

SEMI E133.1-1107

PROVISIONAL SPECIFICATION FOR XML MESSAGING FOR PROCESS CONTROL SYSTEM (PCS)

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 September 5, 2007. Available at www.semi.org in November 2007; originally published March 2006.

Table of Contents

1 Purpose	2
1.1 XML Implementation Mapping.....	2
2 Scope	2
2.1 Specification Scope	2
3 Limitations.....	2
4 Referenced Standards and Documents	2
4.1 SEMI Standards.....	2
4.2 OMG Standards	2
4.3 W3C Standards	2
5 Terminology	2
5.1 Acronyms and Abbreviations	3
5.2 Definitions	3
6 Conventions.....	3
6.1 Documenting XML Schema With Diagrams.....	3
6.2 XML Schema Sample.....	4
7 Implementation Overview	5
7.1 DataCollection.....	5
7.2 Point-to-point and Daisy Chain Communication.....	6
7.3 Extending the XSD.....	6
8 Generic AnalysisEngineDataSet Structure	7
9 PCS Core Elements	8
9.1 PCSJobIDType	8
9.2 PCSAETargetType	9
9.3 DataType	9
9.4 StatusType	9
9.5 StatusElementType.....	10
9.6 AnalysisParametersType	10
9.7 GlobalsType.....	11
9.8 CalculatedOutputType.....	11
9.9 LogMessagesType.....	12
9.10 ModuleDataType.....	12
9.11 ProcessDataType	12
9.12 SubstrateDataType.....	13
10 Request vs. Response AnalysisEngineDataSet Structure	13
APPENDIX 1 – PCS Aggregate XSD	14

1 Purpose

1.1 XML Implementation Mapping

1.1.1 The purpose of this Specification is to provide an implementation mapping of the SEMI E133 Specification for an Automated Process Control System Interface. This Specification does not provide a transport protocol for communication of XML messages between client and server (Analysis Engines).

2 Scope

2.1 Specification Scope

2.1.1 This Specification utilizes XML Schema definitions per W3C recommendations.

2.1.2 This Specification provides XML Schema mapping of SEMI E133 PCS messages.

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 This Document represents an XML aggregation model for PCS messaging. Additional specification will be necessary to represent concepts in SEMI E133 that are not covered by the current model.

3.2 This is a Provisional Specification. In order to have the Provisional status removed, the following items must be completed.

- SEMI E133 must have its Provisional status removed.
- This Specification must provide XML Schema mapping for the specific R2R, SPC, FD, FC, FP functional groups.

4 Referenced Standards and Documents

NOTE 1: Definitions or descriptions of the following terms used in this Specification may be found in the SEMI Compilation of Abbreviations and Acronyms, available on the SEMI Web site, <http://www.semi.org>.

4.1 SEMI Standards

SEMI E133 — Provisional Specification for Automated Process Control Systems Interface

4.2 OMG Standard

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

4.3 W3C Standards

Extensible Markup Language (XML) 1.0 (Second Edition) — W3C, 6 October 2000; <http://www.w3c.org/TR/2000/REC-xml-20001006/>

Namespaces in XML — W3C, 14 January 1999; <http://www.w3c.org/TR/1999/REC-xml-names-19990114/>

XML Schema Part 0: Primer — W3C, 2 May 2001; <http://www.w3c.org/TR/xmlschema-0/>

XML Schema Part 2: Datatypes — W3C, 2 May 2001; <http://www.w3c.org/TR/xmlschema-2/>

XML Schema Part 0: Primer — W3C, 2 May 2001; <http://www.w3c.org/TR/xmlschema-0/>

XML Path Language (XPath) — W3C, 16 November 1999; <http://www.w3c.org/TR/xpath/>

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

5 Terminology

NOTE 2: Terminology is defined as acronyms, abbreviations and term definitions.

5.1 Abbreviations and Acronyms

The following abbreviations and acronyms are unique to this Standard.

5.1.1 UML — Unified Modeling Language

5.1.2 W3C — World Wide Web Consortium

5.1.3 XSD — XML Schema Definition

5.1.4 See also SEMI E133 for additional acronyms.

5.2 Definitions

5.2.1 No additional term definitions are unique to this Standard.

6 Conventions

6.1 Documenting XML Schema with Diagrams

6.1.1 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.

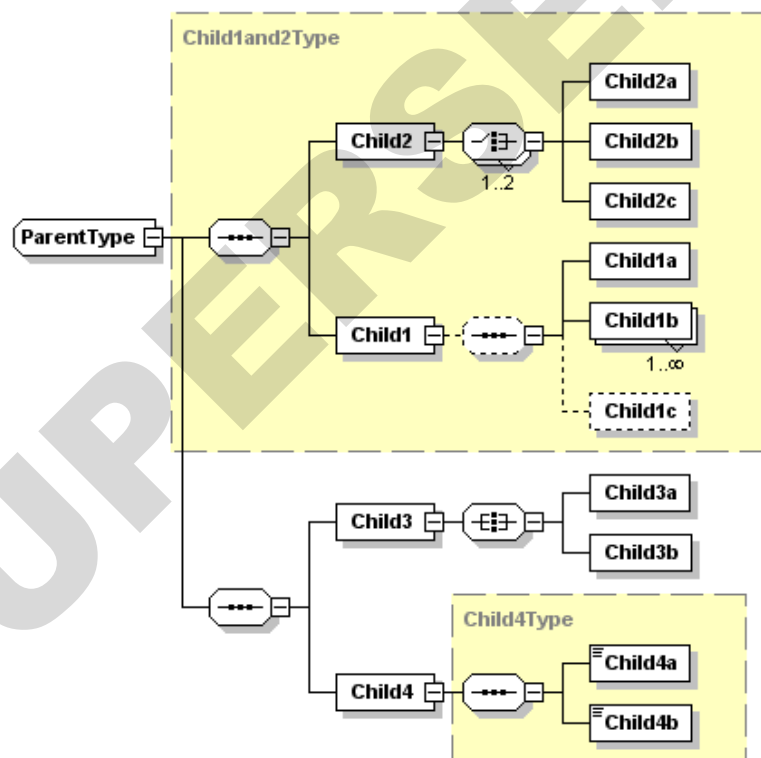
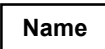


Figure 1
XML Schema Example Diagram

6.1.2 In the diagram, rectangular boxes represent XML element definitions. Ownership or containment is read from right to left 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 1 for an explanation of these symbols.

Table 1 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.
	Denotes an element that contains parsed character data.

6.1.3 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.

6.1.4 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.

6.1.5 The yellow (or gray 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.

6.1.6 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.

6.2 XML Schema Sample

6.2.1 The sample XML Schema for the example shown in Figure 1, is presented below.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xs:element name="parentType">
    <xs:complexType>
      <xs:complexContent>
        <xs:extension base="Child1and2Type">
          <xs:sequence>
            <xs:element name="Child3">
              <xs:complexType>
                <xs:all>
                  <xs:element name="Child3a"/>
                  <xs:element name="Child3b"/>
                </xs:all>
              </xs:complexType>
            </xs:element>
            <xs:element name="Child4" type="Child4Type"/>
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="Child1and2Type">
    <xs:sequence>
      <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="Child1">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="Child1a"/>
            <xs:element name="Child1b" maxOccurs="unbounded"/>
            <xs:element name="Child1c" minOccurs="0"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="Child4Type">
    <xs:sequence>
      <xs:element name="Child4a" type="xs:string"/>
      <xs:element name="Child4b" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

Figure 2
XML for Sample

7 Implementation Overview

7.1 DataCollection

7.1.1 The SEMI E133 Specification suggests that an Element ‘DataCollection’ exists with an attribute of ‘BinCategory’. To allow for XML validating parsers to work with the XSD and instance XML, the DataCollection element is not used. Instead, the name of the BinCategory is used as the element name. For example, we use:

```
<Globals>
  <item1>123</item1>
  <item2>ABC</item2>
</Globals>
```

Instead of:

```
<DataCollection BinCategory="Globals">
  <item1>123</item1>
  <item2>ABC</item2>
</DataCollection>
```

7.1.2 This allows the following to be enforced:

- Order using `xs:sequence` to ensure Globals is before ModuleData, etc.
- Children can be validated for structure and content.

7.2 Point-to-Point and Daisy Chain Communication

7.2.1 Point-to-point communication represents one client communicating to one server. In extending the XSD in this Document, a closed form XSD can be defined for request and response XML instances.

7.2.2 Daisy chain communication is where one client communicates to one server, and that server communicates to an additional server, and so on and so forth. It is left to the field implementation to coordinate the XSDs that are passed from clients and servers and how data is passed through from upstream servers back to the client.

7.3 Extending the XSD

7.3.1 This Document does not provide a complete XSD form for use in a specific application. Instead, 'Data', 'Context' and 'Extension' elements are used as placeholders for additional structure to be implemented in the field.

7.3.2 For example, if 'SomeDataType' within this schema has the following definition:

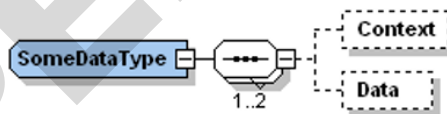


Figure 3
Generic XML Schema Diagram for SomeDataType

7.3.3 The actual implementation of 'SomeDataType' would add elements under Context and/or Data specific to that application such as: