

*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/>)

*Web Services Description Language (WSDL) 1.1* — W3C Note, (<http://www.w3.org/TR/wsdl>)

*Simple Object Access Protocol (SOAP) 1.1* — W3C Note, (<http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>)

#### 4.4 Other Standards

Secure Sockets Layer (SSL) Protocol 3.0 (available from <http://wp.netscape.com/eng/ssl3/draft302.txt>)

Web Services Interoperability (WS-I): Basic Profile Version 1.0a (available from <http://www.ws-i.org/Profiles/Basic/2003-08/BasicProfile-1.0a.html>)

SEMI Specification for XML Semiconductor Common Components

SEMI Specification for the Representation of Measurement Units in XML

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

## 5 Terminology

### 5.1 Abbreviations and Acronyms

5.1.1 *SOAP* — Simple Object Access Protocol

5.1.2 *UML* — Unified Modeling Language

5.1.3 *W3C* — World Wide Web Consortium

5.1.4 *WSDL* — Web Service Description Language

5.1.5 *XML* — eXtensible Markup Language

### 5.2 Definitions

5.2.1 *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.

5.2.2 *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.

## 6 Conventions

### 6.1 Translating UML to XML Schema

6.1.1 The reader is expected to have a working knowledge of the UML, XML, WSDL, SOAP, and Schema specifications (see ¶¶4.2 and 4.3). This document does not provide tutorial information on these subjects.

6.1.2 This document follows the guidelines for XML as outlined in SEMI E121.

6.1.3 Some of the key guidelines followed in this document are summarized here:

- Attributes of UML classes from SEMI E125 are generally represented as XML attributes, to improve efficiency in transferring metadata descriptions from the equipment. Exceptions to this convention are UML attributes that correspond to data structures or other constructs that cannot be represented as XML attributes, or may include text values that may need to include non-parsed character data (CDATA).
- Composition associations are mapped to a single XML element (or multiple such elements if the specified multiplicity is greater than 1).

- Aggregations are mapped to contained elements or arrays of references to one or more unique attributes of the target type.
- Inheritance is modeled using complexType extension. In cases where a reference to an abstract base class occurs in the UML model, a choice compositor for each of the possible derived types is used in order to restrict instance documents from including arbitrary extension through type substitution.

6.1.4 The translation of a UML class to XML Schema types is documented using a table format illustrated by Table 1. Operations defined for a UML class are translated into WSDL operations.

#### 6.1.5 Translation Table Column Header Description

- *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.
- *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 association types are “Composition”, “Aggregation”, or the basic “Association”. UML defines these three types of association.
- *XML Element or Attribute* — Lists the type of XML construct used to represent the UML attribute or association.
- *XML Name/Type* — Provides the name and data type of the resulting XML construct. The type may be a built-in type (for example, xsd:string), or a named type defined within the XML Schema.

**Table 1 Example Translation Table**

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

#### 6.1.6 Translating Operations to XML Schema

6.1.6.1 The translation of an operation defined for UML interface classes to XML Schema types is documented using a table format illustrated in Table 2. One row is provided for each possible input/output argument for the operation being described.

**Table 2 Example Translation Table for Operation Input/Output Arguments**

<i>Argument Name</i>	<i>Format</i>	<i>XML Element or Attribute</i>	<i>XML Name/Type</i>
<argument name from abstract specification>	<argument format from abstract specification>	<whether the argument is modeled as an Element or Attribute>	<namespace-qualified name of the attribute/element>

#### 6.2 Documenting XML Schema and WSDL Files

6.2.1 Associated with this document are XML Schema and WSDL files that are core components of the specification. Each such document is described using a table format illustrated by Table 3.

**Table 3 Example Schema/WSDL Document Description Table**







<i>File Name</i>	<.xsd or .wsdl file name>
<i>Target Namespace</i>	<target namespace defined in the schema or WSDL file>
<i>Imported/Referenced Namespaces</i>	<namespaces imported or used by the schema or WSDL file>
<i>Description</i>	<brief description of the purpose/function of the schema or WSDL file>

### 6.3 Documenting XML Schema with Diagrams<sup>3</sup>

6.3.1 This document provides graphical representations of the included XML Schema data type and element definitions. Although no standard graphical notation for XML Schema could be found, various XML tools have their own notation. This document will use the notation provided by XMLSpy from Altova Corporation. Figure 1 shows a sample XML Schema diagram that will be used to provide a basis for explanation of the schema graphical notation used in the rest of the document.

6.3.2 In the diagram, rectangular boxes represent XML element definitions. Ownership or containment is read from left to right in the diagrams. In the sample diagram, ParentType contains Child1, Child2, Child 3, and Child 4. In turn, Child2 contains Child2a, Child2b, and Child2c. The additional symbols (8-sided boxes) represent sequences or choices. See Table 4 for an explanation of these symbols.

**Table 4 Altova XMLSPY Schema Diagram Symbols**

	Denotes a required ordered sequence of the right hand elements with a cardinality of one for each element. (sequence)
	Denotes an optional ordered sequence of the right hand elements with a cardinality of one for each element. (sequence)
	Denotes a required, but unordered, sequence of the right hand elements with a cardinality of one for each element. (all)
 1..2	Denotes a required choice of the right hand elements. Exactly one or two of the right hand elements must be present. (choice)
	Denotes an element that contains parsed character data
	Denotes a reference to a group or element defined elsewhere

6.3.3 A graphic using a solid line is a required element; using a dashed line represents an optional element. Numbers or ranges in the lower right hand corner represent cardinality. The default cardinality is one.

6.3.4 To simplify a diagram or help focus on a particular aspect, detail may be hidden. The 8-sided symbols have a small square on the right end. If a minus sign “-” is in the box, then all detail is shown. If the box contains a plus sign “+”, then all detail to the right of that symbol is hidden. The example has no hidden detail.

6.3.5 The yellow (or grey if printed in monochrome) boxes indicate the use of other defined types. So, Child4 is of type “Child4Type”. Child4Type defines Child4a and Child4b. This detail may be hidden in the diagram. Object oriented inheritance is typically represented in XML as type extension. In Figure 1, ParentType extends Child1and 2Type by adding a sequence that includes Child3 and Child4.

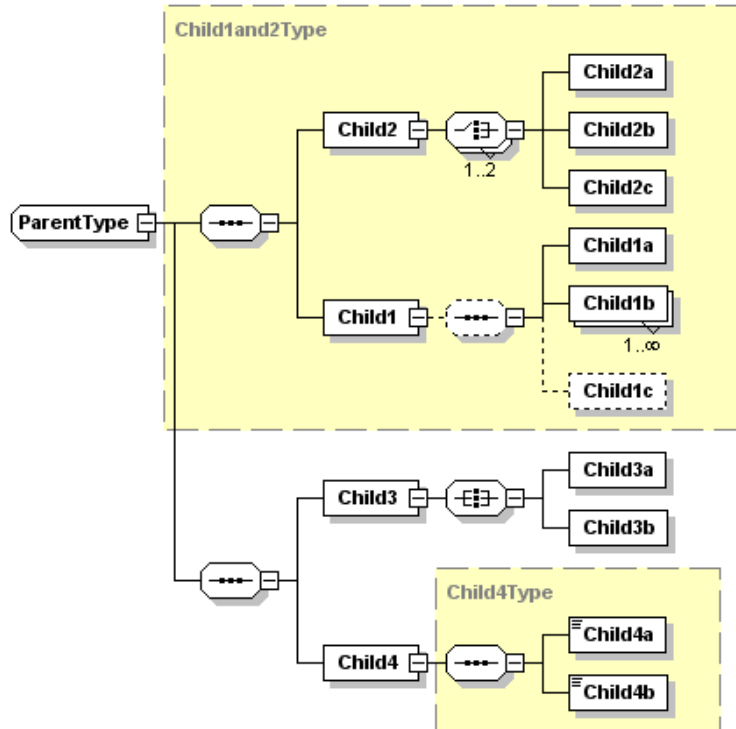
6.3.6 Reading Figure 1 would yield the following additional information:

1. Child1and2Type is an ordered sequence of two items: Child2 and Child1.
2. Child1 contains an optional ordered sequence of Child1a, one or more Child1b, and (optionally) Child1c.
3. Child2 contains a choice of one or two of the following: Child2a, Child2b, and Child2c.
4. Child3 contains an unordered sequence of Child3a and Child3b.

### 6.4 XML Schema Sample

6.4.1 The sample XML Schema for the example shown in Figure 1, is presented below. Refer to the XML documentation referenced in ¶4.3 for a complete description of the syntax and semantics of XML Schema.

<sup>3</sup> All images/graphics were created using Altova’s XMLSPY®. Copyright 2003 Altova GmbH and reprinted with permission of Altova.



**Figure 1**  
**XML Schema Example Diagram**

```
<xsd:element name="Parent">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="Child1"/>
      <xsd:complexType>
        <xsd:sequence minOccurs="0">
          <xsd:element name="Child1a"/>
          <xsd:element name="Child1b" maxOccurs="unbounded"/>
          <xsd:element name="Child1c" minOccurs="0"/>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="Child2">
      <xsd:complexType>
        <xsd:choice maxOccurs="2">
          <xsd:element name="Child2a"/>
          <xsd:element name="Child2b"/>
          <xsd:element name="Child2c"/>
        </xsd:choice>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="Child3">
      <xsd:complexType>
        <xsd:all>
          <xsd:element name="Child3a"/>
          <xsd:element name="Child3b"/>
        </xsd:all>
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
</xsd:element>
```

**Figure 2**  
**XML for Sample**

## 6.5 Translating UML to WSDL

6.5.1 In general, UML interface classes defined in the abstract specification are translated into WSDL as portType definitions, and each portType definition has a corresponding WSDL binding definition. The following tables show the convention used for documenting the WSDL port type and binding definitions for a given UML interface class.

**Table 5 Example Interface WSDL Port Type Table**

<i>Class Name</i>	<UML interface name from abstract specification>
<i>WSDL Port Type Name</i>	<port type name used in WSDL>
<i>SEMI E125 Operation</i> → <i>WSDL Operation</i>	<operation name from UML interface> → <WSDL port type operation name>

**Table 6 Example Interface WSDL Binding Table**

<i>SEMI E125 Class Name</i>	<UML interface name from abstract specification>
<i>WSDL Binding Name</i>	<binding name used in WSDL>
<i>SOAP Binding Style</i>	<RPC or document>
<i>SOAP Transport</i>	<transport identifying URI>

6.5.2 Each operation defined for a given UML interface class has a corresponding operation definition and binding. Operations are described using a table format illustrated by Table 7 and Table 8.

**Table 7 Example Operation Binding Table**

<i>SOAPAction</i>	<the SOAPAction HTTP header value to be used for this operation>
<i>Input Headers (WSDL Message, Required)</i>	<the name of the WSDL message providing input headers, and whether or not the headers are required>
<i>Output Headers (WSDL Message, Required)</i>	<the name of the WSDL message providing output headers, and whether or not the headers are required>

**Table 8 Example PortType Operation Table**

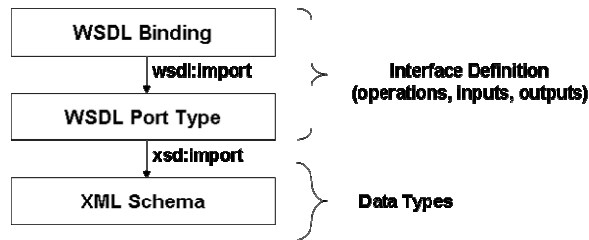
<i>Input Message Name</i>	<WSDL message name used as input for the WSDL port type operation>
<i>Output Message Name</i>	<WSDL message name used as output for the WSDL port type operation>

## 7 Mapping of SEMI E125 UML to XML Schema and WSDL

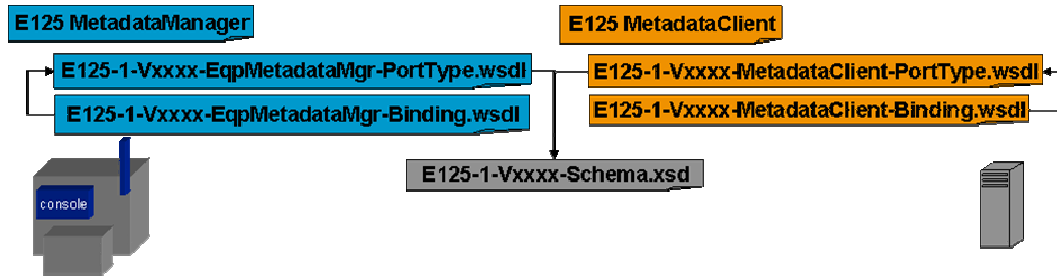
### 7.1 WSDL Organization

7.1.1 Each interface definition in SEMI E125 is mapped to a WSDL portType and binding definition. WSDL portType definitions are named after the interface and its operations as they appear in SEMI E125. WSDL binding definitions for each portType are used to specify the SOAP 1.1 envelope contents for each operation, and to define the corresponding XML encoding styles and HTTP header usage. All SEMI E125 WSDL interfaces use document/literal encoding, with the complete SOAP header and body contents defined in XML Schema file(s) via global element definitions.

7.1.2 Figure 1 shows the relationship between the WSDL binding and portType definitions and the XML Schema types used for each interface. The portType definition imports XML Schema type definitions used in each operation via the XML Schema “import” statement. The WSDL binding definition imports the portType definition via the WSDL “import” statement. Figure 2 shows the specific XML Schema and WSDL files defined for SEMI E125.



**Figure 3**  
**XML Schema and WSDL File Organization**



**Figure 4**  
**WSDL and XML Schema Files for SEMI E125**

## 7.2 *EquipmentMetadataManager*

### 7.2.1 *Application of SEMI E132 Sessions*

7.2.1.1 This section is provisional, pending the approval of SEMI E132.1 and the SEMI E138.

7.2.1.2 Although no special privileges are specified by SEMI E125 in order to access equipment metadata, SEMI E125 information is proprietary to the equipment supplier, and should be protected against unauthorized access within the factory.

7.2.1.3 For this reason, all operations defined by the SEMI E125 EquipmentMetadataManager interface can be exchanged using the HTTPS transport (HTTP 1.1 over SSL 3.0) by enabling SSL authentication as specified in SEMI E132.1. Once enabled via SEMI E132.1, the equipment shall require mutual client-server SSL authentication for all SEMI E125 operations supported by the equipment, and the equipment shall not provide access to any operations except via the HTTPS transport.

7.2.2 There may be some integration, development, and/or test environments in which it is not practical or feasible to install equipment and/or client certificates in order to communicate with the equipment. As a practical consideration for such environments only, the equipment shall support the ability of the user to disable SSL authentication for the SEMI E125 EquipmentMetadataManager interface, using the equipment configuration specified by SEMI E132.1. Users should be aware that disabling SSL authentication opens up complete unrestricted access to SEMI E125 data for any application with network access to the equipment.

7.2.2.1 Regardless of whether or not SSL authentication is enabled, the equipment shall reject all requests from clients that do not provide a valid SEMI E132 session identifier in the SOAP envelope header provided with the request. The SEMI E132 session identifier shall be provided via the SEMI E132 Header element, as specified by SEMI E132.1. If a client does not provide a recognized session identifier, the equipment shall immediately reject the request using the SEMI Common Error data type, with an error code of '6005' and a source of 'urn:semi-org:E132' as specified in SEMI E132.1. The equipment shall perform no other processing of a SEMI E125 request if the request is found to include an invalid SEMI E132 session identifier, or if no session identifier is provided.

### 7.2.3 *Use of Common Error Type*

7.2.3.1 This section is provisional, pending the approval of SEMI E138.

7.2.3.2 This specification does not define any SEMI E125-specific error codes, but reserves the common error type “source” name of “urn:semi-org:E125” for future use.

#### 7.2.4 XML Schema and WSDL Files

7.2.4.1 This section is provisional, pending the approval of SEMI E132.1 and SEMI E120.1.

7.2.4.2 The XML Schema and WSDL defined by this specification for the EquipmentMetadataManager interface is contained in the following documents:

**Table 9 XML Schema**

<i>File Name</i>	E125-1-V0305-Schema.xsd
<i>Target Namespace</i>	urn:semi-org:xsd:E125-1.V0305.esd
<i>Imported/Referenced Namespaces</i>	urn:semi-org:xsd:E120-1.V1104.CommonEquipmentModel urn:semi-org:xsd:CommonComponents.V0305.ccs <a href="http://www.w3.org/2001/XMLSchema">http://www.w3.org/2001/XMLSchema</a>
<i>Description</i>	This file defines all of the E125 data types and elements used by both the EquipmentMetadataManager and MetadataClient interfaces.

**Table 10 EquipmentMetadataManager PortType Definitions**

<i>File Name</i>	E125-1-V0305-EqpMetadataMgr-PortType.wsdl
<i>Target Namespace</i>	urn:semi-org:ws:E125-1.V0305.esdMetaEqp-portType
<i>Imported/Referenced Namespaces</i>	urn:semi-org:xsd:E132-1.V0305.auth urn:semi-org:xsd:E125-1.V0305.esd <a href="http://schemas.xmlsoap.org/wsdl/">http://schemas.xmlsoap.org/wsdl/</a> <a href="http://www.w3.org/2001/XMLSchema">http://www.w3.org/2001/XMLSchema</a>
<i>Description</i>	This file defines all of the input/output messages and operations for the EquipmentMetadataManager interface, based on the data types defined in the SEMI E125 XML Schema.

**Table 11 EquipmentMetadataManager Binding Definitions**

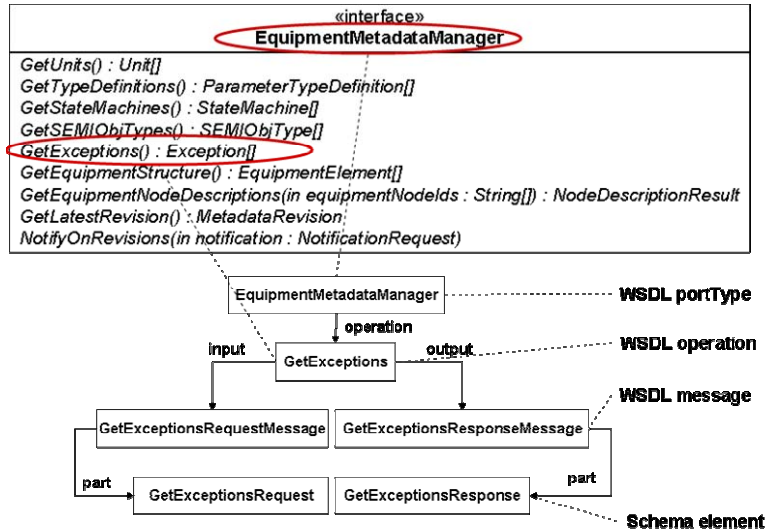
<i>File Name</i>	E125-1-V0305-EqpMetadataMgr-Binding.wsdl
<i>Target Namespace</i>	urn:semi-org:ws:E125-1.V0305.esdMetaEqp-binding
<i>Imported/Referenced Namespaces</i>	urn:semi-org:ws:E125-1.V0305.esdMetaEqp-portType <a href="http://schemas.xmlsoap.org/wsdl/soap/">http://schemas.xmlsoap.org/wsdl/soap/</a> <a href="http://schemas.xmlsoap.org/wsdl/">http://schemas.xmlsoap.org/wsdl/</a>
<i>Description</i>	This file binds the abstract portType definition to HTTP and SOAP, specifying required SOAP and HTTP headers for each operation and the XML encoding style.

7.2.4.3 These documents are a part of this specification and should accompany this document. The contents of these documents constitute the core part of this specification.

7.2.4.4 E125-1-V0305-Schema.xsd imports types from the SEMI E120.1 Common Equipment Model XML Schema namespace

#### 7.2.5 WSDL Port Type Overview

7.2.5.1 The EquipmentMetadataManager WSDL portType definition organizes the SEMI E125.1 XML Schema data types into a collection of named operations with inputs and outputs that correspond to the UML operations that are defined for the EquipmentMetadataManager interface in SEMI E125. The WSDL portType itself is named after the SEMI E125 interface, and each WSDL portType operation is named after the corresponding UML operation defined for the interface. Each WSDL operation consists of two WSDL message definitions corresponding to the input and output (request and response) for the UML operation defined in SEMI E125. Figure 3 illustrates the relationship between the SEMI E125 UML interface and the WSDL portType definition; Table 12 describes this in tabular form. The WSDL message and operation definitions are described in ¶¶7.2.8 through 7.2.16 .



**Figure 5**  
Mapping the EquipmentMetadataManager to a WSDL portType

**Table 12 EquipmentMetadataManager Port Type**

<i>SEMI E125 Class Name</i>	EquipmentMetadataManager
<i>WSDL Port Type Name</i>	EquipmentMetadataManager
<i>SEMI E125 Operation → WSDL Operation</i>	GetUnits → GetUnits GetTypeDefinitions → GetTypeDefinitions GetStateMachines → GetStateMachines GetSEMIObjTypes → GetSEMIObjTypes GetExceptions → GetExceptions GetEquipmentStructure → GetEquipmentStructure GetEquipmentNodeDescriptions → GetEquipmentNodeDescriptions GetLatestRevision → GetLatestRevision NotifyOnRevisions → NotifyOnRevisions

## 7.2.6 WSDL Binding Overview

7.2.6.1 The WSDL EquipmentMetadataManager binding definition specifies a message and transport protocol to use for a given portType (SOAP and HTTP for SEMI E125.1), the interface style used (document style for SEMI E125.1). These settings are shown in Table 13. For each WSDL portType operation, the binding defines any SOAP headers used and whether or not they are required, the HTTP SOAPAction header value to use for that operation, and the XML encoding to use (literal). The binding definitions for each portType operation are described in ¶¶7.2.8 through 7.2.16.

**Table 13 EquipmentMetadataManager Binding**

<i>E125 Class Name</i>	EquipmentMetadataManager
<i>WSDL Binding Name</i>	EquipmentMetadataManagerBinding
<i>SOAP Binding Style</i>	document
<i>SOAP Transport</i>	http://schemas.xmlsoap.org/soap/http

## 7.2.7 WSDL Service Overview

7.2.7.1 This specification does not provide a WSDL service definition. WSDL service definitions provide one way to define and locate the endpoint(s) at which a given interface can be accessed by a client. Such information cannot be determined until the equipment has been installed on a factory network, and therefore is out of scope for this specification. Users are responsible for defining the mechanisms by which clients locate SEMI E125 equipment services in the factory.

## 7.2.8 GetUnits

7.2.8.1 This section is provisional, pending approval of SEMI E132.1 and SEMI E138.

### 7.2.8.2 WSDL Operation Binding

7.2.8.2.1 See the operation binding for “GetUnits” in E125-1-V0305-EqpMetadataMgr-Binding.wsdl. Key information is shown for convenience in the following table.

**Table 14 GetUnits Operation Binding**

<i>SOAPAction</i>	urn:semi-org:ws.E125-1.V0305.esdMetaEqp-binding:GetUnits
<i>Input Headers (WSDL Message, Required)</i>	E132HeaderMessage, required
<i>Output Headers (WSDL Message, Required)</i>	E132HeaderMessage, required

### 7.2.8.3 WSDL PortType Operation

7.2.8.3.1 See the portType operation definition for “GetUnits” in E125-1-V0305-EqpMetadataMgr-PortType.wsdl. Key information is shown for convenience in the following table.

**Table 15 GetUnits PortType Operation**

<i>Input Message Name</i>	GetUnitsRequestMessage
<i>Output Message Name</i>	GetUnitsResponseMessage

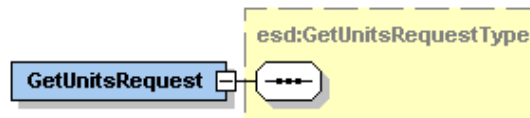
### 7.2.8.4 WSDL Message(s)

7.2.8.4.1 See the message definitions for “E132HeaderMessage”, “GetUnitsRequestMessage”, and “GetUnitsResponseMessage” in E125-1-V0305-EqpMetadataMgr-PortType.wsdl. Key information is shown for convenience in the following table.

**Table 16 EquipmentMetadataManager Binding**

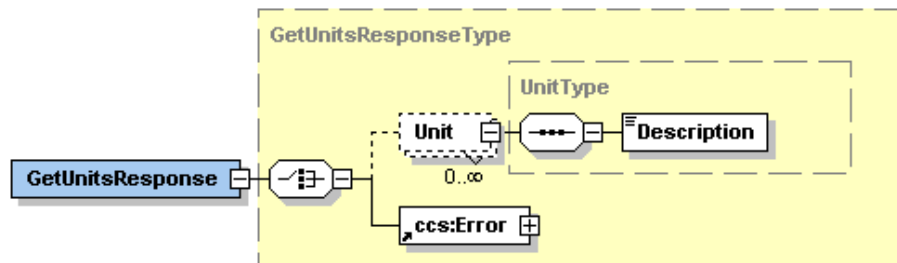
<i>Message Name → Schema Element Name</i>	E132HeaderMessage → auth:E132Header GetUnitsRequestMessage → esd:GetUnitsRequest GetUnitsResponseMessage → esd:GetUnitsResponse
---	---

### 7.2.8.5 XML Schema Types



**Figure 6**  
**GetUnitsRequest Global Element**

7.2.8.5.1 *GetUnitsRequest* — This global element acts as a top-level container entity representing the SOAP body of a GetUnits request. As defined in SEMI E125, this operation has no input arguments, so the element has no contents.



**Figure 7**  
**GetUnitsResponse Global Element**

7.2.8.5.2 *GetUnitsResponse* — This global element acts as a top-level container entity representing the SOAP body of the response to an SEMI E125 GetUnits request. It contains either zero or more elements named “Unit” of complexType “UnitType” or an element named “Error” of complexType “ErrorType”, defined in SEMI E138.

**Table 17 Translation Table for Output Arguments for the GetUnits Operation**

<i>Argument Name</i>	<i>Format</i>	<i>XML Element or Attribute</i>	<i>XML Name/Type</i>
units	Unordered list of structured data, of type Unit.	Element	Unit: UnitType (zero or more)
error	Structured data, of type Error, defined in the SEMI Specification for Semiconductor Common Components.	Element	Error: ccs:ErrorType

7.2.8.5.3 *UnitType* — This global complexType is the XML Schema representation of the SEMI E125 Unit class. The following table describes the mapping of the UML class to this XML Schema complexType.

**Table 18 E125 Unit → UnitType Translation Table**

<i>Attribute or Role Name</i>	<i>UML Name/Type</i>	<i>XML Element or Attribute</i>	<i>XML Name/Type</i>
id	text	attribute	id: xsd:string
name	text	attribute	name: xsd:string
symbol	text	attribute	symbol: xsd:string
description	text	element	Description: xsd:string

## 7.2.9 *GetTypeDefinitions*

7.2.9.1 This section is provisional, pending approval of SEMI E132.1 and SEMI E138

### 7.2.9.2 *WSDL Operation Binding*

7.2.9.2.1 See the operation binding for “GetTypeDefinitions” in E125-1-V0305-EqpMetadataMgr-Binding.wsdl. Key information is shown for convenience in the following table.

**Table 19 GetTypeDefinitions Operation Binding**

<i>SOAPAction</i>	urn:semi-org:ws.E125-1.V0305.esdMetaEqp-binding:GetTypeDefinitions
<i>Input Headers (WSDL Message, Required)</i>	E132HeaderMessage, required
<i>Output Headers (WSDL Message, Required)</i>	E132HeaderMessage, required

### 7.2.9.3 *WSDL PortType Operation*

7.2.9.3.1 See the portType operation definition for “GetTypeDefinitions” in E125-1-V0305-EqpMetadataMgr-PortType.wsdl. Key information is shown for convenience in the following table.

**Table 20 GetUnits PortType Operation**

<i>Input Message Name</i>	GetTypeDefinitionsRequestMessage
<i>Output Message Name</i>	GetTypeDefinitionsResponseMessage

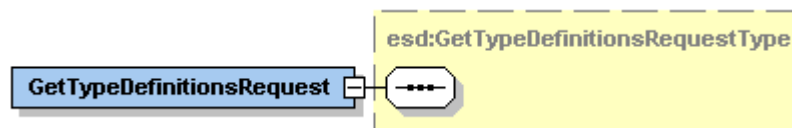
#### 7.2.9.4 WSDL Message(s)

7.2.9.4.1 See the message definitions for “E132HeaderMessage”, “GetTypeDefinitionsRequestMessage”, and “GetTypeDefinitionsResponseMessage” in E125-1-V0305-EqpMetadataMgr-PortType.wsdl. Key information is shown for convenience in the following table.

**Table 21 EquipmentMetadataManager Binding**

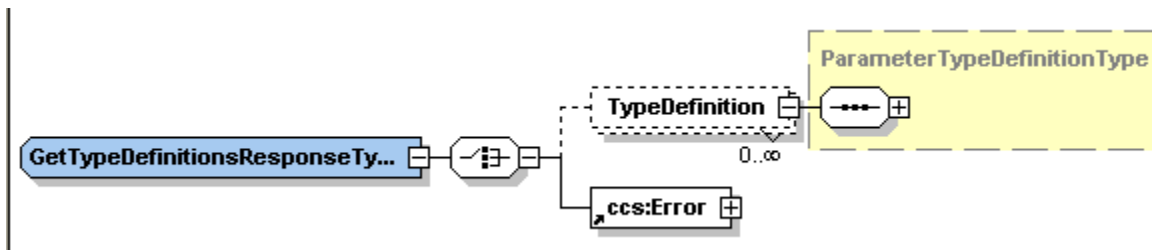
<i>Message Name → Schema Element Name</i>	E132HeaderMessage → auth:E132Header GetTypeDefinitionsRequestMessage → esd:GetTypeDefinitionsRequest GetTypeDefinitionsResponseMessage → esd:GetTypeDefinitionsResponse
---	---

#### 7.2.9.5 XML Schema Types



**Figure 8**  
**GetTypeDefinitionsRequest Global Element**

7.2.9.5.1 *GetTypeDefinitionsRequest* — This global element acts as a top-level container entity representing the SOAP body of a GetTypeDefinitions request. As defined in SEMI E125, this operation has no input arguments, so the element has no contents.



**Figure 9**  
**GetTypeDefinitionsResponse Global Element**

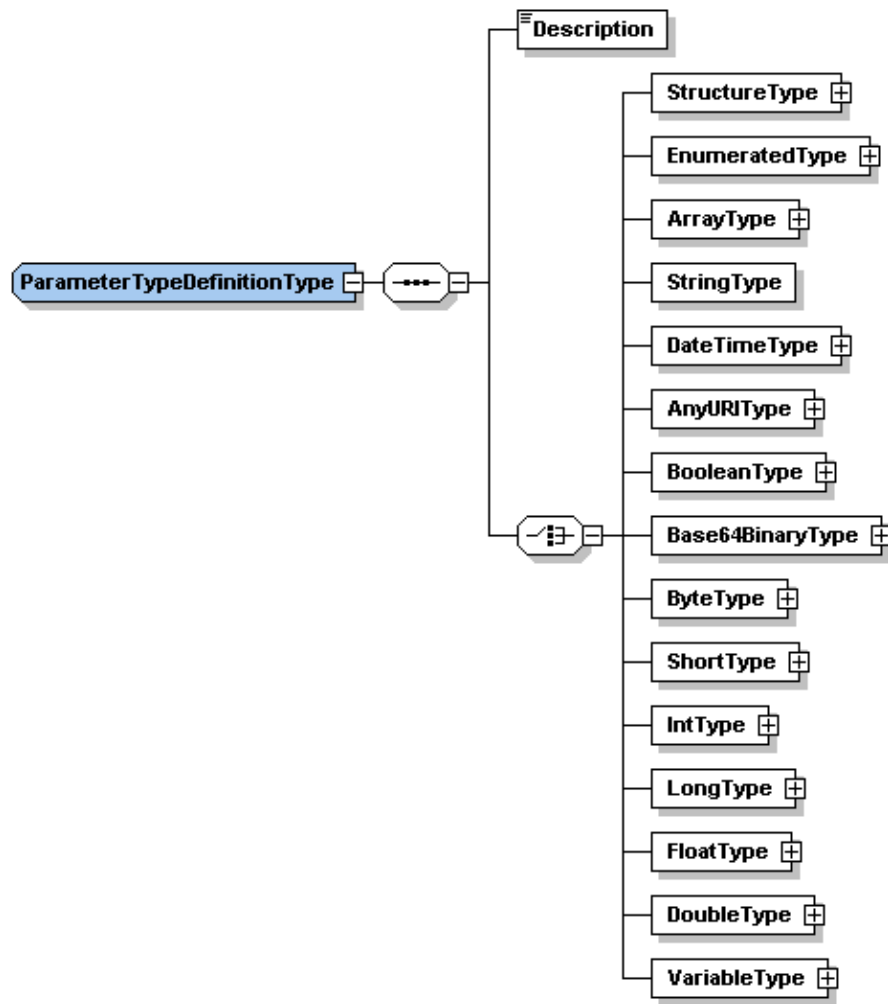
7.2.9.5.2 *GetTypeDefinitionsResponse* — This global element acts as a top-level container entity representing the SOAP body of the response to an SEMI E125 GetTypeDefinitions request. It contains either zero or more elements named “TypeDefinition” of complexType “ParameterTypeDefinitionType” or an element named “Error” of complexType “ErrorType”, defined in SEMI E138.

**Table 22 Translation Table for Output Arguments for the GetTypeDefinitions Operation**

<i>Argument Name</i>	<i>Format</i>	<i>XML Element or Attribute</i>	<i>XML Name/Type</i>
types	Unordered list of structured data, of type TypeDefinition.	Element	TypeDefinition: ParameterTypeDefinitionType (zero or more)
error	Structured data, of type Error, defined in SEMI E138.	Element	Error: ccs:ErrorType

7.2.9.5.3 *ParameterTypeDefinitionType* — This global complexType is the XML Schema representation of the E125 ParameterTypeDefinition class. The following table describes the mapping of the UML class to this XML Schema complexType. The association between the ParameterTypeDefinition class and the abstract base class

“ParameterType” is mapped using a reference to an XML Schema model group named “TypeDescriptionSelectionGroup”. This model group consists of a choice compositor (see ¶6.1.3) containing elements representing instances of each of the possible derived type description classes.



**Figure 10**  
ParameterTypeDefinitionType Global complexType

**Table 23 SEMI E125 ParameterTypeDefinition → ParameterTypeDefinitionType Translation Table**

Attribute or Role Name	UML Name/Type	XML Element or Attribute	XML Name/Type
name	text	attribute	name: xsd:string
description	text	element	Description: xsd:string
typeInformation	ParameterType	N/A (group reference)	TypeDescriptionSelectionGroup

#### 7.2.9.5.4 Extended Parameter Type Description

7.2.9.5.4.1 This specification adds new type description complexTypes to the minimal set defined in SEMI E125. These additional types correspond to built-in primitive and derived types defined by the XML Schema specifications, and a variable type for Parameters whose values have variable data types at runtime. If a Parameter’s type is described using the SEMI E125.1 complexTypes, the Parameter’s value in XML instance documents must be reported using the corresponding XML Schema types. The following table shows how each SEMI E125 type

description class has been mapped to the XML Schema complexTypes in this specification, and lists the additional type description classes not defined in SEMI E125. The table also shows the corresponding schema types that shall be used for reporting values in XML of Parameters that have been described using the SEMI E125.1 complexTypes.

**Table 24 SEMI E125 Type Description Mapping Table**

<i>E125 Type Description UML Class</i>	<i>E125.1 Type Description complexType</i>	<i>XML Schema Type of Parameter Values</i>
StringType	StringTypeType	xsd:string
BooleanType	BooleanTypeType	xsd:boolean
BinaryType	Base64BinaryTypeType	xsd:base64Binary
IntegerType	ByteTypeType, ShortTypeType, IntTypeType, LongTypeType	xsd:byte, xsd:short, xsd:int, xsd:long
RealType	FloatTypeType, DoubleTypeType	xsd:float, xsd:double
N/A	DateTimeTypeType	xsd:dateTime
N/A	AnyURITypeType	xsd:anyURI
N/A	VariableTypeType	See ¶7.2.9.5.16
ArrayType	ArrayTypeType	See ¶7.2.9.5.18.1
StructureType	StructureTypeType	See ¶7.2.9.5.19.1
EnumeratedType	EnumeratedTypeType	See ¶7.2.9.5.20.1

#### 7.2.9.5.5 SECS-II Type Mapping

7.2.9.5.5.1 Many equipment suppliers may provide data items via the traditional SECS/GEM control interface that can be described via SEMI E125. For consistency between those data types and their metadata descriptions, refer to ¶10.5.4 of SEMI E125. For specific types supported in SECS-II, see the following table.

**Table 25 SECS-II Type Description Mapping Table**

<i>SECS-II Type Name</i>	<i>SEMI E125 Type Description</i>	<i>XMLSchema Built-In Base Type</i>
Binary	Base64BinaryType	xsd:base64Binary
Boolean	BooleanType	xsd:boolean
ASCII	StringType	xsd:string
JIS-8	StringType	xsd:string
2 byte character	StringType	xsd:string
8 byte integer (signed)	LongType	xsd:long
1 byte integer (signed)	ByteType	xsd:byte
2 byte integer (signed)	ShortType	xsd:short
4 byte integer (signed)	IntType	xsd:int
8 byte floating point	DoubleType	xsd:double
4 byte floating point	FloatType	xsd:float
8 byte integer (unsigned)	LongType, Base64Binary	xsd:long, xsd:base64Binary (big-endian)
1 byte integer (unsigned)	ShortType	xsd:short
2 byte integer (unsigned)	IntType	xsd:int
4 byte integer (unsigned)	LongType	xsd:long
List	ArrayType, StructureType	n/a

NOTE 2: For 8-byte unsigned integers, it is recommended either that such values are described as LongType if the actual value range in use can be stored via an unsigned 8-byte integer, or sent as a (big-endian) byte array. In general, use of unsigned integer types is discouraged.

7.2.9.5.6 *StringTypeType* — This global complexType is the XML Schema representation of the SEMI E125 StringType class. The following table describes the mapping of the UML class to this XML Schema complexType. The “language” attribute is of type “LanguageCodeType”, which is a regular expression that enforces the IETF RFC 1766 language code format specified in SEMI E125.

**Table 26 E125 StringType → StringType Translation Table**

<i>Attribute or Role Name</i>	<i>UML Name/Type</i>	<i>XML Element or Attribute</i>	<i>XML Name/Type</i>
language	text	attribute	language: esd:LanguageCodeType
maxCharacters	integer	attribute	maxCharacters: xsd:int

**7.2.9.5.7 BooleanTypeType** — This global complexType is the XML Schema representation of the SEMI E125 BooleanType class. No attributes are defined for this class in SEMI E125, thus the BooleanTypeType definition includes no elements or attributes.

**7.2.9.5.8 Base64BinaryTypeType** — This global complexType is the XML Schema representation of the SEM E125 BinaryType class. This specification adds an optional attribute to describe the format of the binary variable using industry standard MIME media type names.

**Table 27 SEMI E125 BinaryType → Base64BinaryTypeType Translation Table**

<i>Attribute or Role Name</i>	<i>UML Name/Type</i>	<i>XML Element or Attribute</i>	<i>XML Name/Type</i>
N/A	N/A	attribute	contentType: xsd:string

**7.2.9.5.9 ByteTypeType** – This global complexType is the XML Schema representation of the SEMI E125 IntegerType class. The following table describes the mapping of the UML class to this XML Schema complexType. The association from IntegerType to the Unit class in SEMI E125 has been modeled using a reference attribute that refers to the specific unit using its unique identifier. The association to the UnitConfig class has been modeled using a container element for a collection of unit id’s, a reference to the Parameter that controls their settings, and the enumeration settings that select those units.

**Table 28 SEMI E125 IntegerType → ByteTypeType Translation Table**

<i>Attribute or Role Name</i>	<i>UML Name/Type</i>	<i>XML Element or Attribute</i>	<i>XML Name/Type</i>
units	Unit	element	Units: esd:UnitRefType
units	UnitConfig	element	AvailableUnits: esd:UnitConfigArrayType (see ¶7.2.9.5.10).

**7.2.9.5.10 UnitConfigArrayType** — This global complexType is the XML Schema representation of the SEMI E125 UnitConfig class. It provides a reference to the configuration Parameter that controls the reported units for Parameters of this type, and a list of the units that can be selected by different settings for that configuration Parameter.

**Table 29 SEMI E125 UnitConfig → UnitConfigArrayType Translation Table**

<i>Attribute or Role Name</i>	<i>UML Name/Type</i>	<i>XML Element or Attribute</i>	<i>XML Name/Type</i>
configParameter	Parameter	attributes	configParamSource: cem:LocatorType configParamName: xsd:string See ¶¶7.2.9.5.10.1 and 7.2.9.5.10.2 for details.
unit	Unit	attribute, 1 for each entry in the UnitConfigArrayType.	unitId: xsd:string
setting	EnumeratedString	attribute, 1 for each entry in the UnitConfigArrayType.	setting: xsd:string See ¶7.2.9.5.10.3 for details.

**7.2.9.5.10.1 configParamSource** — This attribute shall be set to a valid CEM Locator identifying the equipment node that provides the unit configuration Parameter.

**7.2.9.5.10.2 configParamName** — This attribute shall be set to the value of the “name” attribute of the Parameter that establishes the units to be used.

7.2.9.5.10.3 *setting* — The value of this attribute shall be equal to one of the valid enumeration values defined for the Parameter identified by the configParamSource/Name attributes.

7.2.9.5.11 *ShortTypeType* — This global complexType is the XML Schema representation of the SEMI E125 IntegerType class. The following table describes the mapping of the UML class to this XML Schema complexType. The association from IntegerType to the Unit class in SEMI E125 has been modeled using a reference attribute that refers to the specific unit using its unique identifier. The association to the UnitConfig class has been modeled using a container element for a collection of unit id's, a reference to the Parameter that controls their settings, and the enumeration settings that select those units.

**Table 30 SEMI E125 IntegerType → ShortTypeType Translation Table**

<i>Attribute or Role Name</i>	<i>UML Name/Type</i>	<i>XML Element or Attribute</i>	<i>XML Name/Type</i>
units	Unit	element	Units: esd:UnitRefType
units	UnitConfig	element	AvailableUnits: esd:UnitConfigArrayType (See ¶7.2.9.5.10.)

7.2.9.5.12 *IntTypeType* — This global complexType is the XML Schema representation of the SEMI E125 IntegerType class. The following table describes the mapping of the UML class to this XML Schema complexType. The association from IntegerType to the Unit class in SEMI E125 has been modeled using a reference attribute that refers to the specific unit using its unique identifier. The association to the UnitConfig class has been modeled using a container element for a collection of unit ids, a reference to the Parameter that controls their settings, and the enumeration settings that select those units.

**Table 31 SEMI E125 IntegerType → IntTypeType Translation Table**

<i>Attribute or Role Name</i>	<i>UML Name/Type</i>	<i>XML Element or Attribute</i>	<i>XML Name/Type</i>
units	Unit	element	Units: esd:UnitRefType
units	UnitConfig	element	AvailableUnits: esd:UnitConfigArrayType (See ¶7.2.9.5.10.)

7.2.9.5.13 *LongTypeType* — This global complexType is the XML Schema representation of the SEMI E125 IntegerType class. The following table describes the mapping of the UML class to this XML Schema complexType. The association from IntegerType to the Unit class in SEMI E125 has been modeled using a reference attribute that refers to the specific unit using its unique identifier. The association to the UnitConfig class has been modeled using a container element for a collection of unit ids, a reference to the Parameter that controls their settings, and the enumeration settings that select those units.

**Table 32 E125 IntegerType → LongTypeType Translation Table**

<i>Attribute or Role Name</i>	<i>UML Name/Type</i>	<i>XML Element or Attribute</i>	<i>XML Name/Type</i>
units	Unit	element	Units: esd:UnitRefType
units	UnitConfig	element	AvailableUnits: esd:UnitConfigArrayType (See ¶7.2.9.5.10.)

7.2.9.5.14 *FloatTypeType* — This global complexType is an XML Schema representation of the SEMI E125 RealType class (see also DoubleTypeType). The following table describes the mapping of the UML class to this XML Schema complexType. The association from RealType to the Unit class in SEMI E125 has been modeled using a reference attribute that refers to the specific unit using its unique identifier. The association to the UnitConfig class has been modeled using a container element for a collection of unit ids, a reference to the Parameter that controls their settings, and the enumeration settings that select those units.

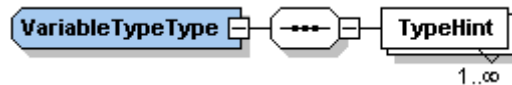
**Table 33 SEMI E125 RealType → FloatType Translation Table**

Attribute or Role Name	UML Name/Type	XML Element or Attribute	XML Name/Type
units	Unit	element	Units: esd:UnitRefType
units	UnitConfig	element	AvailableUnits: esd:UnitConfigArrayType (See ¶7.2.9.5.10.)
digitsOfPrecision	Integer	attribute	digitsOfPrecision: xsd:int

**7.2.9.5.15 DoubleTypeType** — This global complexType is an XML Schema representation of the SEMI E125 RealType class. The following table describes the mapping of the UML class to this XML Schema complexType. The association from RealType to the Unit class in SEMI E125 has been modeled using a reference attribute that refers to the specific unit using its unique identifier. The association to the UnitConfig class has been modeled using a container element for a collection of unit ids, a reference to the Parameter that controls their settings, and the enumeration settings that select those units.

**Table 34 E125 RealType → DoubleType Translation Table**

Attribute or Role Name	UML Name/Type	XML Element or Attribute	XML Name/Type
units	Unit	element	Units: esd:UnitRefType
units	UnitConfig	element	AvailableUnits: esd:UnitConfigArrayType (see ¶7.2.9.5.10)
digitsOfPrecision	Integer	attribute	digitsOfPrecision: xsd:int

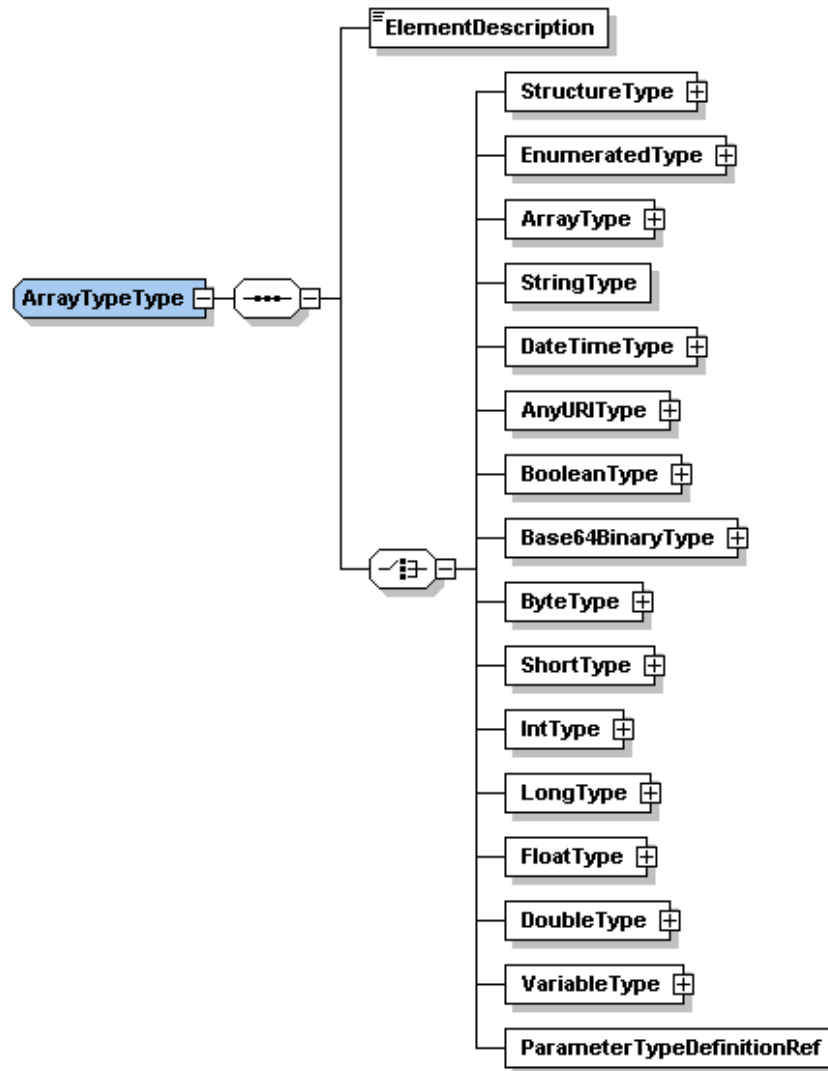


**Figure 11**  
**VariableTypeType Global complexType**

**7.2.9.5.16 VariableTypeType** — This global complexType is an extension to the SEMI E125 type description classes. VariableTypeType describes Parameters whose values may be of varying type at runtime. For example, a Parameter’s value may communicate an array of different control variable settings, each setting with a potentially different data type. The VariableType complexType can be used to describe such Parameters.

**7.2.9.5.16.1** The VariableTypeType provides a collection of elements named “TypeHint”, of type xsd:string. Each TypeHint shall be equal to the name of a ParameterTypeDefinition provided elsewhere in the equipment metadata. The collection of TypeHints together describes the set of possible types that a Parameter value may assume at runtime.

**7.2.9.5.17 DateTimeTypeType** — This global complexType is an extension to the SEMI E125 type description classes. DateTimeTypeType describes Parameters whose values are communicated using the XML Schema primitive type “xsd:dateTime”.



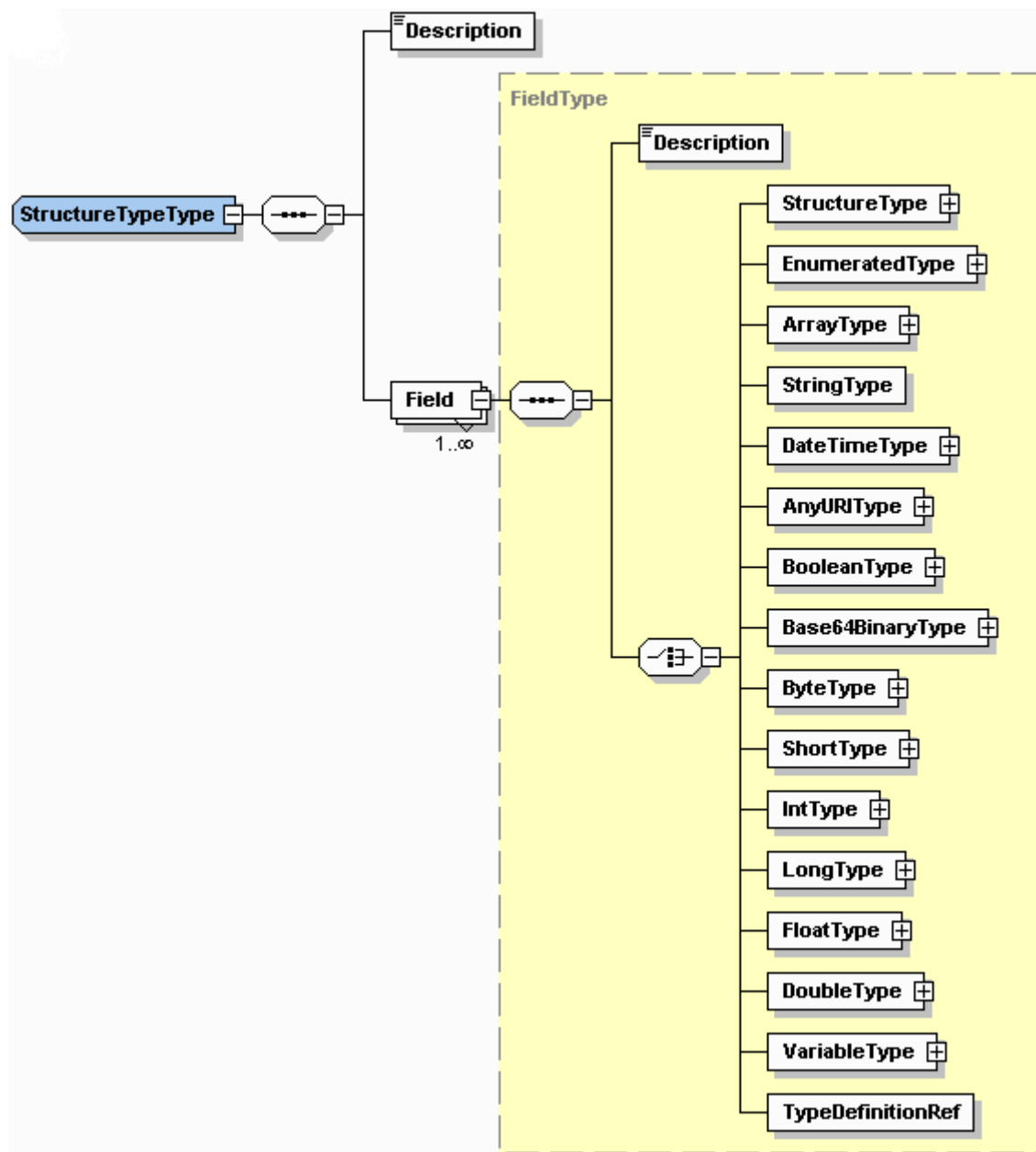
**Figure 12**  
**ArrayTypeType Global complexType**

7.2.9.5.18 *ArrayTypeType* — This global complexType is the XML Schema representation of the SEMI E125 ArrayType class. The type of each element in the array is defined using one of the complexTypes provided for the various primitive and composite types. The ArrayTypeType can describe arrays of arrays in a recursive fashion.

7.2.9.5.18.1 Values of Parameters that are described as arrays shall be communicated using a single container element representing the array as a whole, with each element of the array communicated using a nested element whose data type matches the type defined for the array.

**Table 35 SEMI E125 ArrayType → ArrayTypeType Translation Table**

<i>Attribute or Role Name</i>	<i>UML Name/Type</i>	<i>XML Element or Attribute</i>	<i>XML Name/Type</i>
minElements	integer	attribute	minElements: xsd:int
maxElements	integer	attribute	maxElements: xsd:int
elementDescription	text	element	ElementDescription: xsd:text
typeInformation	ParameterType	element	TypeInformation: esd:TypeInformationType
typeInformation	ParameterTypeDefinition	element (see above)	(see above)



**Figure 13**  
**StructureTypeType Global complexType**

7.2.9.5.19 *StructureTypeType* — This global complexType is the XML Schema representation of the SEMI E125 StructureType class. The type of each field in the structure is defined using the FieldType complexType, which incorporates the complexTypes provided for the various primitive and composite types. The StructureTypeType can describe nested structures.

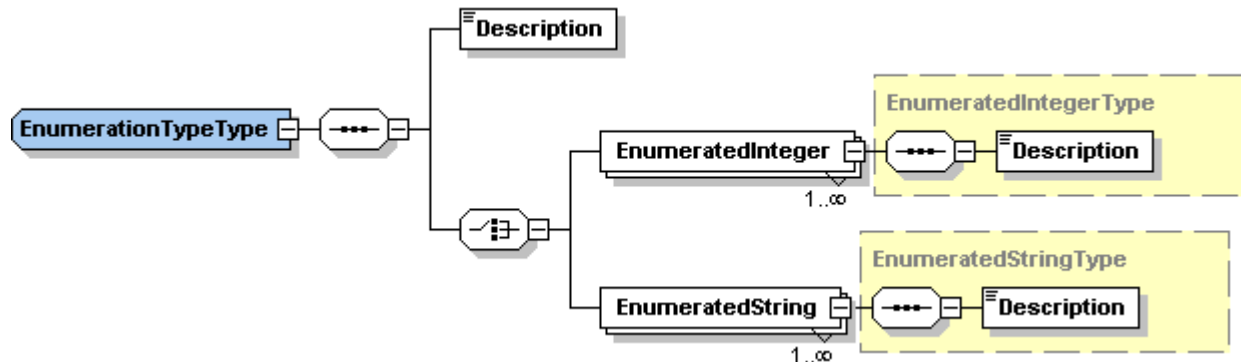
7.2.9.5.19.1 Values of Parameters that are described as structures shall be communicated using a single container element representing the structure as a whole, with each field of the structure communicated using a nested element whose data type matches the type defined for the described field.

**Table 36 SEMI E125 StructureType → StructureTypeType Translation Table**

Attribute or Role Name	UML Name/Type	XML Element or Attribute	XML Name/Type
description	text	element	Description: xsd:string
fields	Ordered list of one or more structured data elements of type Field	element	Fields: esd:FieldArrayType

**Table 37 SEMI E125 Field → FieldType Translation Table**

Attribute or Role Name	UML Name/Type	XML Element or Attribute	XML Name/Type
name	text	attribute	name: xsd:string
description	text	element	Description: xsd:string
canBeNull	boolean	attribute	canBeNull: xsd:boolean
typeInformation	ParameterType	element	TypeInformation: esd:TypeInformationType
typeInformation	ParamterTypeDefinition	element (see above)	(see above)



**Figure 14**  
**EnumeratedTypeType Global complexType**

7.2.9.5.20 *EnumerationTypeType* — This global complexType is the XML Schema representation of the SEMI E125 EnumerationType class.

7.2.9.5.20.1 Values of Parameters that are described as enumerations shall be communicated using a single container element whose data type is either xsd:string or xsd:int, according to whether the Parameter is string- or integer-based. Values of string-based enumerations are communicated using the English language character set.

**Table 38 SEMI E125 EnumerationType → EnumerationTypeType Translation Table**

Attribute or Role Name	UML Name/Type	XML Element or Attribute	XML Name/Type
description	text	element	Description: xsd:string
values	ordered list of one or more elements of type EnumeratedInteger	element	EnumeratedInteger: esd:EnumeratedIntegerType
values	ordered list of one or more elements of type EnumeratedString	element	EnumeratedString: esd:EnumeratedStringType

**Table 39 SEMI E125 EnumeratedString → EnumeratedStringType Translation Table**

Attribute or Role Name	UML Name/Type	XML Element or Attribute	XML Name/Type
description	text	element	Description: xsd:string
value	text	attribute	value: xsd:string

**Table 40 SEMI E125 EnumeratedInteger → EnumeratedIntegerType Translation Table**

Attribute or Role Name	UML Name/Type	XML Element or Attribute	XML Name/Type
description	text	element	Description: xsd:string
value	integer	attribute	value: xsd:int

### 7.2.10 GetStateMachines

7.2.10.1 This section is provisional, pending approval of SEMI E132.1 and SEMI E138.

#### 7.2.10.2 WSDL Operation Binding

7.2.10.2.1 See the operation binding for “GetStateMachines” in E125-1-V0305-EqpMetadataMgr-Binding.wsdl. Key information is shown for convenience in the following table.

**Table 41 GetStateMachines Operation Binding**

<i>SOAPAction</i>	urn:semi-org:ws.E125-1.V0305.esdMetaEqp-binding:GetStateMachines
<i>Input Headers (WSDL Message, Required)</i>	E132HeaderMessage, required
<i>Output Headers (WSDL Message, Required)</i>	E132HeaderMessage, required

#### 7.2.10.3 WSDL PortType Operation

7.2.10.3.1 See the portType operation definition for “GetStateMachines” in E125-1-V0305-EqpMetadataMgr-PortType.wsdl. Key information is shown for convenience in the following table.

**Table 42 GetStateMachines PortType Operation**

<i>Input Message Name</i>	GetStateMachinesRequestMessage
<i>Output Message Name</i>	GetStateMachinesResponseMessage

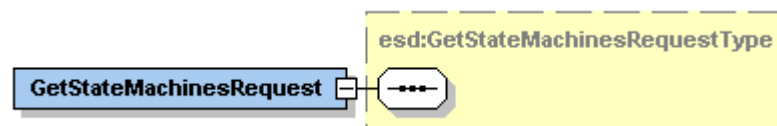
#### 7.2.10.4 WSDL Message(s)

7.2.10.4.1 See the message definitions for “E132HeaderMessage”, “GetStateMachinesRequestMessage”, and “GetStateMachinesResponseMessage” in E125-1-V0305-EqpMetadataMgr-PortType.wsdl. Key information is shown for convenience in the following table.

**Table 43 EquipmentMetadataManager Binding**

<i>Message Name → Schema Element Name</i>	E132HeaderMessage → auth:E132Header GetStateMachinesRequestMessage → esd:GetStateMachinesRequest GetStateMachinesResponseMessage → esd:GetStateMachinesResponse
---	---

#### 7.2.10.5 XML Schema Types



**Figure 15**  
**GetStateMachinesRequest Global Element**

7.2.10.5.1 *GetStateMachinesRequest* — This global element acts as a top-level container entity representing the SOAP body of a GetStateMachines request. As defined in SEMI E125, this operation has no input arguments, so the element has no contents.

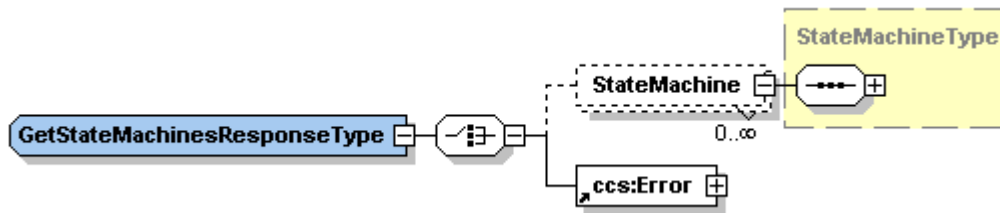


Figure 16  
GetStateMachinesResponse Global Element

7.2.10.5.2 *GetStateMachinesResponse* — This global element acts as a top-level container entity representing the SOAP body of the response to an SEMI E125 GetStateMachines request. It contains either zero or more elements named “StateMachine” of complexType “StateMachineType” or an element named “Error” of complexType “ErrorType”, defined in SEMI E138.

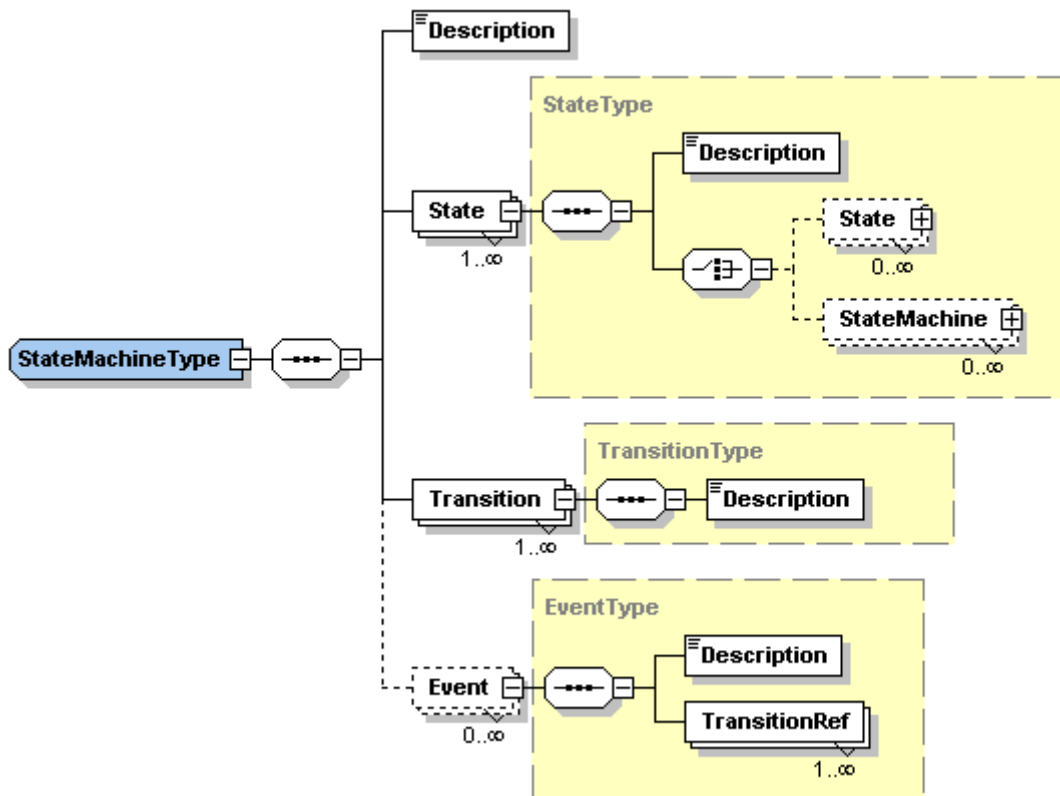


Figure 17  
StateMachineType Global complexType

**Table 44 Translation Table for Output Arguments for the GetStateMachines Operation**

<i>Argument Name</i>	<i>Format</i>	<i>XML Element or Attribute</i>	<i>XML Name/Type</i>
stateMachines	Unordered list of elements of type StateMachine	Element	StateMachine: StateMachineType (zero or more)
error	Structured data, of type Error, defined in SEMI E138	Element	Error: ccs:ErrorType

7.2.10.5.3 *StateMachineType* — This global complexType is the XML Schema representation of the SEMI E125 StateMachine class.

**Table 45 SEMI E125 StateMachine → StateMachineType Translation Table**

<i>Attribute or Role Name</i>	<i>UML Name/Type</i>	<i>XML Element or Attribute</i>	<i>XML Name/Type</i>
name	text	attribute	name: xsd:string
id	text	attribute	id: xsd:string
description	text	element	Description: xsd:string
top	One and only one element of type State	element	State: esd:StateType
transitions	List of one or more elements of type Transition	element	Transition: esd:TransitionType (one or more)
events	List of zero or more elements of type Event	element	Event: esd:EventType (zero or more)

7.2.10.5.4 *StateType* — This global complexType is the XML Schema representation of the SEMI E125 State class. The XOR relationship between the “substates” and “stateMachines” associations is modeled using a choice compositor.

**Table 46 SEMI E125 State → StateType Translation Table**

<i>Attribute or Role Name</i>	<i>UML Name/Type</i>	<i>XML Element or Attribute</i>	<i>XML Name/Type</i>
name	text	attribute	name: xsd:string
id	text	attribute	id: xsd:string
description	text	element	Description: xsd:string
substates	List of zero or more elements of type State	element	State: esd:StateType (zero or more)
stateMachines	List of zero or more elements of type StateMachine	element	StateMachine: esd:StateMachineType (zero or more)

7.2.10.5.5 *TransitionType* — This global complexType is the XML Schema representation of the SEMI E125 Transition class.

**Table 47 SEMI E125 Transition → TransitionType Translation Table**

<i>Attribute or Role Name</i>	<i>UML Name/Type</i>	<i>XML Element or Attribute</i>	<i>XML Name/Type</i>
name	text	attribute	name: xsd:string
id	text	attribute	id: xsd:string
description	text	element	Description: xsd:string
source	References one and only one state within this StateMachine	attribute	sourceStateId: xsd:string Must be equal to the “id” attribute of a State defined in this StateMachine.

<i>Attribute or Role Name</i>	<i>UML Name/Type</i>	<i>XML Element or Attribute</i>	<i>XML Name/Type</i>
target	References one and only one state within this StateMachine	attribute	targetStateId: xsd:string Must be equal to the “id” attribute of a State defined in this StateMachine.

7.2.10.5.6 *EventType* — This global complexType is the XML Schema representation of the SEMI E125 Event class. References to transitions are modeled using a collection of elements named “TransitionRef”, each of which has a single attribute named “transitioned”. The value of each “transitioned” attribute must be equal to the “id” attribute of a transition defined within the current StateMachine description.

**Table 48 SEMI E125 Event → EventType Translation Table**

<i>Attribute or Role Name</i>	<i>UML Name/Type</i>	<i>XML Element or Attribute</i>	<i>XML Name/Type</i>
name	text	attribute	name: xsd:string
id	text	attribute	id: xsd:string
description	text	element	Description: xsd:string
transitions	One or more elements of type Transition	element	TransitionRef: esd:TransitionRefType (one or more)

### 7.2.11 *GetSEMIObjTypes*

7.2.11.1 This section is provisional, pending approval of SEMI E132.1 and SEMI E138.

#### 7.2.11.2 *WSDL Operation Binding*

7.2.11.2.1 See the operation binding for “GetSEMIObjTypes” in E125-1-V0305-EqpMetadataMgr-Binding.wsdl. Key information is shown for convenience in the following table.

**Table 49 GetSEMIObjTypes Operation Binding**

<i>SOAPAction</i>	urn:semi-org:ws.E125-1.V0305.esdMetaEqp-binding:GetSEMIObjTypes
<i>Input Headers (WSDL Message, Required)</i>	E132HeaderMessage, required
<i>Output Headers (WSDL Message, Required)</i>	E132HeaderMessage, required

#### 7.2.11.3 *WSDL PortType Operation*

7.2.11.3.1 See the portType operation definition for “GetSEMIObjTypes” in E125-1-V0305-EqpMetadataMgr-PortType.wsdl. Key information is shown for convenience in the following table.

**Table 50 GetSEMIObjTypes PortType Operation**

<i>Input Message Name</i>	GetSEMIObjTypesRequestMessage
<i>Output Message Name</i>	GetSEMIObjTypesResponseMessage

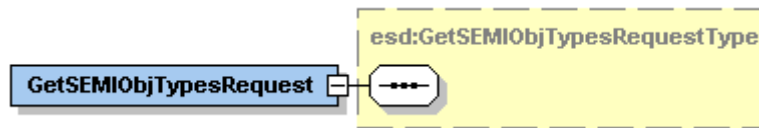
#### 7.2.11.4 *WSDL Message(s)*

7.2.11.4.1 See the message definitions for “E132HeaderMessage”, “GetSEMIObjTypesRequestMessage”, and “GetSEMIObjTypesResponseMessage” in E125-1-V0305-EqpMetadataMgr-PortType.wsdl. Key information is shown for convenience in the following table.

**Table 51 EquipmentMetadataManager Binding**

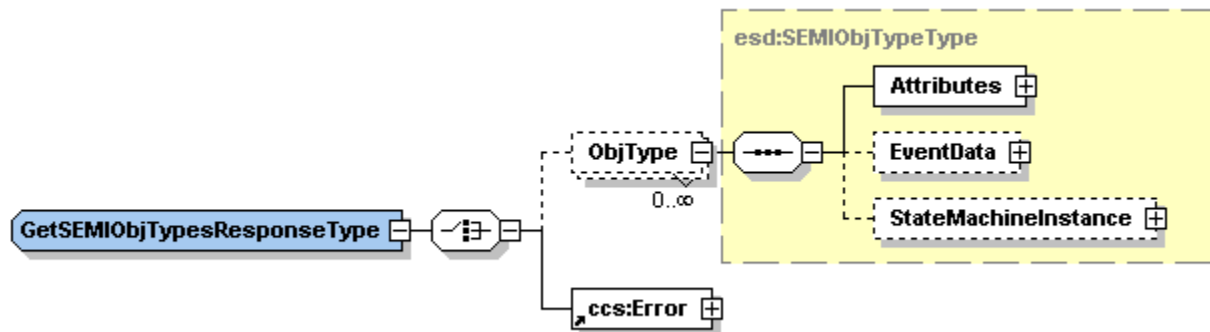
<i>Message Name → Schema Element Name</i>	E132HeaderMessage → auth:E132Header GetSEMIObjTypesResponseMessage → esd:GetSEMIObjTypesResponse GetSEMIObjTypesResponseMessage → esd:GetSEMIObjTypesResponse
---	---

## 7.2.11.5 XML Schema Types



**Figure 18**  
**GetSEMIObjTypesRequest Global Element**

**7.2.11.5.1 GetSEMIObjTypesRequest** — This global element acts as a top-level container entity representing the SOAP body of a GetSEMIObjTypes request. As defined in SEMI E125, this operation has no input arguments, so the element has no contents.



**Figure 19**  
**GetSEMIObjTypesResponse Global Element**

**7.2.11.5.2 GetSEMIObjTypesResponse** — This global element acts as a top-level container entity representing the SOAP body of the response to an SEMI E125 GetSEMIObjTypes request. It contains either zero or more elements named “ObjType” of complexType “SEMIObjTypeType” or an element named “Error” of complexType “ErrorType”, defined in SEMI E138.

**Table 52 Translation Table for Output Arguments for the GetSEMIObjTypes Operation**

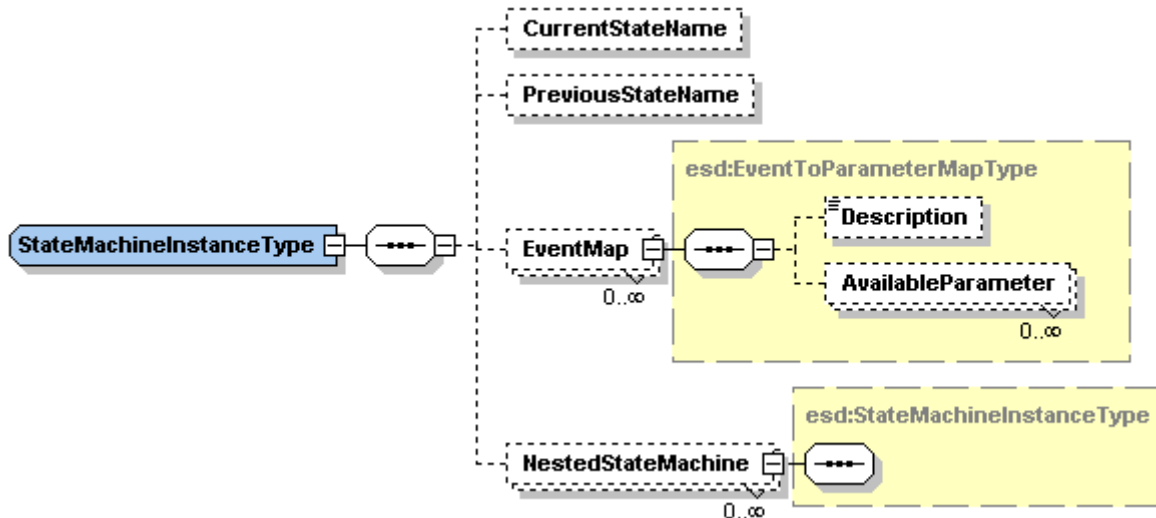
Argument Name	Format	XML Element or Attribute	XML Name/Type
objTypes	Unordered list of elements of type SEMIObjType	Element	ObjTypes: SEMIObjTypeType (zero or more)
error	Structured data, of type Error, defined in SEMI E138	Element	Error: ccs:ErrorType  No SEMI E125-specific error codes are defined for this operation.

**7.2.11.5.3 SEMIObjTypeType** — This global complexType is the XML Schema representation of the SEMI E125 SEMIObjType class. Attributes and eventData are provided using the ParameterType, defined in ¶7.2.14.5.4.

**Table 53 SEMI E125 SEMIObjType → SEMIObjTypeType Translation Table**

Attribute or Role Name	UML Name/Type	XML Element or Attribute	XML Name/Type
standardVersion	text	attribute	standardVersion: esd:SEMISTandardVersionType
objType	text	attribute	id: xsd:string
attributes	Parameter	element	Attributes: esd:ParameterCollectionType (see ¶7.2.14.5.4 for description of the ParameterType complexType)

Attribute or Role Name	UML Name/Type	XML Element or Attribute	XML Name/Type
eventData	Parameter	element	EventData: esd:ParameterCollectionType (see ¶7.2.14.5.4 for description of the ParameterType complexType)
stateMachine	StateMachineInstance	element	StateMachine: esd:StateMachineInstanceType



**Figure 20**  
**StateMachineInstanceType Global complexType**

7.2.11.5.4 *StateMachineInstanceType* — This global complexType is the XML Schema representation of the SEMI E125 StateMachineInstance class.

**Table 54 SEMI E125 StateMachineInstance → StateMachineInstanceType Translation Table**

Attribute or Role Name	UML Name/Type	XML Element or Attribute	XML Name/Type
stateMachineId	text	attribute	stateMachineId: xsd:string
currentStateName	Parameter	element	CurrentStateName: esd:ParameterReferenceType
previousStateName	Parameter	element	PreviousStateName: esd:ParameterReferenceType
eventMaps	EventToParameterMap	element	EventMap: esd:EventToParameterMapType (zero or more)
nestedStateMachines	StateMachineInstance	element	NestedStateMachine: esd:StateMachineInstanceType (zero or more)

7.2.11.5.5 *ParameterReferenceType* — This global complexType is the XML Schema representation of a reference to a Parameter defined in the equipment metadata. Referenceable Parameters are defined either by SEMIObjTypes as “Attributes” or “EventData”, or by equipment nodes in the set of Parameters provided with “EquipmentNodeDescription” metadata. Equipment nodes and SEMIObjTypes are said to be valid Parameter “sources”. Parameters defined for Exceptions are not referenceable. This type defines two attributes, “source” and “name” that are used to uniquely identify Parameters in equipment metadata.

7.2.11.5.5.1 Parameter references may be “absolute” or “relative”. An absolute Parameter reference is used when referring to a Parameter that is defined by a *different* source than the one making the reference. A relative reference

is used when referring to a Parameter that is defined by the *same* source as the one making the reference. It is expected that the most common Parameter references in equipment metadata will be relative references.

7.2.11.5.5.1.1 For an example of an “absolute” reference, consider two equipment nodes: a “TemperatureController” equipment node that provides a “TemperatureSetpoint” Parameter; and a “TemperatureSensor” equipment node that provides a “MeasuredTemperature” Parameter. The “TemperatureSetpoint” Parameter is a control Parameter whose setting affects the “MeasuredTemperature” Parameter, so it may need to refer to that Parameter from the context of the “TemperatureController” node. In this case, an “absolute” reference is required, since the “referred-to” Parameter is defined by a different node than the node making the reference.

7.2.11.5.5.1.2 For an example of a “relative” reference, consider a SEMIObjType that has a SEMI E39 attribute defined for it that stores the current state of the object. In the StateMachineInstance for that same SEMIObjType, the “CurrentStateName” element is used to refer to a Parameter that can be used to determine the current state of that StateMachine. In this case, the Parameter belongs to the same “source” (the same SEMIObjType) that implements the StateMachine, so a relative reference can be used.

7.2.11.5.5.2 *source* — This attribute is optional and shall be either not present (relative reference) or present and set to a valid CEM Locator (absolute reference) identifying the equipment node that provides the referenced Parameter. If not present, the source is inferred by the context that the reference is made (either a SEMIObjType or an equipment node).

7.2.11.5.5.3 *name* — For relative references to Parameters provided directly from an equipment node or SEMIObjType, this attribute shall be set to the value of the “name” attribute of the Parameter that is referred to. For absolute references to Parameters provided directly from an equipment node, this attribute shall be set to the value of the “name” attribute of the corresponding Parameter. For absolute references to SEMIObjType attributes or event data Parameters provided from a specific equipment node, this shall be a string of the form “urn:semi-org:objType:<typeName>:objAttr:<parameter name>”, where “<typeName>” is the ObjType name, and “<parameter name>” is the name of the Parameter representing the attribute or event data.

7.2.11.5.5.4 Some examples of valid Parameter references are shown in the following table.

**Table 55 Example Parameter References**

<i>Absolute / Relative</i>	<i>Referring Source</i>	<i>Referenced Source</i>	<i>Referenced Parameter Name</i>
relative	SEMIObjType definition	“”	“PRJobState”
relative	Equipment Node	“”	“MeasuredTemperature”
absolute	Equipment Node	“Equipment/Oven/RTD-1”	“OvenTemp”
absolute	Equipment Node	“Equipment”	“urn:semi-org:objType:ProcessJob:objAttr:PRJobState”

7.2.11.5.6 *EventToParameterMapType* — This global complexType is the XML Schema representation of the SEMI E125 EventToParameterMap class.

**Table 56 SEMI E125 EventToParameterMap → EventToParameterMapType Translation Table**

<i>Attribute or Role Name</i>	<i>UML Name/Type</i>	<i>XML Element or Attribute</i>	<i>XML Name/Type</i>
eventId	text	attribute	eventId: xsd:string
description	text	element	Description: xsd:string
availableParameters	Structured data of type Parameter	element	AvailableParameter: esd:ParameterReferenceType (zero or more, see ¶7.2.11.5.5 for description of Parameter references)

## 7.2.12 *GetExceptions*

7.2.12.1 This section is provisional, pending approval of SEMI E132.1 and SEMI E138.

### 7.2.12.2 *WSDL Operation Binding*

7.2.12.2.1 See the operation binding for “GetExceptions” in E125-1-V0305-EqpMetadataMgr-Binding.wsdl. Key information is shown for convenience in the following table.

**Table 57 GetExceptions Operation Binding**

<i>SOAPAction</i>	urn:semi-org:ws.E125-1.V0305.esdMetaEqp-binding:GetExceptions
<i>Input Headers (WSDL Message, Required)</i>	E132HeaderMessage, required
<i>Output Headers (WSDL Message, Required)</i>	E132HeaderMessage, required

### 7.2.12.3 *WSDL PortType Operation*

7.2.12.3.1 See the portType operation definition for “GetExceptions” in E125-1-V0305-EqpMetadataMgr-PortType.wsdl. Key information is shown for convenience in the following table.

**Table 58 GetExceptions PortType Operation**

<i>Input Message Name</i>	GetExceptionsRequestMessage
<i>Output Message Name</i>	GetExceptionsResponseMessage

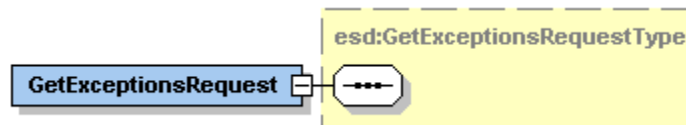
### 7.2.12.4 *WSDL Message(s)*

7.2.12.4.1 See the message definitions for “E132HeaderMessage”, “GetSEMIObjTypesRequestMessage”, and “GetSEMIObjTypesResponseMessage” in E125-1-V0305-EqpMetadataMgr-PortType.wsdl. Key information is shown for convenience in the following table.

**Table 59 EquipmentMetadataManager Binding**

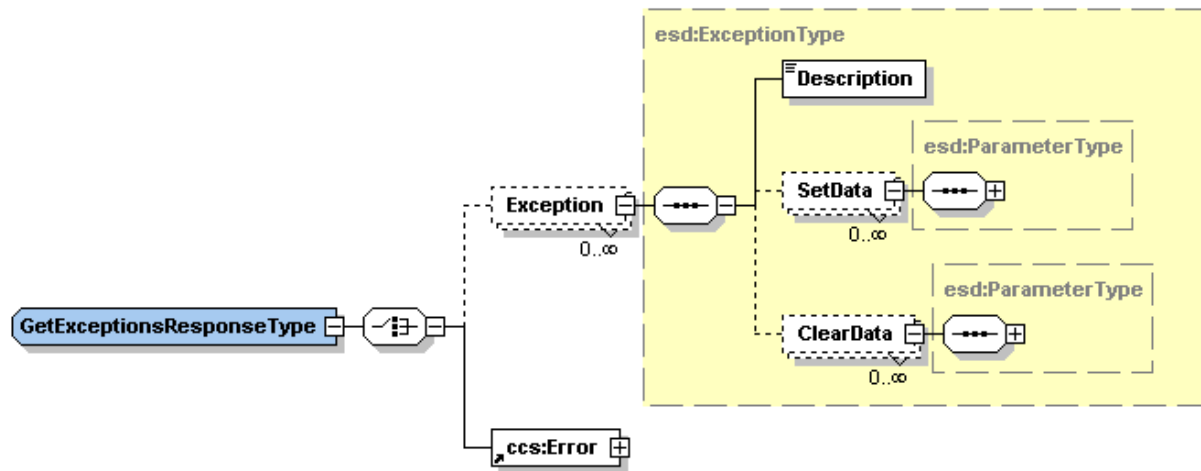
<i>Message Name → Schema Element Name</i>	E132HeaderMessage → auth:E132Header GetExceptionsRequestMessage → esd:GetExceptionsRequest GetExceptionsResponseMessage → esd:GetExceptionsResponse
---	---

### 7.2.12.5 *XML Schema Types*



**Figure 21**  
**GetExceptionsRequest Global Element**

7.2.12.5.1 *GetExceptionsRequest* — This global element acts as a top-level container entity representing the SOAP body of a GetExceptions request. As defined in SEMI E125, this operation has no input arguments, so the element has no contents.



**Figure 22**  
**GetExceptionsResponse Global Element**

**7.2.12.5.2 GetExceptionsResponse** — This global element acts as a top-level container entity representing the SOAP body of the response to an SEMI E125 GetExceptions request. It contains either zero or more elements named “Exception” of complexType “ExceptionType” or an element named “Error” of complexType “ErrorType”, defined in SEMI E138.

**Table 60 Translation Table for Output Arguments for the GetExceptions Operation**

Argument Name	Format	XML Element or Attribute	XML Name/Type
exceptions	Unordered list of elements of type Exception	Element	Exception: esd:ExceptionType (zero or more)
error	Structured data, of type Error, defined in SEMI E138	Element	Error: ccs:ErrorType  No SEMI E125-specific error codes are defined for this operation.

**7.2.12.5.3 ExceptionType** — This global complexType is the XML Schema representation of the SEMI E125 Exception class. The “setData” and “clearData” associations to Parameters in SEMI E125 are modeled using zero or more elements of type “ParameterType” (see ¶7.2.14.5.4).

**Table 61 SEMI E125 Exception → ExceptionType Translation Table**

Attribute or Role Name	UML Name/Type	XML Element or Attribute	XML Name/Type
id	text	attribute	id: xsd:string
description	text	element	Description: xsd:string
severity	text	attribute	severity: xsd:string
setData	Ordered list of zero or more elements of type Parameter	element	SetData: esd:ParameterType (Zero or more, see ¶7.2.14.5.4 for a description of the ParameterType complexType.)
clearData	Ordered list of zero or more elements of type Parameter	element	ClearData: esd:ParameterType (zero or more, see ¶7.2.14.5.4 for a description of the ParameterType complexType)

### 7.2.13 GetEquipmentStructure

7.2.13.1 This section is provisional, pending approval of SEMI E132.1, SEMI E138, and SEMI E120.1.

#### 7.2.13.2 WSDL Operation Binding

7.2.13.2.1 See the operation binding for “GetEquipmentStructure” in E125-1-V0305-EqpMetadataMgr-Binding.wsdl. Key information is shown for convenience in the following table.

**Table 62 GetEquipmentStructure Operation Binding**

<i>SOAPAction</i>	urn:semi-org:ws.E125-1.V0305.esdMetaEqp-binding:GetEquipmentStructure
<i>Input Headers (WSDL Message, Required)</i>	E132HeaderMessage, required
<i>Output Headers (WSDL Message, Required)</i>	E132HeaderMessage, required

#### 7.2.13.3 WSDL PortType Operation

7.2.13.3.1 See the portType operation definition for “GetEquipmentStructure” in E125-1-V0305-EqpMetadataMgr-PortType.wsdl. Key information is shown for convenience in the following table.

**Table 63 GetEquipmentStructure PortType Operation**

<i>Input Message Name</i>	GetEquipmentStructureRequestMessage
<i>Output Message Name</i>	GetEquipmentStructureResponseMessage

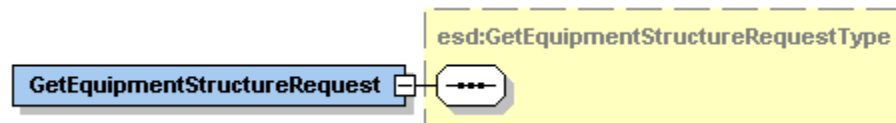
#### 7.2.13.4 WSDL Message(s)

7.2.13.4.1 See the message definitions for “E132HeaderMessage”, “GetEquipmentStructureRequestMessage”, and “GetEquipmentStructureResponseMessage” in E125-1-V0305-EqpMetadataMgr-PortType.wsdl. Key information is shown for convenience in the following table.

**Table 64 EquipmentMetadataManager Binding**

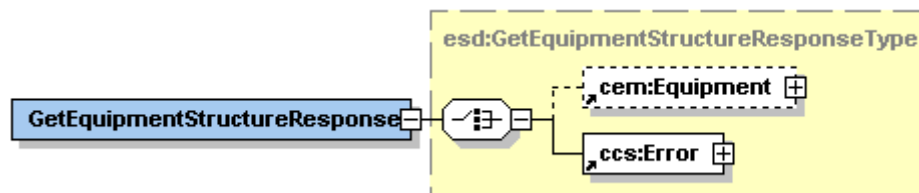
<i>Message Name → Schema Element Name</i>	E132HeaderMessage → auth:E132Header GetEquipmentStructureRequestMessage → esd: GetEquipmentStructureRequest GetEquipmentStructureResponseMessage → esd: GetEquipmentStructureResponse
---	---

#### 7.2.13.5 XML Schema Types



**Figure 23**  
**GetEquipmentStructureRequest Global Element**

7.2.13.5.1 *GetEquipmentStructureRequest* — This global element acts as a top-level container entity representing the SOAP body of a GetEquipmentStructure request. As defined in SEMI E125, this operation has no input arguments, so the element has no contents.



**Figure 24**  
**GetEquipmentStructureResponse Global Element**

7.2.13.5.2 *GetEquipmentStructureResponse* — This global element acts as a top-level container entity representing the SOAP body of the response to an E125 GetEquipmentStructure request. It contains either an element named “Equipment” of complexType “EquipmentType” as defined in SEMI E120.1, or an element named “Error” of complexType “ErrorType”, defined in SEMI E138.

**Table 65 Translation Table for Output Arguments for the GetEquipmentStructure Operation**

Argument Name	Format	XML Element or Attribute	XML Name/Type
equipmentStructure	Structured data, of a type derived from the E120 type EquipmentElement, that represents the equipment as a whole	element	Equipment: cem:EquipmentType
error	Structured data, of type Error, defined in SEMI E138	element	Error: ccs:ErrorType  No SEMI E125-specific error codes are defined for this operation.

7.2.13.5.3 *Equipment* — This global element is defined in SEMI E120.1. Please refer to that specification for a description of the contents of this element.

#### 7.2.14 *GetEquipmentNodeDescriptions*

7.2.14.1 This section is provisional, pending approval of SEMI E132.1 and SEMI E138.

#### 7.2.14.2 *WSDL Operation Binding*

7.2.14.2.1 See the operation binding for “GetEquipmentNodeDescriptions” in E125-1-V0305-EqpMetadataMgr-Binding.wsdl. Key information is shown for convenience in the following table.

**Table 66 GetEquipmentNodeDescriptions Operation Binding**

<i>SOAPAction</i>	urn:semi-org:ws.E125-1.V0305.esdMetaEqp-binding:GetEquipmentNodeDescriptions
<i>Input Headers (WSDL Message, Required)</i>	E132HeaderMessage, required
<i>Output Headers (WSDL Message, Required)</i>	E132HeaderMessage, required

#### 7.2.14.3 *WSDL PortType Operation*

7.2.14.3.1 See the portType operation definition for “GetEquipmentNodeDescriptions” in E125-1-V0305-EqpMetadataMgr-PortType.wsdl. Key information is shown for convenience in the following table.

**Table 67 GetEquipmentNodeDescriptions PortType Operation**

<i>Input Message Name</i>	GetEquipmentNodeDescriptionsRequestMessage
<i>Output Message Name</i>	GetEquipmentNodeDescriptionsResponseMessage

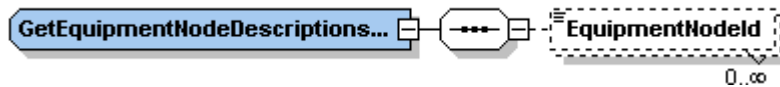
#### 7.2.14.4 *WSDL Message(s)*

7.2.14.4.1 See the message definitions for “E132HeaderMessage”, “GetEquipmentNodeDescriptionsRequestMessage”, and “GetEquipmentNodeDescriptionsResponseMessage” in E125-1-V0305-EqpMetadataMgr-PortType.wsdl. Key information is shown for convenience in the following table.

**Table 68 EquipmentMetadataManager Binding**

<i>Message Name → Schema Element Name</i>	E132HeaderMessage → auth:E132Header GetEquipmentNodeDescriptionsRequestMessage → esd:GetEquipmentNodeDescriptionsRequest GetEquipmentNodeDescriptionsResponseMessage → esd:GetEquipmentNodeDescriptionsResponse
---	---

#### 7.2.14.5 XML Schema Types

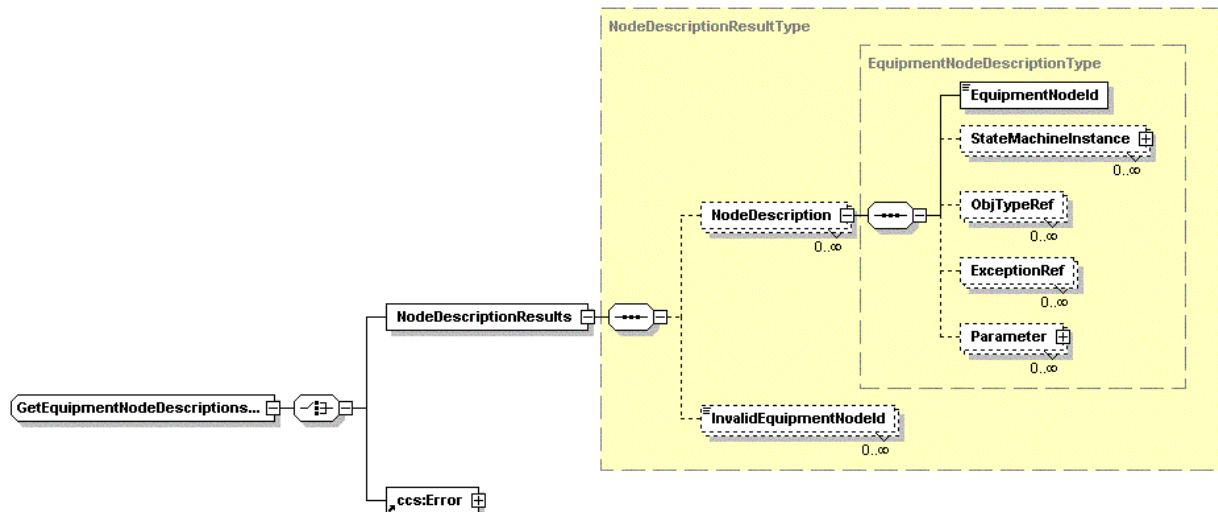


**Figure 25**  
**GetEquipmentNodeDescriptionsRequest Global Element**

**7.2.14.5.1 GetEquipmentNodeDescriptionsRequest** — This global element acts as a top-level container entity representing the SOAP body of a GetEquipmentNodeDescriptions request. As defined in SEMI E125, this operation has one input argument: a list of SEMI E120 Locator values. The list is modeled using zero or more elements named “EquipmentNodeId” of type “cem:LocatorType”, defined in SEMI E120.1.

**Table 69 Translation Table for Input Arguments for the GetEquipmentNodeDescriptions Operation**

Argument Name	Format	XML Element or Attribute	XML Name/Type
equipmentNodeIds	List of text, each entry equal to a valid SEMI E120 Nameable “uid” attribute	optional element	EquipmentNodeId: cem:LocatorType



**Figure 26**  
**GetEquipmentNodeDescriptionsResponse Global Element**

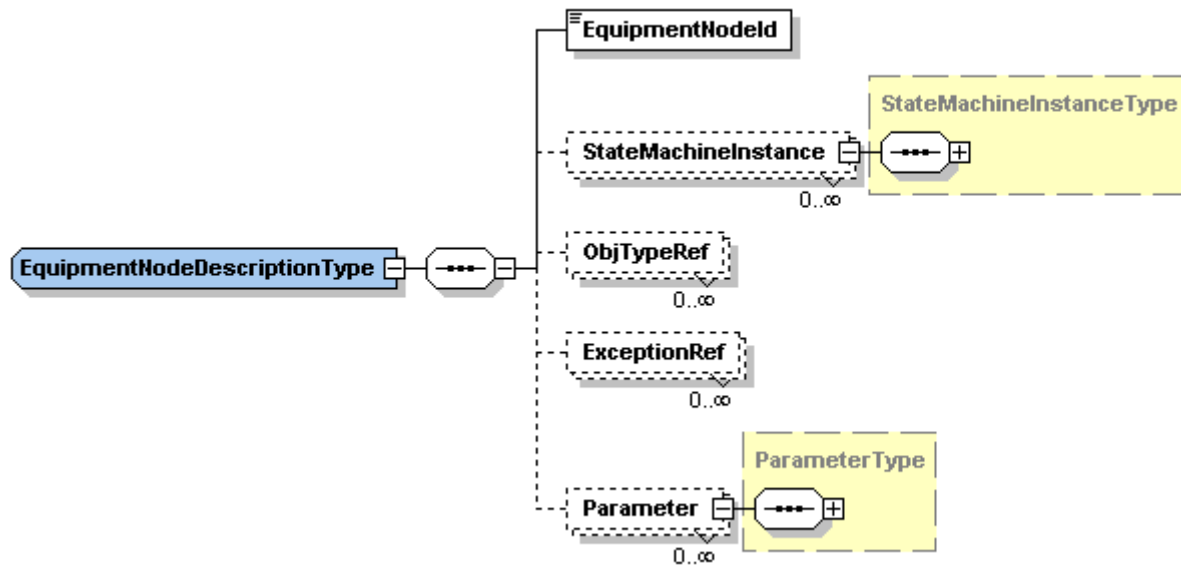
**7.2.14.5.2 GetEquipmentNodeDescriptionsResponse** — This global element acts as a top-level container element representing the SOAP body of the response to an SEMI E125 GetEquipmentNodeDescriptions request. It contains either an element named “NodeDescriptionResults” of complexType “NodeDescriptionResultType” or an element named “Error” of complexType “ErrorType”, defined in SEMI E138.

**7.2.14.5.2.1** As defined in SEMI E125, this operation returns an unordered list of instances of the SEMI E125 EquipmentNodeDescription class. The list is modeled as zero or more elements named “NodeDescription” of type “EquipmentNodeDescriptionType”.

**7.2.14.5.2.2** As defined in SEMI E125, the output for this operation can include an unordered list of strings to identify requests that include one or more invalid CEM Locators as input. If invalid Locator values are provided in the request, the equipment shall return one or more “InvalidEquipmentNodeId” elements, consisting of the list of all invalid CEM Locator’s provided in the original request.

**Table 70 Translation Table for Output Arguments for the GetEquipmentNodeDescriptions Operation**

Argument Name	Format	XML Element or Attribute	XML Name/Type
nodeDescriptions	Zero or more elements of type EquipmentNodeDescription	element	NodeDescription: esd:EquipmentNodeDescriptionType (zero or more)
error	List of text, each entry equal to an unrecognized equipment node identifier provided in the GetEquipmentNodeDescriptions request	element	InvalidEquipmentNodeId: esd:EquipmentNodeIdType (zero or more)



**Figure 27**  
**EquipmentNodeDescriptionType Global complexType**

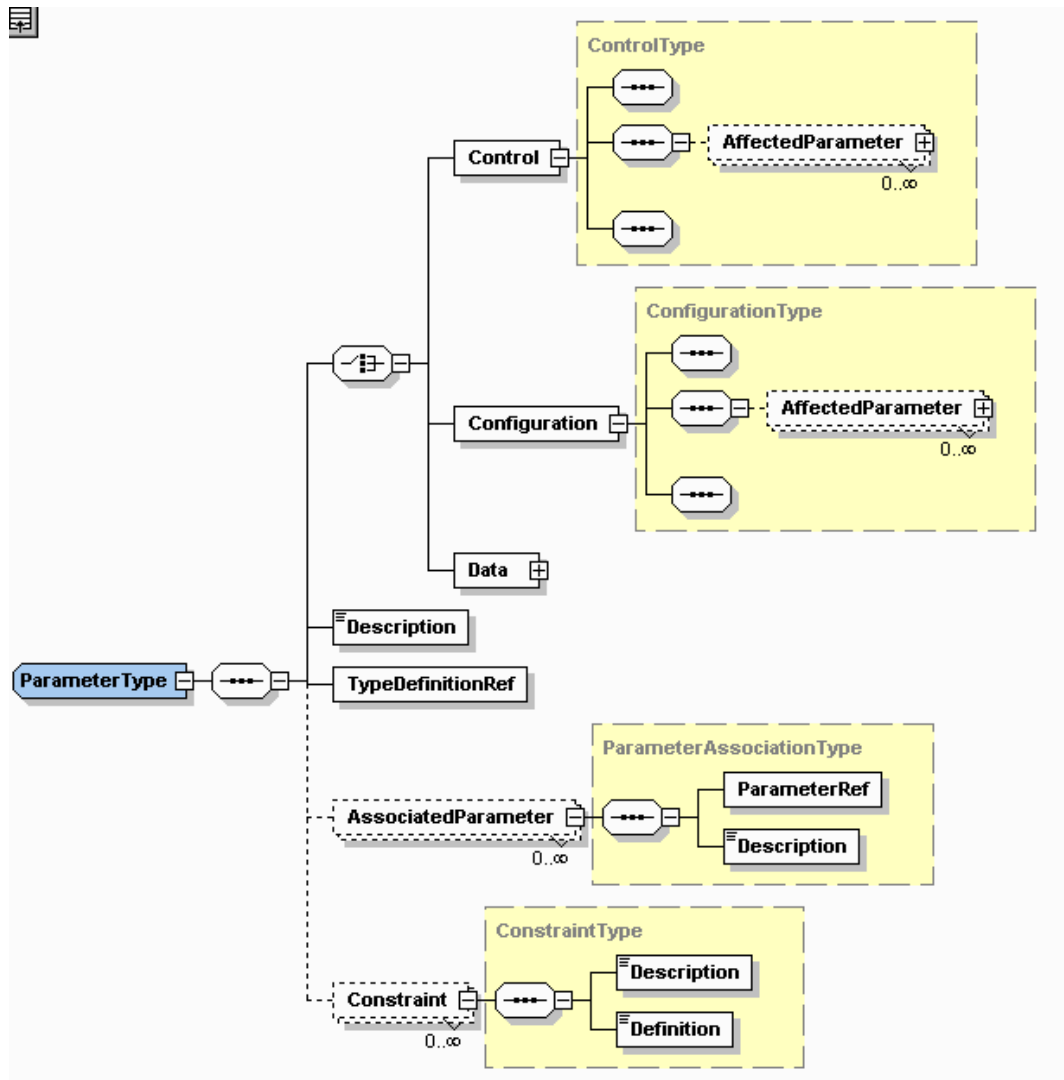
7.2.14.5.3 *EquipmentNodeDescriptionType* — This global complexType is the XML Schema representation of the SEMI E125 EquipmentNodeDescription class. The CEM equipment node being described is identified by a CEM Locator, via an element named “EquipmentNodeId” of type cem:LocatorType, defined in SEMI E120.1.

7.2.14.5.3.1 StateMachines supported by the equipment node are described using a collection of one or more “StateMachineInstance” elements of type “StateMachineInstanceType” (see ¶7.2.11.5.4).

7.2.14.5.3.2 ObjTypes provided by the equipment node are referenced using a collection of one or more “SEMIObjTypeRef” elements of type “SEMIObjTypeReferenceType”; these must refer to one or more valid SEMIObjTypes returned via the “GetSEMIObjTypes” operation.

7.2.14.5.3.3 Exceptions provided by the equipment node are referenced using a collection of one or more “ExceptionRef” elements of type “ExceptionReferenceType”; these must refer to one or more valid Exception id’s returned via the “GetExceptions” operation.

7.2.14.5.3.4 Parameters provided by the equipment node are described using a collection of one or more “Parameter” elements of type “ParameterType” (see ¶7.2.14.5.4).



**Figure 28**  
**ParameterType Global complexType**

**Table 71 SEMI E125 EquipmentNodeDescription → EquipmentNodeDescriptionType Translation Table**

Attribute or Role Name	UML Name/Type	XML Element or Attribute	XML Name/Type
node	Element of a type derived from the E120 Nameable class	element	EquipmentNodeId: cem:LocatorType
stateMachines	List of elements of type StateMachineInstance	element	StateMachineInstance: esd:StateMachineInstanceType (zero or more)
objTypes	List of elements of type SEMIObjType	element	ObjTypeRef: esd:SEMIObjTypeReferenceType (zero or more)
exceptions	List of elements of type Exception	element	ExceptionRef: esd:ExceptionReferenceType (zero or more)
parameters	List of elements of type Parameter	element	Parameter: esd:ParameterType (zero or more)

7.2.14.5.4 *ParameterType* — This global complexType is the XML Schema representation of the SEMI E125 Parameter class.

7.2.14.5.4.1 Type information for a Parameter is referenced using a “TypeDefinitionRef” element whose value must be equal to a valid ParameterTypeDefinition returned via the “GetTypeDefinitions” operation.

7.2.14.5.4.2 Parameter classification is modeled using a choice compositor to select one and only one of the three possible subclasses of the SEMI E125 ReadOnly and ReadWrite classes (Control, Configuration, or Data).

**Table 72 SEMI E125 Parameter → ParameterType Translation Table**

<i>Attribute or Role Name</i>	<i>UML Name/Type</i>	<i>XML Element or Attribute</i>	<i>XML Name/Type</i>
name	text	attribute	name: xsd:string
description	text	element	Description: xsd:string
constraints	List of structured data, of type Constraint	element	Constraint: esd:ConstraintType (zero or more, see ¶7.2.14.5.5 for mapping of the SEMI E125 Constraint class)
associatedParameters	List of structured data, of type AssociatedParameter	element	AssociatedParameter: esd:AssociatedParameterType (zero or more, see ¶7.2.14.5.6 for mapping of the SEMI E125 AssociatedParameter class).
typeInformation	One and only one element of type ParameterTypeDefinition	element	TypeDefinitionRef: esd:ParameterTypeDefinitionReferenceType
classification	One and only one element of any type derived from ParameterClassification	element	Control, Configuration, or Data: esd:ConfigurationType, ControlType, or DataType (one and only one, see ¶¶7.2.14.5.7 through 7.2.14.5.12 for a mapping of the SEMI E125 classification classes)

7.2.14.5.5 *ConstraintType* — This global complexType is the XML Schema representation of the SEMI E125 Constraint class.

**Table 73 SEMI E125 Constraint → ConstraintType Translation Table**

<i>Attribute or Role Name</i>	<i>UML Name/Type</i>	<i>XML Element or Attribute</i>	<i>XML Name/Type</i>
name	text	attribute	name: xsd:string
description	text	element	Description: xsd:string
definition	text	element	Definition: xsd:string

7.2.14.5.6 *ParameterAssociationType* – this global complexType is the XML Schema representation of the E125 AssociatedParameter class.

**Table 74 SEMI E125 AssociatedParameter → ParameterAssociationType Translation Table**

<i>Attribute or Role Name</i>	<i>UML Name/Type</i>	<i>XML Element or Attribute</i>	<i>XML Name/Type</i>
description	text	element	Description: xsd:string
parameter	Unordered list of structured data, of type Parameter	element	ParameterRef: esd:ParameterReferenceType (see ¶7.2.11.5.5)

7.2.14.5.7 *ParameterClassificationType* — This global complexType is the XML Schema representation of the SEMI E125 ParameterClassification class.

**Table 75 SEMI E125 ParameterClassification → ParameterClassificationType Translation Table**

<i>Attribute or Role Name</i>	<i>UML Name/Type</i>	<i>XML Element or Attribute</i>	<i>XML Name/Type</i>
isTransient	boolean	attribute	isTransient: xsd:boolean

7.2.14.5.8 *ReadWriteType* — This global complexType is the XML Schema representation of the SEMI E125 ReadWrite class. It is an extension of the ParameterClassification complexType.

**Table 76 SEMI E125 ReadWrite → ReadWriteType Translation Table**

<i>Attribute or Role Name</i>	<i>UML Name/Type</i>	<i>XML Element or Attribute</i>	<i>XML Name/Type</i>
affectedParameters	Unordered list of structured data, of type AssociatedParameter	element	AffectedParameters: esd:AssociatedParameterArrayType

7.2.14.5.9 *ReadOnlyType* — This global complexType is the XML Schema representation of the SEMI E125 ReadOnly class. It is an extension of the ParameterClassification complexType, and defines no additional attributes or elements.

7.2.14.5.10 *ControlType* — This global complexType is the XML Schema representation of the SEMI E125 Control class. It is an extension of the ReadWrite complexType, and defines no additional attributes or elements.

7.2.14.5.11 *ConfigurationType* — This global complexType is the XML Schema representation of the SEMI E125 Configuration class. It is an extension of the ReadWrite complexType, and defines no additional attributes or elements.

7.2.14.5.12 *DataType* — This global complexType is the XML Schema representation of the SEMI E125 Data class. It is an extension of the ReadOnly complexType, and defines no additional attributes or elements.

## 7.2.15 GetLatestRevision

7.2.15.1 This section is provisional, pending approval of SEMI E132.1 and the SEMI E138

### 7.2.15.2 WSDL Operation Binding

7.2.15.2.1 See the operation binding for “GetLatestRevision” in E125-1-V0305-EqpMetadataMgr-Binding.wsdl. Key information is shown for convenience in the following table.

**Table 77 GetLatestRevision Operation Binding**

<i>SOAPAction</i>	urn:semi-org:ws.E125-1.V0305.esdMetaEqp-binding:GetLatestRevision
<i>Input Headers (WSDL Message, Required)</i>	E132HeaderMessage, required
<i>Output Headers (WSDL Message, Required)</i>	E132HeaderMessage, required

### 7.2.15.3 WSDL PortType Operation

7.2.15.3.1 See the portType operation definition for “GetLatestRevision” in E125-1-V0305-EqpMetadataMgr-PortType.wsdl. Key information is shown for convenience in the following table.

**Table 78 GetLatestRevision PortType Operation**

<i>Input Message Name</i>	GetLatestRevisionRequestMessage
<i>Output Message Name</i>	GetLatestRevisionResponseMessage

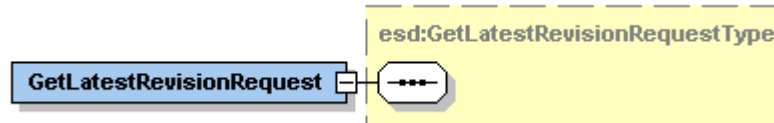
#### 7.2.15.4 WSDL Message(s)

7.2.15.4.1 See the message definitions for “E132HeaderMessage”, “GetLatestRevisionRequestMessage”, and “GetLatestRevisionResponseMessage” in E125-1-V0305-EqpMetadataMgr-PortType.wsdl. Key information is shown for convenience in the following table.

**Table 79 EquipmentMetadataManager Binding**

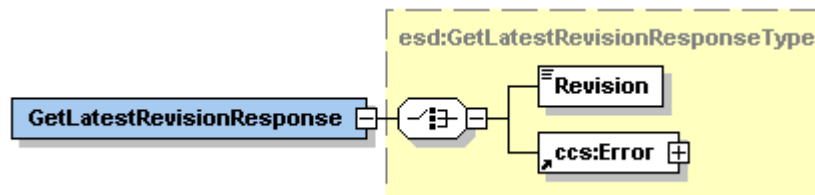
<i>Message Name → Schema Element Name</i>	E132HeaderMessage → auth:E132Header GetLatestRevisionRequestMessage → esd:GetLatestRevisionRequest GetLatestRevisionResponseMessage → esd:GetLatestRevisionResponse
---	---

#### 7.2.15.5 XML Schema Types



**Figure 29**  
**GetLatestRevisionRequest**

7.2.15.5.1 *GetLatestRevisionRequest* — This global element acts as a top-level container entity representing the SOAP body of a GetLatestRevision request. As defined in SEMI E125, this operation has no input arguments, so the element has no contents.



**Figure 30**  
**GetLatestRevisionResponse Global Element**

7.2.15.5.2 *GetLatestRevisionResponse* — This global element acts as a top-level container entity representing the SOAP body of the response to an SEMI E125 GetLatestRevision request. It contains either an element named “Revision” of built-in type “xsd:dateTime” or an element named “Error” of complexType “ErrorType”, defined in SEMI E138.

**Table 80 Translation Table for Output Arguments for the GetLatestRevision Operation**

<i>Argument Name</i>	<i>Format</i>	<i>XML Element or Attribute</i>	<i>XML Name/Type</i>
revision	Structured data of type MetadataRevision	element	Revision: xsd:dateTime
error	Structured data, of type Error, defined in the SEMI Specification for Semiconductor Common Components	Element	Error: ccs:ErrorType  There are no SEMI E125-defined error codes for this operation.

#### 7.2.16 NotifyOnRevisions

7.2.16.1 This section is provisional, pending approval of SEMI E132.1 and SEMI E138.

##### 7.2.16.2 Application of SEMI E132 Sessions

7.2.16.3 The equipment shall maintain revision notification settings on a per-Session basis. For example, the same SEMI E132 Principal may establish one or more SEMI E132 Sessions on the equipment. If only one of those



Sessions requests revision notification, the equipment shall only enable notifications for that specific session. The equipment shall not notify other Sessions that have not enabled notification. As described in ¶7.3.1.2, the equipment shall use the URL provided with the HTTPEndpoint information provided with the SEMI E132 EstablishSession request to send all SEMI E125 notifications.

7.2.16.4 SEMI E125 specifies that the equipment shall disable notifications if it detects that a client is no longer reachable. Because this specification requires clients to establish a SEMI E132 Session in order to honor SEMI E125 requests. Clients (SEMI E132 Principals) are considered reachable if and only if their SEMI E132 Session has not been closed (that is, the Session is in either the “Established” or “Frozen” state), as defined by the SEMI E132 Session State Model. If the equipment or the client terminates an SEMI E132 Session for any reason (causes the Session to close), notifications for that Session shall be automatically disabled by the equipment. That client will be required to re-establish an SEMI E132 Session and re-send a NotifyOnRevisions request in order to re-establish revision notifications. If the client’s Session is placed in the “Frozen” state, the equipment shall retain that Session’s notification settings in non-volatile memory, and resume notifications when the Session returns to the “Established” state.

#### 7.2.16.5 WSDL Operation Binding

7.2.16.5.1 See the operation binding for “NotifyOnRevisions” in E125-1-V0305-EqpMetadataMgr-Binding.wsdl. Key information is shown for convenience in the following table.

**Table 81 NotifyOnRevisions Operation Binding**

<i>SOAPAction</i>	urn:semi-org:ws.E125-1.V0305.esdMetaEqp-binding:NotifyOnRevisions
<i>Input Headers (WSDL Message, Required)</i>	E132HeaderMessage, required
<i>Output Headers (WSDL Message, Required)</i>	E132HeaderMessage, required

#### 7.2.16.6 WSDL PortType Operation

7.2.16.6.1 See the portType operation definition for “NotifyOnRevisions” in E125-1-V0305-EqpMetadataMgr-PortType.wsdl. Key information is shown for convenience in the following table.

**Table 82 NotifyOnRevisions PortType Operation**

<i>Input Message Name</i>	NotifyOnRevisionsRequestMessage
<i>Output Message Name</i>	NotifyOnRevisionsResponseMessage

#### 7.2.16.7 WSDL Message(s)

7.2.16.7.1 See the message definitions for “E132HeaderMessage”, “GetLatestRevisionRequestMessage”, and “GetLatestRevisionResponseMessage” in E125-1-V0305-EqpMetadataMgr-PortType.wsdl. Key information is shown for convenience in the following table.

**Table 83 EquipmentMetadataManager Binding**

<i>Message Name → Schema Element Name</i>	E132HeaderMessage → auth:E132Header NotifyOnRevisionsRequestMessage → esd:NotifyOnRevisionsRequest NotifyOnRevisionsResponseMessage → esd:NotifyOnRevisionsResponse
---	---

#### 7.2.16.8 XML Schema Types

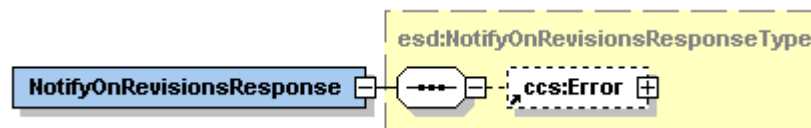
**NotifyOnRevisionsRequest**

**Figure 31  
NotifyOnRevisionsRequest Global Element**

7.2.16.8.1 *NotifyOnRevisionsRequest* — This global element acts as a top-level container entity representing the SOAP body of a GetLatestRevision request. As defined in SEMI E125, this operation has one input argument: a Boolean indicating the client’s notification preference. This argument is modeled as a single attribute of the NotifyOnRevisionsRequest element.

**Table 84 Translation Table for Input Arguments for the NotifyOnRevisions Operation**

Argument Name	Format	XML Element or Attribute	XML Name/Type
notification	boolean	attribute	enable: xsd:boolean



**Figure 32**  
**NotifyOnRevisionsResponse Global Element**

**7.2.16.8.2 NotifyOnRevisionsResponse** — This global element acts as a top-level container entity representing the SOAP body of the response to an SEMI E125 NotifyOnRevisions request. It contains an optional element named “Error” of complexType “ErrorType”, defined in SEMI E138. As defined in SEMI E125, this operation returns no arguments, so there is no other content defined for this response.

**Table 85 Translation Table for Output Arguments for the GetUnits Operation**

Argument Name	Format	XML Element or Attribute	XML Name/Type
error	Structured data, of type Error, defined in the SEMI Specification for Semiconductor Common Components	Element	Error: ccs:ErrorType  No SEMI E125-specific error codes are defined for this operation.

### 7.3 MetadataClient

7.3.1.1 This section is provisional, pending approval of SEMI E132.1 and SEMI E138.

#### 7.3.1.2 Application of SEMI E132 Sessions

7.3.1.3 Operations defined by the SEMI E125 MetadataClient interface shall be exchanged via HTTP and shall not be exchanged via HTTPS. The equipment shall include a hash of the client’s SEMI E132 session identifier in the SOAP envelope header provided with the operation using the E132HashHeader element, as specified by SEMI E132.1

7.3.1.4 The equipment shall only send SEMI E125 notifications to instances of the MetadataClient interface if the corresponding SEMI E132 Session(s) for that client are in the “Established” state. The equipment shall use the URL provided with the HTTPEndpoint information included in the SEMI E132 EstablishSession request to send all SEMI E125 notifications to the MetadataClient interface.

### 7.3.2 XML Schema and WSDL Files

7.3.2.1 The XML Schema and WSDL defined by this specification for the MetadataClient interface is contained in the following documents:

**Table 86 XML Schema**

File Name	E125-1-V0305-Schema.xsd
Target Namespace	urn:semi-org:xsd:E125-1.V0305.esd
Imported/Referenced Namespaces	urn:semi-org:xsd:E120-1.V1104.CommonEquipmentModel urn:semi-org:xsd.CommonComponents.V0305.ccs <a href="http://www.w3.org/2001/XMLSchema">http://www.w3.org/2001/XMLSchema</a>
Description	This file defines all of the E125 data types and elements used by both the EquipmentMetadataManager and MetadataClient interfaces.

**Table 87 MetadataClient PortType Definitions**

<i>File Name</i>	E125-1-V0305-MetadataClient-PortType.wsdl
<i>Target Namespace</i>	urn:semi-org:ws.E125-1.V0305.esdMetaClnt-portType
<i>Imported/Referenced Namespaces</i>	urn:semi-org:xsd.E132-1.V0305.auth urn:semi-org:xsd.E125-1.V0305.esd http://schemas.xmlsoap.org/wsdl/ http://www.w3.org/2001/XMLSchema
<i>Description</i>	This file defines all of the input/output messages and operations for the MetadataClient interface, based on the data types defined in the SEMI E125 XML Schema.

**Table 88 EquipmentMetadataManager Binding Definitions**

<i>File Name</i>	E125-1-V0305-MetadataClient-Binding.wsdl
<i>Target Namespace</i>	urn:semi-org:ws.E125-1.V0305.esdMetaClnt-binding
<i>Imported/Referenced Namespaces</i>	urn:semi-org:ws.E125-1.V0305.esdMetaClnt-portType http://schemas.xmlsoap.org/wsdl/soap/ http://schemas.xmlsoap.org/wsdl/
<i>Description</i>	This file binds the abstract portType definition to HTTP and SOAP, specifying required SOAP and HTTP headers for each operation and the XML encoding style.

### 7.3.3 WSDL Port Type Overview

7.3.3.1 The MetadataClient WSDL portType definition organizes the SEMI E125.1 XML Schema data types into a collection of named operations with inputs and outputs that correspond to the UML operations that are defined for the MetadataClient interface in SEMI E125. The WSDL portType itself is named after the SEMI E125 interface, and each WSDL portType operation is named after the corresponding UML operation defined for the interface. Each WSDL operation consists of two WSDL message definitions corresponding to the input and output (request and response) for the UML operation defined in SEMI E125. The MetadataClient interface defines only one operation with no outputs (one-way).

**Table 89 EquipmentMetadataManager Port Type**

<i>SEMI E125 Class Name</i>	MetadataClient
<i>WSDL Port Type Name</i>	MetadataClient
<i>SEMI E125 Operation → WSDL Operation</i>	MetadataRevised → MetadataRevised

### 7.3.4 WSDL Binding Overview

7.3.4.1 The WSDL MetadataClient binding definition specifies a message and transport protocol to use for a given portType (SOAP and HTTP for SEMI E125.1), the interface style used (document style for SEMI E125.1). These settings are shown in Table 13. For each WSDL portType operation, the binding defines any SOAP headers used and whether or not they are required, the HTTP SOAPAction header value to use for that operation, and the XML encoding to use (literal). The binding definitions for each portType operation are described in ¶¶7.2.8 through 7.2.16.

**Table 90 EquipmentMetadataManager Binding**

<i>SEMI E125 Class Name</i>	MetadataClient
<i>WSDL Binding Name</i>	MetadataClient Binding
<i>SOAP Binding Style</i>	document
<i>SOAP Transport</i>	http://schemas.xmlsoap.org/soap/http

### 7.3.5 WSDL Service Overview

7.3.5.1 This specification does not provide a WSDL service definition for the MetadataClient interface. WSDL service definitions define the address(es) at which a given interface can be accessed by a client. Such information is

provided dynamically to the equipment during SEMI E132 Session establishment, using the EstablishSession message.

### 7.3.6 MetadataRevised

#### 7.3.6.1 WSDL Operation Binding

7.3.6.1.1 See the operation binding for “MetadataRevised” in E125-1-V0305-MetadataClient-Binding.wsdl. Key information is shown for convenience in the following table.

**Table 91 MetadataRevised Operation Binding**

<i>SOAPAction</i>	urn:semi-org:ws.E125-1.V0305.esdMetaEqp-binding:MetadataRevised
<i>Input Headers (WSDL Message, Required)</i>	E132HashHeaderMessage, required
<i>Output Headers (WSDL Message, Required)</i>	E132HashHeaderMessage, required

#### 7.3.6.2 WSDL PortType Operation

7.3.6.2.1 See the portType operation definition for “MetadataRevised” in SEMI E125- MetadataClient - PortType.wsdl. Key information is shown for convenience in the following table.

**Table 92 MetadataRevised PortType Operation**

<i>Input Message Name</i>	MetadataRevisedNotificationMessage
---------------------------	------------------------------------

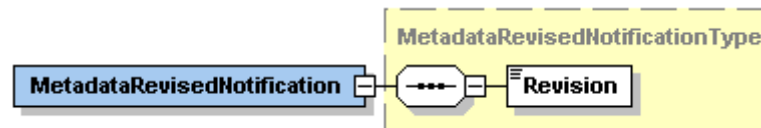
#### 7.3.6.3 WSDL Message(s)

7.3.6.3.1 See the message definitions for “E132HashHeaderMessage” and “MetadataRevisedNotificationMessage” in E125-1-V0305-MetadataClient-PortType.wsdl. Key information is shown for convenience in the following table.

**Table 93 EquipmentMetadataManager Binding**

<i>Message Name → Schema Element Name</i>	E132HashHeaderMessage → auth:E132HashHeader NotifyOnRevisionsRequestMessage → esd: MetadataRevisedNotification
---	---

#### 7.3.6.4 XML Schema Types



**Figure 33  
MetadataRevisedNotification Global Element**

7.3.6.4.1 *MetadataRevisedNotification* — This global element acts as a top-level container entity representing the SOAP body of a MetadataRevised notification. As defined in SEMI E125, this operation has one input argument: the current revision of the equipment metadata. This argument is modeled as a single element named “Revision” of built-in type “xsd:dateTime”.

**Table 94 Translation Table for Input Arguments for the NotifyOnRevisions Operation**

<i>Argument Name</i>	<i>Format</i>	<i>XML Element or Attribute</i>	<i>XML Name/Type</i>
revision	Structured data of type RevisionNotice	element	Revision: xsd:dateTime.

7.3.6.4.2 *Revision* — This global element is the XML Schema representation of the SEMI E125 RevisionNotice class.

**Table 95 SEMI E125 RevisionNotice → Revision Translation Table**

<i>Attribute or Role Name</i>	<i>UML Name/Type</i>	<i>XML Element or Attribute</i>	<i>XML Name/Type</i>
equipment	One and only one element of a type derived from the SEMI E120 type EquipmentElement	n/a	No SEMI E120 equipment node is provided. Clients determine which equipment has been updated via information provided in the SEMI E132 header.
newRevision	Structured data, of type MetadataRevision	element	Revision: xsd:dateTime

**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 E126-0305**

## **SPECIFICATION FOR EQUIPMENT QUALITY INFORMATION PARAMETERS (EQIP)**

This specification was technically approved by the Global Information & Control Committee and is the direct responsibility of the Japanese Information & Control Committee. Current edition approved by the Japanese Regional Standards Committee on January 11, 2005. Initially available at [www.semi.org](http://www.semi.org) February 2005; to be published March 2005. Originally published July 2003; last published July 2004.

### **1 Purpose**

1.1 This standard identifies key equipment quality information parameters, or EQIPs, for equipment process types or groups. The identification of these quality parameters is intended to help equipment users and suppliers specify and utilize capabilities for process control.

1.2 This standard is designed to be updated as EQIPs are identified for new and/or existing equipment process types in this standard.

1.3 *Intended Audience* — This standard is intended for use by OEMs, users, and third party equipment and process control system suppliers who need to achieve equipment and process control of select process quality parameters, utilizing standardized methods for actuation of process control parameters.

### **2 Scope**

2.1 The scope of this standard is limited to identifying key EQIPs for select equipment process types, specifying that the equipment shall indicate a non-null list of settable process program parameters that impact each of these EQIPs, and specifying that the equipment shall allow individual actuation of each of these parameters using SEMI standard methods. The standard is not intended to identify all quality parameters. It is not intended to specify a complete set of equipment types.

2.2 While this standard requires that a capability shall exist for actuation of process program parameters to influence key EQIPs, the standard does not specify how this actuation will take place. For example, it does not define the communication protocol for actuation or the equipment or host models that will support and enable this actuation capability. The scope is limited to defining the key EQIPs and their linking to process program parameters.

2.3 The standard is structured so that EQIPs for equipment types may be added to an existing set, as the community reaches consensus on these additions. Further, EQIPs may be defined for a new equipment type not previously addressed in this standard. Additions to the standard are made utilizing the SEMI standards balloting process.

**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 The standard is structured so that EQIPs for equipment types may be added to an existing set, as the community reaches consensus on these additions. Further, EQIPs may be defined for a new equipment type not previously addressed in this standard. Thus, this standard does not necessarily specify a complete EQIP set for an equipment type, and does not necessarily address all equipment types.

3.2 This standard does not specify how actuation of equipment settable process program parameters associated with these EQIPs will take place. It is limited to referencing other SEMI standards as examples of standardized methods for providing this capability.

## 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)

SEMI E40 — Standard for Processing Management

SEMI E94 — Provisional Specification for Control Job Management

### 4.2 Other Sources

Object Oriented Modeling and Design, — Prentice Hall, Englewood Cliffs, NJ, c1991 by James Rumbaugh, et. al.

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

## 5 Terminology

### 5.1 Abbreviations and Acronyms

5.1.1 *EQIP (Equipment Quality Information Parameter)* — a parameter that relates an aspect or quality of a wafer process or product. Specifically, an EQIP is a parameter of measure on a wafer that relates to the quality of the wafer, i.e., the closeness of the wafer to a design specification or the performance of the device being produced. EQIPs may be measurable directly or indirectly. EQIPs do not have to be measured at the process tool. EQIPs may also be referred to as “process quality parameters”.

5.1.2 *PPP (Process Program Parameter)* — a parameter in the preplanned and reusable portion of the set of instructions, settings, and parameters under control of the equipment that determine the processing environment seen by the manufactured object and that may be subject to change between runs or processing cycles.

5.1.3 *R2R Control—Run-to-Run Control* — techniques for varying settings in one run based on analysis of either incoming product (feedforward) or product from an earlier run (see SEMI E98). In this document, R2R Control is further defined as a form of discrete process control in which settable process attributes are adjusted, generally between process runs, to better achieve selected process quality targets.

### 5.2 Definitions

5.2.1 *equipment type* — a categorization or grouping of equipment based on capability, method of operation, effect on wafer, etc. The boundary of the type should be aligned with a generally perceived categorization of equipment in the industry.

5.2.2 *linked process program parameter* — an equipment process program parameter that can be altered to effect change of an EQIP with which it is associated.

## 6 Background

6.1 Equipment processes can generally be categorized or grouped into types. An equipment or equipment process can be associated with one of more types. Further, each type can be associated with a finite set of quality parameters that relate to the quality of process on the wafer. These process quality parameters are a function of tunable process inputs as well as external uncontrollable conditions, as shown in Figure 1.

6.2 Control solutions can provide a capability to improve process quality parameters by suggesting changes to tunable input parameters. Standards are needed to identify control capabilities for equipment. Specifically, a standard is needed to specify these quality parameters so as to guide the use of metrology and process program modification to support capabilities such as run-to-run (R2R) control.

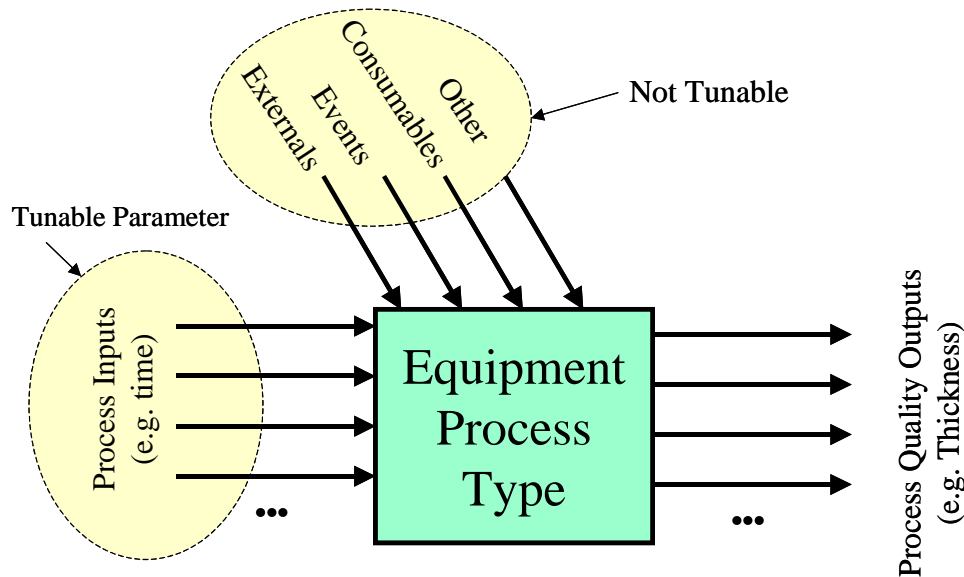


Figure 1

#### Equipment and Process Quality Parameters

(note that, in the figure, EQIPs are Process Quality Outputs and Linked Process Program Parameters—PPPs are Process Inputs that are tunable according to SEMI defined standard methods)

## 7 Equipment Type Organization

7.1 This standard categorizes semiconductor equipment into types according to functionality, effect, and or operational characteristics. The number of types is not limited. An equipment complying with this standard must belong to at least one type; an equipment can belong to more than one type. Note that the list of types in this standard does not necessarily provide complete specification for all equipment. Note also that the list of types is extendible as new equipment types are added.

7.2 For each specified equipment type, the following information shall be provided.

7.2.1 *Equipment Type Name* — A definition for the type shall be provided that represents a common name used for this equipment type in the industry.

7.2.2 *Equipment Type Definition* — A definition of the type shall be provided. This definition shall indicate what the equipment does and, as necessary, how it does it. It shall also provide a scope of the equipment type, indicating as necessary which equipment belongs to this type and which equipment does not belong to this type.

7.2.3 List of Equipment Quality Information Parameters (EQIPs). For each equipment type, a non-null list of EQIPs must be specified.

7.3 For each EQIP specified, the following information shall be provided.

7.3.1 *Name* — A name for the EQIP shall be provided that represents a common name used for this parameter in the industry.

7.3.2 *Definition* — A precise definition of the EQIP shall be provided. If the parameter is derived from measurement data, the form of this derivation shall be provided. If this parameter is provided in different variants, each of these variants shall be listed and defined. The definition could include an indication of how the parameter is or is not measured.

7.3.3 *Format Descriptor Fields* — A listing of data type fields to be used to specify the format of an EQIP shall be provided. These fields could include data type, data length, data units, indication of requirement, range of values, and values used to represent error/fault/invalid data, etc. Note that the specification does not contain values for these fields; it is the listing of the fields.

7.4 *Specification Mechanism* — The listing and definitions of equipment types and EQIPs, and linking of EQIPs to types are specified utilizing the following components.

7.4.1 *Equipment Type Definitions Table* — This table (Table 1) contains a definition of all equipment type elements. Fields in this table include Equipment Type Name and Equipment Type Definition as specified in ¶7.2.

7.4.2 *Equipment Type to EQIP Associations Table* — This (Table 2) provides a linkage between equipment type elements and EQIPs. Fields in this table include Equipment Type Name, and “linked” EQIPs.

7.4.3 *EQIP Definitions and Format Table* — This table (Table 3) contains a definition of all EQIPs specified. Fields in this table include EQIP Name, EQIP Definition and Format Descriptor.

7.5 *Adding to the List of Equipment Types* — The standard is structured so that EQIPs for equipment types may be added to an existing set, as the community reaches consensus on these additions. Further, EQIPs may be defined for a new equipment type not previously addressed in this standard. Additions to the standard are made utilizing the SEMI standards balloting process. §10 provides a template of a SEMI ballot for adding to the collection. The following sub-sections define limitations in adding these equipment types and quality parameters.

7.5.1 *Adding Equipment Types* — Equipment types added shall comply with the specifications provided in this document. No addition shall invalidate an existing type. Wherever possible, an equipment type should have a corollary to a physically distinct classification of equipment or process.

7.5.2 *Adding EQIPs* — EQIPs added to the collection shall comply with the specifications provided in this document. A EQIP shall be associated with at least one equipment type. An attempt should be made to define quality parameters for a type that are exclusive of existing quality parameters.

## 8 Equipment Quality Information Parameter (EQIP) Specification

8.1 A list of equipment types currently specified, and a definition of each type is provided in Table 1.

8.2 A listing of EQIPs associated with each equipment type is provided in Table 2. Note that all types are required to have EQIPs associated with them. Note that an EQIP may be associated with more than one equipment type.

8.3 A definition of each EQIP is provided in Table 3.

**Table 1 Equipment Type Definitions**

<i>Equipment Type Name</i>	<i>Equipment Type Definition</i>
Removal	Type of equipment that removes material from the surface of a wafer. The method of removal is not restricted and could include chemical, physical, or nuclear/plasma, or a combination of these methods.
Polishing	Type of equipment that removes material from the surface of a wafer using of physical and possibly chemical means. The physical mechanism involves utilizing abrasive friction and/or grinding methods. All chemical-mechanical polishers belong to this type, including rotary and linear polishers. Equipment that utilize purely chemical or non-abrasive physical and chemical methods, such as equipment commonly referred to as etchers, do not belong to this type.
Lithography	Type of equipment cluster that transfers a pattern from a mask or reticle to a layer of resist deposited on the surface of a wafer.
Deposition	Type of equipment that adds material on the surface of a wafer. It does not include equipment that adds content to the wafer, but does not add material to the surface of the wafer, such as implant. The method of deposition is not restricted and could include chemical, physical, or nuclear/plasma, or a combination of these methods.
Diffusion	Type of equipment that enables the motion of species in solids in the direction of a concentration gradient usually utilizing heat. The process often results in a layer (e.g., oxide) growing on the surface of the wafer.
Etching	Type of equipment that removes material from the surface of a wafer using chemical and possibly physical means. It includes wet etching and dry etching. In wet etching the material is dissolved in liquid chemicals; in dry etching, which includes reactive ion etching, the material is removed through conversion into gaseous compounds.
Chemical Vapor	A type of deposition equipment where films are deposited on the surface of the wafer as the result of

Deposition (CVD)	chemical reactions between gaseous components at or near the surface of the wafer, usually occurring at an elevated temperature.
------------------	--

**Table 2 Equipment Type to EQIP Association**

<i>Equipment Type Name</i>	<i>EQIP (Required (Y/N)?)</i>	<i>Comments</i>
Removal	Amount Removed (Y)	
Polishing	Amount Removed (Y), Uniformity (Y)	
Deposition	Amount Deposited (Y)	
Lithography	Overlay Offset (Y)	
Lithography	CD (Y)	
Diffusion	Resistivity (Y)	
Etching	Amount Removed (Y), CD (N), Uniformity (N)	
Chemical Vapor Deposition	Amount Deposited (Y), Uniformity (N)	

**Table 3 EQIP Definitions**

<i>#</i>	<i>EQIP Name</i>	<i>EQIP Definition</i>	<i>Format Descriptor Fields</i>
1	Amount Removed	The average amount of material removed from a wafer during the course of a process. Amount Removed is computed as (average thickness prior to removal step – average thickness directly after removal process).	Name, Units, Validity
2	Uniformity	A measure of the variation of thickness of a feature on a wafer. It is calculated utilizing thickness data collected at different points across the wafer. The number of data points can vary, but is generally greater than or equal to five. Typical uniformity metrics include the difference between the maximum and minimum points, the variance or standard deviation of the points, or the average difference between the center points and the outer points, commonly referred to as “center-to-edge”. Definition of method for calculating this parameter is outside the scope of this standard.	Name, Units, Validity, Method of Calculation
3	Amount Deposited	The average amount of material deposited on a wafer during the course of a process. Amount Deposited is computed as: (average thickness directly after deposition process – average thickness immediately prior to deposition step)	Name, Units, Validity
4	Overlay Offset	A measure of the displacement of a feature at the top layer with respect to another layer (usually the layer directly below it) on a wafer. It is usually reported as a delta in the ‘X’ and ‘Y’ directions for each measured point on the wafer.	Name, Units, Validity
5	CD	Critical Dimension — The average width of the smallest patterned feature on the wafer.	Name, Units, Validity
6	Resistivity	A material’s opposition to the flow of electric current usually used to detect properties of buried features in a wafer.	Name, Units, Validity

## 9 Compliance to this Standard

9.1 This section defines requirements placed on equipment types for claiming compliance with this standard and verifying compliance with this standard.

**9.2 Declaration of Equipment Type(s)** — An equipment complying with this standard shall provide an indication (i.e., declaration) of its equipment type or types. This declaration shall be determined through matching of equipment description to the equipment type definitions in Table 1.

**9.3 Specification of Linked Process Program Parameters** — An equipment complying with this standard must specify a non-null set of settable equipment process program parameters that can be altered to effect change of each EQIP associated with this equipment type or types. These process program parameters are termed “linked process program parameters”. It is not required that the actual mathematical relationship between each EQIP and associated process program parameters be specified.

**9.4 Method for Effecting Control of EQIPs** — An equipment complying with this standard must support and specify at least one SEMI standard method for individual actuation of each linked process program parameters. An EQIP is not supported unless a SEMI standard method for effecting control of the EQIP through individual actuation of linked process program parameters is supported and defined. The following are SEMI standard methods for individual actuation of process program parameters that could be utilized.

**9.4.1 SEMI E30 (Generic Model for Communications and Control of Manufacturing Equipment (GEM)) Application Note A.8: Process Parameter Modification for Process and Equipment Control** — This specification defines a method for effecting control of EQIPs utilizing the GEM “Equipment Constants” capability. These Equipment Constants could be set up as Linked PPPs thereby enabling control of EQIPs. Refer to these sections of SEMI E30 for a precise definition of the GEM capability to be used and an example of its use for process control.

**9.4.2 SEMI E40 (Standard for Process Management)** — This standard specifies a method for setting recipe variables, which can be assigned a value at Process Job creation time, and can be adjusted at a later time. These recipe variables could be set up as Linked PPPs, thereby enabling control of EQIPs. Refer to ¶¶7.5.4 and 10.4 of SEMI E40 for additional information.

**9.5 Compliance Sheets** — Compliance to this standard by equipment shall be indicated with compliance sheets that shall be provided with the equipment. Templates for the compliance sheets are provided in Tables 4 and 5.

**9.5.1 EQIP and Linked PPP Support Compliance Sheet** — A template for this compliance sheet is provided in Table 4. This compliance sheet has the following fields:

**9.5.1.1 Unique Identifier Corresponding to Equipment** — This field provides for a unique identification of the equipment type for that manufacturer. The contents of this field are manufacturer specific, but could include the company name and equipment model number.

**9.5.1.2 Equipment Type** — This field will indicate the specific type or types, from Table 1, to which this equipment belongs.

**9.5.1.3 Listing of EQIPs** — This field will indicate the EQIPs defined for each equipment type specified for this equipment, copied from the list in Table 2 associated with each equipment type. Modifiers for this field will indicate, for each EQIP, whether the EQIP is required and whether it is supported for this equipment model. Note that if an EQIP is specified as required for that equipment type in Table 2, then this EQIP must be specified as supported. If the EQIP is specified as optional for that equipment type then the field modifier shall indicate whether that EQIP is supported.

**9.5.1.4 Listing of Linked Process Program Parameters** — This field will indicate, for each supported EQIP, a non-null list of linked process program parameters.

**9.5.1.5 Additional Information** — This optional field will provide additional information that could include forms of models relating EQIPs to linked process program parameters, value limits on actuation of these parameters, preferred method(s) and timing for actuation of these parameters, and behavior of equipment in response to actuation of these parameters. The contents of this field are manufacturer specific.

**9.5.2 EQIP and Linked PPP Data Format Compliance Sheet** — A template for this compliance sheet is provided in Table 5. This compliance sheet has the following fields:

**9.5.3 Listing of Supported EQIPs** — This field is a listing of supported EQIPs. This is a subset of the listed EQIPs in the corresponding “EQIP and Linked PPP Support Compliance Sheet” (see ¶9.5.1.3).

**9.5.4 EQIP Format Descriptors** — These fields collectively provide the description of the format of the supported EQIPs. The fields shall be copied from the list for each supported EQIP specified in Table 3.

9.5.5 *Listing of Supported Linked Process Program Parameters* — This field is a listing of supported PPPs as indicated in the corresponding “EQIP and Linked PPP Support Compliance Sheet” (see ¶9.5.1.3).

9.5.6 *Linked PPP Format Descriptors* — These fields collectively provide the description of the format of the supported EQIPs. Note that required descriptor fields are Name, Data Type, and Data Units. Additional descriptor fields can be provided as desired.

**Table 4 EQIP and Linked PPP Support Compliance Sheet Template**

<b>Company</b>	<i>Company Name</i>				
<b>Unique Identifier</b>	<i>Key for Identifying Classification of Equipment Within Company (e.g., Model Number(s))</i>				
<b>Equipment Type(s)</b>	<i>Types from Figure 2; minimum of one type must be specified</i>				
	<b>EQIP</b>	<b>Required</b>	<b>Supported</b>	<b>Linked PPPs</b>	<b>Comments</b>
	<i>From Table 2, linked to Equipment Type</i>	<i>From Table 2</i>	<i>Y/N</i>		<i>Could include recommended method for measuring and/or calculating EQIP parameter</i>
	...				
<b>Supported Methods for Individual Actuation of Linked PPPs</b>					
<b>Additional Information</b>	<i>Additional Information, See ¶9.5.1.5</i>				

<sup>#1</sup> Text in italics provides further description of a field; it is not part of the template.

**Table 5 EQIP and Linked PPP Data Format Compliance Sheet Template**

Company	Company Name					
Unique Identifier	Key for Identifying Classification of Equipment Within Company (e.g., Model Number(s))					
Equipment Type (s)	From list in Table 1					
EQIP DATA FORMATS						
		Format Descriptors				
	EQIP	From Table 3	From Table 3	From Table 3	...	Comments
	Supported EQIPs (from EQIP and Linked PPP Support Compliance Sheet)				Could include additional descriptors	
	...					
LINKED PPP DATA FORMATS						
		Format Descriptors				
	Linked PPP	Name	Data Type	Data Units	...	Comments
	Supported Linked PPPs (from EQIP and Linked PPP Support Compliance Sheet)				Could include additional descriptors	

	...					

<sup>#1</sup> Text in italics provides further description of a field; it is not part of the template. Format descriptors for EQIPs represent union of required descriptors for supported EQIPs. Format descriptor fields for an EQIP do not have to be filled out if the entity claiming compliance and submitting the compliance sheets does not provide a measurement of the EQIP (e.g., the EQIP for the process tool is measured by another metrology tool). A value of “N.A.” indicating “not applicable” is inserted wherever the descriptor is not specified and is not supported for an EQIP.

## 10 Template for Adding Equipment Types and EQIPs to this Standard

10.1 The standard is structured so that it can be updated. Specifically, EQIPs for equipment types may be added to an existing set, as the community reaches consensus on these additions. Further, EQIPs may be defined for a new equipment type not previously addressed in this standard. Thus typical additions to this standard will impact Table 2 and Table 3, and possibly Table 1 of this standard. These typical additions should not impact any other portions of this standard.

10.2 *Template* — The template for typical additions to this standard is shown in Table 6. Note that this template addresses only the technical content of the ballot addition to the standard. For development and submission of the complete ballot, refer to SEMI documentation.

**Table 6 Template for Typical Additions to this Standard**

#	Ballot Component	Conditions for including in Ballot
1	Add the following rows to Table 1	<i>Include only if adding equipment types. Add one row for each type added.</i>
2	Add the following EQIP information to the EQIP column of Table 2	<i>Include only if adding EQIPs to existing equipment types. Add for each new EQIP defined. Add to each equipment type that will use this EQIP. Indicate requirement (Y/N). Note that one of ballot components 2 and 3 must be included in the ballot.</i>
3	Add the following rows to Table 2	<i>Include only if adding equipment types. Define the EQIPs for these new types. Note that one of ballot components 2 and 3 must be included in the ballot.</i>
4	Add the following rows to Table 3	<i>This component must be included in the ballot. Add a row for each new EQIP defined.</i>

<sup>#1</sup> Column 1 is an index column utilized only for reference.

**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 E127-0705**

## **SPECIFICATION FOR INTEGRATED MEASUREMENT MODULE**

### **COMMUNICATIONS: CONCEPTS, BEHAVIOR, AND SERVICES (IMMC)**

This specification 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](http://www.semi.org) in June 2005 and on CD-ROM in July 2005. Originally published July 2003; previously published November 2004.

#### **Contents**

1 Purpose .....	4
1.1 <i>Integrated Measurement</i> .....	4
1.2 <i>Specification for the Integrated Measurement Module Communications (IMMC)</i> .....	4
2 Scope .....	4
2.1 <i>Scope of This Document</i> .....	4
3 Limitations .....	4
3.1 <i>Host/Integrated Equipment Interface</i> .....	4
3.2 <i>Object-Based Implementation</i> .....	4
3.3 <i>In-Situ Processing</i> .....	4
4 Referenced Standards and Documents .....	4
4.1 <i>SEMI Standards</i> .....	4
4.2 <i>Other Sources</i> .....	5
5 Terminology .....	5
5.1 <i>Abbreviations and Acronyms</i> .....	5
5.2 <i>Definitions</i> .....	5
6 Conventions .....	6
6.1 <i>Object Conventions</i> .....	6
6.2 <i>State Model Conventions</i> .....	7
6.3 <i>Service Message Representation</i> .....	7
7 General Requirements .....	8
7.1 <i>Protocol Requirements</i> .....	8
7.2 <i>Required Standards</i> .....	8
7.3 <i>Required Basic Capabilities</i> .....	8
7.4 <i>State Models</i> .....	9
7.5 <i>Objects</i> .....	9
7.6 <i>Service Requirements</i> .....	10
8 Overview .....	10
8.1 <i>Background</i> .....	10
9 Communication Ports and Clients .....	10
9.1 <i>Ports</i> .....	10
9.2 <i>Clients</i> .....	10
9.3 <i>Bandwidth and Performance</i> .....	11
10 Integrated Measurement Module Object Model .....	11
10.1 <i>Description</i> .....	11
11 The Integrated Measurement Module Object .....	12
11.1 <i>Relationship of IMM Object And Physical Module</i> .....	12
11.2 <i>IMM Service State Model</i> .....	12
11.3 <i>Object Attributes</i> .....	13
11.4 <i>System Initialization</i> .....	15
11.5 <i>IMM Events and Alarms</i> .....	16
11.6 <i>IMM Object Services</i> .....	17
12 Substrate Transfer Path Object .....	18
12.1 <i>Description</i> .....	18
12.2 <i>Object State Model</i> .....	18
12.3 <i>Object Attributes</i> .....	20