is abbreviated as "Exception" in this document and often 'object' is omitted.

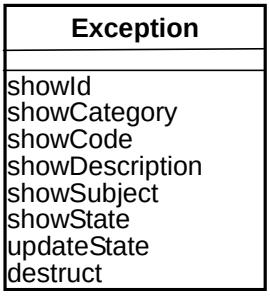

Figure 3
Exception Class

9.3.2 *Exception Class* — Exception Class defines its inherent data, behavior and services. This class is illustrated as above for reference.

9.3.2.1 *Responsibility of Exception* — This class is responsible for identifying situation, its status and such subject as Jigs or Implements.

9.3.2.1.1 To complete the responsibility it accepts notification of state change or adding appendix for the state. They are defined as services of Exception in this specification. Service names may be different from capabilities above.

9.3.2.1.2 Inherent data is used to complete the responsibility. The data is also referred to as attributes and they have to be defined in some way. This document doesn't define attributes directly with attribute definition table shown in some SEMI standard documents but defines them indirectly as service parameters.

9.3.2.1.3 Behavior is also defined to complete the responsibility. Behavior is often represented by collaboration with outside of this class and state model. Collaboration reflects a part of associations of this class with others. Collaboration works by asking services defined on accompanying object which is the other end of association on class diagram. State model is representation of the other part of behavior of this class. Because this class has explicit state, it defines following states and state model by state chart and transition definition table.

9.3.2.1.3.1 *SET* — Unusual situation is still effective.

9.3.2.1.3.2 *CLEARED* — Unusual situation is removed and it is normal condition about the cause of exception.

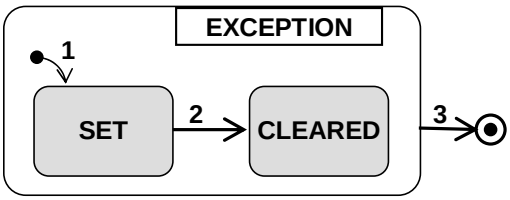

Figure 4
Exception State

INACTIVE

Documents Document Sharing Hub
Full document details: SEMI T12-0710

Table 1 Transition Definition Table of Exception

#	*Previous State*	*Trigger*	*New State*	*Actions*	*Comments*
1	(No state)	Unusual situation happens.	SET		
2	SET	Unusual Condition has been removed.	CLEARED	Exception information may be added to history.	It's not always termination.
3	CLEARED	Termination is instructed.	(No state)		

9.4 *Expiry* — Expiry represents a kind of nature of a Jig or Implement which has explicit valid duration: processed time period, critical process time, cycle and so on. An Expiry is prepared for a specific duration to allow special action on the Attachment when it has expired.

9.4.1 *Expiry Object* — Expiry object encapsulates a series of information about one of Expiry conditions on an Attachment. It is created at the time Secondment, a representation of an Attachment on equipment, is created or an instruction to specify the expiry is accepted by the Secondment and destructed when the Secondment is removed from equipment. It is also destructed to withdraw expiration after special action is invoked when it fires the expiration. Expiry object is an instance of Expiry class defined later. This object is abbreviated as "Expiry" in this document and often 'object' is omitted.

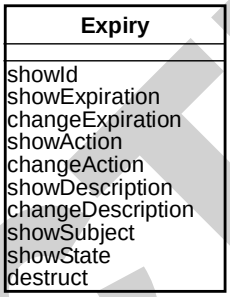

Figure 5
Expiry Class

9.4.2 *Expiry Class* — Expiry class defines its inherent data, behavior and services. This class is illustrated as above for reference.

9.4.2.1 *Responsibility of Expiry* — This class is responsible for maintaining expiration of such subject as Jigs or Implements.

9.4.2.1.1 To complete the responsibility it checks expiration subject of the Attachment. This is started at the time it is created. Services to help the responsibility are defined in this specification. Service names may be different from capabilities above.

9.4.2.1.2 Inherent data is used to complete the responsibility. The data is also referred to as attributes and they have to be defined in some way. This document doesn't define attributes directly with attribute definition table shown in some SEMI standard documents but defines them indirectly as service parameters.

9.4.2.1.3 Behavior is also defined to complete the responsibility. Behavior is often represented by collaboration with outside of this class and state model. Collaboration reflects a part of associations of this class with others. Collaboration works by asking services defined on accompanying object which is the other end of association on class diagram. State model is representation of the other part of behavior of this class. Because this class has explicit state, it defines following states and state model by state chart and transition definition table.

9.4.2.1.3.1 *VALID* — Attachment is valid for a subject.

9.4.2.1.3.2 *EXPIRED* — Attachment is invalid because of expiration of specified duration.

INACTIVE

Documents
Document Sharing Hub
Full document details: SEMI T12-0710

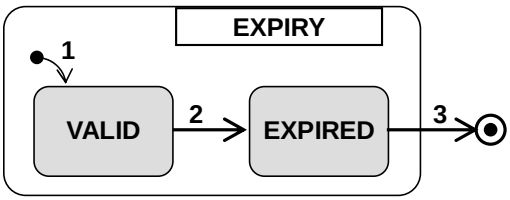

Figure 6
Expiry State

Table 2 Transition Definition Table of Expiry

#	*Previous State*	*Trigger*	*New State*	*Actions*	*Comments*
1	(No state)	Expiry object is created.	VALID	Watchdog is started.	
2	VALID	Subject has been expired.	EXPIRED	Registered special action is carried out.	
3	EXPIRED	The action is started.	(No state)		

9.5 *History* — History is the concept to track what happens on Jigs and/or Implements and how they are used. A few object classes are modeled to complete this capability.

9.6 *HistoryBuffer* — HistoryBuffer is tentative history information of a Jig or Implement on equipment. The information is reported to Inventory periodically or at some milestones. When it is reported, the information is cleared for new information.

9.6.1 *HistoryBuffer Object* — HistoryBuffer object represents buffer area of tentative history information of an attachment on equipment. It is created at the time an attachment is created and destructed after the Attachment is removed. HistoryBuffer object is an instance of HistoryBuffer class defined later. This object is abbreviated as "HistoryBuffer" in this document and often 'object' is omitted.

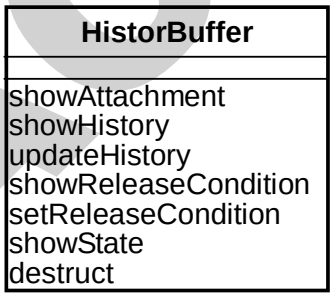

Figure 7
HistoryBuffer Class

9.6.2 *HistoryBuffer Class* — HistoryBuffer class defines its inherent data, behavior and services. This class is illustrated as above for reference.

9.6.2.1 *Responsibility of HistoryBuffer* — This class is responsible for keeping temporary history information of Jig or Implement on equipment and clear the information after it is reported to Inventory.

9.6.2.1.1 To complete the responsibility it accepts information what happens on an Attachment or how it is used including period or cycle. It also accepts inquiries. They are defined as services of HistoryBuffer in this specification. Service names may be different from capabilities above.

9.6.2.1.2 Inherent data is used to complete the responsibility. The data is also referred to as attributes and they have to be defined in some way. This document doesn't define attributes directly with attribute definition table shown in some SEMI standard documents but defines them indirectly as service parameters.

INACTIVE

Documents Document Sharing Hub
Full document details: SEMI T12-0710

9.6.2.1.3 Behavior is also defined to complete the responsibility. Behavior is often represented by collaboration with outside of this class and state model. Collaboration reflects a part of associations of this class with others. Collaboration works by asking services defined on accompanying object which is the other end of association on class diagram. State model is representation of the other part of behavior of this class. Because this class has explicit state, it defines following states and state model by state chart and transition definition table.

9.6.2.1.3.1 *Empty* — No history information is accumulated.

9.6.2.1.3.2 *Buffered* — Some history information are accumulated and they are not reported.

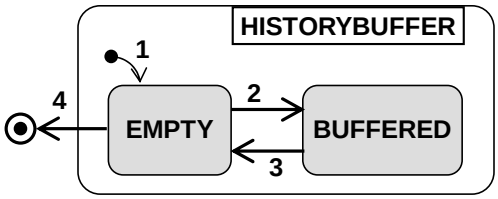

Figure 8
HistoryBuffer State

Table 3 Transition Definition Table of HistoryBuffer State

#	*Previous State*	*Trigger*	*New State*	*Actions*	*Comments*
1	(No state)	A Secondment has been created.	EMPTY		
2	EMPTY	History information is put.	BUFFERED		The information has to be reported at some moment.
3	BUFFERED	History information is reported and buffer area is cleared.	EMPTY	The information has to be reported before it is cleared.	
4	EMPTY	Corresponding Attachment has been removed.	(No state)		

9.7 *History Data Base* — HistoryDB is an interface of whole history information for application.

9.7.1 *History Data Base object* — History Data Base object keeps history data of all Jigs and Implements in a Inventory. History Data Base object is an instance of History Data Base class defined later. There is just one History Data Base object corresponding to an Inventory. This object is abbreviated as "HistoryDB" in this document and often 'object' is omitted.

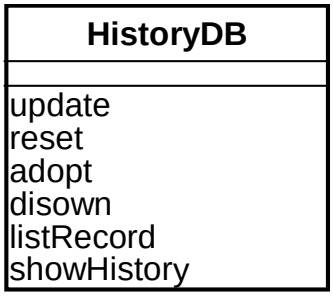

Figure 9
HistoryDB Class

9.7.2 *History Data Base Class* — HistoryDB class defines its inherent data, behavior and services. This class manages history of all Jigs and Implements in an Inventory. This class is illustrated as above for reference.

INACTIVE

Documents
Document Sharing Hub
Full document details: SEMI T12-0710

9.7.2.1 *Responsibility of HistoryDB* — This class is responsible for managing and organizing history information for all Jigs and/or Implements registered in an Inventory.

9.7.2.1.1 To complete the responsibility it accepts adoption/reset of Attachment history, updating requests and inquiry. They are defined as services of HistoryDB in this specification. Service names may be different from capabilities above.

9.7.2.1.2 Inherent data is used to complete the responsibility. The data is also referred to as attributes and they have to be defined in some way. This document doesn't define attributes directly with attribute definition table shown in some SEMI standard documents but defines them indirectly as service parameters.

9.7.2.1.3 Behavior is also defined to complete the responsibility. Behavior is often represented by collaboration with outside of this class and state model. Collaboration reflects a part of associations of this class with others. Collaboration works by asking services defined on accompanying object which is the other end of association on class diagram. State model is representation of the other part of behavior of this class. However it has to be defined if this class has explicit state, state model is not defined here because this class is stateless.

9.8 *HistoryRecord* — HistoryRecord is a record for each Attachment registered in Inventory.

9.8.1 *HistoryRecord Object* — HistoryRecord object keeps history data of a certain Jig or Implement. HistoryRecord object is an instance of HistoryRecord class defined later. HistoryRecord objects are aggregated by a HistoryDB object. This object is abbreviated as "HistoryRecord" in this document and often 'object' is omitted.

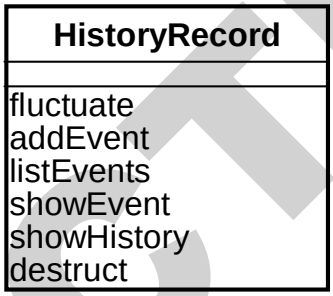

Figure 10
HistoryRecord Class

9.8.2 *HistoryRecord Class* — HistoryRecord class defines its inherent data, behavior and services. This class manages history of a Jig or Implement belonging to a HistoryDB. This class is illustrated as above for reference.

9.8.2.1 *Responsibility of HistoryRecord* — This class is responsible to hold and organize history information for a Jig or Implement belonging to a HistoryDB.

9.8.2.1.1 To complete the responsibility it accepts removal of a HistoryRecord, adding events or increment time/duration requests and inquiry. They are defined as services of HistoryRecord in this specification. Service names may be different from above capabilities.

9.8.2.1.2 Inherent data is used to complete the responsibility. The data is also referred to as attributes and they have to be defined in some way. This document doesn't define attributes directly with attribute definition table shown in some SEMI standard documents but defines them indirectly as service parameters.

9.8.2.1.3 Behavior is also defined to complete the responsibility. Behavior is often represented by collaboration with outside of this class and state model. Collaboration reflects a part of associations of this class with others. Collaboration works by asking services defined on accompanying object which is the other end of association on class diagram. State model is representation of the other part of behavior of this class. However it has to be defined if this class has explicit state, state model is not defined here because this class is stateless.

9.9 *Inventory* — There is an Inventory to track or manage all such Attachments as Jigs and Implements in a factory. The Inventory may be decomposed into a couple of sub-inventories for Jigs and Implements respectively. Or it may be divided into a few sub-inventories by type or category for convenience of managing people. Even if so an inventory of a type of Attachments used for specific type of equipment is required to be seen as one whole inventory

INACTIVE

Documents Document Sharing Hub
Full document details: SEMI T12-0710

by inventory user. Because Inventory is a logical matter rather than a physical one, some of the Attachments can be shared by another inventory for different types of equipment and both inventories can be managed without any confusion.

9.9.1 *Inventory Object* — The Inventory is treated as Inventory object in most part of this specification. It may be written sometimes as "Inventory" for short but it must be started with the capital letter 'I'. Inventory object is an instance of Inventory class which is defined later. Tracking system has just one Inventory object.

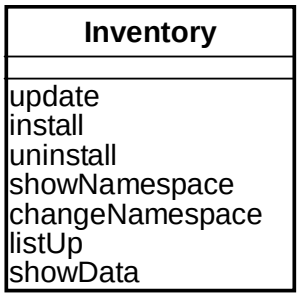

Figure 11
Inventory Class

9.9.2 *Inventory Class* — Inventory Class defines its inherent data, behavior and services. This class manages all Jigs and Implements in certain area. The area here may not be always special but may be logical in equipment type, attachment type, supplier or nay combination of such category. This class is illustrated as above for reference.

9.9.2.1 *Responsibility of Inventory* — This class is responsible for managing and organizing registered Jigs and/or Implements.

9.9.2.1.1 To complete the responsibility it accepts registration/unregistration of attachments, updating requests and inquiry. They are defined as services of Inventory in this specification. Service names may be different from above capabilities.

9.9.2.1.2 Inherent data is used to complete the responsibility. The data is also referred to as attributes and they have to be defined in some way. This document doesn't define attributes directly with attribute definition table shown in some SEMI standard documents but defines them indirectly as service parameters.

9.9.2.1.3 Behavior is also defined to complete the responsibility. Behavior is often represented by collaboration with outside of this class and state model. Collaboration reflects a part of associations of this class with others. Collaboration works by asking services defined on accompanying object which is the other end of association on class diagram. State model is representation of the other part of behavior of this class. However it has to be defined if this class has explicit state, state model is not defined here because this class is stateless.

9.10 *Invoice* — Invoice is a set of information for such attachment as Jig or an Implement which is to be transported to equipment. The information set is sent to the equipment before or after actual Attachment has arrived. If there is something wrong with the Attachment or the information on the equipment, equipment ask for new one. When another Attachment or information has arrives, the equipment checks it again.

9.10.1 *Invoice Object* — Invoice object is created at the time a set of invoice information has received at equipment. If the equipment has more than one load port for the Attachment, port ID may be specified in the information set. If port ID is not specified, destination port must be anonymous. If more than one attachment IDs or Load Ports are specified, they are candidate but they are not ordering information nor correspondence between a couple of sets. However there must be correspondent between Attachments and histories. This object is removed at the time all information has been verified regardless of the result. Invoice object is an instance of Invoice class defined later. This object is abbreviated as "Invoice" in this document and often 'object' is omitted.

INACTIVE

Documents
Document Sharing Hub
Full document details: SEMI T12-0710

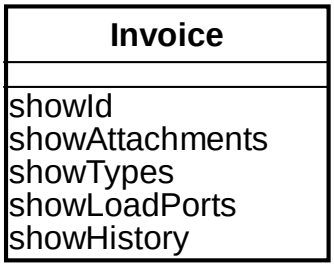

Figure 12
Invoice Class

9.10.2 *Invoice Class* — Invoice Class defines its inherent data, behavior and services. This class provides information about Jig or Implement to be tracked on equipment. This class is illustrated as above for reference.

9.10.2.1 *Responsibility of Invoice* — This class is responsible for holding and organizing invoice information as well as verification. This object verifies physical attachments with invoice information at the time this object has been created with unverified attachments on equipment or new physical attachment arrives.

9.10.2.1.1 To complete the responsibility it accepts inquiries of inherent information. They are defined as services of Invoice in this specification. Service names may be different from above capabilities.

9.10.2.1.2 Inherent data is used to complete the responsibility. The data is also referred to as attributes and they have to be defined in some way. This document doesn't define attributes directly with attribute definition table shown in some SEMI standard documents but defines them indirectly as service parameters.

9.10.2.1.3 Behavior is also defined to complete the responsibility. Behavior is often represented by collaboration with outside of this class and state model. Collaboration reflects a part of associations of this class with others. Collaboration works by asking services defined on accompanying object which is the other end of association on class diagram. State model is representation of the other part of behavior of this class. However it has to be defined if this class has explicit state, state model is not defined here because this class is stateless.

9.11 *JitMachine* — JitMachine is a piece of equipment with full tracking capability of Jigs and/or Implements. JitMachie collaborate with Inventory so that the Inventory fulfills its responsibility.

9.11.1 *JitMachine Object* — JitMachine object represents a piece of equipment which completes responsibility for tracking such attachments as Jigs and/or Implements on equipment. It verifies passed attachments whether they are right ones to be used on the equipment or not even if they are expected type or compatible. It also let the inventory know that certain attachments have arrived at the equipment or left for the Inventory. JitMachine object is an instance of JitMachine class defined later. This object is abbreviated as "JitMachi" in this document and often 'object' is omitted.

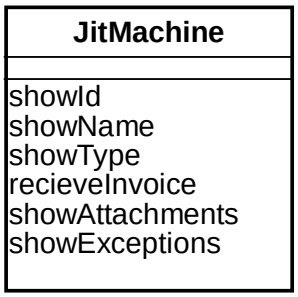

Figure 13
JitMachine Class

9.11.2 *JitMachine Class* — JitMachine Class defines its inherent data, behavior and services. This class is illustrated as above for reference.

INACTIVE

Documents Document Sharing Hub
Full document details: SEMI T12-0710

9.11.2.1 *Responsibility of JitMachine* — This class is responsible for verifying attachments, reporting in/out of equipment and organizing Jigs or Implements.

9.11.2.1.1 To complete the responsibility it accepts invoice which shows what are delivered attachments and related requests. They are defined as services of JitMachine in this specification. Service names may be different from capabilities above.

9.11.2.1.2 Inherent data is used to complete the responsibility. The data is also referred to as attributes and they have to be defined in some way. This document doesn't define attributes directly with attribute definition table shown in some SEMI standard documents but defines them indirectly as service parameters.

9.11.2.1.2.1 Behavior is also defined to complete the responsibility. Behavior is often represented by collaboration with outside of this class and state model. Collaboration reflects a part of associations of this class with others. Collaboration works by asking services defined on accompanying object which is the other end of association on class diagram. State model is representation of the other part of behavior of this class. Because this class has explicit state, it defines following states and state model by state chart and transition definition table.

9.11.2.1.2.2 *INSERVICE* — JitMachine is available to track Attachments.

9.11.2.1.2.3 *OUTOFSERVICE* — JitMachine is not available to track Attachments.

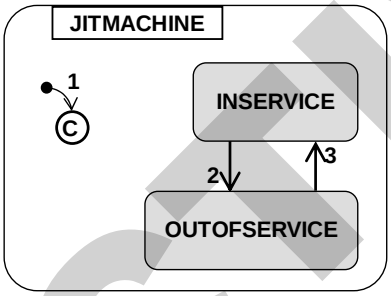

Figure 14
JitMachine State

Table 4 Transition Definition Table for JitMachine

#	*Previous State*	*Trigger*	*New State*	*Actions*	*Comments*
1	(No state)	JitMachine has been initialized.	INSERVICE [or] OUTOFSERVICE		
2	INSERVICE	Problem happens on equipment to prevent from tracking including users operation to suspend.	OUTOFSERVICE		Service except inquiry may not be accepted.
3	OUTOFSERVICE	Problem to prevent equipment from tracking is removed including users operation to resume.	INSERVICE		

9.12 *Public Location* — Public Location represents a location of a Jig or an Implement on the tracking system. This is just for official location other than such private location as detail position in equipment. A public Location may have more than one Jigs and/or Implements.

9.12.1 *Public Location Object* — Public Location object represents an official location potentially occupied by such supporting resource as a Jig or an Implement and identifies what occupant is if it is occupied. Public Location object is an instance of Public Location class defined later. This object doesn't identify physical location in equipment. This object is abbreviated as "PublicLocation" in this document and often 'object' is omitted.

9.12.1.1 Public Location object is assigned on a piece of equipment and represent one and only one official location of the equipment. However a piece of equipment may have more than one Public Location objects, they have to be

INACTIVE

Documents
Document Sharing Hub
Full document details: SEMI T12-0710