

SEMI E125.1-0414

SPECIFICATION FOR SOAP BINDING FOR EQUIPMENT SELF DESCRIPTION (EqSD)

This Standard was technically approved by the Information & Control Global Technical Committee. This edition was approved for publication by the global Audits and Reviews Subcommittee on December 23, 2013. Available at www.semiviews.org and www.semi.org in April 2014; originally published March 2005; previously published July 2010.

NOTICE: This Standard or Safety Guideline has an Inactive Status because the conditions to maintain Current Status have not been met. Inactive Standards or Safety Guidelines are available from SEMI and continue to be valid for use.

NOTICE: The section symbol (§) is used to reference a numbered section, paragraph, or header and all subordinate paragraphs, headers, and embedded material (i.e., EXCEPTIONs, lists) therein. The paragraph symbol (§) is used to reference a particular numbered paragraph and embedded material therein. When duplicated (i.e., §§ and §§) the symbols are used to reference more than one section or paragraph, respectively.

Table of Contents

1 Purpose	4
1.1 <i>SOAP Implementation Mapping</i>	4
2 Scope	4
3 Limitations	4
3.1 None	4
4 Referenced Standards and Documents	4
4.1 <i>SEMI Standards</i>	4
4.2 <i>OMG Standards</i>	4
4.3 <i>W3C Standards</i>	4
4.4 <i>Other Standards</i>	5
5 Terminology	5
5.1 <i>Abbreviations and Acronyms</i>	5
5.2 <i>Definitions</i>	5
6 Conventions	5
6.1 <i>Translating UML to XML Schema</i>	5
6.2 <i>Documenting XML Schema and WSDL Files</i>	6
6.3 <i>Documenting XML Schema with Diagrams</i>	7
6.4 <i>XML Schema Sample</i>	8
6.5 <i>Translating UML to WSDL</i>	9
7 Mapping of SEMI E125 UML to XML Schema and WSDL	9
7.1 <i>WSDL Organization</i>	9
7.2 <i>EquipmentMetadataManager</i>	10
7.3 <i>MetadataClient</i>	39
8 Message Transport	42
RELATED INFORMATION 1	43

List of Figures

Figure 1 XML Schema Example Diagram	8
Figure 2 XML for Sample	8
Figure 3 XML Schema and WSDL File Organization	10
Figure 4 WSDL and XML Schema Files for SEMI E125	10
Figure 5 Mapping the EquipmentMetadataManager to a WSDL portType	12
Figure 6 GetUnitsRequest Global Element	14
Figure 7 GetUnitsResponse Global Element	14
Figure 8 GetTypeDefinitionsRequest Global Element	15
Figure 9 GetTypeDefinitionsResponse Global Element	16
Figure 10 GetStateMachinesRequest Global Element	17

Figure 11 GetStateMachinesResponse Global Element	17
Figure 12 StateMachineType Global complexType	18
Figure 13 GetSEMIObjTypesRequest Global Element	21
Figure 14 GetSEMIObjTypesResponse Global Element	21
Figure 15 StateMachineInstanceType Global complexType	23
Figure 16 GetExceptionsRequest Global Element	25
Figure 17 GetExceptionsResponse Global Element	26
Figure 18 GetEquipmentStructureRequest Global Element	27
Figure 19 GetEquipmentStructureResponse Global Element	28
Figure 20 GetEquipmentNodeDescriptionsRequest Global Element	29
Figure 21 GetEquipmentNodeDescriptionsResponse Global Element	29
Figure 22 EquipmentNodeDescriptionType Global complexType	31
Figure 23 ParameterType Global complexType	33
Figure 24 GetLatestRevisionRequest	36
Figure 25 GetLatestRevisionResponse Global Element	37
Figure 26 NotifyOnRevisionsRequest Global Element	38
Figure 27 NotifyOnRevisionsResponse Global Element	39
Figure 28 MetadataRevisedNotification Global Element	41

List of Tables

Table 1 Example Translation Table	6
Table 2 Example Translation Table for Operation Input/Output Arguments	6
Table 3 Example Schema/WSDL Document Description Table	7
Table 4 Altova XMLSPY Schema Diagram Symbols	7
Table 5 Example Interface WSDL Port Type Table	9
Table 6 Example Interface WSDL Binding Table	9
Table 7 Example Operation Binding Table	9
Table 8 Example PortType Operation Table	9
Table 9 XML Schema	11
Table 10 EquipmentMetadataManager PortType Definitions	11
Table 11 EquipmentMetadataManager Binding Definitions	11
Table 12 EquipmentMetadataManager Port Type	12
Table 13 EquipmentMetadataManager Binding	13
Table 14 GetUnits Operation Binding	13
Table 15 GetUnits PortType Operation	13
Table 16 EquipmentMetadataManager Binding	14
Table 17 Translation Table for Output Arguments for the GetUnits Operation	14
Table 18 GetTypeDefinitions Operation Binding	15
Table 19 GetUnits PortType Operation	15
Table 20 EquipmentMetadataManager Binding	15
Table 21 Translation Table for Output Arguments for the GetTypeDefinitions Operation	16
Table 22 GetStateMachines Operation Binding	16
Table 23 GetStateMachines PortType Operation	16
Table 24 EquipmentMetadataManager Binding	17
Table 25 Translation Table for Output Arguments for the GetStateMachines Operation	19
Table 26 SEMI E125 StateMachine → StateMachineType Translation Table	19
Table 27 SEMI E125 State → StateType Translation Table	19
Table 28 SEMI E125 Transition → TransitionType Translation Table	19
Table 29 SEMI E125 Event → EventType Translation Table	20
Table 30 GetSEMIObjTypes Operation Binding	20
Table 31 GetSEMIObjTypes PortType Operation	20
Table 32 EquipmentMetadataManager Binding	21
Table 33 Translation Table for Output Arguments for the GetSEMIObjTypes Operation	22
Table 34 SEMI E125 SEMIObjType → SEMIObjTypeType Translation Table	22
Table 35 SEMI E125 StateMachineInstance → StateMachineInstanceType Translation Table	23
Table 36 Example Parameter References	24

Table 37 SEMI E125 EventToParameterMap → EventToParameterMapType Translation Table	24
Table 38 GetExceptions Operation Binding	25
Table 39 GetExceptions PortType Operation	25
Table 40 EquipmentMetadataManager Binding	25
Table 41 Translation Table for Output Arguments for the GetExceptions Operation	26
Table 42 SEMI E125 Exception → ExceptionType Translation Table	26
Table 43 GetEquipmentStructure Operation Binding	27
Table 44 GetEquipmentStructure PortType Operation	27
Table 45 EquipmentMetadataManager Binding	27
Table 46 Translation Table for Output Arguments for the GetEquipmentStructure Operation	28
Table 47 GetEquipmentNodeDescriptions Operation Binding	28
Table 48 GetEquipmentNodeDescriptions PortType Operation	28
Table 49 EquipmentMetadataManager Binding	29
Table 50 Translation Table for Input Arguments for the GetEquipmentNodeDescriptions Operation	29
Table 51 Translation Table for Output Arguments for the GetEquipmentNodeDescriptions Operation	30
Table 52 SEMI E125 EquipmentNodeDescription → EquipmentNodeDescriptionType Translation Table	32
Table 53 SEMI E125 SimpleEvent → SimpleEventType Translation Table	32
Table 54 SEMI E125 ExceptionReference → ExceptionReferenceType Translation Table	32
Table 55 SEMI E125 SECSVarRef → SECSVarRefType Translation Table	33
Table 56 SEMI E125 Parameter → ParameterType Translation Table	34
Table 57 SEMI E125 Constraint → ConstraintType Translation Table	34
Table 58 SEMI E125 AssociatedParameter → AssociatedParameterType Translation Table	34
Table 59 SEMI E125 Units → UnitType Translation	35
Table 60 SEMI E125 UnitConfigArray → UnitConfigArrayType Translation	35
Table 61 SEMI E125 UnitConfig → UnitConfigType Translation	35
Table 62 SEMI E125 ParameterClassification → ParameterClassificationType Translation Table	35
Table 63 SEMI E125 ReadWrite → ReadWriteType Translation Table	35
Table 64 GetLatestRevision Operation Binding	36
Table 65 GetLatestRevision PortType Operation	36
Table 66 EquipmentMetadataManager Binding	36
Table 67 Translation Table for Output Arguments for the GetLatestRevision Operation	37
Table 68 NotifyOnRevisions Operation Binding	38
Table 69 NotifyOnRevisions PortType Operation	38
Table 70 EquipmentMetadataManager Binding	38
Table 71 Translation Table for Input Arguments for the NotifyOnRevisions Operation	38
Table 72 Translation Table for Output Arguments for the GetUnits Operation	39
Table 73 XML Schema	39
Table 74 MetadataClient PortType Definitions	39
Table 75 MetadataClient Binding Definitions	40
Table 76 MetadataClient Port Type	40
Table 77 MetadataClient Binding	40
Table 78 MetadataRevised Operation Binding	41
Table 79 MetadataRevised PortType Operation	41
Table 80 MetadataRevised Binding	41
Table 81 Translation Table for Arguments for the MetadataReviseNotification Operation	42

1 Purpose

1.1 SOAP Implementation Mapping

1.1.1 The purpose of this Specification is to provide an implementation mapping of the SEMI E125 Specification to the SOAP 1.2 protocol. This Specification provides a description of the XML Schema data types used to support the UML classes defined in SEMI E125, and also describes the WSDL port type and binding definitions used to support the operations defined by the interfaces specified in SEMI E125.

2 Scope

2.1 The scope of this Specification is the representation of the EqSD model in an XML Schema and corresponding WSDL port types and bindings. It will not add new domain information or concepts to the SEMI E125 model. The only additions made are those needed to render useful WSDL or XML Schema.

2.2 This Specification requires SEMI E132 authentication, and defines requirements for the application of SEMI E132 concepts to the SEMI E125 Specification.

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

3 Limitations

3.1 None.

4 Referenced Standards and Documents

4.1 SEMI Standards and Safety Guidelines

SEMI E120.1 — XML Schema for the Common Equipment Model

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

SEMI E125 — Specification for Equipment Self Description (EqSD)

SEMI E128 — Specification for XML Messaging

SEMI E132.1 — Specification for SOAP Binding of Equipment Client Authentication and Authorization

SEMI E138 — XML Semiconductor Common Components

SEMI E145 — Classification for Measurement Unit Symbols in XML

4.2 OMG Standards¹

OMG Specification 01-09-67 — Unified Modeling Language (UML) Specification, Version 1.4²

4.3 W3C Standards³

Extensible Markup Language (XML) 1.0 (Fourth Edition)⁴

Namespaces in XML (Second Edition)⁵

XML Schema Part 0: Primer (Second Edition)⁶

XML Schema Part 1: Structures (Second Edition)⁷

¹ Object Management Group, Inc., 250 First Ave. Suite 100, Needham, MA 02494, USA. Telephone: +1.781.444.0404; Fax: +1.781.444.0320; <http://www.omg.org>

² http://www.omg.org/technology/documents/modeling_spec_catalog.htm

³ World Wide Consortium, Massachusetts Institute of Technology (MIT) Computer Science and Artificial Intelligence Laboratory (CSAIL), 32 Vassar Street Room 32-G515, Cambridge, MA 02139, USA. Telephone: +1.617.253.2613; Fax: +1.617.258.5999; <http://www.w3c.org>

⁴ W3C 16 August 2006; <http://www.w3.org/TR/2006/REC-xml-20060816/>

⁵ W3C 16 August 2006; <http://www.w3.org/TR/2006/REC-xml-names-20060816/>

⁶ W3C, 28 October 2004; <http://www.w3.org/TR/2004/REC-xmlschema-0-20041028/>

⁷ W3C, 28 October 2004; <http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/>

XML Schema Part 2: Datatypes (Second Edition)⁸

SOAP Version 1.2 Part 1: Messaging Framework (Second Edition)⁹

SOAP Version 1.2 Part 2: Adjuncts (Second Edition)¹⁰

Web Services Description Language (WSDL) 1.1¹¹

4.4 Other Documents

RFC 2616 — Hypertext Transfer Protocol – HTTP/1.1¹²

Secure Sockets Layer (SSL) Protocol 3.0¹³

Web Services Interoperability (WS-I): Basic Profile Version 1.0a¹⁴

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 *Unified Modeling Language (UML)* — 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 *eXtensible Markup Language (XML)* — 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 (refer to ¶¶ 4.2 and 4.3). This Specification does not provide tutorial information on these subjects.

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

6.1.3 Some of the key guidelines followed in this Specification 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).

⁸ W3C, 28 October 2004; <http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/>

⁹ W3C, 27 April 2007; <http://www.w3.org/TR/2007/REC-soap12-part1-20070427/>

¹⁰ W3C, 27 April 2007; <http://www.w3.org/TR/2007/REC-soap12-part2-20070427/>

¹¹ W3C Note; <http://www.w3.org/TR/wsdl>

¹² <http://www.ietf.org/rfc/rfc2616.txt>

¹³ <http://wp.netscape.com/eng/ssl3/draft302.txt>

¹⁴ <http://www.ws-i.org/Profiles/Basic/2003-08/BasicProfile-1.0a.html>

- 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 (e.g., 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 Complementary 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>Complementary 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 Diagrams¹⁵

6.3.1 This Specification 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 Specification 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 Specification.

6.3.2 In the diagram, rectangular boxes represent XML element definitions. The owner elements are on the left side while the elements contained within the owner element are on the right side. 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)
	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 Child1 and 2Type by adding a sequence that includes Child3 and Child4.

6.3.6 Reading Figure 1 would yield the following additional information:

1. Child1and 2Type 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.

¹⁵ All images/graphics were created using Altova’s XMLSPY®. Copyright 2003 Altova GmbH and reprinted with permission of Altova.

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.

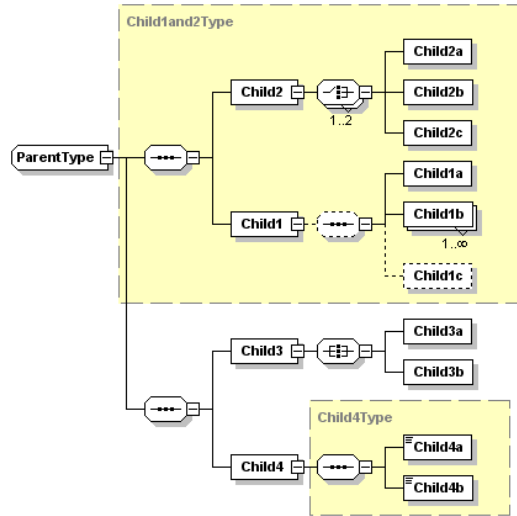


Figure 1
XML Schema Example Diagram

```
<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: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 → 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 Content-Type header 'action' 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 1.2 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 3 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 4 shows the specific XML Schema and WSDL files defined for SEMI E125.

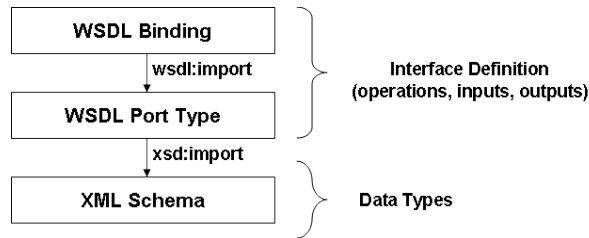


Figure 3
XML Schema and WSDL File Organization

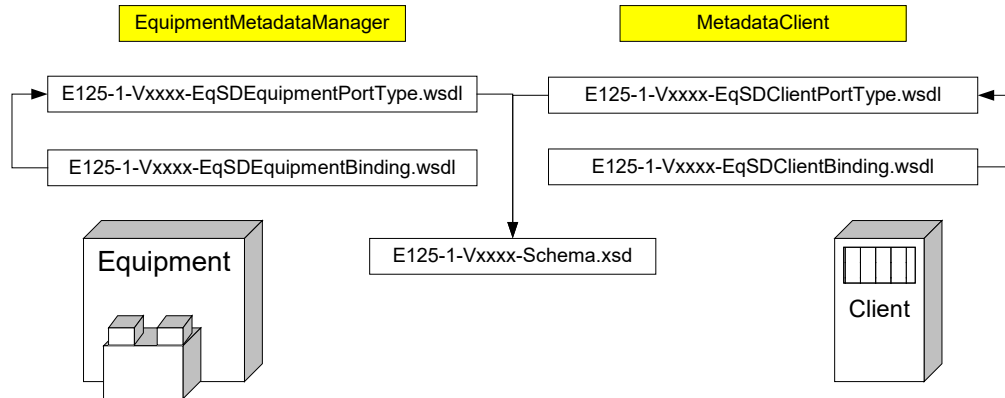


Figure 4
WSDL and XML Schema Files for SEMI E125

7.2 *EquipmentMetadataManager*

7.2.1 *Application of SEMI E132 Sessions*

7.2.1.1 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.

7.2.1.2 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 E128 session identifier in the SOAP envelope header provided with the request. The SEMI E128 session identifier shall be provided via the SEMI E128 SyncHeader element, as specified by SEMI E128. 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 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 The XML Schema and WSDL defined by this Specification for the EquipmentMetadataManager interface are contained in the following Complementary Files:

Table 9 XML Schema

<i>Complementary File Name</i>	E125-1-V0710-Schema.xsd
<i>Target Namespace</i>	urn:semi-org:xsd.E125-1.V0710.esd
<i>Imported/Referenced Namespaces</i>	urn:semi-org:xsd.E120-1.V0310.CommonEquipmentModel urn:semi-org:xsd.E138.V0709.ccs http://www.w3.org/2001/XMLSchema
<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>Complementary File Name</i>	E125-1-V0710-EqSDEquipmentPortType.wsdl
<i>Target Namespace</i>	urn:semi-org:ws.E125-1.V0710.EqSDEquipmentPortType
<i>Imported/Referenced Namespaces</i>	urn:semi-org:xsd.E128.V0706.xms urn:semi-org:xsd.E125-1.V0710.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 EquipmentMetadataManager interface, based on the data types defined in the SEMI E125 XML Schema.

Table 11 EquipmentMetadataManager Binding Definitions

<i>Complementary File Name</i>	E125-1-V0710-EqSDEquipmentBinding.wsdl
<i>Target Namespace</i>	urn:semi-org:ws.E125-1.V0710.EqSDEquipmentBinding
<i>Imported/Referenced Namespaces</i>	urn:semi-org:ws.E125-1.V0710.EqSDEquipmentPortType http://schemas.xmlsoap.org/wsdl/soap12/ 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.

7.2.4.2 These Complementary Files are an official part of and accompany this Specification. The contents of these documents constitute the core part of this Specification.

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 5 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.

7.2.5.2 Detail for the port type operations for the EquipmentMetadataManager are defined in the wsdl files listed in Table 12.

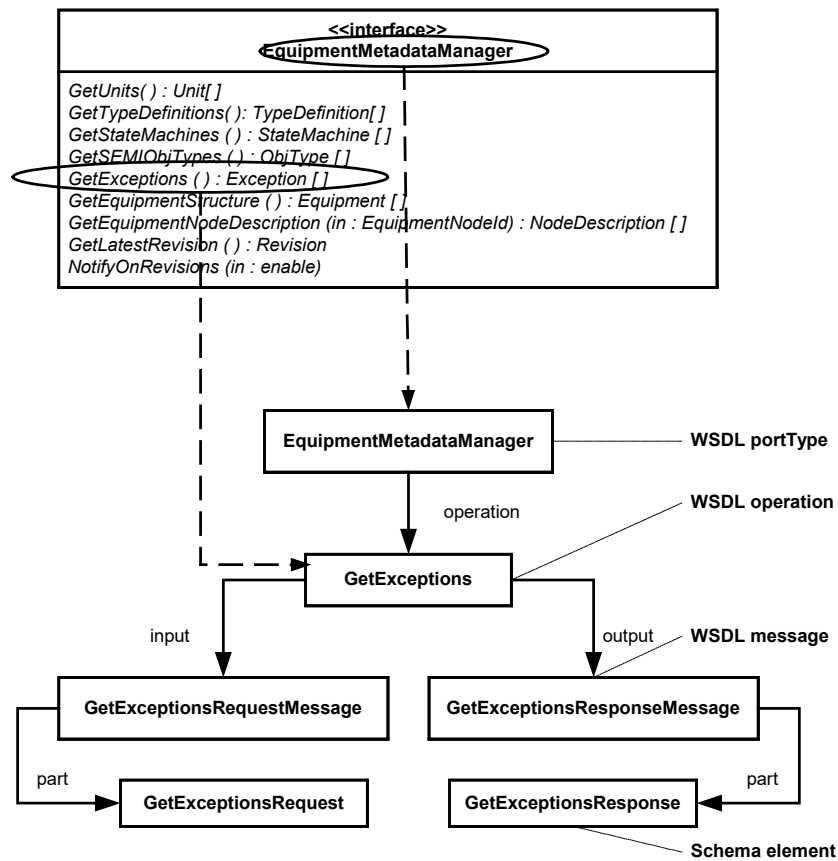


Figure 5
Mapping the EquipmentMetadataManager to a WSDL portType

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), as well as 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 Content-Type header ‘action’ 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.

7.2.6.2 Details for the binding operation for the EquipmentMetadataManager are defined by the wsdl file in Table 11. The SOAP action used in all EquipmentMetadataManager operations uses the binding target namespace from Table 11 followed by a colon and the operation name (i.e., soapAction = <binding target namespace>:<operation-name>).

soapAction = ‘urn:semi-org:ws.E125.V0710.EqSDEquipmentBinding:<operation-name>’

Table 13 EquipmentMetadataManager Binding

<i>SEMI 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 WSDL Operation Binding

7.2.8.1.1 See the operation binding for ‘GetUnits’ in the binding wsdl file. Key information for the binding operation is shown for convenience in the following table.

Table 14 GetUnits Operation Binding

<i>SOAPAction</i>	<binding target namespace>:GetUnits
<i>Input Headers (WSDL Message, Required)</i>	SyncHeaderMessage, required
<i>Output Headers (WSDL Message, Required)</i>	SyncHeaderMessage, required

7.2.8.2 WSDL PortType Operation

7.2.8.2.1 See the port type operation definition for ‘GetUnits’ in the port type wsdl file. Key information for the port type operation 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

7.2.8.3 WSDL Message(s)

7.2.8.3.1 See the message definitions for ‘SyncHeaderMessage’, ‘GetUnitsRequestMessage’, and ‘GetUnitsResponseMessage’ in the corresponding port type wsdl file. Key information is shown for convenience in the following table.

Table 16 EquipmentMetadataManager Binding

<i>Message Name → Schema Element Name</i>	SyncHeaderMessage → xms:SyncHeader
	GetUnitsRequestMessage → esd:GetUnitsRequest
	GetUnitsResponseMessage → esd:GetUnitsResponse

7.2.8.4 XML Schema Types

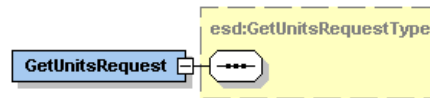


Figure 6
GetUnitsRequest Global Element

7.2.8.4.1 *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.

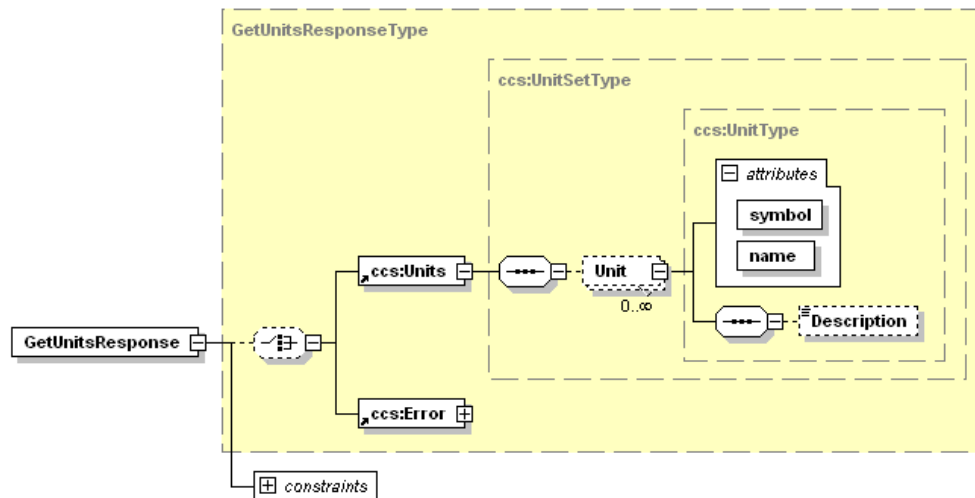


Figure 7
GetUnitsResponse Global Element

7.2.8.4.2 *GetUnitsResponse* — This global element acts as a top-level container entity representing the SOAP body of the response to a SEMI E125 GetUnits request. It contains either zero or more elements named ‘Units’ of complexType ‘UnitSetType’ that represent a collection of ‘UnitType’ or an element named ‘Error’ of complexType ‘ErrorType’, as 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 UnitSetType as defined in SEMI E138.	element	Units: ccs:UnitSetType
error	Structured data, of type Error, defined in the SEMI E138.	element	Error: ccs:ErrorType