Table 20 Map Upload Parameter Dictionary

Parameter Name	*Description*	*Format: Possible values*
MapData	The MapData to upload. The MapData may contain one or more substrates.	Text. Other standards that define an implementation of this service and reference this standard will define how the MapData is represented in text.

Table 21 Map Upload Service Definition

Parameter	*Req/Ind*	*Comment*
MapData	M	The MapData to be uploaded. The MapData may contain one or more substrates.

APPENDIX 1
WAFER MAP

NOTICE: The material in this appendix is an official part of SEMI E142 and was approved by full letter ballot procedures on December 10, 2004.

A1-1 This appendix shows how the map data items may be applied to wafers as well as some additional information specific to this substrate type. See Figures A1-1 through A1-8.

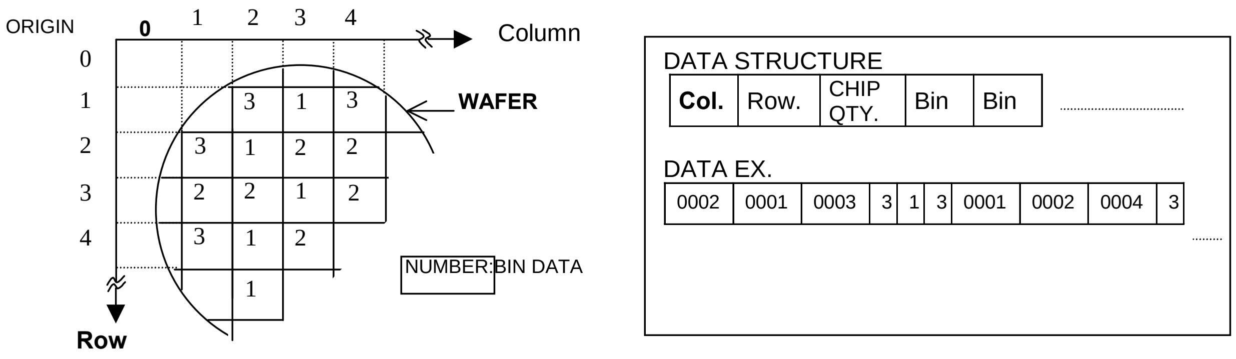

Figure A1-1a
Row/Column Format

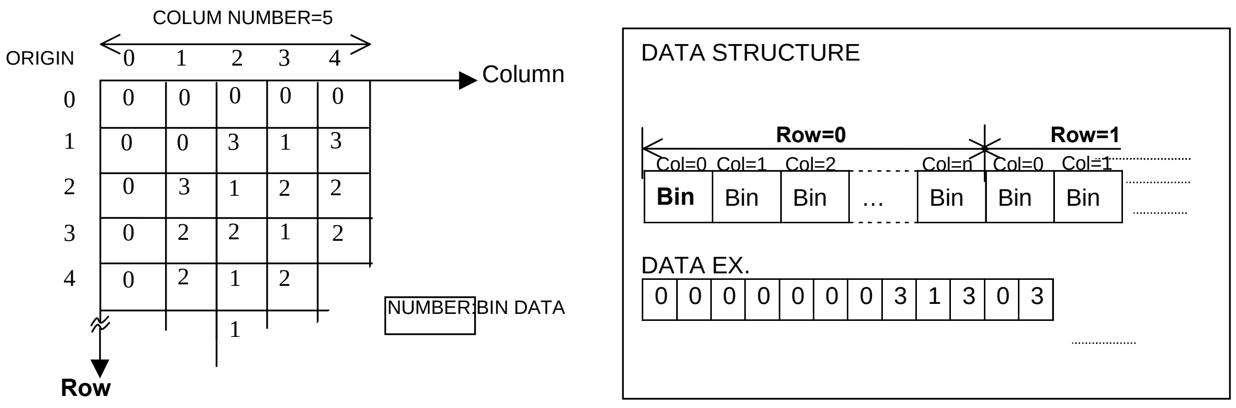

Figure A1-1b
Array Format

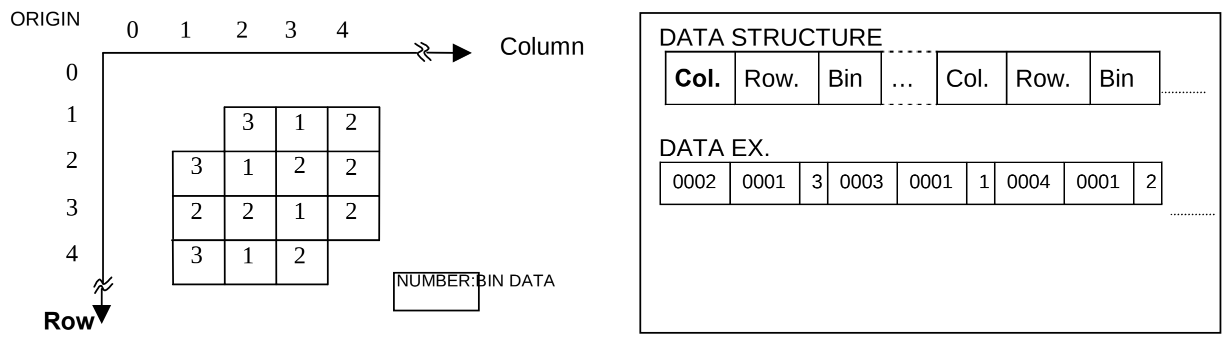

Figure A1-1c
Coordinate Format

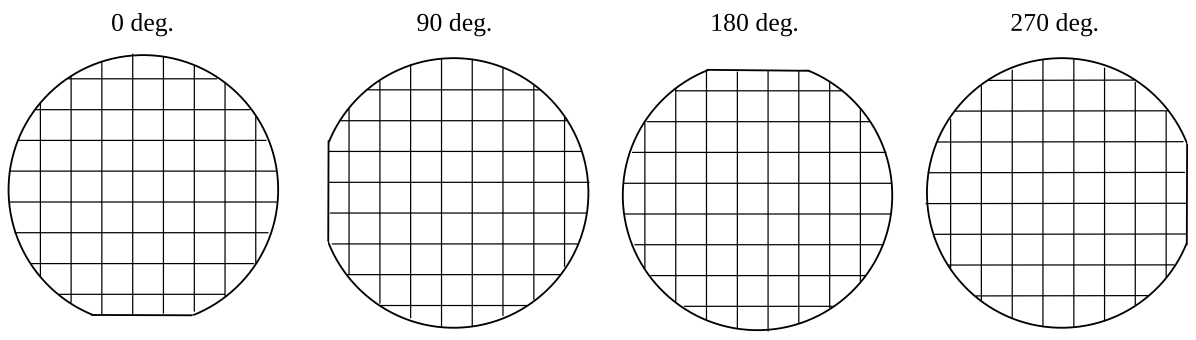

Figure A1-2
Orientation of the Wafer Flat or Notch (Orientation)

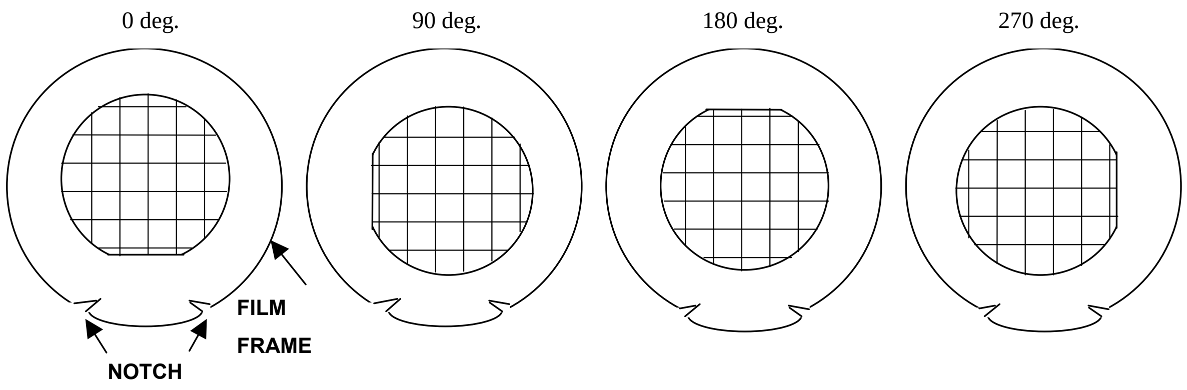

Figure A1-3
Orientation of the Wafer Flat or Notch on the Film Frame

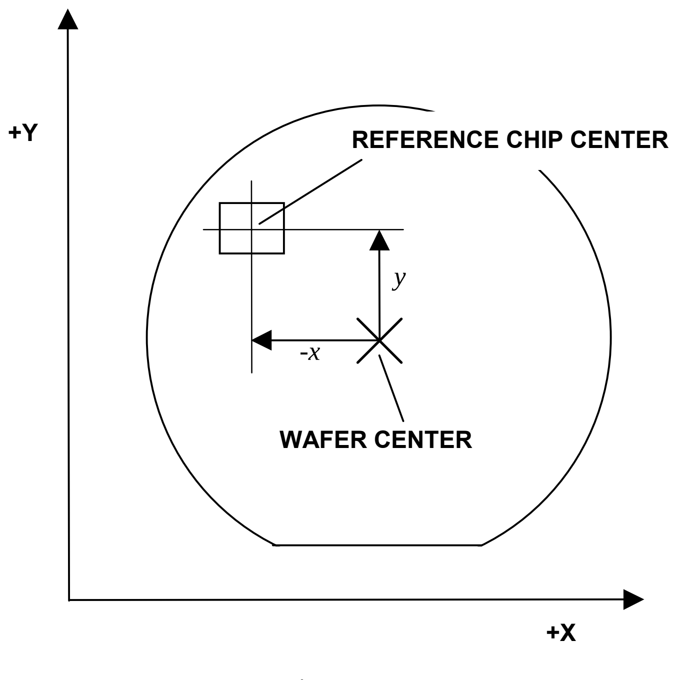

Figure A1-4
Reference Device Position

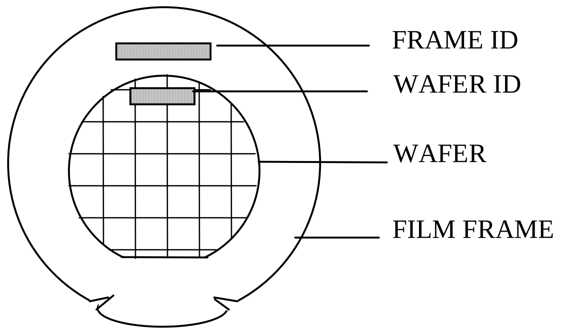

Figure A1-5
Example of WaferID and FrameID Position

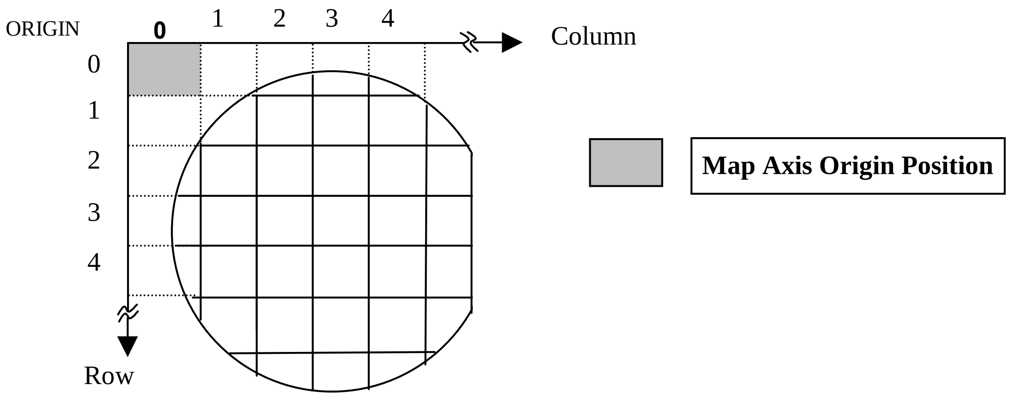

Figure A1-6
(Ex.) Map Coordinates System (OriginLocation = Top Left Top Side)

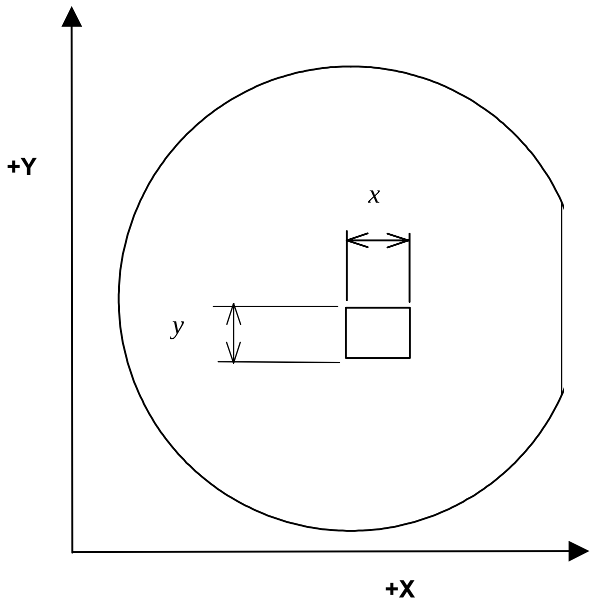

Figure A1-7
Device Size Coordinates

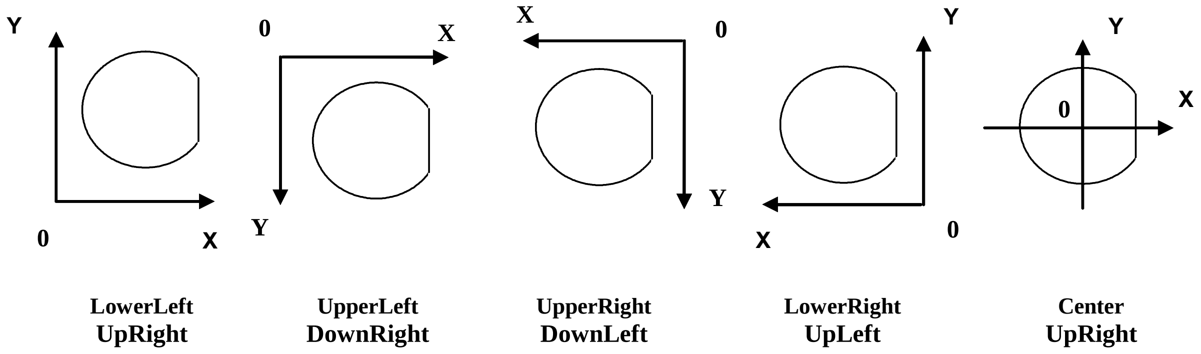

Figure A1-8
Wafer Coordinate to Address Devices (OriginLocation)

APPENDIX 2
STRIP MAP

NOTICE: The material in this appendix is an official part of SEMI E142 and was approved by full letter ballot procedures on December 10, 2004.

A2-1 This appendix shows how the map data items may be applied to strips as well as some additional information specific to this substrate type.

A2-2 The StripID shall follow the SEMI T9 specification, when applicable, or may be user definable by the factory.

A2-3 For any given process there must be a way to ensure correct orientation of the strip. The end user is responsible for providing a master manufacturing drawing showing the top side of the strip. This drawing must show any special markings or patterns that are needed to reliably flip and rotate a physical strip until it is oriented the same as the drawing. Figure A2-1 defines what is meant by top side and hence bottom side. It also defines what is meant by upper, lower and left and right.

A2-4 Each strip type will have a factory defined value for OriginLocation. This is the origin reference for the row 1, column 1 location on the strip. The factory origin, OriginLocation is selected as one of four corners by the factory host system. The row and column index will be referenced from the selected reference. This information will be provided to the equipment by the host in each strip map download scenario.

1 = Upper right (UR) top side

2 = Upper left (UL) top side

3 = Lower left (LL) top side

4 = Lower right (LR) top side

6 = Upper right (UR) bottom side

7 = Upper left (UL) bottom side

8 = Lower left (LL) bottom side

9 = Lower right (LR) bottom side

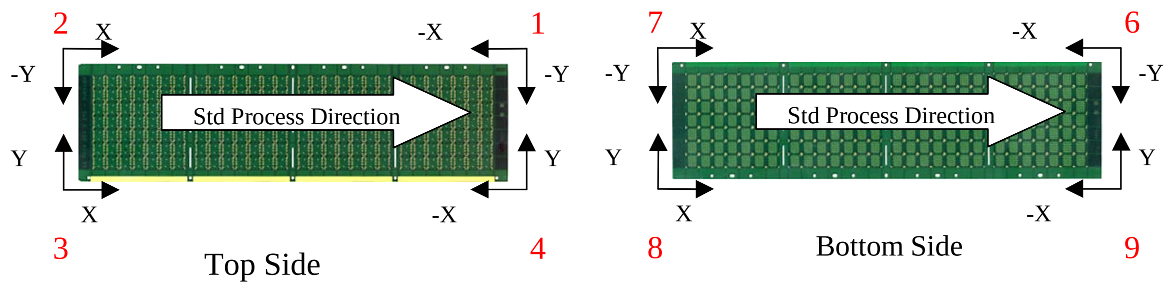

Figure A2-1
Strip OriginLocation

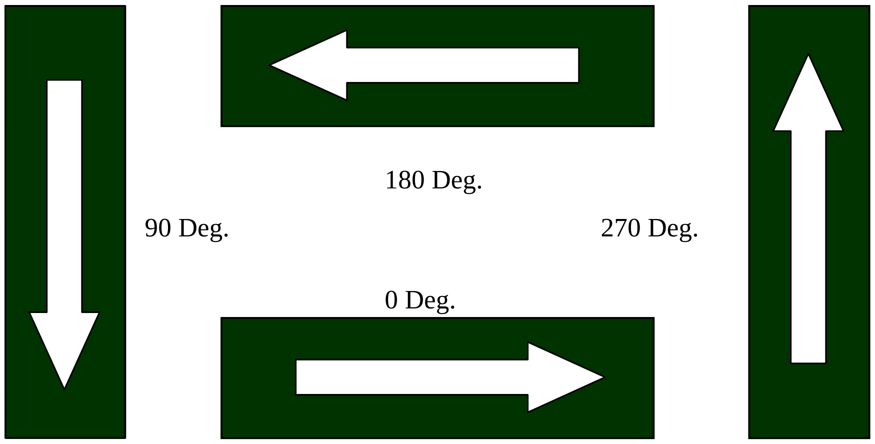

Figure A2-1
Strip Orientation

APPENDIX 3
TRAY MAP ORIENTATION

NOTICE: The material in this appendix is an official part of SEMI E142 and was approved by full letter ballot procedures on December 10, 2004.

A3-1 This appendix shows how the map data items may be applied to JEDEC trays as well as some additional information specific to this substrate type.

A3-2 The X-axis (Column) is assigned to the long axis of the JEDEC tray; the Y-axis (Row) is assigned to the short axis of the tray. This corresponds to Cartesian coordinate "1". For the origin of the tray, the pocket nearest the bevel (at the bottom left corner of the tray in Figure A3-1 below) is defined as coordinate location 0,0. (Row/Column).

A3-3 The following values for OriginLocation apply to the JEDEC tray:

1 = Upper right (UR) top side

2 = Upper left (UL) top side

3 = Lower left (LL) top side

4 = Lower right (LR) top side

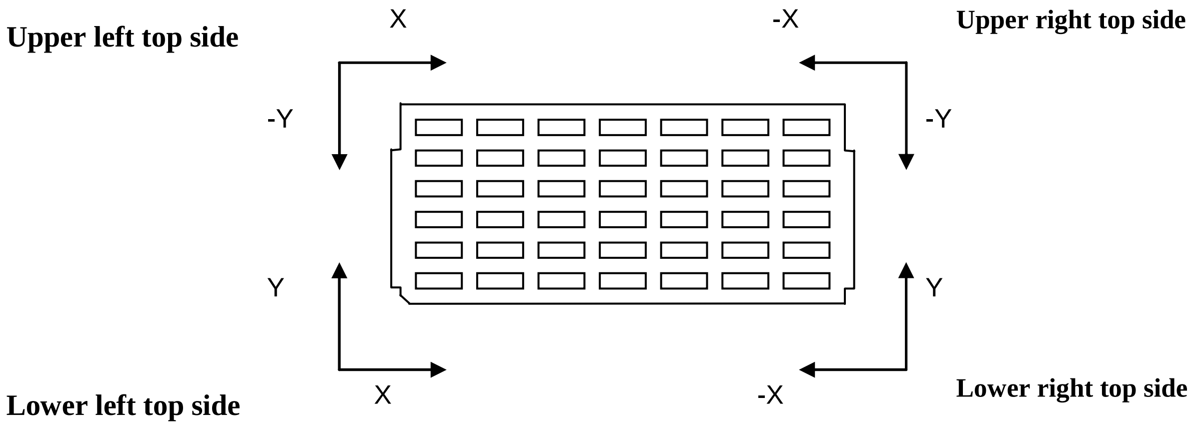

Figure A3-1
OriginLocation

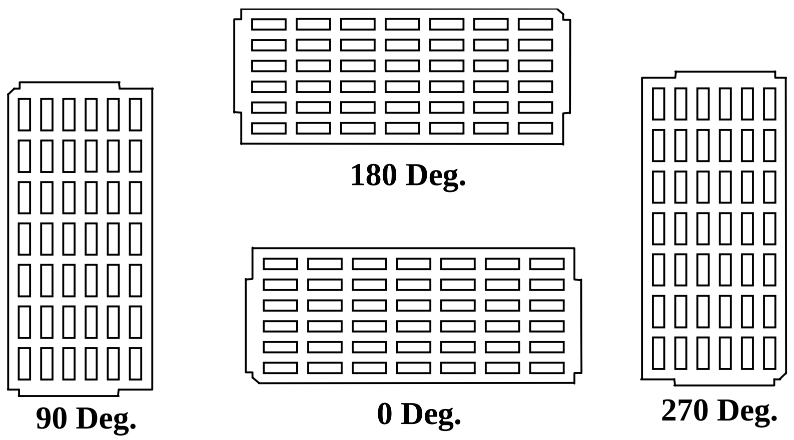

Figure A3-2
Tray Orientation

RELATED INFORMATION 1
MAP EXAMPLES

NOTICE: This related information is not an official part of SEMI E142 and was derived from the North American Information & Control Committee. This related information was approved for publication by full letter ballot procedures on December 10, 2004.

R1-1 Wafer Map Example

R1-1.1 The example MapData below is for three wafers that all share the same layout. The layout includes an FDI target, used for alignment, and an array of devices.

Wafer1 includes ReferenceDevices and BinDefinitions. Wafer2 and Wafer3 do not.

Wafer1 uses the Row/Column map representation type including null bin devices.

Wafer2 uses the Row/Column map representation type excluding null bin devices.

Wafer3 uses the Array map representation type.

Wafer4 uses the Coordinate map representation type.

```
<?xml version="1.0" encoding="utf-8" ?>
<MapData xmlns="urn:semi-org:xsd.4032.V0804.SubstrateMap" >
    <Layouts>
        <Layout LayoutId="WaferLayout" DefaultUnits="mm" TopLevel="true" >
            <DeviceSize X="300" Y="300" />
            <Dimension X="1" Y="1" />
            <ChildLayouts>
                <ChildLayout LayoutId="FDI Target" />
                <ChildLayout LayoutId="Devices" />
            </ChildLayouts>
        </Layout>
        <Layout LayoutId="FDI Target" DefaultUnits="microns" >
            <Dimension X="1" Y="1" />
            <LowerLeft X="0" Y="0" />
            <DeviceSize X="0" Y="0" />
        </Layout>
        <Layout LayoutId="Devices" DefaultUnits="microns" >
            <Dimension X="4" Y="3" />
            <LowerLeft X="0" Y="0" />
            <DeviceSize X="0" Y="0" />
            <ProductId>Product1</ProductId>
        </Layout>
    </Layouts>
    <Substrates>
        <Substrate SubstrateType="Wafer" SubstrateId="Wafer1" >
            <LotId>Lot1</LotId>
            <AliasIds>
                <AliasId Type="FrameId" Value="Frame1" />
            </AliasIds>
        </Substrate>
        <Substrate SubstrateType="Wafer" SubstrateId="Wafer2" />
        <Substrate SubstrateType="Wafer" SubstrateId="Wafer3" />
        <Substrate SubstrateType="Wafer" SubstrateId="Wafer4" />
    </Substrates>
    <SubstrateMaps>
        <SubstrateMap SubstrateType="Wafer" SubstrateId="Wafer1"
                      LayoutSpecifier="WaferLayout/FDI Target" >
            <Overlay>
                <ReferenceDevices>
                    <ReferenceDevice Name="FDI Target" >
                        <Coordinates X="0" Y="0" />
                    </ReferenceDevice>
                </ReferenceDevices>
            </Overlay>
        </SubstrateMap>
        <SubstrateMap SubstrateType="Wafer" SubstrateId="Wafer1"
                      LayoutSpecifier="WaferLayout/Devices" >
            <Overlay MapName="SortGrade" MapVersion="1" >
                <ReferenceDevices>
                    <ReferenceDevice Name="FirstDevice" >
                        <Coordinates X="1" Y="2" />
                    </ReferenceDevice>
                </ReferenceDevices>
                <BinCodeMap BinType="Ascii" NullBin="." >
                    <BinDefinitions>
                        <BinDefinition BinCode="1" BinCount="5" BinQuality="Pass"
                        BinDescription="Tested Ok" Pick="true" />
                        <BinDefinition BinCode="2" BinCount="3" BinQuality="Fail"
                        BinDescription="Test Failed" Pick="false" />
                    </BinDefinitions>
                    <BinCode>.12.</BinCode>
                    <BinCode>1112</BinCode>
                    <BinCode>.21.</BinCode>
                </BinCodeMap>
            </Overlay>
        </SubstrateMap>
        <SubstrateMap SubstrateType="Wafer" SubstrateId="Wafer2"
                        LayoutSpecifier="WaferLayout/Devices" >
            <Overlay MapName="SortGrade" MapVersion="1" >
                <BinCodeMap BinType="Ascii" NullBin="." >
                    <BinCode X="1" Y="2">12</BinCode>
                    <BinCode X="0" Y="1">1112</BinCode>
                    <BinCode X="1" Y="0">21</BinCode>
                </BinCodeMap>
            </Overlay>
        </SubstrateMap>
        <SubstrateMap SubstrateType="Wafer" SubstrateId="Wafer3"
                        LayoutSpecifier="WaferLayout/Devices" >
            <Overlay MapName="SortGrade" MapVersion="1" >
                <BinCodeMap BinType="Ascii" NullBin="." >
                    <BinCode>.12.1112.21.</BinCode>
                </BinCodeMap>
            </Overlay>
        </SubstrateMap>
        <SubstrateMap SubstrateType="Wafer" SubstrateId="Wafer4"
                        LayoutSpecifier="WaferLayout/Devices"
        Orientation="180" >
            <Overlay MapName="SortGrade" MapVersion="1" >
                <BinCodeMap BinType="Ascii" NullBin="." >
                    <BinCode X="1" Y="2">1</BinCode>
                    <BinCode X="2" Y="2">2</BinCode>
                    <BinCode X="0" Y="1">1</BinCode>
                    <BinCode X="1" Y="1">1</BinCode>
                    <BinCode X="2" Y="1">1</BinCode>
                    <BinCode X="3" Y="1">2</BinCode>
                    <BinCode X="1" Y="0">2</BinCode>
                    <BinCode X="2" Y="0">1</BinCode>
                </BinCodeMap>
            </Overlay>
        </SubstrateMap>
    </SubstrateMaps>
 </MapData>
```

R1-2 Strip Map Example

R1-2.1 The example strip map below contains a more complex nested layout, one strip substrate, and three data maps.

```
<?xml version="1.0" encoding="utf-8" ?>
<MapData xmlns="urn:semi-org:xsd.4032.V0804.SubstrateMap">
    <Layouts>
        <Layout LayoutId="StripLayout"  DefaultUnits="100Microns" >
            <Dimension X="1" Y="1" />
            <DeviceSize X="2510.00" Y="414.02" />
            <ChildLayouts>
                <ChildLayout LayoutId="SRAM" />
            </ChildLayouts>
        </Layout>
        <Layout LayoutId="SRAM" DefaultUnits="100Microns">
            <Dimension X="10" Y="3" />
            <LowerLeft X="7.00" Y="8.33" />
            <DeviceSize X="75.00" Y="46.67" />
            <StepSize X="78.00" Y="49.67" />
            <Z Order="0" Height="500"  Units="microns" />
            <ChildLayouts>
                <ChildLayout LayoutId="FLASH" />
            </ChildLayouts>
        </Layout>
        <Layout LayoutId="FLASH" DefaultUnits="100Microns">
```

```
            <Dimension X="2" Y="1" />
            <LowerLeft X="10.00" Y="6.67" />
            <DeviceSize X="25.00" Y="33.00" />
            <StepSize X="30.00" Y="0.00" />
            <Z Order="1" Height="300" Units="microns" />
        </Layout>
    </Layouts>
    <Substrates>
        <Substrate SubstrateType="Strip" SubstrateId="Strip1" />
    </Substrates>
    <SubstrateMaps>
        <SubstrateMap SubstrateType="Strip" SubstrateId="Strip1"
                        LayoutSpecifier="StripLayout/SRAM"
        Orientation="180">
            <Overlay MapName="SortGrade" MapVersion="1" >
                <BinCodeMap BinType="Ascii" NullBin="." >
                    <BinDefinitions>
                        <BinDefinition BinCode="1" BinCount="24" BinQuality="Pass"
                        BinDescription="Bond Ok" Pick="true" />
                        <BinDefinition BinCode="2" BinCount="3" BinQuality="Fail"
                        BinDescription="Bond Failed" Pick="false" />
                    </BinDefinitions>
                    <BinCode>.111121111</BinCode>
                    <BinCode>.111111121</BinCode>
                    <BinCode>.112111111</BinCode>
                </BinCodeMap>
            </Overlay>
            <Overlay MapName="2D Matrix Mark" MapVersion="1" >
                <DeviceIdMap>
                    <Id X="0" Y="7">Device1</Id>
                    <Id X="0" Y="5">Device2</Id>
                </DeviceIdMap>
            </Overlay>
            <Overlay MapName="WaferToStrip" MapVersion="1" >
                <TransferMap FromSubstrateType="Wafer" FromSubstrateId="Wafer1" >
                    <T FX="1" FY="2" TX="1" TY="0"/>
                    <T FX="0" FY="1" TX="1" TY="1"/>
                </TransferMap>
            </Overlay>
        </SubstrateMap>
    </SubstrateMaps>
</MapData>
```

R1-2.2 Figure R1-1 shows a portion of the layout for the strip in the example above. The strip contains “SRAM” devices with “Flash” devices stacked on top of them.

R1-2.3 Figure R1-2 illustrates one DeviceId element of the map.

R1-2.4 Figure R1-3 illustrates one Transfer element of the map.

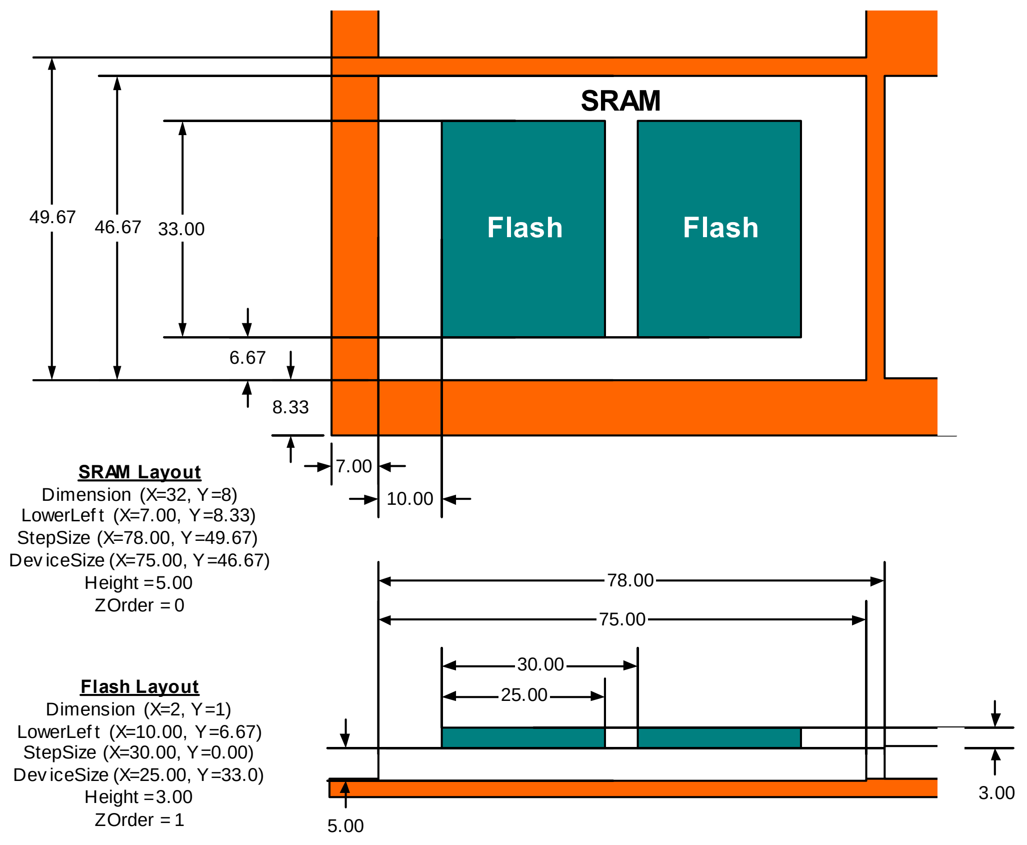

Figure R1-1
Strip Map Example – Layout

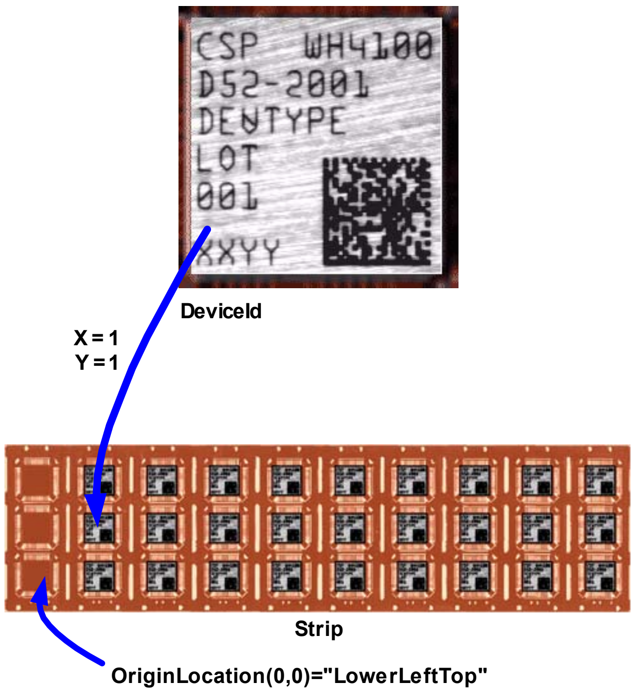

Figure R1-2
Strip Map Example – DeviceIdMap

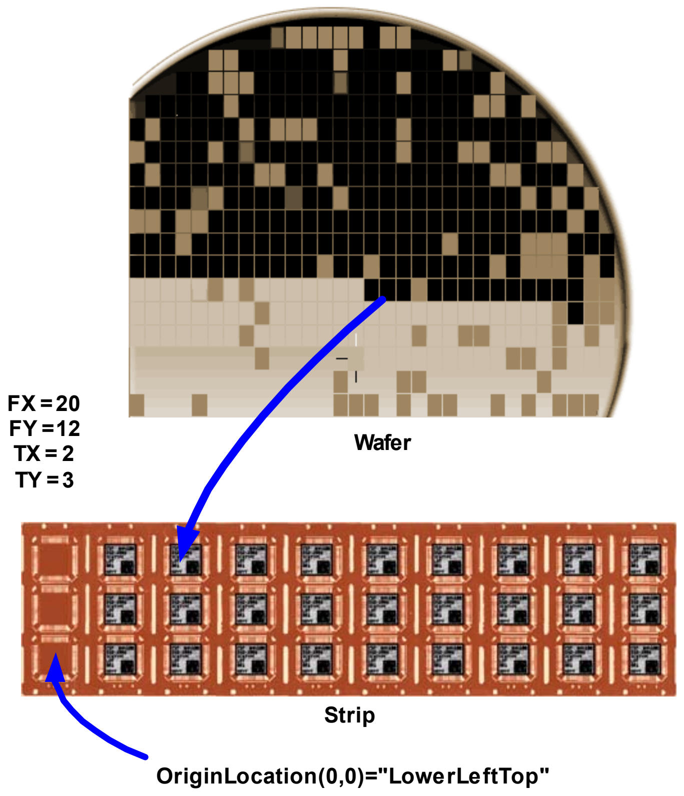

Figure R1-3
Strip Map Example – TransferMap

NOTICE: SEMI makes no warranties or representations as to the suitability of the standards set forth herein for any particular application. The determination of the suitability of the standard is solely the responsibility of the user. Users are cautioned to refer to manufacturer's instructions, product labels, product data sheets, and other relevant literature, respecting any materials or equipment mentioned herein. These standards are subject to change without notice.

By publication of this standard, Semiconductor Equipment and Materials International (SEMI) takes no position respecting the validity of any patent rights or copyrights asserted in connection with any items mentioned in this standard. Users of this standard are expressly advised that determination of any such patent rights or copyrights, and the risk of infringement of such rights are entirely their own responsibility.

SEMI E142.1-0705
XML SCHEMA FOR SUBSTRATE MAPPING

This standard was technically approved by the global Information & Control Committee. This edition was approved for publication by the global Audits and Reviews Subcommittee on April 7, 2005. It was available at www.semi.org in June 2005 and on CD-ROM in July 2005.

1 Purpose

1.1 The purpose of this specification is to provide an XML schema that corresponds to the data model for equipment defined by SEMI E142, Specification for Substrate Mapping. This schema is available to equipment communication standards that want to use SEMI E142 -defined information in XML formatted equipment communication.

2 Scope

2.1 The scope of this document is the representation of the SEMI E142 object model in an XML schema. It will not add new domain information or concepts to the model. The only additions made are those needed to render a useful XML schema.

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 Referenced Standards and Documents

3.1 *SEMI Standards*

SEMI E142 — Specification for Substrate Mapping

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

3.2 *OMG Standards*

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

3.3 *W3C Standards*

Extensible Markup Language (XML) 1.0 (Second Edition) — W3C, 6 October 2000 (http://www.w3.org/TR/2000/REC-xml-20001006/).

Namespaces in XML — W3C, 14 January 1999 (http://www.w3.org/TR/1999/REC-xml-names-19990114/).

XML Schema Part 0: Primer — W3C, 2 May 2001 (http://www/w3/org/TR/xmlschema-0/).

XML Schema Part 1: Structures — W3C, 2 May 2001 (http://www/w3/org/TR/xmlschema-1/).

XML Schema Part 2: Datatypes — W3C, 2 May 2001 (http://www/w3/org/TR/xmlschema-2/).

XML Path Language (Xpath) — W3C, 16 November 1999 (http://www/w3/org/TR/xpath/).

4 Terminology

4.1 *Abbreviations and Acronyms*

4.2 *UML* — Unified Modeling Language

4.3 *W3C* — World Wide Web Consortium

4.4 *XML* — eXtensible Markup Language

4.5 *Definitions*

4.5.1 *UML (Unified Modeling Language)* — A notation for representing object-oriented designs and views created by Booch, Rumbaugh, and Jacobson in order to merge their three popular notations plus aspects of other existing notations into a single object-oriented notation intended to be usable by all.

4.5.2 *XML (eXtensible Markup Language)* — A markup language used for representing data rich with context and content in documents and in communications. XML is an extension of SGML, a document-oriented markup language. It was created by W3C for use on the Internet. XML can represent object-oriented structures.

5 Conventions

5.1 This section discusses the conventions used in this specification for translating UML to XML and for documenting the XML. These conventions are heavily influenced by the SEMI E121 Guide for Style & Usage of XML for Semiconductor Manufacturing Applications.

5.2 The reader is expected to have a working knowledge of the UML, XML, and Schema specifications (See ¶3.2 and ¶3.3). This document does not provide tutorial information on these subjects.

5.3 *Translating UML to XML*

5.3.1 This document follows the guidelines for XML as outlined in SEMI E121 "Guide For Style & Usage of XML for Semiconductor Manufacturing Applications".

5.3.2 The translation of a UML class to XML is documented using a table format illustrated by Table 1.

5.4 *Translation Table Column Header Description*

5.4.1 *Attribute or Role Name* — If an attribute, the name of the attribute is placed here. If an association (including aggregation or composition), the role name from the UML diagram is placed here. Compositions are often not assigned role names. In that case "none" is placed here.

5.4.2 *UML Name/Type* — If an attribute, the data type of the UML attribute is placed here. If an association, the type of association is placed here. The possible types are "Composition", "Aggregation", or the basic "Association". UML defines these three types of associations.

5.4.3 *XML Element or Attribute* — Lists the type of XML construct used to represent the UML attribute or association.

5.4.4 *XML Name/Type* — Provides the name and data type of the resulting XML construct. The type may be a built-in type (for example, xs:string), or a named type defined within the XML schema.

Table 1 Example Translation Table

Attribute or Role Name	*UML Name/Type*	*XML Element or Attribute*	*XML Name/Type*
friend	association	element	Friend: HumanReferenceArray
employees	aggregation	element	Employees: HumanArray
family	composition	element	Family: HumanArray
name	string	element	Name: xs:string

6 XML Schema

6.1 The actual XML schema defined by this specification is contained in a separate document. The contents of the schema document constitute the core part of this specification.

6.2 The target namespace for the schema is "urn:semi-org:xsd.E142-1.V0105.SubstrateMap".

6.3 Table 2 describes the complex types defined in the schema and how they relate to one another.

6.4 Table 3 describes the simple types defined in the schema.

6.5 Table 4 shows how the element and attribute names in the schema map to the attributes defined in SEMI E142 Substrate Mapping.

Table 2 Complex Elements and Types Defined in Schema

Element/Type	*Description*
MapData	Top level MapDataType element.
MapDataType	Top level element that contains zero or one LayoutsType elements, zero or one SubstratesType elements and zero or one SubstrateMapsType elements.
LayoutsType	A collection of one or more LayoutType elements. The uniqueness of the LayoutId is enforced by the xs:key LayoutKey defined in MapData.
LayoutType	An element containing the attributes defined for the Layout object in SEMI E142.
LogicalCoordinatesType	An element containing the attributes defined for the LogicalCoordinates object in SEMI E142.
PhysicalCoordinatesType	An element containing the attributes defined for the PhysicalCoordinates object in SEMI E142.
ZDimensionsType	An element containing the attributes defined for the ZDimensions object in SEMI E142.
ChildLayoutsType	A collection one or more ChildLayoutType elements.
ChildLayoutType	A element containing a reference (LayoutId) of the child layout. This relationship is enforced by the xs:KeyRef ChildLayoutRef defined in MapData.
SubstratesType	A collection of one or more SubstrateType elements. The uniqueness of the combined SubstrateType and SubstrateId key is enforced by the xs:key SubstrateKey defined in MapData.
SubstrateType	An element containing the attributes defined for the Substrate object in SEMI E142.
AliasIdsType	A collection of one or more AliasIdType elements.
AliasIdType	An element containing the attributes defined for the AliasId object in SEMI E142.
SubstrateMapsType	A collection of one or more SubstrateMapType elements.
SubstrateMapType	An element containing the attributes defined for the SubstrateMap object in SEMI E142.
Overlay	An element containing the attributes defined for the Overlay object in SEMI E142. The uniqueness of the combined MapName and MapVersion key is enforced by the xs:key OverlayKey defined in SubstrateMapsType.
ReferenceDevicesType	A collection of one or more ReferenceDeviceType elements.
ReferenceDeviceType	An element containing the attributes defined for the ReferenceDevice object in SEMI E142.
BinCodeMapType	An element containing the attributes defined for the BinCodeMap object in SEMI E142.
BinCodeType	The content of this element corresponds to the Value attribute defined for the BinCode object in SEMI E142. The remaining attributes are represented as XML attributes.
BinDefinitionsType	A collection of one or more BinDefinitionType elements.
BinDefinitionType	An element containing the attributes defined for the BinDefinition object in SEMI E142.
DeviceIdMapType	An element containing the attributes defined for the DeviceIdMap object in SEMI E142.
DeviceIdType	An element containing the attributes defined for the DeviceId object in SEMI E142.
TransferMapType	An element containing the attributes defined for the TransferMap object in SEMI E142.
TransferType	An element containing the attributes defined for the Transfer object in SEMI E142.

Table 3 Simple Types Defined in Schema

Type	*Description*
MaterialIdType	String of 1-32 characters
BinCodeContentType	String of 1 or more characters
OrientationType	Integer from 0 to 359
BinTypeEnum	Enumeration: Ascii Decimal Hexadecimal Integer2
SubstrateSideEnum	Enumeration: TopSide BottomSide

Type	*Description*
OriginLocationEnum	Enumeration: LowerLeft UpperLeft LowerRight UpperRIght Center
AxisDirectionEnum	Enumeration: UpRight DownRight UpLeft DownLeft
SubstrateTypeEnum	Enumeration: Wafer Frame Strip Tray

Table 4 Mapping XML Names to Attributes

Object.Attribute	*XML Name (Element or Attribute)*	*Type*
Layout	Layout (Element)	LayoutType
Layout.LayoutId	LayoutId (Attribute)	xs:string
Layout.DefaultUnits	DefaultUnits (Attribute)	xs:string
Layout.TopLevel	TopLevel (Attribute)	xs:boolean
Layout.Dimension	Dimension (Element)	LogicalCoordinates
Layout.LowerLeft	LowerLeft (Element)	PhysicalCoordinates
Layout.DeviceSize	DeviceSize (Element)	PhysicalCoordinates
Layout.StepSize	StepSize (Element)	PhysicalCoordinates
Layout.Z	Z (Element)	ZDimensions
Layout.TopImage	TopImage (Element)	xs:string
Layout.BottomImage	BottomImage (Element)	xs:string
Layout.ProductId	ProductId (Element)	xs:string
Substrate	Substrate (Element)	SubstrateType
Substrate.SubstrateType	SubstrateType (Attribute)	SubstrateTypeEnum
Substrate.SubstrateId	SubstrateId (Attribute)	MaterialId
Substrate.LotId	LotId (Element)	MaterialId
Substrate.CarrierType	CarrierType (Element)	xs:string
Substrate.CarrierId	CarrierId (Element)	MaterialId
Substrate.SlotNumber	SlotNumber (Element)	xs:positiveInteger
Substrate.SubstrateNumber	SubstrateNumber (Element)	xs: positiveInteger
Substrate.GoodDevices	GoodDevices (Element)	xs: positiveInteger
Substrate.SupplierName	SupplierName (Element)	xs:string
Substrate.Status	Status (Element)	xs:string
SubstrateMap	SubstrateMap (Element)	SubstrateMapType
SubstrateMap.SubstrateType	SubstrateType (Attribute)	SubstrateTypeEnum
SubstrateMap.SubstrateId	SubstrateId (Attribute)	MaterialId
SubstrateMap.LayoutSpecifier	LayoutSpecifier (Attribute)	xs:string

Object.Attribute	*XML Name (Element or Attribute)*	*Type*
SubstrateMap.SubstrateSide	SubstrateSide (Attribute)	SubstrateSideEnum
SubstrateMap.Orientation	Orientation (Attribute)	OrientationEnum
SubstrateMap.OriginLocation	OriginLocation (Attribute)	OriginLocationEnum
SubstrateMap.AxisDirection	AxisDirection (Attribute)	AxisDirectionEnum
AliasId	AliasId (Element)	AliasIdType
AliasId.Type	Type (Attribute)	xs:string
AliasId.Value	Value (Attribute)	MaterialId
Overlay	Overlay (Element)	OverlayType
Overlay.MapName	MapName (Attribute)	xs:string
Overlay.MapVersion	MapVersion (Attribute)	xs:string
ReferenceDevice	ReferenceDevice (Element)	ReferenceDeviceType
ReferenceDevice.Coordinates	Coordinates (Element)	LogicalCoordinates
ReferenceDevice.Position	Position (Element)	PhysicalCoordinates
BinCodeMap	BinCodeMap (Element)	BinCodeMapType
BinCodeMap.BinType	BinType (Attribute)	BinTypeEnum
BinCodeMap.NullBin	NullBin (Attribute)	xs:string
BinCode	BinCode (Element)	BinCodeType
BinCode.X	X (Attribute)	xs:integer
BinCode.Y	Y (Attribute)	xs:integer
BinCode.Number	Number (Attribute)	xs:integer
BinCode.Value	BinCode (Element content)	BinCodeContentType
BinDefinition	BinDefinition (Element)	BinDefinitionType
BinDefinition.BinCode	BinCode (Attribute)	xs:string
BinDefinition.BinCount	BinCount (Attribute)	xs:positiveInteger
BinDefinition.BinQuality	BinQuality (Attribute)	xs:string
BinDefinition.BinDescription	BinDescription (Attribute)	xs:string
BinDefinition.Pick	Pick (Attribute)	xs:boolean
DeviceId	Id (Element)	DeviceIdType
DeviceId.X	X (Attribute)	xs:integer
DeviceId.Y	Y (Attribute)	xs:integer
DeviceId.Value	Id (Element content)	xs:string
Transfer	T (Element)	TransferType
Transfer.FX	FX (Attribute)	xs:integer
Transfer.FY	FY (Attribute)	xs:integer
Transfer.TX	TX (Attribute)	xs:integer
Transfer.TY	TY (Attribute)	xs:integer
LogicalCoordinates.X	X (Attribute)	xs:integer
LogicalCoordinates.Y	Y (Attribute)	xs:integer
PhysicalCoordinates.X	X (Attribute)	xs:double
PhysicalCoordinates.Y	Y (Attribute)	xs:double
PhysicalCoordinates.Units	Units (Attribute)	xs:string
ZDimensions.Order	Order (Attribute)	xs:integer
ZDimensions.Height	Height (Attribute)	xs:double
ZDimensions.Units	Units (Attribute)	xs:string

RELATED INFORMATION 1
ADDING SUPPLIER DATA

NOTICE: This related information is not an official part of SEMI E and was derived from full letter ballot procedures on April 7, 2005.

R1-1 Schema Extension Example

R1-1.1 The schema may be extended such that user specific data can be added. To do this: First develop a schema that defines extensions to one or more of the complex types defined in the standard schema, as shown below.

```
<?xml version="1.0" ?>
<xs:schema id="MapData"
  targetNamespace="urn:semi-org:xsd.E142-1.V0105.SubstrateMap_ACMEProductsInc"
  xmlns:sm="urn:semi-org:xsd.E142-1.V0105.SubstrateMap"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns="urn:semi-org:xsd.E142-1.V0105.SubstrateMap_ACMEProductsInc"
  elementFormDefault="qualified">

  <xs:import namespace="urn:semi-org:xsd.E142-1.V0105.SubstrateMap"
    schemaLocation="E142-1-V0105-Schema.xsd"/>

  <xs:element name="MapData" type="sm:MapDataType"/>

  <xs:complexType name="SubstrateTypeExtension">
    <xs:complexContent>
      <xs:extension base="sm:SubstrateType">
        <xs:sequence>
          <xs:element ref="SupplierData" minOccurs="0"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

  <xs:element name="SupplierData" type="SupplierDataType"/>

  <xs:complexType name="SupplierDataType">
    <xs:sequence>
      <xs:element ref="SetupFile"/>
      <xs:element ref="TestSystem"/>
      <xs:element ref="TestProgram"/>
      <xs:element ref="Prober"/>
      <xs:element ref="Operator"/>
      <xs:element ref="ProbeCard"/>
      <xs:element ref="TestStartTime"/>
      <xs:element ref="TestEndTime"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="SetupFile" type="SupplierValue"/>
  <xs:element name="TestSystem" type="SupplierValue"/>
  <xs:element name="TestProgram" type="SupplierValue"/>
  <xs:element name="Prober" type="SupplierValue"/>
  <xs:element name="Operator" type="SupplierValue"/>
  <xs:element name="ProbeCard" type="SupplierValue"/>
  <xs:element name="TestStartTime" type="SupplierValue"/>
  <xs:element name="TestEndTime" type="SupplierValue"/>

  <xs:complexType name="SupplierValue">
    <xs:attribute name="Value" type="xs:string"/>
  </xs:complexType>

</xs:schema>
```

R1-2 BinCodeMap Extension Example

R1-2.1 Second: Add the namespace of the extension to your instance file:

R1-2.1.1 “xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance”. Now you can specify the attribute xsi:type in the Substrate element and include the extended data alongside the standard data. Both will be validated against their respective schemas. See the example below that validates against the schema extension in §R1-1.

```
<?xml version="1.0" encoding="utf-8" ?>
<MapData xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

 xsi:schemaLocation="urn:semi-org:xsd.E142-1.V0105.SubstrateMap_AcmeProductsInc
    E142-1-V0105-Schema-Extension.xsd
    urn:semi-org:xsd.E142-1.V0105.SubstrateMap
    E142-1-V0105-Schema.xsd"

 xmlns:sme="urn:semi-org:xsd.E142-1.V0105.SubstrateMap_AcmeProductsInc"
 xmlns="urn:semi-org:xsd.E142-1.V0105.SubstrateMap">

    <Substrates>
        <Substrate SubstrateType="Wafer" SubstrateId="Wafer1"
                   xsi:type="sme:SubstrateTypeExtension" >
            <GoodDevices>148</GoodDevices>
            <SupplierName>XDFG</SupplierName>
            <Status>Version 001</Status>
            <sme:SupplierData>
                <sme:SetupFile Value="58501AP"/>
                <sme:TestSystem Value="TA99K04"/>
                <sme:TestProgram Value="VW550CMG_151"/>
                <sme:Prober Value="PRBRP05"/>
                <sme:Operator Value="supeq4"/>
                <sme:ProbeCard Value="PRBRB04"/>
                <sme:TestStartTime Value="20040521180000000"/>
                <sme:TestEndTime Value="20040521570000000"/>
            </sme:SupplierData>
        </Substrate>
    </Substrates>
</MapData>
```

NOTE 1: Not all XML tools support xsi extensions yet even though it is a published XML standard.

SEMI PR8-0703
PROPOSED STANDARD FOR EQUIPMENT DATA ACQUISITION SOLUTIONS

This proposed standard was technically approved by the Global Information and Control Committee and is the direct responsibility of the Japanese Information and Control Committee. Current edition approved by the Japanese Regional Standards Committee on January 10, 2003. Initially available at www.semi.org January 2003; to be published March 2003.

NOTICE: The designation of this document was updated during the 0703 publication cycle to reflect the addition of Related Information Sections 5-8 by committee vote.

1 Purpose

1.1 The purpose of this document is to facilitate the development of an Equipment Data Acquisition (EDA) interface that can function independently of the SECS/GEM interface. It can be used as a reference for providing early implementations and prototypes of such an interface in the absence of a complete set of SEMI specifications enabling these capabilities.

1.2 *Intended Audience* — This proposed standard is intended for use by original equipment manufacturers, subsystem suppliers, third party automation software suppliers, and device makers who need to implement or interface to semiconductor equipment data acquisition capabilities.

2 Scope

2.1 *In Scope*

2.1.1 *Essential Concepts*

2.1.1.1 This proposed standard provides a description of the essential concepts necessary to provide an EDA interface implementation.

2.1.2 *Use of Communication Technologies*

2.1.2.1 This proposed standard specifies technologies that shall be used for communicating messages and data in the implementation of an EDA implementation. This proposed standard specifies the appropriate usage of these technologies.

2.1.3 *Structure of Messages and Data*

2.1.3.1 This proposed standard defines a set of messages and their associated input and output arguments, error conditions, and data types using the specified technology.

2.1.4 *Optional Features*

2.1.4.1 This proposed standard describes which interface features are considered optional for an EDA implementation.

2.2 *Out of Scope*

2.2.1 *Equipment Process Control Capabilities*

2.2.1.1 This proposed standard will not address features needed to initiate or control the execution of material processing, upload/download of recipes, movement of material, or any other capability that does not directly relate to the setup and collection of data.

2.2.2 *Internal Design of Implementations*

2.2.2.1 This proposed standard does not describe or recommend any particular internal design approach for the implementation of an EDA interface.

2.2.3 *Factory System Architectures*

2.2.3.1 This proposed standard does not describe or endorse any factory system architectures based on the implementations of the interface described herein. Examples will be used where necessary to provide context or illustrate concepts, but such descriptions are not an endorsement of, or prescription for, the depicted architecture.

2.2.4 *Design or Function of Applications*

2.2.4.1 This proposed standard does not describe details regarding the design or function of any application that uses implementations of the EDA interface described herein. Examples will be used where necessary to provide context or illustrate concepts, but such descriptions are not an endorsement of, or prescription for, the depicted application.

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 proposed standard provides a specification for implementing a functional EDA interface in the absence of a complete set of specifications intended to address the full set of capabilities. Some or all of the capabilities supported by this proposed standard may overlap, and may not be compatible with, any such specifications once completed.

3.2 This proposed standard document provides a textual specification of the interface, and is intended for human comprehension and understanding of the capabilities supported by this specification only. For the technologies in use, it is best to utilize machine-readable XML Schema and WSDL documents for implementation to ensure interoperability in the factory environment. The corresponding schema and WSDL documents can be obtained through SEMI separately from this document.

3.3 This proposed standard applies to semiconductor production equipment. Other types of equipment have not been examined.

4 Referenced Standards

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)

4.2 *Other Standards*

Extensible Markup Language (XML) 1.0, W3C Recommendation, October 6, 2000 (http://www.w3.org/TR/REC-xml)

Hyper Text Transfer Protocol (HTTP) 1.1, Internet Engineering Task Force (IETF) RFC 2616, June 1999 (http://www.w3.org/Protocols/rfc2616/rfc2616.html)

Simple Object Access Protocol (SOAP) 1.1, W3C Note, May 08, 2000 (http://www.w3.org/TR/SOAP)

Uniform Resource Identifiers (URI): Generic Syntax, IETF RFC 2396, August 1998 (http://www.ietf.org/rfc/rfc2396.txt)

Uniform Resource Name (URN) Syntax, IETF RFC 2141, May 1997 (http://www.ietf.org/rfc/rfc2141.txt)

Web Service Definition Language (WSDL) 1.1, W3C Note, March 15, 2001 (http://www.w3.org/TR/wsdl)

XML Schema , W3C Recommendation, May 02, 2001 (http://www.w3.org/TR/xmlschema-0, http://www.w3.org/TR/xmlschema-1, http://www.w3.org/TR/xmlschema-2)

5 Terminology

5.1 *Abbreviations and Acronyms*

5.1.1 *DCP* — Data Collection Plan

5.1.2 *EDA* — Equipment Data Acquisition

5.1.3 *FDC* — Fault Detection and Classification

5.1.4 *HTTP* — Hypertext Transfer Protocol

5.1.5 *OEE* — Overall Equipment Effectiveness

5.1.6 *SEMI* — Semiconductor Equipment and Materials International

5.1.7 *SOAP* — Simple Object Access Protocol

5.1.8 *URI* — Universal Resource Identifier

5.1.9 *W3C* — World Wide Web Consortium

5.1.10 *XML* — Extensible Markup Language

5.2 *Definitions*

5.2.1 *data acquisition message* — any message sent from an off-equipment software client to the equipment via an interface supported by the equipment that supports the reporting, configuration, activation, or de-activation of automatic or ad-hoc collection of data such as alarms/exceptions, events, sampling of time-varying data, etc.

5.2.2 *EDA Interface* — an interface provided by semiconductor production equipment that enables setup of data collection plans and collection of data independent of the equipment's SECS/GEM interface.

5.2.3 *process message* — any message sent from an off-equipment software client to the equipment via an interface supported by the equipment that causes physical movement, initiates, pauses, or aborts processing, directly affects the course of processing, or that results in the modification of state information maintained by the equipment that directly affects processing.

6 Overview

6.1 *Interface Scope*

6.1.1 The interface described by this proposed standard only supports the acquisition of data from equipment. Process messages will continue to remain in the domain of the SECS/GEM interface to the equipment. This means that any application that requires the use of process messages in order to achieve its objectives will continue to be required to send those messages through the SECS/GEM interface.

6.1.2 Figure 1 shows a conceptual model of the relationship between applications that can utilize this interface for data acquisition and the host system that uses the SECS/GEM interface to perform process control tasks. If process messages must be sent to the equipment based on data obtained via the EDA interface, the host system must be integrated with those components that utilize the EDA interface, so the corresponding process messages can be sent to the equipment via SECS/GEM.

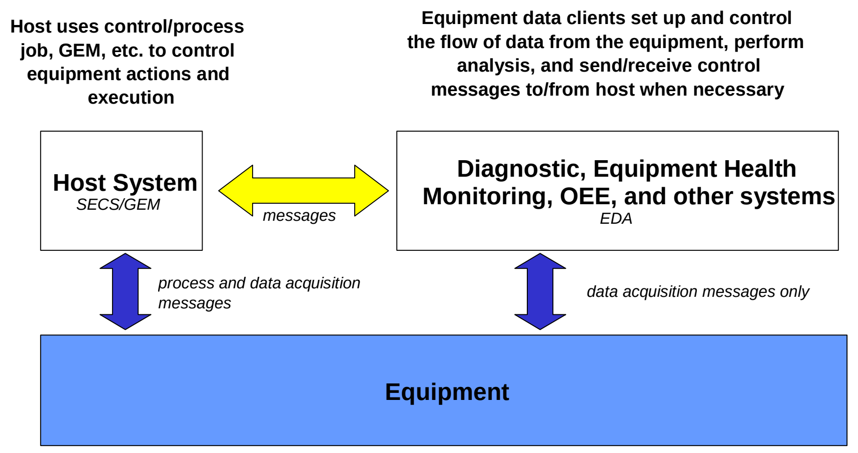
Host uses control/process job, GEM, etc. to control equipment actions and execution
Equipment data clients set up and control the flow of data from the equipment, perform analysis, and send/receive control messages to/from host when necessary
Host System
SECS/GEM
messages
Diagnostic, Equipment Health Monitoring, OEE, and other systems
EDA
process and data acquisition messages
data acquisition messages only
Equipment

Figure 1
Control and Data Interfaces to the Tool

6.1.3 Figure 2 shows a conceptual picture of the capability supported by this specification. Note that some data acquisition messages can be sent to the equipment via the SECS/GEM interface, if the equipment has been so configured. This capability is in place to facilitate factory architectures in which all data management operations must be centralized at the SECS/GEM host during the period when systems based on this specification are in development.

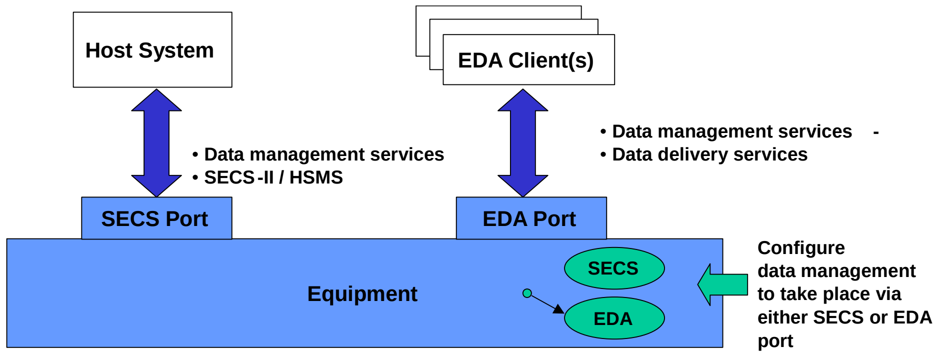
Host System
EDA Client(s)
• Data management services
• SECS-II / HSMS
• Data management services -
• Data delivery services
SECS Port
EDA Port
Equipment
SECS
EDA
Configure data management to take place via either SECS or EDA port

Figure 2
EDA Interface Features

6.2 *Interface Features*

6.2.1 This section itemizes interface features. Not all features require implementation from suppliers. The "Implementation Required" column indicates whether the feature must be implemented in order to be in compliance with this proposed standard. "Optional" in the Implementation Required column indicates that the feature is not required for compliance with the standard, but shall be considered for advanced implementations, and need not be implemented except at the supplier's discretion. The column entitled "Supported by Interface Definition" indicates whether the feature is supported via the SECS/GEM or XML/SOAP message specifications defined in this document. A "-" (dash) in the supported by interface definition column indicates that the feature is supported outside of the interface, whereas a "✓"(checkmark) indicates that the feature is supported through the interface.

Table 1 EDA Interface Features

Covered in Section #	*Feature Description*	***Supported By Interface Definition***	***Implementation Required***
General Interface			
7.1.3	Messages are communicated via Ethernet using HTTP on top of TCP/IP as the message transport.	✓	✓
7.1.9	Messages are represented as XML data within SOAP envelopes.	✓	✓
7.7.3.2	Data collection events, state model transitions, equipment exceptions, and parametric data defined by existing software standards (such as SEMI E30, E40, E87, E90, E94, etc…) that are made available through the SECS/GEM interface are supported.	✓	Optional
7.7.3.2	Transmission of milestones or tasks during subsystem processes, actuator operations, is supported.	✓	Optional
7.7.2	A time resolution of 0.01 seconds is supported in messages requiring a timestamp.	✓	Optional
Data Collection Management			
7.6.2.7	Ability to list the names of, and activate/de-activate data collection plans via the EDA interface is supported.	✓	✓
7.6.2.6	Ability to list the names of, and activate/de-activate data collection plans via the –SECS/GEM interface is supported.	✓	✓
7.4.1	Equipment configuration is provided to switch listing/activation of data collection plans to either the SECS/GEM or the EDA interface.	-	✓
7.4.4	Multiple data collection clients are supported.	✓	Optional
7.7.4	Ability to warn clients of equipment performance degradation is supported.	✓	✓
7.9	The equipment shall provide a method for the user to modify, add and delete data collection plans.	-	✓
7.9	The equipment shall provide a method for the user to view the contents of the data collection plans.	-	✓
7.6.2.2	The equipment shall provide the names of Data Collection Plans available for selection/activation by applications through the interface.	✓	✓
7.6.2.6.1	Data transmission will continue independent of the SECS/GEM communication state of the equipment.	-	✓
Security			
7.4.2, 7.6.2	Ability to specify the URI of data consumers and data managers at the equipment is supported.	-	✓
Metadata			
7.8	Equipment metadata is accessible to off-tool clients.	-	✓
7.8	Metadata shall include the equipment's physical structure down to the level of each individual sensor/actuator that can be included in a data collection plan.	-	✓
7.8	Metadata shall include the names and definitions of any parameters that can be included in a data collection plan.	-	✓
7.8	Metadata shall include the possible exceptions or errors that can be included in data collection plans.	-	✓
7.8	Metadata shall include what transitions from SEMI standard and equipment supplier defined state models are available through the EDA port.	-	✓

Covered in Section #	*Feature Description*	***Supported By Interface Definition***	***Implementation Required***
Data Collection Plans			
7.9	The data to be transmitted through the EDA port shall be defined in a set of named data collection plans.	-	✓
7.9	Data collection plans shall describe which events, exceptions, data variables will be sent through the interface when that plan is activated.	-	✓

6.3 *Risks for Preliminary Implementations*

6.4 There are some inherent risks to implementers of the EDA interface specified in this document. Significant risks include:

6.4.1 *TCP/HTTP Limitations on Data Latency* — Any use of this standard in an application requiring deterministic latency in the communication of data to an off-tool application is at risk of failure due to late delivery of data. Occasional delays in data delivery on the order of a second or more can occur in TCP communication due to the need for packet retransmission. This is inherent in the TCP/IP protocol used by HTTP, and cannot be addressed through application design.

6.4.2 *Data Collection Plan and Metadata Structure and Formats are not Specified* — The content and transmission format of Data Collection plans and Metadata for the proposed standard are left up to the supplier. Standards are in process to define Metadata and data collection plan content and data transmission format in a standard way. The standard definitions may be different than those defined by the supplier for the proposed standard.

6.4.3 *SOAP Protocol Usage* — The SOAP usage defined in this standard is targeted for point-to-point, synchronous communication. This style of SOAP communication may not be suitable for more complicated communication models, and other approaches may be required to achieve the objectives of such applications.

6.4.4 *XML/SOAP Limitations on Data Throughput* — While much has been done to optimize commercial implementations of XML/SOAP messaging support, end-to-end implementations on semiconductor equipment have not been studied to quantify the performance limitations of these technologies for data collection, or to identify what hardware/software configuration options are significant factors in providing optimal throughput performance with these technologies. If an implementation of this standard requires meeting specific data throughput goals, it may be necessary to evaluate the choice of networking software drivers, application design, messaging software toolkits used, hardware memory/cpu resources or other elements when designing an implementation of this specification. If the desired throughput goals cannot be met by these design options, it may impact the design of the message specification, the choice of transport, or the mechanism for encoding the data described in this specification.

NOTE 1: For information on factors contributing to SOAP messaging performance, see http://www.extreme.indiana.edu/~ mgovinda/research/papers/soap-hpdc2002.pdf

NOTE 2: For information on factors contributing to HTTP performance, see http://www.w3.org/Protocols/HTTP/Performance/

NOTE 3: For information purposes only, one example of possible worst-case data throughput expectations for FDC applications is available at http://www.sematech.org/public/resources/ediag/background/eec032702.pdf.

7 Proposed EDA Interface Specification

7.1 *Transport and Messaging*

7.1.1 *XML Schema Namespace* — All XML element information items defined in this specification belong to the reserved SEMI namespace "urn:semi-org:schema:eda_ps_v0.0".

7.1.2 *Separate Data Port* — The proposed Equipment Data Acquisition interface shall be a separate logical port from the SECS/GEM control port on the tool.

7.1.3 *Transport Mechanism* — The transport mechanism for the proposed EDA interface shall be HTTP 1.1. The HTTP POST operation shall be used to send messages to and from the equipment. There shall be an HTTP connection from the equipment to the client used for data only and another HTTP connection from the client to the equipment, which will be used for data management. These two HTTP connections together make up a single EDA interface.

7.1.4 *HTTP Performance Features* — The client of the equipment's data port must operate in persistent connection mode, as defined in HTTP 1.1. It is not required that the data management connection of the EDA port operate in persistent connection mode, though that is the default mode for HTTP 1.1.

7.1.5 *SOAP Binding* — The SOAP binding specified in this document shall be HTTP 1.1 in accordance with the SOAP 1.1 W3C Note.

7.1.6 *SOAP Message Style* — Message encoding will be XML document-style and not Remote Procedure Call style. In document-style message exchange, XML documents conforming to some application-defined schema are exchanged via SOAP messages.

7.1.7 *SOAP Action* — The SOAP action HTTP header field is used to facilitate dispatching individual messages defined by this specification. The field will contain a URI that indicates the specific operation in the message. The URI will be a URN of the form: "urn:semi-org:ws:eda_ps_v0.0:<OperationName>". Where OperationName corresponds to the name of the root tag for each message.

7.1.8 *SOAP Headers* — This document specifies the required use of a SOAP header element to describe the source and destination of all messages between the equipment and EDA clients (see Section 7.3.2 and to correlate responses with requests (see Section 7.3.3). The specification does not prohibit the use of additional SOAP header elements, should an implementation of this specification find them necessary.

7.1.9 *SOAP encodingStyle* — The SOAP encodingStyle attribute is not used (i.e., not present on any XML element information item) for the messages defined in this specification.

7.1.10 *SOAP Body* — The SOAP body will contain an XML fragment as defined in the Data Management and XML Data Formats sections below.

7.2 *SOAP Message Structure*

7.2.1 Figure 3 shows the organization of each SOAP message exchanged between the equipment and EDA clients. This specification defines the value of the SOAPAction HTTP header to use for each message (see Sections 7.6.2.1 through 7.6.2.5), the SOAP headers to be used for each message (Section 7.3), and the elements that are included in the body of the SOAP envelope for each message (see Sections 7.6.2.7 and 7.7).

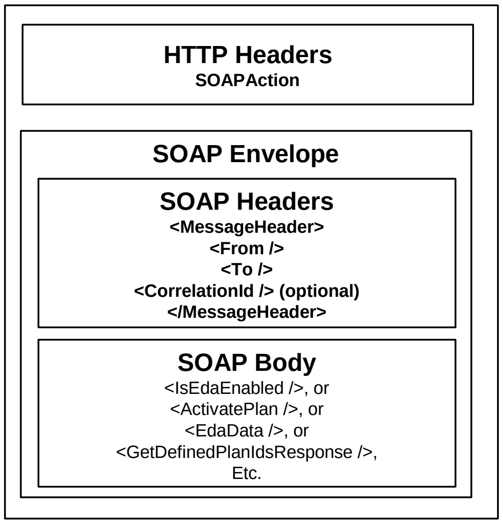

Figure 3
EDA Message Structure

7.3 *SOAP Headers*

7.3.1 This section describes a SOAP header element (MessageHeader) that can be used to facilitate different messaging implementations on both the equipment and EDA client side of the interactions defined in this specification. The SOAP header defined in this section is provided to identify the sender and recipient of each SOAP message included in this specification, and to associate a specific message with other messages sent by a given entity.

7.3.2 There are two elements specified for the purpose of identifying the sender and intended recipient of a SOAP message: a "From" element and a "To" element. Both are required elements in the MessageHeader element of each message exchanged in an implementation of this specification. These elements identify the sender or intended recipient of a given SOAP message. These identifiers can be used to route a message to an appropriate entity for processing, to distinguish a specific message from other messages received from other sources, or as a simple mechanism for enforcing access controls.

7.3.3 This section also defines an optional element ("CorrelationId") within the MessageHeader element that can be used to associate related messages (for example, a request and its reply) among all messages sent and received by a given entity. The entity in the client role generates a value for the CorrelationId that would be used by the server entity in its reply to establish the correlation. Use of this element is optional, and communicating entities are not required to process it in order to exchange the messages in this specification.

Table 2 MessageHeader XML Definition

Tag Name	*Description*	*XML Datatype*	*Behavior*	*mustUnderstand*
MessageHeader	Container element for From/To/CorrelationId.	complex	Required	1
From	Identifies the originator of this message.	anyURI	Required	n/a
To	Identifies the intended recipient of this message.	anyURI	Required	n/a
CorrelationId	Associates this message with other messages in the same message interaction.	string	Optional (0 or 1)	n/a

7.3.3.1 *From Element* — This element identifies the sender of a given SOAP message, whether the message is an event, a request, or a response to a request. It is of XML type anyURI and can be any valid URL or URN. This specification does not place any restriction on the format used to identify a sender, regardless of whether the sender is the equipment or an EDA client. All messages originating from the same sender must provide the same value for the From element so that the recipient may distinguish that sender from others.

7.3.3.1.1 *Examples* — Some examples of valid From elements are provided as follows:

- <From>urn:equipment-client:icm:utracking-01</From>
- <From>http://myfactory.com/furnace/KF011/EDA</From>

7.3.3.2 *To Element* — This element identifies the intended recipient of a given SOAP message. It is of XML type anyURI and can be any valid URL or URN. This specification does not place any restriction on the form (URL or URN) used to identify a recipient, regardless of whether the recipient is the equipment or an EDA client. All messages intended for the same recipient must provide the same value for the To element so that the recipient may distinguish that message from others intended for other recipients. If the message is a response to a request, the value of the To element in the response must be equal to the value of the From element in the original request message.

7.3.3.2.1 *Examples* — Some examples of valid To elements are provided as follows:

- <To>urn:equipment:robofurnace-inc:3JK9UI-02-99UI</To>
- <To>http://factory.icm.com/fdc/app-01</To>

7.3.3.3 *CorrelationId Element* — If used, the value of this element in the initial request message should be unique for that client. Each request shall have a different CorrelationId value. For messages that are replies, the value shall be equal to the value of the CorrelationId element provided in the original request message.

7.3.3.3.1 *Examples* — some examples of valid CorrelationId elements are provided as follows:

- <CorrelationId>5a389ad2-22dd-11d1-aa77-002035b29092</CorrelationId>
- <CorrelationId>4776</CorrelationId>

7.4 *Equipment Configuration*

7.4.1 *Data Management Configuration* — There shall be a persistent equipment configuration setting that controls whether data management (see Section 7.6) is performed via SECS/GEM or via SOAP. The implementation and behavior of this configuration setting is to be defined by the supplier, but shall not be implemented in such a way that it is possible to perform data management via both SOAP and SECS/GEM concurrently. When the equipment is configured to accept data management messages via SOAP, the equipment shall not accept these messages via SECS/GEM. Likewise, when the equipment is configured to accept data management messages via SECS/GEM, the equipment shall not accept these messages via SOAP. It must be possible for the factory to change this configuration setting without requiring the involvement of the supplier.

7.4.2 *EDA Client Configuration* — There shall be a persistent equipment configuration setting for each EDA client that establishes the URL to which equipment-initiated SOAP messages will be sent for that client, and the value of the 'From' element of the 'MessageHeader' SOAP header that that client will use in all client-initiated SOAP messages to the equipment. If an EDA client will be performing data management via SECS/GEM, it is not necessary to configure the value of the 'From' element of the 'MessageHeader' SOAP header for that client, but the configuration must indicate that the client can perform data management via SECS/GEM. It must be possible for the factory to change these configuration settings without requiring the involvement of the supplier.

7.4.3 *Equipment Identity Configuration* — There shall be a persistent equipment configuration setting that establishes the value of the 'From' element of the 'MessageHeader' SOAP header that will be used for all SOAP messages sent by the equipment. It must be possible for the factory to change this configuration setting without requiring the involvement of the supplier.

7.4.4 *Multi-Client Considerations* — When the data management interface configuration setting is set to SECS/GEM, the equipment shall accept data management messages only from the single configured SECS/GEM client, and shall send all equipment-initiated SOAP messages only to the single URL that was configured for that client. When the data management configuration setting is set to SOAP, the EDA interface may be multi-client, if the equipment supports this optional feature, and shall accept data management messages from any valid configured EDA SOAP client.

7.5 *EDA State Models*

7.5.1 *EDA Communications State Diagram*

7.5.1.1 Figure 4 shows the state model for the availability of the EDA interface to the factory. When disabled, the equipment is not in a condition whereby it can process data management messages, nor transmit data to clients. When enabled, the equipment can both process EDA data management messages, as well as transmit data to clients.

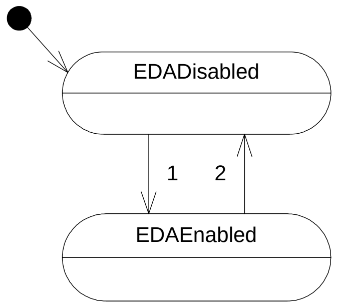

Figure 4
EDA Communications State Diagram

7.5.2 *EDA Communications Transition Definition Table*

Table 3 EDA Communications Transition Definition Table

Num	*Previous State*	*Trigger*	*New State*	*Actions*
1	EDADisabled	Equipment powers up, achieves condition to accept EDA data management requests and transmit DCP data.	EDAEnabled	For each configured client, equipment begins sending EdaEnabled message, as described in Section 7.7.4.1.
2	EDAEnabled	Equipment is entering a state wherein it will not be possible to perform EDA activities.	EDADisabled	For each configured client, equipment sends EdaDisabled message, as described in Section 7.7.4.2.

7.5.3 *Data Collection Plan State Diagram*

7.5.3.1 Figure 5 shows the EDA client's view of equipment behavior for each DCP defined on the equipment. Each DCP for a given client follows this behavior model independently of every other DCP on the equipment. For example, if client A activates "DCP1", and client B does not, and client A subsequently requests the list of active DCP's from the equipment, "DCP1" will be returned in the list of active plans, because it is in the "Active" state for client A. If client B requests a list of active DCP's from the equipment, "DCP1" will not be returned in the list of active plans, because client B did not activate "DCP1", and it is still in the "Inactive" state for client B.

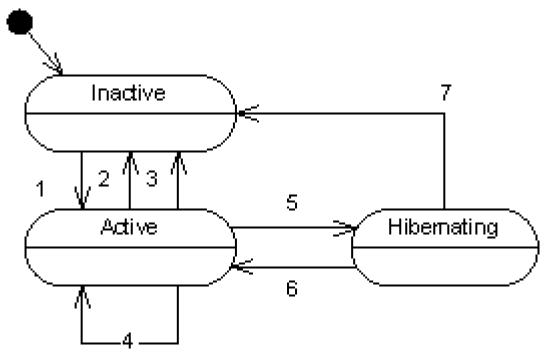

Figure 5
Data Collection Plan State Diagram

7.5.4 *Data Collection Plan Transition Definition*

Table 4 Data Collection Plan Transition Definition Table

Num	*Previous State*	*Trigger*	*New State*	*Actions*
1	Inactive	A client requests the activation of a DCP using a valid DCP id, via the ActivatePlan operation	Active	The equipment responds affirmatively to the request, and begins sending the data included in the plan to the client. If activated with “UntilDeactivated” set to “true”, the equipment shall store a record in non-volatile memory of the DCP ID and its client’s address information.
2	Active	A client requests the deactivation of a DCP using a valid DCP id via the DeactivatePlan operation.	Inactive	The equipment responds affirmatively to the request, and discontinues sending data defined by that DCP to the client. If the DCP had been activated with “UntilDeactivated” set to “true”, the equipment shall remove the previously stored record regarding the DCP and its client from non-volatile memory.
3	Active	The equipment is entering a state in which it will not be possible to transmit data to its client. The DCP was activated with “UntilDeactivated” set to “false” (see Section 7.6.2.4.1).	Inactive	The equipment shall cease sending data included in the DCP to the client.
4	Active	A client requests the activation of the DCP, via the ActivatePlan operation. The DCP is already active.	Active	The equipment responds with an error, indicating that the DCP is already active.
5	Active	The equipment is entering a state in which it will not be possible to transmit data to its client. The DCP was activated with “UntilDeactivated” set to “true” (see Section 7.6.2.4.1)	Hibernating	The equipment shall cease sending data included in the DCP to the client.
6	Hibernating	The equipment is entering a state in which it will be possible to transmit data to its client. The DCP was activated with “UntilDeactivated” set to “true” (see Section 7.6.2.4.1). The client has responded to the EdaEnabled message within the configured number of attempts (see Section 7.7.6.1).	Active	The equipment shall begin sending the data included in the DCP to the client that originally activated it, as specified in the record that was stored in non-volatile memory for the DCP when it was activated.

Num	*Previous State*	*Trigger*	*New State*	*Actions*
7	Hibernating	The equipment is entering a state in which it will be possible to transmit data to its client. The DCP was activated with "UntilDeactivated" set to "true" (see Section 7.6.2.4.1). The client has not responded to the EdaEnabled message within the configured number of attempts (see Section 7.7.6.1).	Inactive	The equipment shall cease attempting to send the data included in the DCP to the client.

7.6 *Data Management*

7.6.1 *Overview* — Data management is defined by the set of messages necessary to control the flow of data from the equipment for a given client. Requested data is organized by data collection plans (DCPs) that can be individually activated and de-activated by name. Additionally, EDA clients are able to determine which plans are defined on the equipment, and which plans are currently active. Data management can be performed by either the SECS/GEM client or any configured SOAP client, as established through configuration (see Section 7.4).

7.6.2 *Data Management Messages* — The equipment must support the following data management messages, whether data management takes place via SECS/GEM or SOAP. The equipment shall only respond to data management requests originating from configured clients. SOAP clients are identified by the value of the 'From' element of the 'MessageHeader' SOAP header in each message originating from that client. The equipment shall respond to all data management messages received from unrecognized clients with an error.

7.6.2.1 *IsEdaEnabled* **—** This operation allows the data management client to ascertain whether the equipment is ready to receive EDA data management communications and to transmit data through the EDA port.

7.6.2.1.1 When this message is sent via SOAP, the SOAPAction HTTP header used shall be "urn:semi-org:ws:eda_ps_v0.0:IsEdaEnabled". When this message is sent via SECS/GEM, the S14F19 message shall be used, with the SVCNAME data variable equal to "IsEdaEnabled" (see Table 5).

7.6.2.1.2 The equipment shall always respond to this request, whether it originates from the SECS/GEM port or a valid configured SOAP client, and independent of the setting of the data management configuration switch.

7.6.2.2 *GetDefinedPlanIds* — In this operation the data management client requests the names of the data collection plans that are available for activation by the client at the current time. The equipment returns the names of all data collection plans that are currently defined, regardless of their activation status.

7.6.2.2.1 When this message is sent via SOAP, the SOAPAction HTTP header used shall be "urn:semi-org:ws:eda_ps_v0.0:GetDefinedPlanIds". When this message is sent via SECS/GEM, the S14F19 message shall be used, with the SVCNAME data variable equal to "GetDefinedPlanIds" (see Table 7).

7.6.2.2.2 The equipment shall only respond to this request if it originates from a client that is compatible with the setting of the data management configuration switch. For example, if the configuration switch is set to enable data management via SECS/GEM and the request originates from the SECS/GEM client, the equipment shall respond with the list of defined DCP ids. If the request originates from a SOAP client, the equipment shall respond with an error.

7.6.2.3 *GetActivePlanIds* — In this operation the data management client requests the names of the data collection plans currently active for that client. The equipment EDA port responds with the list of data collection plans currently active for the requesting client.

7.6.2.3.1 When this message is sent via SOAP, the SOAPAction HTTP header used shall be "urn:semi-org:ws:eda_ps_v0.0:GetActivePlanIds". When this message is sent via SECS/GEM, the S14F19 message shall be used, with the SVCNAME data variable equal to "GetActivePlanIds" (see Table 9).

7.6.2.3.2 The equipment shall only respond to this request if it originates from a client that is compatible with the setting of the data management configuration switch. For example, if the configuration switch is set to enable data management via SECS/GEM and the request originates from the SECS/GEM client, the equipment shall respond with the list of active DCP ids. If the request originates from a SOAP client, the equipment shall respond with an error.

7.6.2.4 *ActivatePlan* — In this operation the data management client requests that a specific data collection plan is activated. The equipment responds successfully if it can fulfill the activation request. A given DCP need not be restricted to activation by a single client. There is no guarantee of the time between activation request and the transmission of data from the newly activated DCP to the client.

7.6.2.4.1 Data collection plans can be activated in two modes, determined by the value of the "UntilDeactivated" parameter (see Tables 11 and 15). If the plan is activated with "UntilDeactivated" set to "true", then the plan will remain active until a DeactivatePlan request is sent for that plan. In this mode, the equipment shall place the plan in the "Hibernating" state when the equipment shuts down, to be re-activated during equipment startup. If the plan is activated with "UntilDeactivated" set to "false", then the equipment shall place the plan in the "Inactive" state when the equipment shuts down, requiring an "ActivatePlan" request from a client to be made in order to activate the plan again.

7.6.2.4.2 When this message is sent via SOAP, the SOAPAction HTTP header used shall be "urn:semi-org:ws:eda_ps_v0.0:ActivatePlan". When this message is sent via SECS/GEM, the S14F19 message shall be used, with the SVCNAME data variable equal to "ActivatePlan" (see Table 11).

7.6.2.4.3 The equipment shall only act on this request if it originates from a client that is compatible with the setting of the data management configuration switch. For example, if the configuration switch is set to enable data management via SECS/GEM and the request originates from the SECS/GEM client, the equipment shall respond by activating the requested DCP. If the request originates from a SOAP client, the equipment shall respond with an error.

7.6.2.5 *DeactivatePlan* — In this operation the data management client requests the deactivation of one or all of the currently activated data collection plans for that client. If the data management client intends to de-activate all data collection plans then Deactivate Plan ID field should contain "ALL". The equipment responds with the list of data collection plans that have been deactivated. If the equipment is unable to de-activate the requested DCP it should respond with an error message.

7.6.2.5.1 When this message is sent via SOAP, the SOAPAction HTTP header used shall be "urn:semi-org:ws:eda_ps_v0.0:DeactivatePlan". When this message is sent via SECS/GEM, the S14F19 message shall be used, with the SVCNAME data variable equal to "DeactivatePlan" (see Table 13).

7.6.2.5.2 The equipment shall only act on this request if it originates from a client that is compatible with the setting of the data management configuration switch. For example, if the configuration switch is set to enable data management via SECS/GEM and the request originates from the SECS/GEM client, the equipment shall respond by deactivating the requested DCP. If the request originates from a SOAP client, the equipment shall respond with an error.

7.6.2.6 *SECS/GEM Data Management Messages (S14,F19/20)*

7.6.2.6.1 Each data management request sent via SECS/GEM is implemented using the Generic Service Request stream and function defined in SEMI E5. Tables 5–14 define the arguments for the requests and acknowledgements for each of these messages. Although data management via SECS/GEM may not be performed if the SECS/GEM port enters the offline state or otherwise becomes unavailable, the equipment shall continue to send DCP data activated while the SECS/GEM was available.

Table 5 Generic Service Request for IsEdaEnabled

Field Name	*Value*
OPID	0
OBJSPEC	"DataCollectionManager"
SVCNAME	"IsEdaEnabled"

Table 6 Generic Service Acknowledge for IsEdaEnabled

Field Name	*Value*
SVCACK	0, 2
LINKID	0
ERRCODE$_n$	Supplier chooses the number and value of ERRCODE$_n$ and ERRTEXT$_n$ entries for error responses.
ERRTEXT$_n$	Supplier chooses the number and value of ERRCODE$_n$ and ERRTEXT$_n$ entries for error responses.

7.6.2.6.2 This message does not provide any return parameters (list item 3 of the S14F20 message has zero length).

Table 7 Generic Service Request for GetActivePlanIds

Field Name	*Value*
OPID	0
OBJSPEC	"DataCollectionManager"
SVCNAME	"GetActivePlanIds"

Table 8 Generic Service Acknowledge for GetActivePlanIds

Field Name	*Value*
SVCACK	0, 2, 5 Use 5 only when data management configuration switch is not set to SECS/GEM
LINKID	0
SPNAME$_1$	“ActivePlanIds”
SPVAL$_1$	L,n n=number of active DCPs 1. <OBJID$_1$> 2. <OBJID$_2$> . . . n. <OBJID$_n$> Each plan id is a separate entry (format 20) in this list.
ERRCODE$_n$	Supplier chooses the number and value of ERRCODE$_n$ and ERRTEXT$_n$ entries for error responses
ERRTEXT$_n$	Supplier chooses the number and value of ERRCODE$_n$ and ERRTEXT$_n$ entries for error responses

Table 9 Generic Service Request for GetDefinedPlanIds

Field Name	*Value*
OPID	0
OBJSPEC	“DataCollectionManager”
SVCNAME	“GetDefinedPlanIds”

Table 10 Generic Service Acknowledge for GetDefinedPlanIds

Field Name	*Value*
SVCACK	0, 2, 5 Use 5 only when data management configuration switch is not set to SECS/GEM
LINKID	0
SPNAME$_1$	“DefinedPlanIds”
SPVAL$_1$	L,n n=number of defined DCPs 1. <OBJID$_1$> 2. <OBJID$_2$> . . . n. <OBJID$_n$> Each plan id is a separate entry (format 20) in this list.
ERRCODE$_n$	Supplier chooses the number and value of ERRCODE$_n$ and ERRTEXT$_n$ entries for error responses
ERRTEXT$_n$	Supplier chooses the number and value of ERRCODE$_n$ and ERRTEXT$_n$ entries for error responses

Table 11 Generic Service Request for ActivatePlan

Field Name	*Value*
OPID	0
OBJSPEC	“DataCollectionManager”
SVCNAME	“ActivatePlan”
SPNAME$_1$	“PlanID”
SPVAL$_1$	<OBJID>, Format 20, equal to the id of the plan to be activated
SPNAME$_2$	“UntilDeactivated”
SPVAL$_2$	<ACKA>, set to true if DCP should remain active across shutdowns

Table 12 Generic Service Acknowledge for ActivatePlan

Field Name	*Value*
SVCACK	0, 2, 3, 5 Use 5 only when data management configuration switch is not set to SECS/GEM
LINKID	0
SPNAME$_1$	“IsActivated”
SPVAL$_1$	<ACKA>, set to true if the requested plan is activated
ERRCODE$_n$	Supplier chooses the number and value of ERRCODE$_n$ and ERRTEXT$_n$ entries for error responses
ERRTEXT$_n$	Supplier chooses the number and value of ERRCODE$_n$ and ERRTEXT$_n$ entries for error responses

Table 13 Generic Service Request for DeactivatePlan

Field Name	*Value*
OPID	0
OBJSPEC	“DataCollectionManager”
SVCNAME	“DeactivatePlan”
SPNAME$_1$	“PlanID”
SPVAL$_1$	<OBJID>, Format 20, equal to the id of the plan to be activated

Table 14 Generic Service Acknowledge for DeactivatePlan

Field Name	*Value*
SVCACK	0, 2, 3, 5 Use 5 only when data management configuration switch is not set to SECS/GEM
LINKID	0
SPNAME$_1$	"DeactivatedPlanIds"
SPVAL$_1$	L,n n=number of deactivated DCP's 1. <OBJID$_1$> 2. <OBJID$_2$> . . . n. <OBJID$_n$> Each plan id is a separate entry (format 20) in this list.
ERRCODE$_n$	Supplier chooses the number and value of ERRCODE$_n$ and ERRTEXT$_n$ entries for error responses
ERRTEXT$_n$	Supplier chooses the number and value of ERRCODE$_n$ and ERRTEXT$_n$ entries for error responses

7.6.2.7 *EDA Data Management via SOAP* — The EDA client may also perform EDA data management via SOAP. Data management requests issued from an EDA client to the equipment via SOAP will apply to the data port associated with that client.

7.6.2.7.1 *EDA Data Management Message Structure* — The data management messages from an EDA client will be structured in the same way as the data acquisition messages defined below in Section 7.7.2. The XML fragments inside the SOAP body element are structured as described in Tables 15 and 16.

7.6.2.7.2 *XML Definition of Data Management Requests*

Table 15 XML Data Management Request Definition

Tag Name	*Description*	*XML Datatype*	*Behavior*				
			IsEda-Enabled	GetDefined-PlanIds	GetActive-PlanIds	ActivatePlan	Deactivate-Plan
see columns 4-8 in this row for tag names, shown in quotes	Operation Name Element	Complex	Required- "IsEda-Enabled"	Required- "GetDefined PlanIds"	Required- "GetActive-PlanIds"	Required- "Activate-Plan"	Required- "Deactivate-Plan"
EquipmentID	EquipmentID element	Complex	Required	Required	Required	Required	Required
PlanID	Data collection plan to be activated or deactivated ("ALL" is reserved for DeactivatePlan)	String	N/A	N/A	N/A	Required	Required
UntilDeactivated	Indicates mode of activation	Boolean	N/A	N/A	N/A	Required	N/A

7.6.2.7.3 *XML Definition of Data Management Responses* — Responses to data management requests are embedded in the HTTP response to the data management request. The response will contain only the operation type element as defined below.

Table 16 XML Data Management Response Definition

Tag Name	*Description*	*XML Datatype*	*Behavior*				
			IsEda-Enabled	GetDefined-PlanIds	GetActive-PlanIds	ActivatePlan	Deactivate-Plan
See columns 4-8 in this row for tag names, shown in quotes	Operation Name Element	Complex	Required- “IsEda-Enabled Response”	Required- “GetDefined PlanIds Response”	Required- “GetActive-PlanIds Response”	Required- “Activate-Plan Response”	Required- “Deactivate-Plan Response”
DefinedPlanIds	Space delimited array of defined data collection plan names	List	N/A	Required	N/A	N/A	N/A
ActivePlanIds	Space delimited array of Active data collection plan names	List	N/A	N/A	Required	N/A	N/A
IsEnabled	Whether or not the EDA port is enabled	Boolean	Required	N/A	N/A	N/A	N/A
IsActivated	Whether or not the equipment activated the requested plan	Boolean	N/A	N/A	N/A	Required	N/A
Deactivated-PlanIds	Data collection plan ids de-activated by this operation	List	N/A	N/A	N/A	N/A	Required
Error	Include if data management request could not be executed	Complex	Optional	Optional	Optional	Optional	Optional
ErrorTime	Time of the error	DateTime	Optional	Optional	Optional	Optional	Optional
ErrorType	Category of error	String	Optional	Optional	Optional	Optional	Optional
ErrorCode	Unique identifier of error	String	Optional	Optional	Optional	Optional	Optional
ErrorDesc	Text description of error	String	Optional	Optional	Optional	Optional	Optional

7.7 *XML Message Formats*

7.7.1 This section describes the XML fragments that make up the body of the SOAP messages that will be sent by equipment to EDA clients through the EDA port. The section describes the general construction of the SOAP body and the structure of the elements that make up the SOAP body. Each EDA client must implement all of the messages defined in this section.

7.7.2 *General Construction* — Each SOAP body will have the following structure: The root element is the Operation Name element, and is named after the operation being performed. The first element inside the root element will be the Equipment ID element. The Equipment ID element is followed by one or more Operation Name elements. Each Operation Name element in a data acquisition message may contain 0 or more Param elements.

7.7.3 *Equipment ID*

7.7.3.1 Each message has one and only one EquipmentID element. The EquipmentID element identifies the equipment for which all data in the message apply.

Table 17 Equipment ID XML definition

Tag Name	*Description*	*XML Datatype*	*Behavior*
EquipmentID	Complex element that contains all the equipment identifier fields	Complex	Required
Supplier	Supplier Name	String	Required
Model	Equipment Model	String	Required
ImmutableID	Serial number or Immutable ID of the specific tool	String	Required

7.7.3.2 *General Structure Example*

```
<EdaData>
      <EquipmentID>
            <Supplier>Some Supplier</Supplier>
            <Model>8400</Model>
            <ImmutableID >998948</ImmutableID >
      </EquipmentID>
      <Event>
            ……..
            …..
      </Event>
      <ExEvent>
            ……..
      </ExEvent>
</EdaData>
```

7.7.4 *Param Element*

7.7.4.1 There may be zero or more Param elements inside any Operation Name element. Param elements can describe measured values or descriptive attributes. Any tag names not explicitly defined in this proposed standard must be communicated through param elements. All param elements that may be communicated through the EDA interface must be defined in the Metadata files.

7.7.4.2 All param elements must contain the Name and Value elements, at a minimum. Other, optional elements are listed in the table below. The equipment supplier may define additional optional elements. The elements to be included in the communication of specific Names must be defined in the Metadata files. The units and the data type of specific Names must be defined in the Metadata files.

Table 18 Param Element Definition

Tag Name	*Description*	*XML Datatype*	*Behavior*
Param	Complex element containing all fields pertaining to a single parameter.	Complex	Required
Name	The name of the parameter.	String	Required
Value	The value of the parameter. See below for more definition.	Complex	Required
IntVal	An integer value	Integer	Optional; maxOccurrances=unbounded
IntArrayVal	An array of integers	List; itemType=integer	Optional; maxOccurrances=unbounded
FloatVal	A float value	Float	Optional; maxOccurrances=unbounded
FloatArrayVal	An array of Float values	List; itermType=Float	Optional; maxOccurrances=unbounded
DoubleVal	A double value	Double	Optional; maxOccurrances=unbounded
DoubleArrayVal	An array of double values	List; itemType=double	Optional; maxOccurrances=unbounded
StringVal	A string	String	Optional; maxOccurrances=unbounded
StringArrayVal	An array of strings	List: itemType=string	Optional; maxOccurrances=unbounded
DateTimeVal	A dateTime	DateTime	Optional; maxOccurrances=unbounded

Tag Name	*Description*	*XML Datatype*	*Behavior*
DateTimeArrayVal	An array of dateTimes	List: itermType=dateTime	Optional; maxOccurrances=unbounded
Base64BinaryVal	A base64 values	Base64Binary	Optional; maxOccurrances=unbounded
AnyURIVal	A URI	AnyURI	Optional; maxOccurrances=unbounded
BoolVal	A Boolean value	Boolean	Optional; maxOccurrances=unbounded
BoolArrayVal	An array of Boolean values	List: itemType=boolean	Optional; maxOccurrances=unbounded
StructVal	Container for groups of simple types. May contain an unlimited number of the simple types defined for value.	Complex	Optional; maxOccurrances=unbounded
MeasTime	Time of the measurement, if applicable.	dateTime	Optional
Locator	Unique identifier for the tool entity, or namable, to which the param applies.	String	Required
Extension	Element containing any additional fields provided by the supplier.	Any	Optional

7.7.4.3 *Value Elements* — The elements within the Value element may contain a single scalar value, a space delimited array, or a StructVal element. StructVal elements may not be nested.

Example 1: simple parametric data point named pressure

```
<Param>
    <Name>Pressure</Name>
    <Value>
        <FloatVal>43.56</FloatVal>
    </Value>
    <MeasTime>1999-05-31T13:20:00.087-05:00</MeasTime
</Param>
```

Example 2: Collection event ID

```
<Param>
    <Name>CEID</Name>
    <Value>
        <IntVal>1278</ IntVal >
        <StructVal>
            <DateTimeVal>……</DateTimeVal>
            <FloatVal>…….</FloatVal>
        </StructVal>
    </Value>
</Param>
```

7.7.5 *DateTime XML Datatype* — Whenever an element is given the dateTime XML DataType in this specification it shall conform to the dateTime built-in data type defined in XML Schema Part 2-Datatypes. The dateTime representation shall include the difference from UTC and the fractional seconds precision shall be at least .01 seconds (i.e., ss.ss). Equipment shall pad the fractional seconds field with zeros beyond its measurable precision.

7.7.6 *Eda Client Operations* — This specification defines three different notifications that EDA clients must support: EdaEnabled, EdaDisabled, and EdaData. The EdaData notification includes both Event and Exception data. Extensions to the Event data for the EdaData message are described in related information. The SOAPAction

HTTP headers used for these messages are “urn:semi-org:ws:eda_ps_v0.0:EdaEnabled”, “urn:semi-org:ws:eda_ps_v0.0:EdaDisabled”, and “urn:semi-org:ws:eda_ps_v0.0:EdaData”, respectively.

7.7.6.1 *EdaEnabled* — Whenever the equipment EDA port enters the EDAEnabled state, it shall send the EdaEnabled message to each configured client. The equipment will not send any additional messages to a configured client until it receives a successful response to the EdaEnabled message from that client. If the response is not received the equipment will retry the EdaEnabled message a specific number of times and at a specific interval. Both the number of retries and the retry interval shall be configurable at the equipment. If at the end of this retry cycle the expected response to the EdaEnabled message has not been received from that client then the equipment shall deactivate all of that client’s hibernating data collection plans and each of those DCPs shall enter the Inactive state (see transition 7 in Figure 5). The equipment will activate those DCPs again only when a subsequent ActivatePlan request is received from a configured client.

7.7.6.2 *EdaDisabled* — Whenever the equipment EDA port is entering the EDADisabled state, the equipment shall send the EdaDisabled message to each configured client.

Table 19 EdaEnabled/Disabled Message

Tag Name	*Description*	*XML Datatype*	*Behavior*	
			EdaEnabled	EdaDisabled
See columns 3 and 4 in this row for tag names, shown in quotes.	Operation Name element	Complex	Required-“EdaEnabled”	Required- “EdaDisabled”
EquipmentID	EquipmentID element	Complex	Required	Required

7.7.7 *Event Data* — The Event data format is a simple, general-purpose event format intended for communicating fundamental events from the SECS/GEM interface, and other similar data. The event may be used as a container for parametric data that does not have a particular context within either an operational segment or an exception event. Event data can be provided anywhere within the EdaData message structure following the EquipmentID element. It can appear zero or more times, and can be interspersed with Exception data.

Table 20 Event Description

Tag Name	*Description*	*XML Datatype*	*Behavior*
Event	Operation Name	Complex	Required
EventTime	Event time and date stamp.	dateTime	Required
Locator	Unique identifier for the tool entity, or namable, to which the event applies.	String	Required
EventID	Event identifier	String	Required
Context	Complex element containing all context Param elements.	Complex	Optional- 0 or 1
Param	Complex element that contains all fields pertaining to a single context param.	Complex	1 Required (if Context element included), Many Optional
Data	Complex element containing all data param elements.	Complex	Required-Exactly 1
Param	Complex element that contains all fields pertaining to a single data param.	Complex	1 Required, Many Optional
Extension	Element containing any additional information provided by the supplier.	Any	Optional

Example: Equipment Event

```
<EdaData>
    <EquipmentID>
        <Supplier>Some Supplier</Supplier>
        <Model>8400</Model>
        <ImmutableID >998948< ImmutableID >
    </EquipmentID>
    <Event>
        <EventTime>1999-05-31T13:20:00-05:00</EventTime>
        <Locator>cluster.tool.module</Locator>
        <EventID>274</EventID>
        <Context>
            <Param>
                <Name>PrJobID</Name>
                <Value>
                    <IntVal>898887</IntVal>
                </Value>
            </Param>
            <Param>
                <Name>ShotNum</Name>
                <Value>
                    <IntVal>898</IntVal>
                </Value>
            </Param
        </Context>
        <Data>
            <Param>
                <Name>Temperature</Name>
                <Value>
                    <IntVal>184</IntVal>
                </Value>
            </Param>
            <Param>
                <Name>Pressure</Name>
                <Value>
                    <IntVal>9984</IntVal>
                </Value>
            </Param>
        </Data>
    </Event>
</EdaData>
```

7.7.8 *Locator* — This attribute provides a unique identifier for a particular equipment component based on its position in the tool hierarchy. Legal values for the *locator* attribute are derived by starting with the name of root equipment component in the hierarchy followed by the '.' character, followed by the names of any components between the root and target component (each name separated by the '.' character), followed by the name of the target component. A shorthand notation for this pattern is: "root.node.node.node.target".

7.7.9 *Exception Event*

7.7.9.1 Exception event is an event concerned with errors, faults, or other unexpected events detected by the equipment. Exception data can be provided anywhere within the EdaData message structure following the EquipmentID element. It can appear zero or more times, and can be interspersed with Event data.

Table 21 Exception Event Format Definition, valid between "Msg" tags

Tag Name	*Description*	*XML Datatype*	*Behavior*
ExEvent	Operation Name element	Complex	Required
ExTime	Exception time and date stamp	dateTime	Required
Locator	Unique identifier for the tool entity , or namable, to which the event applies.	String	Required
ErrorCode	Unique code indicating the error type. May be the SECSGEM ALID.	String	Required
ExType	Name of the event type. "Alarm" is reserved for E30 alarms.	String	Required
ExState	"set" and "clear" are used for exceptionType "Alarm". Otherwise, use of this field is supplier-defined. If this field is not applicable for the exceptionType, the contents shall be "none".	String	Required
ExDesc	A text description of the exception.	String	Required
Severity	The supplier-defined severity.	String	Supplier Optional
Data	Complex element that contains all Param elements.	Complex	Optional
Extension	Reserved for supplier extensions.	Any	Supplier Optional

Example: Exception Event with Attached Parametric Detail

```
<EdaData>
    <EquipmentID>
        <Supplier>Some Supplier</Supplier>
        <Model>8400</Model>
        <ImmutableID >998948< ImmutableID >
    </EquipmentID>
    <ExEvent>
        <ExTime>1999-05-31T13:20:00-05:00</ExTime>
        <Locator>stepper.stage.optics </Locator>
        <ErrorCode>19898</ErrorCode>
        <ExType>Alarm</Extype>
        <ExState>set</ExState>
        <ExDesc>Failed to see clearly</ExceptionDesc>
        <Severity>2</Severity>
        <Data>
            <Param>
                <Name>AlignLandingEnergy</Name>
                <Value>
                    <Float>0.343</Float>
                </Value>
            </Param>
        </Data>
    </ExEvent>
</EdaData>
```

7.7.9.2 *Representation of SEMI E30 Alarms* — It is expected that the supplier will need to transmit exception event messages corresponding to SEMI E30 alarms, and possibly other exceptions, through the EDA interface. Whenever an exception event message corresponding to an E30 alarm is transmitted, the ErrorCode element shall contain the value of the ALID data item, and the ExDesc element shall contain the value of the ALTX data item. If the E30 alarm message is triggered by the Alarm Detected event then the contents of the ExState element shall contain "set", for Alarm Set. If the E30 alarm message is triggered by alarm clear then the ExState element shall contain “clear”, for alarm clear. If it is necessary to include the value of ALCD in the exception, this information shall be provided as a separate parameter within the Data element, the name of the parameter shall be “ALCD”, and its value shall be of type Base64Binary. The only other required fields for transmission of an E30 alarm are MsgType, EventTime and Locator.

7.7.10 *Notification of Interference with Tool Operations* — If the function of the EDA port, or the volume of data, is interfering with tool operations or the output of the SECS/GEM port then the tool will notify all configured clients of the condition. The format of this notification is defined in the following section.

7.7.10.1 *Tool Interference Message* — The following message format will be used to indicate interference with tool operations. The table below describes the “Operation” element of the message. The message takes the general form of XML data messages as described in Section 7.7.2. The SOAPAction HTTP header for this message is “urn:semi-org:ws:eda_ps_v0.0:EdaError”. The ErrorType field for tool interference messages shall have the value “PerformanceWarning”. Note that this message can be used to communicate other error conditions not defined in this specification.

Table 22 EdaError XML Defintion

Tag Name	*Description*	*XML Datatype*	*Behavior*
EdaError	Operation Name	Complex	Required
ErrorTime	Error time and date stamp	dateTime	Required
ErrorType	The category or type of error	String	Required
ErrorCode	A unique identifier for the error	String	Required
ErrorDesc	A more specific description of the problem	String	Required

7.8 *Metadata*

7.8.1 *Metadata* — Metadata must be provided to the user in the form of XML documents. The structure and format of metadata is left to the equipment supplier. The provided metadata must include sufficient information to describe the following items:

1) Definition of all Param Names and other non-standard tags that will be communicated. These definitions will include, where applicable, units, data types, max sampling frequencies, valid limits, the tool entity to which each param applies, and any other aspects necessary to understand and use the data element,

2) Definition of the equipment structure and hierarchy,

3) Definition of the available events and the tool entities to which they apply,

4) Definition of any state models that are supported,

5) Definition of which parameters can be communicated with each event, and

6) Definition of what exceptions can be communicated, to what equipment entities they apply, and what parameters will be included with each exception.

7.9 *Data Collection Plans* — Data collection from the equipment will be enabled through the activation of specific data collection plans. The definition of available data collection plans must be provided to the user in the form of an XML file. The structure and format of the data collection XML file(s) is left to the supplier.

7.9.1 *Modifying and Adding Data Collection Plans* — The equipment supplier must provide a means for customers to change the content and number of available data collection plans.

7.10 *Schema and WSDL documents*

7.10.1 Three machine-readable files are provided to help ensure interoperable messaging between factory applications and supplier implementations of the specification. This section describes each file and its usage.

7.10.2 *eda.xsd*

7.10.2.1 This file is an XML Schema document that describes the structure of the XML document fragments specified in Tables 15–22. This file can be used to validate, or to facilitate the serialization and de-serialization of, the body element of the SOAP envelope for each of the messages defined in this specification. It cannot be used to enforce or specify the usage of the HTTP and SOAP protocols.

7.10.3 *eda_equipment.wsdl*

7.10.3.1 This file is an interface description document that describes the types, messages, and SOAP and HTTP protocol usage for each of the operations that the equipment must support (Tables 15 and 16). Note that it does not include specification of the endpoint address for these operations, since this is unique for each equipment installation that provides the service. This file can be used by SOAP messaging toolkits that support WSDL to generate interoperable client and server code stubs. The contents of eda.xsd have been directly included in the body of this file, so it is not necessary to use that file separately in order to implement these messages.

7.10.4 *eda_consumer.wsdl*

7.10.4.1 This file is an interface description document that describes the types, messages, and SOAP and HTTP protocol usage for each of the operations that any EDA client must support (Tables 17-22). Note that it does not include specification of the endpoint address for these operations, since this is unique for each EDA application. This file can be used by SOAP messaging toolkits that support WSDL to generate interoperable client and server code stubs. The contents of eda.xsd have been directly included in the body of this file, so it is not necessary to use that file separately in order to implement these messages.

RELATED INFORMATION 1
OPERATIONAL SEGMENT EXTENSION TO EDA

NOTICE: This related information is not an official part of SEMI PR8. This related information is not intended to modify or supersede the official proposed standard. Determination of the suitability of the material is solely the responsibility of the user.

R1-1 Operational Segments

R1-1.1 *Operational Segment Event* — Operational segments are optional, at the discretion of the supplier, and can be provided as an extension to the equipment events described in the specification. Operational Segment events are provided for those suppliers that desire to model their tools in a more structured and hierarchical way than is possible with simple events. The operational segment is a generalization of the task model defined in SEMI E116 (Equipment Performance Tracking).

R1-1.1.1 *Operational Segments* — An operational segment, or segment, is a persistent entity that is expected to occur as part of normal or planned equipment operations. A segment always has only one start time and only one complete time, so if a segment is viewed on a timeline or Gantt chart it literally occupies a segment of time. Most segments of interest are general equipment operations that are expected to repeat on a regular basis. For example, for an equipment robot that performs the act of loading a wafer into a chamber many times, the robot is the locator, the "load" task is a segment, and each occasion that the equipment performs that task is an instance of that "load" segment.

R1-1.1.2 *Segment Instances* — Segments of the same name repeat frequently over time. It is necessary to identify a particular occurrence of a segment and to relate it to other segments involved in the same or dependant operations. InstanceIDs perform this function. The network of operational segments can be reconstructed by connecting each segment event to the segment that it describes and then relating each segment to its parent segment, if any. For example, a wafer "get" might have a start and complete event, which are related via an InstanceID to describe the "get" sub-task. The "get" sub-task might be part of a load wafer task. The "get" sub-task would be related to its parent segment via the ParentInstanceID.

Table R1-1 Operational Segment Event

Tag Name	*Description*	*XML Datatype*	*Behavior*
OpSegEvent	Operation Name	Complex	Required
SegmentType	Identifier for the type of segment being executed (e.g. Process Job, Substrate, Task, etc.).	String	Required

EventType	Enum--
	0 = Complete;
	1 = Start;
	2 = Suspend;
	3 = Resume;
	4 = InProgress;
	5 = Transitional;
	6–9 = Equipment Defined
		NonNegative Integer; Max inclusive = 9	Required

TaskName	Name of Task	String	Optional, include if task or sub-task
Purpose	The intended use of the segment (e.g. Production, Engineering, maintenance, etc.).	String	Optional, include if task or sub-task
SegInstID	Unique identifier of the specific segment instance to which the message applies. May be UUID.	String	Optional
ParSegInstID	Unique identifier of the parent of the segment instance that the message applies. May be UUID.	String	Optional, include if segment start event

Example: OpSeg Event

```
<EdaData>
    <EquipmentID>
        <Supplier>Some Supplier</Supplier>
        <Model>8400</Model>
        <ImmutableID>998948</ImmutableID>
    </EquipmentID>
    <Event>
        <EventTime>1999-05-31T13:20:00-05:00</EventTime>
        <Locator> Inspector.robot.aligner </Locator>
        <EventID>OpSegEvent</EventID>
        <Extension>
             <OpSegEvent xmlns=" urn:semi-org:schema:opseg_v0.0" >
                 <SegmentType>Task</SegmentType>
                 <EventType>1</EventType>
                 <TaskName>Align</TaskName>
                 <Purpose>Manufacturing</Purpose>
                 <SeglInstID>1999-05-31T13:00:00-05:00</SegInstID>
                 <ParSegInstID>1999-05-31T12:00:00-05:00</ParSegInstID>
            </ OpSegEvent>
        </Extension>
    </OpSegEvent>
</EdaData>
```

R1-1.1.2.1 Note that the EventID child element of the Event element from the specification is used to identify the extension as an operation segment extension. The additional information used for operational segments is contained within the extension element, and resides in a different namespace than that used to define the EDA data types.

R1-1.1.3 *Enumerations* — Three enumerations are included in the Operational segment format for ease of parsing and extraction from a database. These enumerations are: 1) SegmentType, 2) EventType, and 3) Purpose.

R1-1.1.4 *EventType Enumeration* — The EventType enumeration is a list of the types of events that may occur as part of the execution of a segment. This list describes the different event types that naturally occur as part of the execution of a segment. Six (6) common event types are defined and the equipment may add custom types as required.

Enumeration Value	*Name*	*Description*
0	Complete	The segment completes, either normally or abnormally
1	Start	The segment begins
2	Suspend	The segments stops, with the potential to resume
3	Resume	The segment resumes execution after a suspend
4	In-Progress	A milestone is reached, or a parametric report is issued
5	Transitional	One segment ends and another begins simultaneously
6–9	Equipment Defined	These enumerations can be defined by the supplier

R1-1.2 *Schema Document*

R1-1.2.1 A machine-readable XML Schema file is provided to help ensure interoperable usage of the operational segment message described in this section of the Related Information for this specification.

R1-1.2.2 *OpSeg.xsd*

R1-1.2.2.1 This file is an XML Schema document that describes the structure of the XML document fragment specified in Table R1-1 of this section of the Related Information. This file can be used in conjunction with the schema and WSDL files provided with the primary specification to validate, or to facilitate, the serialization and deserialization of, operational segment data enclosed in the Extension element of the Event message.

RELATED INFORMATION 2
EXAMPLE EDA CLIENT APPLICATIONS

NOTICE: This related information is not an official part of SEMI PR8. This related information is not intended to modify or supersede the official proposed standard. Determination of the suitability of the material is solely the responsibility of the user.

R2-1 Typical Users of Proposed EDA Solutions

R2-1.1 This section provides a very basic overview of the different classes of data collection clients that proposed EDA implementations are intended to support from the viewpoints of the types of data that are used, and the basic dynamics of each client. The different classes of data collection capabilities may be required in a combination or separately depending on the necessity.

R2-1.1.1 *Equipment Utilization Tracking*

R2-1.1.1.1 *Scope of Utilization Tracking Applications*

R2-1.1.1.1.1 Utilization tracking applications are responsible for collecting information regarding the use of the equipment in the factory. These applications typically look for events and alarms/exceptions from the to determine when the equipment is being exercised, when it is idle, and when it is unable to perform its intended function due to errors or alarms.

R2-1.1.1.2 *Timeliness and Accuracy of Utilization Data*

R2-1.1.1.2.1 Such applications are not time-critical, and can accept data off-equipment with very high latencies (on the order of seconds to minutes or greater) with no adverse effect on the ability of the application to function. However, as these applications are meant to calculate where the equipment is spending its time, they require adequately accurate time stamping of all events and alarms/exceptions. They also depend on the equipment to properly communicate when tasks begin and end, when alarms are set and when they are cleared, etc. It is important for the equipment not to drop these events, to double-count them, or to provide incorrect timestamps for them.

R2-1.1.2 *Equipment Health Monitoring*

R2-1.1.2.1 *Scope of Equipment Health Monitoring Applications*

R2-1.1.2.1.1 Equipment health monitoring applications are responsible for collecting information regarding the details of equipment operation, and to analyze equipment operational data to detect or predict negative trends in the performance of the equipment. These applications will typically look for equipment operational events describing the behavior of individual subsystems, exceptions and alarms, trace data on select parameters, and actuator status. The EDA interface shall support a detailed level of equipment operations that has not been conventionally supported so as to achieve detailed equipment health monitoring or diagnosis.

R2-1.1.2.2 *Timeliness and Accuracy of Equipment Health Data*

R2-1.1.2.2.1 These applications are not time-critical, and can accept data off-equipment with relatively high latencies (on the order of seconds to minutes) with no real adverse effect on the ability of the application to function. They may be more reactive than utilization tracking systems, looking for negative conditions on which they may take some action (such as paging an engineer, or warning of the need for a possible maintenance run).

R2-1.1.3 *Run-to-Run Control*

R2-1.1.3.1 *Scope of Run-to-Run Applications*

R2-1.1.3.1.1 Run-to-Run applications are reactive systems that collect process variable data from the equipment (typically a metrology tool) at the end of a run, and perform analysis on the results to determine if any adjustments in control parameters shall be made to future processing at the step that was measured. The new settings are calculated and made accessible for subsequent use at the corresponding process step.

R2-1.1.3.2 *Timeliness of Run-to-Run Data*

R2-1.1.3.2.1 These applications are not very time critical in today's systems, where control parameter adjustments are performed on a lot basis. As it becomes possible to collect measurement data on a wafer basis, they will become more time-sensitive, and may function properly with latencies on the order of seconds.

R2-1.1.3.3 *Run-to-Run Data Interests*

R2-1.1.3.3.1 These applications are typically interested in measurement result sets, events, exceptions, or alarms.

R2-1.1.4 *Fault Detection and Classification (FDC)*

R2-1.1.4.1 *Scope of FDC Systems*

R2-1.1.4.1.1 FDC systems are data-intensive, time-critical applications. They typically collect process variable, event, and exception/alarm data from the equipment as it's produced, and analyze the results, looking for faults that indicate or could lead to mis-processing. If such a condition is detected, an FDC system may send a control signal to the equipment through the host, to abort processing before further material is lost.

R2-1.1.4.2 *Timeliness of FDC Data*

R2-1.1.4.2.1 FDC applications can be very time critical, sometimes in the sub-second range. As the time criticality of FDC applications becomes more acute, the ability of this interface to support these applications will be severely tested.

RELATED INFORMATION 3
EQUIPMENT DATA LIFECYCLE

NOTICE: This related information is not an official part of SEMI PR8. This related information is not intended to modify or supersede the official proposed standard. Determination of the suitability of the material is solely the responsibility of the user.

R3-1 The Data Life Cycle

R3-1.1 The data originating from the equipment may be processed or converted into many forms depending on how the data is to be utilized. The data obtained through the EDA interface will be processed by data consumers and may be converted into a command that initiates certain control sequences, or perhaps converted into an STS or EPT report. Other data obtained through the EDA interface will be linked together with JOB information (e.g. CJ, PJ, carrier ID, slot ID, etc…) and used as process data. For APC purposes, process data and equipment health monitoring data may be linked together to yield suitable APC/FDC engine input. The examples below show the data life cycle steps from data generation to its consumption. The data through the same EDA interface can have different data performance attributes such as data resolution depending on data, and may have different data life cycle scenarios.

R3-1.1.1 *Data Life Cycle Schematic*

R3-1.1.1.1 The data life cycle schematic shows not only data acquisition but also utilization in many ways. The life cycles shown in the diagram are all possible life cycle scenarios. Although this standard does not define all of the life cycle scenarios (only scenarios #1 and #3), it is important this standard does not preclude other possible scenarios such as data query and application calls. The A interface of an equipment may have all of these data life cycle models depending on data utilization or applications.

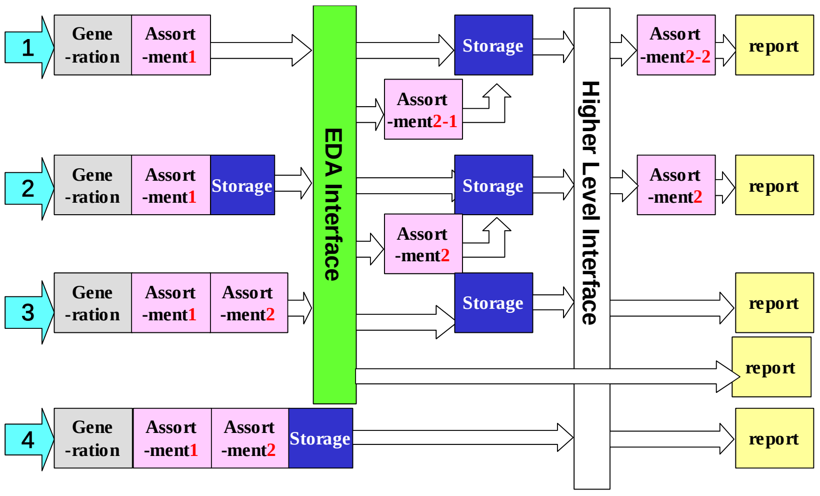

Figure R3-1
The Data Life Cycles

Table R3-1 Possible Data Life Cycle Scenarios

Steps	*Description*
Generation	Generation of data which can be reported out of the equipment.
Assortment 1	Least value adding processing for data handling; give tag such as name or index or time stamp.
Assortment 2, 2-1, 2-2	Make collected data be easily utilized by applications; linkage with other data or any value adding data processing (calculation of average, max. or min.).
Storage	Store EE data for a certain period of time. May provide query function from the higher level applications. This includes storage of value added data created by assortment 2.
Report	Output by a certain application to viewers or in the form of files.

R3-1.1.1.2 The form of the data may vary depending on how much capability the equipment has to process the data, or on the resolution that the application requires.

R3-1.1.1.3 Above data life cycle schematic show that:

1) The data resolution requirement varies with data itself, the application, or the system.

- Detailed equipment data such as detailed equipment event data (DEE)
- Data associated with JOB information
- Processed and value added data

2) Data acquisition method may vary.

- Trace data requirement
- Query (this is not the scope of this standard)
- Subscription by application

R3-1.1.2 *Example of Data Life Cycle for Low Level Data such as Detailed Equipment Data (DEE)*

R3-1.1.2.1 This example describes how the Calculation of equipment utilization Data for EPT and STS may be generated from low level equipment data.

Step	*Description*
Assortment 1	Input data is I/O event data of sensors and actuators. Give name and time stamp.
Assortment 2-1	From the consequence abstract the sequence start and stop events. Link with equipment structure model or JOB information.
Assortment 2-2	Calculate necessary indices.
Storage	Store calculated data. Provide subscription capability to show past data.
Report	Report or show.

Scenario	*Function/application in equipment*	*Higher level applications*
Scenario 1	Give elementary data its name and time stamp.	Detect necessary sequence starts and stops so as to calculate necessary indices. Report formation function.
Scenario 4	Calculation for EPT/STS embedded in equipment and put out data.	Application acquires the result of equipment utilization calculation, and this application only has viewer output function.

R3-1.1.3 *Equipment Health Monitoring from Analog Data and Low Level Equipment Data*

R3-1.1.3.1 This example describes a health check of vacuum level for each wafer.

Step	*Description*
Assortment 1	Input is analog data from vacuum gauge and time stamp will be given.
Assortment 2-1	Input is detailed equipment event data, and pump down sequence is detected for starts and ends, and these events are linked with JOB information and with carrier ID, slot ID and so on.
Assortment 2-2	The vacuum level is compared with reference data per wafer so as to examine if there is vacuum level deterioration.
Storage	Vacuum level data, its consequence or context, and the result of judgment are stored for a certain period of time. This provides subscription function from the higher level application.
Report	Application acquires the result, and this application has viewer output function.

Scenario	*Function/application in equipment*	*Higher level applications*
Scenario 1	Low level data such as detailed equipment event data is given timestamp.	Pump down sequence is detected for starts and ends, compare with reference data per wafer so as to detect any problem. Problem will be reported.
Scenario 3	Pump down sequence is detected for starts and ends, compare with reference data per wafer so as to detect any problem.	Acquire result of malfunction detection from equipment. Show the result.

R3-1.1.4 *Data Available from EDA Port*

R3-1.1.4.1 The data available from the EDA port contains some reporting information such as EPT. These reports are high-level equipment management information utilized by the fab-wide productivity monitoring application. Logical model based data includes the status, for example, of the load port, or job status (CJ and PJ). These data have the data life cycle of scenario #3.

R3-1.1.4.2 The other type of the data available from the EDA port is equipment data such as I/O status, sensor data, and all the equipment constants. These data have the data life cycle of scenarios #1 and #3.

R3-1.1.4.3 Both areas of data should be available for efficient equipment management and OEE improvement.

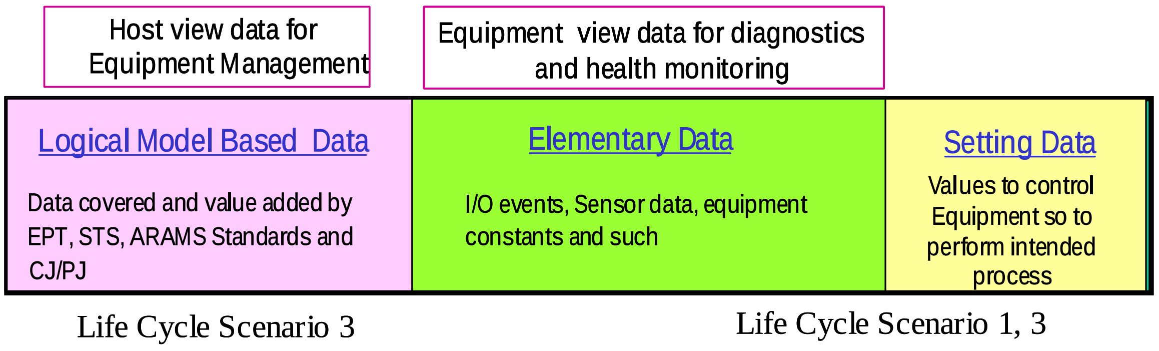
Host view data for Equipment Management
Equipment view data for diagnostics and health monitoring
Logical Model Based Data
Data covered and value added by EPT, STS, ARAMS Standards and CJ/PJ
Elementary Data
I/O events, Sensor data, equipment constants and such
Setting Data
Values to control Equipment so to perform intended process
Life Cycle Scenario 3
Life Cycle Scenario 1, 3

Figure R3-2
Data Available from EDA Port

R3-1.1.4.4 In the case where the equipment sends out equipment data right after "assortment 1", the application above EDA interface needs to have such functions as to link the equipment data with JOB information, to compare the data with some threshold values, or to calculate for statistic data abstraction. JOB information, the setting values which may be described in the recipes or used in the equipment internally, the equipment control structure, or the data naming conventions are to be made available through EDA port or some other means.

R3-1.1.4.5 The data that are supposed to be made available through the EDA port are listed below.

1) Control Job related

1. ControlJob ID
2. CarrierInputSpec: Carrier ID list
3. ProcessingCtrlSpec: PRjob list
4. ProcessOrderMgmt: PRjob execution sequence
5. MtrlOutSpec: Wafer origin and target specification

2) PR Job related

1. PRJobID
2. PRMtlType: Unit of material
3. PRMtlNameList: material name list
4. RecID
5. RecVariableList: Variable parameter name list and values
6. PRProcessStart: start method

   NOTE 1: If process condition is changed, the content is to be reported for the name of parameters and values.

3) Recipe related

1. Recipe ID
2. Recipe body (detailed content)
3. Setting value data the equipment use internally for each of the process control devices (temperature controllers, mass flow controllers and such)

RELATED INFORMATION 4
MESSAGE EXAMPLES

NOTICE: This related information is not an official part of SEMI PR8. This related information is not intended to modify or supersede the official proposed standard. Determination of the suitability of the material is solely the responsibility of the user.

R4-1 Example EDA Messages

R4-1.1 This section provides simple examples of each of the EDA messages defined in the specification. HTTP headers are shown in full only for the IsEdaEnabled request. Thereafter, only the SOAPAction header is shown for brevity.

R4-1.2 *Data Management Messages (supported by the equipment)*

R4-1.2.1 *IsEdaEnabled Request*

```
POST /EDAEquipmentService HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; MS Web Services Client Protocol
1.0.3705.0)
Content-Type: text/xml; charset=utf-8
SOAPAction: "urn:semi-org:ws:eda_ps_v0.0:IsEdaEnabled"
Content-Length: 433
Expect: 100-continue
Connection: Keep-Alive
Host: localhost
```

```
<?xml version="1.0" encoding="utf-8" ?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <soap:Header>
        <MessageHeader soap:mustUnderstand="1" xmlns="urn:semi-org:schema:eda_ps_v0.0">
            <To>urn:robofurnace:zippo:furnace-00899</To>
            <From>urn:icm:equipment.client:app-1</From>
        </MessageHeader>
    </soap:Header>
    <soap:Body>
        <IsEdaEnabled xmlns="urn:semi-org:schema:eda_ps_v0.0">
            <EquipmentID>
                <Supplier>RoboFurnace, Inc.</Supplier>
                <Model>Zippo 355</Model>
                <ImmutableID>39d-JDII-Uj399</ImmutableID>
            </EquipmentID>
        </IsEdaEnabled>
    </soap:Body>
</soap:Envelope>
```