SEMI E134.1-0310
SPECIFICATION FOR SOAP BINDING OF DATA COLLECTION MANAGEMENT (DCM)

This standard was technically approved by the global Information & Control Committee. This edition was approved for publication by the global Audits & Reviews Subcommittee on December 16, 2009. Initially available at www.semi.org in February 2010. Originally published March 2005; previously published July 2009.

SUPERSEDED

Table of Contents

1 Purpose 5
1.1 SOAP Implementation Mapping 5
2 Scope 5
3 Limitations 5
4 Referenced Standards and Documents 5
4.1 *SEMI Standards* 5
4.2 *OMG Standards* 5
4.3 *W3C Standards* 5
4.4 *Other Standards* 6
5 Terminology 6
5.1 *Abbreviations and Acronyms* 6
5.2 *Definitions And Acronyms* 6
6 Conventions 6
6.1 *Translating UML to XML Schema* 6
6.2 *Documenting XML Schema and WSDL Files* 7
6.3 *Documenting XML Schema With Diagrams* 8
6.4 *XML Schema Sample* 9
6.5 *Translating UML to WSDL* 10
7 Mapping of E134 UML to XML Schema and WSDL 11
7.1 *WSDL Organization* 11
7.2 *DataCollectionManager* 12
7.3 *DCPConsumer* 50
8.1 Message Transport 65

List of Figures

Figure 1 XML Schema Example Diagram 9
Figure 2 XML for Sample 10
Figure 3 XML Schema and WSDL File Organization 11
Figure 4 WSDL and XML Schema Files for SEMI E134 11
Figure 5 Mapping the DataCollectionManager to a WSDL portType 17
Figure 6 DefinePlanRequest 18
Figure 7 DataCollectionPlanType 20
Figure 8 EventRequestType 20
Figure 9 TraceRequestType 22
Figure 10 DefinePlanResponseType 24
Figure 11 InvalidPlanType 25
Figure 12 InvalidEventRequestType 26
Figure 13 InvalidTraceRequestType 27
Figure 14 GetDefinedPlanIdsRequest 29
Figure 15 GetDefinedPlanIdsResponse 29
Figure 16 GetPlanDefinitionRequest 31
Figure 17 GetPlanDefinitionResponse 31
Figure 18 ActivatePlanRequest 33
Figure 19 ActivatePlanResponse 33

Documents Document Sharing Hub
Full document details: SEMI E134.1-0310

Figure 20 GetActivePlanIdsRequest....35
Figure 21 GetActivePlanIdsResponse....35
Figure 22 DeactivatePlanRequest....37
Figure 23 DeactivatePlanResponse....37
Figure 24 DeletePlanRequest....39
Figure 25 DeletePlanResponse....40
Figure 26 GetParameterValuesRequest....41
Figure 27 GetParameterValuesResponse....42
Figure 28 GetObjTypeInstanceIdsRequest....43
Figure 29 GetObjTypeInstanceIdsResponse....44
Figure 30 GetCurrentPerformanceStatusRequest....46
Figure 31 GetCurrentPerformanceStatusResponse....47
Figure 32 GetCurrentDateTimeRequest....49
Figure 33 GetCurrentDateTimeResponse....50
Figure 34 NewDataNotification....53
Figure 35 EventReportType....54
Figure 36 ExceptionReportType....55
Figure 37 CollectedDataType....56
Figure 38 PerformanceWarningNotification....58
Figure 39 PerformanceRestoredNotification....59
Figure 40 DCPDeactivationNotification....61
Figure 41 DCPHibernationNotification....62
Figure 42 DCPDefinedNotification....63
Figure 43 DCPDeletedNotification....65

List of Tables

Table 1 Example Translation Table....7
Table 2 Example Translation Table for Operation Input/Output Arguments....7
Table 3 Example Schema/WSDL Document Description Table....7
Table 4 Altova XMLSPY Schema Diagram Symbols....8
Table 5 Example Interface WSDL Port Type Table....10
Table 6 Example Interface WSDL Binding Table....10
Table 7 Example Operation Binding Table....11
Table 8 Example PortType Operation Table....11
Table 9 SEMI E134 to SEMI E132 Privilege Mapping....13
Table 10 ObjType Event/Exception/Parameter Naming Convention....13
Table 11 ObjType Event Request Example....14
Table 12 SEMI E134 Error Codes....15
Table 13 XML Schema....15
Table 14 DataCollectionManager PortType Definitions....16
Table 15 DataCollectionManager Binding Definitions....16
Table 16 DefinePlan Operation Binding....18
Table 17 DefinePlan PortType Operation....18
Table 18 DataCollectionManagement Binding....18
Table 19 Translation Table for Input Arguments for the DefinePlan Operation....19
Table 20 Translation Table for DataCollectionPlanType....19
Table 21 Translation Table for EventRequestType....20
Table 22 Translation Table for ParameterRequestType....21
Table 23 Translation Table for ExceptionRequestType....21
Table 24 Translation Table for TraceRequestType....21
Table 25 Translation Table for EventTriggerType....22
Table 26 Translation Table for ExceptionTriggerType....23
Table 27 Translation Table for ParameterRequestType....23
Table 28 Translation Table for Output Arguments for the DefinePlan Operation....23
Table 29 Translation Table for DCPDefinedType....24
Table 30 Translation Table for InvalidPlanType....25

SUPERSEDED

Documents
Document Sharing Hub
Full document details: SEMI E134.1-0310

Table 31 Translation Table for InvalidEventRequestType 26
Table 32 Translation Table for InvalidExceptionRequestType 26
Table 33 Translation Table for InvalidParameterRequestType 27
Table 34 Translation Table for InvalidTraceRequestType 27
Table 35 Translation Table for InvalidTriggerType 28
Table 36 Translation Table for InvalidIntervalType 28
Table 37 Translation Table for InvalidCycleType 28
Table 38 GetDefinedPlanIds Operation Binding 28
Table 39 GetDefinedPlanIds PortType Operation 29
Table 40 DataCollectionManagement Binding 29
Table 41 Translation Table for Output Arguments for the GetDefinedPlanIds Operation 30
Table 42 GetPlanDefinition Operation Binding 30
Table 43 GetPlanDefinition PortType Operation 30
Table 44 DataCollectionManagement Binding 30
Table 45 Translation Table for Input Arguments for the GetPlanDefinitionOperation 31
Table 46 Translation Table for Output Arguments for the GetPlanDefinitionOperation 32
Table 47 Translation Table for NoSuchPlanType 32
Table 48 ActivatePlan Operation Binding 32
Table 49 ActivatePlan PortType Operation 32
Table 50 DataCollectionManagement Binding 33
Table 51 Translation Table for Input Arguments for the ActivatePlanOperation 33
Table 52 Translation Table for Output Arguments for the GetDefinedPlanIds Operation 34
Table 53 Translation Table for DCPActivatedType 34
Table 54 GetActivePlanIds Operation Binding 34
Table 55 GetActivePlanIds PortType Operation 35
Table 56 DataCollectionManagement Binding 35
Table 57 Translation Table for Output Arguments for the ActivatePlan Operation 36
Table 58 DeactivatePlan Operation Binding 36
Table 59 DeactivatePlan PortType Operation 36
Table 60 DataCollectionManagement Binding 36
Table 61 Translation Table for Input Arguments for the DeactivatePlan Operation 37
Table 62 Translation Table for Output Arguments for the DeactivatePlan Operation 38
Table 63 Translation Table for DeactivatePlanResponseType 38
Table 64 Translation Table for DCPNotActiveType 38
Table 65 DeletePlan Operation Binding 38
Table 66 DeletePlan PortType Operation 39
Table 67 DataCollectionManagement Binding 39
Table 68 Translation Table for Input Arguments for the DeletePlan Operation 39
Table 69 Translation Table for Output Arguments for the DeletePlanOperation 40
Table 70 Translation Table for DCPDeletedType 40
Table 71 GetParameterValues Operation Binding 41
Table 72 GetParameterValues PortType Operation 41
Table 73 DataCollectionManagement Binding 41
Table 74 Translation Table for Input Arguments for the GetParameterValues Operation 42
Table 75 Translation Table for Output Arguments for the GetParameterValues Operation 42
Table 76 GetObjTypeInstanceIds Operation Binding 43
Table 77 GetObjTypeInstanceIds PortType Operation 43
Table 78 DataCollectionManagement Binding 43
Table 79 Translation Table for Input Arguments for the GetObjTypeInstanceIds Operation 44
Table 80 Translation Table for ObjTypeRequestType 44
Table 81 Translation Table for Output Arguments for the GetObjTypeInstanceIds Operation 44
Table 82 Translation Table for ObjTypeResultType 45
Table 83 Translation Table for ObjTypeInstanceType 45
Table 84 Translation Table for ObjTypeRequestErrorType 45
Table 85 GetCurrentPerformanceStatus Operation Binding 45
Table 86 GetCurrentPerformanceStatus PortType Operation 46

SUPERSEDED

Documents Document Sharing Hub
Full document details: SEMI E134.1-0310

Table 87 DataCollectionManagement Binding 46
Table 88 Translation Table for Output Arguments for the GetCurrentPerformanceStatus Operation 47
Table 89 Translation Table for PerformanceStatusType 48
Table 90 Translation Table for ActiveJobType 48
Table 91 Supported Job Type Table for JobTypeEnumType 48
Table 92 Translation Table for AffectedSystemType 48
Table 93 GetCurrentDateTime Operation Binding 49
Table 94 GetCurrentDateTime PortType Operation 49
Table 95 DataCollectionManagement Binding 49
Table 96 Translation Table for Output Arguments for the GetCurrentTime Operation 50
Table 97 XML Schema 50
Table 98 DCPConsumer PortType Definitions 51
Table 99 DCPConsumer Binding Definitions 51
Table 100 NewData Operation Binding 52
Table 101 NewData PortType Operation 52
Table 102 DCPConsumer Binding 52
Table 103 Translation Table for Input Arguments for the NewDataNotification Operation 53
Table 104 Translation Table for DataCollectionReportType 54
Table 105 Translation Table for EventReportType 54
Table 106 Translation Table for ExceptionReportType 55
Table 107 Translation Table for TraceReportType 56
Table 108 Translation Table for CollectedDataType 56
Table 109 PerformanceWarning Operation Binding 57
Table 110 PerformanceWarning PortType Operation 57
Table 111 DCPConsumer Binding 57
Table 112 Translation Table for Input Arguments for the PerformanceWarning Operation 58
Table 113 PerformanceRestored Operation Binding 58
Table 114 PerformanceRestored PortType Operation 59
Table 115 DCPConsumer Binding 59
Table 116 Translation Table for Input Arguments for the PerformanceRestored Operation 60
Table 117 DCPDeactivation Operation Binding 60
Table 118 DCPDeactivation PortType Operation 60
Table 119 DCPConsumer Binding 60
Table 120 Translation Table for Input Arguments for the DCPDeactivation Operation 61
Table 121 DCPHibernation Operation Binding 61
Table 122 DCPHibernation PortType Operation 61
Table 123 DCPConsumer Binding 62
Table 124 Translation Table for Input Arguments for the DCPHibernation Operation 62
Table 125 Translation Table for DCPHibernatedType 62
Table 126 DCPDefined Operation Binding 63
Table 127 DCPDefined PortType Operation 63
Table 128 DCPConsumer Binding 63
Table 129 Translation Table for Input Arguments for the DCPDefined Operation 64
Table 130 DCPDeleted Operation Binding 64
Table 131 DCPDeleted PortType Operation 64
Table 132 DCPConsumer Binding 64
Table 133 Translation Table for Input Arguments for the DCPDeleted Operation 65

SUPERSEDED

Documents
Document Sharing Hub
Full document details: SEMI E134.1-0310

1 Purpose

1.1 *SOAP Implementation Mapping*

1.1.1 The purpose of this specification is to provide an implementation mapping of the SEMI E134 specification for Data Collection Management to the SOAP 1.2 protocol. This document provides a description of the XML Schema data types used to support the UML classes defined in SEMI E134, and also describes the WSDL port type and binding definitions used to support the operations defined by the interfaces specified in SEMI E134.

2 Scope

2.1 The scope of this specification is the faithful representation of the DCM model in an XML Schema and corresponding WSDL port types and bindings. It will not add new domain information or concepts to the SEMI E134 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 E134 specification.

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 None.

4 Referenced Standards and Documents

4.1 *SEMI Standards*

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

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

SEMI E125.1 — Specification for SOAP Implementation of Equipment Self Description

SEMI E128 — Specification for XML Messaging

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

SEMI E138 — XML Semiconductor Common Components

4.2 *OMG Standards*[1]

OMG 01-09-67 — Unified Modeling Language, v1.4

4.3 *W3C Standards*[2]

Extensible Markup Language (XML) 1.0 (Fourth Edition) — W3C 16 August 2006; http://www.w3.org/TR/2006/REC-xml-20060816/

Namespaces in XML (Second Edition) — W3C 16 August 2006; http://www.w3.org/TR/2006/REC-xml-names-20060816/

XML Schema Part 0: Primer (Second Edition) — W3C, 28 October 2004; http://www.w3.org/TR/2004/REC-xmlschema-0-20041028/

XML Schema Part 1: Structures (Second Edition) — W3C, 28 October 2004; http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/

1 Object Management Group, Inc., 250 First Ave. Suite 100, Needham, MA 02494, U.S.A. Telephone: 781.444.0404; Fax:781.444.0320; http://www.omg.org

2 World Wide Web Consortium, Massachusetts Institute of Technology (MIT), Computer Science and Artificial Intelligence Laboratory (CSAIL), 32 Vassar Street Room 32-G515, Cambridge, MA 02139, USA. Telephone: 617.253.2613; Fax: 617.258.5999; http://www.w3.org

SUPERSEDED

Documents Document Sharing Hub
Full document details: SEMI E134.1-0310

XML Schema Part 2: Datatypes (Second Edition) — W3C, 28 October 2004; http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/

SOAP Version 1.2 Part 1: Messaging Framework (Second Edition) — W3C, 27 April 2007; http://www.w3.org/TR/2007/REC-soap12-part1-20070427/

SOAP Version 1.2 Part 2: Adjuncts (Second Edition) — W3C, 27 April 2007; http://www.w3.org/TR/2007/REC-soap12-part2-20070427/

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

4.4 *Other Standards*

Internet Engineering Task Force (IETF) RFC 2616 — Hypertext Transfer Protocol – HTTP/1.1; http://www.ietf.org/rfc/rfc2616.txt

Secure Sockets Layer (SSL); http://wp.netscape.com/eng/ssl3/draft302.txt

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

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, 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 modeled as contained elements or arrays of references to one or more unique attributes of the target type.

SUPERSEDED

Documents
Document Sharing Hub
Full document details: SEMI E134.1-0310

- Inheritance is mapped to 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 place 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., xs:string), or a named type defined within the XML Schema.

Table 1 Example Translation Table

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

6.1.6 *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

Argument Name	*Format*	*XML Element or Attribute*	*XML Name/Type*
<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 Common 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

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

SUPERSEDED

Documents Document Sharing Hub
Full document details: SEMI E134.1-0310

6.3 *Documenting XML Schema with Diagrams*

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 Table 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. 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 Child1and2Type 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.

SUPERSEDED

Documents
Document Sharing Hub
Full document details: SEMI E134.1-0310

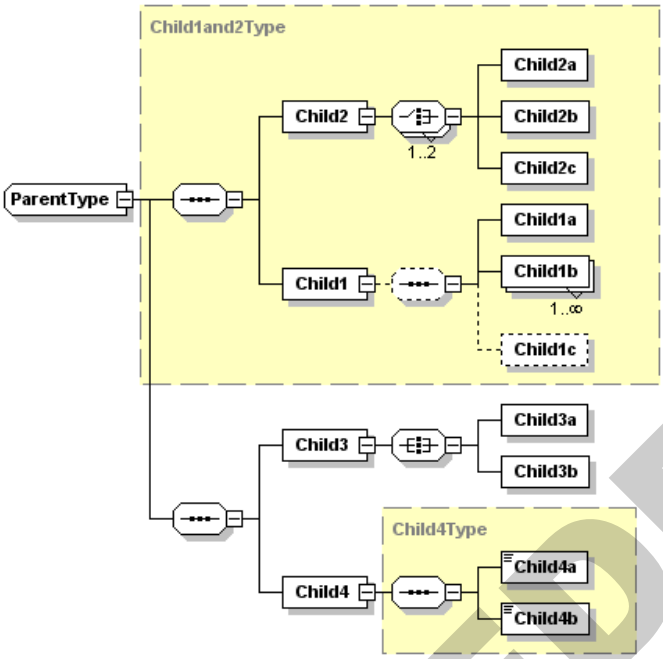

Figure 1
XML Schema Example Diagram

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.

SUPERSEDED

Documents Document Sharing Hub
Full document details: SEMI E134.1-0310

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

Figure 2
XML for Sample

SUPERSEDED

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

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

Table 6 Example Interface WSDL Binding Table

SEMI E125 Class Name	<UML interface name from abstract specification>
WSDL Binding Name	<binding name used in WSDL>
SOAP Binding Style	<RPC or document>
SOAP Transport	<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.

Documents
Document Sharing Hub
Full document details: SEMI E134.1-0310

Table 7 Example Operation Binding Table

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

Table 8 Example PortType Operation Table

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

7 Mapping of SEMI E134 UML to XML Schema and WSDL

7.1 *WSDL Organization*

7.1.1 Each interface definition in SEMI E134 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 E134. 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 E134 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 E134.

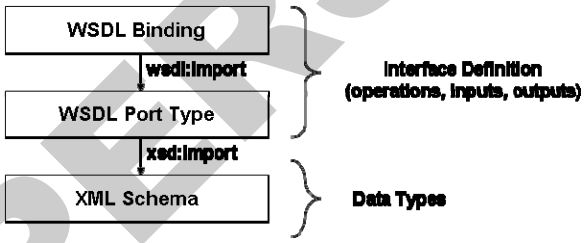

Figure 3
XML Schema and WSDL File Organization

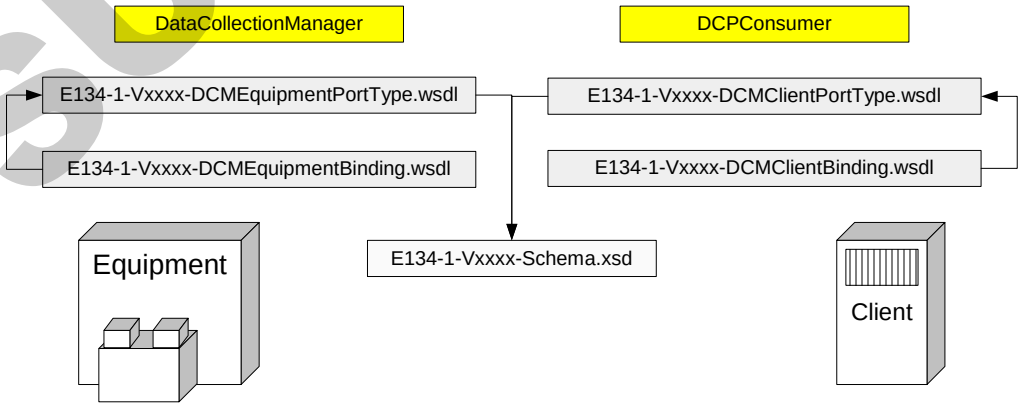

Figure 4
WSDL and XML Schema Files for SEMI E134

SUPERSEDED
Documents Document Sharing Hub
Full document details: SEMI E134.1-0310

7.2 *DataCollectionManager*

7.2.1 *Application of SEMI E132 Sessions*

7.2.1.1 The DataCollectionManager shall use SEMI E132.1 for authentication and authorization.

7.2.1.2 Use of the interfaces defined in SEMI E134 to collect data can directly affect overall data throughput for the equipment, and may indirectly affect the equipment's ability to perform its intended function. Additionally, data accessible from the equipment via the SEMI E134 interfaces may be proprietary to the equipment supplier and/or the factory that uses the equipment, and should be protected against unauthorized access within the factory.

7.2.1.3 For this reason, all operations defined by the SEMI E134 DataCollectionManager 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 E134 DataCollectionManager 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 E134 DataCollectionManager 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 E134 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 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 an SEMI E134 request if the request is found to include an invalid SEMI E128 session identifier, or if no session identifier is provided.

7.2.3 *Session and DCP State Interaction*

7.2.3.1 The equipment shall maintain DCP activity on a per-Session basis. If two separate SEMI E132 sessions attempt to activate the same DCP, the equipment shall activate the DCP for both sessions. Similarly if one active SEMI E132 session attempts to activate the same DCP twice, the equipment shall respond to the second request with the DCPIsActive error, as defined in SEMI E134.

7.2.3.2 As described in ¶ 7.2.1.2, the equipment shall use the URL provided with the HTTPEndpoint information provided with the SEMI E132 EstablishSession request to send all SEMI E134 communication.

7.2.3.3 If a consumer's SEMI E132.1 Session is closed for any reason (either expressly closed or through a communication failure), the equipment shall stop sending data from all DCP's previously activated by that consumer. This shall not affect data transmission from those DCP's to consumers that still have SEMI E132 Sessions in the Established state.

7.2.3.3.1 If one or more of these DCP's has more than one consumer, the equipment shall take the actions defined in transition 5 of the SEMI E134 DataCollectionPlan State model. These DCP's will remain active, and the equipment shall remove the consumer's session from the active consumers list and continue to send data to the other consumers for those DCP's.

7.2.3.3.2 If one or more of these DCP's has only one consumer, the equipment shall take the actions defined in transition 8 of the SEMI E134 DataCollectionPlan State model. The equipment shall remove the consumer's session from the active consumers list and change the state of these DCP's to Inactive.

7.2.3.4 If a DCP is returning from the Hibernating state and there is at least one consumer with an SEMI E132 persistent Session that had previously requested activation of the DCP, then the equipment shall transition the DCP to the Active state, and resume sending data to each consumer with an SEMI E132 persistent Session.

7.2.3.5 If a DCP is returning from the Hibernating state and there are no consumers with SEMI E132 persistent Sessions, then the equipment shall transition the DCP to the Inactive state.

SUPERSEDED

Documents
Document Sharing Hub
Full document details: SEMI E134.1-0310

7.2.4 *Authentication and Authorization Using SEMI E132.1*

7.2.4.1 The equipment shall implement the SEMI E134 privilege levels as SEMI E132 privileges.

7.2.4.2 The equipment shall implement the SEMI E134 privilege levels as specified in Table 9. The privileges listed are of the format urn:semi-org:priv.<priv-name>.

Table 9 SEMI E134 to SEMI E132 Privilege Mapping

SEMI E134 Privilege Level	*SEMI E132.1 PrivilegeId Entry*
ManageOnlyAuthoredDCPs	urn:semi-org:priv.ManageOnlyAuthoredDCPs
UseAnyDCP	urn:semi-org:priv.UseAnyDCP
ManageAnyDCP	urn:semi-org:priv.ManageAnyDCP

7.2.5 *Using SEMI E125 Metadata to Identify Sources, Events, Exceptions, and Parameters*

7.2.5.1 Sources shall be named using a valid SEMI E120 Locator value and the SEMI E120.1 LocatorType.

7.2.5.2 SEMI E125 Events are identified by using the value of the SEMI E125 Event "id" attribute as the value of the SEMI E134 EventRequest "eventId" attribute included in an SEMI E134 DataCollectionPlan.

7.2.5.3 SEMI E125 Exceptions are identified by using the value of the SEMI E125 Exception "id" attribute as the value of the SEMI E134 ExceptionRequest "exceptionId" attribute included in an SEMI E134 DataCollectionPlan.

7.2.5.4 SEMI E125 Parameters are identified by using the value of the SEMI E125 Parameter "name" attribute as the value of the SEMI E134 ParameterRequest "parameterName" attribute included in an SEMI E134 DataCollectionPlan.

7.2.5.5 If the events, exceptions, or parameters are defined by an SEMI E39-compliant objType, then use the following naming conventions to define their source. These definitions distinguish between information from any instance vs. a specific instance.

Table 10 ObjType Event/Exception/Parameter Naming Convention

SEMI E125 Item	*Naming Convention*
ObjType Attribute (any instance)	objType:<typeName>:<parameter name>
ObjType Event (any instance)	objType:<typeName>:<eventId>
ObjType Exception (any instance)	objType:<typeName>:<exceptionId>
ObjType Attribute (specific instance)	objInst:<typeName>:<objId>: <parameterName>
ObjType Event (specific instance)	objInst:<typeName>:<objId>:<eventId>
ObjType Exception (specific instance)	objInst:<typeName>:<objId>:<exceptionId>

7.2.5.6 <typeName> and <objId> referenced in the above table shall be of the following format. "typeName" shall be the SEMI E39 objType name for the object type of interest. "objId" shall be the value of that instance's SEMI E39 objId attribute.

7.2.5.7 *Any ObjType Instance (see rows 1–3 of Table 10)* — This syntax is used when the DCP author does not or cannot know in advance the specific SEMI E39 object id's that produce the exception, event, or parameter (attribute) of interest. This is most often the case with "transient" SEMI E39 objects, or SEMI E39 objects that are dynamically created and destroyed while the equipment is operating. Examples of transient SEMI E39 objects in SEMI standards are the SEMI E40 PROCESSJOB ObjType, the SEMI E87 Carrier ObjType, etc.

7.2.5.7.1 A DCP that includes an Event/ExceptionRequest containing an event/exceptionId using the "any instance" syntax means that the equipment shall generate an Event/ExceptionReport whenever any SEMI E39 object of the specified ObjType generates the requested event or exception. If multiple instances of the same ObjType exist and

SUPERSEDED

Documents Document Sharing Hub
Full document details: SEMI E134.1-0310

each generate the requested event or exception, the equipment shall generate a separate Event/ExceptionReport for each SEMI E39 object.

7.2.5.7.2 An EventRequest that uses the "any instance" syntax may include ParameterRequests that use the "any instance" syntax to request attributes from SEMI E39 objects of the same ObjType. For such ParameterRequests, the corresponding ParameterValues provided in an EventReport for a specific SEMI E39 object must be from the same SEMI E39 object that generated the requested event. See Table 11 for examples.

7.2.5.7.3 An EventRequest that uses the "any instance" syntax for the eventId must not include ParameterRequests that use the "any instance" syntax for attributes from a different ObjType. An EventRequest using the "any instance" syntax must not include ParameterRequests that use the "specific instance" syntax (see ¶ 7.2.5.5). An EventRequest that does not use the "any instance" syntax for the eventId must not include ParameterRequests that use the "any instance" syntax at all. The equipment shall reject any DCP's containing EventRequests with any of these errors via the InvalidEventRequest type defined for the "DefinePlan" operation.

7.2.5.7.4 The equipment shall reject any GetParameterValues request that uses the "any instance" syntax for parameters. The equipment shall treat such request as though the parameter is not defined by returning "NoValue" with "NoSuchParameter" as the reason.

7.2.5.8 *Specific ObjType Instance (see rows 4–6 of Table 10)* — This syntax is used when the DCP author knows in advance the specific SEMI E39 object id's that produce the exception, event, or parameter (attribute) of interest. This is most often the case with "static" SEMI E39 objects, or SEMI E39 objects that are created once during equipment initialization and retained while the equipment is operating, potentially even across shutdowns. Examples of static SEMI E39 objects in SEMI standards are the SEMI E90 SubstLoc ObjType, the SEMI E116 EPTTracker ObjType, etc.

7.2.5.8.1 A DCP that includes an Event/ExceptionRequest containing an event/exceptionId using the "specific instance" syntax means that the equipment shall generate an Event/ExceptionReport whenever the specific SEMI E39 object (and only that object) generates the requested event or exception. If multiple instances of the same ObjType exist and each generate the requested event or exception, the equipment shall generate a Event/ExceptionReport only for the requested SEMI E39 object.

7.2.5.8.2 An EventRequest that uses the "specific instance" syntax may include ParameterRequests that use the "specific instance" syntax to request attributes from the same SEMI E39 object identified in the EventRequest. For such ParameterRequests, the corresponding ParameterValues provided in the EventReport for that object must be from the same SEMI E39 object that generated the requested event. See Table 11 for examples.

7.2.5.8.3 An EventRequest that uses the "specific instance" syntax for the eventId must not include ParameterRequests that use the "specific instance" syntax for attributes from a different ObjType or for a different instance of the same ObjType. An EventRequest using this syntax must not include ParameterRequests that use the "any instance" syntax. The equipment shall reject any DCP's containing EventRequests with any of these errors via the InvalidEventRequest type defined for the "DefinePlan" operation.

7.2.5.9 The below example is for two objects of type "PROCESSJOB" and two objects of type "SubstLoc". The "PROCESSJOB" objects have a Process Job State Transition #3 event (PJST3). The "SubstLoc" objects have a Substrate Location State Model Transition #1 event (SLSMT1). The first "PROCESSJOB" object has an object ID of "PJID1" and the second object has an object ID of "PJID2". The first "SubstLoc" object has an object ID of "Chamber1" and the second object has an object ID of "Chamber2".

Table 11 ObjType Event Request Example

Event Request	*Results*
Event: objType:PROCESSJOB:PJST3 Parameter: objType:PROCESSJOB:ObjID	The equipment sends an event report when the Process Job State Transition #3 event occurs on "PJID1" and when it occurs on "PJID2". The parameter value returned in the event report for "PJID1" is the value of "ObjID" for "PJID1". The parameter value returned in the event report for "PJID2" is the value of "ObjID" for "PJID2".

SUPERSEDED

Documents
Document Sharing Hub
Full document details: SEMI E134.1-0310

Event Request	*Results*
Event: objInst:SubstLoc:Chamber1:SLSMT1 Parameter: bjInst:SubstLoc:Chamber1:SubstID	The equipment sends an event report when the Substrate Location State Model Transition #1 occurs on "Chamber1". The parameter value returned in the event report is the value of "SubstID" for "Chamber1".
Event: objType:SubstLoc:SLSMT1 Parameter: objInst:SubstLoc:Chamber1:SubstID	This will result in an error since the event is for any instance of type "SubstLoc" while the parameter is for the specific instance "Chamber1".
Event: objInst:SubstLoc:Chamber1:SLSMT1 Parameter: objType:SubstLoc:SubstID	This will result in an error since the event is for the specific instance "Chamber1" while the parameter is for any instance of type "SubstLoc".
Event: objType:PROCESSJOB:PJST3 Parameter: objType:SubstLoc:SubstID	This will result in an error since the event is for any instance of type "PROCESSJOB" while the parameter is for any instance of type "SubstLoc". The event and parameter must use the same objType.

7.2.6 *Use of Common Error Type*

7.2.6.1 Table 12 below provides a mapping of the SEMI E134 defined errors to its common error code. The SEMI E134 mapped error codes below are in addition to any common error codes defined in the Common Components Specification. SEMI E134 Error codes shall be denoted in the common error type structure by defining the "source" name of "urn:semi-org:E134".

7.2.6.2 SEMI E134.1 implements the SEMI E134 Unauthorized Operation Class as the SEMI E132.1 UnauthorizedOperation class and will return the SEMI E132.1 error code "6005" and a source of "urn:semi-org:E132" Operation not authorized in the Common Components Error structure.

Table 12 SEMI E134 Error Codes

SEMI E134 Error	*Common Error Code*
Operation not authorized.	6000 (as defined in SEMI E132.1)
InvalidPlan.	8000
NoSuchPlan.	8001
DCPIsActive.	8002
DCPNotActive.	8003

7.2.7 *XML Schema and WSDL Files*

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

Table 13 XML Schema

File Name	E134-1-V0310-Schema.xsd
Target Namespace	urn:semi-org:xsd.E134.V0310.DCM
Imported/Referenced Namespaces	urn:semi-org:xsd.E138.V0709.ccs urn:semi-org:xsd.E132-1.V0310.auth http://www.w3.org/2001/XMLSchema
Description	This file defines all of the SEMI E134 data types and elements used by both the DataCollectionManager and DCPConsumer interfaces.

SUPERSEDED

Documents Document Sharing Hub
Full document details: SEMI E134.1-0310

Table 14 DataCollectionManager PortType Definitions

File Name	E134-1-V0310-DCMEquipmentPortType.wsdl
Target Namespace	urn:semi-org:ws.E134.V0310.DCMEquipmentPortType
Imported/Referenced Namespaces	urn:semi-org:xsd.E128.V0706.xms urn:semi-org:xsd.E134.V0310.DCM http://schemas.xmlsoap.org/wsdl/ http://www.w3.org/2001/XMLSchema
Description	This file defines all of the input/output messages and operations for the DataCollectionManager interface, based on the data types defined in the SEMI E134 XML Schema.

Table 15 DataCollectionManager Binding Definitions

File Name	E134-1-V0310-DCMEquipmentBinding.wsdl
Target Namespace	urn:semi-org:ws.E134.V0310.DCMEquipmentBinding
Imported/Referenced Namespaces	urn:semi-org:ws.E134.V0310.DCMEquipmentPortType http://schemas.xmlsoap.org/wsdl/soap12/ http://schemas.xmlsoap.org/wsdl/
Description	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.

7.2.7.2 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.

7.2.8 *WSDL Port Type Overview*

7.2.8.1 The DataCollectionManager WSDL portType definition organizes the SEMI E134.1 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 DataCollectionManager interface in SEMI E134, ¶ 9.1. The WSDL portType itself is named after the SEMI E134 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 E134. Figure 5 illustrates the relationship between the SEMI E134 UML interface and the WSDL portType definition. The WSDL message and operation definitions are described in ¶¶ 7.2.10–7.2.19.

7.2.8.2 Details for the port type operations for the DataCollectionManager are defined by the wsdl file in Table 14.

SUPERSEDED

Documents
Document Sharing Hub
Full document details: SEMI E134.1-0310

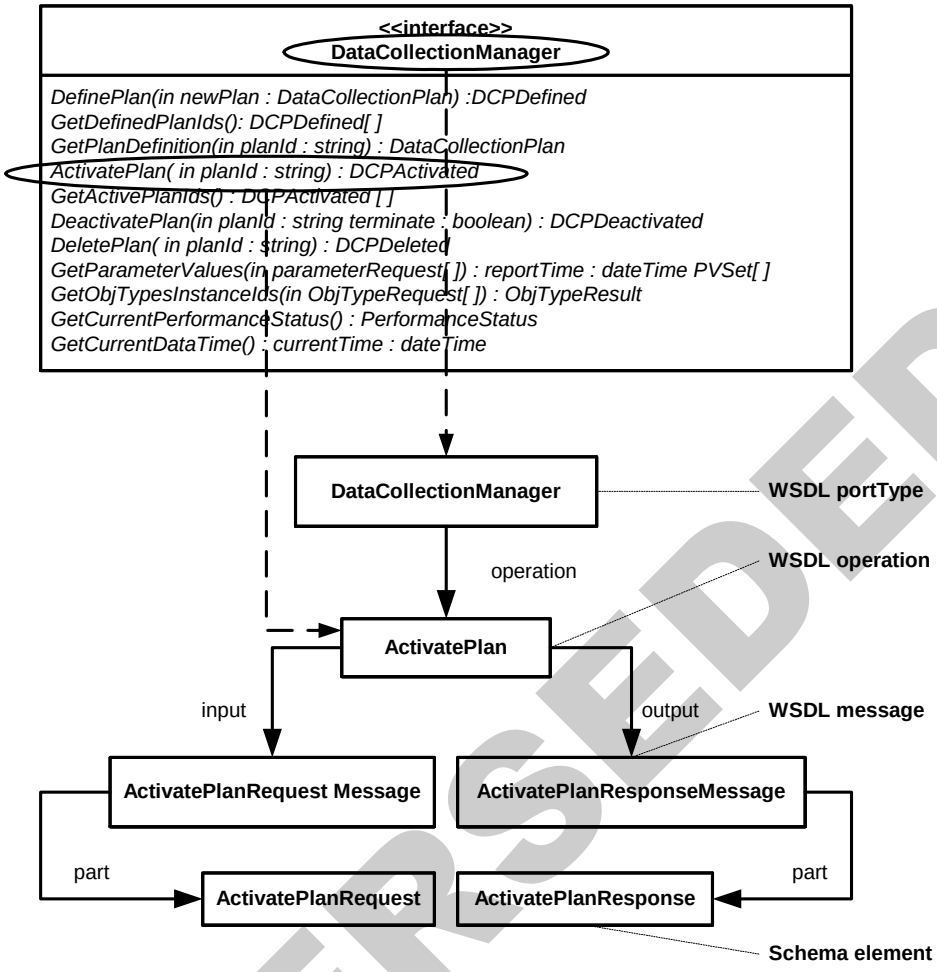

Figure 5
Mapping the DataCollectionManager to a WSDL portType

SUPERSEDED

7.2.9 *WSDL Binding Overview*

7.2.9.1 The WSDL DataCollectionManager binding definition specifies a message and transport protocol to use for a given portType (SOAP and HTTP for SEMI E134.1), the interface style used (document style for SEMI E134.1). 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 ¶¶ 7.2.10–7.2.19.

7.2.9.2 Details for the binding operation for the DataCollectionManager are defined by the wsdl file in Table 15. The SOAP action used in all SEMI E134 operations uses the binding target namespace followed by a colon and the operation name (i.e., soapAction = <binding target namespace>:<operation-name>).

soapAction = "urn:semi-org:ws.E134.V0310.DCMEquipmentBinding:<operation-name>"

7.2.10 *WSDL Service Overview*

7.2.10.1 This specification does not provide a WSDL service definition. WSDL service definitions define the address(es) at which a given interface can be accessed by a client. Such information cannot be determined until the

Documents Document Sharing Hub
Full document details: SEMI E134.1-0310

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 E134 equipment services in the factory.

7.2.11 *DefinePlan*

7.2.11.1 *WSDL Operation Binding*

7.2.11.1.1 See the operation binding for "DefinePlan" in the binding wsdl file. Key information for the binding operation is shown for convenience in the following table.

Table 16 DefinePlan Operation Binding

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

7.2.11.2 *WSDL Operation*

7.2.11.2.1 See the port type operation definition for "DefinePlan" in the port type wsdl file. Key information for the port type operation is shown for convenience in the following table.

Table 17 DefinePlan PortType Operation

Input Message Name	DefinePlanRequest
Output Message Name	DefinePlanResponse

7.2.11.3 *WSDL Message(s)*

7.2.11.3.1 See the message definitions for "SyncHeaderMessage", "DefinePlanRequest", and "DefinePlanResponse" in the corresponding port type wsdl file. Key information is shown for convenience in the following table.

Table 18 DataCollectionManagement Binding

Message Name → Schema Element Name	SyncHeaderMessage → xms:SyncHeader DefinePlanRequest → dcm:DefinePlanRequestType DefinePlanResponse → dcm:DefinePlanResponseType

7.2.11.4 *XML Schema Types*

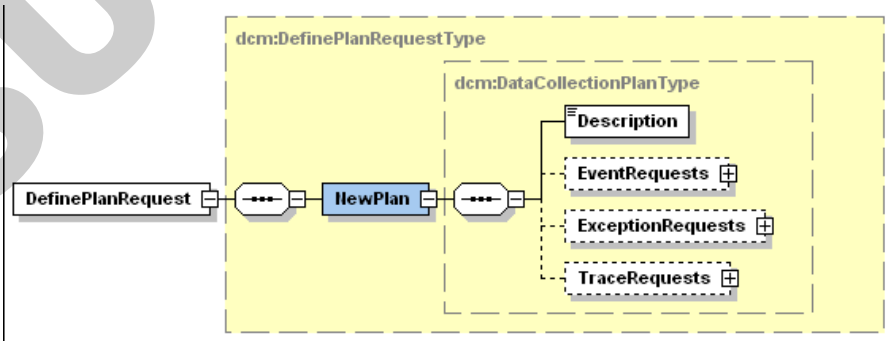

Figure 6
DefinePlanRequest

SUPERSEDED

Documents
Document Sharing Hub
Full document details: SEMI E134.1-0310

7.2.11.4.1 DefinePlanRequest - this global element acts as a top-level container entity representing the SOAP body of a DefinePlan request. As defined in SEMI E134, this operation has one input argument: the new DCP that is being defined. This argument is modeled as a single element named "NewPlan" of complexType "DataCollectionPlanType".

Table 19 Translation Table for Input Arguments for the DefinePlan Operation

Argument Name	*Format*	*XML Element or Attribute*	*XML Name/Type*
newPlan	Structured data, of type DataCollectionPlan	Element	newPlan: dcm:DataCollectionPlanType

7.2.11.4.2 *DataCollectionPlanType* — This global complexType is the XML Schema representation of the SEMI E134 Data Collection Plan class. The following table describes the mapping of the UML class to this XML Schema complexType.

Table 20 Translation Table for DataCollectionPlanType

Attribute or Role Name	*UML Type*	*XML Element or Attribute (or Reference)*	*XML Name/Type*
name	Text	Attribute	name: xsd:string
id	Text	Attribute	id: xsd:string
intervalInMinutes	Positive Integer > = 0	Attribute	intervalInMinutes: xsd:int
isPersistent	Boolean	Attribute	isPersistent: xsd:boolean
description	Text	Element	description: xsd:string
eventRequests	List of structured data, of type EventRequest	Element	eventRequest: dcm:EventRequestType
exceptionRequests	List of structured data of type ExceptionRequest	Element	exceptionRequests: dcm:ExceptionRequestType
traceRequests	List of structured data of type TraceRequest	Element	traceRequests: dcm:TraceRequestType

SUPERSEDED

Documents Document Sharing Hub
Full document details: SEMI E134.1-0310

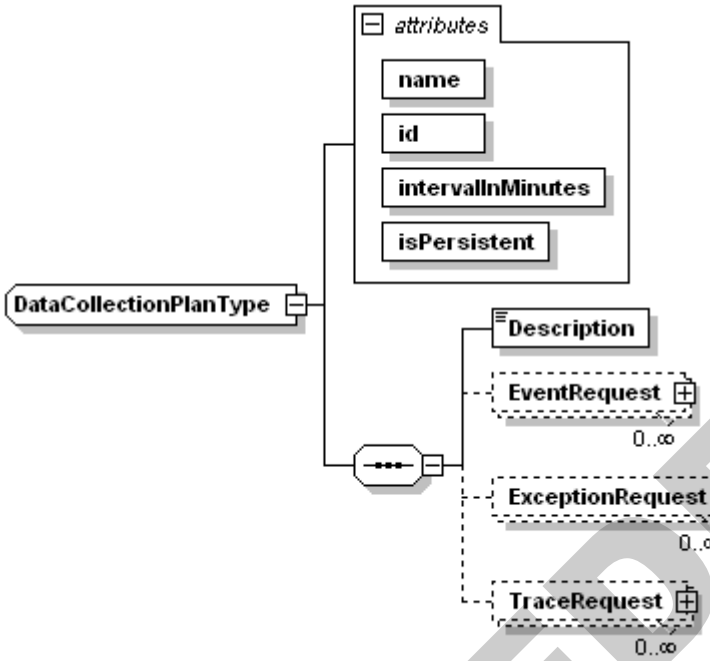

Figure 7
DataCollectionPlanType

7.2.11.4.3 *EventRequestType* — This global complexType is the XML Schema representation of the SEMI E134 Event Request class. The following table describes the mapping of the UML class to this XML Schema complexType.

Table 21 Translation Table for EventRequestType

Attribute or Role Name	*UML Type*	*XML Element or Attribute (or Reference)*	*XML Name/Type*
sourceId	Text	Attribute	sourceId: xsd:string
eventId	Text	Attribute	eventId: xsd:string
parameterRequests	Ordered list of structured data, of type ParameterRequest	Element	parameterRequests: dcm:ParameterRequest

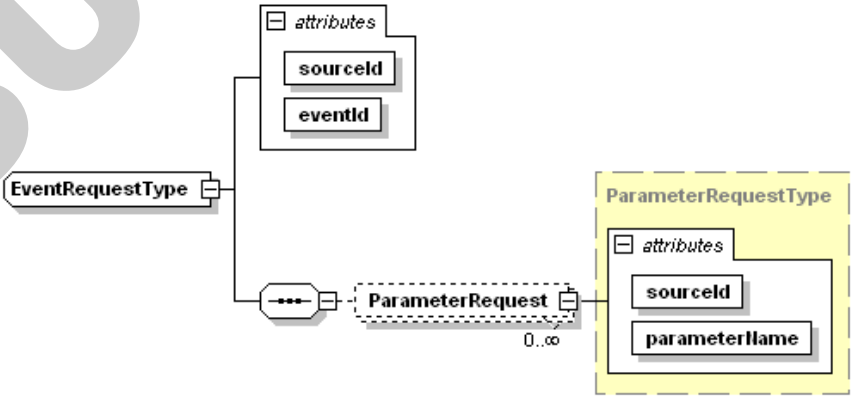

Figure 8
EventRequestType

SUPERSEDED
Documents
Document Sharing Hub
Full document details: SEMI E134.1-0310